2023-05-01

Modeling And Predicting Emerging Threats Using Disparate Data

Ismael Villanueva Miranda
University of Texas at El Paso

MODELING AND PREDICTING EMERGING THREATS USING DISPARATE DATA

ISMAEL VILLANUEVA-MIRANDA

Doctoral Program in Computer Science

APPROVED:

Monika Akbar, Ph.D., Chair

Mahmud Shahriar Hossain, Ph.D.

Fang Jin, Ph.D.

Stephen Crites, Ph.D.
Dean of the Graduate School

MODELING AND PREDICTING EMERGING THREATS USING DISPARATE DATA

by

ISMAEL VILLANUEVA-MIRANDA

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

Department of Computer Science

THE UNIVERSITY OF TEXAS AT EL PASO

May 2023

Abstract

Early detection is crucial to mitigate the impact of emerging threats. This work proposes four innovative frameworks that build machine learning and deterministic epidemiological models using multiple domain-specific datasets to detect the onset of emerging threats in two domains: infectious diseases and cybersecurity. Our models are designed to detect infectious disease outbreaks, model their spread, detect malware activity, and analyze the relationship between software/hardware weaknesses and attack techniques.

First, we present a novel framework to detect multiple infectious disease outbreaks by integrating standardized disease-specific domain knowledge and public search trend data. Our framework showed high performance in identifying infectious disease outbreaks — diseases that are among the leading causes of illness and death in the United States— using people's search data. In addition to detecting outbreaks, studying their spread within a region is equally important. Therefore, we present the SEIRD+m model, which integrates human mobility data into the classical deterministic SEIR epidemiological model to provide a more accurate approach to modeling epidemics. We demonstrated its efficacy using COVID-19 as a case study, showing that restricting mobility only in COVID-19 hotspots can effectively reduce predicted infections and deaths among at-risk populations, including those based on race, income, and age.

Both infectious diseases and computer malware require timely and accurate detection to minimize their impact. Therefore, we extended our disease outbreak detection framework to detect malware activity over a geographic region. We use natural language processing (NLP) approaches to connect disparate cybersecurity datasets, enabling the development of a machine learning model for detecting malware activities based on people's search trends in a specific location. Our model has proven effective in identifying malware activity in four real-world attack case studies. Aside from detecting malware activities, it is necessary to investigate the properties of software vulnerabilities and how these properties

are used to compromise systems, in order to prevent cyberattacks and mitigate their impacts. Thus, we propose a framework that leverages NLP techniques to find connections between attack techniques and software vulnerabilities. The effectiveness of our framework is demonstrated through three case studies, highlighting its potential in identifying potential security/software vulnerability exploitation of multiple software weaknesses.

The approaches presented in this work provide evidence that the integration of domain-specific datasets and user-generated dynamic data can enable the development of highly effective computational models for detecting emerging threats. By leveraging these models, decision-makers can rapidly identify and respond to potential threats, leading to a more efficient allocation of resources. Our work opens up exciting opportunities for further research in this area.

Table of Contents

Page Chapter

List of Figures

List of Tables

Chapter 1

Introduction

Recent global events have emphasized the urgent need for advanced technological tools to address the numerous and complex challenges faced by the world today. Threats such as disease outbreaks, cyber-attacks, and pandemics have significant risks to humanity, with profound and long-lasting consequences.

Disease outbreaks can spread rapidly across borders and continents, leading to widespread illness and death [1]. Similarly, cyber-attacks can compromise critical infrastructure and disrupt essential services, such as healthcare and energy infrastructure [2]. These challenges require the use of interdisciplinary knowledge and the implementation of technological tools to mitigate their impacts.

To address these threats, computer science techniques (i.e., machine learning and big data analytics) used in conjunction with epidemiology models (i.e., compartmental models) can play a critical role. For instance, AI-enabled early warning systems can analyze real-time data to detect and track disease outbreaks or malware activity, providing crucial insights into the spread and nature of the threat [3]. Moreover, the use of big data analytics can help policymakers understand the economic and social consequences of these challenges, enabling them to make well-informed decisions regarding global or local interventions.

The primary objective of this study is to develop tools to detect and predict emerging threats using disparate data, with a particular focus on infectious diseases and cybersecurity. Furthermore, this research aims to integrate the fields of epidemiology, cybersecurity, and Artificial Intelligence to improve our comprehension of how threats emerge and to minimize their negative impact on society.

In the following sections, we present the motivation behind our work and define the

research questions that we address in each chapter of this dissertation. Additionally, Table 1.1 provides a summary of our research and contributions.

Table 1.1: Guzdial chart summarizing our research and contributions.

Topic	Research Questions	Data	Methods	Evaluation	Contributions
Multi-disease Outbreak Detection	**RQ1:** What is the effect of using HSDN disease-specific symptoms in detecting the outbreaks of infectious diseases? **RQ2:** How does different weighting of symptoms affect the detection of infectious disease outbreaks? **RQ3:** What effect does the number of symptoms have on the detection of outbreaks?	**D1:** Human Symptom-Disease Network **D2:** National Notifiable Diseases Surveillance System **D3:** Google Trends Scores	**M1:** Logistic Regression, Gaussian Naive Bayes, Decision Trees, K-Nearest Neighbor, Gaussian Process Classifier, Stochastic Gradient Descent, Linear Discriminant Analysis, XGBoost, Support Vector Machine, Random Forest, Multi-Layer Perceptron-Neural Network **M2:** TF-IDF	**E1:** Precision **E2:** Recall **E3:** Accuracy **E4:** F1-Score **E5:** Cross-Entropy Loss	**C1:** A method to detect infectious disease outbreaks by using standardized disease symptoms in conjunction with Google trends data **C2:** A robust and scalable framework for detecting multiple disease outbreaks
Human Mobility Driven Modeling of an Infectious Disease	**RQ4:** What is the effect of human mobility on the number of infections and deaths caused by highly contagious diseases? **RQ5:** What is the effect of human mobility on the number of infections and deaths caused by COVID-19 based on race, income, and age?	**D4:** COVID-19 data **D5:** SafeGraph Open Census Data **D6:** US CB TIGER **D7:** HUD-USPS **D8:** SafeGraph Mobility Data	**M2:** SEIRD+m **M3:** COVID-19 hotspost method	**E6:** Quantitative comparison to the baseline (actual COVID-19 data)	**C3:** A SEIRD+m model to simulate the dynamics of Coronavirus-like diseases **C4:** Integration of human mobility into the Ordinary Differential Equations of the SEIR model **C5:** A new approach to detect COVID-19 hotspots based on factors such as race, income, and age
Detecting Malware Activity Using Public Search Data	**RQ6:** How can we detect malware activities with little or no dependency on intrusion detection systems?	**D9:** Common Vulnerabilities and Exposures **D10:** MITRE ATT&CK **D11:** CISSM **D12:** CIRA **D13:** DMA **D14:** Google Trends Scores	**M4:** K-Nearest Neighbors algorithm **M5:** Moving Average Algorithm **M6:** Outliers using Standard Deviation	**E7:** The KNN class probabilities of the input samples	**C6:** A novel approach to linking knowledge from heterogeneous and specialized datasets (CVE, MITRE ATT&CK) using a sentence embedding approach **C7:** An novel approach to detect malware activity over a regional area using standardized and specialized datasets and people's search interest data. **C8:** We provide study cases for the Conti and Ryuk malware using data from real attacks.

1.1 Multi-disease Outbreak Detection

In the last century, infectious diseases have caused 15 out of 57 million annual deaths worldwide [4]. To address the emergence and re-emergence of infectious diseases, it is crucial to develop new outbreak detection systems that guide public health policies and implement timely mitigation strategies. Researchers have developed various systems over the past two decades using different methods to detect outbreaks of infectious diseases. Recently, researchers have focused on using data from web search engines, such as Google Trends, to detect disease outbreaks. However, the use of non-standardized disease-related terms in Google Trends poses a limitation, as different researchers may use different terms to refer to the same disease.

Moreover, most of the proposed outbreak detection models are designed to detect a single disease, which limits their usefulness in detecting other infectious diseases. Detecting outbreaks using web search data with non-standardized terms is also challenging, as it requires epidemiology expertise to select the most appropriate search terms relevant to a specific disease.

In Chapter 3, we propose a single framework to simultaneously detect multiple infectious disease outbreaks. Our approach combines standardized disease symptoms, derived from the Human Symptoms-Disease Network (HSDN), with Google Trends scores. The novelty of our study is the integration of static and standardized medical terms, such as symptoms, with dynamic Google Trends data to detect outbreaks of infectious diseases.

We address the following research questions in Chapter 3:

- What is the effect of using HSDN disease-specific symptoms in detecting outbreaks of infectious diseases?
- How do different weighting of symptoms affect the detection of infectious disease outbreaks?
- What effect does the number of symptoms have on the detection of outbreaks?

In chapter 3, we also present the results of our framework's performance using 11 infectious diseases reported by the Centers for Disease Control and Prevention (CDC). Our experimental results show that the proposed framework achieves good results. We also found that using HSDN symptoms significantly enhances the framework's performance compared to only using Google Trends data, indicating the critical role of integrating static HSDN data with dynamic Google Trends data in detecting outbreaks. Furthermore, our study suggests that diseases with a higher number of symptoms are more easily detected than those with fewer symptoms. Finally, using TF-IDF scoring of HSDN symptoms improves the detection of outbreaks. Our findings suggest that standardized disease symptoms and user search data are excellent indicators for detecting outbreaks of infectious diseases.

1.2 Human Mobility Driven Modeling of an Infectious Disease

Human mobility is an integral part of daily life but can increase the risk of contracting infectious diseases like COVID-19. Mortality rates due to COVID-19 in the United States vary among different ethnic groups, with disparities observed across populations [5]. The death rate is higher for individuals over the age of 80, with mortality rates reaching as high as 20Despite the established links between COVID-19 and age and race factors, research on the relationship between poverty levels and infectious diseases has been relatively scarce. However, recent studies have shown that income inequality is directly linked to a 4% increase in COVID-19 cases per million and a 5% increase in COVID-19 deaths per million for every 1% increase in the Gini coefficient, which measures inequality in a population [6].

In Chapter 4, we introduce our model, SEIRD+m, which adds a new compartment, D (Deaths), to the Susceptible-Exposed-Infected-Recovered (SEIR) model and incorporates human mobility aspects into all its components. Our model can be used to investigate the impact of mobility restrictions on COVID-19 hotspots based on demographic factors such as income, age, and race.

In Chapter 4, we address the following research questions:

- What is the effect of human mobility on the number of infections and deaths caused by highly contagious diseases?
- What is the impact of human mobility on the number of COVID-19 infections and deaths based on race, income, and age?

In chapter 4, we demonstrate the use of our model to investigate the spread of COVID-19 when human mobility is restricted only in COVID-19 hotspots at various intensities and time periods. Our experiments show that reducing mobility in COVID-19 hotspots can significantly reduce infections and deaths, rather than implementing mobility restrictions across an entire region. Moreover, we found that restricting mobility in COVID-19 hotspots

positively affects infection and death rates at both the regional and local (within hotspots) levels, making it a valuable strategy to protect vulnerable areas based on income, race, or age.

1.3 Detecting Malware Activity Using Public Search Data

Malware, a type of intrusive software developed by cyber criminals and commonly transmitted through computer networks, poses a significant threat to several critical sectors of the global economy, including finance, healthcare, and energy. Recent reports indicate that the cost of cybercrime has surged to $6 trillion, causing significant harm to the world economy[1].

On May 7, 2021, the Colonial Pipeline system suffered a ransomware attack, resulting in a six-day shutdown[2]. This pipeline, which has a length of 5,500 miles and a daily transportation capacity of 2.5 million barrels of fuel, supplies 45% of the fuel consumed in the Southeast and East Coast regions. The attack was executed by DarkSide, a group of hackers who operate using a "ransomware as a service" business model. They targeted the company's billing infrastructure and obtained nearly 100 gigabytes of data within two hours. The FBI intervened to identify the source of the attack, and the company had to halt all pipeline operations.

The identification of malware activities can be a crucial factor in preventing attacks, such as the attack that occurred on the Colonial Pipeline system. In this context, public search data patterns represent a valuable resource for the early detection of malware activity, with the potential to significantly reduce the severity of their impact. It is important to note that while traditional approaches for malware detection usually rely on intrusion detection tools or techniques, people often search online to identify symptoms of malware infection.

In Chapter 5, we propose a novel method to detect malware activities that combine

[1]https://cybersecurityventures.com/cybercrime-damages-6-trillion-by-2021/

[2]https://www.colpipe.com/news/press-releases/media-statement-colonial-pipeline-system-disruption

publicly available databases with web search data. Our approach involves creating Attack-specific Vulnerability ExposuRe Terms (AVERT) datasets by utilizing the Common Vulnerabilities and Exposures (CVE) and the MITRE ATT&CK databases and public search data obtained from the Google Trends API. These datasets are used to train a machine-learning model. Our results demonstrate that it is possible to detect abnormal malware activity using malware-related search terms.

Chapter 5 aims to investigate **how to detect malware activity through public search data**. Our findings indicate that user web searches tend to increase approximately seven days before and after an attack. When the time series of search queries exhibit the highest values with the presence of outliers, it is highly probable that an ongoing malware attack is occurring. These results provide evidence for the potential utility of public search data patterns in detecting malware activities and identifying potential cybersecurity threats.

1.4 Uncovering Threat Vectors through Attack Analysis

The rapid advancement of technology has led to an increased need for secure software and Industrial Control Systems (ICS). Cyberattacks and data breaches have become significant threats to individuals and organizations, causing extensive damage. Researchers and industry professionals have focused on developing techniques to identify and mitigate software vulnerabilities to improve cybersecurity.

One technique that has gained significant attention is the automatic linkage between Common Vulnerabilities and Exposures (CVEs), Common Weakness Enumeration (CWEs), and Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK) techniques. CVE provides a standardized method of identifying and tracking vulnerabilities in software systems, while CWE is a classification system for software weaknesses that can lead to vulnerabilities. The ATT&CK framework is a knowledge base of adversarial tactics, techniques, and procedures that can be used to test and evaluate the security of software

systems.

However, linking CVEs, CWEs, and ATT&CK techniques automatically poses significant challenges, and some researchers have focused on manually linking these datasets, which is time-consuming. Automatic linkage techniques have the potential to improve the speed and efficiency of the vulnerability identification and mitigation process, as well as provide valuable insights into the tactics and techniques used by attackers, enabling better defense against cyber threats.

In Chapter 6, we propose a novel approach to automatically link weaknesses and techniques for improved cybersecurity in ICS. We aim to investigate the role of software vulnerabilities and weaknesses in ICS attacks.

Chapter 2

Related work

2.1 Detecting infectious diseases outbreaks

Several studies for detecting outbreaks have been conducted in the last two decades A variety of approaches were used in these studies to detect outbreaks, including *regression analysis*, *time series analysis*, *multivariate methods*, and, more recently, *web search data*. The following subsections briefly present some of the works in these areas.

2.1.0.1 Regression analysis

Regression models have been used in surveillance systems to detect and notify outbreaks. Pelat et al. [7] proposed an online tool based on a regression model for detecting and quantifying outbreaks within a seasonal period. Wieland et al. [8] developed an outbreak detection method using a generalized additive model to obtain a stable specificity. In these studies, an observation is classified as atypical based on a threshold. In contrast, Frisén et al. [9] proposed parametric and semi-parametric regression models to identify epidemic onset.

2.1.0.2 Time series analysis

The Auto-Regressive Integrated Moving Average model (ARIMA) is one of the most commonly used time series analysis models to detect epidemics of infectious diseases. Miller et al. [10] present an ARIMA-based model to predict the enrollment of patients with influenza-like symptoms at ambulatory care centers. Using hierarchical time series models, Heisterkamp et al. [11] investigate the possibility of detecting deviations (i.e., outbreaks) from

an expected incidence. Since these count data are noisy, these models are suitable for detecting signals from noise and accounting for possible auto-correlation.

2.1.0.3 Multivariate methods

Multivariate analysis is a set of statistical methods to analyze patterns in multidimensional datasets. These methods consider more than two variables, which allows the analysis of more complex problems. As an example, in the United Kingdom, Marshall et al. [12] have explored the possibility of monitoring disease outcomes across multiple units. Kulldorff et al. [13] examine the simultaneous incorporation of several data sets into a single likelihood function in order to generate an outbreak signal if it appears in only one or several datasets.

2.1.0.4 Web search query data

In recent years, Google Trends (GT) data has been widely used in outbreak detection systems [14], [15]. Gianfredi et al. [16] presented the results of a study related to Pertussis outbreaks. In this work, GT was mined in different European countries for nine years. To find cases of Pertussis, they employed the 'search topic' strategy. Data on Pertussis cases and deaths obtained from the *European Centre for Disease Analysis and Control* database was correlated with GT results related to Pertussis. A strong correlation between Pertussis cases and GT-based search volumes was observed among European countries (ranging between 0.94 and 0.97). In [17], GT was used to investigate Internet activity related to loss of smell in eight countries. Spearman rank correlation coefficients were analyzed to observe the correlation between loss-of-smell relative search volumes (RSVs) and the increase in daily confirmed cases of COVID-19. The authors found a close correlation (ranging from 0.63 to 0.95) between RSVs related to loss of smell and the number of COVID-19 cases and death rates across all eight countries.

2.2 Using epidemiological models to study the impact of human mobility on the spread of infectious diseases.

Human mobility is an integral part of everyday life that enables humans to fulfill their basic needs, such as attending work, participating in recreational activities, and purchasing food. However, mobility can expose individuals to a variety of health risks, including infections from contagious diseases such as COVID-19.

Additionally, social factors such as race, income, and age are essential determinants of the number of infections and deaths related to infectious diseases (e.g., COVID-19). In the U.S.A, for example, the mortality statistics related to COVID-19 show that per 100,000 inhabitants, 256 were among Indigenous Americans, 180 among Black Americans, 177 among Pacific Islander Americans, 150 among White Americans, 147 among Latino Americans, and 96 among Asian Americans [5]. In terms of income inequality, an increase of 1% in the Gini coefficient (a measure of inequality in a population) is directly related to a 4% increase in COVID-19 cases per million and a 5% increase in COVID-19 deaths per million [6]. A significant factor influencing COVID-19 mortality rates is age. For example, the mortality rate varies significantly between countries but may reach 10% on average [18]. However, mortality rates may reach 20% in people over 80 years of age [19], with or without multi-morbidity [20].

2.2.1 Epidemiological models and COVID-19

Epidemiological models, including the classic SIR and SEIR models, have been extensively applied to analyze the COVID-19 pandemic. Researchers have explored public health interventions (e.g., social distancing) using the SIR and SEIR models. Researchers from various disciplines have studied the COVID-19 pandemic. Specifically, the numerical simulation of COVID-19 using mathematical models based on the SIR and SEIR models has been widely used.

A study that utilized the SIR model found that when the number of active infections in the general population exceeds 1% and 10%, the health system is seriously challenged, and severe staffing shortages occur. [21]. An extended SEIR model was used to study the role of testing and case-dependent quarantine, finding that testing at a high rate combined with targeted quarantine policies mitigated the economic effects of COVID-19 [22]. A study analyzed the impact of human mobility restrictions in Shenzhen, China, using the SEIR model. The authors found that the city experienced a significant reduction in the peak of cases between 33-66% by reducing mobility between 20-60% [23]. In another study, researchers analyzed the mobility of small groups of active individuals and gathering venues. They found that the total number of infections was reduced by applying mobility interventions focused on these individuals and venues [24].

2.2.2 Epidemiological models and human mobility to predict the spread of COVID-19

Researchers from various disciplines have studied the COVID-19 pandemic. Specifically, numerical simulation of COVID-19 using mathematical models based on the SIR and SEIR models augmented with mobility data have been used (See table 2.1). [25] estimate the transmission rates as a function of the population mobility. They found a stable association between mobility and transmission rates that is conserved across several significant counties, and it outperforms non-mobility models when forecasting future deaths. [26] incorporate human mobility in the Susceptible and Exposed compartments of the SEIR model. Their findings suggest that the heterogeneity of race and age are essential factors in the spread of COVID-19, meaning that policymakers will need to consider these heterogeneities when designing policies for mitigating COVID-19's spread. Similarly, [27] uses mobility in the transition between the Susceptible and Exposed compartments finding that a small minority of super spreader points of interest account for a large majority of the infections and that restricting the maximum occupancy at each point of interest is more effective than uniformly

reducing mobility.

Table 2.1: Comparison between approaches using epidemiological models and human mobility

Author	Model	Mobility data	Resolution	Mobility Incorporation
[25]	SEIR	SafeGraph	County	Estimates transmission rate as a function of the population mobility.
[26]	Stochastic SEIR	SafeGraph	Regions	Mobility is incorporated only in the Susceptible and Exposed compartments.
[27]	SEIR	Safegraph	Metro areas	Mobility is incorporated only in the transicion between the Susceptible and Exposed compartments.
[28]	SEIR	Baidu migration data	Provinces	Mobility (m) is used to compute the effective size of the populations at risk (q=m/N). Then q is used to compute the initial susceptible people (S0=qN).
[23]	SEIR	Unicom mobility data	Districts	Mobility is used to compute the force of the infection in the Susceptible and Exposed compartments.
[29]	SIRD	Google mobility	US States	Human mobility is used to estimate the infection rate (beta) in the Susceptible and Infected compartments during the initial, lockdown, and riot periods.
[30]	SEIR + Deep Learning	Google and Unacast mobility data	US Counties	The susceptible to infectious transition rate (β) is modeled as a function of mobility and social behavior.

In another work, [28] uses mobility data to compute the adequate size of the populations at risk (q). Then q is used to compute the initial Susceptible people. They found that the value of q has a linear relationship with human mobility data. [23] uses human mobility to compute the force of the infection in the Susceptible and Exposed compartments. According to their findings, a decrease in mobility of 20-60% within a city significantly impacted the COVID-19 spread by more than 33% (95% for UI 21-42). The effects of mobility restrictions were enhanced when combined with reductions in virus transmissibility of 25% or 50%. [29] examined human mobility during periods of lockdown, riot, and the initial period of infection to estimate the rate of infection. They found a strong correlation between the number of COVID-19 cases and the number of people visiting parks. As a result of the riots, residents began to use parks more frequently, which increased the infection rate of

the epidemic, thereby postponing the turning points in the USA as a whole and specifically in certain states. Similarly, [30] found that policies regarding working from home had a significant effect on reducing reproduction rates. Moreover, their analysis revealed a delay between changes in mobility, social behavior, and reproduction numbers. Variations in reproduction numbers observed on a particular day are influenced primarily by behaviors that occurred 17 to 21 days ago.

2.3 Cyber epidemics modeling

The term malware refers to malicious software or code that allows an attacker to cause significant damage to computers, mobile devices, and networks or allow unauthorized access to personal information. In addition to worms and Trojan horses, malicious actors generally use viruses, rootkits, spyware, and ransomware as their primary malware. This section will review previous work on malware detection approaches.

2.3.1 Malware detection approaches

The number of research works on malware detection has increased rapidly in recent years. One of the first and most popular methods was the signature-based detection method. Despite its effectiveness against known malware, this method is ineffective in detecting zero-day malware [31]. Over time, it became necessary to develop new strategies for malware detection such as behavior-based [32] [33] [34] [35] [36] [37], heuristic-based [38] [39] [40] [41], model checking-based [42] [43] [44] [45] [46] [47], deep learning-based [48] [49] [50] [51] [52] [53], cloud-based [54] [55] [56], mobile-based [57] [58] [59], and IoT-based [60] [61] [62], [63] [64].

2.3.1.1 Detecting malware using web data

Recently, some malware detection techniques have been using web-based data. In [65], the authors propose a Markov model for studying the effect of restricting infected web

pages from appearing in search results and decreasing their ranking on search engines. Another research [66] presents a method for identifying landing pages that lead to drive-by downloads. They collect malicious content from landing pages by querying the webpage cache of a commercial search engine to determine landing pages containing similar or the same content. In the same way, ShoposLabs [67] conducted research on how attackers are using blackhat Search Engine Optimization techniques to flood legitimate websites with content that will rank highly in search engine results and direct users to malicious websites.

