A Novel Transformer-based Multi-step Approach for Predicting Common Vulnerability Severity Score

by

Saeid Bahmanisangesari

Bachelor of Computer Engineering, FUM, 2021

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Computer Science

In the Graduate Academic Unit of Computer Science

Supervisors:	Ali A. Ghorbani, Ph.D., Computer Science
	Haruna Isah, Ph.D., Computer Science
Examining Board:	Roozbeh Razavi-Far, Ph.D., Computer Science, Chair
	Saqib Hakak, Ph.D., Computer Science
	Zhen Lei, Ph.D., Civil Engineering

This thesis is accepted by the
Dean of Graduate Studies

THE UNIVERSITY OF NEW BRUNSWICK

June, 2024

Abstract

The timely prediction of Common Vulnerability Severity Scores (CVSS) following the release of Common Vulnerabilities and Exposures (CVE) announcements is crucial for enhancing cybersecurity responsiveness. A delay in acquiring these scores may make it more difficult to prioritize risks effectively, resulting in the misallocation of resources and a delay in mitigating actions. Long exposure to untreated vulnerabilities also raises the possibility of exploitative attacks, which could lead to serious breaches of security that compromise data integrity and harm users and organizations. This thesis develops a multi-step predictive model that leverages DistilBERT, a distilled version of the BERT architecture, and Artificial Neural Networks (ANNs) to predict CVSS scores prior to their official release. Utilizing a dataset from the National Vulnerability Database (NVD), the research examines the effectiveness of incorporating contextual information from CVE source identifiers and the benefits of incremental learning in improving model accuracy. The models achieved better results compared to the top-performing models among other works with an average accuracy of 91.96% in predicting CVSS category scores and an average F1 score of 91.87%. The results demonstrate the model's capability to predict CVSS scores across multiple categories effectively, thereby potentially reducing the response time to cybersecurity threats.

Dedication

I would like to dedicate this thesis to my parents, Professor Nahid Ashrafi and Dr. Rahman Bahmanisangesari, who have supported me throughout every stage of my life. I also extend my deepest gratitude to my brother, Dr. Sajjad Bahmanisangesari, for his constant encouragement and motivation. Additionally, I am profoundly thankful to my best friend, Masoud Kermani Poor, whose unwavering support and friendship since our undergraduate studies have been invaluable.

Acknowledgements

I would like to express my sincere gratitude to my supervisors, Dr. Ali Ghorbani and Dr. Haruna Isah, for their superior guidance and support throughout my master's thesis journey.

Table of Contents

List of Tables

List of Figures

Abbreviations

AI	Artificial Intelligence
ANN	Artificial Neural Network
API	Application Programming Interface
BERT	Bidirectional Encoder Representations from Transformers
Bi-LSTM	Bidirectional Long Short-Term Memory
CNN	Convolutional Neural Network
CVE	Common Vulnerabilities and Exposures
CVSS	Common Vulnerability Scoring System
DL	Deep Learning
DistilBERT	Distilled BERT
IoT	Internet of Things
JSON	JavaScript Object Notation
LSTM	Long Short-Term Memory
ML	Machine Learning
NLP	Natural Language Processing
NVD	National Vulnerability Database
OSINT	Open Source Intelligence
RNN	Recurrent Neural Network
SGD	Stochastic Gradient Descent
SHAP	SHapley Additive exPlanations
SIG	Special Interest Group
SVM	Support Vector Machine

Chapter 1

Introduction

1.1 Overview

The Common Vulnerabilities and Exposures (CVE) system serves as a standardized catalog of publicly known cybersecurity vulnerabilities and exposures, providing a crucial reference for the information security community. This system facilitates a unified approach to identifying, discussing, and managing vulnerabilities, which is essential for maintaining consistent communication within the security industry and among researchers [152]. The Common Vulnerability Scoring System (CVSS) complements the CVE by offering a framework to evaluate the severity of software vulnerabilities. This scoring system is an open and standardized method that helps to prioritize vulnerabilities based on their potential impact, using a metric that combines likelihood and impact to assess risk [8, 9, 49].

Despite the effectiveness of the CVE in cataloging vulnerabilities, there often exists a significant delay in the subsequent severity assessment through CVSS. This delay in scoring can leave systems vulnerable for extended periods, highlighting a critical need for more immediate response mechanisms [32, 118].To bridge this gap, this thesis proposes a novel approach that utilizes advanced machine learning techniques to predict CVSS scores promptly and accurately. By leveraging DistilBERT—a streamlined version of the powerful BERT transformer model—the proposed method processes and extracts salient features from the textual descriptions of vulnerabilities within CVE entries. These features are then used in conjunction with Artificial Neural Networks (ANNs) to estimate CVSS scores, thereby aiming to expedite the response to newly disclosed vulnerabilities.

This research adopts a multi-step, incremental learning strategy that not only adjusts to the initial dataset but also continuously adapts to new data. This adaptive approach is crucial in the cybersecurity field, where the nature and tactics of threats are constantly evolving. By ensuring that the predictive model remains effective and up-to-date, this methodology significantly enhances the readiness and response times of cybersecurity measures, contributing to more secure systems amidst an increasingly complex threat landscape. The CVEs are crucial in the field of cybersecurity as they provide a common reference point for the information security community for identifying, discussing, and managing vulnerabilities. The CVE list is used to provide a standard way of identifying vulnerabilities and exposures, and it is widely used by the security industry and researchers to ensure that there is a common understanding of the vulnerabilities that are being referred to [152].

Moreover, the proactive prediction of CVSS scores enables organizations to anticipate potential risks and vulnerabilities before official scores are available. This foresight empowers them to implement timely and targeted mitigation strategies, reducing the window of vulnerability and minimizing the likelihood of exploitation by malicious actors. Furthermore, accurate CVSS score predictions contribute to the efficiency of incident response and vulnerability management processes. By having early insights into the severity of a vulnerability, organizations can streamline their decision-making processes, allocate resources efficiently, and implement appropriate security measures in a timely manner.

1.2 Research Problem

Several studies (i.e.,[131, 35, 130, 46, 84]) have explored various methodologies for predicting CVSS. While these models achieve satisfactory levels of accuracy and F1 scores, they often lack mechanisms for ongoing model maintenance and adaptation over time. This thesis advances the field by introducing a novel approach that not only enhances accuracy and F1 scores but also incorporates source identifiers an element overlooked in previous research. Additionally, this work leverages an incremental learning technique, enabling the model to adapt continuously to new data. This capability ensures sustained accuracy.

1.3 Summary of Contributions

The following are the main contributions of this thesis:

- A comprehensive review of Vulnerability Management.
- A precise framework for classifying Common Vulnerability Scoring System (CVSS) metrics.
- Incorporating source identifiers into the model significantly improves prediction outcomes for CVSS scores.
- A novel method for applying incremental learning to the DistilBERT transformer, enabling continuous model improvement and adaptation to new data.
- Demonstration of the impact of advanced text preprocessing on improving model outcomes.
- Demonstration of the model interpretability by employing SHAP for Artificial Neural Networks and LIME for transformers.

1.4 Thesis Organization

The rest of this thesis is organized as follows:

Chapter 2 provides an extensive review of the literature concerning vulnerability management and CVE. This chapter begins by defining and discussing the importance of vulnerability management, followed by an exploration of the roles and challenges associated with CVE in cybersecurity. It also covers the evolution and structure of CVEs, along with their applications across different domains.

Chapter 3 presents the methodology of the predictive model using DistilBERT and ANN for predicting CVSS scores. This includes the data collection from the NVD, data preprocessing, and the specifics of the incremental learning approach employed. Additionally, the integration of source identifiers and the methodologies for enhancing model interpretability using SHAP and LIME are discussed.

Chapter 4 describes the experimental setup and results. It details the performance of the proposed model, comparisons with other works, and the effectiveness of the incremental learning enhancements. The impact of advanced preprocessing techniques on the model's accuracy is also evaluated.

Chapter 5 concludes the thesis by summarizing the key findings and contributions. It discusses the implications of this research for cybersecurity practices and proposes directions for future research to build on the advancements made in this thesis.

Chapter 2

Background and Literature Review

2.1 Introduction to Vulnerability Management

2.1.1 Definition and Importance

Vulnerability management is a fundamental component of cybersecurity, dedicated to systematically identifying, classifying, prioritizing, and remediating software vulnerabilities to mitigate the risks they pose. In the ever-evolving landscape of cybersecurity, where new vulnerabilities continually emerge, the importance of an effective vulnerability management process is paramount. These vulnerabilities, essentially flaws or weaknesses within software, pose significant risks to information system security. Without timely intervention, they threaten to undermine the integrity, confidentiality, and availability of data [159].

The vulnerability management process begins with the detection of vulnerabilities, utilizing a combination of automated technologies and manual testing to ensure thorough coverage. Upon discovery, the subsequent evaluation phase assesses the severity of these vulnerabilities, often utilizing frameworks like the Common Vulnerability Scoring System (CVSS). As illustrated in Figure 2.1, this system assigns a numerical severity score, providing a standardized measure for the potential impact of vulnerabilities and enabling a prioritized remediation strategy based on score severity [130, 131].

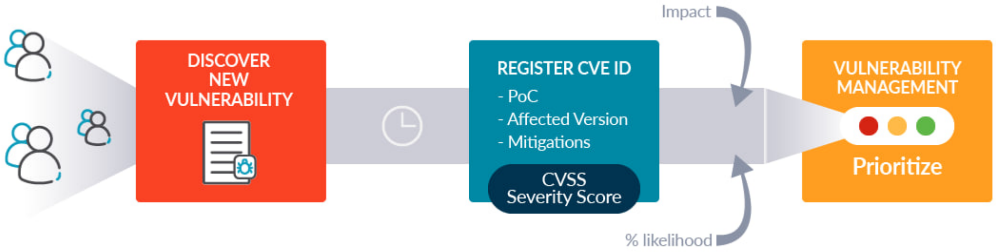


Figure 2.1: The lifecycle of a vulnerability [62]

Extending beyond the initial identification and evaluation stages, vulnerability management in cybersecurity covers the critical tasks of assessing, prioritizing, and mitigating vulnerabilities mostly across software and hardware landscapes. This procedure is essential to preventing future cyberattacks and protecting private data. In the face of growing cyber threats, a thorough cybersecurity vulnerability management approach that includes software validation and risk analysis is essential for proactively fixing vulnerabilities and strengthening the security of an organization [120].

Furthermore, the deployment of standardized vulnerability assessment tools and services emerges as a vital strategy in navigating the increasing cyber risks [90]. These tools bolster the capability to discern and prioritize vulnerabilities, directing organizational efforts towards mitigating the most pressing security concerns. By embracing proactive and systematic approaches to vulnerability management, organizations can significantly enhance their resilience against cyber intrusions and protect their critical data assets [90]. The complex process of managing vulnerabilities, which is highlighted by the unpredictable and severe nature of cyber attacks, presents numerous obstacles for decision-makers when allocating resources to cybersecurity efforts [72].

Taking cybersecurity strategies to the next level requires more than just meeting compliance standards. It involves gaining real dedication from the organization and getting employees on board with cybersecurity initiatives. In this context, training plays a vital role in improving employees' understanding of cybersecurity, influencing how they perceive vulnerability and threat severity, and encouraging secure behaviors [36, 61]. In addition, by employing methodological approaches that incorporate human factors, organizations can better evaluate their level of preparation in dealing with cyber threats [111].

Artificial intelligence (AI) in cybersecurity has transformed the field by offering advanced capabilities in asset prioritization, control allocation, vulnerability management, and threat detection with unbeatable efficiency. AI technologies are essential for protecting Internet-connected systems from cyber threats, attacks, damage, or unauthorized access[125]. Integrating artificial intelligence (AI) with cybersecurity signifies a fundamental change in protecting digital assets from an ever-changing threat environment, therefore playing a crucial role in fighting cybercrime and intrusion detection [135, 82]. Open-source vulnerability repositories, such as the CVE and the National Vulnerability Database (NVD), help prioritize vulnerabilities and improve vulnerability management by enabling quantitative prioritization and predictive analyses [121, 75].

Vulnerability management goes beyond basic risk assessment and promotes the integration of comprehensive cybersecurity risk management into enterprise risk frameworks. It involves the use of models such as the CERT Resilience Management Model (CERT-RMM) to ensure a strong and effective approach [112, 113]. Having a proactive and flexible approach to cybersecurity from a service viewpoint enables firms to navigate the ever-changing landscape of cyber threats effectively [143].

To effectively address vulnerabilities particular to various systems, such as telemedicine, in-vehicle networks, and smart charge management systems, it is necessary to implement a complete cybersecurity plan. This approach is strengthened by the exchange of vulnerability information between companies and the utilization of social media to gather intelligence on threats, which together enhance preparedness for cybersecurity. This is supported by various studies [155, 17, 104, 11]. Furthermore, it is crucial to promote cybersecurity awareness among all individuals, including students and employees, in order to strengthen the entire defense against cyber threats [28, 22].

Vulnerability assessment and mitigation in cybersecurity present complex challenges that require a multidimensional approach to address. Several studies highlight the critical aspects of understanding and combating vulnerabilities in various sectors. The systematic review by [109] delves into attacks, vulnerabilities, and defense strategies in Industry 4.0, emphasizing the need to classify these aspects and address emerging challenges in cybersecurity. This review underscores the importance of staying up-to-date of evolving threats and defense mechanisms.

In the healthcare sector, [7] conducted a systematic review focusing on cybersecurity challenges specific to healthcare settings. Their work highlights the significance of identifying threats and vulnerabilities within the healthcare industry to enhance cybersecurity measures effectively. Additionally, [146] emphasize the adverse impacts of cybersecurity vulnerabilities in networked medical devices, stressing the importance of robust risk assessment frameworks to safeguard sensitive patient information and clinical care integrity.

Furthermore, the study by [100] provides insights into cybercrime and cybersecurity risks, emphasizing the need for practitioners and academics to mitigate the consequences of cyber threats effectively. This underscores the importance of proactive measures to address vulnerabilities and enhance cybersecurity resilience across various domains.

2.1.2 Challenges

Addressing vulnerabilities in critical infrastructure, such as transportation systems, [83] highlight challenges in cybersecurity within the railway sector, including legal gaps, low awareness, and limited budgets. Understanding these challenges is crucial for implementing effective mitigation strategies and enhancing the overall cybersecurity posture of transportation networks.

Moreover, the study by [68] illuminates on cybersecurity challenges in intersection management, emphasizing the need for robust cybersecurity measures in intelligent transportation systems. By addressing weaknesses in transportation management systems, efforts can be directed towards mitigating cybersecurity risks and ensuring the secure operation of critical infrastructure.

In the context of emerging technologies like blockchain, [98] discuss the role of cybersecurity certification in mitigating risks associated with Information and Communication Technologies (ICT). This highlights the importance of standardized approaches to enhance cybersecurity readiness and resilience in the face of evolving threats.

One primary challenge in vulnerability assessment is the delay in the manual classification and scoring of vulnerabilities, such as those listed in the CVE system. The manual determination of the Common Vulnerability Scoring System (CVSS) metrics by human experts, based on the textual descriptions of vulnerabilities, is not only time-consuming but also introduces delays in remediating vulnerabilities, thereby increasing security risks [131]. This process, essential for assessing the risk levels of vulnerabilities and prioritizing remediation actions, becomes a bottleneck due to the increasing number of daily discovered exploits and the complexities involved in understanding the textual descriptions of vulnerabilities [14]. Such delays are problematic, given that on average, it can take over 132 days after a vulnerability's publication for it to receive a full CVSS severity assessment [32].

Moreover, the current approaches to vulnerability scoring have faced criticism for their heavy reliance on manual, arithmetic-based assessments, which can further delay the classification and mitigation processes. This delay is especially concerning for electric utilities and similar entities that rely heavily on CVSS base scores to assess risk levels and prioritize vulnerabilities [130]. In response to these challenges, recent advancements have sought to leverage Natural Language Processing (NLP) and machine learning techniques to predict CVSS base scores based solely on CVE descriptions. This approach aims to reduce the time gap between the discovery of a vulnerability and its severity classification, thus allowing for quicker and more effective mitigation strategies [131, 35, 130, 46, 84].

2.1.3 Role of CVE in Vulnerability Management

Vulnerability management is an essential component of cybersecurity, and the CVE system plays a vital role in this field. CVE provides structured and quantified severity information for identified vulnerabilities, which is then stored in vulnerability databases [118]. The main goal of CVE is to simplify the sharing of vulnerability data among different tools, repositories, and services using a standardized enumeration system [99]. To support various forms of data-driven software security research, like automated vulnerability fix and vulnerability prediction, CVEfixes, for instance, automates the collection of vulnerabilities and their fixes from open-source software [16].

In response to the increasing number of vulnerabilities, security experts use critical components of CVE descriptions for vulnerability understanding, management, and mitigation [59]. CVE coordination highlights the significance of CVE in the cybersecurity environment by exhibiting common characteristics of software engineering coordination [119]. As they provide up-to-date information about recently identified vulnerabilities, CVEs are in fact regarded as essential cybersecurity resources [54].

One of the most extensive publicly accessible sources of information about software and hardware vulnerabilities is the CVE database [19].

Vulnerability databases such as CVE and the National Vulnerability Database (NVD) play a crucial role in Cyber Threat Intelligence and are extensively utilized in various security products worldwide [149]. The purpose of these databases is to provide project maintainers with a convenient way to identify security-related patches without having to do it manually [88]. Software engineers depend on CVE reports to mitigate vulnerability exploits and secure vulnerable systems and libraries [56]. The CVE system, along with other vulnerability databases, facilitates the use of quantitative methods to gain prioritization insights and conduct predictive analysis in modern security practices [75].

The role of CVE in vulnerability management extends beyond simply identifying and categorizing vulnerabilities. It plays a crucial role in a larger system that involves automated evaluation of the severity of vulnerabilities, ranking them in order of importance, and taking steps to reduce their impact. From [130], the integration of CVE with automated tools for vulnerability severity prediction illustrates the evolving landscape of cybersecurity. This integration not only speeds up the evaluation process but also enhances our comprehension of vulnerabilities, leading to more informed decision-making in cybersecurity operations.

The CVE system is crucial in vulnerability management as it offers a standardized method for identifying, documenting, and sharing vulnerabilities. By utilizing sophisticated computational techniques and machine learning models such as [131, 35, 130, 46, 84], CVE allows cybersecurity experts to stay informed on the most recent vulnerabilities, prioritize their actions, and effectively secure systems. The finding underlines the changing nature of cybersecurity, where the combination of established vulnerability identification tools and advanced AI approaches enables a more proactive and efficient approach to handling cyber attacks.

2.2 Overview of CVEs

2.2.1 History and Evolution of CVE

The establishment of CVE by MITRE in 1999 created a fundamental standard for identifying and categorizing software and hardware systems vulnerabilities. This project played a crucial role in establishing a common language for addressing and mitigating security vulnerabilities across different platforms and technologies, hence improving the communication and handling of cybersecurity threats [129].

The CVE system has experienced a significant rise in reported vulnerabilities over time, with the number increasing from 894 in 1999 to 18,325 in 2020. This increase represents the increasing complexity and awareness of cybersecurity concerns [138]. This increase highlights the need for ongoing improvements in vulnerability management strategies and the significance of CVEs in the wider context of information security.

The adoption of automated systems, specifically for the detection of cybersecurity attacks in IoT networks, highlights the changing nature of CVE applications. Named Entity Recognition (NER) systems have been created to precisely detect IoT-related CVEs, underlining the importance of adapting CVE management practices to accommodate the expanding landscape of technology [54].

Furthermore, the significance of CVE databases, like the National Vulnerability Database (NVD), is underscored by their role in providing comprehensive information on vulnerabilities. The need for high-coverage approaches in constructing these databases reflects the challenge of dealing with the diverse presentation of vulnerability information across different sources. This necessity highlights the constant effort needed to maintain and enhance the quality of CVE databases so that they can effectively address cybersecurity threats [66, 27].

Efforts to improve the classification and identification of vulnerabilities through frameworks such as V2W-BERT demonstrate the continuous progress in technology aimed at enhancing vulnerability management procedures. These frameworks enhance the precise classification and identification of vulnerabilities, enabling more effective management of cybersecurity risks [42].

Despite these advancements, challenges persist in the maintenance and quality of CVE databases. The current state of these databases often lacks the desired level of effectiveness and accuracy, highlighting the need for continuous improvements [27].

The historical development and progression of CVEs reflect the ever-changing nature of the cybersecurity environment. Since their creation, CVEs have been essential in cybersecurity, playing a crucial role in the communication and management of security vulnerabilities. The continuous endeavors to improve the effectiveness, accuracy, and comprehensiveness of CVE management strategies demonstrate the important role of CVEs in managing cybersecurity risks associated with vulnerabilities in information systems.

2.2.2 Structure and Components of CVE

CVEs serve as the standard for identifying vulnerabilities in cybersecurity. As illustrated in Fig 2.2, these identifiers, typically starting with the prefix ”CVE” followed by a unique number, are integral for referencing and sharing information about vulnerabilities across systems. The structured naming convention facilitates the organization and categorization of information related to information technology systems, software, and packages, enhancing cybersecurity efforts [118, 119, 122]

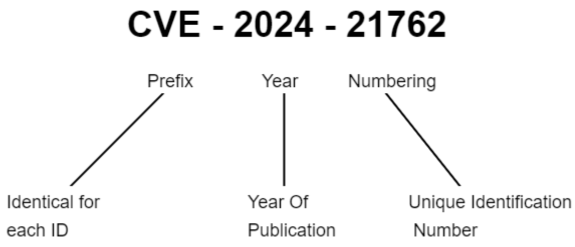


Figure 2.2: CVE Structure

CVE Numbering Authorities (CNAs) play an important part in this system by assigning CVE identifiers to newly reported vulnerabilities. These authorized entities guarantee the accuracy and uniqueness of CVE IDs, supporting a common method for tracking and identifying vulnerabilities across several platforms. Contributions from CNAs are essential to maintaining a solid and effective vulnerability management system, which improves the cybersecurity state of impacted systems and products [37].

2.2.3 Use Cases and Applications of CVE

CVEs let security specialists to share information and collaborate more effectively, improving the field's ability to manage and mitigate cybersecurity risks [6]. The National Vulnerability Database (NVD) lists each CVE entry, which includes an identification number, a description, and reference links for vulnerabilities. This makes the NVD an essential tool for tracking and addressing vulnerabilities. [145].

Beyond standard software vulnerabilities, CVEs are applicable in several domains, including network security, medical devices, and blockchain technologies. Through the use of frameworks such as STRIDE [127], CVEs aid in the risk assessment of networked devices and play a crucial role in identifying vulnerabilities in blockchain systems [157], helping to create more secure digital environments.

But there is still work to be done in terms of assuring data consistency across many vulnerability repositories and mapping vulnerabilities to open-source packages

[40, 75]. The significance of thorough security education that includes knowledge of CVEs and vulnerability databases has been highlighted by the academic and research groups [47]. Furthermore, efforts to automate the setup of environments that replicate vulnerable states for cybersecurity education objectives demonstrate how the application of CVE data is changing [24].

A wide range of topics, including automated vulnerability detection, visual vulnerability analysis, and delay in vulnerability scoring, have been covered in research on CVEs [118, 78, 12]. The vulnerability of a variety of software systems, such as Android apps and IoT networks, which frequently experience inadequate patching and updating and are therefore vulnerable to CVEs, has been highlighted by this work [54, 30]. The Common Vulnerability Scoring System (CVSS) is often used with CVEs to rate the severity of vulnerabilities, showing how important CVEs are to a comprehensive security assessment framework [118].

One noteworthy use of CVE data is the automation of vulnerability classification and severity prediction using natural language processing and machine learning techniques. Utilizing a Multi-Task Learning architecture with a pre-trained BERT model, these studies [35, 130, 46, 84, 14] offer a model that predicts the CVSSv3 severity score and other metrics of a vulnerability based on its CVE text description.

The study [131] presents a novel multi-task deep learning approach, combining DistilBERT and BiLSTM, to enhance the prediction of CVSS. This method leverages semantic features from vulnerability descriptions, achieving approximately 30% higher accuracy than traditional models.

To improve the accuracy and efficiency of CVE assessments, a new method for predicting CVSS metrics based on vulnerability descriptions is presented in the study [35]. By addressing the gap between a vulnerability's first report and its CVSS assessment, this strategy helps cybersecurity experts make decisions more quickly and efficiently.

In the same way, the investigation reported in [130] makes use of the cutting-edge NLP framework, the BERT model, to analyze the textual descriptions of vulnerabilities. This work improves the explainability of the severity predictions in addition to increasing their accuracy, which helps to make the method of assessment clearer and easier to use.

At the time of vulnerability disclosure, the automation of CVSS vector prediction is the subject of another important study [46]. The significance of promptly addressing N-Day vulnerabilities—which have already been made public but do not yet have a patch—is highlighted by this study. The work helps to more effectively mitigate the risks associated with these vulnerabilities by automating the prediction of CVSS vectors.

Finally, the study presented in [84] illustrates how predictive models and open-source intelligence (OSINT) can be combined to determine the CVSS ratings. This method not only expands the dataset used for prediction, but it also improves contextual knowledge of vulnerabilities, increasing the reliability and accuracy of the predictions.

Table 2.1: Applications Of CVEs

Area/Application	**Reference**
Vulnerability Tracking	[145]
Network Security	[127]
Medical Devices	[127]
Blockchain Technologies	[157]
Open-source Software Mapping	[40, 75]
Cybersecurity Education	[47, 24]
Automated Vulnerability Detection	[118, 78]
Visual Vulnerability Analysis	[12]
Software System Vulnerabilities	[54, 30]
Automation in CVSS Prediction	[35, 130, 46, 84, 14]
Combining OSINT for CVSS Ratings	[84]

2.3 Vulnerability Scoring Models

2.3.1 Traditional Scoring Models

The Common Weakness Scoring System (CWSS) is a cybersecurity framework created by The MITRE Corporation to assess and prioritize software vulnerabilities. It functions with the Common Vulnerability Scoring System (CVSS) to provide a standardized method for evaluating weaknesses and vulnerabilities [2].

The Exploit Prediction Scoring System (EPSS) is a system designed to predict the development of functional vulnerability exploits [136]. The Exploit Prediction Scoring System (EPSS) is supported by a Special Interest Group (SIG) with over 200 members worldwide, including practitioners, academics, government agencies, and software developers [3]. EPSS presents a set of 53 traits that professionals have carefully chosen as highly reliable indications of potential exploitation in real-world scenarios. This system is vital in computer security and in predicting exploits [136]. The National Institute of Standards and Technology (NIST) uses the CVSS which is depicted in Fig 2.3 to evaluate the severity of vulnerabilities [142]. CVSS has come to be a widely accepted standard to evaluate the severity of vulnerabilities in many industries. This highlights the significance of standardized systems such as CVSS in cybersecurity.

The Common Vulnerability Scoring System (CVSS) sets itself apart from other cybersecurity scoring systems by providing the ability to create a baseline score for a variety of objects and to generate a customizable score [94]. A key component of cybersecurity is the Common Vulnerability Scoring System (CVSS), which is used to evaluate and determine the severity of vulnerabilities in computer systems. [93] introduced CVSS, enabling for a systematic assessment of vulnerabilities by combining elements like exploitability, impact, and complexity to determine a total severity level. This standardized approach is important for vulnerability management and analysis because it helps organizations prioritize response actions efficiently.

Subsequent studies have highlighted the function of CVSS in establishing baseline scores for different items, hence expanding its value even further. Its customizable nature renders it a preferred system among available scoring models [65, 18]. CVSS's integration into vulnerability management models assists security analysts by providing calculated results in a standardized format, thereby enabling a clearer understanding of detected vulnerabilities' criticality [150].