In order to understand how malware spreads on websites, it is necessary to investigate the method of distribution. In [68], the author built web honeypots containing vulnerable applications to investigate malware distribution on websites. The authors found that anti-virus software frequently fails to detect malware files and that web honeypot traffic patterns help to detect malware on websites. A similar scheme is proposed in [69], where the destination URLs of attacks based on web application vulnerabilities can be corrected by seeing the path structure of honeypots.

Other researchers have focused their efforts on detecting malicious web pages. In [70], the authors present a method to search the web for malicious pages more efficiently. The initial seed is a collection of known malicious pages in this method. The method then generates search engine queries to identify other malicious pages similar to or related to the initial seed. Similarly, in [71], the authors used a collection of web users to track their interactions with websites, including the redirections they used to reach their final destinations. Lastly, they aggregate the redirection chains leading to a specific web page and analyze the characteristics of the resulting redirection graph to detect malicious web pages.

Recently, researchers have used web search results to explore and study malware attacks. In [72], the authors present a study examining the effects of mining Bing search query logs on gaining insights about ransomware attacks. First, they extract queries related to ransomware, and then they build a machine-learning algorithm to identify queries where

users seek assistance regarding ransomware attacks. Their results suggest that ransomware attacks are associated with user search behavior.

2.3.2 Uncovering Threat Vectors through Attack Analysis

In recent years, a limited number of studies have been conducted on the automatic linkage between Common Vulnerabilities and Exposures (CVEs), Common Weakness Enumeration (CWEs), and Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK) techniques. This section reviews some of these works to understand the approaches used and some of their limitations.

An approach presented in [73] proposes a Multi-Head Joint Embedding Neural Network model that leverages a knowledge base of 50 mitigation strategies to map Common Vulnerabilities and Exposures (CVEs) to MITRE ATT&CK techniques. The authors evaluate their proposed model on a dataset of over 62,000 CVE records, demonstrating promising performance. However, the authors also recognize some limitations, such as the incompleteness of the knowledge base and the need for additional data sources to improve mapping accuracy.

In [74], the authors focus their research on linking CVEs to Common Attack Pattern Enumeration and Classification (CAPEC) IDs. They propose a method using three similarity measures (TF-IDF, Universal Sentence Encoder, and Sentence-BERT) to trace related CAPEC-IDs from CVE-IDs. The authors report high accuracy using TF-IDF but also note that their method does not currently take vulnerability severity into account and requires verification of its adaptability to other repositories.

The work in [75] presents a dataset of 1813 CVEs annotated with MITRE ATT&CK techniques and proposes models for automatic linkage between the two based on CVE metadata text descriptions. The best model achieved an F1-score of 47.84%. The authors are aware of the limitations of their training set and suggest future work involving Few-Shot Learning and Semi-supervised learning methods.

In [76], the authors analyze CVE data to identify 94 concepts related to 11 logical

groups and use graph theoretical techniques to analyze the data. They also identify brands and technologies that are more prone to vulnerabilities and suggest prioritizing these when designing security measures. Additionally, the authors suggest using the CVE concepts to examine the coverage of security products and to identify recent hacker community trends.

In [77], the authors describe BRON, an aggregate data graph that links various sources of cybersecurity information, including MITRE ATT&CK MATRIX, Common Weakness Enumerations, CVEs, and CAPEC. The authors demonstrate the utility of BRON in enhancing the analysis of alerts, threats, and vulnerabilities.

In [78], the authors propose a system that uses Natural Language Processing techniques to classify and prioritize CVEs based on their relevance to a particular organization. The authors use a dataset of over 9,000 CVEs and demonstrate the efficacy of their approach in accurately predicting the relevance of a CVE to a given institution. The authors also suggest future work to improve the system's scalability and to address limitations such as the incompleteness of the CVE metadata.

Chapter 3

Integrating Heterogeneous Data for a Multi-disease Outbreak Detection Framework[1]

3.1 Introduction

According to [4], 15 of 57 million annual deaths worldwide are directly attributable to infectious diseases. In response to the emergence and re-emergence of infectious diseases, new outbreak detection systems are needed to help guide public health policies and implement timely mitigation strategies. Over the past two decades, researchers have developed systems using regression models [7], time-series analyses [11], and multivariate methods [13] for detecting outbreaks of infectious diseases. Most recently, researchers concentrated their efforts on detecting disease outbreaks using data compiled from web search engines (e.g., Google Trends) [14]. One limitation of these studies is that Google Trends is used with disease-related terms that are not standardized. In other words, different researchers can use different terms to refer to the same disease. Furthermore, most proposed models are developed specifically for a single disease, hence limiting their use for detecting other infectious diseases. Additionally, using web search data with non-standardized terms to detect outbreaks is challenging as it requires expertise in epidemiology to select the most appropriate search terms relevant to a specific disease.

In this work, we introduce one single framework to detect multiple infectious disease

[1] https://doi.ieeecomputersociety.org/10.1109/BigData52589.2021.9671841

outbreaks simultaneously. Our study combines a set of standardized disease symptoms (derived from the Human Symptoms-Disease Network – HSDN [79]) with Google Trends scores. The **novelty of this study** lies in the integration of static and standardized medical terms (e.g., symptoms) with dynamic Google Trends data to detect outbreaks of infectious diseases. The framework's performance was assessed using 11 infectious diseases reported by the Centers for Disease Control and Prevention(CDC)[2].

Experimental results show that our proposed framework achieves an average accuracy of greater than 95%. We also observed that the HSDN symptoms improve the performance of the proposed framework compared to only using Google trends data —indicating that the integration of static HSDN data and dynamic Google trends data is critical for detecting outbreaks. Experiments also showed that outbreaks of diseases with a higher number of symptoms are more easily detected than diseases with a lower number of symptoms. Lastly, we found that using TF-IDF [79] scoring of HSDN symptoms improves the detection of outbreaks. As a result of these findings, standardized disease symptoms and user search data appear to be excellent indicators for detecting outbreaks of infectious diseases.

Please note that the goal of the paper is to detect the outbreaks of multiple infectious diseases, not to predict the onset of these diseases. The key contributions of this paper include:

1. An innovative method to detect infectious disease outbreaks by using standardized disease symptoms in conjunction with Google trends data.

2. A robust and scalable framework for detecting multiple disease outbreaks.

The rest of this paper is structured as follows: Section 2.1 presents a brief survey of related work. Section 3.2 describes the problem. We present the methodology in Section 6.2 and results in Section 6.3. We conclude the paper in Section 6.4.

[2]https://wwwn.cdc.gov/nndss/infectious-tables.html

3.2 Proposed Approach

Given a set of N diseases $D = \{d_1, d_2, ..., d_N\}$, each disease d_i has a set $S = \{s_1, s_2, ..., s_K\}$ of K symptoms and their corresponding weight $W = \{w_1, w_2, ..., w_K\}$. The weight W of each symptom of a disease is computed from the HSDN disease-symptom dataset using the TF-IDF approach. A TF-IDF score indicates the level of association between a symptom and a disease. A symptom may be strongly linked to one disease but weakly connected to another. A higher TF-IDF score indicates a stronger disease-symptom connection.

Let G be a two-dimensional dataset for a given disease d_i. The columns are the symptoms of d_i. Note that the number of columns varies depending on the disease d_i as different diseases have different numbers of symptoms. The rows indicate the data for a disease d_i in a given week of the year, in a given U.S.A. state (there are 50 states in total). Hence, every row of G is a set $R = \{F_1, ..., F_K, label\}$ that represents a U.S.A. state during a given week of a year. The columns indicate *features* $F_{1,..,K}$ = *Google_ trends_ scores(s_K)*W(s_K)* which are computed using the Google trends scores (ranging from 0 to 100), for every symptom s_K of d_i (obtained from HSDN for a specific disease d_i) multiplied with the weight w_k (TF-IDF) of s_K (as computed from HSDN). The label $\in \{1, 0\}$ indicates whether or not the CDC confirmed and reported the outbreak of d_i in the given U.S.A state at that given week.

We propose a multi-disease outbreak detection framework ODF, such that, given N datasets $D_{train_{1,...,N}}$ and $D_{obs_{1,...,N}}$, the model $ODF_{train}(D_{train_{1,...,N}}) \longmapsto ODF_{classify}(D_{obs_{1,...,N}})$ can identify which instances in $D_{obs_{1,...,N}}$ are outbreaks.

3.3 Methodology

In this Section, we describe the methodology used to design, implement, and test the multi-disease outbreak detection framework. To build and test our framework, we selected 11 infectious diseases that are among the leading causes of illness and death in the United

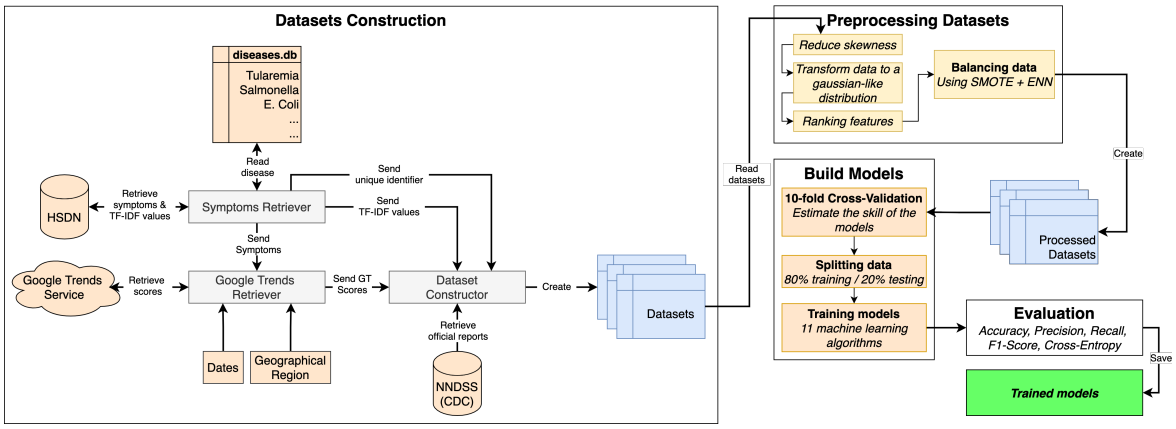

Figure 3.1: Overview of the proposed framework. We construct a dataset for each disease using the *symptom retriever*, *Google Trends retriever*, and *dataset constructor* modules. Symptoms, TF-IDF scores, Google Trends scores, and official CDC reports are incorporated into each dataset. All datasets are pre-processed to reduce skewness, generate Gaussian-like distributions, and balance datasets. The final datasets are used to train and evaluate the proposed framework.

States [80]: COVID-19, Tularemia, Salmonella, E. Coli, Brucellosis, Influenza, Hepatitis A, Hepatitis B, Hepatitis C, Dengue, and Meningitis. The main causative agents of these diseases are either bacteria or viruses, as illustrated in Table 3.1. Some diseases are emerg-

Table 3.1: Diseases, causes, type, and No. of cases reported by CDC in 2019. The number of cases for COVID-19 was retrieved for the period between 01-11-2020 to 07-08-2020.

Disease	Caused by	Emerging	Re-emerging	No. of Cases
Tularemia	Bacteria	✓	✓	48
Salmonella	Bacteria		✓	1029
E. Coli	Bacteria	✓		900
Brucellosis	Bacteria		✓	37
COVID-19	Virus	✓		≈ 12 Million
Influenza	Virus		✓	110
Hepatitis C	Virus	✓		322
Hepatitis B	Virus	✓		325
Hepatitis A	Virus		✓	811
Dengue	Virus	✓		57
Meningitis	Virus/Bacteria	✓		109

ing (e.g., COVID-19), whereas others are re-emerging (e.g., Salmonella). We selected the data from the entire year of 2019 (i.e., 52 weeks) for 10 out of the 11 diseases. At the time we started working on this framework, COVID-19 was emerging. Hence, we selected six months of data (from 01-11-2021 to 07-08-2020) for COVID-19. The number of cases for each disease in the selected timeline varies widely, ranging anywhere from 37 cases in a year (e.g., Brucellosis) to 12 million cases in four months (e.g., COVID-19).

Our work leverages the HSDN dataset. HSDN uses large-scale medical bibliographic records and the related Medical Subject Headings (MeSH) terms from PubMed [81], to generate a symptom-based network of human diseases. We, however, do not use the network structure of the dataset; we only extract the symptoms related to a disease. Figure 3.1 shows the overview of our framework. The framework consists of four modules: a module for dataset construction (Section 3.3.1), a module for pre-processing the dataset (Section 3.3.2), a module for building the multi-disease outbreak detection models (Section 3.3.3), and lastly, an evaluation module (Section 3.3.4). The following subsections describe these modules in more detail.

3.3.1 Datasets Construction

We integrated disparate data from two different sources to generate the dataset that builds the core of this work: 1) static data of standardized medical terms (i.e., symptoms of the diseases), and 2) dynamic data (i.e., Google trends scores) to indicate the interest in those symptoms at a given time, in a given location. The resultant dataset is then used to detect possible outbreaks of infectious diseases at a given time and in a given state (i.e., U.S.A. states). The static symptoms data were obtained from the Human Symptoms-Disease Network (HSDN) [79]. Instead of using the raw frequency of symptoms in HSDN, we used the TF-IDF scores of the symptoms to reflect the strength of the association between symptoms and diseases. We incorporated dynamic data into our dataset by using symptoms as search terms in Google Trends (GT). We utilized the GT scores to consider people's interest in various diseases and their symptoms across all the U.S.A states and

selected time periods. The integrated data (HSDN+GT) formed the core of the dataset that is used later in our framework.

Each one of the selected 11 diseases has its own dataset composed of weekly data as rows and symptoms (i.e., features) as columns. We developed three sub-modules that aid in the construction of these datasets (See Fig. 3.1): 1) *symptoms retriever*, 2) *Google Trends retriever*, and 3) *dataset constructor*.

3.3.1.1 Symptoms Retriever Module

The primary task of this module is to retrieve all the symptoms for each disease from HSDN. The secondary task is to compute the TF-IDF score of the symptoms for a specific disease (from PubMed) and share the output with two other modules. The symptoms data is sent to the *Google Trends Retriever module* for retrieving the GT score of that symptom in the given timeline at the 50 states. *The Dataset Constructor module* receives the disease name, associated symptoms, and the corresponding TF-IDF scores for each symptom of the disease.

3.3.1.2 Google Trends Retriever Module

This module receives a set of symptoms from the *symptoms retriever* module for a given disease. In addition, the module uses the *dates* and *geographical regions* files to determine the timeline and states for retrieving the GT scores. The dates file consists of all the weekly dates of interest (e.g., 52 weeks from 2019 for ten diseases, and 25 weeks of 2020 for COVID-19). The geographical regions file contains a list of the 50 U.S.A states. Although we use a *state-level resolution*, Google Trends allows other resolutions (e.g., subregion, metro, or city) as well.

Given the disease, symptoms, dates, and regional data, this module retrieves the GT score through the *official Google Trends API*[3]. The GT score is returned as a list of lists

[3]https://trends.google.com/

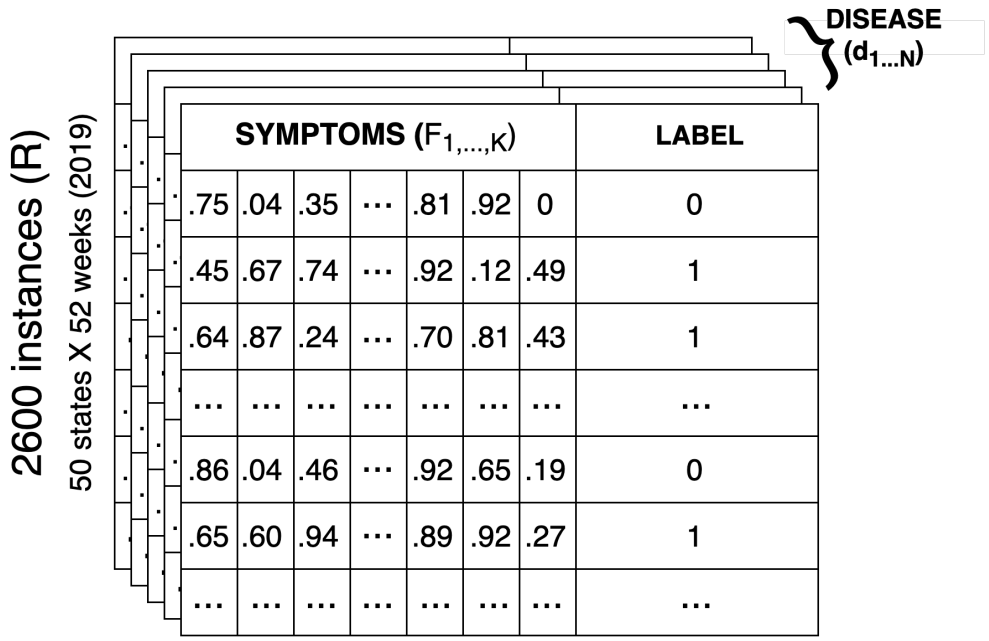

Figure 3.2: The diseases datasets $d_{1,...,N}$ contain 2600 rows (52 weeks for each of the 50 states) R, with k columns. Each column $F_1, ..., k$ contains the multiplication of normalized GT scores of a symptom for disease d_i (in a given week, at a given state) with the TF-IDF score of the symptom in regard to d_i (from HSDN). The label represents if the instance is reported as an outbreak by the NNDSS from CDC (e.g., 1 is outbreak).

(one row per week, per state). This list is then sent to the *Dataset Constructor Module* for further processing.

3.3.1.3 Dataset Constructor Module

This module performs the last step in the creation of the datasets. It retrieves the unique identifier (UI) of one disease at a time along with TF-IDF values of the disease-specific symptoms from the *Symptoms Retriever Module*. This module also receives the GT scores from the *Google Trends Retriever Module*. The official CDC reports are retrieved from the

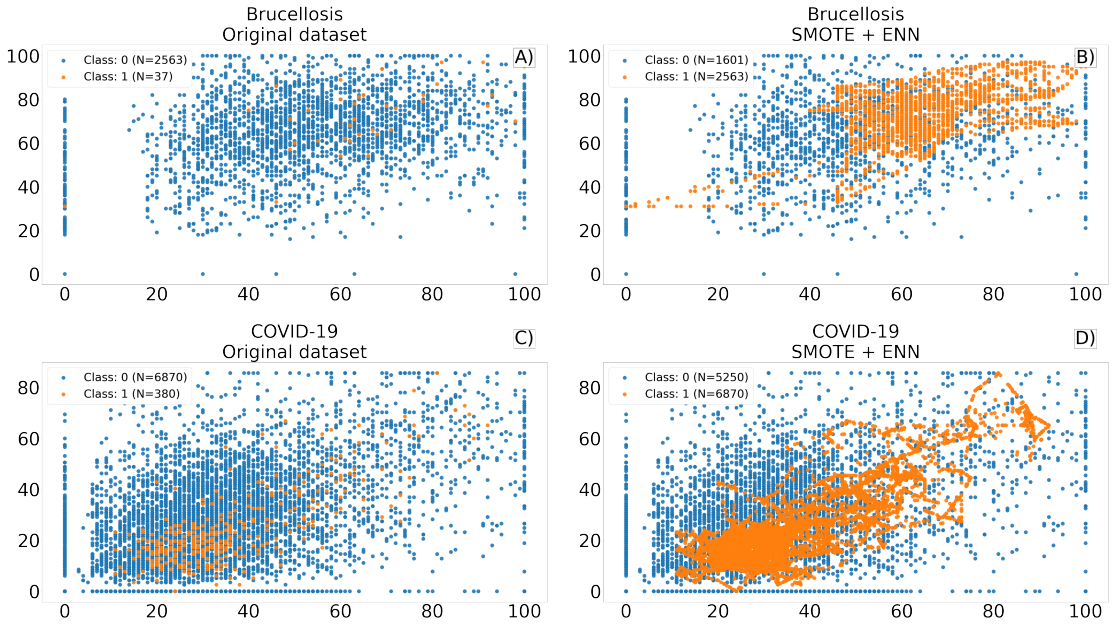

Figure 3.3: (Left) Brucellosis and COVID-19 raw datasets with imbalanced classes (A and C), (Right) Pre-processed datasets with reduced class imbalance after using SMOTE + ENN (B and D).

National Notifiable Diseases Surveillance System (NNDSS)[4].

For each disease d_i, the module constructs a bidimensional dataset. The datasets consist of instances (i.e., rows) $R = \{F_1, ..., F_K, label\}$, where $F_{1,..,K}$ are the features (i.e., columns) represented by the Google Trends score for each symptom of d_i (in a given week, in a given state) multiplied by the TF-IDF score of that symptom of d_i (retrieved from HSDN). We used TF-IDF scores since, although diseases share some symptoms, some symptoms are strongly associated with certain diseases, whereas that same symptoms may be weakly associated with another disease.TF-IDF score can detect this strength of the disease-symptom connection. Lastly, the label column in each row indicates whether or not the CDC has officially recognized an outbreak of that disease, in that week, in that state. Figure 5.3 shows a visual representation of this dataset.

[4]https://www.cdc.gov/nndss/index.html

3.3.2 Preprocessing Datasets

The disease-specific datasets generated in Section 3.3.1, presented several challenges including, *imbalanced datasets*, *overlapping classes*, and *skewed datasets*. We addressed the challenges of *imbalanced datasets* and *overlapping classes* using oversampling and undersampling methods. These two methods are highly effective when used in combination [82] to address dataset imbalance. We tested two combinations, the Synthetic Minority Oversampling TEchnique (SMOTE) [83] + the Edited Nearest Neighbors (ENN) and SMOTE + TOMEK-Links [84]. The combination of SMOTE + ENN produced better results.

Figures 3.3(A) and 3.3(C) show the original imbalanced datasets for the COVID-19 and Brucellosis diseases. Figures 3.3(B) and 3.3(D) show the new balanced datasets using SMOTE + ENN. The blue dots represent instances labeled as no outbreak, and the orange dots are instances labeled as an outbreak. We also applied a log-transformation to reduce the *skewness* of the original datasets. The log-transformation method is widely used to reduce the skewness and consequently transform the data to a Gaussian-like distribution [85].

3.3.3 Building the models

We conducted a 10-fold cross-validation during the modeling process. The cross-validation (CV) technique is used for testing the effectiveness of machine learning models and also for improving models with limited data. We split our datasets with a ratio of 80% training and 20% testing. We followed the scaling law for the validation-set training-set size ratio proposed by Guyon [86].

We built and tested the disease models using several machine learning (ML) algorithms. Our goal was to develop one framework that can detect the outbreaks of multiple infectious diseases. This can be rephrased as an ML problem for binary classification. A number of ML algorithms exist to address such problems. We selected 11 algorithms for further study, based on the following factors:

1. **Size of the training data:** While it is essential to collect as much data as possible to develop reliable models, there are no well-defined rules on how much data is sufficient to make such models [87], [88]. The raw datasets used in this study contain 2600 instances (i.e., 52 weeks of data for 50 states) for 10 out of the 11 diseases, and 1250 instances for COVID-19 (i.e., 25 weeks of data for 50 states). Note that the number of features (i.e., symptoms) is different for each disease - ranging from 14 for Tularemia to 109 for Human Influenza. Considering the size of our datasets, we decided to use low bias/high variance algorithms, such as K-Nearest Neighbor (KNN), Support Vector Machines (SVM), and Decision Trees (DT).

2. **Accuracy and interpretability:** The difference between accuracy and interpretability is that accuracy measures the correctness of classification, while interpretability explains why a classification was made [89].