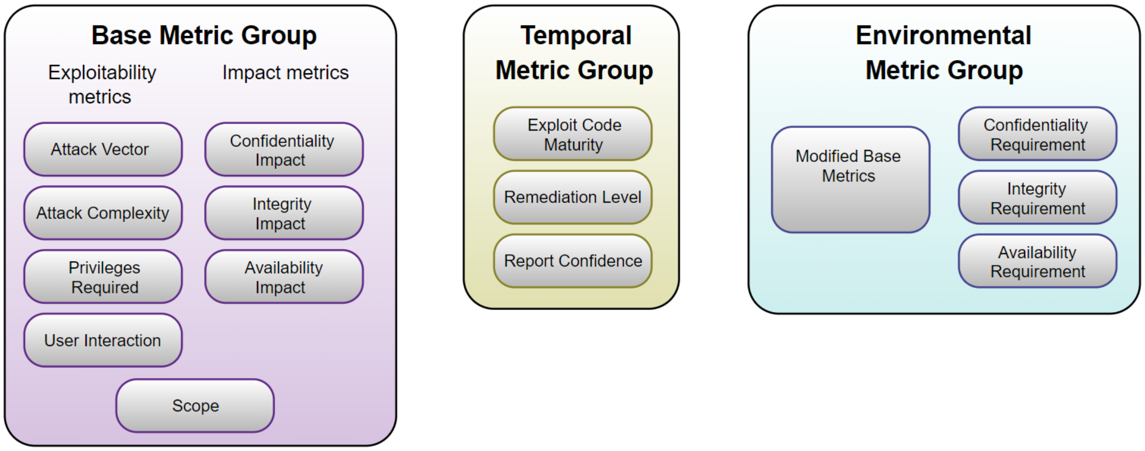


Figure 2.3: CVSS v3.1 Metric Groups [1]

2.3.2 Limitations of Traditional Scoring Models

Traditional models such as the CVSS offer a baseline for scoring and modification, but they are not effective in properly predicting the frequency of cyber-attacks and capturing the complete range of cybersecurity risks [18, 74]. These limitations are highlighted by the requirement for enhancements in stochastic attack-defense models and a full assessment of vulnerabilities, which CVSS aims to tackle by utilizing statistics from its database [154]. The Common Vulnerability Scoring System (CVSS) complements CVEs by offering a method to analyze and score vulnerabilities in systems. In vulnerability management, delays in assigning CVSS ratings to CVEs have a significant impact on the timeliness and efficacy of vulnerability remediation [118]. In addition to identifying vulnerabilities, the CVE and CVSS systems work to assess the severity and potential impact.

Additionally, give an overview of the CVSS's methodology and structure [93]. Research on CVSS has revealed a number of basic shortcomings, including its static scoring system and incapacity to accurately assess the complexity of cyber threats. These problems could make it difficult for the CVSS to accurately represent the dynamic and complex character of modern cybersecurity threats, which could result in errors in vulnerability prioritization and management [65].

The effectiveness of cybersecurity investments, especially in private sector companies, is also influenced by several aspects, such as the value of information, vulnerability probability, and the productivity of cybersecurity efforts. The complex nature of the situation highlights the drawbacks of exclusively depending on traditional scoring models such as the CVSS for resource allocation [57]. It is recommended that large companies take a critical view of CVSS and explore more comprehensive approaches for managing vulnerabilities, focusing on the limits of the system [60].

2.3.3 Recent Advances in Vulnerability Scoring

The creation and deployment of the Common Vulnerability Scoring System (CVSS) have greatly influenced the recent improvements in vulnerability scoring models. The CVSS, which was first introduced in 2006 [93] as an open framework for evaluating and measuring the impacts of software vulnerabilities, has gone through multiple modifications. It has evolved from version 1.0, which was released in 2005 [87] , to the widely adopted version 3.0. The most recent update of the framework included three categories of metrics: base, temporal, and environmental metrics. This update represents a substantial improvement in the standardization and precision of vulnerability scoring in the CVSS framework [101, 85]. The evolution from CVSS version 1 to version 3 underscores continuous improvement and standardization of vulnerability scoring systems, enhancing cybersecurity practices.

The CVSS has been instrumental in providing a standardized framework for assessing and quantifying the impact of vulnerabilities, as highlighted in multiple studies [58, 132]. Efforts to refine the CVSS for better vulnerability scoring have been ongoing, with significant contributions such as Mell et al. [92], emphasizing the system's crucial role in risk management, particularly in specialized applications like metering [58].

Further research has focused on enhancing the CVSS and proposing new methods for vulnerability scoring. Researchers have suggested the implementation of improved systems, such as the Improved Vulnerability Scoring System with ”Vulnerability Type” (IVSV) [31], as well as frameworks like the Weighted Impact Vulnerability Scoring System (WIVSS) [144].

Table 2.2: Summary of Reviewed Papers Utilizing CVE Data To Predict.

Authors	Objective	Method	Data Used	ML/Deep Learning models used	Key Findings
Shan et al. (2023) [131]	Automatically determine CVSS vector using DistilBERT and BiLSTM	Multi-task deep learning approach integrating DistilBERT and BiLSTM	NVD	DistilBERT and BiLSTM	The DistilBERT+BiLSTM model outperformed traditional models by approximately 30%.
Masaki Aota et al. (2020) [13]	Automate classifying vulnerabilities by type, enhancing automation.	Random Forests, Boruta for CWE-ID classification.	NVD	Random Forests with Boruta, compared various models.	96.92% accuracy classifying vulnerabilities into CWE-IDs.
Mustafizur et al.(2021) [130]	Automatically determine CVSS vector using BERT	BERT, a gradient base for explainability.	NVD	BERT-small	High accuracy and explainability with BERT.
Clément Elbaz et al. (2020) [46]	Automatically determine CVSS vector	linear regression, bag-of-words	NVD	Linear regression and bag-of-words.	Predict CVSS vectors with reasonable accuracy
Ahmet Okutan et al. (2022) [102]	Automatically determine CVE severity, exploitability.	CNNs to predict CVE severity, exploitability.	NVD, exploit-db.com, and cvedetails.com..	CNNs with Word Embeddings	F-Measure values above 0.8.
Joana Cabral Costa et al.(2022) [35]	Automatically determine CVSS vector	Uses DistilBERT, explores preprocessing, vocabulary	NVD	DistilBERT, BERT, RoBERTa, ALBERT, and DeBERTa.	DistilBERT outperforms.
Ion Babalau et al. (2021) [14]	Automatically determine CVSS vector	Multi-Task Learning enhances BERT	NVD	Pre-trained BERT, BiLSTM, smallBERT, Multi-Task Learning.	accuracy of 73.76%
Shaofeng Kai et al. (2023) [79]	Automating CPS	Data augmentation, DistilBERT, linear classification.	NVD	DistilBERT with LSTM, CNN, and linear layers.	96.62% accuracy.

2.4 Concluding Remarks

This chapter begins with an in-depth examination of vulnerability management, emphasizing its definition and importance. This foundational discussion sets the stage for a deeper exploration into the role of CVE in cybersecurity, where the use cases and applications of CVEs across various domains are detailed. Following this, the chapter provides an extensive overview of CVEs, including their history, structure, and the pivotal role they play in the cybersecurity landscape. It culminates with a discussion on the broad applicability of CVEs, highlighting their significance in enhancing security measures across different technological fields.

The chapter then transitions into a detailed discussion on "Vulnerability Scoring Models." It starts with traditional approaches, including the Common Vulnerability Scoring System (CVSS), the Common Weakness Scoring System (CWSS), and Exploit Prediction Scoring System (EPSS). This sets the groundwork for introducing recent AI-based approaches to vulnerability scoring. These modern methodologies utilize advanced technologies like DistilBERT and Artificial Neural Networks (ANNs) to predict CVSS scores, thereby enhancing the responsiveness of cybersecurity measures before the publication of official scores.

The concluding remarks highlight the evolving nature of threat assessment and the necessity for models that can adapt to the complexities of modern cyber threats. The integration of AI and machine learning not only improves the accuracy of predictions but also reduces the response time to new threats, which is crucial for maintaining robust cybersecurity defenses. Furthermore, the chapter underscores the importance of adopting a proactive strategy in vulnerability management. Using predictive models allows organizations to anticipate and mitigate potential risks before they materialize, thus enhancing their defensive strategies against more sophisticated cyber attacks.

As the thesis transitions to the methodologies outlined in the next chapter, it becomes evident that the theoretical insights gained in this chapter form a solid foundation for the practical applications that follow. Predictive models are poised to play a vital role in the future of cybersecurity, as they continue to be developed and refined. These models will push the boundaries of what can be achieved in vulnerability management, ensuring that cybersecurity measures remain effective in an ever-changing threat landscape.

Chapter 3

Methodology

3.1 Overview

The research emphasizes the delays between when CVEs are published in the National Vulnerability Database (NVD) and when CVSS data related to these CVEs becomes available [118]. This delay underscores the importance of efficient and accurate prediction mechanisms to bridge this gap and ensure timely assessment and mitigation of vulnerabilities [118]. To improve the efficiency and speed of vulnerability management processes, automated prediction models have been developed to predict CVSS vectors at disclosure [131, 35, 130, 46, 84].

In this section, insights acquired from prior research is leveraged to delineate the proposed methodology for efficiently predicting the Common Vulnerability Scoring System (CVSS) scores. This approach not only aims at enhancing the accuracy and efficiency of predictions but also underscores the development of a more robust model through the application of an incremental technique. By integrating these advanced methodologies, this research seeks to significantly improve the reliability and precision of CVSS score predictions, thereby contributing to the advancement of cybersecurity measures.

There are various reasons why it is important to predict the Common Vulnerability Scoring System (CVSS) for Common Vulnerabilities and Exposures (CVEs). According to [46], the CVSS is regarded as the industry standard for characterizing software vulnerabilities and measuring their level of severity. It is essential to prioritize vulnerabilities based on scores, which aid in determining the possible risks associated with these vulnerabilities [35]. Furthermore, the CVSS predicts different vulnerability attributes, offering a structured approach to understanding and addressing vulnerabilities [59].

3.2 Proposed Approach

The proposed methodology is structured into two distinct yet interconnected components. The initial phase depicted in Fig 3.1 involves predicting the Common Vulnerability Scoring System (CVSS) scores by employing a novel hybrid model that combines the strengths of transformers and Artificial Neural Networks (ANNs). This innovative approach aims to harness the deep contextual understanding capabilities of transformers, alongside the pattern recognition prowess of ANNs, to achieve a higher level of accuracy in CVSS score prediction.

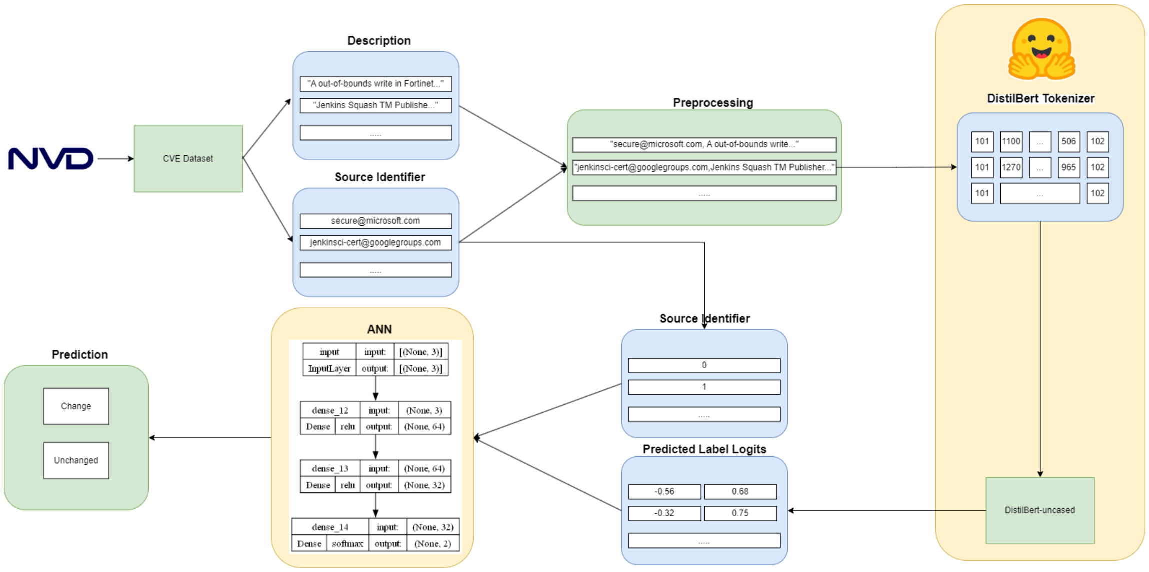


Figure 3.1: First Phase: Scenario for Scope

The second phase depicted in Fig 3.2 builds upon the foundation established by the first, aiming to enhance the robustness of the transformer model through the integration of an incremental learning technique. This method leverages the insights and knowledge derived from the initial phase to improve and adapt the model. This process ensures that the transformer model remains effective and efficient in predicting CVSS scores, even as it encounters new and evolving cybersecurity threats. By employing this two-pronged methodology, the research endeavors to significantly advance the precision and reliability of CVSS score predictions, thereby contributing to stronger and more adaptive cybersecurity defenses.

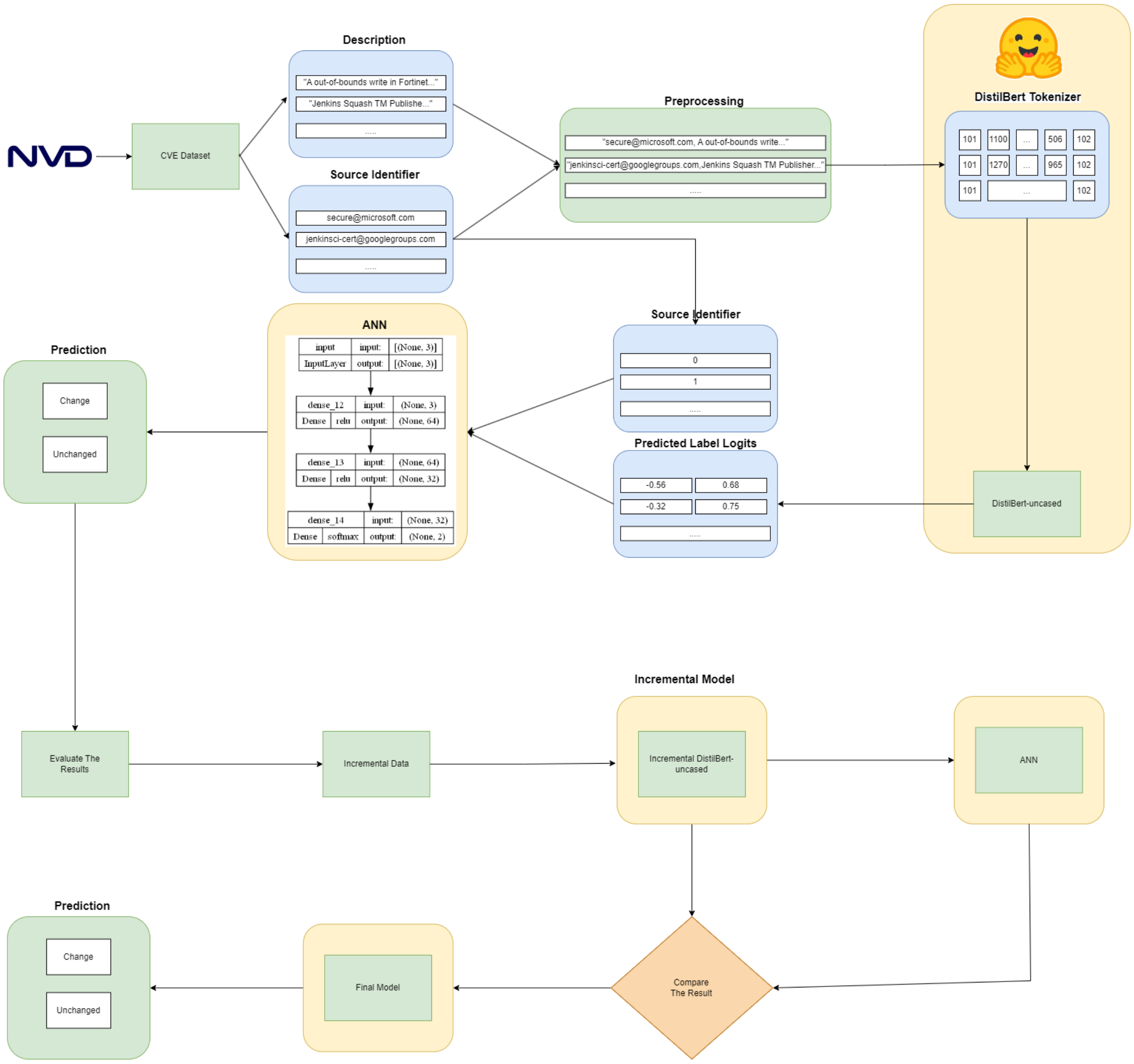

NVD
CVE Dataset
Description
"A out-of-bounds write in Fortinet..."
"Jenkins Squash TM Publishe..."
.....
Source Identifier
secure@microsoft.com
jenkinsci-cert@googlegroups.com
.....
Preprocessing
"secure@microsoft.com, A out-of-bounds write..."
"jenkinsci-cert@googlegroups.com,Jenkins Squash TM Publisher..."
.....
DistilBert Tokenizer
101 1100 ... 506 102
101 1270 ... 965 102
101 ... 102
DistilBert-uncased
Source Identifier
0
1
.....
Predicted Label Logits
-0.56 0.68
-0.32 0.75
.....
ANN
input InputLayer input: [(None, 3)] output: [(None, 3)]
dense_12 Dense relu input: (None, 3) output: (None, 64)
dense_13 Dense relu input: (None, 64) output: (None, 32)
dense_14 Dense softmax input: (None, 32) output: (None, 2)
Prediction
Change
Unchanged
Evaluate The Results
Incremental Data
Incremental Model
Incremental DistilBert-uncased
ANN
Compare The Result
Final Model
Prediction
Change
Unchanged

Figure 3.2: Second Phase: Scenario for Scope

In the following subsections, each component depicted in the diagrams of the respective phases will be comprehensively outlined.

3.2.1 CVE Dataset

3.2.1.1 Approach

The National Vulnerability Database (NVD) offers an API to access Comprehensive Vulnerability Enumeration (CVE) data, with the primary endpoint being [1]. This API supports multiple optional arguments to cater to various data retrieval needs. For the specific requirement—to accumulate the entire dataset the API endpoint [2] was utilized. This particular API call is structured to provide the first 2,000 CVEs starting from a defined startIndex, making it an efficient tool for data collection in bulk. To comprehensively acquire the full dataset of CVEs, an iterative request strategy was employed, dividing the total number of CVEs by 2,000. This approach ensured a systematic retrieval of all available CVE data, facilitating a robust dataset for the research analysis and contributing to the development of a predictive model for CVSS scores with enhanced accuracy and efficiency.

Upon successfully gathering the entire dataset from the NVD API, which delivers responses in JSON format, it became necessary to parse these JSON files into a format that is more conducive to analysis. The JSON responses from the NVD API comprise several elements: resultsPerPage, startIndex, totalResults, format, version, timestamp, and vulnerabilities. The essence of the CVE data is encapsulated within the vulnerabilities element, which houses the detailed information pertaining to each vulnerability.

To transform the JSON responses into a structured and analytically useful dataset, the approach focused on processing the data contained within the vulnerabilities key. This involved meticulously extracting the relevant details of each CVE and organizing them into a coherent format to enhance analysis ease and accessibility..A pivotal part of this restructuring was concatenating the source identifier to its cor-

[1] https://services.nvd.nist.gov/rest/json/cves/2.0

[2] https://services.nvd.nist.gov/rest/json/cves/2.0/?startIndex=startIndex

responding description, a process that significantly enriched our dataset by linking unique identifiers with detailed explanations. This meticulous approach to parsing and reorganizing the data from the vulnerabilities element enabled me to compile a foundational dataset, which is crucial for the subsequent phases of this research.

In our research, the dataset comprises 220,914 vulnerability descriptions and corresponding categories. However, we exclusively focus on descriptions pertinent to version 3 of the Common Vulnerability Scoring System (CVSS) for this study. Consequently, the total dataset size is reduced to 92,213 instances. This dataset is then divided into training and testing sets, comprising 73,770 and 18,443 instances, respectively. This division reflects a testing ratio of 0.2, ensuring that each set maintains a similar proportion of classes, as detailed in table 3.1.

There exists a temporal gap between the initial data collection phase and the subsequent experimental analysis. During this interval, we amassed an additional 16,000 new data entries. These entries as 'new data' were meticulously labeled, ensuring that all new instances adhere to descriptions related to version 3 of CVSS. This approach not only updates our dataset but also ensures continuity and relevance by aligning with the specific focus on CVSS version 3 vulnerabilities.

3.2.1.2 Background

The National Vulnerability Database (NVD) is a fundamental resource for cybersecurity research and analysis. Researchers extensively utilize NVD data in various studies to comprehend vulnerability severity, exploits, and cybersecurity knowledge graphs. Some studies concentrate on comparing vulnerability severity and exploits through case-control studies [10], while others focus on constructing cybersecurity knowledge graphs from malware after-action reports [110]. The NVD dataset has played a crucial role in predicting CVSS metrics [35], vulnerability discovery patterns [96], and vulnerability exploitation [67].

In addition to facilitating empirical research, the utilization of the NVD dataset underscores the importance of collaboration and information sharing in the cybersecurity community. By contributing to the collective knowledge base of vulnerabilities, researchers, practitioners, and organizations can collectively bolster defenses against emerging threats and enhance the resilience of software systems. Moreover, the availability of the NVD API enhances the ability to replicate and validate research outcomes, promoting a culture of accountability and thoroughness in the field of cybersecurity studies.

The NVD dataset has been utilized to develop vulnerability datasets by examining Common Vulnerabilities and Exposures (CVE) and other databases [76]. It has also been used to forecast the discovery pattern of publicly known exploited vulnerabilities [96]. Additionally, the NVD dataset has been integrated into vulnerability prediction models [95], vulnerability assessment [86], and security defect analysis [108].

Sponsored by the Department of Homeland Security, the NVD is considered a standard source due to its reliability and comprehensive coverage [153]. However, to enhance the reliability and precision of vulnerability assessments, researchers recommend combining statistical interpretations of CVE and NVD datasets with other live security-related data sources [73].

Furthermore, the NVD dataset is a cornerstone in cybersecurity research, offering valuable insights into vulnerability severity, exploits, prediction models, and assessment. Its extensive coverage and reliability make it an essential tool in understanding and addressing cybersecurity challenges.

Table 3.1: Percentage Distribution of Categories in Train, Test, and new Data

Category		**Train**	**Test**	**New Data**
Attack Vector	Network	72.02	71.85	76.55
	Local	24.55	24.52	20.80
	Adjacent_network	2.33	2.52	1.80
	Physical	1.08	1.09	0.83
Attack Complexity	Low	95.18	94.94	96.80
	High	4.81	5.05	3.19
Privileges Required	None	59.38	59.35	58.39
	Low	30.88	30.40	32.43
	High	9.73	10.24	9.16
User Interaction	None	67.07	66.65	64.46
	Required	32.92	33.34	35.53
Scope	Unchanged	82.62	82.35	79.91
	Changed	17.37	17.64	20.08
Confidentiality Impact	High	58.13	57.78	58.01
	None	22.02	22.03	18.02
	Low	19.84	20.18	23.95
Integrity Impact	High	49.68	49.44	47.51
	None	31.56	31.35	28.74
	Low	18.75	19.19	23.73
Availability Impact	High	58.22	57.96	54.03
	None	39.50	39.81	44.28
	Low	2.27	2.22	1.67

3.2.2 Preprocessing

3.2.2.1 Approach

After preparing the data, we perform preprocessing to improve the dataset's quality. These procedures include common Natural Language Processing (NLP) techniques, such as removing stop words and lemmatization and removing numbers. Initially, the process begins by stripping the text of any numerical figures. Subsequently, we filter out stop words—commonly occurring words that offer minimal value for analysis—to refine the content further. The next crucial step involves the application of lemmatization, a process that transforms words into their root forms, facilitating a more accurate and meaningful analysis of the text. Each of these steps is meticulously detailed in Section A.1, providing a comprehensive overview of our text preprocessing methodology. Also, an example of the preprocessing output is illustrated in 3.2.

Table 3.2: Illustration of Preprocessing Result

CVE-ID	CVE-2024-21762
Source Identifier	secure@microsoft.com
Description	A out-of-bounds write in Fortinet FortiOS versions 7.4.0 through 7.4.2, 7.2.0 through 7.2.6, 7.0.0 through 7.0.13, 6.4.0 through 6.4.14, 6.2.0 through 6.2.15, 6.0.0 through 6.0.17, FortiProxy versions 7.4.0 through 7.4.2, 7.2.0 through 7.2.8, 7.0.0 through 7.0.14, 2.0.0 through 2.0.13, 1.2.0 through 1.2.13, 1.1.0 through 1.1.6, 1.0.0 through 1.0.7 allows attacker to execute unauthorized code or commands via specifically crafted requests.
Preprocessed Text	secure@microsoft.com, A out-of-bounds write. Fortinet FortiOS version FortiProxy version allow attacker execute unauthorized code command via specifically craft requests.

3.2.2.2 Background

Standard preprocessing techniques commonly used include operations such as word segmentation, tokenization, stemming, stop-word removal, lemmatization, and normalization. These processes are essential not only for preparing text data for analysis but also for improving the accuracy and effectiveness of NLP models in different fields. These techniques play a crucial role in improving the quality of text data before it is analyzed further [91, 25, 44]. Preprocessing has a vital role in reducing noise and enhancing the quality of text, consequently impacting the finalperformance of NLP systems [20, 91]

These preprocessing methods are not only applicable to traditional text data, but also to social media data. They have been found to be effective in handling and retrieving valuable information from social media [162].

Furthermore, the significance of preprocessing is emphasized in various applications and areas. Text cleaning and normalization play a crucial role in sentiment analysis, providing accurate sentiment classification [139]. Effective preprocessing strategies greatly enhance the performance of named entity recognition (NER), text clustering, topic recognition, and the handling of multilingual data [128, 141, 81].

Preprocessing is essential in specialized tasks such as plagiarism detection and cyberbullying classification. It ensures that the text data is formatted correctly for accurate analysis. These techniques play a crucial role in the field of natural language processing (NLP), as emphasized by [50, 77]

These steps refine and optimize the data, making it ready for subsequent analysis. Preprocessing is an indispensable step in structuring raw text data into a format that automated computing systems can effectively utilize. The quality of data preprocessing significantly influences the accuracy and efficiency of NLP applications [97].