   The proposed framework is designed to be used by both experts and non-experts in computer science. Hence, it is important that the framework produces results that are both accurate and explainable. Therefore, we selected the following 11 algorithms which are likely to be accurate yet easy to interpret, for further study:

   - *Low Accuracy-High Interpretability:* Logistic Regression (LR) [90].
   - *Medium Accuracy-Medium Interpretability:* Gaussian Naive Bayes (GNB) [91], Decision Trees (DT) [92], K-Nearest Neighbor (KNN) [93], Gaussian Process Classifier (GPC) [94], Stochastic Gradient Descent (SGD) [93], Linear Discriminant Analysis (LDA) [95].
   - *High Accuracy-Medium Interpretability:* XGBoost [93], Support Vector Machine (SVM) [93].
   - *High Accuracy-High Interpretability:* Random Forest (RF) [93].
   - *High Accuracy-Low Interpretability:* Multi-Layer Perceptron-Neural Network ($MLP\text{-}NN$) [93].

3.3.4 Evaluation Metrics

There are a variety of performance metrics used to evaluate the quality of learning methods and generated disease models. As part of our interest in minimizing the number of classification errors, we used precision, recall, F1-measure, and accuracy, which are categorized as threshold metrics by Ferri, et al. [96]. Additionally, we used the cross-entropy loss (e.g., probability metrics) to analyze the detected class probabilities. A cross-entropy value of 0.0 indicates a perfect detection and probably overfitting [97]. In our analysis, we considered any cross-entropy score greater than 0.0 and less than 0.3 as good classification performance. We also used the Pearson correlation coefficient to investigate possible correlations between these metrics and some properties of the diseases.

3.4 Results and Discussion

Our work focuses on developing a single framework for detecting multiple infectious disease outbreaks using standardized medical terms (e.g., symptoms) coupled with their Google Trends scores. Disease-specific standard symptoms play a key role in this framework. Hence it is important to check if the symptoms are indeed important for detecting outbreaks or if we can achieve similar performance with disease names only. We ran a test to check how the proposed approach compares to a baseline approach where only disease names and their Google Trend scores are used for outbreak detection. We compared these results with our proposed system that utilizes HSDN disease-specific symptoms, the TF-IDF scores of each symptom, and the Google Trend scores of those symptoms. Table 3.5 shows the F1-score of various algorithms when using the baseline approach (i.e., only disease name, no symptoms) and the proposed approach. The table shows the best F1-score of any of the 11 algorithms used. Results show that the proposed framework outperforms the baseline approach for detecting outbreaks for all diseases. This table clearly shows that disease-specific symptoms are better indicators of outbreaks than only disease names.

To better understand different aspects of the proposed framework, we seek to answer

Table 3.2: A comparison of the three best-performing algorithms for each disease using datasets constructed with *HSDN symptoms* vs. *related queries*. We report the accuracy, F1-score, and binary cross-entropy scores. The results indicate that the proposed framework performs better when using HSDN symptoms as primary indicators of outbreaks.

Disease	HSDN Symptoms		Related Queries		HSDN Symptoms		Related Queries		HSDN Symptoms		Related Queries	
	Algorithm	Accuracy	Algorithm		Algorithm	F1-Score	Algorithm		Algorithm	Cross-Entropy	Algorithm	
COVID-19	MLP-NN	**99.4**	98.1	MLP-NN	MLP-NN	**99.5**	98.2	MLP-NN	MLP-NN	**0.02**	0.06	MLP-NN
	SVM	99.2	97.9	XGBoost	SVM	99.3	98.1	XGBoost	SVM	0.03	0.06	XGBoost
	RF	99.0	97.8	SVM	RF	99.1	98.0	SVM	RF	0.11	0.07	SVM
Tularemia	MLP-NN	99.2	**99.7**	MLP-NN	MLP-NN	99.2	**99.7**	MLP-NN	MLP-NN	0.03	**0.02**	MLP-NN
	GP	98.2	99.0	RF	GP	98.4	99.0	RF	GP	0.31	0.08	RF
	RF	98.1	97.8	KNN	RF	98.3	98.0	KNN	RF	0.10	0.15	KNN
Salmonella	GP	87.2	**89.3**	GP	GP	92.2	**93.5**	GP	GP	0.44	0.59	GP
	SVM	87.2	87.2	KNN	SVM	92.1	92.3	KNN	SVM	**0.34**	1.26	KNN
	KNN	87.0	85.5	MLP-NN	KNN	91.9	91.1	MLP-NN	KNN	0.87	0.35	MLP-NN
Meningitis	SVM	**99.3**	97.6	MLP-NN	SVM	**99.5**	97.9	MLP-NN	SVM	**0.01**	0.07	MLP-NN
	MLP-NN	98.5	97.0	GP	MLP-NN	98.9	97.4	GP	MLP-NN	0.06	0.38	GP
	RF	97.7	96.2	RF	RF	98.3	96.70	RF	RF	0.12	0.18	RF
Influenza	SVM	**99.4**	99.0	SVM	SVM	**99.6**	99.1	SVM	SVM	**0.01**	0.03	SVM
	MLP-NN	99.0	99.0	MLP-NN	MLP-NN	99.3	99.1	MLP-NN	MLP-NN	0.04	0.04	MLP-NN
	XGBoost	98.4	98.5	RF	XGBoost	98.87	98.7	RF	XGBoost	0.07	0.09	RF
Hepatitis C	MLP-NN	95.3	**95.5**	GP	MLP-NN	**97.1**	96.3	GP	MLP-NN	0.14	0.43	GP
	SVM	95.2	94.9	MLP-NN	SVM	97.0	95.9	MLP-NN	SVM	**0.11**	0.17	MLP-NN
	RF	93.5	91.3	KNN	RF	95.9	93.2	KNN	RF	0.30	0.33	KNN
Hepatitis B	MLP-NN	**95.9**	93.7	MLP-NN	MLP-NN	**97.4**	95.1	MLP-NN	MLP-NN	0.18	0.20	MLP-NN
	SVM	94.0	93.1	GP	SVM	96.3	94.7	GP	SVM	**0.11**	0.44	GP
	RF	93.4	90.9	RF	RF	95.9	92.8	RF	RF	0.21	0.30	RF
Hepatitis A	GP	**97.2**	93.1	GP	GP	**98.4**	95.1	GP	GP	0.64	0.52	GP
	MLP-NN	96.9	90.6	MLP-NN	MLP-NN	98.2	93.2	MLP-NN	MLP-NN	0.14	0.30	MLP-NN
	SVM	96.5	89.4	KNN	SVM	98.0	92.5	KNN	SVM	**0.09**	0.83	KNN
E. Coli	GP	**95.5**	87.9	GP	GP	**97.6**	92.6	GP	GP	0.65	0.43	GP
	SVM	94.7	87.9	KNN	SVM	97.2	92.4	KNN	SVM	**0.16**	1.06	KNN
	MLP-NN	94.7	86.3	RF	MLP-NN	97.1	91.6	RF	MLP-NN	0.23	0.78	RF
Dengue	RF	**99.4**	99.3	RF	RF	**99.5**	99.3	RF	RF	0.07	0.06	RF
	SVM	99.3	99.2	KNN	SVM	99.4	99.2	KNN	SVM	**0.00**	0.09	KNN
	MLP-NN	98.7	99.0	MLP-NN	MLP-NN	98.9	99.0	MLP-NN	MLP-NN	0.03	0.05	MLP-NN
Brucellosis	SVM	**100.0**	97.6	RF	SVM	**100.0**	97.8	RF	SVM	**0.00**	0.15	RF
	MLP-NN	99.3	96.9	KNN	MLP-NN	99.4	97.2	KNN	MLP-NN	0.02	0.24	KNN
	RF	98.9	90.4	XGBoost	RF	99.1	91.4	XGBoost	RF	0.05	0.26	XGBoost

MLP-NN: Multilayer Perceptron Neural Network. **GP** = Gaussian Process. **RF** = Random Forest.
SVM = Support Vector Machine. **kNN** = K-Nearest Neighbors

the following questions in this section:

1. What is the effect of using HSDN disease-specific symptoms in detecting outbreaks of infectious diseases?
2. How do different weighting of symptoms affect the detection of infectious disease outbreaks?
3. What effect does the number of symptoms have on the detection of outbreaks?

3.4.1 The role of HSDN symptoms in detecting outbreaks of infectious diseases

The first question we seek to answer is: *what is the effect of using HSDN disease-specific symptoms in detecting the outbreaks of infectious diseases?* To answer this question, we tested our framework with 1) the original HSDN datasets that include disease-specific symptoms, and 2) a different disease dataset for each infectious disease that is built using related queries (instead of the HSDN disease-specific symptoms). For the second version of the dataset, we obtained an average of 20 related queries for each disease from the Google Trends API with the disease name as the input. For example, *"covid 19"* and *"covid symptoms"* are examples of related queries for the *COVID-19* disease.

Table 3.2 presents the results of this analysis. We list three evaluation metrics for each infectious disease to test the two approaches: accuracy, F1-score, and cross-entropy. We list the performance of the two versions we tested for each of the metrics: the proposed HSDN-symptoms-based framework and the related-query-based framework. For simplicity, we list the three best-performing algorithms for both versions. The best score of the three algorithms is highlighted with bold text.

In terms of accuracy, when using HSDN symptoms, the accuracy was higher (>95%) for 8 out of 11 diseases compared to the related-query-based approach. We observed a similar trend while analyzing the F1 scores, where the results were higher (>97%) with HSDN symptoms for 9 out of the 11 diseases. Note that even though we processed our datasets to reduce the data imbalance, the final datasets still exhibit a slight imbalance (see Figure 3.3). The higher F1 scores indicate that the proposed framework has low *false positives*, i.e., events that are mistakenly detected as outbreaks, and low *false negatives*, i.e., outbreaks that are not detected as outbreaks. This indicates that the proposed framework is effective in detecting outbreaks even with slightly imbalanced datasets.

For both accuracy and F1-score, when we check the performance of all the 11 algorithms, Support Vector Machine (SVM), Multi-Layer Perceptron Neural Network (MLP-

NN), Gaussian Process (GP), and Random Forest (RF) algorithms are consistently ranked among the top three algorithms. The *MLP-NN* algorithm had more than 95% accuracy in detecting outbreaks of *COVID-19* (99.4%), *Tularemia* (99.2%), *Hepatitis C* (95.5%), and *Hepatitis B* (95.9%). The *GP* algorithm achieved maximum accuracy in detecting outbreaks of *Salmonella* (89.7%), *Hepatitis A* (97.2%), and *E. Coli* (95.6%). The *SVM* algorithm was best in detecting *Meningitis*, *Influenza*, and *Brucellosis* outbreaks with 99.3%, 99.4%, and 100% accuracy, respectively. The *RF* and *SVM* algorithms were more accurate than the other algorithms in detecting *Dengue* outbreaks, with an accuracy of 99.4%.

The last metric in the table shows the cross-entropy value. Cross-entropy is a measure of the performance of a classification model whose output is a probability value between 0 and 1. The cross-entropy increases as the predicted probability diverges from the actual label. Therefore, a cross-entropy value close to 0 suggests that all predicted probabilities are closer to the target labels. In our experiments with the HSDN symptoms, the cross-entropy scores were lower (<0.34) than the similar query-based approach for 10 out of 11 diseases. As we notice in the cross-entropy values, the SVM algorithm has better performance at detecting outbreaks.

In the exceptional case of Salmonella, we achieved 89.3% accuracy (GP) with related queries compared to 87.2% (SVM) using HSDN symptoms. It represents a difference of 2.1%. Likewise, the F1-Score exhibited similar behavior, scoring 93.5% (GP) for related queries and 92.1% (SVM) for HSDN symptoms. It is a difference of 1.3%. However, as evidenced by the cross-entropy score, the HSDN symptom-based approach performed slightly better with the SVM algorithm. In particular, the SVM algorithm obtained a cross-entropy score of 0.34 when using symptoms, as opposed to 0.59 with GP - even though GP achieved the best performance in accuracy and F1-score. In other words, the SVM algorithm with HSDN symptoms is more reliable but less accurate in identifying outbreaks caused by Salmonella compared to the GP algorithm with related queries. Note that, while the MLP-NN algorithm has the lowest cross-entropy (0.35) using related queries, the accuracy of MLP-NN was lower (85.5%) than GP (89.3%). While MLP-NN is less accurate

Table 3.3: Accuray, F1-Score, and Cross-Entropy using datasets with the TF-IDF (weighted symptoms) and without TF-IDF (unweighted symptoms).

Disease	Using TF-IDF		Without TF-IDF		Using TF-IDF		Without TF-IDF		Using TF-IDF		Without TF-IDF	
	Algorithm	**Accuracy**		Algorithm	Algorithm	**F1-Score**		Algorithm	Algorithm	**Cross-Entropy**		Algorithm
COVID-19	MLP-NN	**99.4**	99.1	MLP-NN	MLP-NN	**99.5**	99.2	MLP-NN	MLP-NN	**0.02**	0.03	MLP-NN
	SVM	99.2	98.4	SVM	SVM	99.3	98.7	SVM	SVM	0.03	0.04	SVM
	RF	99.0	98.4	RF	RF	99.1	98.6	RF	RF	0.11	0.11	RF
Tularemia	MLP-NN	**99.2**	98.8	MLP-NN	MLP-NN	**99.2**	98.9	MLP-NN	MLP-NN	**0.03**	0.04	MLP-NN
	GP	98.2	98.8	RF	GP	98.4	98.8	RF	GP	0.31	0.09	RF
	RF	98.1	98.7	GP	RF	98.3	98.8	GP	RF	0.10	0.29	GP
Salmonella	GP	87.2	**89.7**	KNN	GP	92.1	**93.5**	KNN	GP	0.44	1.02	KNN
	SVM	87.2	89.0	GP	SVM	92.1	93.0	GP	SVM	0.34	0.39	GP
	KNN	86.9	85.5	XGBoost	KNN	91.8	90.9	XGBoost	KNN	0.87	**0.33**	XGBoost
Meningitis	SVM	**99.3**	99.1	MLP-NN	SVM	**99.5**	99.3	MLP-NN	SVM	**0.01**	0.03	MLP-NN
	MLP-NN	98.5	98.5	SVM	MLP-NN	98.9	98.8	SVM	MLP-NN	0.06	**0.01**	SVM
	RF	97.7	98.5	RF	RF	98.2	98.8	RF	RF	0.12	0.11	RF
Influenza	SVM	**99.4**	98.2	SVM	SVM	**99.6**	98.7	SVM	SVM	**0.01**	**0.01**	SVM
	MLP-NN	99.0	98.2	MLP-NN	MLP-NN	99.3	98.7	MLP-NN	MLP-NN	0.04	0.05	MLP-NN
	XGBoost	98.4	98.2	RF	XGBoost	98.9	98.6	RF	XGBoost	0.07	0.10	RF
Hepatitis C	MLP-NN	95.3	**95.5**	MLP-NN	MLP-NN	**97.1**	97.0	MLP-NN	MLP-NN	0.14	0.19	MLP-NN
	SVM	95.2	94.9	SVM	SVM	97.0	96.6	SVM	SVM	**0.11**	0.13	SVM
	RF	93.4	94.5	GP	RF	95.9	96.4	GP	RF	0.30	0.54	GP
Hepatitis B	MLP-NN	**95.9**	**95.9**	MLP-NN	MLP-NN	**97.4**	97.3	MLP-NN	MLP-NN	0.18	0.15	MLP-NN
	SVM	94.0	95.1	GP	SVM	96.3	96.8	GP	SVM	**0.11**	0.53	GP
	RF	93.4	94.8	SVM	RF	95.9	96.6	SVM	RF	0.21	**0.11**	SVM
Hepatitis A	GP	**97.2**	93.9	GP	GP	**98.4**	96.4	GP	GP	0.64	0.63	GP
	MLP-NN	96.9	93.0	MLP-NN	MLP-NN	98.2	95.8	MLP-NN	MLP-NN	0.14	0.25	MLP-NN
	SVM	96.5	92.1	XGBoost	SVM	97.9	95.3	XGBoost	SVM	**0.09**	0.21	XGBoost
E. Coli	GP	95.5	**95.6**	GP	GP	**97.6**	97.4	GP	GP	0.65	0.64	GP
	SVM	94.7	92.7	XGBoost	SVM	97.2	95.6	XGBoost	SVM	**0.16**	0.22	XGBoost
	MLP-NN	94.7	92.7	MLP-NN	MLP-NN	97.1	95.6	MLP-NN	MLP-NN	0.23	0.27	MLP-NN
Dengue	RF	**99.4**	**99.4**	SVM	RF	**99.5**	**99.5**	SVM	RF	0.07	**0.00**	SVM
	SVM	99.3	99.0	RF	SVM	99.4	99.1	RF	SVM	**0.00**	0.06	RF
	MLP-NN	98.7	98.9	MLP-NN	MLP-NN	98.9	99.0	MLP-NN	MLP-NN	0.03	0.02	MLP-NN
Brucellosis	SVM	**100.0**	99.8	SVM	SVM	**100.0**	99.8	SVM	SVM	**0.00**	**0.00**	SVM
	MLP-NN	99.3	99.3	MLP-NN	MLP-NN	99.4	99.4	MLP-NN	MLP-NN	0.02	0.02	MLP-NN
	RF	98.9	99.2	RF	RF	99.1	99.3	RF	RF	0.05	0.05	RF

MLP-NN=Multilayer Perceptron Neural Network. **GP**=Gaussian Process. **RF**=Random Forest.
SVM=Support Vector Machine. **KNN**= K-Nearest Neighbors

than GP, it is more reliable.

We achieve the best performance in detecting Hepatitis using HSDN symptoms. The proposed approach can detect and differentiate between three types of Hepatitis outbreaks: Hepatitis A, Hepatitis B, and Hepatitis C. These three variants of Hepatitis share 34 symptoms, including fever, fatigue, and diarrhea. Despite the overlap in symptoms, the proposed approach achieved better accuracies (>95%) and better F1 scores (>97%) than the related query-based approach.

Emerging diseases often share some similarities with previous diseases as seen in the

case of COVID-19. The COVID-19 and Influenza (A, B, C, and D) diseases are caused by similar RNA viruses (e.g., Alphainfluenzavirus, Betainfluenzavirus, Gammainfluenzavirus, and Deltainfluenzavirus). This group of viruses affects the respiratory tract, exhibits similar symptoms, and uses surface proteins to infect the host [98]. Based on the similarity of symptoms, we investigated which Influenza symptoms could be used to detect COVID-19 outbreaks. As a result of a literature search, we determined that 51 Influenza symptoms are also symptoms of COVID-19. Some of which include, delirium [99], hallucinations [100], nausea [101], catatonia [102], seizures [103], and arthralgia [104]. We used these symptoms and related queries to detect COVID-19 outbreaks. The results are shown in the first row of Table 3.2. Using HSDN symptoms as the primary indicators of outbreaks, the symptom-based approach outperformed the related queries approach, reaching maximum accuracy (99.4%) and F1-score (99.5%) with the lowest cross-entropy score (0.02). Overall, the HSDN symptoms performed better than the related queries in:

- 8 of 11 diseases using accuracy,
- 9 of 11 diseases using F1-score,
- 10 of 11 diseases using cross-entropy.

In conclusion, the proposed approach of using HSDN disease-specific symptoms with corresponding GT scores for outbreak detection has a better performance compared to the related queries-based approach.

3.4.2 The effect of weighted and unweighted symptoms in outbreak detection

The connections between the symptoms and disease play a crucial role in our approach. We use TF-IDF weighting to reflect the strength of the connection between a symptom and a disease. The TF-IDF score of a symptom of a disease is multiplied by the GT score of that symptom (in a given week, in a given state). To understand the effects of the TF-IDF

scoring in the proposed framework, in this section, we seek to answer the question: *how do different weighting of symptoms affect the detection of infectious disease outbreaks?*

As noted earlier, each disease has a set of symptoms, and a symptom can appear in multiple diseases. However, a symptom may have a strong connection with one disease but a weaker connection with another. In order to quantify the strength of the connection between a symptom and a disease, we utilize TF-IDF weighting, retrieved from HSDN. TF-IDF scores reflect how often a given symptom appears with a particular disease (in PubMed, as used by HSDN) compared to how frequently the symptom appears in all the diseases. A symptom that is frequently associated with a particular disease will have a high TF-IDF score. To examine the effects of the TF-IDF weighting, we compared the performance of the proposed framework with and without the TF-IDF weighting of the HSDN symptoms. We constructed 11 datasets (one for each disease) without TF-IDF scores (unweighted symptoms) and 11 datasets with TF-IDF scores (weighted symptoms) (see section 3.3.1). Table 3.3 summarizes the results using accuracy, F1-score, and cross-entropy scores of the three best-performing algorithms for both the TF-IDF weighing and unweighted symptoms.

As seen in this table, the proposed TF-IDF weighing reached the highest accuracy of 95% or more for *eight of the 11 diseases* (i.e., COVID-19, Tularemia, Meningitis, Influenza, Hepatitis B, Hepatitis A, Dengue, and Brucellosis). On the other hand, *five of the 11* diseases (i.e., Salmonella, Hepatitis C, Hepatitis B, E. Coli, and Dengue) reached the highest accuracy (89% or more) using unweighted symptoms. Note that both weighting approaches have similar accuracy for Dengue and Hepatitis B.

In terms of F1 scores, we observed better results with TF-IDF weighting, which achieved the highest F1 scores (97% or more) for 10 out of 11 diseases (i.e., except for Salmonella). Only for *Salmonella*, the unweighted symptoms-based approach achieved a higher F1 score (93.5% using KNN). Both weighting approaches had the same F1 score for *Dengue*.

According to cross-entropy, TF-IDF weighting performs equally well or better than the unweighted symptoms in 10 out of the 11 diseases. Only in the case of *Salmonella*, the

Table 3.4: A comparison of the three best-performing algorithms for each disease using datasets constructed with *symptoms*. We report the No. of symptoms, algorithms, precision, recall, accuracy, F1-score, and cross-entropy scores.

Disease	No. of Symptoms	Algorithm	Precision	Recall	Accuracy	F1-Score	Cross-Entropy
Influenza	**109**	**SVM**	99.2	100.0	99.4	99.6	0.01
		MLP-NN	98.6	100.0	99.0	99.3	0.04
		XGBoost	98.1	99.6	98.4	98.9	0.07
E. Coli	**82**	**GP**	95.3	100.0	95.5	97.6	0.65
		SVM	94.7	99.7	94.7	97.2	0.16
		MLP-NN	95.2	99.1	94.7	97.1	0.23
Brucellosis	**79**	**SVM**	100.0	100.0	100.0	100.0	0.00
		MLP-NN	98.8	100.0	99.3	99.4	0.02
		RF	98.8	99.4	98.9	99.1	0.05
Hepatitis A	**75**	**GP**	97.0	99.7	97.2	98.4	0.64
		MLP-NN	97.3	99.2	96.9	98.2	0.14
		SVM	96.8	99.2	96.5	98.0	0.09
Hepatitis C	**72**	**MLP-NN**	95.0	99.3	95.3	97.1	0.14
		SVM	94.8	99.3	95.2	97.0	0.11
		RF	95.0	96.7	93.5	95.9	0.30
Meningitis	**69**	**SVM**	99.0	100.0	99.3	99.5	0.01
		MLP-NN	97.8	100.0	98.5	98.9	0.06
		RF	97.4	99.2	97.7	98.3	0.12
Hepatitis B	**64**	**MLP-NN**	95.0	100.0	95.9	97.4	0.18
		SVM	92.8	100.0	94.0	96.3	0.11
		RF	93.7	98.2	93.4	95.9	0.21
Dengue	**53**	**RF**	99.4	99.6	99.4	99.5	0.07
		SVM	98.8	100.0	99.3	99.4	0.01
		MLP-NN	97.8	100.0	98.7	98.9	0.04
SARS-CoV-2	**51**	**MLP-NN**	99.0	100.0	99.4	99.5	0.02
		SVM	98.7	99.7	99.1	99.3	0.03
		RF	99.3	99.0	99.0	99.1	0.11
Tularemia	**22**	**MLP-NN**	98.5	100.0	99.2	99.2	0.03
		GP	96.8	100.0	98.2	98.4	0.31
		RF	97.7	98.9	98.1	98.3	0.10
Salmonella	**14**	**GP**	87.4	97.4	87.2	92.2	0.44
		SVM	87.8	96.8	87.2	92.1	0.34
		KNN	88.7	95.2	87.0	91.8	0.88
		Correlations	**0.48**	**0.57**	**0.51**	**0.55**	**-0.02**

MLP-NN=Multilayer Perceptron Neural Network. **GP**=Gaussian Process. **RF**=Random Forest.
SVM=Support Vector Machine. **KNN**= K-Nearest Neighbors

unweighted symptom approach had better cross-entropy than the TF-IDF weighting. While the performance of both weighted (i.e., TF-IDF) and unweighted symptoms are similar,

weighted symptoms resulted in a higher score in accuracy and F1 measure. Results indicate that the inclusion of TF-IDF will ensure better detection of infectious disease when used with the GT score. However, if symptoms are not well-measured, the proposed approach still can be used to detect the outbreak of infectious diseases – with lesser accuracy.