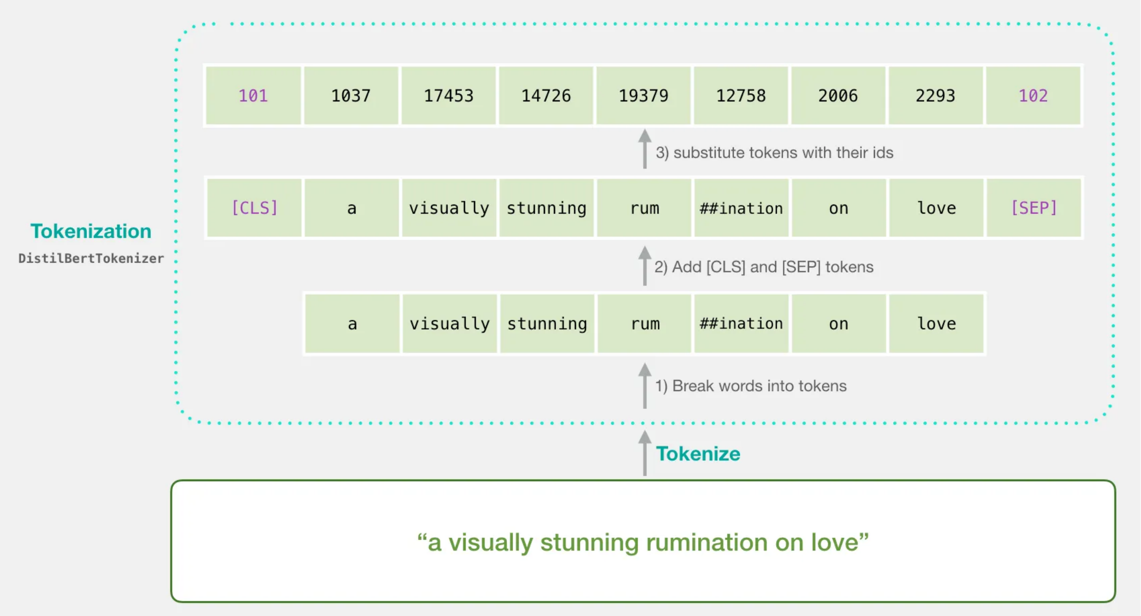
In the broader scope of NLP, advanced language representation models like Bidirectional Encoder Representations from Transformers (BERT) have significantly enhanced NLP capabilities, establishing new standards for the field [43]. Despite these advancements, the importance of text preprocessing remains undiminished. It serves as a foundational step enabling the extraction of meaningful insights from unstructured text data [25].

The practice of preprocessing in NLP encompasses a range of techniques from basic operations like tokenization to more advanced methods like topic modeling. These techniques are pivotal in enhancing the quality of text data for subsequent analysis, thereby improving the accuracy and effectiveness of NLP systems across a variety of domains [69, 20, 91, 25, 44] This comprehensive approach to preprocessing underscores its critical role in the success of NLP applications, from sentiment analysis and named entity recognition to the management of multilingual data and beyond.

3.2.3 Tokenization

3.2.3.1 Approach

Following the preprocessing of our dataset, the next critical step involves preparing the data for our transformer model. This preparation is achieved through the use of a tokenizer, a tool essential for converting textual data into a format that our model can understand and process efficiently. Given the selection of DistilBert for our model architecture, the distilbert-base-uncased tokenizer for this task were employed. This specific tokenizer is designed to work seamlessly with the distilbert-base-uncased model, ensuring optimal compatibility and performance. The process of tokenizing an input example using the DistilBERT tokenizer is depicted in 3.3.


Tokenization
DistilBertTokenizer
3) substitute tokens with their ids
2) Add [CLS] and [SEP] tokens
1) Break words into tokens
Tokenize
“a visually stunning rumination on love”

101	1037	17453	14726	19379	12758	2006	2293	102
[CLS]	a	visually	stunning	rum	##ination	on	love	[SEP]
	a	visually	stunning	rum	##ination	on	love	

Figure 3.3: Tokenization in DistilBERT [126]

The distilbert-base-uncased tokenizer plays a pivotal role in breaking down the preprocessed text into tokens, converting these tokens into numerical representations, and standardizing the text input to fit the model's requirements. This process involves lowercasing the text (as indicated by 'uncased') and splitting it into subword units that DistilBert can effectively analyze. The use of this tokenizer not only facilitates a smoother transition of data into the transformer model but also enhances the overall efficiency and accuracy of the CVSS score prediction process, leveraging the distilled knowledge of the DistilBert architecture to achieve superior results.

3.2.3.2 Background

Tokenization is a key component of transformer models, which are recognized by their ability to process complex data using self-attention methods. Tokenization transforms raw text or image patches into a series of tokens. This step is crucial not only for understanding and analyzing the input data but also for enhancing the performance and capabilities of transformer models across various tasks [70, 33, 25, 41].

Tokenization techniques have evolved to include not just the segmentation of text into tokens but also the integration of external knowledge from sources like knowledge graphs into transformer-based models. This integration, facilitated through methods like vector space projection and selective attention, significantly enriches the models' understanding and processing capabilities, particularly in NLP tasks [51, 48].

In the field of natural language processing (NLP), tokenization plays a crucial role, particularly in models like BERT. Tokenization allows for the reconstruction of corrupted tokens and sets standards for transfer learning, as demonstrated in studies by [48, 21].

Table 3.3: Detailed Hyperparameters of the DistilBert

Parameter Name	Value
Batch Size	8
Epochs	10
Loss Function	Sparse Categorical Crossentropy
Optimizer	Adam
Learning rate	3e-5

3.2.4 Transformers

3.2.4.1 Approach

Due to the findings from other researchers [131, 35, 130, 84], who have successfully utilized DistilBERT in their work. Taking these observations into account, DistilBERT for the transformer component of the proposed pipeline was ultimately selected. This decision was based on its superior performance and the widespread adoption and validation by the research community. For the implementation of DistilBert within TensorFlow, the transformers library was utilized and specifically employed the TFDistilBertForSequenceClassification function from Hugging Face. This function is designed to accept a specified number of labels, effectively adding an output layer tailored for classification tasks. This setup facilitates the creation of a sophisticated model capable of handling various classification challenges by leveraging the efficiency and performance of DistilBert. The hyperparameters utilized for training the DistilBERT model are outlined in Table 3.3.

3.2.4.2 Background

Transformers have become a crucial advancement in Natural Language Processing (NLP), representing a significant change in how textual data is analyzed and understood. The introduction of groundbreaking algorithms like OpenAI's Generative Pre-trained Transformer (GPT) [114] and Google's Bidirectional Encoder Representations from Transformers (BERT) [43] in 2018 has led the way for significant advances in Natural Language Processing (NLP) tasks [161]. Thanks to their novel self-attention mechanism, this invention is based on the transformers' capacity to record complex relationships within textual data effectively. This mechanism excels at handling long-range dependencies in sentences, a job that previous models have historically struggled with [107]. The architecture of the transformer model is illustrated in 3.4.

The attention mechanism is a crucial component of transformers that allows the model to dynamically concentrate on various parts of the input sequence during processing, hence contributing to their transformational impact. This capability enables the model to effectively acquire knowledge of long-term dependencies within the data, which is a notable improvement compared to RNNs [147]. The attention mechanism distributes weights to incoming sequence items according to their importance to the present context, allowing for the capture of long-range dependencies within the data [147].

Transformers have a significant influence not just in natural language processing (NLP) but also in other areas like computer vision. Attention-based models, specifically Vision Transformers (ViT), have shown significant improvements in performance across many applications, such as text mining, image classification [4], and biology [148]. The adaptability of the transformer architecture is highlighted by its capacity to effectively capture non-local relationships via its attention mechanism, making it applicable across many domains [147].

Moreover, transformers are a class of deep learning models that have successfully overcome the constraints of earlier designs such as recurrent neural networks (RNNs) or Long short-term memory (LSTM) [64], by efficiently capturing long-term relationships in data. This is achieved through their unique processing of the entire input sequence concurrently, as opposed to the sequential processing typical of RNNs [39]. In order to enhance their capacity to handle larger amounts of data and improve their effectiveness, new versions of Transformers called Sparse Transformers and Performers have been developed. These variants are designed to handle other sorts of data, not only text, which expands the range of applications for Transformers [45, 34]. The transformers demonstrate their versatility by being applied to time-series analysis [5] and integrated with convolutional backbones to improve visual recognition tasks. This showcases their ability to adapt to different domains beyond their original use in natural language processing [105]. The capacity to adapt, together with the difficulties of implementing transformers on hardware because of their high computing requirements, highlights the continuous development of these models in their pursuit of efficiency and effectiveness in different tasks [151].

DistilBERT, introduced by [123] from Hugging Face, is a streamlined version of BERT designed to be smaller, faster, and more efficient, yet nearly as effective as its predecessor. This model retains 97% of BERT's language understanding capabilities while being 40% smaller and 60% faster, making it an ideal choice for environments with limited computational resources, such as mobile devices. The efficiency and compactness of DistilBERT, especially in comparison to other transformer models, are represented in 3.5, which illustrates its parameter count relative to its counterparts, highlighting its relative efficiency in terms of size.. DistilBERT's development was motivated by the growing concern over the environmental and computational costs associated with the increasing size of state-of-the-art language models. By employing a knowledge distillation technique during the pre-training phase, the authors successfully compressed BERT without significantly compromising its performance. This process involves training DistilBERT (the student) to mimic BERT (the teacher) using a triple loss function that combines language modeling, distillation, and cosine-distance losses. DistilBERT's architecture mirrors that of BERT but with reduced complexity, including fewer layers and the absence of token-type embeddings and the pooler. The resulting model not only demonstrates comparable

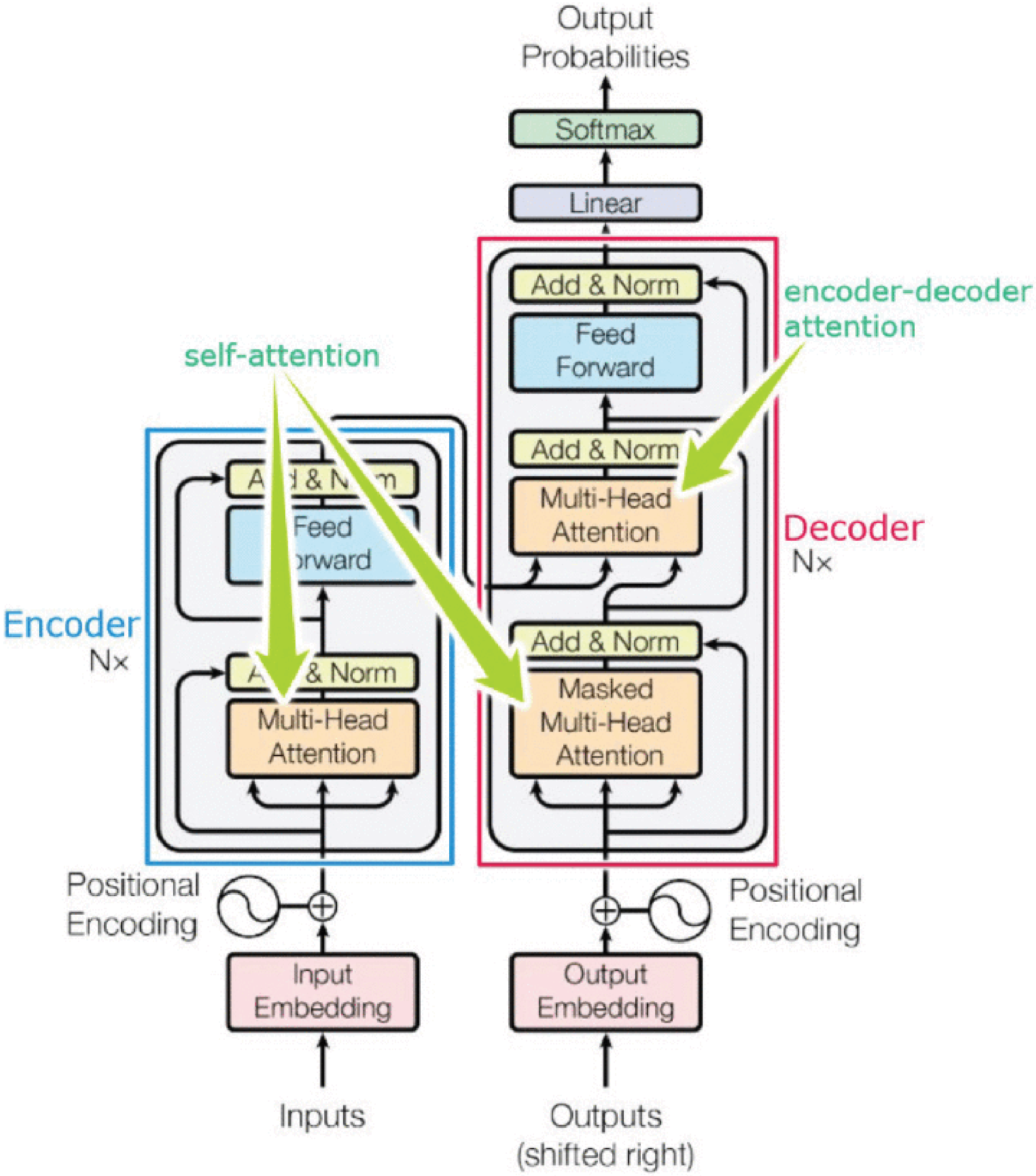
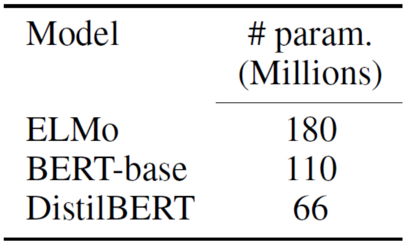


Figure 3.4: The Transformer - model architecture [123]

performance on a wide range of NLP tasks but also underscores the feasibility of using smaller, efficient models for on-device applications and beyond.



Model	# param. (Millions)
ELMo	180
BERT-base	110
DistilBERT	66

Figure 3.5: Parameter Counts: DistilBERT Versus Other Transformer Models [123]

3.2.5 ANN

3.2.5.1 Approach

This model, as illustrated in 3.6, underwent meticulous refinement through rigorous experimentation. In particular, the Artificial Neural Network (ANN) incorporated both the logits from the transformer's output and the numerically transformed source identifiers as its input. This strategic choice allowed for a less complex architectural design, effectively reducing the risk of overfitting—a common issue in more intricate models. Remarkably, the performance metrics detailed in 4.7 showcase the ANN's superiority over previous methods that are disccused in Chapter 4. It achieved enhanced accuracy and F1 scores compared to those of the standalone transformer model. This achievement highlights the critical importance of customizing the ANN's architecture to align with the unique attributes of the input data. It exemplifies how striking a balance between architectural complexity and model efficacy is vital to avoid overfitting, The hyperparameters of the ANN archtirecure is shown in 3.4

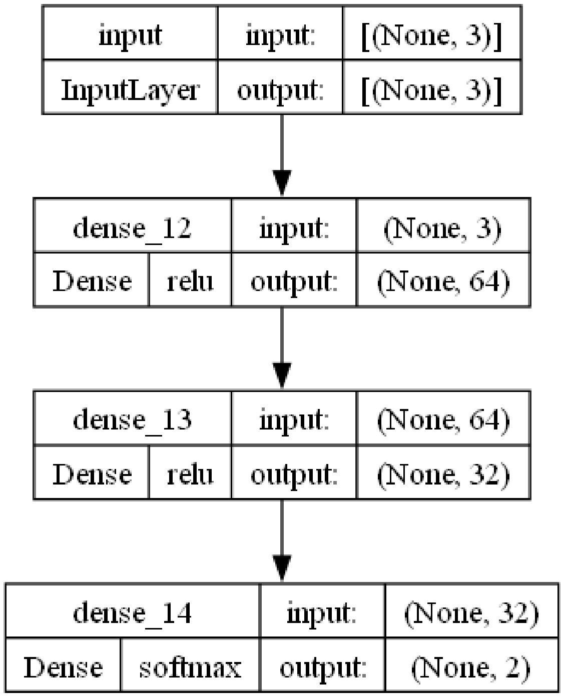


Figure 3.6: Architecture of ANN for Scope Scenario.

Table 3.4: Detailed Hyperparameters of the ANN Architecture

Parameter Name	Value
Batch Size	128
Epochs	100
Loss Function	Categorical Crossentropy
Optimizer	Adam
Learning rate	0.001

3.2.5.2 Background

Artificial Neural Networks (ANNs) have become a fundamental component in the area of artificial intelligence. They are inspired by the neural networks found in the human brain and have seen considerable advancements in tackling complicated challenges in several fields. Introduced as a model for information storage and organization in the brain by [117], the perceptron laid the groundwork for the development of ANNs. The overview of perceptron's architecture is illsutred in 3.7. This basic but fundamental notion showcased the capacity of computer models to acquire patterns and make projections, hence facilitating the development of more sophisticated architecture.

The neuron represents the most basic unit of processing within an Artificial Neural Network (ANN) system. The output is determined by applying an activation function to the inputs and their associated weights. The term "biases" (b) refers to the connection of weight values to individual nodes. These weight values across the network are established through the repetitive processing of training data. Throughout the training phase, the weight values are adjusted as the network learns to recognize specific patterns based on the features of the given input data [115]. The fundamental architecture of an ANN is illustrated in F3.8.

The adaptability and efficacy of ANNs have been shown in several domains, including meteorology, medicine, engineering, and others [80, 158, 140]. These networks have performed exceptionally well in tasks such as regression, classification, and approximation-based learning processes. They have been applied in various areas including but not limited to predicting monsoon rainfall, estimating air temperature, estimating mechanical properties of materials, and improving medical decision-making processes [38, 53, 134, 15, 23, 137, 103, 26, 29, 115].

The adaptability of ANNs has been further demonstrated through their integration with various techniques, such as particle swarm optimization and Fast Block Least Mean Square algorithms, significantly improving prediction accuracies in fields like wind speed forecasting and the modeling of the Consumer Price Index [124, 52]. Moreover, the history of ANNs from their inception as simple perceptrons to the sophisticated deep learning architectures of today illustrates their significant evolution. The development of learning algorithms and network architectures, such as backpropagation in the 1980s and the exploration of CNNs and Recurrent Neural Networks (RNNs), has enabled unparalleled performance in pattern recognition, classification tasks. Convolutional Neural Networks (CNNs), a class of ANNs, have revolutionized computer vision tasks in radiology, while other network architectures have been instrumental in solving differential equations and intelligent support in strategic decision-making for energy development [156, 160, 55, 106, 116, 89, 63]. This continuous innovation has expanded ANNs' applicability, integrating them with technologies like fuzzy logic and neuro-fuzzy systems [71]

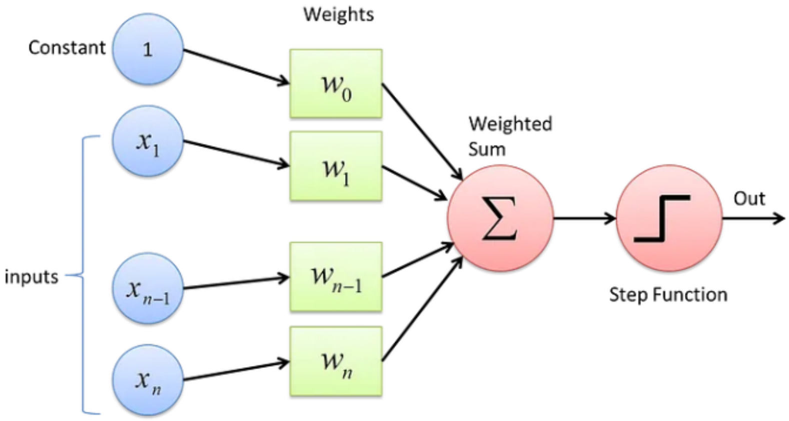


Figure 3.7: Overview Of Perceptron Architecture [133].

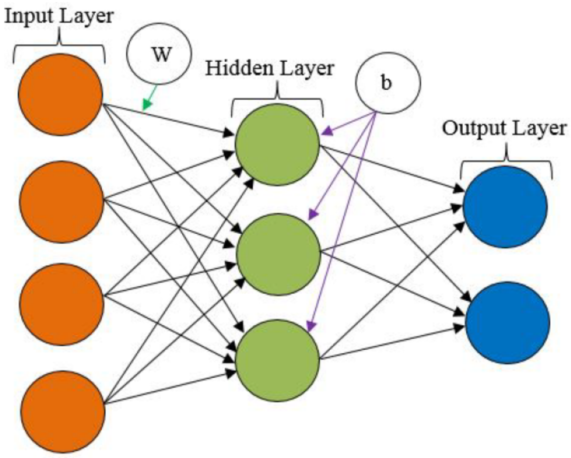


Figure 3.8: The basic ANN architecture. [115].

3.3 Second Phase

3.3.1 Incremental learning

The results presented in in table 4.7 demonstrate an improvement in both accuracy and F1 score when an Artificial Neural Network (ANN) is integrated with the transformer model. This enhancement is evident from a comparison between the outcomes derived solely from the transformer and those where the ANN is used in conjunction. Specifically, the input to the ANN consists of the output from the last layer of the transformer classifier, where predictions are typically made based on the maximum value in this layer. This configuration allows the ANN to effectively refine the classification process.

Interestingly, the ANN appears to correct certain errors commonly introduced by the transformer's method of determining labels through maximum values. These errors might be due to overlooked patterns that the ANN can capture and address, thus enhancing the overall model performance.

To address these discrepancies and improve model robustness, incremental learning technique was employed. This approach was necessary as no pre-existing libraries support this method for transformers or deep learning models.

For the practical implementation of these enhancements, various strategies were employed. The most effective involved reducing the learning rate to one-tenth of its original value. Additionally, a strategic selection of instances—specifically 8,000, or one-tenth of the training dataset, combined with where the transformer was correct but the ANN erred, combined with instances where the ANN correctly predicted outcomes that the transformer initially misclassified, were used to fine-tune the model. This subset was instrumental in recalibrating the model's learning process.

The experimental results validate this approach, confirming that the refined model is substantially more robust, as evidenced by the improved performance metrics. The hyper-parameters of this process is illustrated in table 3.5

Table 3.5: Detailed Hyperparameters of the incremental DistilBert

Parameter Name	Value
Batch Size	4
Epochs	2
Loss Function	Sparse Categorical Crossentropy
Optimizer	Adam
Learning rate	3e-6

After refining the transformer model through incremental learning, a similar process was applied to the Artificial Neural Network (ANN). It is noteworthy that, given the relatively brief training duration of approximately two minutes for the ANN, it is feasible to retrain it from scratch when necessary. Additionally, due to the simpler architecture of the ANN, a more substantial reduction in the learning rate was implemented, decreasing it by a factor of 100 instead of 10. This adjustment was made to optimize the incremental learning process under the constraints of the ANN's design. Details of the hyperparameters used in the incremental learning process of the ANN are provided in Table 3.6.

The performance of the ANN varied across different use cases. For instance, in scenarios involving the attack vector on new data, the ANN exhibited lower accuracy compared to the transformer model. However, for confidentiality impact and scope, the accuracy of the ANN mirrored that of the transformer. For other use cases, the accuracy of the ANN improved.

This variability suggests that the inclusion of the ANN in the operational framework is conditional. Specifically, the ANN should be incorporated only if it enhances the evaluation metrics. The underlying rationale is that the incrementally trained transformer model has already corrected its initial errors, limiting the additional insights the ANN can provide based on the existing data. However, as new data becomes available, it may be beneficial to retrain the ANN or apply the incremental learning approach to it. This would allow for a direct comparison with the transformer model's enhanced learning capabilities, potentially enabling the ANN to further learn from and correct any residual errors.

Table 3.6: Detailed Hyperparameters of the incremental learning process of the ANN

Parameter Name	Value
Batch Size	32
Epochs	20
Loss Function	Categorical Crossentropy
Optimizer	Adam
Learning rate	0.00001

3.4 Concluding Remarks

This chapter has provided a detailed account of the methodology employed to predict CVSS scores using a transformative approach that integrates DistilBERT with ANN. The application of a multi-step, incremental learning model is particularly noteworthy. This approach ensures that the predictive model adapitably refines its performance as new data emerges, maintaining its accuracy and relevance despite the dynamic nature of cybersecurity threats. The integration of incremental learning with the base model allows for continuous improvement of the model.

Moreover, this chapter extensively discusses the preparation and preprocessing of the data, which are critical steps that ensure the quality and reliability of the model's predictions. By detailing the steps involved in data collection, preprocessing, and the configuration of machine learning models, this chapter not only provides a blueprint for replicating the study but also sets a benchmark for future research in predictive modeling within cybersecurity. As the thesis progresses into experimental results and analysis, the methodologies described in this chapter will be vital for understanding the effectiveness of the proposed models in practical scenarios.

Chapter 4

Experimental Results

4.1 Data Exploration

This section details the CVE dataset that was gathered from the National Vulnerability Database (NVD). This dataset forms the basis for the analyses conducted in this study, providing critical insights into the vulnerabilities it encompasses.

As shown in Figure 4.1, there is a notable increase in the number of CVEs reported each year. For instance, the number rose from approximately 5,000 CVEs in 2013 to over 30,000 CVEs in 2023. In recent years, the number of CVEs has skyrocketed, and according to the trends depicted in the figure, we anticipate this rise to continue in the future. This trend underscores the need for organizations and companies to pay closer attention to vulnerabilities.

The figures, referenced from 4.2 to 4.9, collectively illustrate the distribution of CVEs reported by top 10 sources in the National Vulnerability Database (NVD) from 1988 to 2023, categorized by various CVSSv3 metrics including attack vector, complexity, privileges required, user interaction, scope, confidentiality impact, integrity impact, and availability impact.

- Attack Vector (4.2): Most vulnerabilities cataloged by cve@mitre.org are exploitable over network connections, with Microsoft showing significant vulnerabilities that require local access.

- Attack Complexity (4.3): A large number of CVEs, especially from cve@mitre.org and GitHub, have low attack complexity, indicating easier exploitation methods.

- Privileges Required (4.4): High numbers of CVEs reported by Cisco and Android require advanced user access, reflecting more serious access control challenges.

- User Interaction (4.5): Many vulnerabilities, particularly from Adobe, necessitate user involvement for exploitation, emphasizing the role of user behavior in security.

- Scope (4.6): GitHub reports a relatively high number of CVEs that could lead to broader security implications beyond the initially compromised component.

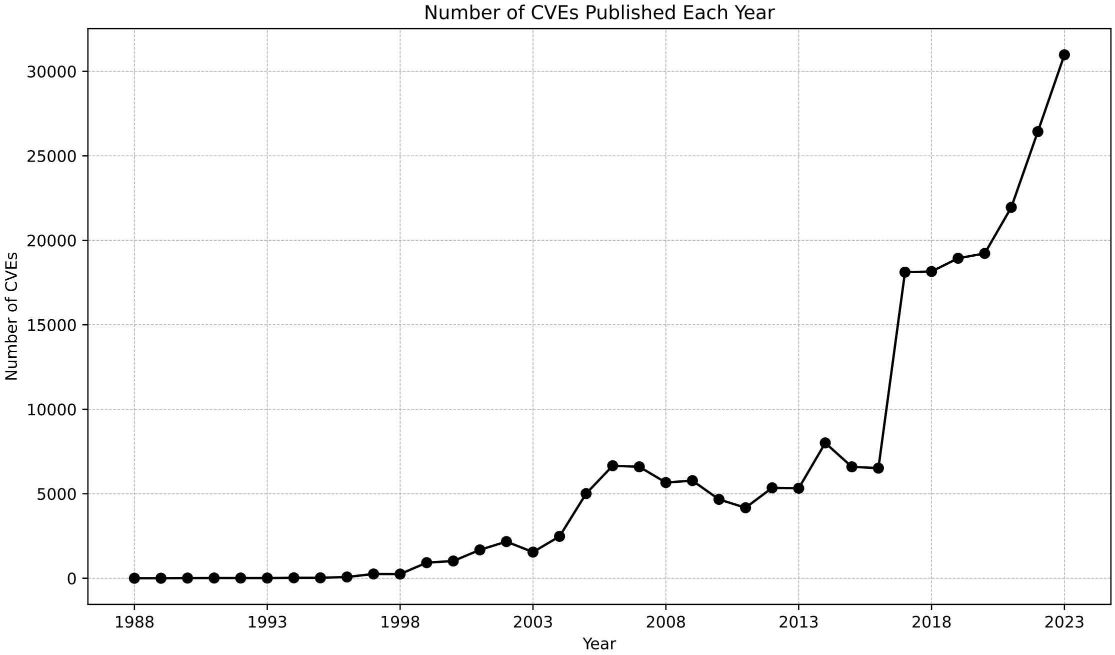


Figure 4.1: Number Of CVEs Each Year

- Confidentiality Impact (4.7): cve@mitre.org and Microsoft report many high-impact CVEs that could lead to serious data breaches.