3.4.3 The effect of the number of symptoms in outbreak detection

In our study, the symptoms are used as the primary indicator for detecting infectious disease outbreaks. However, it is important to note that different disease has different types and numbers of symptoms. Among the 11 diseases we studied, the number of symptoms varied from 14 for Salmonella to 109 for Human Influenza. In an effort to understand if there is any relation between the number of symptoms and the performance of the proposed detection approach, we carried out further tests. In this section, we seek to answer the question: *what effect does the number of symptoms have on the detection of infectious disease outbreaks?*

Table 3.4, sorted based on the number of symptoms, shows the result of the analyses. The table presents the disease, the number of symptoms, and three of the best-performing algorithms (out of the 11 algorithms). We present the precision, recall, accuracy, F1-score, and cross-entropy data for each of the three algorithms. A gray highlight indicates the best performance in any metric. Among the 11 infectious diseases we studied, Salmonella has the lowest number of symptoms (i.e., 14). We notice that the algorithms had the worst performance for Salmonella with the lowest F1-score and accuracy scores. The best cross-entropy value for Salmonella is 0.34, which is the worst *best cross-entropy* value for any disease studied. Note that, *E. Coli* has a cross-entropy value of 0.65 (using GP), however, the best cross-entropy score for *E. Coli* is 0.16 (using SVM). Salmonella also had the lowest precision and lowest recall scores among all the diseases.

On the other side, diseases with a higher number of symptoms are showing better performances. Human Influenza has the highest number of symptoms (i.e., 109) and obtains the second-best F1-score of 99.6% using the SVM algorithm. The precision and recall for Influenza ranged from 99 to 100%. The disease with the second-highest number of

symptoms (E. Coli with 82 symptoms) also strongly performs in all metrics. However, with E. Coli, we notice that the best-performing algorithm in terms of F1 score and accuracy has the worst cross-entropy score (GP). In this case, SVM achieves the best cross-entropy value (0.16) and has an F1 score similar to GP. We computed Pearson's correlation to measure any association between the number of symptoms and the metrics. The results indicate that there is a moderate positive association between the number of symptoms and accuracy (0.51), as well as with F1 scores (0.55) (Table 3.4, last row). The correlations indicate that when the number of symptoms increases, the framework's performance also increases.

The correlation data for cross-entropy and the number of symptoms portray a very different picture. In this case, the correlation coefficient is -0.02, which indicates a barely negative correlation. While the negative correlation was expected (since a lower cross-entropy value is better, we expected lower cross-entropy with a higher number of symptoms), the low magnitude raised questions. One reason for the lower magnitude could be the fact that the algorithm with the best cross-entropy does not necessarily have the best accuracy or F1 score (e.g., E. Coli, Hepatitis A, Dengue) (Table 3.4). In many cases, the best cross-entropy comes from the second-best algorithm (e.g., E. Coli, Hepatitis C). This could have resulted in a comparatively lower correlation coefficient with cross-entropy. In terms of algorithms, SVM, MLP-NN, and RF repeatedly appeared among the top three algorithms. However, as pointed out earlier in Table 3.4, any particular metric alone may not be the most reliable indicator of performance. When cross-entropy is taken into consideration, we notice that SVM has the best performance in 9 out of the 11 diseases. This indicates that among all the algorithms tested, SVM is more reliable in detecting outbreaks. Table 3.5 shows the performance of SVM for all the diseases along with the correlation metric. This shows SVM has a moderate correlation with F1 score (0.6) and a weak correlation with cross-entropy (-0.4). Note that a negative correlation in cross-entropy indicates that with a higher number of symptoms we achieve better cross-entropy (lower cross-entropy is desirable). We also ran a similar test with the MLP-NN algorithm since it also appears

Table 3.5: Correlations between the number of symptoms and evaluation metrics for the SVM algorithm.

Disease	No. of symptoms	Precision	Recall	Accuracy	F1-Score	Cross-Entropy
Influenza	109	99.2	100.0	99.4	99.6	0.010
E. Coli	82	94.7	99.7	94.7	97.2	0.160
Brucellosis	79	100.0	100.0	100.0	100.0	0.000
Hepatitis A	75	96.8	99.2	96.5	98.0	0.090
Hepatitis C	72	94.8	99.3	95.2	97.0	0.110
Meningitis	69	99.0	100.0	99.3	99.5	0.010
Hepatitis B	64	92.8	100.0	94.0	96.3	0.110
Dengue	53	98.8	100.0	99.3	99.4	0.010
COVID	51	98.7	99.7	99.1	99.3	0.030
Tularemia	22	97.4	95.4	100.0	97.7	0.018
Salmonella	14	87.8	96.8	87.2	92.1	0.340
Correlations		**0.5**	**0.8**	**0.4**	**0.6**	**-0.4**

frequently in the top three algorithms (Table 3.6). As we see from Table 3.6, the overall performance of MLP-NN is slightly lower than SVM. Overall, various data show that the number of symptoms of an infectious disease plays an important role in outbreak detection – the more symptoms there are, the better the detection. We also observed that of all the algorithms tested, SVM has the best performance for detecting outbreaks.

Table 3.6: Correlations between the number of symptoms and evaluation metrics for the MLP-NN algorithm.

Disease	No. of symptoms	Precision	Recall	Accuracy	F1-Score	Cross-Entropy
Influenza	109	98.6	100.0	99.0	99.3	0.040
E. Coli	82	95.2	99.1	94.7	97.1	0.230
Brucellosis	79	98.8	100.0	99.3	99.4	0.020
Hepatitis A	75	97.3	99.2	96.9	98.2	0.140
Hepatitis C	72	95.0	99.3	95.3	97.1	0.140
Meningitis	69	97.8	100.0	98.5	98.9	0.060
Hepatitis B	64	95.0	100.0	95.9	97.4	0.180
Dengue	53	97.8	100.0	98.7	98.9	0.040
COVID	51	99.0	100.0	99.4	99.5	0.020
Tularemia	22	99.0	100.0	99.4	99.5	0.020
Salmonella	14	84.7	87.5	93.6	90.4	0.364
Correlations		**0.5**	**0.6**	**0.2**	**0.5**	**-0.3**

3.5 Conclusions and Future Work

This paper presents an approach to integrate static and dynamic data to detect outbreaks of multiple infectious diseases. We investigated if disease-specific symptoms retrieved from a medical database are useful in detecting the outbreaks of those diseases. We also studied the effects of weighting the symptoms in an effort to reflect the strength of the connection between a symptom and a disease. Furthermore, we investigated if the number of symptoms of a disease affects the accuracy of outbreak detection.

Our study highlighted that disease symptoms are better detectors of outbreaks compared to disease names (Table 3.7). Studies also highlighted that HSDN symptoms are better indicators of infectious disease outbreaks than related queries (of a disease) retrieved from Google Trends. However, in scenarios where disease symptoms may be lacking in HSDN (e.g., with the emergence of a new infectious disease such as COVID-19), related queries can be used, as we notice that related query-driven approach is also able to detect outbreaks with considerable accuracy (Table 3.2). Each infectious disease has multiple

Table 3.7: Baseline vs. the proposed approach. Baseline is computed using disease name and their Google Trend scores.

	F1-Score	
Disease	**Baseline**	**Proposed Approach**
COVID-19	73.8	**99.5**
Tularemia	69.9	**99.2**
Salmonella	75.4	**99.2**
Meningitis	85.8	**99.5**
Influenza	77.3	**99.6**
Hepatitis C	71.9	**97.1**
Hepatitis B	72.2	**97.4**
Hepatitis A	86.4	**98.4**
E. Coli	48.7	**97.6**
Dengue	59.6	**99.5**
Brucellosis	40.7	**100.0**

symptoms. However, different diseases can share a number of those symptoms. A symptom

can be more critical in one disease than another. We studied the effects of disease-symptom connection strengths using TF-IDF weighting. The TF-IDF score of a symptom for a disease indicates how frequently the symptom appears only with that disease. As shown in Table 3.3, TF-IDF-based weighting generated better F1 scores (more than 97%) and cross-entropy values (less than 0.11), for 10 of 11 diseases. However, the performance of unweighted symptoms is close to the TF-IDF results. This indicates that while TF-IDF weighting produces better results, unweighted symptoms can also be used for detecting outbreaks. Regarding the number of symptoms used to detect outbreaks, we notice that a higher number of symptoms is usually associated with better accuracy and F1 scores. The Pearson correlation coefficient also suggests a moderate correlation between the number of symptoms and the F1 scores, indicating outbreaks of diseases with more symptoms are easier to detect.

Of the 11 infectious diseases, the lowest outbreak detection performance comes from Salmonella (F1-score is 92.2%). Other diseases with low to mid-level F1 scores (ranging from 92% to 98%) include Salmonella, E. Coli, and the Hepatitis disease family. The proposed framework achieves F1 scores of 95% or higher for most of the diseases. This indicates that the proposed approach performs well for different types of infectious diseases (e.g., viral, bacterial) and can be used for detecting outbreaks of these diseases.

Even though the framework used 11 algorithms, this paper reports the top three algorithms for each disease. The SVM and MLP-NN algorithms appear the most among the top three algorithms (i.e., ten times). Next, the RF algorithm appears seven times. Some of the other algorithms appearing among the top three include GP, KNN, and XGBoost. Our analyses suggested SVM has the overall best performance among the algorithms tested.

The proposed approach and corresponding studies strongly suggest that disease-specific symptoms, coupled with the GT score, can be used to detect outbreaks of a number of infectious diseases. The proposed approach does not need data collection at the field level yet integrates dynamic data generated by Google search engine users. Even though we tested the framework with U.S.A. states, the proposed approach can be used by any other

region of the world. As for the next step, we are currently extending the framework to include contextual features (e.g., population, human mobility, race). We also plan to study the network structure between the symptoms and diseases in building disease models. It is also in the future plan to implement a web interface of the proposed framework.

Chapter 4

Human Mobility Driven Modeling of an Infectious Disease[1]

4.1 Introduction

Human mobility is an integral part of everyday life. It provides valuable insights into the movement patterns and behavior of individuals or groups within a specific area or across various locations. This data encompasses diverse information, such as location data, which tracks individuals' geographic coordinates over time, revealing details about where people reside, work, travel, and spend their time. However, Mobility can expose individuals to a variety of health risks, including infections from contagious diseases such as COVID-19. Yet, existing systems to model and predict the spread of infectious diseases do not include mobility in all aspects of the models. For example, the models described in papers [23,25–30] rely on the core deterministic epidemic model SEIR (Susceptible, Exposed, Infected, and Recovered) but include mobility only on a subset of the SEIR components.

In the United States, the mortality statistics related to COVID-19 show that the death rate differs in different ethnic groups. For example, 256 Indigenous Americans, 180 African Americans, 177 Pacific Islander Americans, 150 White Americans, 147 Latino Americans, and 96 Asian Americans per 100,000 inhabitants died due to COVID-19 [5]. Another significant factor influencing COVID-19 mortality rates is age. For example, mortality rates may reach 20% among people over the age of 80 [19]. Although there have been clear links between COVID-19 and age and race factors, research linking poverty levels to

[1]https://doi.org/10.1109/ICDMW58026.2022.00155

infectious diseases has been relatively scarce. In terms of income inequality, an increase of 1% in the Gini coefficient (a measure of inequality in a population) is directly related to a 4% increase in COVID-19 cases per million, and a 5% increase in COVID-19 deaths per million [6].

This paper presents our model called SEIRD+m, which adds a new compartment D in the SEIR model and includes human mobility aspects in all its components. Our model can be used to study how the mobility restriction on COVID-19 hotspots based on demographic factors such as age, and race, can reduce the number of infections, and death counts due to COVID. We provide a detailed analysis of the effects of the model on the spread of COVID-19 at the level of the Census Block Group (CBG). A CBG is one of the smallest aggregation units for an area – usually containing 600 to 3,000 individuals – for which the US Census Bureau provides demographic information. The **key contributions** of this paper are:

1. We include a human-mobility-driven D compartment in the SEIR model, making it SEIRD. D describes the dynamics of the transition from Infected (I) to Dead (D) based on human mobility. While there are efforts to include the D component into the SEIR model [105, 106], the inclusion of human mobility with D is novel, to the best of our knowledge.

2. We incorporate human mobility into all the Ordinary Differential Equations (ODEs) of our SEIRD model's components.

3. We provide a case study using our SEIRD+m model to **restrict human mobility** at detected **COVID-19 hotspots** using social factors, such as race, age, and income.

4. , Unlike other models, in our model human mobility is restricted in the COVID-19 hotspots rather than the entire region, which will save resources and keep additional services functioning alongside the essential ones.

4.2 Methodology

In this section, we present our proposed SEIRD+m model in detail. The following subsections explain the entire process, from data acquisition to model building.

4.2.1 The Proposed SEIRD+m Model

The SEIR (Susceptible, Exposed, Infected, Removed) model is an extended version of the SIR (Susceptible, Infected, Removed) model [107] that incorporates the incubation period E (Exposed) of an infectious disease [108]. However, the SEIR model does not include human mobility, even though it is associated with the exposure and spreading of an infectious disease. We enhanced the SEIR model by incorporating human mobility (m) and deaths (D). This mobility data was extracted from the Safegraph mobility data which is measured through the collection and analysis of anonymized location data from various sources, including mobile devices and applications. We start from the premise that people who die due to COVID-19 will die after they are infected with COVID-19. People can also recover from the disease once they are infected. Therefore, our model must transition from I (infection) to D (death) and from I (infection) to R (recovery) (See Fig. 4.2).

In addition, it is necessary to consider a new parameter – ρ, representing how quickly a person dies – to quantify the period between infection and death. Quantifying the period between infections and deaths will help estimate a person's probability of going from the model's Infected (I) to the Dead (D) compartment. As an example of the parameter ρ, if a person takes ten days to move from I to D, then $\rho = 1/10$. The probability of going from I to D is equal to the *death rate* (death cases per unit of time), represented by α. The complete transition from I to D is shown in Eq. 4.5. Finally, the probability of going from I to R (recovery) is 1-α. The complete transition is shown in Eq. 4.4. The extended SEIRD version is represented by Eqs. (4.1) to (4.5).

The total susceptible population is:

$$\frac{dS}{dt} = -\beta \cdot I \cdot \frac{S}{N} \tag{4.1}$$

In this equation, β is the expected number of people an infected person infects per day, S is the number of people that can be infected, I is the number of infected people, and N is the size of the population, computed over every timestamp of the data.

The following equation gives the total exposed population over time. The exposed state refers to the condition when a person is infected, but the virus is incubating; hence, the person is not yet experiencing symptoms.

$$\frac{dE}{dt} = \beta \cdot I \cdot \frac{S}{N} - \delta \cdot E \tag{4.2}$$

In this equation, δ is the length of the incubation period, and E is the number of exposed people.

The following equation gives the total number of infected people over time.

$$\frac{dI}{dt} = \delta \cdot E - (1 - \alpha) \cdot \gamma \cdot I - \alpha \cdot \rho \cdot I \tag{4.3}$$

In Eq. 4.3, γ is the proportion of infected people recovering per day.

The following equation provides the total number of recovered people over time.

$$\frac{dR}{dt} = (1 - \alpha) \cdot \gamma \cdot I \tag{4.4}$$

The next equation gives the total number of deaths computed over time.

$$\frac{dD}{dt} = \alpha \cdot \rho \cdot I \tag{4.5}$$

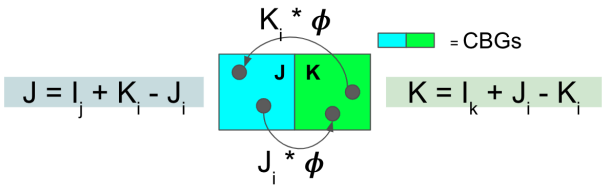

Figure 4.1: An example of incorporating mobility between two CBGs for the **I** compartment of the SEIRD model.

The SEIRD model predicts COVID-19 evolution over time in each CBG separately. The SEIRD model views each CBG as *an isolated region* even though individuals move

between them. To incorporate the dynamic aspect of human mobility, we need to consider the inflow and outflow traffic of each CBG.

We define human mobility (m) as a function that provides information about the movement patterns of individuals within a specific area or across various locations. In this work, human mobility (m) captures diverse information, including location data (L), which tracks the geographic coordinates of individuals (C) over time (t), revealing details about their places of residence (R), work (W), travel (T), and activity locations (A). Mathematically, we can express it as: m = f(L, C, t, R, W, T, A)

Now, let us consider a region with only two CBGs, J and K (Fig. 4.1). People travel from J to K without any mobility restrictions and vice versa. During this busy period of mobility, individuals belong to a group—i.e., Susceptible, Exposed, Infected, or Recovered. For example, a fraction of individuals infected in J (J_i) travel to K, and a fraction of individuals infected in K (K_i) travel to J. Now, K has $I_K + J_i - K_i$ infections, and J has $I_J + K_i - J_i$ infections. The same behavior occurs for the other groups (S, E, R).

Let us consider the same region with imposed mobility restrictions. The restriction is quantified by ϕ in a range between 0 and 1, where 0 is a total restriction, and 1 means no restrictions. The COVID-19 dynamics are likely to change with the imposed restrictions. For example, a fraction of individuals infected in CBG J (J_i) travel to CBG K, and a fraction of individuals infected in K (K_i) travel to J. Now, K has $I_K + (J_i - K_i)\phi$ infections and J has $I_J + (K_i - J_i)\phi$ infections. In other words, the number of infected individuals traveling between CBGs is restricted by ϕ. The same behavior occurs for the other compartments of the model (S, E, and R).

We incorporate human mobility into our SEIRD model (Eqs. (4.1) to (4.5)) for a complex geographical region containing many CBGs. The geographical region in our model is represented by a QxQ matrix, where Q is the number of CBGs within the region. The mobility is represented by a TxQxQ Origin-Destination matrix, where T is the number of timestamps (in units of days).

The following three Eqs. describe the human mobility of Susceptible M_s, Exposed M_e,

and Infected M_i people traveling from the CBG j to the CBG k at a time t.

$$M_s = \sum_{k=0}^{Q} s_{k,j}^t - \sum_{k=0}^{Q} s_{j,k}^t \tag{4.6}$$

$$M_e = \sum_{k=0}^{Q} e_{k,j}^t - \sum_{k=0}^{Q} e_{j,k}^t \tag{4.7}$$

$$M_i = \sum_{k=0}^{Q} i_{k,j}^t - \sum_{k=0}^{Q} i_{j,k}^t \tag{4.8}$$

Eq. 4.9 provides the total number of Susceptible individuals, considering human mobility over time.

$$S_{j,t+1} = -\beta \cdot (I + (M_i) \cdot \phi^t) \cdot \frac{(S + (M_s) \cdot \phi^t)}{N} \tag{4.9}$$

In Eq. 4.9, M_i and M_s are Infected and Susceptible individuals traveling from the CBG j to the CBG k at a time t (Eqs. 4.6 and 4.8) and ϕ^t is the strength of the mobility restriction at time t.

The following equation, Eq. 4.10, gives the total Exposed population over time.

$$\begin{aligned} E_{j,t+1} = \beta \cdot (I + (M_i) \cdot \phi^t) \cdot \frac{(S + (M_s) \cdot \phi^t)}{N} \\ -\delta \cdot (E + (M_e) \cdot \phi^t) \end{aligned} \tag{4.10}$$

M_e is the Exposed individuals (Eq. 4.7) traveling from the CBG j to the CBG k at time t.

The following equation provides the total number of Infected people, considering human mobility over time.

$$\begin{aligned} I_{j,t+1} = \delta \cdot (E + (M_e) \cdot \phi^t) \\ -(1-\alpha) \cdot \gamma \cdot (I + (M_i) \cdot \phi^t) - \alpha \cdot \rho \cdot (I + (M_i) \cdot \phi^t) \end{aligned} \tag{4.11}$$

The following equation provides the total number of Recovered people considering human mobility over time.

$$R_{j,t+1} = (1-\alpha) \cdot \gamma \cdot (I + (M_i) \cdot \phi^t) \tag{4.12}$$

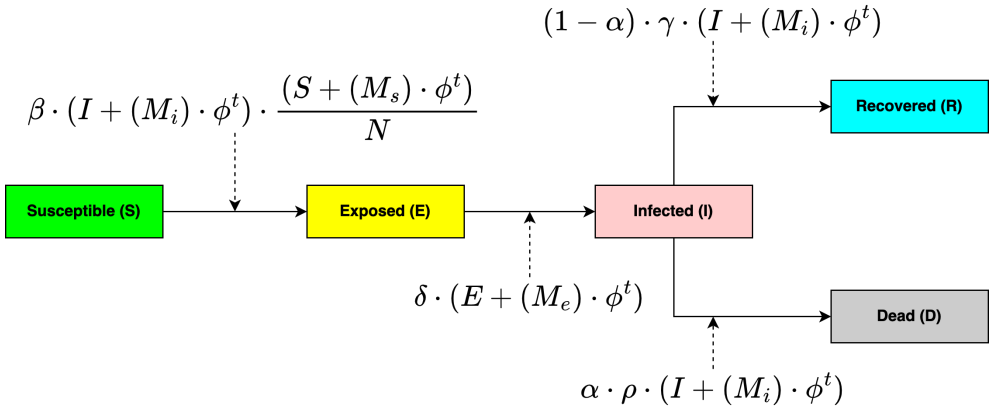

Figure 4.2: Overview of the SEIRD+m model.

The next equation provides the total number of Dead people when human mobility over time is considered.

$$D_{j,t+1} = \alpha \cdot \rho \cdot (I + (M_i) \cdot \phi^t) \tag{4.13}$$

The final version of our extended SEIRD+m model, using M_s, M_e, and M_i, is expressed using the Eqs. from 4.9 to 4.13 and an overview of this is showed in Fig. 4.2.

4.2.2 Data Acquisition

We retrieved data from four sources: 1) El Paso Strong website[2] (actual COVID-19 data), 2) SafeGraph datasets[3] (mobility data), 3) U.S. Census Bureau TIGER/Line Shapefiles (USCB TIGER)[4] to extract El Paso's CGBs and 4) HUD Aggregated USPS Administrative Data On Address (HUD-USPS)[5] to map the data from the ZIP to CBG level.

We created an origin-destination (OD) baseline matrix for the El Paso region using SafeGraph data from 2019 (before COVID). We consider that the year 2019 had normal

[2] https://www.epstrong.org/results.php

[3] https://www.safegraph.com/

[4] https://www.census.gov/geographies/mapping-files/time-series/geo/tiger-line-file.html

[5] https://www.huduser.gov/portal/datasets/usps.html

mobility patterns. For each month of 2019, we computed the mobility averages for each day of the week (e.g., the average of all Mondays of each month) for every CBG. The dimensions of our OD baseline matrix are 12x7x508 (i.e., 12 months, 7 days, and 508 CBGs). All our predictions later used this OD baseline matrix for the baseline information required for simulation. We can create new OD matrices using the baseline.

4.2.3 Fitting Data to Identify Model Parameters

We can use our SEIRD+m model in two ways: to study past events or to make predictions. In this paper, we used it to study a past event – COVID-19 waves in the El Paso, Texas region. In this case, it is necessary to perform a data fitting process. We performed a curve fitting process using the first and second COVID-19 waves to find the parameters i_0, β, γ, and δ of our SEIRD+m model. The i_0 parameter is the initial number of infected people reported by the El Paso Strong website, and its value is optimized during the fitting process. We used values from existing literature for the parameters α and ρ [109]. The least-squares minimization method [110] was used to quantify the performance of the fitting process. The results are presented in Table 4.1.