- Integrity Impact (4.8): Substantial risks to system integrity are reported by cve@mitre.org and Microsoft, highlighting vulnerabilities that could allow unauthorized data modifications.

- Availability Impact (4.9): Microsoft and other sources show a range of vulnerabilities affecting system availability, with varying degrees of impact severity.

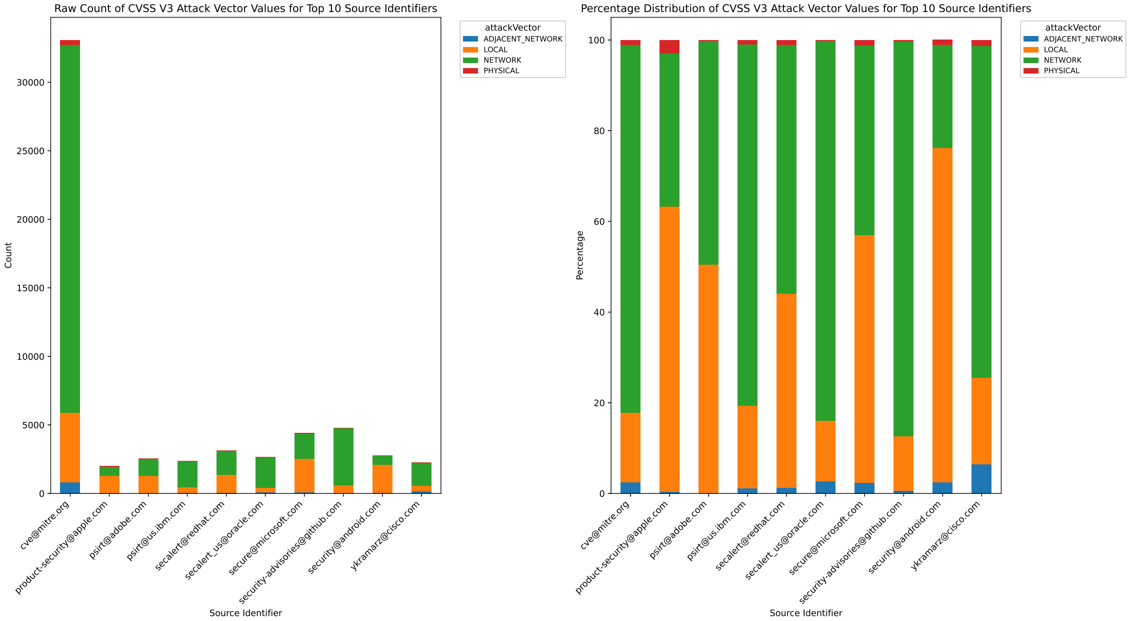


Figure 4.2: Distribution of CVSS V3 Attack Vector Values for Top 10 Source Identifiers

The box plot 4.10 visually quantifies the base scores of vulnerabilities for the most reported vendors in the National Vulnerability Database, showing a moderate to high median score for each. Dell and Microsoft, with a notable stretch of outliers, indicate a significant number of vulnerabilities at both ends of the severity spectrum, possibly requiring targeted attention for the more extreme cases. HP and Qualcomm are characterized by higher median scores, implying a tendency toward more severe vulnerabilities. The interquartile ranges for F5 and Intel, wider than those of other

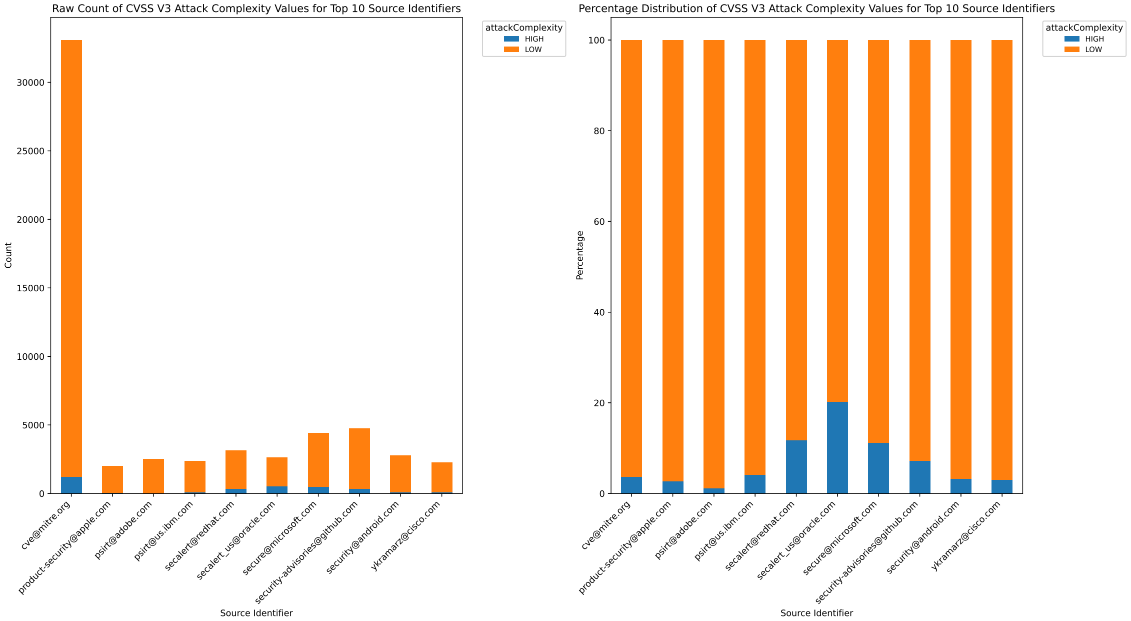


Figure 4.3: Distribution of CVSS V3 Attack Complexity Values for Top 10 Source Identifiers

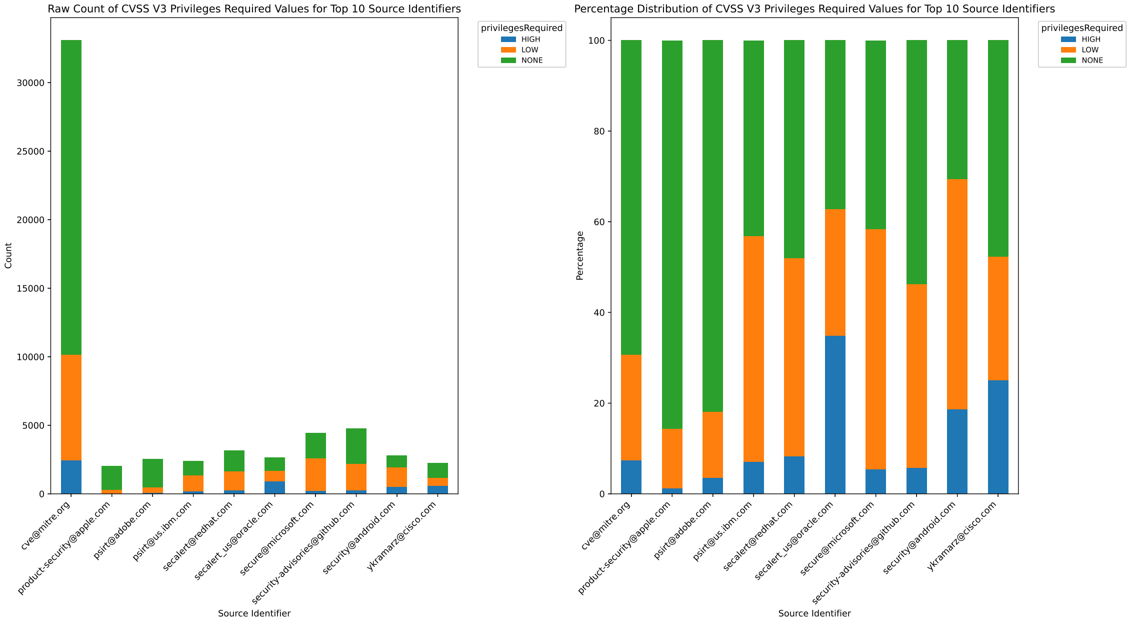


Figure 4.4: Distribution of CVSS V3 Privileges Required Values for Top 10 Source Identifiers

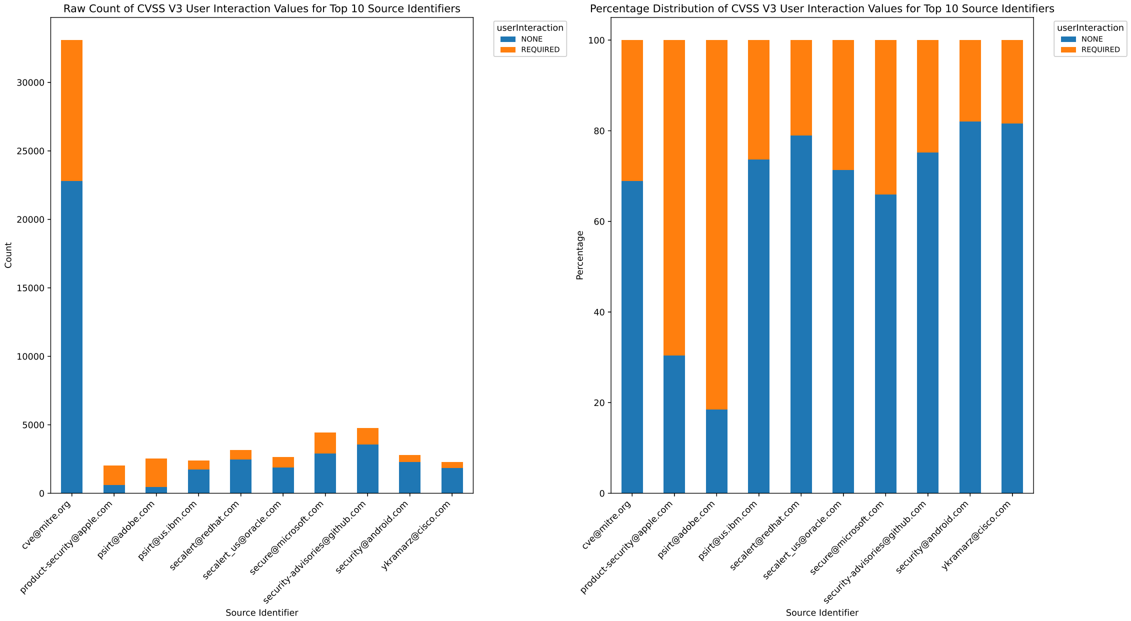


Figure 4.5: Distribution of CVSS V3 User Interaction Values for Top 10 Source Identifiers

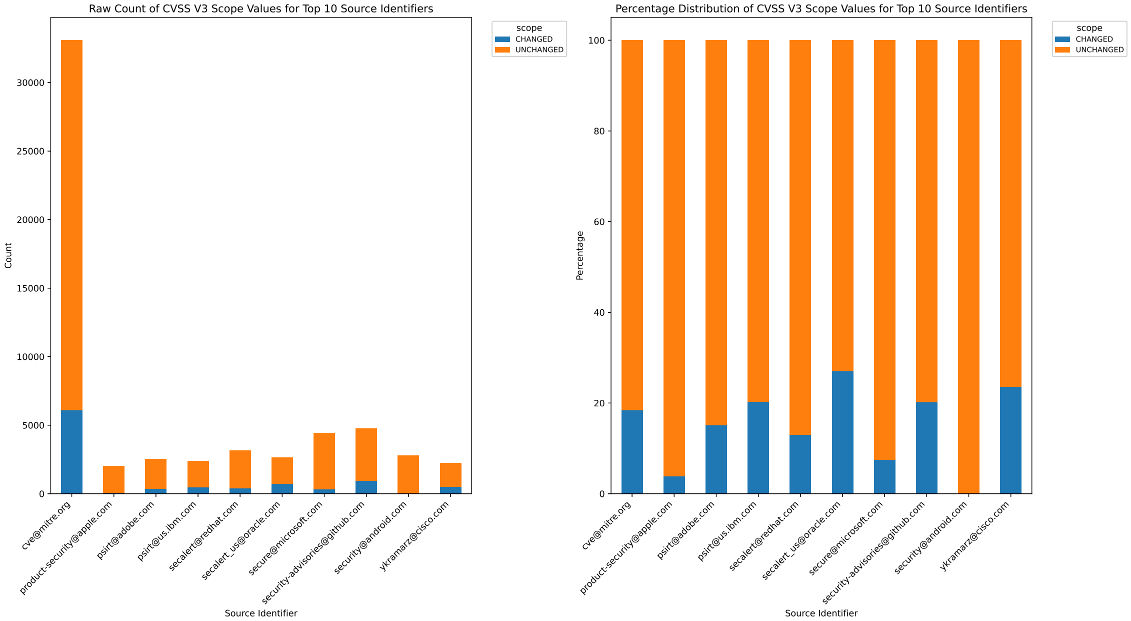


Figure 4.6: Distribution of CVSS V3 Scope Values for Top 10 Source Identifiers

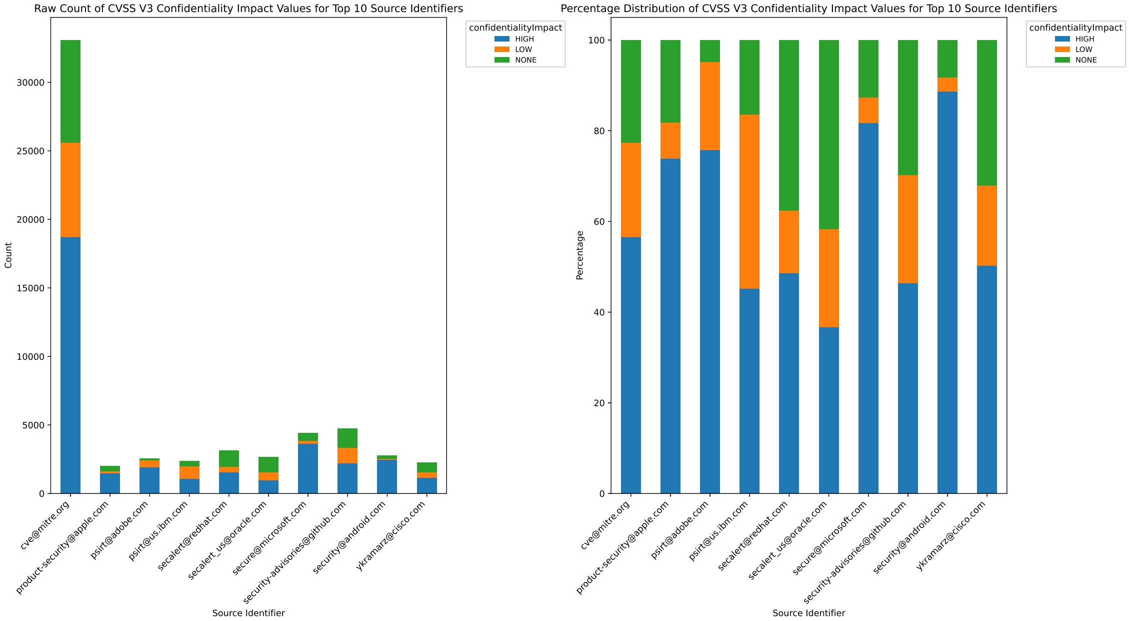


Figure 4.7: Distribution of CVSS V3 Confidentiality Impact Values for Top 10 Source Identifiers

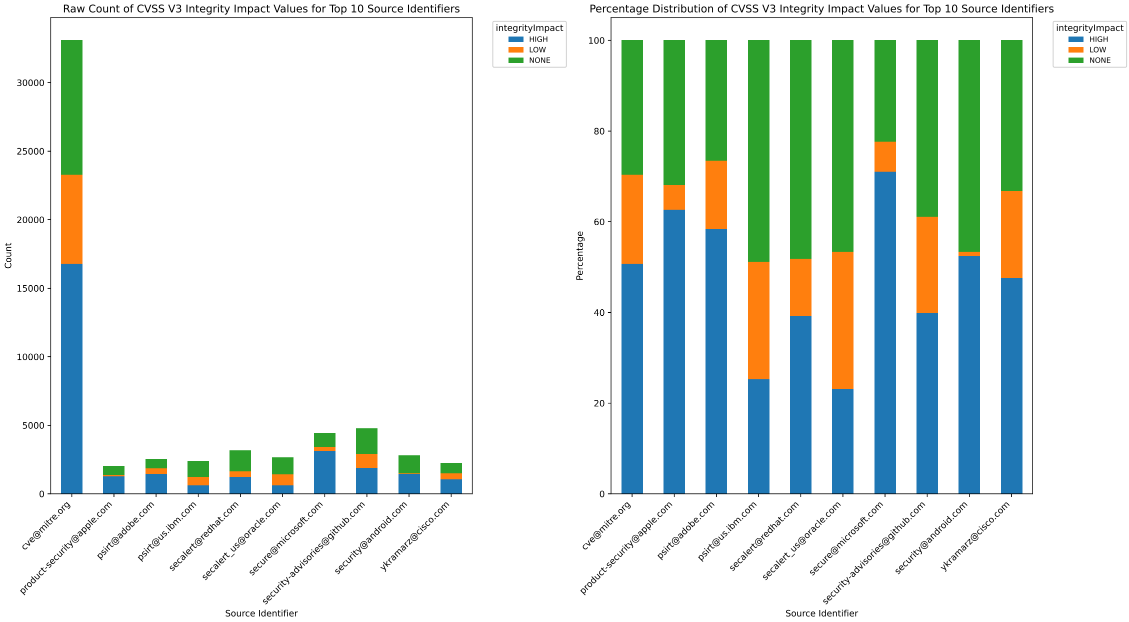


Figure 4.8: Distribution of CVSS V3 Integrity Impact Values for Top 10 Source Identifiers

vendors, suggest varied severity levels in their reported vulnerabilities. Qualcomm's narrower range points to a consistency in reporting higher severity vulnerabilities.

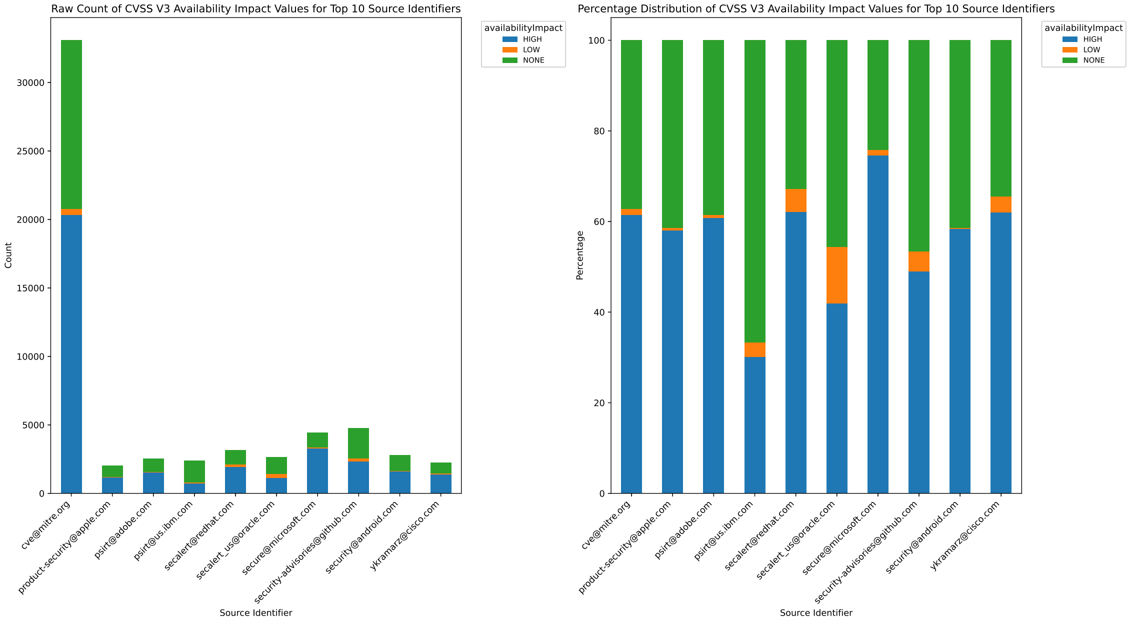


Figure 4.9: Distribution of CVSS V3 Availability Impact Values for Top 10 Source Identifiers

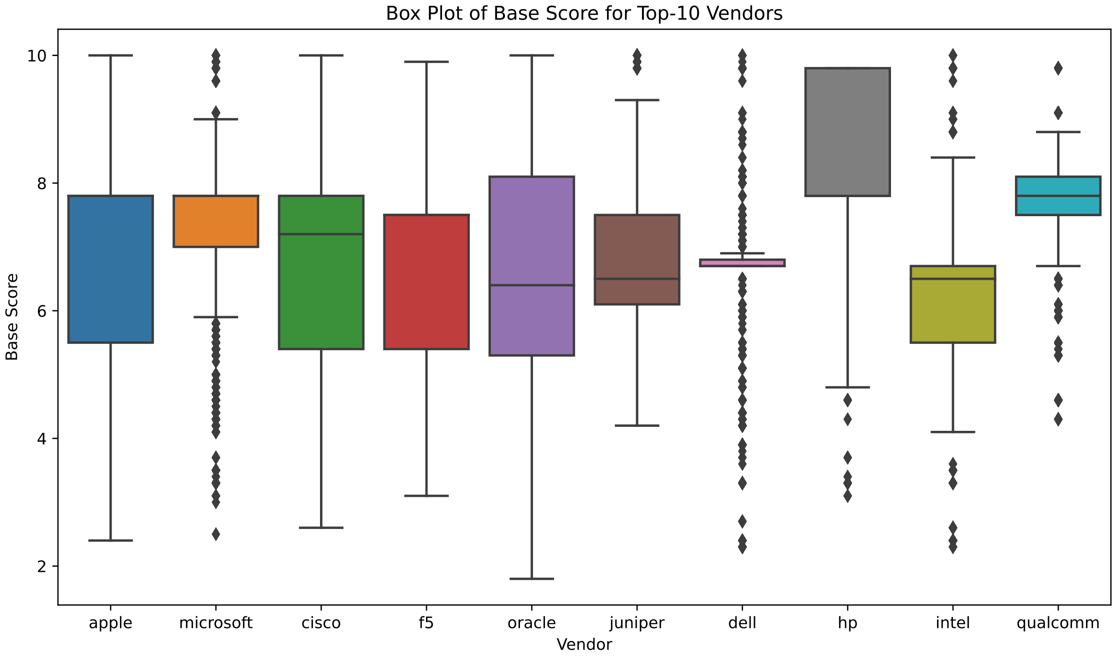


Figure 4.10: Box Plot of CVSS V3 Base Score Values for Top-10 Vendors

The box plot 4.11 provides a comparative visualization of exploitability scores across top 10 vendors. Vendors such as Apple, Juniper, Microsoft, and Dell exhibit more uniform exploitability scores, with Dell showing a particularly tight interquartile range that indicates a general trend of lower exploitability risk across most of its products; however, Dell also presents a noticeable number of outliers, suggesting that some vulnerabilities significantly deviate from this trend. On the other hand, vendors like Cisco , Qualcomm, Hp, and Oracle display broader score ranges, pointing to a more variable exploitability landscape within their offerings, with Oracle showing the highest median score of all, indicative of a greater average exploitability risk. HP's plot, with a median around 2, has the largest interquartile range.

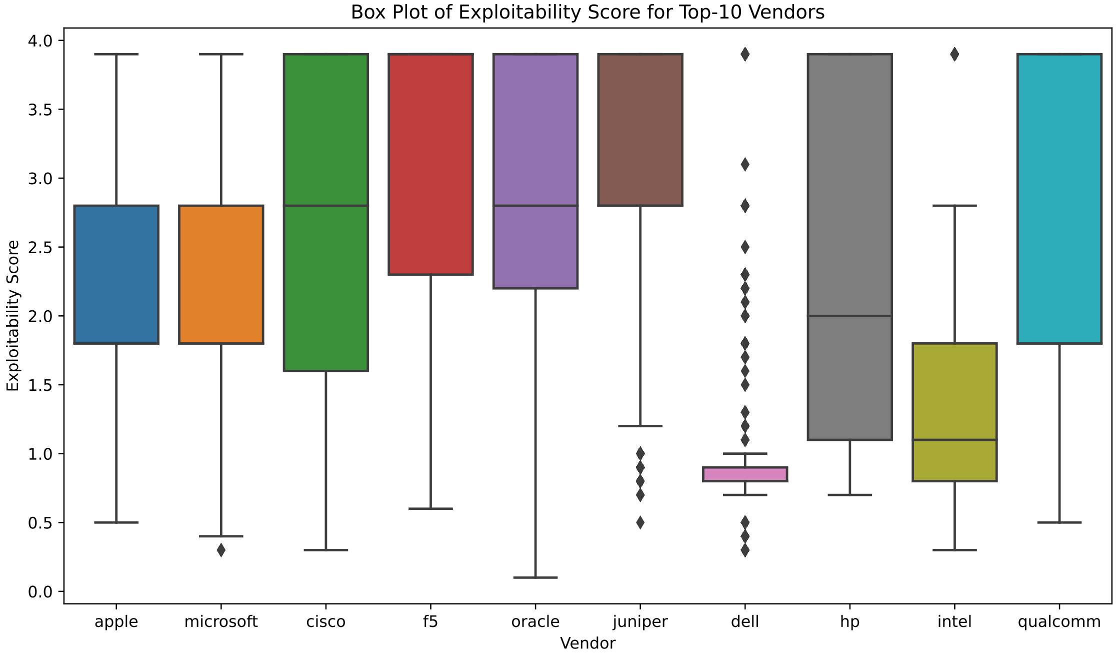


Figure 4.11: Box Plot of CVSS V3 Exploitability Score Values for Top-10 Vendors

The box plot 4.12 of impact scores for top 10 vendors illustrates a broad range of potential security impacts, with most vendors showing median impact scores around 4, indicating generally high potential impacts. Notably, Oracle stands out with a wide interquartile range, suggesting a significant variance in the impact of its vulnerabilities, including some with particularly high scores. In contrast, Juniper and f5 show more consistent, lower-range median scores, which implies a generally lower impact from their vulnerabilities. On the other hand, Dell and HP stand out with notably high median impact scores compared to other vendors, suggesting that vulnerabilities associated with their products tend to have a higher impact more consistently. Despite the overall trend of high impact scores across most vendors, the presence of outliers for almost 50% of the vendors underscores the existence of exceptional vulnerabilities that could have unusually high or low impacts

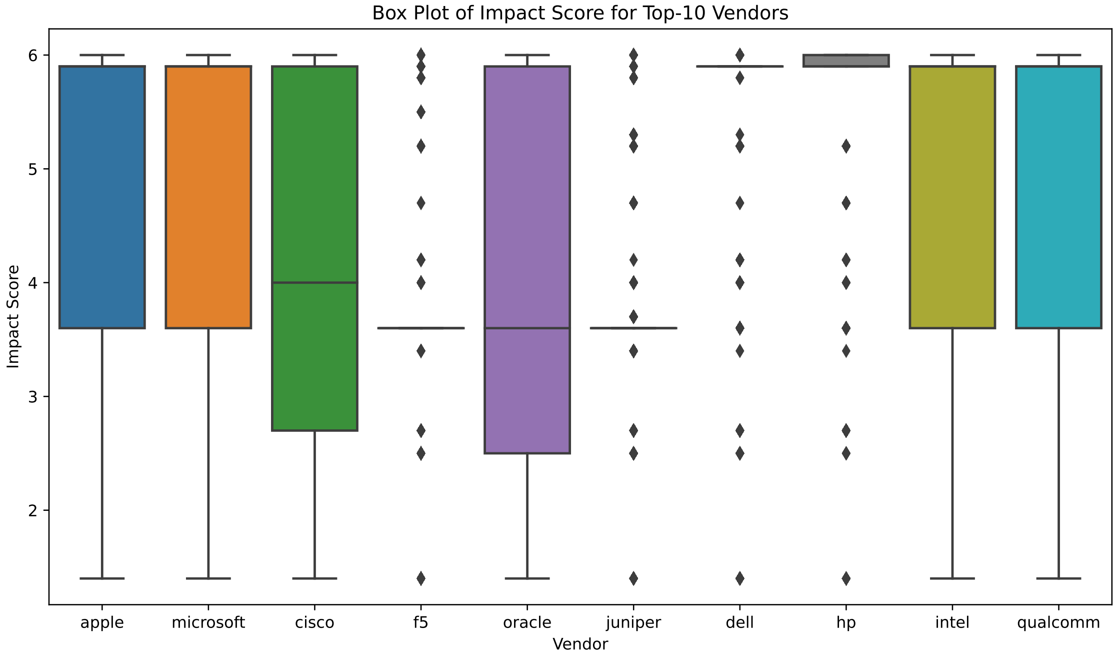


Figure 4.12: Box Plot of CVSS V3 Impact Score Values for Top-10 Vendors

4.2 Transformer

To identify the optimal model for transformer, a series of experiments were conducted. The initial strategy involved designing and training a deep learning model from scratch. As depicted in the accompanying figure 4.13, a BERT encoder and augmented it with additional layers to enhance the model's training process was utilized. Although this approach yielded acceptable results, there was a significant

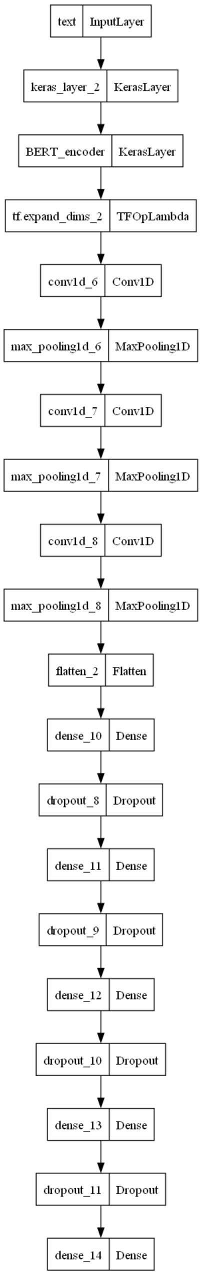


Figure 4.13: Unsuccessful Approach Structure

performance gap compared to other pre-trained models, such as DistilBERT. This discrepancy underscored the limitations of developing a deep learning model from the ground up without leveraging pre-existing structures.