Table 4.1: Parameters utilized in the SEIRD+m model during the simulations.

a) Obtained during the fitting process		
Parameter	**1st wave**	**2nd wave**
i_0	21.658	40.930
γ	0.424	0.378
δ	0.019	1.715
β	2.263	0.528
b) Obtained from the literature		
Parameter	**1st wave**	**2nd wave**
α	0.030	0.030
ρ	0.054	0.054

Fig. 4.3a shows, in dark black and red lines, the actual COVID-19 infections of the second wave and the SEIRD+m model's simulated infections using the final values of the parameters found during the fitting process. The SEIRD+m model's final result (in the red

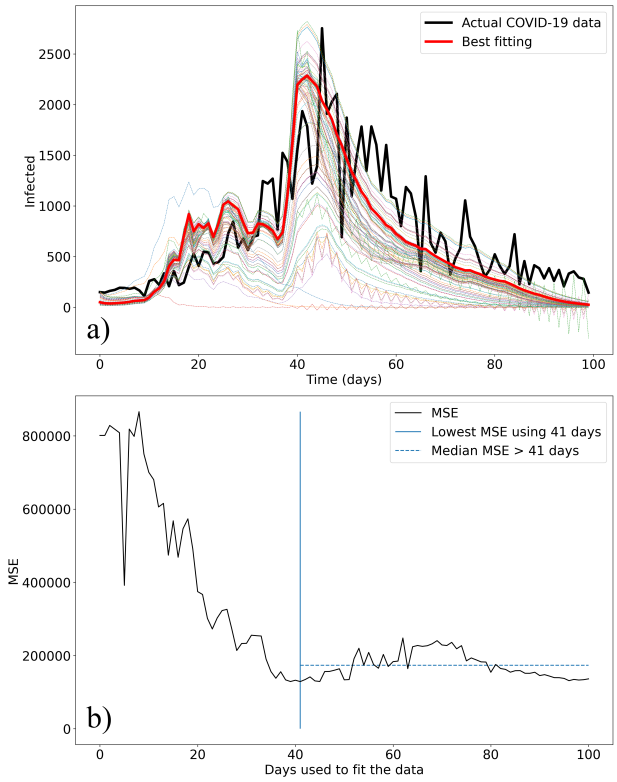

Figure 4.3: Fitted curves using the first N days of the second wave of actual COVID-19 data. The black line indicates the actual data and the red one indicates the fitted line using parameters derived from the first 41 days.

line) follows the trend of the actual data (the black line) with the lowest MSE. The figure also contains many light lines using intermediate values of the parameters while they were being optimized during the fitting process. SEIRD+m started with a trend line almost close to the plot's X-axis and kept improving over the iterations of the error minimization process. In each iteration, the process includes an additional day of the actual COVID-19 data. Our observation is that the lowest MSE is found when the first 41 days worth of data were used, as demonstrated in Fig. 4.3b with a vertical line at 41 days. The red line in Fig. 4.3a uses the first 41 days to determine the parameters. That is, the SEIRD+m model can use initial data of a disease outbreak to simulate the later part; in the case of Fig. 4.3, the initial 41 days' worth of data can be used to simulate the last 59 days of infections.

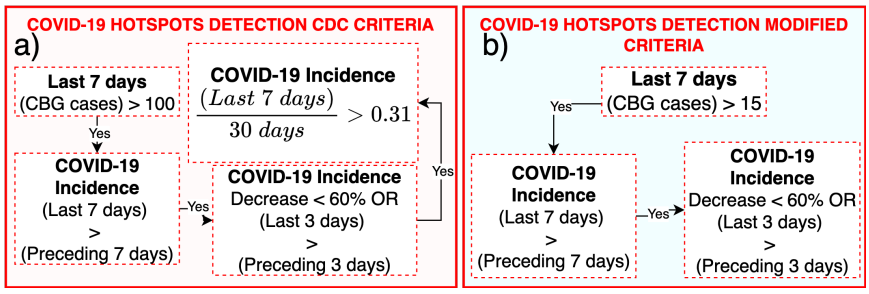

Figure 4.4: Hotspot detection (a) CDC criteria and (b) proposed criteria.

4.2.4 COVID-19 Hotspots Detection

We studied the effect of mobility restrictions on the spread of COVID-19. **Contrary to works that studied mobility restriction over an entire region, we focused on mobility restriction in COVID-19 hotspots.** In the context of infectious diseases, a **hotspot** refers to an area with a high disease load or high transmission efficiency [111]. CDC collaboratively developed a standardized criteria [112] (Fig. 4.4a).

We used the CDC's criteria (Fig. 4.4a), and our new modified version of the CDC's criteria (Fig. 4.4b) to detect hotspots and restrict mobility only on hotspots. Our modified criteria (for hotspot detection) differ from the standard CDC criteria in the following ways: (1) County-level resolution (CDC) vs. CBG-level resolution, (2) The number of new cases (100 for CDC at the county level, 15 for ours at CBG level), and (3) CDC uses a 7-day/30-day incidence count, limiting the algorithm to wait for 30 days before checking for hotspots; we reduced the time to 15 days, similar to the incubation period of COVID-19 (i.e., 14-15 days) [113].

4.3 Results and Discussion

This paper proposes a modified SEIR model version named SEIRD+m with human mobility incorporated into all compartments. We used our model to study the evolution of the COVID-19 disease in El Paso, Texas. This section presents our study on the effect of

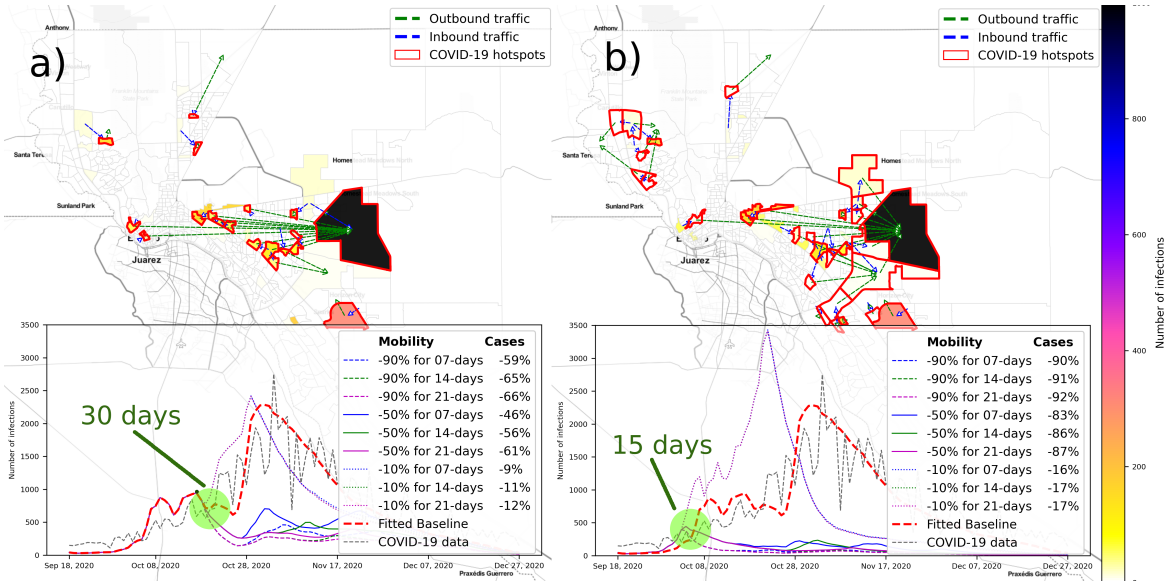

Figure 4.5: Reduction of new COVID-19 cases during the 2nd wave using a) the CDC criteria and b) our modified criteria for detecting hotspots.

human mobility on the spread of COVID-19 (4.3.1). Further, we discuss the interaction between mobility restrictions and demographic factors in the spread of COVID-19 (4.3.2) and provide a comparative analysis (4.3.3).

4.3.1 The Effect of Restricting Mobility on COVID-19 Hotspots.

This section examines the impact on new infections and death incidence when mobility is restricted in COVID-19 hotspots. We focused our experiments on the second wave since it was the most significant wave that affected the El Paso region, causing more than 74,000 infections and more than 1,516 deaths. We used the SEIRD+m model, the CDC's criteria (Fig. 4.4a), and our proposed criteria (Fig. 4.4b) to detect COVID-19 hotspots. We simulate three scenarios where mobility is reduced by 90%, 50%, and 10% over 7 days, 14 days, and 21 days, respectively, for the hotspots detected.

In Fig. 4.5, we used two different visual elements to show our results 1) hotspots maps (top) and 2) time series charts (bottom). The hotspot maps show the CBGs (yellow filled with a red borderline) identified as COVID-19 hotspots. Additionally, the number of

predicted infections in each CBG is filled with a color according to the color bar located on the right side. The green arrows indicate the main outbound traffic between the hotspots and other CBGs. The blue arrows indicate the most significant inbound traffic from other CBGs to the hotspots.

The time series charts at the bottom of Fig. 4.5 present the predicted COVID-19 cases according to the mobility restriction strength and duration, e.g., 90% mobility restriction for 7 days is represented by a blue dashed line. In addition, we plotted a red dashed line, which represents the *fitted data* to the actual COVID-19 data, and a gray dashed line, which represents the actual COVID-19 data.

Using the SEIRD+m model and the CDC's criteria, 19 COVID-19 hotspots were identified (Fig. 4.5a - top). Under this configuration, we observed significant reductions in COVID-19 infections. For example, by restricting mobility by 50% for 7, 14, and 21 days, the number of COVID-19 infections decreased by 46%, 56%, and 61%, respectively (Fig. 4.5a - bottom). The reduction of COVID-19 infections improved (Fig. 4.5b) when we applied our proposed criteria (Fig. 4.4b) for detecting hotspots. For example, when mobility is restricted by 50% during periods of 7, 14, and 21 days, the number of COVID-19 infections decreased by *(Our)***83%** vs 46%*(CDC)*, *(Our)***86%** vs 56%*(CDC)*, and *(Our)***87%** vs 61%*(CDC)*, respectively (Fig. 4.5b - bottom). This improvement is due to the reduction in the number of days, i.e, from 30 to 15 days, that our model is required to wait to detect hotspots (green circles in Fig. 4.5a and 4.5b).

The results of our study demonstrate that by systematically reducing mobility in highly contagious regions, i.e., hotspots, rather than an entire region, it is possible to significantly reduce the number of infections and deaths.

In Table 4.2, we present the predicted infections and deaths using the CDC's criteria and our modified criteria to detect hotspots. The columns *Reduction %* present the reductions in infections and deaths when the mobility restrictions of the column *Mobility Restriction* are applied. The columns *Reduction %* were computed using the actual COVID-19 cases (74,202) and the number of deaths (9,050) predicted by our model using the parameters

Table 4.2: Simulations results of the predicted cases and deaths based on the CDC's criteria and our modified criteria to detect hotspots. The columns *Reduction %* show the reductions of new cases and deaths applying the mobility restrictions of the column *Mobility Restriction*.

COVID-19 Wave & criteria	**Mobility Restrictions (%)**	**Duration Days**	**Number of new cases**	**Reduction %**	**Number of deaths**	**Reduction %**
	90	8	29,832	59	3,969	56
2nd wave	90	15	25,293	65	3,599	60.2
CDC hotspots	90	22	24,960	66	3,572	60.5
original criteria	50	8	39,372	46	4,990	44
	50	15	32,481	56	4,357	51
New cases: 74,202	50	22	28,798	61	4,086	54
Deaths: 9,050	10	8	66,976	9	9,016	0.37
	10	15	65,537	11	8,905	1.6
	10	22	65,002	12	8,878	1.9
	90	8	6,737	90	873	89
2nd wave	90	15	6,063	91	799	90.7
CDC hotspots	90	22	5,809	92	777	90.9
modified criteria	50	8	12,343	83	1,807	79
	50	15	10,353	86	1,575	81
New cases: 74,202	50	22	9,061	87	1,427	83
Deaths: 8,615	10	8	61,634	16	10,408	-20
	10	15	61,161	17	10,373	-20
	10	22	61,010	17	10,366	-20

found during the fitting process. It is important to note that since we used the number of cases during the fitting process, we then used the predicted number of deaths. This table shows that our SEIRD+m model could reduce infections better using our proposed COVID-19 hotspot detection criteria, compared to CDC's criteria (columns Reduction % in Table 4.2). Following our proposed criteria, strong (90%) or relaxed (50%) mobility restrictions for periods of 7, 14, and 21 days can significantly reduce the number of cases and death counts of COVID-19.

4.3.2 The Effects of Restricting Mobility to COVID-19 Hotspots Based on Race, Income, and Age.

This section aims to examine the effects of new infections and death incidence when mobility is restricted based on race, income, and age in COVID-19 hotspots.

4.3.2.1 Race

There are studies that report high mortality rates associated with COVID-19 and race [114]. In this section, we present our experiments and results considering five underrepresented races: 1) Hispanic or Latino (HL), 2) Black or African-American (BAM), 3) American Indian and Alaska Native (AIAN), 4) Asian (A), and 5) Native Hawaiian and Other Pacific Islander (NHOPI). We computed a regional median for each race using its total population in each CBG. The medians were used as threshold values to identify which races are predominant in the CBGs. For example, Asian people are considered one of the predominant races in a CBG when its total population is above its regional median. We used an extended version of our proposed criteria presented in Fig. 4.4b, but it includes a new condition to check if the variable of interest (e.g., race) is above or below the threshold.

In Table 4.3, we report the number of deaths in COVID-19 hotspots where each race is predominant. The table has 12 columns; the first two are for mobility restrictions and duration. From the third column onwards, the columns must be read in pairs. It is

important to note that the number of deaths reported on the header of columns 3, 5, 7, 9, and 11 (i.e., 106, 57, 72, 61, and 103) includes all races without applying mobility restrictions. The dominant race is reported at the top of the header. For example, the third column for Hispanics or Latinos reported 106 deaths in COVID-19 hotspots where the Hispanic or Latino race was one of the predominant. The assumption is that if a race is predominant in a hotspot, as the number of deaths decreases, the number of deaths associated with that race will also decrease. An additional note is that some results in the table are negatives (e.g., row 8, column 4). A negative result indicates that the predicted deaths were higher (overestimated by our model) than the actual COVID-19 deaths. The

Table 4.3: Reductions of new COVID-19 deaths in the CBGs above the race threshold.

Mobility Reduction (%)	Duration Days	Hispanic or Latino 106 deaths	Reduction %	Black or African-American 57 deaths	Reduction %	American Indian and Alaska Native 72 deaths	Reduction %	Asian 61 deaths	Reduction %	Native Hawaiian and Other Pacific Islander 103 deaths	Reduction %
90	7	4	96	4	93	6	92	4	93	6	94
	14	3	97	4	93	4	94	3	95	5	95
	21	2	98	3	95	4	94	3	95	4	96
50	7	7	93	13	77	16	78	11	82	15	85
	14	6	94	10	82	12	83	8	87	11	89
	21	5	95	9	84	10	86	6	90	8	92
10	7	114	-8	53	7	70	3	51	16	92	11
	14	112	-6	53	7	69	4	50	18	91	12
	21	112	-6	53	7	69	4	50	18	91	12

results shown in Table 4.3 demonstrate that a mobility restriction of 90% in COVID-19 hotspots for any period reduces mortality by more than 92% in all races but can be considered a strong measure. On the other hand, for scenarios where mobility was restricted by 50% in COVID-19 hotspots for any period, it was highly effective in reducing mortality by more than 93% for the Hispanic or Latino race. However, for the other four races, the same mobility restrictions of 50% for any period were very effective in reducing mortality but slightly less effective (e.g., between 77–92%) compared with Latino mortality. This difference can be explained by the fact that the Hispanic population is predominant in the El Paso, Texas region.

We, therefore, conclude that restricting mobility by 50% (a less restrictive measure) or 90% (a severe measure) is an effective non-pharmacological measure to protect vulnerable populations in COVID-19 hotspots based on race.

Table 4.4: Simulation results when mobility restrictions were applied to CBGs above and below the poverty threshold.

a) Above Threshold					
Mobility Restriction		**Deaths (163)**		**Cases (65942)**	
Strength	**Days**	**#**	**Reduction %**	**#**	**Reduction %**
90%	7	8	95	4032	94
	14	7	96	3220	95
	21	6	96	2795	96
50%	7	23	86	9262	86
	14	18	89	7469	89
	21	14	91	5962	91
10%	7	166	-2	63899	3
	14	164	-1	63227	4
	21	163	0	63027	4
b) Below Threshold					
Mobility Restriction		**Deaths (3)**		**Cases (1721)**	
Strength	**Days**	**#**	**Reduction %**	**#**	**Reduction %**
90%	7	1	67	270	84
	14	0	100	227	87
	21	0	100	225	87
50%	7	1	67	351	80
	14	1	67	303	82
	21	1	67	286	83
10%	7	2	33	1102	36
	14	2	33	1050	39
	21	2	33	1031	40

4.3.2.2 Income

Studies have found that income inequality is strongly connected with a higher incidence of COVID-19 cases and deaths [115]. In this section, we used the poverty threshold ($26,302 USD) [116] to classify low and high-income COVID-19 hotspots. We demonstrate the regional results (R) in Tables 4.4-A and 4.4-B and local results (L) (within hotspots) in Tables 4.4-C and 4.4-D. According to our results, when mobility is restricted by 50% for 7, 14, or 21 days in high-income COVID-19 hotspots (Table 4.4-A Cases), the number of infections dropped between 80.7–85.1% in the entire region. A stronger mobility restriction of 90% for 7, 14, or 21 days improved the reduction of the number of new cases between 87.7–89.4%. Compared with a mobility restriction of 50%, there is an average difference of 6% in the reduction of new cases. However, a restriction of 50% is more flexible and can reduce inconvenience among the population.

The COVID-19 scenario is entirely different when mobility is reduced only in low-income COVID-19 hotspots (Table 4.4-B Cases). According to our results, the regional number

of COVID-19 cases was slightly reduced by 5–6%, and the number of deaths was reduced by less than 1%. This can be explained by the limited number of hotspots detected (only 5). According to these results, the first impression is that reducing mobility in low-income hotspots is not an adequate measure to stop the spread of COVID-19 in an entire region.

However, we found a drastic drop in the number of deaths when we analyzed the simulation's results only in the five low-income COVID-19 hotspots – i.e., local results. (Table 4.4-D Cases). For example, if mobility is not restricted, the number of new cases reaches 1,721. However, by restricting mobility by 50% for any period (e.g., 7, 14, or 21 days) on these five low-income COVID-19 hotspots, the predicted number of cases was reduced between 79.6–83.4% (Table 4.4-D Cases). Thus, new cases are reduced locally.

Using the same concept, we only analyzed the evolution of COVID-19 in the 39 high-income COVID-19 hotspots (Table 4.4-C). If mobility is not restricted in these hotspots, our SEIRD+m model predicts 65,942 infections. This indicates that 51.8% of the population in these hotspots was infected (the total number of residents in these hotspots is 127,263). However, for example, when mobility is reduced by 50%, the number of new infections drops by 93.9% in those hotspots (Table 4.4-C Cases).

We conclude that restricting human mobility is an effective non-pharmaceutical measure to reduce the number of infections and deaths in low or high-income COVID-19 hotspots. Our results suggest that restricting mobility has positive effects at regional and local levels (Tables 4.4-A vs. 4.4-C) or sometimes can be used to protect vulnerable regions (Tables 4.4-B vs. 4.4-D).

4.3.2.3 Age

Older adults are another vulnerable population highly impacted by COVID-19 [114]. We simulated two scenarios (i.e., identifying COVID-19 hotspots above and below the threshold of 65 years of age) using our criteria presented in Section 4.2.4.

We demonstrate the regional results (R) in Tables 4.5-A and 4.5-B and local results (L) (within hotspots) in Tables 4.5-C and 4.5-D. For example, if mobility is not reduced

Table 4.5: Reduction in the number of deaths in the CBGs above and below the threshold of older adults 65 years and over.

a) Above Threshold					
Mobility Restriction		**Deaths (29)**		**Cases (12182)**	
Strength	**Days**	**#**	**Reduction %**	**#**	**Reduction %**
90%	7	2	93	1057	91
	14	2	93	895	93
	21	2	93	774	94
50%	7	6	79	2423	80
	14	5	83	2145	83
	21	4	86	1594	87
10%	7	27	7	10494	14
	14	26	10	10258	16
	21	26	10	10179	16
b) Below Threshold					
Mobility Restriction		**Deaths (137)**		**Cases (55481)**	
Strength	**Days**	**#**	**Reduction %**	**#**	**Reduction %**
90%	7	7	95	3244	94
	14	5	96	2552	95
	21	5	96	2246	96
50%	7	18	87	7189	87
	14	14	90	5627	90
	21	11	92	4654	92
10%	7	141	-3	54507	2
	14	140	-2	54019	3
	21	140	-2	53879	3

in COVID-19 hotspots where older adults predominate, the **regional** number of infections reaches 71,202. However, by reducing mobility at different strengths by any period, we obtained slight reductions in the number of infections between 7–19.6% (Table 4.5-A). *These results can be explained by the fact that we identified a reduced number of hotspots.* However, analyzing the COVID-19 hotspots (**local results**) and where the population of older adults is predominant, we found interesting results. **We observed a dramatic reduction in the number of deaths related to COVID-19 (Table 4.5-C).** For example, using a relaxed mobility restriction of 50% for any period (e.g., 7, 14, or 22 days) only on hotspots above the threshold (CBGs where the population of older adults predominates), the predicted number of deaths decreased between 79.3–88.2% (Table 4.5-C Deaths). Our results show when mobility is restricted to 50% for 7, 14, or 21 days in COVID-19 hotspots where older adults are not predominant, the number of infections drops between 69.4–72.8% in the entire region (Table 4.5-B). When a stronger mobility restriction of 90% is applied for 7, 14, or 21 days, the number of new cases is reduced

between 74.7–76%. There is a difference of 5% in the number of new cases.

On the other hand, when applying the same mobility restrictions to COVID-19 hotspots where the population of older adults is below the threshold, the predicted number of deaths decreased between 87–92% **locally** (Table 4.5-D Deaths). **Thus, comparing Tables 4.5-A and 4.5-C, we can see that mobility restriction does not necessarily have a positive impact at the regional level but can help to protect specific sub-regions – e.g., regions where older adults predominate.**

4.3.3 Comparing SEIRD+m With Other Approaches

In this section, we compare our SEIRD+m model with relevant variations (e.g., with and without mobility and several other combinations of the compartments).

For disease modeling, it is required a simulation performs its best in identifying the peak of the number of infections. We observe that SEIRD+m has better predictions during the pandemic's peak than the other variations, as shown in Fig. 4.6.

In Fig. 4.6, we examined the effects of human mobility on the following three versions of our SEIRD+m model: 1) not using mobility in the Infected and Recovered compartments (SED+m)(IR), 2) not using mobility in the Exposed, Infected, and Recovered compartments (SD+m)(EIR), and 3) not using mobility in any compartment (SEIRD). We used the fitting process described in Section 4.2.3.

The X-axis of Fig. 4.6a is the number of days, k, worth of how much data is used, and the Y-axis is the median MSE of the error computed for the rest of the days of the second wave of COVID-19. All the MSE values of the models exhibited a similar pattern of high MSE values when lesser than 20 days' worth of data were used. This is expected because the models require sufficient days' worth of data to converge to reasonable parameters. After using 40 days most of the MSE values stabilized.