Subsequently, the use of Google's Universal Sentence Encoder as an alternative for the transformer component was explored. This entailed designing the remaining parts of the architecture and conducting a comparative analysis with DistilBERT. Unfortunately, the gap in performance between this model and DistilBERT was substantial and deemed unacceptable. The comparative analysis clearly demonstrated the inferiority of the Universal Sentence Encoder in this context, especially when compared to DistilBERT.

It was also in alignment with findings from other researchers [79, 35, 84], who have successfully utilized DistilBERT and compared it to other's transformer models in their work. Taking these observations into account, DistilBERT was ultimately selected for the transformer component of the proposed pipeline. This decision was based on its superior performance and the widespread adoption and validation by the research community.

The two tables 4.1 and 4.2 provided compare the performance of eight models across eight CVSS metrics, evaluating both processed and unprocessed data under two different scenarios: using only CVE descriptions and using CVE descriptions along with a source identifier. This analysis reveals several critical insights into the behavior of these models under varying conditions.

Firstly, there is a clear consistency in how data processing affects model performance. Preprocessing tends to improve or at least maintain performance metrics across most categories, though the enhancements are typically modest. This suggests that the models are fundamentally robust, capable of performing well even with minimal preprocessing. The gains from preprocessing, while slight, indicate that even small optimizations in data handling can contribute to better model accuracy and reliability.

Secondly, the inclusion of the source identifier alongside CVE descriptions in the data (as shown in Table 4.2) results in slightly better performance metrics (Accuracy, F1 Score, and Balanced Accuracy) compared to using just CVE descriptions (as shown in Table 4.1). This improvement highlights the value of additional contextual information, which appears to aid the models in making more accurate predictions. The source identifier likely provides contextual cues that enhance the models' ability to discern subtleties in the data, which might be less discernible when relying solely on CVE descriptions.

Looking at specific metrics, Attack Complexity and Scope consistently exhibit high performance across both tables and processing conditions. These categories might benefit from clearer or more consistent descriptions within the CVE data, making them easier for models to predict accurately. Similarly, User Interaction and Integrity also show strong results, particularly with the inclusion of the source identifier. This suggests that these metrics, too, are well-served by the additional data, which may provide critical context that aids in prediction.

However, the Availability metric shows notable variation in performance, especially in Balanced Accuracy. This metric's Balanced Accuracy improves significantly with the addition of the source identifier, suggesting sensitivity to the quality and type of information used in training.

Overall, the analysis underscores the importance of integrating comprehensive data to improve prediction accuracy. While preprocessing shows limited but positive effects, the significant value added by source identifiers, suggests that for optimal model performance, leveraging as much relevant information as possible is beneficial. This approach not only enhances accuracy but also aids in achieving a more balanced performance across various assessment metrics.

Table 4.1: Model Result On Data Description Field

Category	Unprocessed Data			Preprocessed Data		
	Accuracy	F1	BA	Accuracy	F1	BA
Attack Vector	92.27	92.15	76.49	**92.33**	**92.18**	**77.06**
Attack Complexity	97.14	96.68	73.28	**97.17**	**96.90**	**76.14**
Privileges Required	84.85	84.75	**80.19**	**85.19**	**85.00**	80.03
User Interaction	94.24	94.21	93.11	**94.50**	**94.45**	**93.15**
Scope	96.73	96.66	92.40	**96.76**	**96.70**	**92.92**
Confidentiality	88.07	87.86	83.78	**88.20**	**88.11**	**84.25**
Integrity	88.93	88.90	87.55	**88.95**	**88.93**	**87.86**
Availability	90.51	89.99	67.79	**90.52**	**90.15**	**70.13**

Table 4.2: Model Result On Data Description + Source Identifier Fields

Category	Unprocessed Data			Preprocessed Data		
	Accuracy	F1	BA	Accuracy	F1	BA
Attack Vector	92.36	92.20	77.19	**92.40**	**92.34**	**79.93**
Attack complexity	97.16	96.77	74.56	**97.28**	**96.97**	**76.65**
Privileges Required	85.30	85.11	**80.44**	**85.53**	**85.32**	80.40
User Interaction	94.55	94.51	93.21	**94.70**	**94.68**	**93.75**
Scope	96.75	96.70	92.93	**96.77**	**96.72**	**92.95**
Confidentiality	88.25	88.20	84.97	**88.41**	**88.30**	**85.18**
Integrity	89.00	88.98	88.03	**89.10**	**89.09**	**88.14**
Availability	**90.54**	90.18	69.00	**90.54**	**90.27**	**71.64**

4.3 ANN

In this section, the utilization of the transformer's output as input for subsequent models was addressed. The primary goal is to ascertain the most efficacious model for processing this data. Initially, unsupervised learning techniques were employed, evaluating various machine learning models such as K-means, DBSCAN, and Birch for clustering. For instance, the K-means algorithm, when applied to the Scope Scenario with an expected two labels, incorrectly identified three to four classes as it shown in figure 4.14. Despite adjustments to constrain the clusters to two, the results were unsatisfactory, with a notable discrepancy between the accuracy of the K-means model (82.35%) and the transformer's accuracy (96.77%). The DBSCAN algorithm resulted in the excessive identification of 295 clusters, and although Birch achieved a higher accuracy of 96.40%, it still underperformed relative to the transformer and failed to provide additional insights.

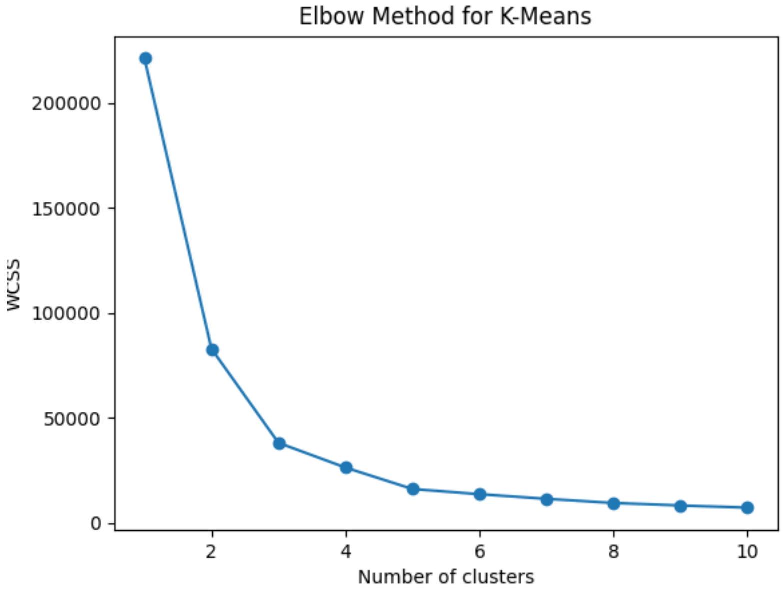


Figure 4.14: Elbow Method For K-Means For Scope Scenario

Shifting to supervised learning, traditional machine learning models such as Random Forest and SVM were utilized. These models demonstrated improved results compared to the unsupervised approaches, as evidenced by the data in Tables 4.3, 4.4, and 4.5. It became apparent that the most suitable model for the Attack Vector and Availability categories was Logistic Regression, while the LGBMClassifier proved optimal for Integrity. Based on Table 4.6, for the Privileges Required category, none of the models could surpass the transformer's performance.However, employing different models for each category posed significant challenges for system maintenance should they be deployed, thus prompting further exploration into alternative solutions.

The pivotal moment in the research came with the design and implementation of an Artificial Neural Network (ANN), illustrated in Figure 3.6. This ANN was developed through a process of meticulous experimentation. A key feature of the ANN was its ability to utilize the logits from the transformer model's output along with source identifiers as inputs. This strategic choice led to a simpler architectural design to prevent overfitting a common issue with deeper networks. The outcomes were remarkable. As shown in Table 4.7, the ANN not only outperformed the previous methods, including the transformer model, in terms of accuracy and F1 scores. This approach effectively balanced the model's depth with its performance, ensuring it remained robust without facing to overfitting,

Table 4.3: Evaluation Of Traditional Machine Learning for Attack Vector Use Case

Attack Vector		
Model	**Accuracy**	**F1 Score**
LogisticRegression	0.925392	0.924567
GradientBoostingClassifier	0.923819	0.923307
CatBoostClassifier	0.924416	0.923869
LGBMClassifier	0.923874	0.923279
KNeighborsClassifier	0.922952	0.922222
RandomForestClassifier	0.924850	0.924266
XGBClassifier	0.924036	0.923424
ExtraTreesClassifier	0.924741	0.924133
QuadraticDiscriminantAnalysis	0.921000	0.921299
GaussianNB	0.918668	0.919435
SGDClassifier	0.920295	0.920054
DecisionTreeClassifier	0.920837	0.920432
LinearDiscriminantAnalysis	0.920132	0.918263

Table 4.4: Evaluation Of Traditional Machine Learning for Availability Use Case

Availability		
Model	**Accuracy**	**F1 Score**
LogisticRegression	0.908095	0.905179
LGBMClassifier	0.907065	0.904437
GradientBoostingClassifier	0.906740	0.904063
XGBClassifier	0.906740	0.903948
CatBoostClassifier	0.906360	0.903700
AdaBoostClassifier	0.906143	0.903456
GaussianNB	0.905655	0.903560
LinearDiscriminantAnalysis	0.899691	0.899141
RandomForestClassifier	0.905926	0.903439
KNeighborsClassifier	0.901968	0.898829
ExtraTreesClassifier	0.902619	0.900425
QuadraticDiscriminantAnalysis	0.897793	0.899074
RidgeClassifier	0.900667	0.890357
SGDClassifier	0.903758	0.899394

Table 4.5: Evaluation Of Traditional Machine Learning for Integrity Use Case

Integrity		
Model	**Accuracy**	**F1 Score**
LGBMClassifier	0.892751	0.892522
LogisticRegression	0.892317	0.892118
CatBoostClassifier	0.892046	0.891840
XGBClassifier	0.891720	0.891519
GradientBoostingClassifier	0.891883	0.891729
GaussianNB	0.889660	0.889458
RandomForestClassifier	0.888847	0.888681
KNeighborsClassifier	0.888792	0.888593
QuadraticDiscriminantAnalysis	0.887057	0.887014
RidgeClassifier	0.889768	0.889292
ExtraTreesClassifier	0.887112	0.886945
LinearDiscriminantAnalysis	0.889172	0.888716
SGDClassifier	0.882666	0.882666
AdaBoostClassifier	0.889768	0.889633

Table 4.6: Evaluation Of Traditional Machine Learning for Privileges Required Use Case

Privileges Required		
Model	**Accuracy**	**F1 Score**
LGBMClassifier	0.848452	0.848007
CatBoostClassifier	0.848560	0.848163
LogisticRegression	0.849428	0.848864
XGBClassifier	0.847259	0.846962
GradientBoostingClassifier	0.847910	0.847635
RandomForestClassifier	0.844223	0.843928
QuadraticDiscriminantAnalysis	0.847096	0.846973
GaussianNB	0.841078	0.842507
KNeighborsClassifier	0.843030	0.843010
SGDClassifier	0.849374	0.848402
AdaBoostClassifier	0.832186	0.830092
ExtraTreesClassifier	0.843084	0.842942
LinearDiscriminantAnalysis	0.852139	0.849846
RidgeClassifier	0.851488	0.848768
DecisionTreeClassifier	0.831047	0.831169

Table 4.7: Evaluation Of ANN

Category	Transformer		ANN	
	Accuracy	F1	Accuracy	F1
Attack Vector	92.40	92.34	**92.57**	**92.48**
Attack complexity	97.28	96.97	**97.30**	**96.99**
Privileges Required	85.53	85.32	**85.58**	**85.33**
User Interaction	94.70	94.68	**94.76**	**94.74**
Scope	96.77	96.72	**96.81**	**96.75**
Confidentiality	88.41	88.30	**88.46**	**88.36**
Integrity	89.10	89.09	**89.31**	**89.29**
Availability	90.54	90.27	**90.90**	**90.60**

4.4 Comparison with other works

Base on Table 4.8, it is evident that both the transformer model and ANN outperformed other approaches in most metrics, with exceptions in the F1 score for Privileges Required and both accuracy and F1 score for Scope. However, on average, the ANN demonstrated superior performance compared to other models. Specifically, the ANN achieved the best results compared to the top-performing model among other works, with an increase in accuracy of 1.33% and an improvement in F1 score of 0.87%.

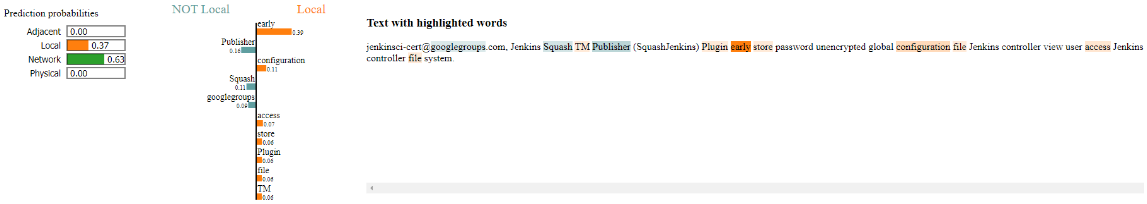
Table 4.8: A comparison with other works

Category	Transformer		ANN		Costa [35]	Shan [131]		Shahid [130]		Babalau [14]
	Accuracy	F1	Accuracy	F1	Accuracy	Accuracy	F1	Accuracy	F1	Accuracy
Attack Vector	92.40	92.34	**92.57**	**92.48**	91.41	91	92	91.15	90.89	70.48
Attack Complexity	97.28	96.97	**97.30**	**96.99**	95.2	95	95	96.07	95.74	95.21
Privileges Required	85.53	85.32	**85.58**	85.33	86.42	85	**86**	83.79	83.78	56.20
User Interaction	94.7	94.68	**94.76**	**94.74**	93.33	91	91	93.21	93.19	69.81
Scope	96.77	96.72	96.81	96.75	96.4	**97**	**97**	95.45	95.48	89.35
Confidentiality	88.41	88.3	**88.46**	**88.36**	86.71	87	88	87.04	86.81	66.44
Integrity	89.1	89.09	**89.31**	**89.29**	87.61	89	89	87.35	87.31	72.53
Availability	90.54	90.27	**90.9**	**90.6**	88.81	90	90	88.94	88.63	70.08
AVG	91.841	91.711	**91.961**	**91.875**	90.7363	90.625	91	90.375	90.2288	73.76

4.5 Significance Of SourceIdentifier

The positive impact of incorporating the source identifier is evident in the improved performance of the transformer model as can be seen in table 4.2. To emphasize the importance of including the source identifier, the Lime library to help clarify the essential aspects within the text was utilized. Lime is widely recognized for its ability to make sense of how models work, even for those not familiar with technical details.

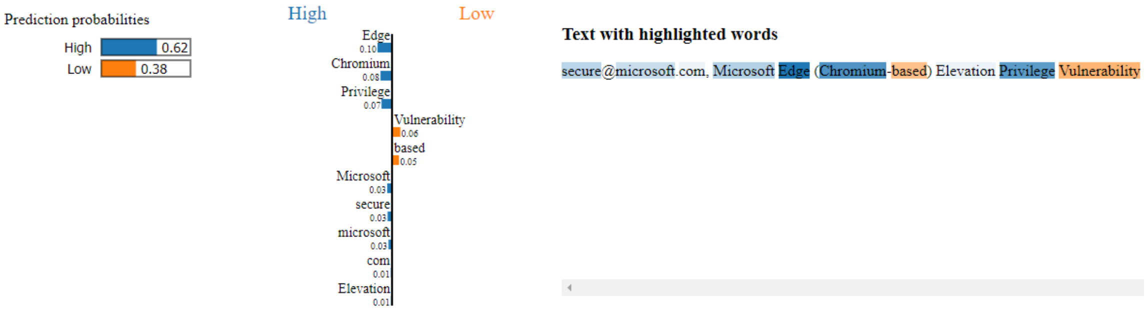
Using Lime, the analysis delved into the text to uncover the top-10 words that significantly impact the prediction of the CVSS Score. This analysis, which can be seen in figures 4.15 and 4.16, illuminates on how integrating the source identifier enriches the description, thereby enhancing the model's comprehension and predictive accuracy. The insights gleaned from Lime provide a clear visualization of the words that carry significant weight in the model's decision-making process. This not only underscores the necessity of including the source identifier but also furnishes valuable insights into the specific linguistic components contributing to the model's predictions. Ultimately, this effort helps improve the transparency and reliability of the predictive model, highlighting the practical significance of enriching the description with contextual information.



Text with highlighted words

jenkinsci-cert@googlegroups.com, Jenkins Squash TM Publisher (SquashJenkins) Plugin early store password unencrypted global configuration file Jenkins controller view user access Jenkins controller file system.

Figure 4.15: Lime For Attack Vector Scenario (CVE-2022-34213)



Text with highlighted words

secure@microsoft.com, Microsoft Edge (Chromium-based) Elevation Privilege Vulnerability

Figure 4.16: Lime For Attack Complexity Scenario (CVE-2022-26895)

Additionally, employing SHAP values for the ANN model reveals that the sourceIdentifier significantly influences the ANN model's decision-making process base on the figures 4.17, 4.18, and 4.19. The SHAP (SHapley Additive exPlanations) values provide a quantitative measure of feature importance, and in this context, highlight the substantial contribution of the source identifier to the ANN model's predictions. This dual confirmation underscores the practical significance of including the source of the identifier, showcasing its meaningful impact on both the transformer and ANN models. The integration of this contextual information not only enhances predictive accuracy but also contributes to a more comprehensive understanding of the models' behavior.

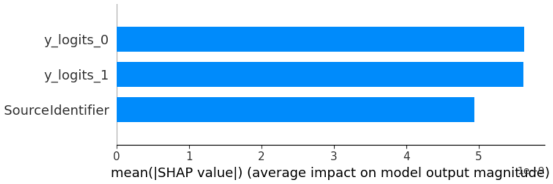


Figure 4.17: ANN SHAP Values For Scope Scenario

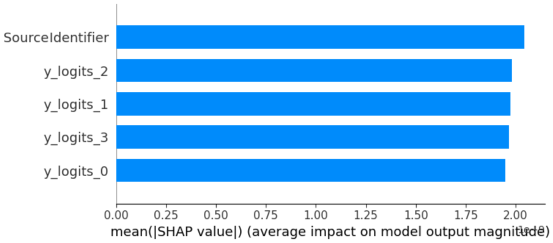


Figure 4.18: ANN SHAP Values For Attack Vector Scenario

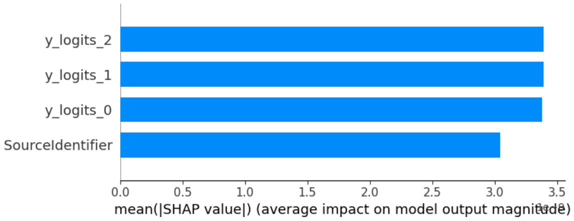


Figure 4.19: ANN SHAP Values For Confidentiality Impact Scenario

4.6 Incremental Learning

Base on the results in ANN section, it showed improvement in both accuracy and F1 score when an Artificial Neural Network (ANN) was added to the transformer model. After carefully reviewing these results, we realized that if the transformer model could incorporate the corrections made by the ANN, it might perform even better.

To put this idea into practice, we decided to update the transformer model through incremental learning. We tested various methods and found that lowering the learning rate to one-tenth of its original value was particularly effective. Additionally, selecting a random sample of 8,000 instances (one-tenth of the training dataset) from our existing training data and adding new data for fine-tuning seemed to best help the model adjust to the corrected information from the ANN. This process of fine-tuning was designed to enhance the model's performance.

To find this approach, first, three of the models using 8,000 samples from the training

Table 4.9: Initial Test Of The Incremental Learning Method

	Base Model (Testing)	Incremental Model (Testing)	Base Model (Training)	Incremental Model (Training)
Integrity	Acc: 89.09 F1: 89.07	Acc: 89.24 F1: 89.22	Acc: 95.23 F1: 95.22	Acc: 95.82 F1: 95.81
Attack Vector	Acc: 92.54 F1: 92.44	Acc: 92.59 F1: 92.49	Acc: 98.69 F1: 98.69	Acc: 98.97 F1: 98.97
User Interaction	Acc: 94.71 F1: 94.69	Acc: 94.83 F1: 94.81	Acc: 97.64 F1: 97.63	Acc: 98.01 F1: 98.07

data, added 100 data points from the test data were tested, and evaluated them based on the remaining test data. This result is shown in table 4.9.

The incremental learning method has demonstrated success, showcasing the ability of the new model to learn from data without compromising information from the training dataset. Subsequently, this effective technique was applied to the existing models. The approach involved selecting 8000 instances from the training data and incorporating additional data points where the ANN either corrected or incorrectly predicted the transformer's output. The results are presented in the following figures, illustrating the impact of incremental learning on the performance of the models and highlighting the effectiveness of this strategy in refining predictive capabilities.

Table 4.10: Evaluation Of the base and incremental models on the new data

Category	**Transformer**						**ANN**					
	Base			**Incremental**			**Base**			**Incremental**		
	Acc	**F1**	**BA**	**Acc**	**F1**	**BA**	**Acc**	**F1**	**BA**	**Acc**	**F1**	**BA**
Attack Vector	91.73	91.62	79.30	91.93	91.81	79.65	91.82	91.68	79.12	**92.08**	**91.94**	**79.68**
Attack Complexity	97.64	97.38	77.00	97.68	97.39	**77.29**	97.60	97.35	77.08	**97.71**	**97.42**	77.12
Privileges Required	84.99	84.62	79.63	85.09	84.65	**79.72**	84.96	84.58	79.52	**85.10**	**84.69**	79.64
User Interaction	94.62	94.61	93.96	94.70	94.70	**94.10**	94.67	94.65	93.88	**94.75**	**94.74**	94.07
Scope	97.05	97.02	94.46	**97.06**	**97.03**	**94.47**	97.05	97.02	94.46	**97.06**	**97.03**	**94.47**
Confidentiality	88.44	88.34	85.12	**88.68**	**88.60**	**85.70**	88.39	88.30	85.35	88.67	**88.60**	85.66
Integrity	88.97	88.97	88.56	88.99	**88.98**	**88.65**	89.04	89.02	88.30	**89.17**	88.91	88.33
Availability	90.26	89.94	66.87	90.40	90.149	**67.31**	90.34	90.00	66.90	**90.62**	**90.25**	66.25
AVG	91.71	91.56	83.11	91.81	91.66	**83.36**	91.73	91.57	83.07	**91.89**	**91.69**	83.15

Table 4.11: Evaluation Of the base and incremental models on the incremental data

Category	**Transformer**						**ANN**		
	Base			**Incremental**			**Incremental**		
	Acc	**F1**	**BA**	**Acc**	**F1**	**BA**	**Acc**	**F1**	**BA**
Attack Vector	92.94	92.86	80.37	92.94	92.87	80.65	**93**	**92.92**	**80.68**
Attack Complexity	97.30	97.00	76.84	**97.33**	**97.02**	76.86	97.32	97.00	**76.87**
Privileges Required	86.05	85.81	80.68	**86.18**	**85.91**	**80.89**	86.13	85.89	**80.89**
User Interaction	94.96	94.94	93.93	95.03	95.01	**94.11**	**95.07**	**95.05**	94.07
Scope	96.88	96.83	93.09	**96.92**	**96.97**	**93.18**	96.83	96.78	93.07
Confidentiality	88.80	88.70	85.75	88.78	88.65	85.82	**88.88**	**88.79**	**85.99**
Integrity	89.94	89.92	88.87	**90.04**	90.02	88.97	**90.04**	**90.03**	**89.02**
Availability	91.60	91.29	72.10	91.61	91.33	**72.46**	**91.68**	**91.34**	71.73
AVG	92.308	92.168	83.9537	92.353	92.222	**84.117**	**92.368**	**92.225**	84.04

The evaluation of the incremental models,as shown in tables 4.10 and 4.11 reveals promising results, demonstrating consistent improvement and out performance across all of the categories. Notably, the model exhibited commendable retention of information from the original training data, showcasing its robust learning capabilities. The obtained results underscore the commendable performance of the models, with the incremental model particularly demonstrating noteworthy enhancements. This effective use of new data and incremental datasets adds robustness to the assessment, reinforcing the reliability and generalizability of the models' predictive capabilities. Interestingly, the role of the ANN in the incremental model is nuanced, as observed through varied impacts, ranging from marginal improvements to slight hindrances in performance for the new data data. Given the incremental model's consistent overall superiority in performance, there arises a suggestion that the final model could potentially be exclusively based on the incremental transformer approach. Alternatively, a prudent strategy may involve selecting specific models based on their superior performance. This nuanced decision-making process ensures the selection of models that exhibit robust performance across different categories, contributing to a more comprehensive and reliable final model.