Earlier, in Fig. 4.3, we demonstrated that our SEIRD+m model needs around the first 41 days of observed COVID-19 data to simulate the later part of the data. The peak of the second wave of COVID-19 was between the 41st and 50th days. The ability to predict

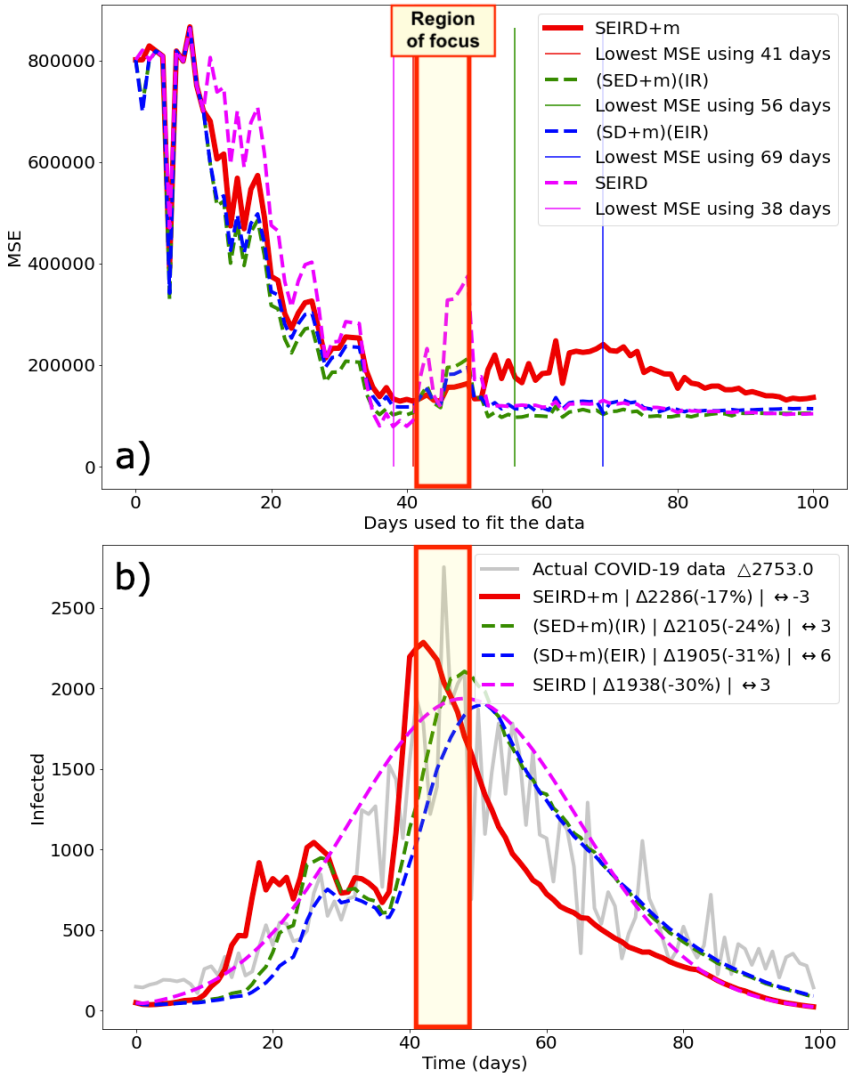

Figure 4.6: (a) Days' worth of data used for the models vs MSE, and (b) Simulation of infections using different methods.

better during this region is critical for better control of the disease. The yellow box in Fig. 4.6a shows that the MSE is lower for SEIRD+m compared to other models during this peak region, indicating that SEIRD+m has the more predictive capability for the COVID-19 peak. The MSE rises when COVID-19 keeps falling after the peak, but the later timeline is less critical.

Correspondingly, Fig. 4.6b shows the actual infection data (in grey) and the simulations

of all the models using 41 days' worth of data. Our model predicts the peak earlier compared to the other models. The actual peak day was on the 45th day of the second wave. Our model predicts the peak on the 42nd/43rd day (safer), whereas the other models predict the peak to be later than the 45th day (likely to cause harm due to reluctant restrictions).

The SEIRD+m model captures human mobility with high fidelity, as can be seen in the peaks and valleys in the time series (red line in Fig. 4.6b). The other lines are smoother, sometimes not practical in disease propagation.

4.4 Conclusions and Future Work

This paper presents an epidemiological model named SEIRD+m. One of our main contributions is the incorporation of human mobility into all its differential equations. We used our model to study how COVID-19 spreads when human mobility is **restricted only in COVID-19 hotspots** at different intensities and time periods. Through our experiments, we demonstrated that it is possible to reduce infections and deaths by systematically reducing mobility in COVID-19 hotspots rather than in the entire region. We also found that restricting mobility in COVID-19 hotspots positively affects the number of infections and deaths at regional and local (within hotspots) levels and may be a useful strategy to protect areas at risk based on income, race, or age. Our future work aims to analyze the impacts of the co-existence of two or more SARS-CoV-2 variants (e.g., delta and omicron) in the same region, along with vaccination. An analysis using national and international regions also remains as future work.

Chapter 5

Detecting Malware Activity Using Public Search Data[1]

The prevalence of malware on the Internet makes malware detection vital as an early warning system for organizations' security. This paper presents a novel approach to linking knowledge from heterogeneous and specialized datasets using a sentence embedding approach. This paper also proposes a novel approach to detect malware activity using standardized and specialized datasets and people's search interest data. We demonstrated the detection capabilities of our approach, assessing our models using four real attack study cases. We found an increase in Google searches and probabilities of our models seven days before and after an attack occurred. In addition, the web search volume and model probabilities time series are characterized by an increase in outliers around 14 days before and after the discovery of the attack. This work should pave the path for integrating domain-specific datasets and user-generated dynamic data for detecting malware activity.

5.1 Introduction

Malware is specially-written files or programs that typically spread through computer networks and carry out malicious activities (e.g., financial transactions). In fact, a recent report stated that the damage caused by cybercrimes puts it in the third position of the world economy [2].

[1] https://doi.org/10.1109/BigData55660.2022.10020883

[2] `https://cybersecurityventures.com/cybercrime-damages-6-trillion-by-2021/`

On May 7, 2021, the Colonial Pipeline system went down for six days due to a cyber-attack [117]. The Colonial Pipeline covers 5,500 miles and transports 2.5 million barrels of fuel daily for the Southeast and the East Coast (45% of all fuel consumed in this region). In response to this attack, the company halted all pipeline operations. The main target of this ransomware attack was the billing infrastructure of the company [118]. The hackers obtained nearly 100 gigabytes of data from Colonial Pipeline's network in just two hours [119].

The attack on Colonial Pipeline is just one high-profile example of a cyber attack's effects on critical infrastructure when operations are disrupted or halted. Our society heavily depends on the energy and communications infrastructure. These are the backbone of national security and economic development. The adversaries are aware of the importance of the energy sector and other industries relying on it; hence these sectors are often the target of adversarial attacks.

The Internet plays an important role in the spread of malware and its early detection can significantly reduce damages (e.g., data loss). Many approaches have been developed for detecting intrusions. However, all these measures rely on the tools or techniques related to intrusion detection. Whether the infected machine has any intrusion detection systems (IDS) or not, with the abundance of information available online, users' first response usually is to look for information on the symptoms they are observing in the infected machines – slower speed, the abrupt closure of programs, etc. In this paper, we show how public search data can play a valuable role in the early detection of malware.

Our proposed approach leverages the Common Vulnerabilities and Exposures (CVE) database and the MITRE ATT&CK database to identify malware features. Google Trends API is then used to search for these features to determine if people are searching for them. Experimental results show that it is possible to detect unusual malware activity using the CVE and MITRE ATT&CK databases and people's search data.

The key contributions of this paper are:

1. A novel approach to linking knowledge from heterogeneous and specialized datasets

using a sentence embedding approach.

2. A novel approach to detect malware activity over a regional area using standardized and specialized datasets and people's search interest data.

3. We provide case studies for the Conti and Ryuk malware using data from real attacks.

5.2 Approach

We seek to answer **how we can detect malware activities with lower to no dependency on any intrusion detection systems**. In this work, we assume that people will search for symptoms and remedies when exposed to malware infections. However, using raw search data for malware activity detection is unsuitable because people often use general search terms that are not standardized. For example, users who are infected with the same malware may use different terms to search for symptoms associated with it. So, it is difficult to connect those search terms with the same malware.

We address this problem by creating a link between malware and entries from two databases: CVE and ATT&CK. This linking generates a dataset of malware-specific terms, which are later used to extract Google trends data. We call it the **A**ttack-specific **V**ulnerability **E**xposu**R**e **T**erms (AVERT) dataset. We generate an AVERT dataset for each malware. This dataset is used to train various classifiers. We carried out a 5-fold cross-validation for each classifier. A detailed description of the construction process of AVERT is given in the following subsections.

5.2.1 Dataset Used

5.2.1.1 CVE Database

The MITRE corporation supervises the Common Vulnerabilities and Exposures (CVE) database[3]. It is funded by the Cybersecurity and Infrastructure Security Agency (CISA).

[3] `https://www.cve.org/`

The CVE database contains a list of approximately 236,000 publicly known security flaws without technical information (e.g., about risks, impacts, or fixes). A valuable benefit of the CVE database is that it helps identify unique vulnerabilities needed to develop tools and solutions that improve security. To assign CVE identifiers, a CVE Numbering Authority (CNA) is necessary. A CNA is a major IT vendor, such as IBM, Cisco, Oracle, Microsoft, etc. New CVE candidates are assigned a CVE ID, described briefly, referenced, and posted to the CVE website by a CNA. The CNA should, however, ensure that all CVEs can be fixed independently of any other bug, they negatively impact security and violate the security policy of the affected system. Additionally, each CVE is assigned a score based on the Common Vulnerability Scoring System (CVSS) [120]. Higher scores indicate a greater severity, ranging from 0.0 to 10.0.

5.2.1.2 MITRE ATT&CK Database

The MITRE ATT&CK (Adversarial Tactics, Techniques, and Common Knowledge) project[4] began in 2013 as part of MITRE's Fort Meade Experiment (FMX), where researchers emulated adversary and defender behavior to improve post-compromise threat detection. The tool is a curated knowledge-base and behavioral model for cyber adversaries. Essentially, it illustrates an adversary's attack cycle and target platforms. Using the tactics and techniques abstraction in the model, both offensive and defensive sides of cybersecurity can understand individual adversary actions. Additionally, it categorizes adversary actions and offers specific ways of defending against them. ATT&CK contains a matrix listing adversaries' techniques to achieve specific objectives. Each of these objectives is categorized as a tactic. There are 14 adversary tactics, including reconnaissance, initial access, privilege escalation, etc. In each tactic of the ATT&CK matrix, adversary techniques describe the activity the adversary has undertaken. In some techniques, some sub-techniques describe how an adversary performs a particular technique in greater detail.

[4] `https://attack.mitre.org/`

5.2.1.3 CISSM Cyber Events Database

The CISSM (Center for International & Security Studies at Maryland) Cyber Events Database[5] is a database that contains publicly available structured information about cyber events from 2014 to 2021. CISSM Cyber Events Database data is derived from open news sites, blogs, and other specialty sites. The data is retrieved using automated techniques paired with manual review and classification by researchers. Updated monthly, the data provides information about the threat actor, motive, victim, industry, and attack effects.

5.2.1.4 CIRA Database

The Critical Infrastructure Ransomware Attacks (CIRA)[6] database was created in September 2019. Ransomware attacks are collected from publicly published media reports or security reports. There are 1,293 records in the database gathered from publicly disclosed incidents between November 2013 and August 31, 2022. The database contains details about the attacks. For example, the date and name of the organization that identified the ransomware attack, the type of ransomware used, and how long it took the organization to pay the ransom and recover.

5.2.1.5 DMA Database

The Designated Market Area (DMA)[7], also known as a media market, defines television and radio markets in the United States. The DMAs cover the entire country and are typically defined as metropolitan areas, with suburbs often incorporated. Approximately 210 DMAs are covering the whole country. The Google Trends API uses a DMA code to identify a city (instead of a city name); thus, it is necessary to map a city's name to its DMA code. We retrieve the city-specific DMA code from the DMA database.

[5] `https://cissm.umd.edu/cissm-cyber-events-database`

[6] `https://sites.temple.edu/care/cira/`

[7] `http://bl.ocks.org/simzou/6459889`

5.2.1.6 Google Trends

The Google Trends[8] (GT) service provides access to a large, unfiltered, and anonymous sample of search requests. The requests are categorized and grouped to determine the topic for a search query. This methodology allows displaying interest in a particular topic from around the world or at the local level. Google Trends normalizes search data using queries' time and location using the following two steps: 1) The total number of searches for a geography and time range is divided by the number of data points, and 2) the results are then scaled from 0 to 100 according to the proportion of searches within each topic.

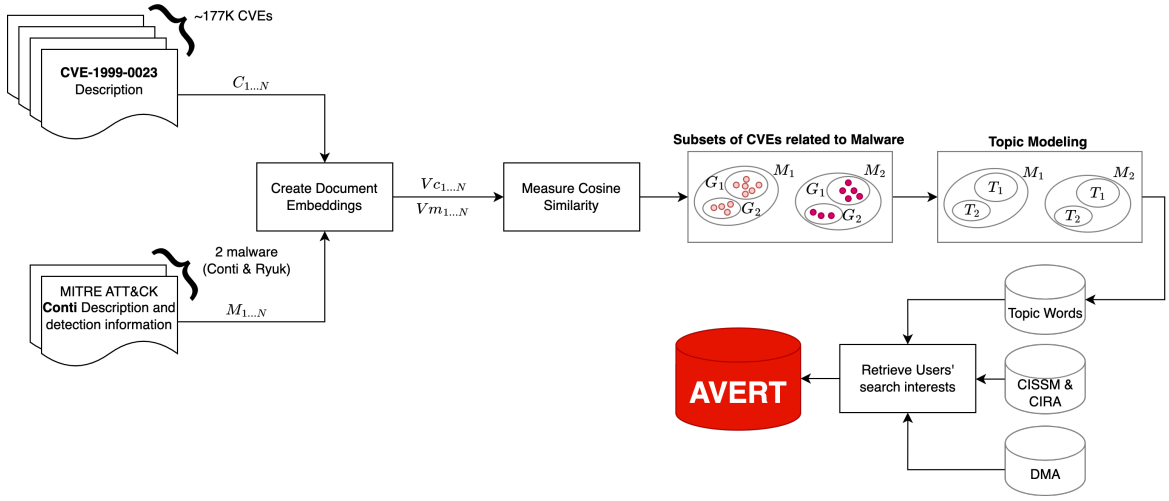

Figure 5.1: **A**ttack-specific **V**ulnerability **E**xposu**R**e **T**erms (AVERT) datasets construction process. The system integrates information from the CVE, ATT&CK, CISSM, and CIRA databases using a sentence embedding approach.

5.2.2 Malware Selection

Ransomware is malware that encrypts files and demands a ransom to decrypt the files and restore the system to its original configuration. In this work, we have selected the Ryuk and

[8] `https://trends.google.com/`

Conti ransomware because of their usage in large-scale attacks against U.S. organizations[9].

5.2.2.1 Ryuk

Ryuk is a family of ransomware that first appeared in 2018. It is considered one of the most dangerous ransomware. Numerous government, academic, healthcare, manufacturing, and technology organizations have suffered attacks by this ransomware version, attributed to the hacker group WIZARD SPIDER. By the end of 2020, Ryuk generated $150 million in profits, and the highest ransom demand was USD 12.5 million in 2019. Usually, Ryuk is installed on a system after being infected by Trojans (e.g., Trickbot) [121].

The first step Ryuk takes when infecting a system is to shut down 180 services and 40 processes that could prevent its attack[10]. Afterward, Ryuk encrypts photos, videos, databases, and documents using AES-256 encryption. Following the symmetric encryption keys, RSA-4096 is used to encrypt them asymmetrically. Ryuk can encrypt remote shares, including administrative shares, remotely. Additionally, it is capable of waking computers for encryption using Wake-On-Lan. Due to these abilities, its encryption is effective, reachable, and can cause considerable harm.

5.2.2.2 Conti

Conti is assumed to be the successor of the Ryuk ransomware. This ransomware family has carried out several high-profile attacks. Conti operators use double extortion techniques, publishing stolen data and selling access to victims who refuse to pay the ransom [122]. Conti typically spreads through phishing emails that contain a link to Google Drive that downloads the ransomware via BazarLoader. As soon as the ransomware is installed, attackers use batch files to disable security tools, obtain domain administrator credentials, and dump the domain controller credentials. The data is then synced to cloud storage. All backups in the victim's network are deleted and data is encrypted using the AES-256

[9] `https://www.cisa.gov/uscert/ncas/alerts/aa21-265a`

[10] `https://www.trendmicro.com/en_us/what-is/ransomware/ryuk-ransomware.html`

algorithm.

5.2.3 Construction of the AVERT Dataset

Malware activity detection using users' search terms requires a set of standardized terms. We constructed the AVERT dataset to satisfy this requirement, integrating data from the CVE, ATT&CK, CISSM, and CIRA databases. In the following subsections, we describe different aspects of the AVERT construction process as shown in Figure 5.1.

5.2.3.1 Finding CVEs Related to a Malware

The CVE database contains approximately 236,000 cybersecurity vulnerabilities identified, defined, and cataloged by organizations worldwide. We selected 177,000 officially accepted CVE entries labeled as *Entry* for further analyses. We use the document embedding approach (mapping documents into numerical vector spaces) to construct the AVERT dataset. We utilized a Siamese neural network architecture [123], with two identical BERT architectures (e.g., $BERT_A$, $BERT_B$) that share the same parameters and weights (Fig. 5.2). Note that a pooling operation is added to BERT's output. The Siamese architecture facilitates the extraction of semantically meaningful document embeddings, which can then be compared using the cosine-similarity method.

Let us say that C_1 is the input of $BERT_A$, and M_1 is the input of $BERT_B$; where C_1 is the description of a CVE entry extracted from the CVE database, and M_1 is the description of a malware extracted from the ATT&CK database. After $BERT_A$ and $BERT_B$ have processed C_1 and M_1, the network uses the mean pooling technique to reduce their dimension and produce the vectors V_{c1} and V_{m1} (document embeddings) of 384 dimensions each. Note that the all-MiniLM-L12-v210 sentence transformer model we used to compute the sentence embeddings maps the sentences to a 384-dimensional vector space. We also used the mean pooling technique because it performs better in document embedding tasks [123]. We then compute the semantic similarity between V_{c1} and V_{m1}. A high similarity indicates

that V_{c1} is related to V_{m1}. Eventually, we have subsets $G_{1...N}$ of CVE entries $C_{1...K}$ related to every malware $M_{1...N}$.

It is important to note that compared to using only BERT, the Siamese neural network approach performs better in computing document semantic similarities. For example, to find the pair with the highest similarity in a collection of 10,000 documents, BERT requires approximately 49 million inference computations; this is about 65 hours on a modern V100 GPU. On the other hand, this same task requires only about 5 seconds using Siamese networks [123].

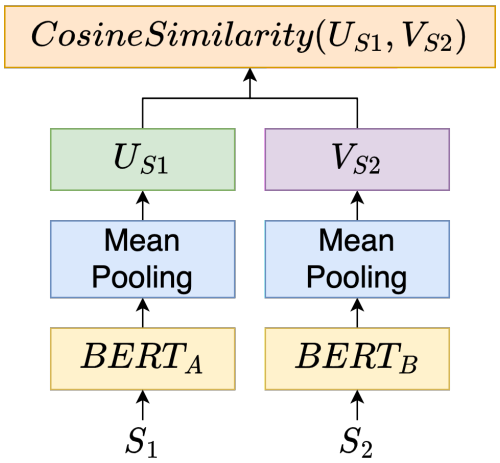

Figure 5.2: Siamese Network used to compute sentence similarity.

5.2.3.2 Topic Modeling on Malware-specific CVEs

Now that we know the clusters $G_{1...N}$ of embedded CVE entries (vectors $V_{c1...N}$) associated with each malware $M_{1...N}$, the next step is to discover the most relevant words in each cluster. The topic modeling technique was used to accomplish this task. Using this approach we created $T_{1...N}$ clusters within G so each cluster $T_{1...N}$ has a similar topic. However, the performance of some clustering algorithms is poor when dealing with high-dimensionality embeddings [124]. We employed the Uniform Manifold Approximation and Projection (UMAP) algorithm to reduce dimensionality from 384 to 5 (default value) [125].

The UMAP algorithm preserves more global details with superior run-time performance than t-SNE (a popular dimensionality reduction method). As a general-purpose dimension reduction technique for machine learning, UMAP has no computational restrictions on embedding dimensions.

Once we reduced the embeddings' dimension of the CVEs embeddings in each $G_{1...N}$, we proceeded to clusterized each $G_{1...N}$ group with at least 15 elements using the HDBSCAN algorithm. A density-based algorithm such as HDBSCAN is helpful in conjunction with UMAP since UMAP maintains many details even in a lower-dimensional space. In addition, HDBSCAN does not group outliers into clusters [126].

We modeled the topics using the documents in each cluster, where each cluster is assigned a topic. Also, in our case, each document is the concatenation of the CVEs descriptions. We adopted the TF-IDF formula (Eq. 5.1) proposed in [127] to calculate a term's importance to a topic. In this equation, the term $tf_{t,c}$ models the frequency of term t in a cluster C, A is the average number of words per cluster C, tf_t is the frequency of term t across all clusters $C_{1...N}$. The number one is added to the division within the logarithm to output only positive values. Lastly, we create a topic representation database using the top 10 non-repeated terms per topic based on their TF-IDF scores (Eq. 5.1).

$$TF - IDF = tf_{t,c} \cdot log(1 + \frac{A}{tf_t}) \tag{5.1}$$

5.2.3.3 Retrieving Users' Search Data

The last part of the AVERT construction process was to retrieve users' search data related to malware attacks. We extracted the date, city, and duration of cyber attacks involving Conti and Ryuk malware from the University of Maryland CISSM Cyber Events and CIRA databases. In addition, we extracted cities' DMAs from the DMA dataset for use in the Google Trends API.

The extracted data was parsed and prepared to retrieve the users' search data using the Google Trends API. This tool allowed us to obtain the users' interest for a term quantified

by a score between 0 and 100. A term with a score close to 100 has a high search volume. We created a version of our AVERT dataset for each malware (Conti and Ryuk). The data include all days of a given year, topic words, topic search scores, and labels. The label 1 means that an attack occurred or was within the attack duration range, and 0 means the attack was not reported in the CISSM and CIRA databases.

5.2.4 Preprocessing the AVERT Dataset

The AVERT datasets must be preprocessed to address their imbalance, overlapping classes, and skewness. The imbalance and overlap were addressed by undersampling and oversampling. Combining these two methods is highly effective in addressing dataset imbalance [128]. We tested two combinations, the Synthetic Minority Over-sampling TEchnique (SMOTE) [83] + the Edited Nearest Neighbors (ENN) and SMOTE + TOMEK-Links [84]. The combination of SMOTE and ENN produced better results in reducing the class imbalance. Our AVERT datasets were also transformed with a log transformation to reduce their skewness. The log-transformation method is commonly used to reduce skewness and transform the data into a Gaussian distribution [86]. A representation of our AVERT dataset is given in Figure 5.3.

5.2.5 Feature Importance

We computed the features' importance using the SHapley Additive exPlanations (SHAP) approach. The SHAP approach uses game theory to explain the output of any machine learning model [129]. A Shapley value is the average marginal contribution of a feature value across all possible coalitions. In the case of N features, Shapley values will result from N! different order combinations. In our case, the values represent those features' importance in determining whether the activity in a row is related to malware. Positive values will generally lead to a prediction of 1 (i.e., malware activity detected). In contrast, negative values will lead to an opposite prediction (i.e., no malware activity detected). In

0 = No malware activity
1 = Malware activity

instances (R) 365 days X specific year(s)	TERMS ($T_{1,...,K}$)							LABEL
	.75	.04	.35	⋯	.81	.92	0	0
	.45	.67	.74	⋯	.92	.12	.49	1
	.64	.87	.24	⋯	.70	.81	.43	1
	⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯
	.86	.04	.46	⋯	.92	.65	.19	0
	.65	.60	.94	⋯	.89	.92	.27	1
	⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯

Figure 5.3: AVERT dataset representation. The columns are the terms (topic words), rows are the days for a given year, each cell is the Google Trends score, and the Label column indicates whether an attack was reported in the CISSM and CIRA databases.

our AVERT datasets, SHAP calculates a base value for every row. Each feature's value contributes positively or negatively to the base value in a specific row. A simple way to see our distance from this base value is to sum the contributions across all features.