4.7 Concluding Remarks

In this chapter, the experimental results about the predictive model for CVSS using a combination of DistilBERT and ANN were comprehensively presented. Initially, the datasets employed for the experiments were introduced, followed by an in-depth analysis of the model's performance in various testing scenarios. The results clearly demonstrated the efficacy of the model in accurately predicting CVSS scores, emphasizing the significant improvements over others work.

Subsequently, the experiments highlighted the utility of incremental learning technique in enhancing the model's adaptability and accuracy in response to new data. This approach proved important in maintaining the model's effectiveness against the evolving landscape of cybersecurity threats. The critical role of source identifiers in improving accuracy, F1, and BA underscores the value of employing contextual information into the model.

Moreover, the application of interpretability tools such as SHAP and LIME provided valuable insights into the decision-making processes of the model. These tools facilitated a deeper understanding of how different features influenced the predictions, offering a clearer picture of the model's internal workings.

Chapter 5

Conclusion and Future Works

5.1 Conclusion

In conclusion, this comprehensive study proposed a novel transformer-based multi-step approach for predicting common vulnerability severity in the early stages of a CVE's publication, addressing the temporal gap before official scores are available. The importance of timely CVSS predictions for informed decision-making, proactive vulnerability management, and overall cybersecurity resilience was emphasized. The undertaken efforts involved a meticulous process, starting with data preparation from the National Vulnerability Database (NVD) and employing preprocessing techniques such as natural language processing (NLP). The proposed methodology contains two phases starts with utilizing DistilBERT, a BERT-based model, and an Artificial Neural Network (ANN) to predict CVSS scores across eight distinct categories. In the DistilBERT model, the output logits are raw scores for each class, which are not automatically processed through a softmax layer. These raw logits preserve the full range of model outputs, capturing detailed gradients in prediction scores that are crucial for nuanced analysis. This is especially beneficial when these logits are used as inputs for an ANN. To further enhance the model's contextual understanding, these logits are combined with the source identifier for each CVE as inputs to the ANN. This combination leverages both the detailed predictive insights from DistilBERT and contextual cues from the source identifiers, allowing the ANN to refine these predictions more effectively. This method ensures maximal information retention and utilizes both textual and contextual data, optimizing the prediction of CVSS scores.

The incorporation of the source identifier's contextual information was highlighted as a significant factor in enhancing model performance in terms of accuracy, F1, and BA. Interpretability exercises using Lime and SHAP values reinforced the impact of the source identifier on both transformer and ANN models. The results showcased improved accuracy and F1 scores when combining the transformer and ANN models. The models achieved an average accuracy of 91.961% in predicting CVSS category scores, with an average F1 of 91.875%. Specifically, the moedls achieved the better results compared to the top-performing models among other works, with an increase in accuracy of 1.33% and an improvement in F1 score of 0.87%.

After deploying the initial DistilBERT and ANN model, the second phase begins with instances where classification differences occur between the transformer and the ANN were analyzed. Specifically, focusing on cases where one model is correct, and the other is not. This analysis helps identify each model's learning weaknesses or biases. These misclassified instances are then used for incremental learning, allowing the transformer to adjust and improve its accuracy by learning from previous mistakes, which enhances the model's robustness. Incremental learning proved to be an effective strategy, demonstrating the model's ability to learn from new data without compromising information from the original training dataset. The evaluation of the incremental model across various categories demonstrated consistent improvements, adding robustness to the models' predictive capabilities. The incremental model demonstrated improved accuracy and F1 scores when evaluated with new data, indicating its enhanced capability to learn from previous errors and adapt. The nuanced role of the ANN in the incremental model suggested a potential shift towards exclusively relying on incremental learning for model development or adopting a selective approach based on superior performance across different metrics.

This study not only contributed valuable insights into early-stage CVSS predictions but also introduced an effective incremental learning approach, highlighting its potential to refine and optimize predictive capabilities. The results underscore the importance of contextual information, model interpretability, and strategic model development strategies in advancing the field of vulnerability prediction and cybersecurity resilience.

5.2 Future Work

The exploration of interpretability using Lime should not be limited to the base models but extended to the incremental models as well. Analyzing the top influential features highlighted by Lime in the context of incremental learning can provide valuable insights into how the models evolve and adapt with the introduction of new data. Understanding the interpretability of the incremental models is crucial for ensuring transparency and trust in their decision-making processes, especially as they continuously learn and improve over time. Additionally, there is an opportunity to explore the concept of model drift in the context of incremental learning. Model drift refers to the phenomenon where the performance of a machine learning model deteriorates over time due to changes in the underlying data distribution. Investigating the presence of model drift in the incremental models and developing strategies to detect and mitigate it would be a valuable avenue for future research. This exploration could involve monitoring the performance of the models over extended periods, assessing their adaptability to evolving data patterns, and implementing mechanisms to address potential drift-induced challenges.

Bibliography

[1] *Cvss v3.1 specification document.*

[2] *cwe.mitre.org/cwss.*

[3] *Exploit prediction scoring system (epss) www.first.org/epss/.*

[4] Jai Prakash Agarwal, *Transformer and natural language processing; a recent development.*, Tuijin Jishu/Journal of Propulsion Technology **44** (2023), no. 1, 140–143.

[5] Sabeen Ahmed, Ian E Nielsen, Aakash Tripathi, Shamoon Siddiqui, Ravi P Ramachandran, and Ghulam Rasool, *Transformers in time-series analysis: A tutorial*, Circuits, Systems, and Signal Processing **42** (2023), no. 12, 7433–7466.

[6] Junaid Akram and Ping Luo, *Sqvdt: A scalable quantitative vulnerability detection technique for source code security assessment*, Software: Practice and Experience **51** (2021), no. 2, 294–318.

[7] Roaa Aljuraid and Taghreed Justinia, *Classification of challenges and threats in healthcare cybersecurity: A systematic review.*, Studies in Health Technology and Informatics **295** (2022), 362–365.

[8] Luca Allodi, Sebastian Banescu, Henning Femmer, and Kristian Beckers, *Identifying relevant information cues for vulnerability assessment using cvss*, Pro-

ceedings of the Eighth ACM Conference on Data and Application Security and Privacy, 2018, pp. 119–126.

[9] Luca Allodi and Fabio Massacci, *A preliminary analysis of vulnerability scores for attacks in wild: The ekits and sym datasets*, Proceedings of the 2012 ACM Workshop on Building analysis datasets and gathering experience returns for security, 2012, pp. 17–24.

[10] ———, *Comparing vulnerability severity and exploits using case-control studies*, ACM Transactions on Information and System Security (TISSEC) **17** (2014), no. 1, 1–20.

[11] Fernando Alves, Ambrose Andongabo, Ilir Gashi, Pedro M Ferreira, and Alysson Bessani, *Follow the blue bird: A study on threat data published on twitter*, Computer Security–ESORICS 2020: 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, September 14–18, 2020, Proceedings, Part I 25, Springer, 2020, pp. 217–236.

[12] Marco Angelini, Graziano Blasilli, Tiziana Catarci, Simone Lenti, and Giuseppe Santucci, *Vulnus: Visual vulnerability analysis for network security*, IEEE transactions on visualization and computer graphics **25** (2018), no. 1, 183–192.

[13] Masaki Aota, Hideaki Kanehara, Masaki Kubo, Noboru Murata, Bo Sun, and Takeshi Takahashi, *Automation of vulnerability classification from its description using machine learning*, 2020 IEEE Symposium on Computers and Communications (ISCC), IEEE, 2020, pp. 1–7.

[14] Ion Babalau, Dragos Corlatescu, Octavian Grigorescu, Cristian Sandescu, and Mihai Dascalu, *Severity prediction of software vulnerabilities based on their*

text description, 2021 23rd International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), IEEE, 2021, pp. 171–177.

[15] Luca Bertolaccini, Piergiorgio Solli, Alessandro Pardolesi, and Antonello Pasini, *An overview of the use of artificial neural networks in lung cancer research*, Journal of thoracic disease **9** (2017), no. 4, 924.

[16] Guru Bhandari, Amara Naseer, and Leon Moonen, *Cvefixes: automated collection of vulnerabilities and their fixes from open-source software*, Proceedings of the 17th International Conference on Predictive Models and Data Analytics in Software Engineering, 2021, pp. 30–39.

[17] Narayan Bhusal, Mukesh Gautam, and Mohammed Benidris, *Cybersecurity of electric vehicle smart charging management systems*, 2020 52nd North American Power Symposium (NAPS), IEEE, 2021, pp. 1–6.

[18] Bandar Abdulrhman Bin Arfaj, Shailendra Mishra, and Mohammed Alshehri, *Efficacy of unconventional penetration testing practices.*, Intelligent Automation & Soft Computing **31** (2022), no. 1.

[19] Grzegorz J Blinowski and Paweł Piotrowski, *Cve based classification of vulnerable iot systems*, Theory and Applications of Dependable Computer Systems: Proceedings of the Fifteenth International Conference on Dependability of Computer Systems DepCoS-RELCOMEX, June 29–July 3, 2020, Brunów, Poland 15, Springer, 2020, pp. 82–93.

[20] Jose Camacho-Collados and Mohammad Taher Pilehvar, *On the role of text preprocessing in neural network architectures: An evaluation study on text categorization and sentiment analysis*, arXiv preprint arXiv:1707.01780 (2017).

[21] Marco Cantone, Claudio Marrocco, Francesco Tortorella, and Alessandro Bria, *Convolutional networks and transformers for mammography classification: an experimental study*, Sensors **23** (2023), no. 3, 1229.

[22] Melissa Carlton, Yair Levy, and Michelle M Ramim, *Validation of a vignettes-based, hands-on cybersecurity threats situational assessment tool*, Online Journal of Applied Knowledge Management (OJAKM) **6** (2018), no. 1, 107–118.

[23] VQ Carneiro, GN Silva, CD Cruz, PCS Carneiro, M Nascimento, and JES Carneiro, *Artificial neural networks as auxiliary tools for the improvement of bean plant architecture*, (2017).

[24] Francesco Caturano, Nicola d'Ambrosio, Gaetano Perrone, Luigi Previdente, and Simon Pietro Romano, *Exploitwp2docker: a platform for automating the generation of vulnerable wordpress environments for cyber ranges*, 2022 International Conference on Electrical, Computer and Energy Technologies (ICECET), IEEE, 2022, pp. 1–7.

[25] Christine P Chai, *Comparison of text preprocessing methods*, Natural Language Engineering **29** (2023), no. 3, 509–553.

[26] Soo-See Chai, Jeffrey P Walker, Oleg Makarynskyy, Michael Kuhn, Bert Veenendaal, and Geoff West, *Use of soil moisture variability in artificial neural network retrieval of soil moisture*, Remote Sensing **2** (2009), no. 1, 166–190.

[27] Alexis Challande, Robin David, and Guénaël Renault, *Building a commit-level dataset of real-world vulnerabilities*, Proceedings of the Twelfth ACM Conference on Data and Application Security and Privacy, 2022, pp. 101–106.

[28] Rajesh Chandarman and Brett Van Niekerk, *Students' cybersecurity awareness at a private tertiary educational institution*, The African Journal of Information and Communication **20** (2017), 133–155.

[29] Surajit Chattopadhyay and Goutami Chattopadhyay, *Identification of the best hidden layer size for three-layered neural net in predicting monsoon rainfall in india*, Journal of Hydroinformatics **10** (2008), no. 2, 181–188.

[30] Efstratios Chatzoglou, Georgios Kambourakis, and Vasileios Kouliaridis, *A multi-tier security analysis of official car management apps for android*, Future Internet **13** (2021), no. 3, 58.

[31] Gagandeep Chawla, Neeraj Sharma, and Narender Kumar Rawal, *Ivsv: An improved cvss base score mechanism with vulnerability type*, International Journal of Engineering and Advanced Technology **8** (2019), no. 6, 4946–4950.

[32] Haipeng Chen, Jing Liu, Rui Liu, Noseong Park, and VS Subrahmanian, *Vest: A system for vulnerability exploit scoring & timing.*, IJCAI, 2019, pp. 6503–6505.

[33] Yu Chen, Jieyu Zhao, and Qilu Qiu, *A transformer-based capsule network for 3d part–whole relationship learning*, Entropy **24** (2022), no. 5, 678.

[34] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al., *Rethinking attention with performers*, arXiv preprint arXiv:2009.14794 (2020).

[35] Joana Cabral Costa, Tiago Roxo, João BF Sequeiros, Hugo Proenca, and Pedro RM Inacio, *Predicting cvss metric via description interpretation*, IEEE Access **10** (2022), 59125–59134.

[36] W Alec Cram and John D'Arcy, *'what a waste of time': An examination of cybersecurity legitimacy*, Information Systems Journal **33** (2023), no. 6, 1396–1422.

[37] Roland Croft, M Ali Babar, and Li Li, *An investigation into inconsistency of software vulnerability severity across data sources*, 2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), IEEE, 2022, pp. 338–348.

[38] Ahmad Dahamsheh and Hafzullah Aksoy, *Artificial neural network models for forecasting intermittent monthly precipitation in arid regions*, Meteorological Applications: A journal of forecasting, practical applications, training techniques and modelling **16** (2009), no. 3, 325–337.

[39] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov, *Transformer-xl: Attentive language models beyond a fixed-length context*, arXiv preprint arXiv:1901.02860 (2019).

[40] Tobias Dam and Sebastian Neumaier, *Challenges of mapping vulnerabilities and exposures to open-source packages*, arXiv preprint arXiv:2206.14527 (2022).

[41] Paulius Danenas and Tomas Skersys, *Exploring natural language processing in model-to-model transformations*, IEEE Access **10** (2022), 116942–116958.

[42] Siddhartha Shankar Das, Edoardo Serra, Mahantesh Halappanavar, Alex Pothen, and Ehab Al-Shaer, *V2w-bert: A framework for effective hierarchical multiclass classification of software vulnerabilities*, 2021 IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA), IEEE, 2021, pp. 1–12.

[43] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, *Bert: Pre-training of deep bidirectional transformers for language understanding*, arXiv preprint arXiv:1810.04805 (2018).

[44] Kareem Mahmoud Diab, Jamie Deng, Yusen Wu, Yelena Yesha, Fernando Collado-Mesa, and Phuong Nguyen, *Natural language processing for breast imaging: a systematic review*, Diagnostics **13** (2023), no. 8, 1420.

[45] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al., *An image is worth 16x16 words: Transformers for image recognition at scale*, arXiv preprint arXiv:2010.11929 (2020).

[46] Clément Elbaz, Louis Rilling, and Christine Morin, *Fighting n-day vulnerabilities with automated cvss vector prediction at disclosure*, Proceedings of the 15th International Conference on Availability, Reliability and Security, 2020, pp. 1–10.

[47] Sarah Elder, Nusrat Zahan, Valeri Kozarev, Rui Shu, Tim Menzies, and Laurie Williams, *Structuring a comprehensive software security course around the owasp application security verification standard*, 2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET), IEEE, 2021, pp. 95–104.

[48] Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, et al., *Prottrans: Toward understanding the language of life through self-supervised learning*, IEEE transactions on pattern analysis and machine intelligence **44** (2021), no. 10, 7112–7127.

[49] Simon Yusuf Enoch, Mengmeng Ge, Jin B Hong, Hani Alzaid, and Dong Seong Kim, *A systematic evaluation of cybersecurity metrics for dynamic networks*, Computer Networks **144** (2018), 216–229.

[50] Hanane Ezzikouri, Mohamed Erritali, and Mohamed Oukessou, *Fuzzy-semantic similarity for automatic multilingual plagiarism detection*, Int. J. Adv. Comput. Sci. Appl **8** (2017), no. 9, 86–90.

[51] Keyur Faldu, Amit Sheth, Prashant Kikani, and Hemang Akbari, *Ki-bert: Infusing knowledge context for better language and domain understanding*, arXiv preprint arXiv:2104.08145 (2021).

[52] Ummuhan Basaran Filik et al., *A new hybrid approach for wind speed prediction using fast block least mean square algorithm and artificial neural network*, Mathematical Problems in Engineering **2016** (2016).

[53] Sławomir Francik and Sławomir Kurpaska, *The use of artificial neural networks for forecasting of air temperature inside a heated foil tunnel*, Sensors **20** (2020), no. 3, 652.

[54] Tiberiu-Marian Georgescu, Bogdan Iancu, and Madalina Zurini, *Named-entity-recognition-based automated system for diagnosing cybersecurity situations in iot networks*, Sensors **19** (2019), no. 15, 3380.

[55] Sasan Golnaraghi, Zahra Zangenehmadar, Osama Moselhi, Sabah Alkass, et al., *Application of artificial neural network (s) in predicting formwork labour productivity*, Advances in Civil Engineering **2019** (2019).

[56] Danielle Gonzalez, Holly Hastings, and Mehdi Mirakhorli, *Automated characterization of software vulnerabilities*, 2019 IEEE International Conference on Software Maintenance and Evolution (ICSME), IEEE, 2019, pp. 135–139.

[57] Lawrence A Gordon, Martin P Loeb, William Lucyshyn, and Lei Zhou, *Empirical evidence on the determinants of cybersecurity investments in private sector firms*, Journal of Information Security **9** (2018), no. 2, 133–153.

[58] Oscar M Guillen, Ralf Brederlow, Ralph Ledwa, and Georg Sigl, *Risk management in embedded devices using metering applications as example*, Proceedings of the 9th Workshop on Embedded Systems Security, 2014, pp. 1–9.

[59] Hao Guo, Sen Chen, Zhenchang Xing, Xiaohong Li, Yude Bai, and Jiamou Sun, *Detecting and augmenting missing key aspects in vulnerability descriptions*, ACM Transactions on Software Engineering and Methodology (TOSEM) **31** (2022), no. 3, 1–27.

[60] JAQUELINE HANS and ROMAN BRANDTWEINER, *Best practices for vulnerability management in large enterprises: A critical view on the common vulnerability scoring system*, Risk Analysis, Hazard Mitigation and Safety and Security Engineering XIII **214** (2022), 123.

[61] Wu He, Ivan Ash, Mohd Anwar, Ling Li, Xiaohong Yuan, Li Xu, and Xin Tian, *Improving employees' intellectual capacity for cybersecurity through evidence-based malware training*, Journal of intellectual capital **21** (2020), no. 2, 203–213.

[62] Miguel Hernández and Miguel Hernández, *Are vulnerability scores misleading you? understanding cvss score*, April 2023.

[63] Tim Hill, Leorey Marquez, Marcus O'Connor, and William Remus, *Artificial neural network models for forecasting and decision making*, International journal of forecasting **10** (1994), no. 1, 5–15.

[64] Sepp Hochreiter and Jürgen Schmidhuber, *Long short-term memory*, Neural computation **9** (1997), no. 8, 1735–1780.

[65] Hannes Holm and Khalid Khan Afridi, *An expert-based investigation of the common vulnerability scoring system*, Computers & Security **53** (2015), 18–30.

[66] Hyunji Hong, Seunghoon Woo, Eunjin Choi, Jihyun Choi, and Heejo Lee, *xvdb: A high-coverage approach for constructing a vulnerability database*, IEEE Access **10** (2022), 85050–85063.

[67] Mohammad Shamsul Hoque, Norziana Jamil, Nowshad Amin, and Kwok-Yan Lam, *An improved vulnerability exploitation prediction model with novel cost function and custom trained word vector embedding*, Sensors **21** (2021), no. 12, 4220.

[68] Yunfei Hou, Kimberly Collins, and Montgomery Van Wart, *Intersection management, cybersecurity, and local government: Its applications, critical issues, and regulatory schemes*, Smart Mobility-Recent Advances, New Perspectives and Applications, IntechOpen, 2022.

[69] Jiajun Hu, Xiaobing Sun, David Lo, and Bin Li, *Modeling the evolution of development topics using dynamic topic models*, 2015 IEEE 22nd international conference on software analysis, evolution, and reengineering (SANER), IEEE, 2015, pp. 3–12.

[70] Maliheh Izadi, Roberta Gismondi, and Georgios Gousios, *Codefill: Multi-token code completion by jointly learning from structure and naming sequences*, Proceedings of the 44th International Conference on Software Engineering, 2022, pp. 401–412.

[71] Raheleh Jafari, Miguel A Contreras, Wen Yu, and Alexander Gegov, *Applications of fuzzy logic, artificial neural network and neuro-fuzzy in industrial engineering*, Industrial and Robotic Systems: LASIRS 2019, Springer, 2020, pp. 9–14.

[72] Mohammad S Jalali, Michael Siegel, and Stuart Madnick, *Decision-making and biases in cybersecurity capability development: Evidence from a simulation*

game experiment, The Journal of Strategic Information Systems **28** (2019), no. 1, 66–82.

[73] Yuning Jiang and Yacine Atif, *Towards automatic discovery and assessment of vulnerability severity in cyber–physical systems*, Array **15** (2022), 100209.

[74] Yuning Jiang, Yacine Atif, Jianguo Ding, and Wei Wang, *A semantic framework with humans in the loop for vulnerability-assessment in cyber-physical production systems*, International Conference on Risks and Security of Internet and Systems, Springer, 2019, pp. 128–143.

[75] Yuning Jiang, Manfred Jeusfeld, and Jianguo Ding, *Evaluating the data inconsistency of open-source vulnerability repositories*, Proceedings of the 16th International Conference on Availability, Reliability and Security, 2021, pp. 1–10.

[76] Dejiang Jing, *Improvement of vulnerable code dataset based on program equivalence transformation*, Journal of Physics: Conference Series, vol. 2363, IOP Publishing, 2022, p. 012010.

[77] Raunak Joshi and Abhishek Gupta, *Performance comparison of simple transformer and res-cnn-bilstm for cyberbullying classification*, arXiv preprint arXiv:2206.02206 (2022).

[78] Jeesoo Jurn, Taeeun Kim, and Hwankuk Kim, *An automated vulnerability detection and remediation method for software security*, Sustainability **10** (2018), no. 5, 1652.

[79] Shaofeng Kai, Fan Shi, Jinghua Zheng, et al., *Vuldistilbert: A cps vulnerability severity prediction method based on distillation model*, Security and Communication Networks **2023** (2023).

[80] Kittipong Kasantikul, Dongkai Yang, Qiang Wang, and Aung Lwin, *A novel wind speed estimation based on the integration of an artificial neural network and a particle filter using beidou geo reflectometry*, Sensors **18** (2018), no. 10, 3350.

[81] Muhamet Kastrati and Marenglen Biba, *Natural language processing for albanian: a state-of-the-art survey*, Int. J. Electr. Comput. Eng.(IJECE) **12** (2022), no. 6, 6432.

[82] Tai-hoon Kim, *A study on the influence of artificial intelligence research on the development of information security research.*, Asia-Pacific Journal of Convergent Research Interchange **7** (2021), no. 12, 41–53.

[83] Ravdeep Kour, Mustafa Aljumaili, Ramin Karim, and Phillip Tretten, *emaintenance in railways: Issues and challenges in cybersecurity*, Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit **233** (2019), no. 10, 1012–1022.

[84] Philipp Kühn, David N Relke, and Christian Reuter, *Common vulnerability scoring system prediction based on open source intelligence information sources*, Computers & Security **131** (2023), 103286.

[85] Aditya Kurniawan, Mohamad Yusof Darus, Muhammad Azizi Mohd Ariffin, Yohan Muliono, and Chrisando Ryan Pardomuan, *Automation of quantifying security risk level on injection attacks based on common vulnerability scoring system metric.*, Pertanika Journal of Science & Technology **31** (2023), no. 3.

[86] Triet HM Le, Huaming Chen, and M Ali Babar, *A survey on data-driven software vulnerability assessment and prioritization*, ACM Computing Surveys **55** (2022), no. 5, 1–39.

[87] Ching-Huang Lin, Chih-Hao Chen, and Chi-Sung Laih, *A study and implementation of vulnerability assessment and misconfiguration detection*, 2008 IEEE Asia-Pacific Services Computing Conference, IEEE, 2008, pp. 1252–1257.

[88] Aravind Machiry, Nilo Redini, Eric Camellini, Christopher Kruegel, and Giovanni Vigna, *Spider: Enabling fast patch propagation in related software repositories*, 2020 IEEE Symposium on Security and Privacy (SP), IEEE, 2020, pp. 1562–1579.

[89] Lyudmila V Massel, Olga M Gerget, Aleksei G Massel, and Timur G Mamedov, *The use of machine learning in situational management in relation to the tasks of the power industry*, EPJ Web of Conferences, vol. 217, EDP Sciences, 2019, p. 01010.

[90] Richard G Mathieu and Alan E Turovlin, *Lost in the middle–a pragmatic approach for erp managers to prioritize known vulnerabilities by applying classification and regression trees (cart)*, Information & Computer Security **31** (2023), no. 5, 655–674.

[91] Phayung Meesad, *Thai fake news detection based on information retrieval, natural language processing and machine learning*, SN Computer Science **2** (2021), no. 6, 425.

[92] Peter Mell and Karen Scarfone, *Improving the common vulnerability scoring system*, IET Information Security **1** (2007), no. 3, 119–127.

[93] Peter Mell, Karen Scarfone, and Sasha Romanosky, *Common vulnerability scoring system*, IEEE Security & Privacy **4** (2006), no. 6, 85–89.

[94] Shailendra Mishra, Majed A Alowaidi, and Sunil Kumar Sharma, *Impact of security standards and policies on the credibility of e-government*, Journal of Ambient Intelligence and Humanized Computing (2021), 1–12.

[95] Yazdan Movahedi, Michel Cukier, and Ilir Gashi, *Vulnerability prediction capability: A comparison between vulnerability discovery models and neural network models*, Computers & Security **87** (2019), 101596.

[96] ______, *Predicting the discovery pattern of publically known exploited vulnerabilities*, IEEE Transactions on Dependable and Secure Computing **19** (2020), no. 2, 1181–1193.

[97] Cherubin Mugisha and Incheon Paik, *Comparison of neural language modeling pipelines for outcome prediction from unstructured medical text notes*, IEEE Access **10** (2022), 16489–16498.

[98] Ricardo Neisse, José L Hernández-Ramos, Sara N Matheu-Garcia, Gianmarco Baldini, Antonio Skarmeta, Vasilios Siris, Dmitrij Lagutin, and Pekka Nikander, *An interledger blockchain platform for cross-border management of cybersecurity information*, IEEE Internet Computing **24** (2020), no. 3, 19–29.

[99] Stephan Neuhaus and Thomas Zimmermann, *Security trend analysis with cve topic models*, 2010 IEEE 21st International Symposium on Software Reliability Engineering, IEEE, 2010, pp. 111–120.

[100] Haitham Nobanee, Ahmad Alodat, Reem Bajodah, Maryam Al-Ali, and Alyazia Al Darmaki, *Bibliometric analysis of cybercrime and cybersecurity risks literature*, Journal of Financial Crime **30** (2023), no. 6, 1736–1754.

[101] Saahil Ognawala, Ricardo Nales Amato, Alexander Pretschner, and Pooja Kulkarni, *Automatically assessing vulnerabilities discovered by compositional analysis*, Proceedings of the 1st International Workshop on Machine Learning and Software Engineering in Symbiosis, 2018, pp. 16–25.

[102] Ahmet Okutan and Mehdi Mirakhorli, *Predicting the severity and exploitability of vulnerability reports using convolutional neural nets*, Proceedings of the 3rd

International Workshop on Engineering and Cybersecurity of Critical Systems, 2022, pp. 1–8.

[103] Ayorinde Tayo Olanipekun, Peter Madindwa Mashinini, Oluwakemi Adejoke Owojaiye, and Nthabiseng Beauty Maledi, *Applying a neural network-based machine learning to laser-welded spark plasma sintered steel: Predicting vickers micro-hardness*, Journal of Manufacturing and Materials Processing **6** (2022), no. 5, 91.

[104] Thomas Olsson, Martin Hell, Martin Höst, Ulrik Franke, and Markus Borg, *Sharing of vulnerability information among companies–a survey of swedish companies*, 2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), IEEE, 2019, pp. 284–291.

[105] Paschalis Panteleris and Antonis Argyros, *Pe-former: Pose estimation transformer*, International Conference on Pattern Recognition and Artificial Intelligence, Springer, 2022, pp. 3–14.

[106] George Parapuram, Mehdi Mokhtari, and Jalel Ben Hmida, *An artificially intelligent technique to generate synthetic geomechanical well logs for the bakken formation*, Energies **11** (2018), no. 3, 680.

[107] N Patwardhan, S Marrone, and C Sansone, *Transformers in the real world: A survey on nlp applications. information, 14 (4), 242*, 2023.

[108] Rajshakhar Paul, Asif Kamal Turzo, and Amiangshu Bosu, *Why security defects go unnoticed during code reviews? a case-control study of the chromium os project*, 2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE), IEEE, 2021, pp. 1373–1385.

[109] Vítor Pedreira, Daniel Barros, and Pedro Pinto, *A review of attacks, vulnerabilities, and defenses in industry 4.0 with new challenges on data sovereignty ahead*, Sensors **21** (2021), no. 15, 5189.

[110] Aritran Piplai, Sudip Mittal, Anupam Joshi, Tim Finin, James Holt, and Richard Zak, *Creating cybersecurity knowledge graphs from malware after action reports*, IEEE Access **8** (2020), 211691–211703.

[111] Alessandro Pollini, Tiziana C Callari, Alessandra Tedeschi, Daniele Ruscio, Luca Save, Franco Chiarugi, and Davide Guerri, *Leveraging human factors in cybersecurity: an integrated methodological approach*, Cognition, Technology & Work **24** (2022), no. 2, 371–390.

[112] Bernardi Pranggono and Abdullahi Arabo, *Covid-19 pandemic cybersecurity issues*, Internet Technology Letters **4** (2021), no. 2, e247.

[113] Stephen Quinn, Stephen Quinn, Nahla Ivy, Matthew Barrett, Larry Feldman, Greg Witte, and RK Gardner, *Identifying and estimating cybersecurity risk for enterprise risk management*, US Department of Commerce, National Institute of Standards and Technology, 2021.

[114] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al., *Improving language understanding by generative pre-training*, (2018).

[115] Md Akizur Rahman, Ravie Chandren Muniyandi, Kh Tohidul Islam, and Md Mokhlesur Rahman, *Ovarian cancer classification accuracy analysis using 15-neuron artificial neural networks model*, 2019 IEEE Student Conference on Research and Development (SCOReD), IEEE, 2019, pp. 33–38.

[116] Muhammad Asif Zahoor Raja, Junaid Ali Khan, and Ijaz Mansoor Qureshi, *A new stochastic approach for solution of riccati differential equation of fractional order*, Annals of Mathematics and Artificial Intelligence **60** (2010), 229–250.

[117] Frank Rosenblatt, *The perceptron: a probabilistic model for information storage and organization in the brain.*, Psychological review **65** (1958), no. 6, 386.

[118] Jukka Ruohonen, *A look at the time delays in cvss vulnerability scoring*, Applied Computing and Informatics **15** (2019), no. 2, 129–135.

[119] Jukka Ruohonen, Sampsa Rauti, Sami Hyrynsalmi, and Ville Leppänen, *A case study on software vulnerability coordination*, Information and Software Technology **103** (2018), 239–257.

[120] Pintu Kumar Sadhu, Venkata P Yanambaka, Ahmed Abdelgawad, and Kumar Yelamarthi, *Prospect of internet of medical things: A review on security requirements and solutions*, Sensors **22** (2022), no. 15, 5517.

[121] Sagar Samtani, Murat Kantarcioglu, and Hsinchun Chen, *Trailblazing the artificial intelligence for cybersecurity discipline: A multi-disciplinary research roadmap*, 2020, pp. 1–19.

[122] Pattaraporn Sangaroonsilp, Hoa Khanh Dam, and Aditya Ghose, *Common privacy weaknesses and vulnerabilities in software applications*, Available at SSRN 4025928 (2021).

[123] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf, *Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter*, arXiv preprint arXiv:1910.01108 (2019).

[124] Pradeepta Kumar Sarangi, Ashok Kumar Sahoo, and Sachin Sinha, *Modeling consumer price index: A machine learning approach*, macromolecular symposia, vol. 401, Wiley Online Library, 2022, p. 2100349.

[125] Iqbal H Sarker, Md Hasan Furhad, and Raza Nowrozy, *Ai-driven cybersecurity: an overview, security intelligence modeling and research directions*, SN Computer Science **2** (2021), no. 3, 173.

[126] Satish1v, *Tokenization for bert models - satish1v - medium*, Medium (2022).

[127] Katherine Seale, Jeffrey McDonald, William Glisson, Harold Pardue, and Michael Jacobs, *Meddevrisk: Risk analysis methodology for networked medical devices*, (2018).

[128] Khaled Shaalan and Mai Oudah, *A hybrid approach to arabic named entity recognition*, Journal of Information Science **40** (2014), no. 1, 67–87.

[129] Aamir Shahab, Mamdouh Alenezi, Muhammad Nadeem, and Raja Asif, *An automated approach to fix buffer overflows.*, International Journal of Electrical & Computer Engineering (2088-8708) **10** (2020), no. 4.

[130] Mustafizur R Shahid and Hervé Debar, *Cvss-bert: Explainable natural language processing to determine the severity of a computer security vulnerability from its description*, 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA), IEEE, 2021, pp. 1600–1607.

[131] Chun Shan, Ziyi Zhang, and Siyi Zhou, *A multi-task deep learning based vulnerability severity prediction method*, 2023 IEEE 12th International Conference on Cloud Networking (CloudNet), IEEE, 2023, pp. 307–315.

[132] Gaurav Sharma, Stilianos Vidalis, Catherine Menon, and Niharika Anand, *Analysis and implementation of semi-automatic model for vulnerability exploitations of threat agents in nist databases*, Multimedia Tools and Applications **82** (2023), no. 11, 16951–16971.

[133] Sagar Sharma, *What the hell is perceptron? - towards data science*, Medium (2019).

[134] Rohan Singh, Harish Chandra Arora, Alireza Bahrami, Aman Kumar, Nishant Raj Kapoor, Krishna Kumar, and Hardeep Singh Rai, *Enhancing sus-*

tainability of corroded rc structures: Estimating steel-to-concrete bond strength with ann and svm algorithms, Materials **15** (2022), no. 23, 8295.

[135] Adewale Daniel Sontan and Segun Victor Samuel, *The intersection of artificial intelligence and cybersecurity: Challenges and opportunities*, World Journal of Advanced Research and Reviews **21** (2024), no. 2, 1720–1736.

[136] Octavian Suciu, Connor Nelson, Zhuoer Lyu, Tiffany Bao, and Tudor Dumitraș, *Expected exploitability: Predicting the development of functional vulnerability exploits*, 31st USENIX Security Symposium (USENIX Security 22), 2022, pp. 377–394.

[137] KP Sudheer, PC Nayak, and KS Ramasastri, *Improving peak flow estimates in artificial neural network river flow models*, Hydrological Processes **17** (2003), no. 3, 677–686.

[138] Xiaobing Sun, Lili Li, Lili Bo, Xiaoxue Wu, Ying Wei, and Bin Li, *Automatic software vulnerability classification by extracting vulnerability triggers*, Journal of Software: Evolution and Process **36** (2024), no. 2, e2508.

[139] Ayisha Tabassum and Rajendra R Patil, *A survey on text pre-processing & feature extraction techniques in natural language processing*, International Research Journal of Engineering and Technology (IRJET) **7** (2020), no. 06, 4864–4867.

[140] Yiting Tao, Miaozhong Xu, Yanfei Zhong, and Yufeng Cheng, *Gan-assisted two-stream neural network for high-resolution remote sensing image classification*, Remote Sensing **9** (2017), no. 12, 1328.

[141] Salimkan Fatma TAŞKIRAN and KAYA Ersin, *Academic text clustering using natural language processing*, Konya Journal of Engineering Sciences **10** (2022), 41–51.

[142] Nazgol Tavabi, Palash Goyal, Mohammed Almukaynizi, Paulo Shakarian, and Kristina Lerman, *Darkembed: Exploit prediction with neural language models*, Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, 2018.

[143] Godwin Thomas and Mary-Jane Sule, *A service lens on cybersecurity continuity and management for organizations' subsistence and growth*, Organizational Cybersecurity Journal: Practice, Process and People **3** (2023), no. 1, 18–40.

[144] Dimitrios Toloudis, Georgios Spanos, and Lefteris Angelis, *Associating the severity of vulnerabilities with their description*, Advanced Information Systems Engineering Workshops: CAiSE 2016 International Workshops, Ljubljana, Slovenia, June 13-17, 2016, Proceedings 28, Springer, 2016, pp. 231–242.

[145] Ekincan Ufuktepe, Tugkan Tuglular, and Kannappan Palaniappan, *Tracking code bug fix ripple effects based on change patterns using markov chain models*, IEEE Transactions on Reliability **71** (2022), no. 2, 1141–1156.

[146] Maureen Van Devender and Jeffrey Todd McDonald, *A quantitative risk assessment framework for the cybersecurity of networked medical devices*, International Conference on Cyber Warfare and Security, vol. 18, 2023, pp. 402–411.

[147] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin, *Attention is all you need*, Advances in neural information processing systems **30** (2017).

[148] Jesse Vig, Ali Madani, Lav R Varshney, Caiming Xiong, Richard Socher, and Nazneen Fatema Rajani, *Bertology meets biology: interpreting attention in protein language models*, arXiv preprint arXiv:2006.15222 (2020).

[149] PR Vishnu, P Vinod, and Suleiman Y Yerima, *A deep learning approach for classifying vulnerability descriptions using self attention based neural network*, Journal of Network and Systems Management **30** (2022), no. 1, 9.

[150] Michał Walkowski, Jacek Oko, and Sławomir Sujecki, *Vulnerability management models using a common vulnerability scoring system*, Applied Sciences **11** (2021), no. 18, 8735.

[151] Hanrui Wang, Zhanghao Wu, Zhijian Liu, Han Cai, Ligeng Zhu, Chuang Gan, and Song Han, *Hat: Hardware-aware transformers for efficient natural language processing*, arXiv preprint arXiv:2005.14187 (2020).

[152] Ju An Wang, Hao Wang, Minzhe Guo, and Min Xia, *Security metrics for software systems*, Proceedings of the 47th Annual Southeast Regional Conference, 2009, pp. 1–6.

[153] Xiajing Wang, Rui Ma, Binbin Li, Donghai Tian, and Xuefei Wang, *E-wbm: An effort-based vulnerability discovery model*, IEEE Access **7** (2019), 44276–44292.

[154] Honghao Wu, Junyong Liu, Jichun Liu, Mingjian Cui, Xuan Liu, and Hongjun Gao, *Power grid reliability evaluation considering wind farm cyber security and ramping events*, Applied Sciences **9** (2019), no. 15, 3003.

[155] Yong Xie, Yu Zhou, Jing Xu, Jian Zhou, Xiaobai Chen, and Fu Xiao, *Cybersecurity protection on in-vehicle networks for distributed automotive cyber-physical systems: state-of-the-art and future challenges*, Software: Practice and Experience **51** (2021), no. 11, 2108–2127.

[156] Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do, and Kaori Togashi, *Convolutional neural networks: an overview and application in radiology*, Insights into imaging **9** (2018), 611–629.

[157] Xiao Yi, Daoyuan Wu, Lingxiao Jiang, Kehuan Zhang, and Wei Zhang, *Diving into blockchain's weaknesses: An empirical study of blockchain system vulnerabilities*, arXiv preprint arXiv:2110.12162 (2021).

[158] Azlan Mohd Zain, Habibollah Haron, and Safian Sharif, *Application of regression and ann techniques for modeling of the surface roughness in end milling machining process*, 2009 Third Asia International Conference on Modelling & Simulation, IEEE, 2009, pp. 188–193.

[159] Piotr Żebrowski, Aitor Couce-Vieira, and Alessandro Mancuso, *A bayesian framework for the analysis and optimal mitigation of cyber threats to cyber-physical systems*, Risk Analysis **42** (2022), no. 10, 2275–2290.

[160] Jiangshe Zhang and Weifu Ding, *Prediction of air pollutants concentration based on an extreme learning machine: the case of hong kong*, International journal of environmental research and public health **14** (2017), no. 2, 114.

[161] Xiang Zhang, Zichun Zhou, Chen Ming, and Yi-Yang Sun, *Gpt-assisted learning of structure–property relationships by graph neural networks: Application to rare-earth-doped phosphors*, The Journal of Physical Chemistry Letters **14** (2023), no. 50, 11342–11349.

[162] Lina Zhou, Dongsong Zhang, Christopher C Yang, and Yu Wang, *Harnessing social media for health information management*, Electronic commerce research and applications **27** (2018), 139–151.

Appendix A

Source Code

```python
import re  # Regular expressions library for text manipulation
# nltk library import for natural language processing
from nltk import pos_tag  # For part-of-speech tagging
from nltk.corpus import wordnet  # For wordnet, used in POS tagging
from nltk.corpus import stopwords  # For filtering out stop words
from nltk.stem import WordNetLemmatizer  # For word lemmatization

# Function to map NLTK’s part-of-speech tags to the format wordnet lemmatizer understands
def get_wordnet_pos(treebank_tag):
    if treebank_tag.startswith(’J’):  # If adjective
        return wordnet.ADJ
    elif treebank_tag.startswith(’V’):  # If verb
        return wordnet.VERB
    elif treebank_tag.startswith(’N’):  # If noun
        return wordnet.NOUN
    elif treebank_tag.startswith(’R’):  # If adverb
        return wordnet.ADV
    else:
        return wordnet.NOUN  # Default to noun if other POS tags

```

```
def preprocess(text):
    text = re.sub(r’(?<=\d),(?=\s*)’, ’’, text)  # Remove commas that are directly after numbers and before optional spaces

    text = re.sub(r’\d+(\.\d+)*’, ’’, text)  # Remove numbers and periods between numbers

    # Fixing spaces before periods and removing extra spaces
    text = re.sub(r’\s*\.\s*’, ’. ’, text)  # Normalize space around periods to ensure one space follows the period

    text = re.sub(r’\s+’, ’ ’, text).strip()  # Remove extra spaces and trim the text

    # Splitting text into tokens/words
    tokens = text.split()

    # Removing stop words (common words that typically don’t carry much meaning)
    stop_words = set(stopwords.words(’english’))
    tokens = [word for word in tokens if word not in stop_words]

    # Initializing the lemmatizer
    lemmatizer = WordNetLemmatizer()

    # Part-of-speech tagging for each token
    tagged_tokens = pos_tag(tokens)

    # Lemmatizing each token based on its part-of-speech tag
    tokens = [lemmatizer.lemmatize(word, get_wordnet_pos(pos)) for word, pos in tagged_tokens]

    # Joining the processed tokens back into a single string
    preprocessed_text = ' '.join(tokens)
    return preprocessed_text  # This line is added to return the processed text

# Example text to be processed
text= "Example text"
preprocessed_text = preprocess(text)
```