5.2.6 Building the Models

We conducted a 5-fold cross-validation during the modeling process. The cross-validation (CV) technique tests the effectiveness of machine learning models. We split our datasets with a ratio of 80% training and 20% testing. We followed the scaling law for the validation-set training-set size ratio proposed by Guyon [86]. We built and tested the models using several machine learning (ML) algorithms. We aim to develop a framework to detect malware activity over a geographical region. This can be viewed as a machine-learning problem for binary classification. Several ML algorithms exist to address such problems. We selected ten algorithms for further study based on their accuracy and interpretability.

The difference between accuracy and interpretability is that accuracy measures the correctness of classification. In contrast, interpretability explains why a classification was made [130]. Our work is intended to be used by experts and non-experts in computer science. Therefore, the framework must produce results that are both accurate and explainable. We selected the following ten algorithms:

- Low Accuracy-High Interpretability: Logistic Regression (LR)
- Medium Accuracy-Medium Interpretability: Gaussian Naive Bayes (GNB), Decision Trees (DT), K-Nearest Neighbor (KNN), Stochastic Gradient Descent (SGD), Linear Discriminant Analysis (LDA)
- High Accuracy-Medium Interpretability: XGBoost [27], Support Vector Machine (SVM)
- High Accuracy-High Interpretability: Random Forest (RF)
- High Accuracy-Low Interpretability: Multi-Layer Perceptron-Neural Network (MLP-NN)

5.2.7 Evaluation Metrics

As part of our interest in minimizing the number of classification errors of the positive class (malware activity), we only used the recall and F1 measures for the positive class.

Recall calculates how many actual positives a model captures by labeling it as positive (true positive). We used the recall measure due to the high cost of false negatives in our malware activity detection problem. For example, suppose our model predicts a malware activity (actual positive) as a non-malware activity (predicted negative). In that case, being unable to detect actual malware activity can have severe consequences. The **F1 score** combines precision and recall into a single score. As we mentioned before, we are interested in reducing the number of false negatives. However, we also used the F1 score

to decide which machine learning algorithm has a better trade-off between precision and recall.

5.3 Results and Discussion

In the following sections, we report the results obtained during the experiments that help answer the question: how can we detect malware activity in a region, with lower to no dependency on intrusion detection systems using public search data?

5.3.1 Feature Importance

We computed the features' importance for each AVERT dataset using the approach described in section 5.2.5. The results of the first top 20 features for the Conti and Ryuk malware are presented in Figure 5.4. The x-axis represents the feature's score in SHAP units, and the y-axis the features.

Each feature has many points (observations or rows in the AVERT datasets) colored according to the bar color located to the right. Each feature's distribution of points and values (color) represents how the feature contributes to the model's prediction. For example, the *impact* feature is in the top 20 on both malware. The high values (red points) of the *impact* feature in the Conti malware contributes more to the model's ability to predict the positive class (malware activity). In the Ryuk malware, higher values of this feature (red points) significantly contribute to model performance. In these cases, the high values of the feature *impact* contribute positively to the model's prediction. On the other hand, high *tracking* feature values in Ryuk malware negatively affect the model's ability to detect the positive class (malware activity).

As stated earlier, we use sentence embedding to measure the similarity between malware and CVEs. The benefit of using sentence embedding is that it retains the context of the sentence from its words. Observing this property in action is possible when we examine the Ryuk dataset and its features' importance. For example, the feature *covid*, appears

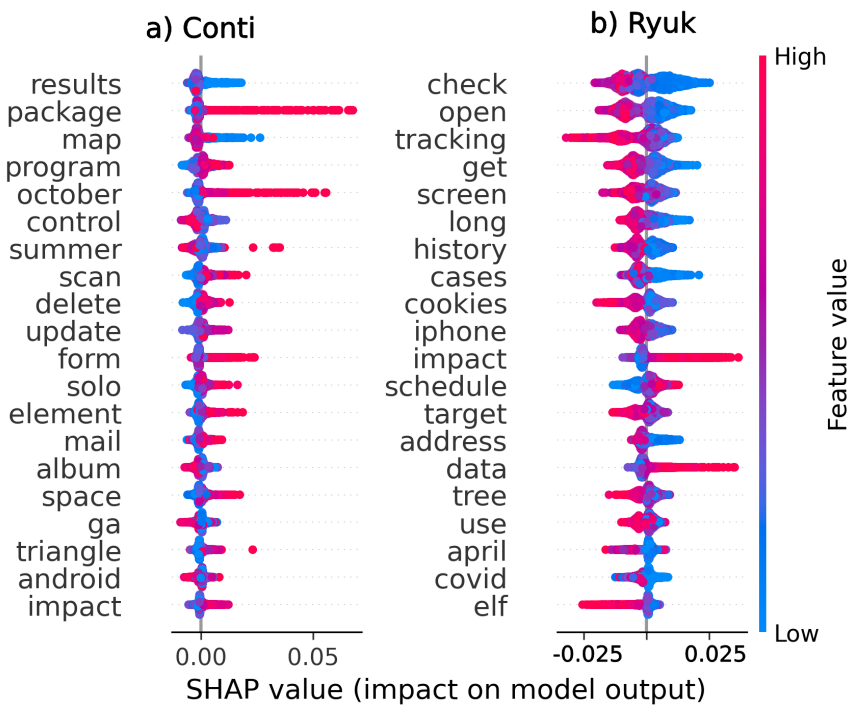

Figure 5.4: The top 20 features for the a) Conti and b) Ryuk malware.

ranked 19th. The model's performance is nearly equally affected by its low and high values. At first glance, the *'covid'* feature seems out of context in the Ryuk dataset. However, during the COVID-19 pandemic, there has been a significant increase in Ryuk ransomware attacks [131]. We use this information to demonstrate the benefits of using sentence embeddings when constructing our AVERT datasets to find semantic similarities between the CVE and ATT&CK databases.

5.3.2 Model Construction

An algorithm performance comparison was conducted using a 5-fold cross-validation and the F1 score and recall metrics. We used two different datasets: 1) our Conti and Ryuk AVERTs datasets and 2) new datasets constructed using terms extracted from Malwarebytes[11]. The principal function of Malwarebytes is to scan and remove malicious software, including

[11] `https://www.malwarebytes.com/`

spyware, adware, and rogue security software. We constructed the two new datasets by extracting all the text related to Conti and Ryuk from Malwarebytes pages. Each paragraph was then considered a document to calculate the TF-IDF scores and obtain the most relevant terms. In the final step, we retrieved the users' search interests based on these terms to construct the new datasets.

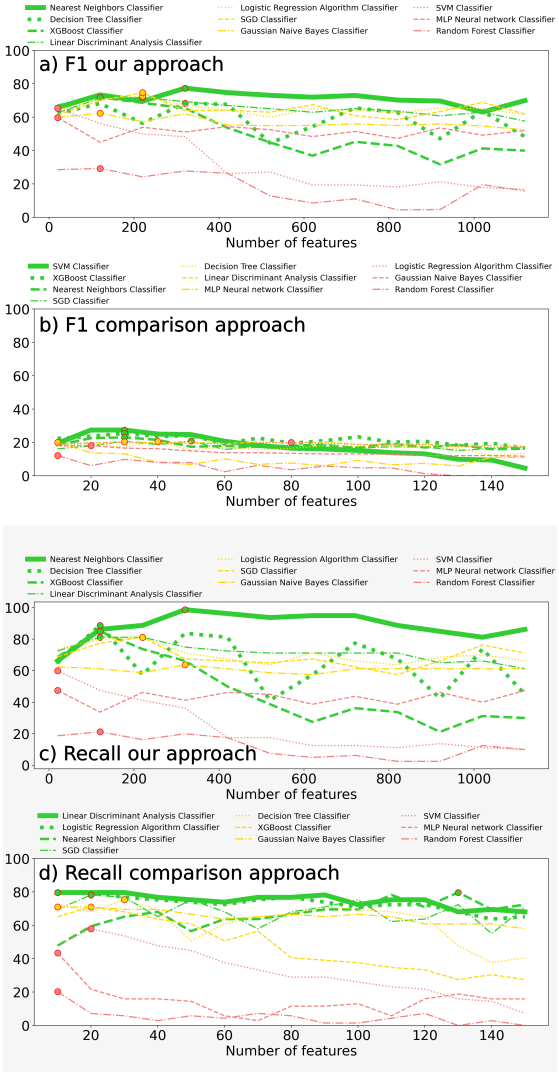

Figure 5.5: Comparison of performance results for detecting Conti malware activity using our approach (AVERT dataset) and a generic approach (Malwarebytes data). Our approach had better F1 score and Recall results.

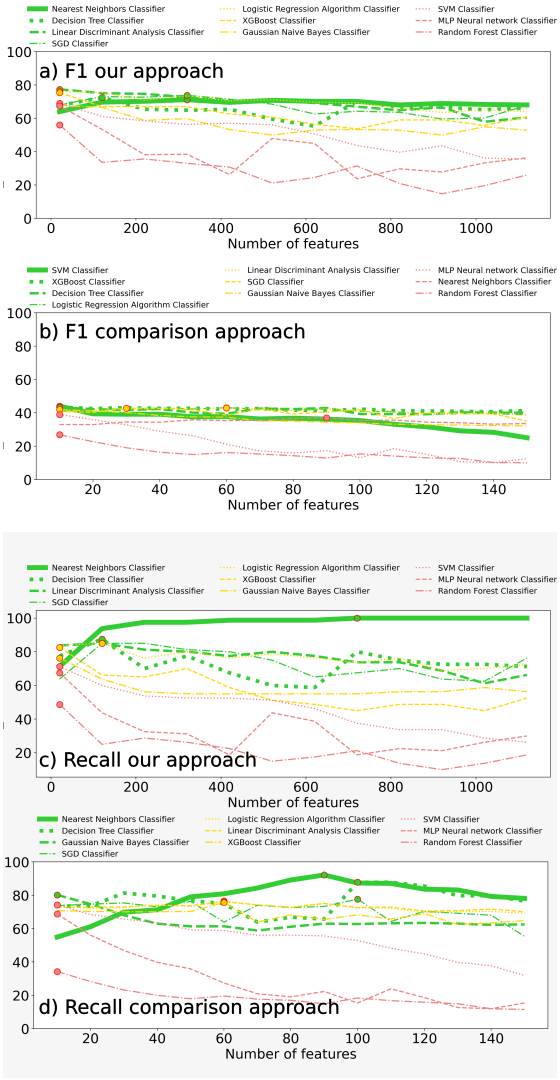

Figure 5.6: Comparison of performance results for detecting Ryuk malware activity. A 5-fold cross-validation was performed using our approach (AVERT dataset) and a generic approach (using Malwarebytes data). Our approach had better F1 score and Recall results.

Figures 5.5 and 5.6 present our comparison using both datasets (i.e., AVERT and Malwarebytes). For each figure, there are four graphs; the first two (a and b) show the F1 scores for the positive class for the AVERT and Malwarebytes datasets, and the last two (c and d) show the recall scores for both datasets. The x-axis represents the number of the top K features used during the training phase, incremented by one in each iteration. The y-axis is the evaluation metric score (F1 score or recall). In each graph, the colored lines represent the performance of a machine-learning algorithm, while the thick green line is the best algorithm. The best performance of each algorithm is shown by a small circle with a red border and the same fill as the line it belongs to.

We can observe that the performance of the algorithms using our AVERTs datasets was much higher than those constructed using Malwarebytes. For example, the Nearest Neighbors algorithm achieved an F1 score of 80% for the Conti malware using the first 320 top features of our AVERTs datasets (Figure 5.5a). In comparison, the SVM algorithm achieved a score of only 30% using the first 31 top features of the Malwarebytes datasets (Figure 5.5b). We also found that the algorithms performed better considering the recall metric when applied to the AVERTs dataset, scoring 100% (Figure 5.5c) compared to 80% for Malwarebytes datasets (Figure 5.5d). We found similar F1 scores (Figures 5.6a and 5.6b) and recall (Figures 5.6c and 5.6d) results for the Ryuk malware.

5.3.2.1 Model Implementation using the Nearest Neighbors algorithm and the AVERTs datasets

We selected the Nearest Neighbors algorithm to implement the Conti and Ryuk models since it had the best performance among all the algorithms using AVERTs datasets (Figures 5.5a, 5.5c, 5.6a, and 5.6c). We trained our models using the first 320 most relevant features (Figures 5.5 and 5.6) that contributed most to improving the performance of the Nearest Neighbors algorithm in identifying Conti and Ryuk malware activity. The results are shown in Table 5.1.

The Conti malware activity detection model (Table 5.1a) achieved a recall of 100% (i.e.,

Table 5.1: Performance of the Nearest Neighbors models trained using the first 320 most important features of AVERTs datasets for the Conti and Ryuk malware

a) CONTI MALWARE ACTIVITY MODEL SCORES				
	Precision	**Recall**	**F1-Score**	**Support**
Class				
0 (No activity)	1	0.36	0.53	80
1 (Activity)	0.6	**1**	**0.75**	76
Accuracy			0.67	156
Macro Avg	0.8	0.68	0.64	156
Weighted Avg	0.8	0.67	0.64	156
b) RYUK MALWARE ACTIVITY MODEL SCORES				
	Precision	**Recall**	**F1-Score**	**Support**
Class				
0 (No activity)	0.95	0.21	0.34	390
1 (Activity)	0.56	**0.99**	**0.71**	390
Accuracy			0.6	780
Macro Avg	0.75	0.6	0.53	780
Weighted Avg	0.75	0.6	0.53	780

it detected all the actual positive cases) and an F1 score of 75% (i.e., it has a good balance between precision and recall). In the case of Ryuk malware activity detection (Table 5.1b), we obtained almost the same results (99% recall, 71% F1).

5.3.3 Case Studies

To assess the effectiveness of our model in detecting malware activity using new unseen data, four cases of Conti and Ryuk malware attacks were collected from internet sources (e.g., webpages, official data breach reports, etc.). These events were classified as complete (i.e., attack and attack discovery dates were found) and incomplete (i.e., only the attack discovery date was found). The following is a brief description of each of the four cases.

Conti case - Meyer Corporation (Complete): Cookware distributor Meyer Corporation is based in Vallejo, California, USA. The corporation was a victim of a Conti attack on October 25, 2021. However, the attack was not discovered until December 01, 2021. The company's data was targeted during the ransomware attack, including names,

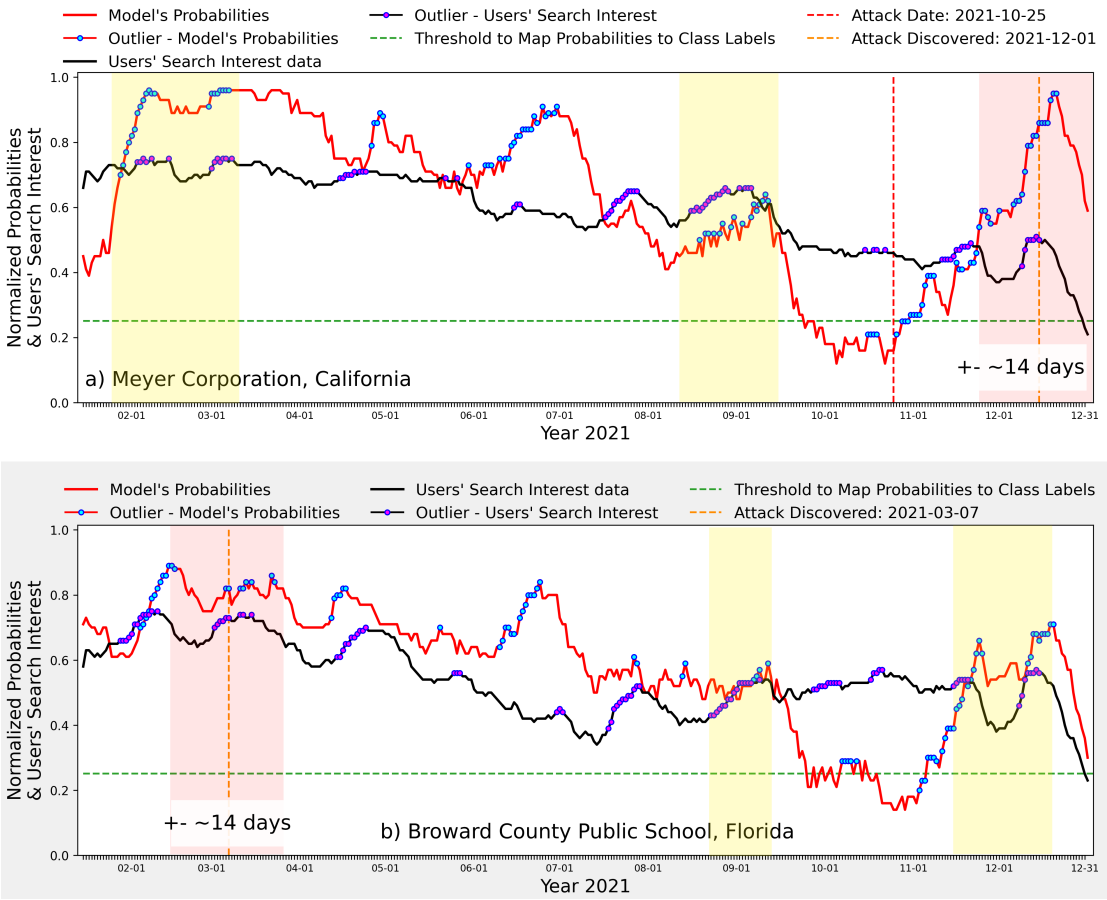

Figure 5.7: Case study: Conti cases in 2021 with (a) Complete (b) Incomplete data.

addresses, etc.

Conti case - Broward County Public Schools (Incomplete): On March 07, 2021, Florida's Broward County Public Schools announced that it had been victim to a ransomware attack, for which hackers demanded $40 million. Broward County Public Schools shut down their IT systems after discovering the attack.

Ryuk case - Riviera Beach, Florida (Complete): The Riviera Beach city was attacked on May 29, 2021. The attack disabled the city's official website, municipal employees' emails, voice-over-internet-protocol phones, and the local water utility's ability to take online payments. It also forced workers in the city's 911 dispatch center to record caller information on paper. Hackers demanded $600K.

Ryuk case - Lake City, Florida (Incomplete): Lake City's IT network was infected with malware on June 10, 2021. A city employee opened an email that contained the Emotet trojan, which later downloaded TrickBot and Ryuk ransomware. The latter spread to the city's entire IT network and encrypted files.

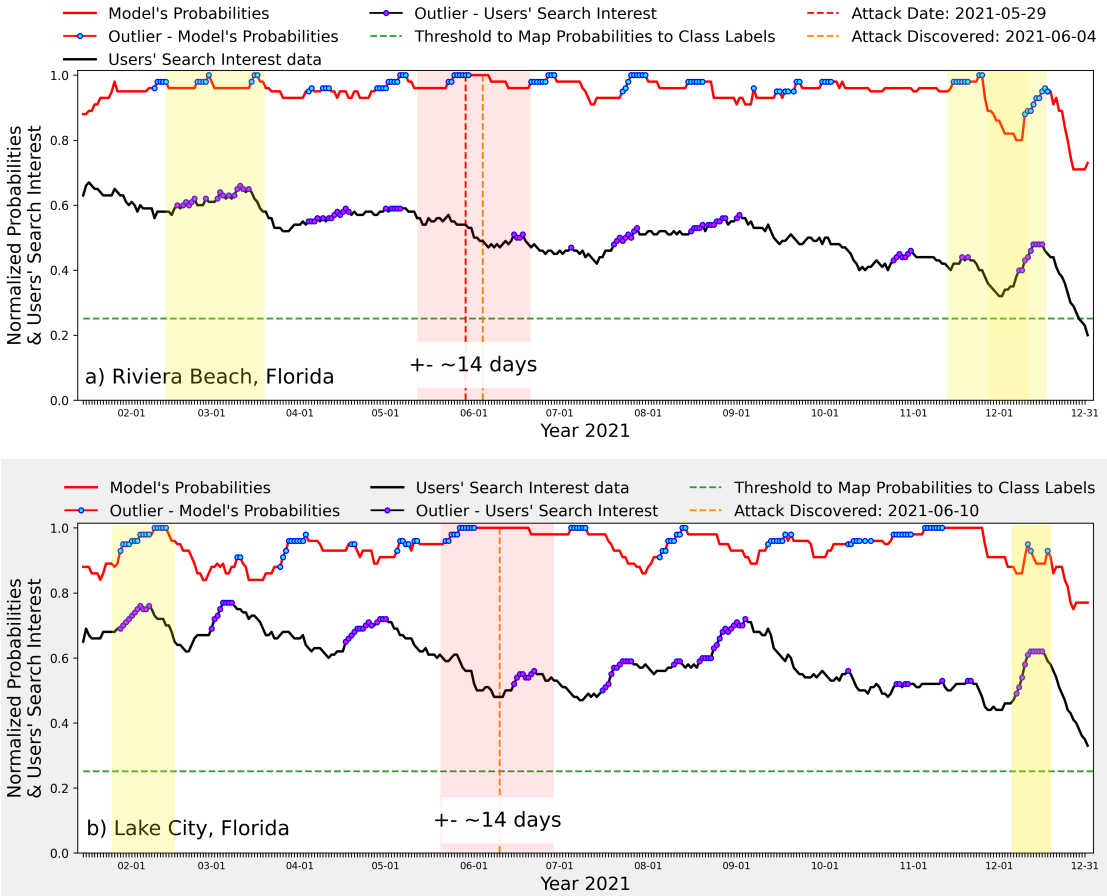

Figure 5.8: Case study: Ryuk cases in 2021 with (a) Complete (b) Incomplete data.

The Google Trends Scores for the four cases were retrieved using the same search terms that we used to construct the AVERTs datasets. The data was then used to detect malware activity. We present the results as time series plots in Figures 5.7 and 5.8. We compute a simple moving average for each time series to show the trends of our models' results and the users' search interests. In red are the probabilities returned by our models (i.e., probability of a positive class indicating malware activity), and in black are the sums of

the users' search interest scores.

Additionally, we detected outliers in the time series when their data points values were greater than the mean of the time series +1 standard deviation to detect unusual activity in both time series (probabilities and users' search interests). These outliers are plotted as blue dots (model probabilities for a positive class) and magenta dots (user search data) in Figures 5.7 and 5.8.

We also plotted three lines (red, orange, and green) for the complete cases and two lines (orange and green) for the incomplete cases. The vertical red line indicates the attack's date, and the orange vertical line indicates when the attack was first detected. The horizontal dotted green line is the threshold (computed during the optimization phase) that we used to identify malware activity.

Analyzing the Conti and Ryuk complete cases (Figures 5.7a and 5.8a), we found that users' search interest and our model's probabilities increased approximately seven days before and after the attack was publicly disclosed (dotted red line). It is possible to use these patterns to identify other possible attacks not reported by other organizations. For example, we examined the yellow areas shown in Figures 5.7a and 5.8a. Both areas were experiencing unusual activity and exhibited similar patterns to those observed when the attack was reported. We also noticed that there are outliers appearing simultaneously or with a slight delay of approximately seven days in both time series. According to this information, we can assume that possible attacks occurred during the yellow periods.

Furthermore, we analyzed the incomplete cases of Conti and Ryuk, in which the compromised organizations published only the date on which they discovered the attack (Figures 5.7b and 5.8b). Thus, there is no data on when the attacks started. In these scenarios, we found a recurring pattern. The outliers in the terms search volume (black) and model probability time series (red) showed an increase 14 days before and after the discovery of the attack (red areas).

This pattern can help to identify other unusual activities in the time series data. For example, in Figures 5.7b and 5.8b, the yellow areas show a similar pattern to the red area,

where outliers constantly increased in both time series. Based on the time series data, we assume that people's interest in Conti and Ryuk malware increased due to factors we cannot directly determine from the data. However, we can infer this interest is probably related to malware activity.