Listing A.1: Python Source Code for Text Preprocessing

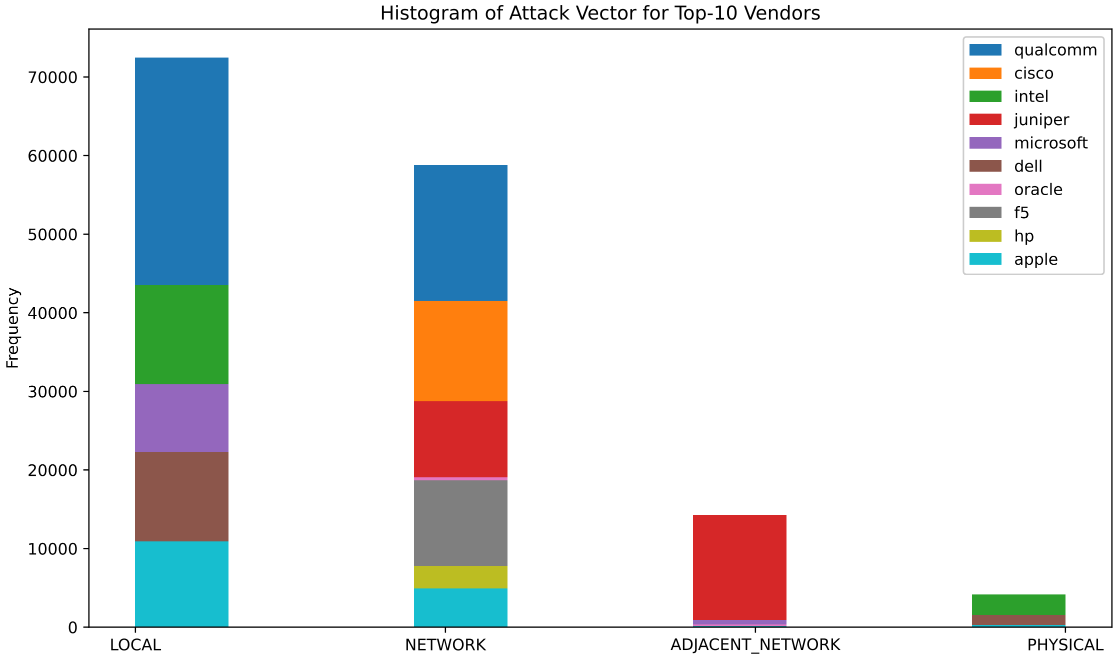


Figure A.1: Histogram of CVSS V3 Attack Vector Values for Top-10 Vendors

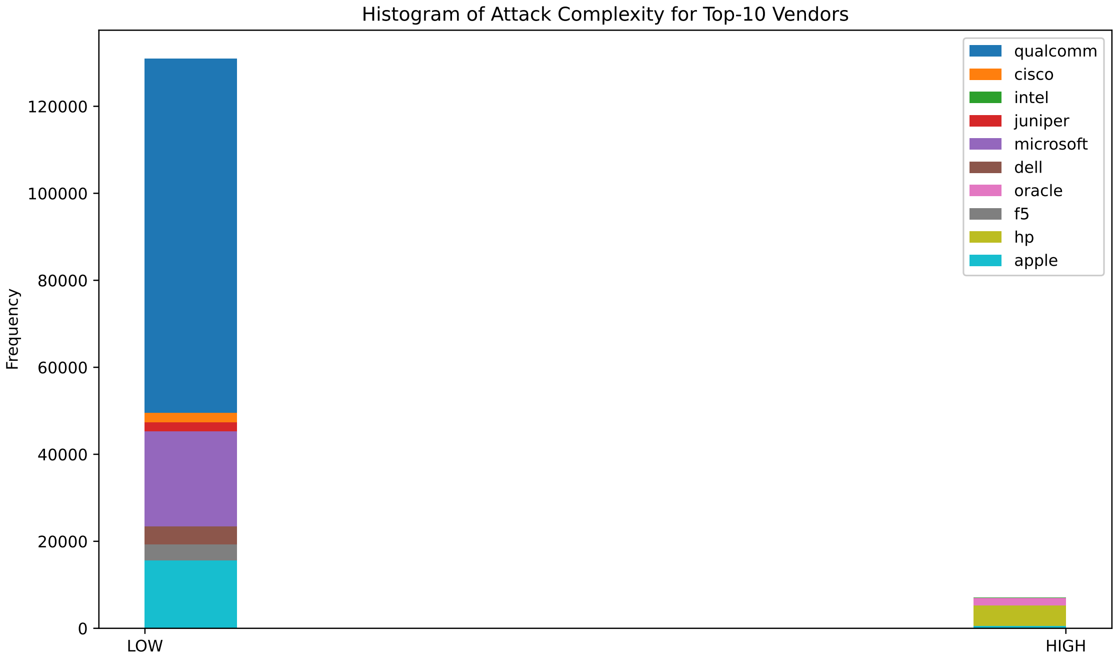


Figure A.2: Histogram of CVSS V3 Attack Complexity Values for Top-10 Vendors

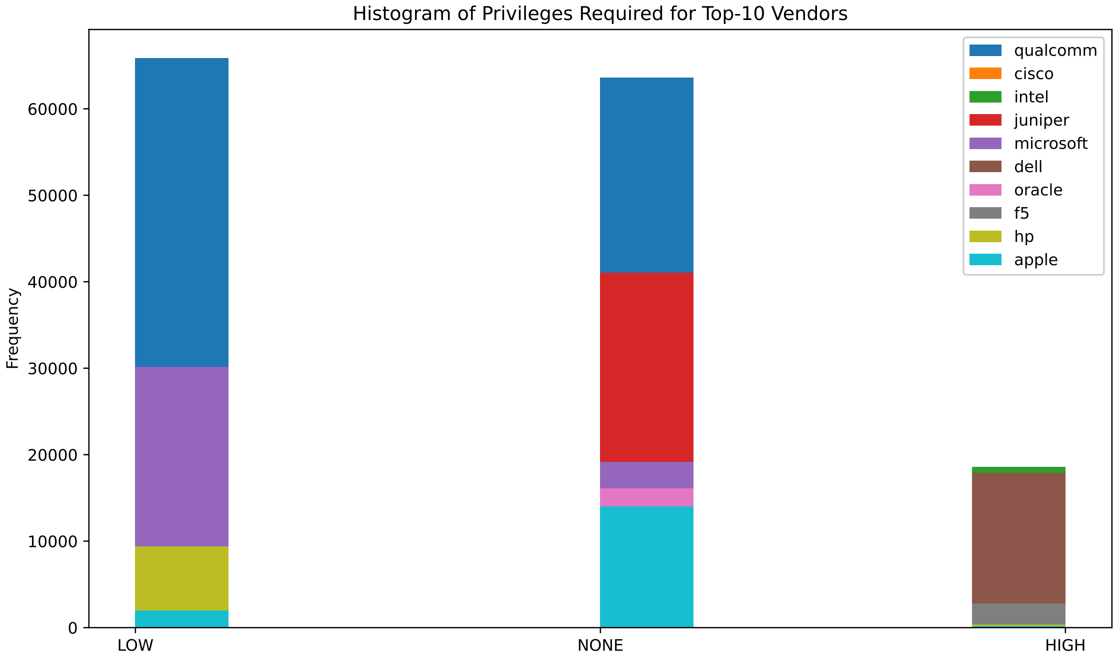


Figure A.3: Histogram of CVSS V3 Privileges Required Values for Top-10 Vendors

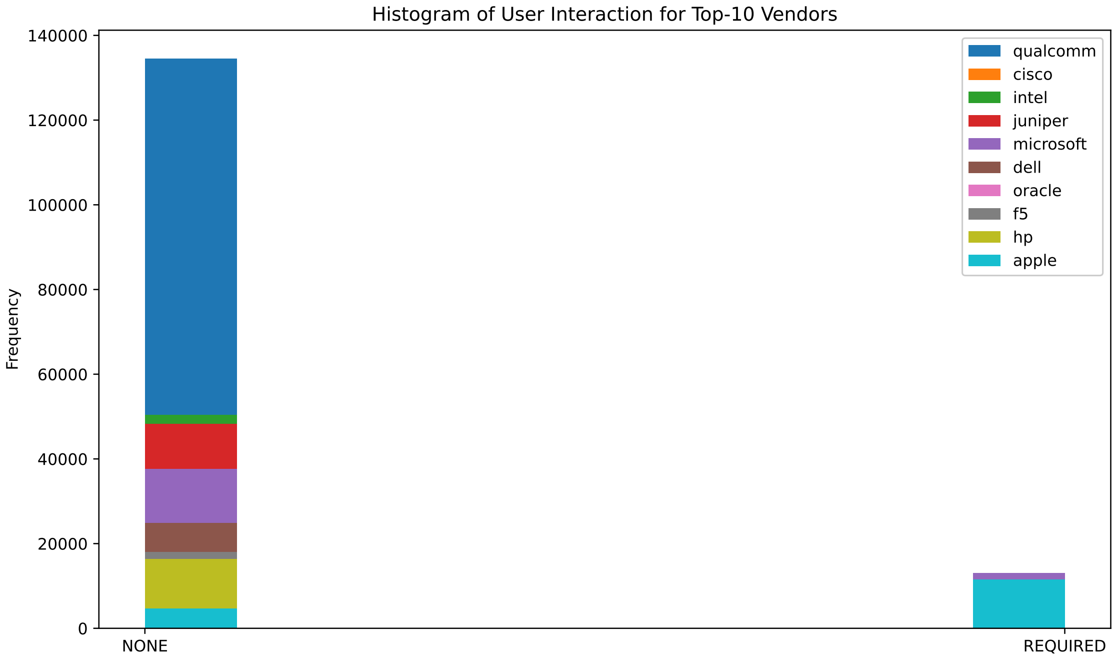


Figure A.4: Histogram of CVSS V3 User Interaction Values for Top-10 Vendors

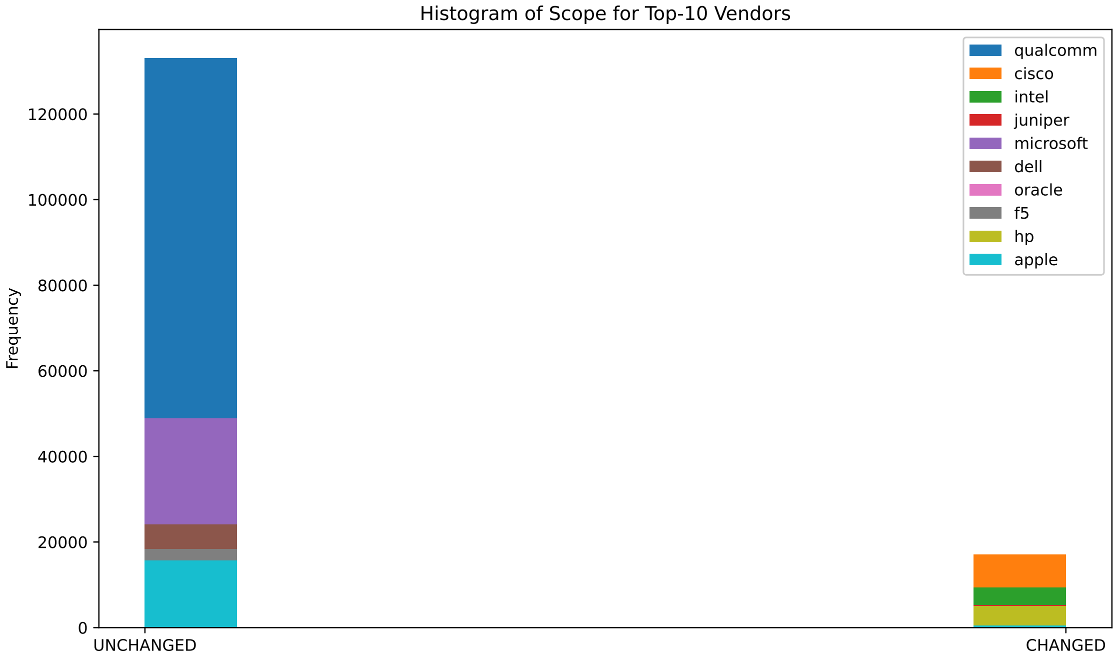


Figure A.5: Histogram of CVSS V3 Scope Values for Top-10 Vendors

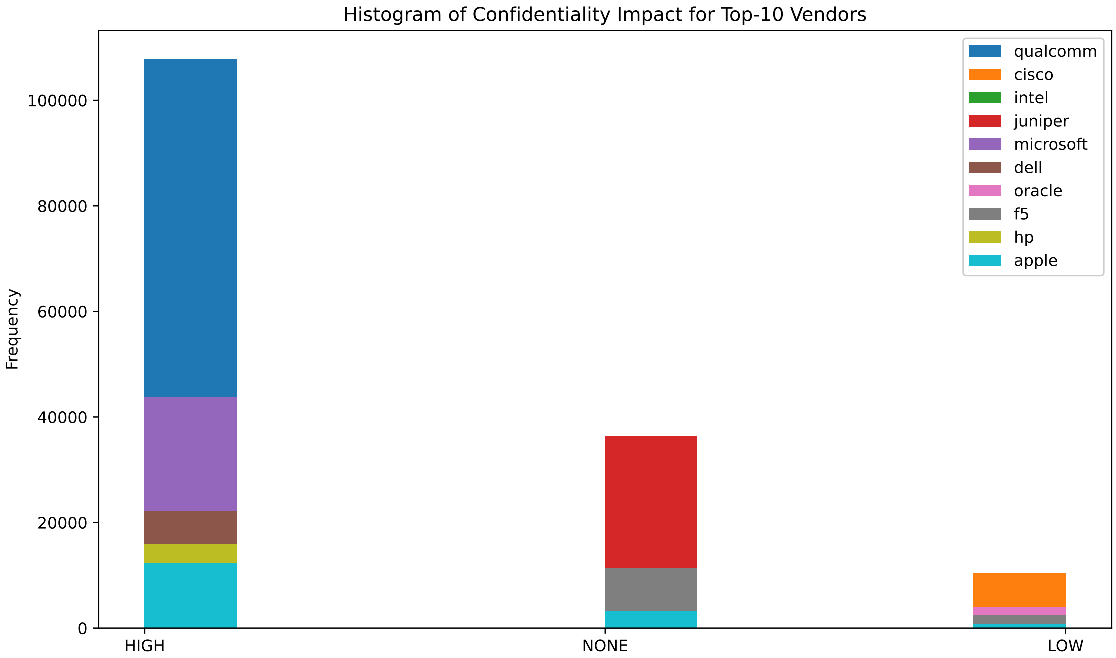


Figure A.6: Histogram of CVSS V3 Confidentiality Impact for Top-10 Vendors

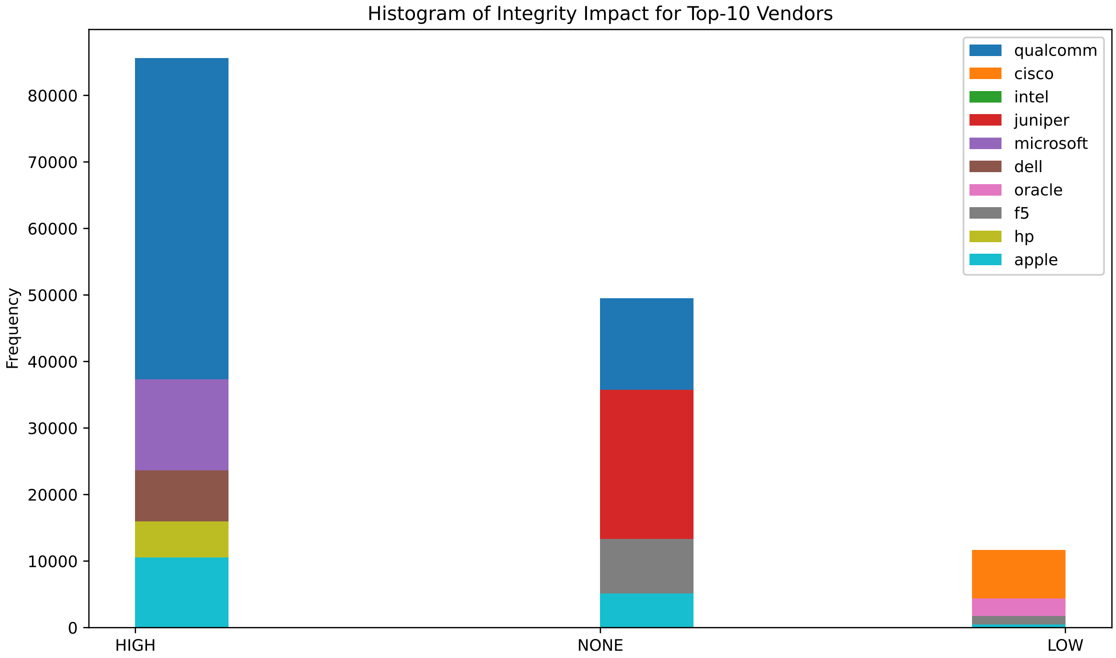


Figure A.7: Histogram of CVSS V3 Integrity Impact for Top-10 Vendors

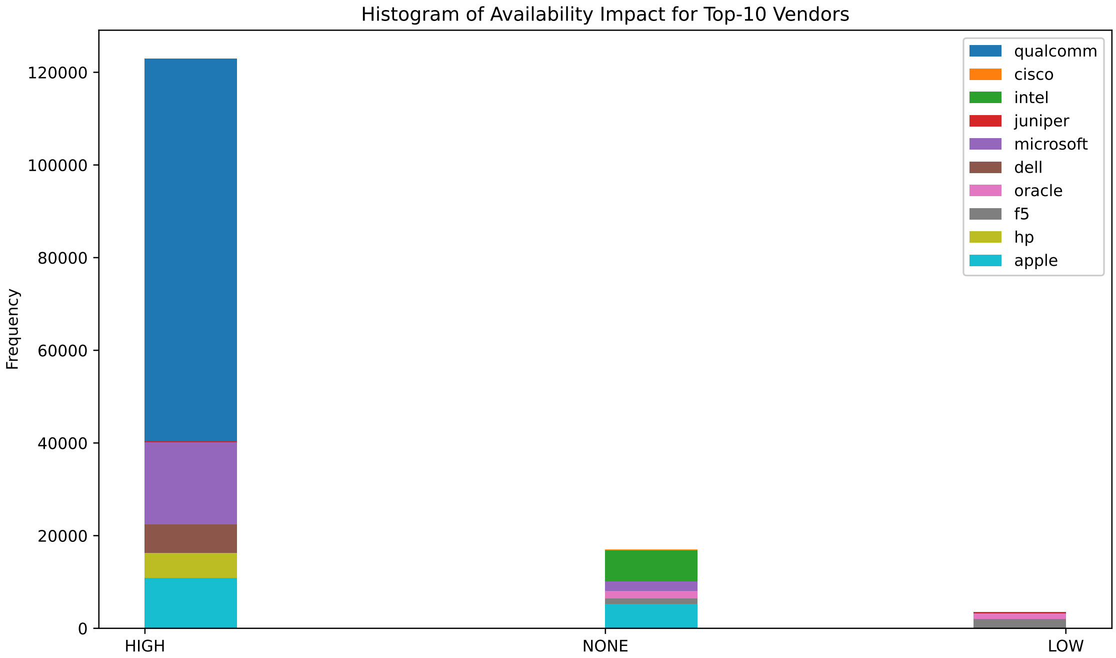


Figure A.8: Histogram of CVSS V3 Availability Impact for Top-10 Vendors

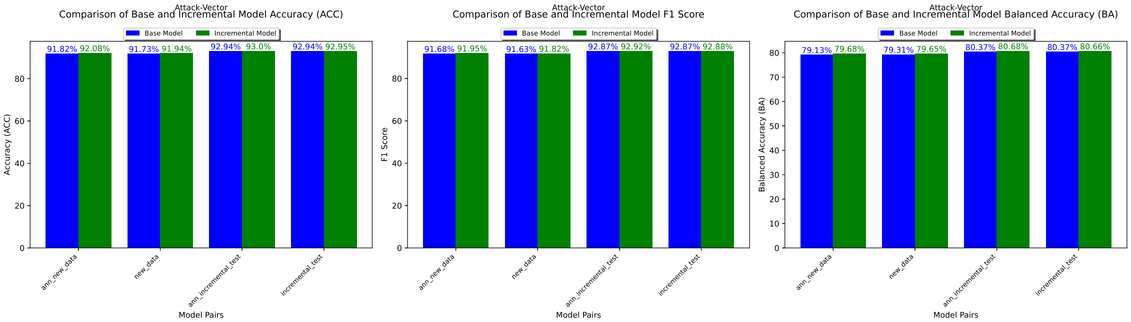


Figure A.9: Comparison of base model and incremental model performances on new data and incremental test data for the Attack Vector

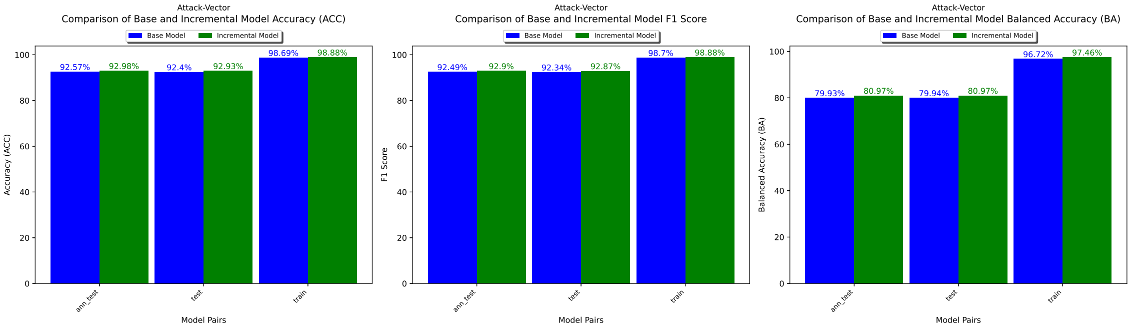


Figure A.10: Comparison of base model and incremental model performances on train and test data for the Attack Vector

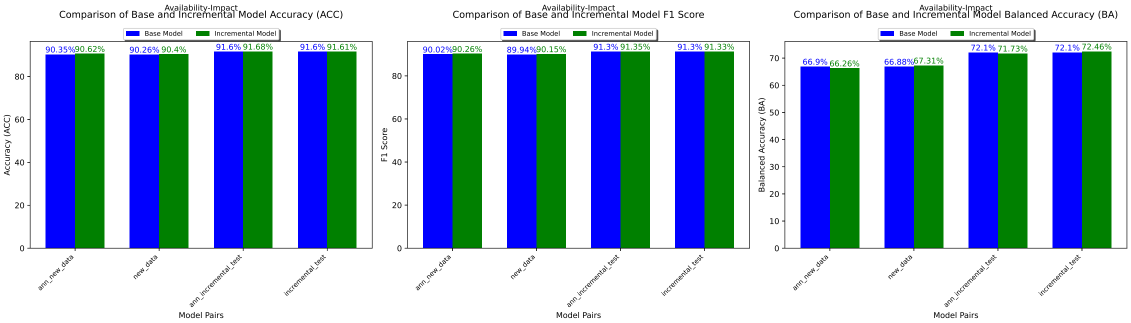


Figure A.11: Comparison of base model and incremental model performances on new data and incremental test data for the Availability Impact

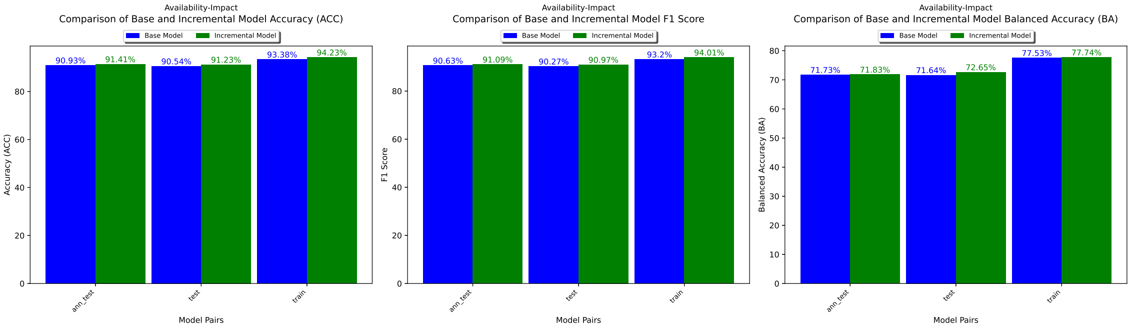


Figure A.12: Comparison of base model and incremental model performances on train and test data for the Availability Impact

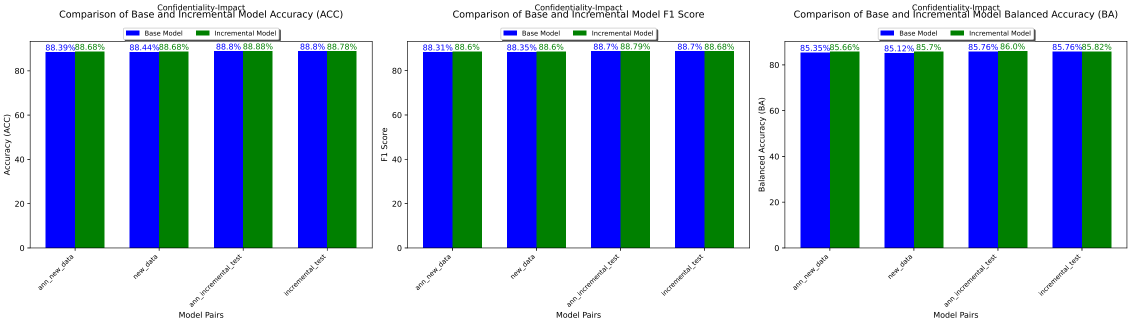


Figure A.13: Comparison of base model and incremental model performances on new data and incremental test data for the Confidentiality Impact

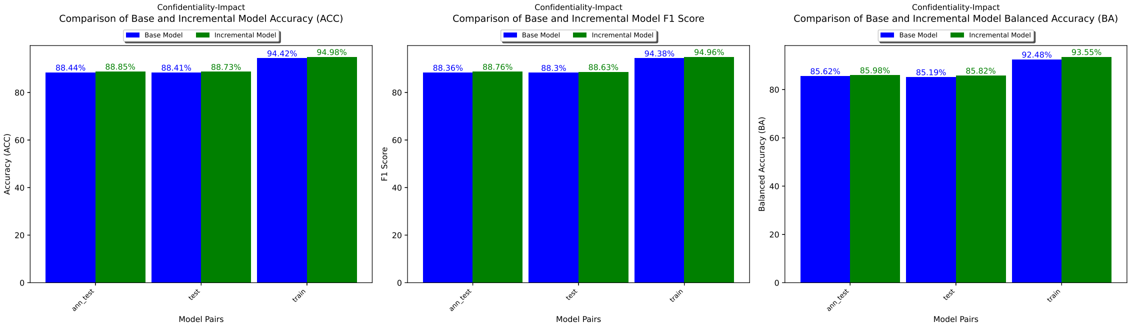


Figure A.14: Comparison of base model and incremental model performances on train and test data for the Confidentiality Impact

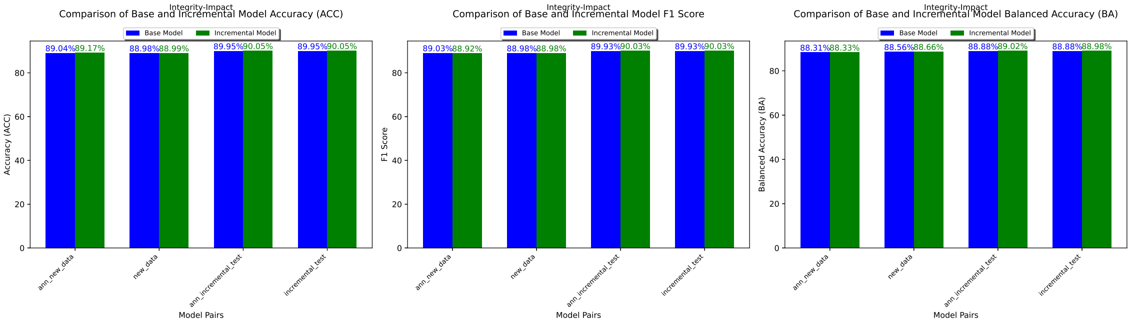


Figure A.15: Comparison of base model and incremental model performances on new data and incremental test data for the Integrity Impact

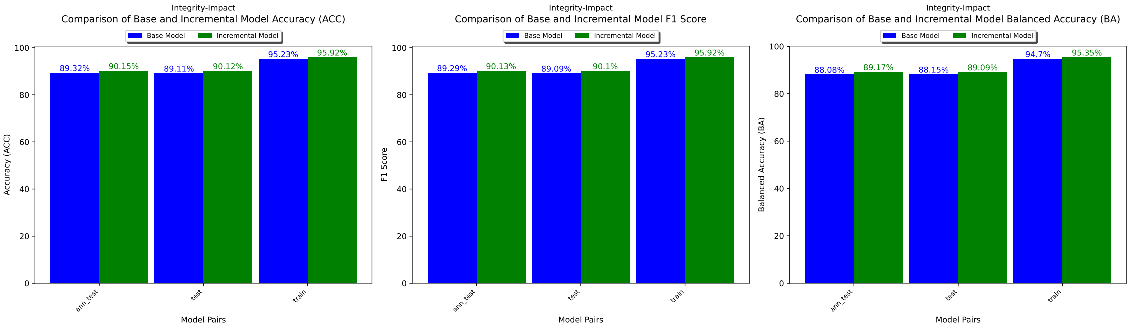


Figure A.16: Comparison of base model and incremental model performances on train and test data for the Integrity Impact

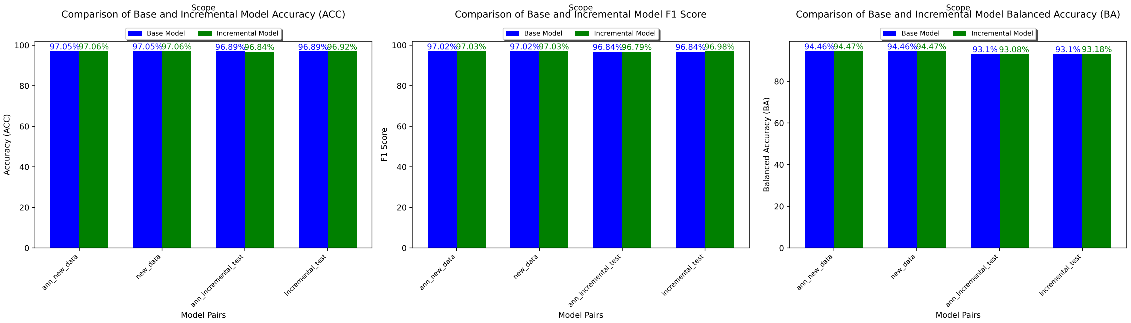


Figure A.17: Comparison of base model and incremental model performances on new data and incremental test data for the Scope

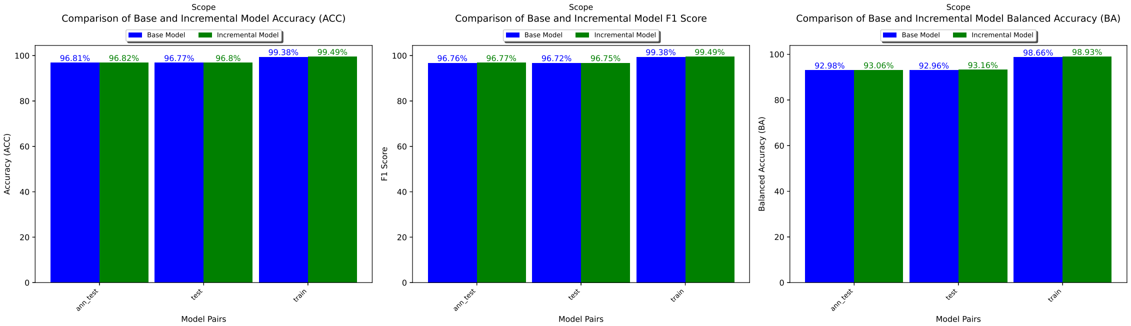


Figure A.18: Comparison of base model and incremental model performances on train and test data for the Integrity Impact

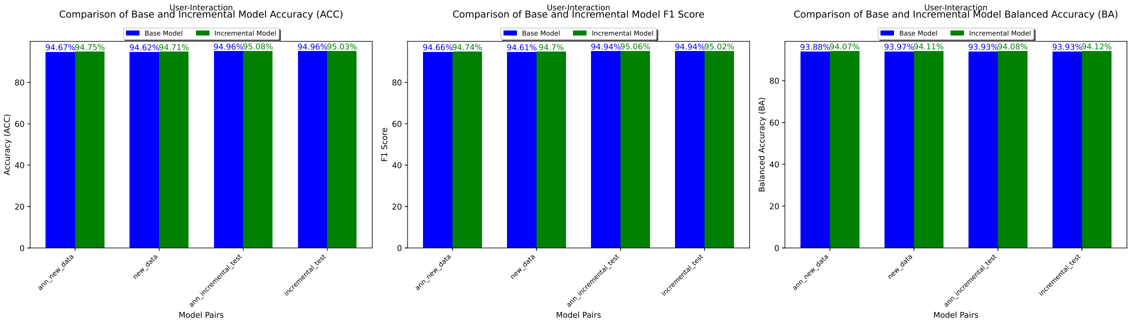


Figure A.19: Comparison of base model and incremental model performances on new data and incremental test data for the User Interaction

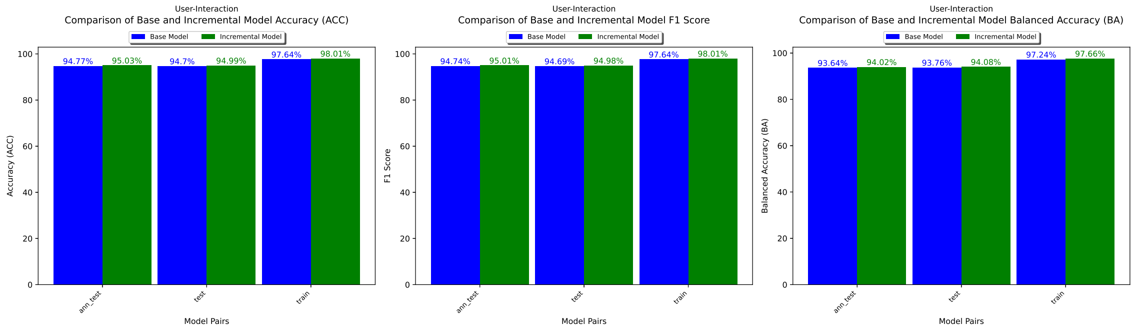


Figure A.20: Comparison of base model and incremental model performances on train and test data for the User Interaction

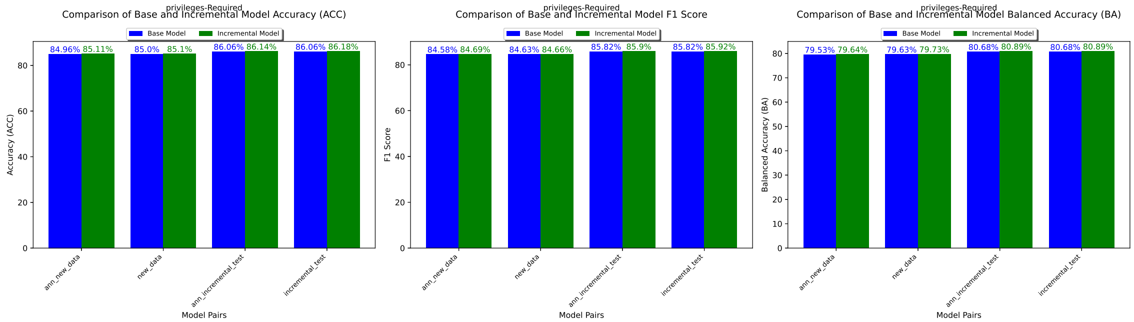


Figure A.21: Comparison of base model and incremental model performances on new data and incremental test data for the Privileges Required

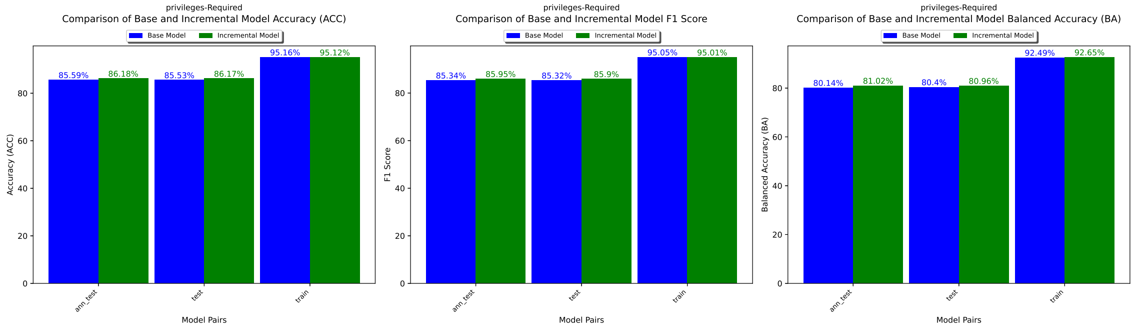


Figure A.22: Comparison of base model and incremental model performances on train and test data for the Privileges Required

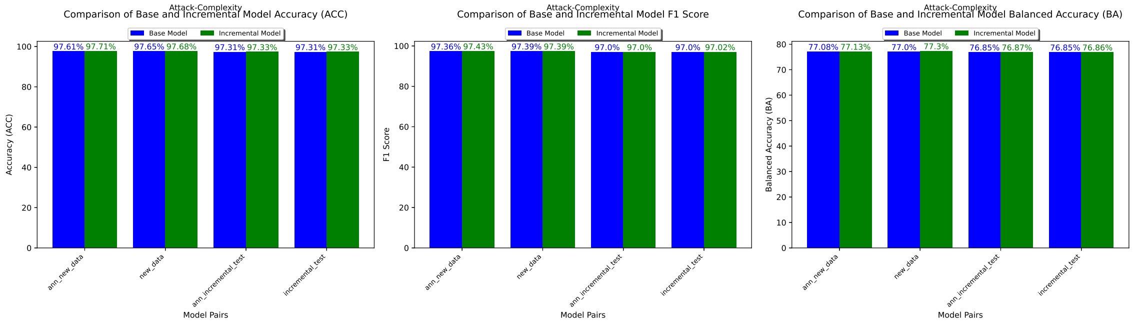


Figure A.23: Comparison of base model and incremental model performances on new data and incremental test data for the Attack Complexity

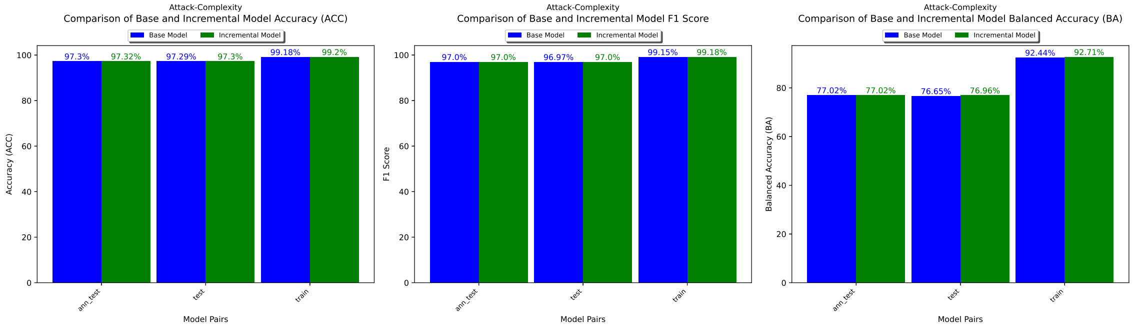


Figure A.24: Comparison of base model and incremental model performances on train and test data for the Attack Complexity

Vita

Candidate's full name: Saeid Bahmanisangesari
University attended (with dates and degrees obtained):

- Master of Computer Science University of New Brunswick
  2022-2024

- Bachelor of Science in Computer Engineering Ferdowsi University of Mashhad
  2016-2021