Moreover, we found a repeating pattern in which the time series declined continuously from their highest values after discovering the attack (orange line in Figures 5.7 and 5.8). This provides valuable information that can be used to detect dangerous malware activities. For example, the first yellow regions (from left to right) in Figures 5.7a, 5.8a, and 5.8b show that both time series reached their maximum values during that period. This pattern is similar to the red regions in Figures 5.7 and 5.8. Our analysis led us to conclude that when both time series are at their highest values and outliers are present, people are likely searching for information about malware. Our findings suggest that the patterns observed in the time series data can be used to detect past or ongoing malware activities.

5.4 Conclusions and Future Work

This paper presents an approach to detect malware activity using AVERTs datasets and people's web search interests. We constructed the AVERTs datasets using sentence embeddings to discover semantic similarities and topics between the CVE and ATT&CK databases. We evaluated and compared our approach using the AVERTs datasets and generic datasets (constructed with search terms from web pages). The results showed that classifier models trained using the AVERTs datasets have better performance in detecting malware activities in public search data. The Nearest Neighbors algorithm has the best performance among all the tested algorithms.

Throughout our experiments, we see that it is possible to detect unusual malware activity using AVERTs datasets and people's web search interests. We found that the number of searches performed by users and the probabilities of our model's detection increased approximately seven days before and after the announcement of the attack. In addition,

we observed an increase in outliers in the web search volume and model probability time series 14 days before and after the discovery of the attack. According to our results, when both time series are at their highest values and outliers are present, people are likely looking for malware information and a malware attack is likely ongoing. Future work includes constructing a single version of the AVERT datasets to be used in a multi-class classifier. A web interface to track various malware activities around the world also remains as a future work.

Chapter 6

Uncovering Threat Vectors through Attack Analysis

6.1 Introduction

As technology continues to evolve, the need for secure software and Industrial control systems (ICS) becomes increasingly critical. Cyberattacks and data breaches have become significant threats to individuals and organizations, resulting in significant damages [132] [133]. In response, researchers and industry professionals have focused on developing techniques to identify and mitigate software vulnerabilities to improve cybersecurity. One of these techniques is the automatic linkage between Common Vulnerabilities and Exposures (CVEs), Common Weakness Enumeration (CWEs), and Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK) techniques.

The CVE system was established in 1999 to provide a standardized method of identifying and tracking vulnerabilities in software systems[1]. The CWE system was introduced in 2006 as a classification system for software weaknesses that can lead to vulnerabilities[2]. The ATT&CK framework was created in 2015 as a knowledge base of adversarial tactics, techniques, and procedures that can be used to test and evaluate the security of software systems[3].

There are significant challenges in linking CVEs, CWEs, and ATT&CK techniques

[1]https://www.cve.org/About/History

[2]https://cwe.mitre.org/about/history.html

[3]https://attack.mitre.org/resources/faq/

automatically. Some researchers have focused on manually linking these datasets, which is time-consuming. Automatic linkage techniques have the potential to significantly improve the speed and efficiency of the vulnerability identification and mitigation process. Also, linking these systems can provide valuable insights into the tactics and techniques used by attackers, enabling better defense against cyber threats.

In this chapter, we aim to investigate the impact of software vulnerabilities and weaknesses on ICS attacks. These vulnerabilities are often exploited by cyber attackers to gain unauthorized access to computer systems, steal confidential data, or disrupt operations. By mapping vulnerabilities and weaknesses to attacker techniques, organizations can prioritize their remediation efforts and focus on the most critical areas of risk.

6.2 Approach

This section provides an overview of our proposed methodology, focusing on data collection and exploring the use of sentence embeddings for mapping CWEs to ICS ATT&CK techniques.

6.2.1 Data Collection

6.2.1.1 Common Weakness Enumeration (CWE)

Common Weakness Enumeration (CWE) is a comprehensive list of software vulnerabilities resulting from common software weaknesses and mistakes[4]. It is an essential tool for identifying, understanding, and addressing software security weaknesses. Using CWE, developers can detect potential security risks and issues early in the software development process, allowing them to proactively address security vulnerabilities before deployment.

In this work, we use the **CWE Research Concepts**, which consists of a set of methodologies designed to support research into software weaknesses, vulnerabilities, and related

[4]https://cwe.mitre.org/

issues[5]. These concepts focus on abstractions of behaviors, rather than specific detection methods or development life cycle stages. This allows researchers to analyze and categorize software weaknesses according to their underlying causes and behaviors. The CWE research concepts include various data elements, such as the description, extended description, modes of intrusion, and applicable platforms.

6.2.1.2 MITRE ATT&CK

This framework provides a comprehensive view of adversarial tactics, techniques, and procedures utilized by cyber attackers[6]. It helps security professionals to understand the attacker's mindset and techniques during different stages of a cyberattack, and develop effective security defenses and mitigation strategies to detect and respond to cyberattacks.

In this work, we used the **MITRE ATT&CK for Industrial Control Systems (ICS) Techniques**, which form a subset of the broader MITRE ATT&CK framework with a specific focus on ICS security[7]. These techniques offer a categorization of the tactics and methods employed by adversaries to infiltrate ICS environments. With this information, security professionals can design effective mitigation strategies and deploy robust security controls to minimize the risk of successful cyber-attacks on ICS systems. The ICS techniques include data such as the technique name, description, mitigation strategies, and detection mechanisms.

6.2.2 Automated Mapping MITRE ATT&CK and CWEs

We analyzed the text information of 1000 Common Weakness Enumeration (CWE) research concepts and 79 MITRE ATT&CK Industrial Control System (ICS) techniques using a document embedding approach. The text information included descriptions, extended descriptions, modes of intrusion, applicable platforms for the CWE, and descriptions for ICS

[5]https://cwe.mitre.org/data/definitions/1000.html

[6]https://attack.mitre.org/

[7]https://attack.mitre.org/techniques/ics/

techniques.

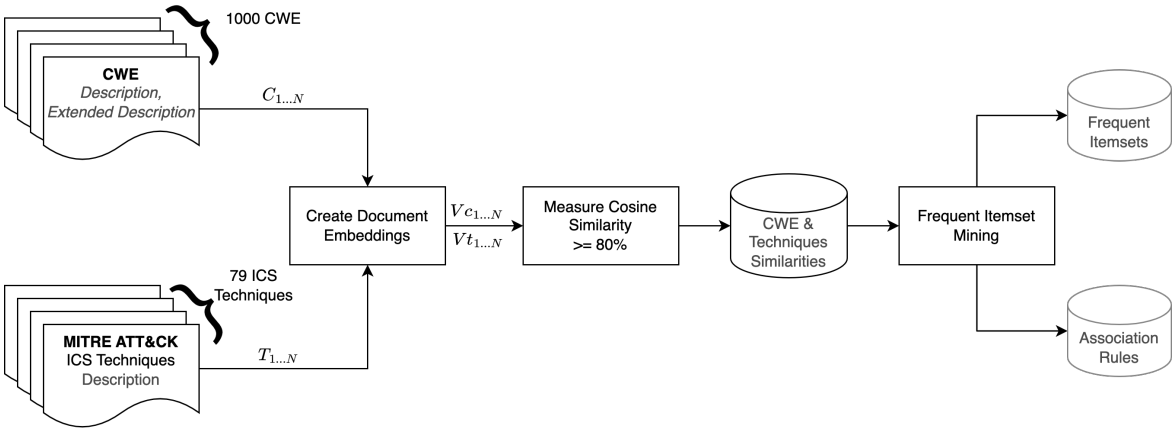

Figure 6.1: Approach: Text analysis of CWEs and MITRE ATT&CK ICS techniques using document embedding methods. Similarities between techniques and CWEs computed using TF-IDF, BERT, and Sentence-BERT approaches. The fp-growth algorithm was used to compute frequent itemsets and association rules between the datasets, revealing frequent co-occurrences of CWE research concepts and MITRE ATT&CK ICS techniques.

6.2.2.1 CWE and MITRE ATT&CK ICS Techniques similarities

To compute the similarity between documents, various methods can be employed. In this work, we tested the TF-IDF, BERT, and Sentence-BERT (SBERT) approaches to compute similarities (mapping) between MITRE ATT&CK ICS techniques and CWEs.

These methods can be grouped into two categories, word count-based and inference-based. Word count-based methods (e.g., TF-IDF [134]), are relatively simple and determine similarity based on the frequency of occurrence of words in a document. While this approach may be effective in identifying documents that share similar keywords, it may not capture the underlying meaning of the text. On the other hand, inference-based methods (e.g., BERT [135] and SBERT [123]) employ machine learning models to learn the meaning and context of the text, resulting in a more precise similarity score.

The main goal of using these three approaches was to evaluate their effectiveness in identifying similarities between MITRE ATT&CK ICS techniques and CWEs. We aimed to determine the most effective approach for use in our subsequent experiments. The efficacy of each approach was assessed by comparing their cosine similarity results.

6.2.2.2 Semantic Similarity

We employed a similar approach used in [136] that uses a Siamese neural network architecture with two identical BERT models, sharing the same parameters and weights, to measure the semantic similarity between CWE and ICS techniques. The Siamese neural network processed the CWE and ICS techniques documents through the BERT models and utilized a mean pooling technique to reduce their dimension, producing vectors of 768 dimensions each. To compute sentence embeddings, we used the pre-trained msmarco-distilbert-base-tas-b model[8]. We then computed the cosine similarity between the CWE and ICS techniques vectors, using a similarity threshold of 0.8. A similarity score of 0.8 or higher was considered a strong relationship between a CWE and a technique. The results were stored in a local database for future use.

6.2.2.2.1 MSMARCO DistilBERT base TAS-B model We used the pre-trained msmarco-distilbert-base-tas-b model to compute our sentence embeddings. The MSMARCO DistilBERT base T5-Base model, also known as the MSMARCO DistilBERT base TAS-B model, is a highly advanced language model designed to perform various natural language processing tasks such as question answering and text classification. It is based on the DistilBERT architecture and has been fine-tuned on a vast dataset of real-world web documents from the Microsoft MARCO dataset. Specifically, it has been optimized for the MS Marco Passage Ranking task, which requires the model to retrieve the most relevant passage from a set of candidate passages given a query. The model's efficiency is one of its key features, as it is capable of processing vast amounts of data quickly.

[8]https://huggingface.co/sentence-transformers/msmarco-distilbert-base-tas-b

6.2.3 Frequent Itemset Mining

Our study aims to analyze the relationships between the Common Weakness Enumeration (CWE) research concepts and MITRE ATT&CK Industrial Control System (ICS) techniques. To achieve this, we used the fp-growth algorithm to compute frequent itemsets and association rules between the two datasets.

The fp-growth algorithm is a robust and efficient data mining technique that is commonly used to discover patterns in large datasets [137]. It works by building a tree structure that represents the data, where each branch of the tree corresponds to an item in the dataset. The algorithm then uses this tree to efficiently generate frequent itemsets and association rules.

To apply the fp-growth algorithm in our study, we first converted the text information of the CWE research concepts and MITRE ATT&CK ICS techniques into numerical vectors using a document embedding approach, as described in Section 6.2.2. We then created a dataset in which each ICS technique was considered a transaction, with the CWEs similar to the technique as items of the transaction. The resulting dataset contained 79 transactions (rows), with each transaction consisting of a variable number of items (CWEs). It is important to mention that the items in each row were sorted based on their cosine similarity with the techniques, as shown in Figure 6.2. We then used the fp-growth algorithm to compute frequent itemsets and association rules using the dataset created.

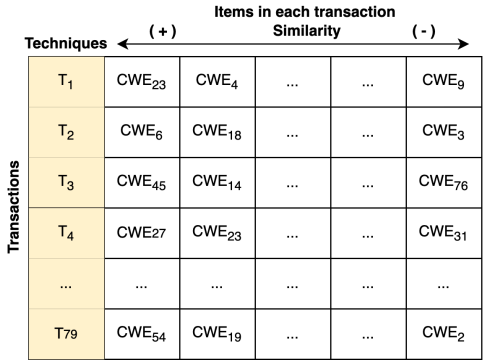

Figure 6.2: Transactions.

The frequent itemsets generated by the fp-growth algorithm represent sets of CWE research concepts and MITRE ATT&CK ICS techniques that frequently occur together. These frequent itemsets were then used to identify patterns in the data and gain insights into the relationships between the two sets of data. The association rules generated by the algorithm describe the relationships between these frequent itemsets and can be used to make predictions about future occurrences of these patterns.

6.2.3.1 Case Studies Data Extractions

We developed a web scraping tool to collect data from the Industrial Control System (ICS) techniques page of the MITRE ATT&CK framework with the objective of identifying the techniques used by attackers to compromise ICS systems and the related CWEs. To accomplish this, we used Python along with the Requests and BeautifulSoup libraries to request and parse HTML content from the ICS techniques page.

Our tool works by looping through specific tags within the webpage and identifying relevant links containing technique ID information. Once a technique is identified, the tool visits the associated page to extract additional information such as ICS advisories and Common Weakness Enumeration (CWE) details. To extract ICS advisories, the tool scans for links containing the substring "ICS-" and follows them to obtain the associated CWE information. Similarly, for ICS advisories on the US-CERT website, the tool searches for links containing the substring "ICSA-".

We utilized the data generated by our web scraping tool to create a dataset (See Table 6.4) that contains information on each technique ID, ICS Alert ID, ICS advisory ID, and the associated CWE(s). It is worth mentioning that the CWEs were extracted from the ICS advisories. However, in some instances, the techniques lacked advisory information, and as such, the ICS advisories had to be derived from the ICS Alerts. This data is used in the analysis of the case studies.

Additionally, we created Table 6.1 which provides definitions for the terms used in the analysis, including ICS Alerts, ICS Advisories, and CWEs. This table is intended to aid in

the interpretation of the data presented in Table 6.4.

Table 6.1: ICS Alerts, ICS Advisories, and CWEs found during the web scraping process

Alert	**Description**
ICS-ALERT-14-281-01B	Russian spear-phishing targets ICSs.
Advisoy	**Description**
ICSA-14-023-01	GE CIMPLICITY: path traversal vulnerability.
ICSA-14-329-02D	Siemens WinCC: unauthenticated remote code execution.
ICSA-18-107-02	Schneider Triconex 3008: arbitrary code execution, system shutdown, or the compromise of safety systems.
CWE	**Description**
CWE-22	Directory traversal allows unauthorized access.
CWE-284	Product lacks proper access control mechanisms.
CWE-119	Unintended memory access leads to vulnerability.

6.3 Results and Discussion

In the following sections, we present the findings of our study on the relationship between software vulnerabilities and weaknesses in industrial control systems (ICS) attacks.

6.3.1 Effectiveness of NLP in Linking Security Weaknesses and Attack Techniques in ICS

In this work, we aimed to compute the similarity between MITRE ATT&CK ICS Techniques and CWEs using three different approaches: SBERT, BERT, and TFIDF. The results are presented in Table 6.2, which includes 11 columns. The first two columns describe the techniques ID and their description, while the next nine columns are grouped in threes and show the results for each approach. Specifically, columns 3 to 5 show the CWEs ID, their descriptions, and the similarity scores obtained using the SBERT approach, while columns 6 to 8 and 9 to 11 show the same information for the BERT and TFIDF approaches, respectively. The CWEs identified as top 1 by all three approaches are highlighted in yellow, while the highest similarity score is denoted in green.

Our results demonstrate that SBERT outperformed BERT and TFIDF in most cases, with an average similarity score of 0.82 compared to 0.81 and 0.29 for BERT and TFIDF, respectively. One possible reason for this is that SBERT is designed, using Siamese and

Table 6.2: Performance of the SBERT and BERT approaches

TECHNIQUE		SBERT			BERT		
ID	Description	CWE	Description	Sim	CWE	Description	Sim
T0800	Adversaries can halt device functions.	1351	Inadequate device security when cooled.	0.84	1277	No firmware updates or patches.	0.82
T0801	Adversaries gather process state information.	205	Product reveals sensitive information differences.	0.82	205	Product reveals important internal differences.	0.81
T0802	Adversaries automate collection of information.	1093	Software has complex data structures.	0.79	1093	Software has complex representation structure.	0.79
T0803	Adversaries can block command messages.	1320	Agents can disable signal alerts.	0.86	1320	Agents can disable signal alerts.	0.83
T0804	Adversaries may block reporting messages.	1320	Agents can disable signal alerts.	0.83	1295	Product exposes sensitive system information.	0.80
T0805	Adversaries may block device communication.	5	Network information vulnerable to attack.	0.82	1319	Device vulnerable to electromagnetic attacks.	0.80
T0806	Repeated I/O changes manipulate processes.	334	Low randomness leads to vulnerability.	0.81	123	Arbitrary write exploit via buffer overflow.	0.81
T0807	Adversaries exploit command-line interfaces.	749	API includes dangerous unrestricted function.	0.85	749	Unsafe API poses external threats.	0.82
T0809	Adversaries destroy data, leave malware.	508	Non-replicating malicious code targets one.	0.85	508	Non-replicating code targets one system.	0.84
T0811	ICS data collection by adversaries.	921	Sensitive data stored insecurely.	0.82	540	Web server code sensitive information.	0.80
T0812	Adversaries use default control passwords.	1391	Product has weak credentials for security.	0.84	1392	Default credentials for critical functions.	0.82
T0813	Adversaries may deny process control.	280	Insufficient privileges cause unexpected behavior.	0.83	537	Attacks exploit unhandled exception errors.	0.83
T0814	Denial-of-service attacks disrupt devices.	589	API function inconsistently available, risk.	0.83	589	API function causes portability problems.	0.82
T0815	Denial of view by adversaries.	435	Integration error causes incorrect behavior.	0.82	537	Unhandled exceptions can be exploited.	0.81
T0816	ICS devices can be disrupted.	455	Security errors ignored, affecting integrity.	0.80	1319	Device vulnerable to electromagnetic attacks.	0.78
T0817	Drive-by compromises exploit web browsers.	521	Weak passwords allow easier hacking.	0.81	521	Weak passwords compromise user accounts.	0.80
T0819	Weaknesses in internet-facing software exploited.	637	Complex software may cause weaknesses.	0.85	537	Attacker exploits unhandled exception errors.	0.83
T0820	Exploitable software vulnerabilities enable evasion.	655	Protective software easily disabled.	0.87	396	Broad exceptions increase error complexity.	0.85
T0821	Adversaries modify controller tasking.	279	Software violates user permissions unintentionally.	0.85	114	Untrusted source can be dangerous.	0.82
T0822	External services allow network access.	662	Issues with resource sharing synchronization.	0.81	27	Path traversal vulnerability due to ../.	0.79
T0823	GUI enhances machine execution capabilities.	749	Unrestricted interface contains dangerous function.	0.83	451	Poor UI can enable phishing.	0.81
T0826	Adversaries might disrupt essential components.	511	Malicious code disrupts software operation.	0.85	356	Lack of warning on unsafe actions.	0.82
T0827	Adversaries aim for loss of control.	1320	Agents may disable alerts, response.	0.83	1320	Disable alerts, compromise response mechanism.	0.81
T0828	Adversaries disrupt productivity and revenue.	537	Exceptions allow unauthorized system access.	0.79	537	Unhandled exceptions allow system access.	0.79
T0829	Adversaries cause equipment view loss.	537	Exceptions exploited for unauthorized access.	0.81	537	Unhandled exceptions lead to attacks.	0.81
T0830	Adversaries modify network traffic attacks.	5	Data can be hacked in transit.	0.82	537	Attacker exploits unhandled exceptions.	0.80
T0831	Manipulation of industrial control systems.	214	Sensitive information in process invocation.	0.82	435	Integration errors cause system weaknesses.	0.81
T0832	Manipulation of reported information possible.	202	Statistics can reveal confidential information.	0.83	1118	Inadequate documentation on error handling.	0.81
T0834	Adversaries access OS functions directly.	749	API includes dangerous unrestricted function.	0.88	749	Insecure API for external actors.	0.84
T0835	PLC I/O image manipulation explained.	1278	Hardware information recoverable through imaging.	0.79	1278	Attacker can recover hardware information.	0.79
T0836	Hackers can alter control system parameters.	1112	Incomplete document on program control.	0.83	1112	Incomplete description of program control.	0.81
T0837	Compromised protective systems cause hazards.	655	Protection mechanism encourages disabling.	0.83	655	Protection mechanism too difficult/inconvenient.	0.82
T0838	Adversaries can alter alarm settings.	11	Debugging messages aid system attacks.	0.85	1295	Product exposes sensitive system information.	0.83
T0839	Adversaries target vulnerable firmware devices.	655	Protection mechanism discourages non-malicious users.	0.84	508	Malicious code stays on target.	0.83
T0840	Adversaries can discover device communication patterns.	1293	Limited data source hinders detection.	0.83	1293	Single-source data limits threat detection.	0.80
T0842	Monitoring data with network interface.	294	Software flaw enables network bypass.	0.83	319	Insecure software exposes sensitive data.	0.80
T0843	Adversaries transfer user program to controller.	434	Software enables dangerous file transfers.	0.84	434	Software enables dangerous file uploads.	0.82
T0845	PLC program upload security risks.	434	Software upload enables dangerous files.	0.84	434	Software enables dangerous file transfer.	0.82
T0846	Adversaries gather system information.	1293	Software uses single data source.	0.81	52	Slash input vulnerability in software.	0.80
T0847	Malware on removable media accesses systems.	497	Access control vulnerability in app.	0.82	537	Attacker exploits unhandled exception errors.	0.81
T0848	Rogue masters can cause chaos.	514	Covert channels transfer unintended information.	0.82	301	Reflection attacks exploit simple authentication.	0.81
T0849	Masquerading disguises malicious content's identity.	646	Software vulnerable to file misclassification.	0.83	646	Software vulnerable to misclassification attacks.	0.81
T0851	Rootkits hide malware on systems.	1061	Lack of data security software.	0.84	1326	Hardware flaw allows boot bypass.	0.82
T0852	Adversaries capture screen of devices.	1278	Hardware information can be recovered.	0.82	1278	Hardware data can be recovered.	0.80
T0853	Adversaries exploit scripting languages, dangers.	211	Errors reveal sensitive system info.	0.84	211	Application triggers external error message.	0.83
T0855	Unauthorized commands pose control system risk.	211	Application triggers external error message.	0.84	204	Product exposes internal state information.	0.82
T0856	Spoofing reporting messages impairs control.	211	App triggers external sensitive error.	0.82	1295	Product inadequately protects sensitive info.	0.80
T0857	Updating firmware for modern devices.	1277	No firmware update available for product.	0.88	1277	No firmware updates or patches.	0.85
T0858	Adversaries exploit controller operating modes.	270	Software mismanages privilege switching.	0.84	270	Software fails to manage privileges.	0.83
T0859	Credential access enables adversary access.	1391	Weak credentials can be guessed.	0.85	537	Attacker exploits errors for access.	0.84
T0860	Wireless compromise for unauthorized access.	5	Network data vulnerable to compromise.	0.83	1279	Input validation essential for cryptography.	0.80
T0861	Adversaries collect points and tags.	214	Sensitive information in process invocation.	0.82	694	Duplicate identifiers in required contexts.	0.79
T0862	Supply chain compromise for access.	1229	Indirect resource creation for attackers.	0.85	1279	Cryptographic risk without input validation.	0.83
T0863	Adversaries use user interaction for malware.	511	Code disrupts software and environment.	0.85	508	Non-replicating malware stays put.	0.83
T0864	Transient assets targeted by adversaries.	1299	Attackers can bypass primary path.	0.82	1299	Attacker can bypass primary protections.	0.79
T0865	Spearphishing with malware attachment explained.	434	Software enables dangerous file uploads.	0.82	434	Software enables dangerous file transfer.	0.80
T0866	Exploiting software for remote access.	511	Malicious code disrupts legitimate software.	0.83	537	Attacker exploits unhandled exceptions errors.	0.82
T0867	Adversaries transfer files for sabotage.	434	Software uploads dangerous files automatically.	0.82	434	Software sends dangerous file types.	0.81
T0868	Adversaries gather PLC information states.	1313	Runtime hardware allows test/debugging.	0.81	88	Command string lacks proper delimiting.	0.80
T0869	Adversaries exploit common protocols for control.	5	Network data can be intercepted.	0.81	301	Authentication vulnerable to reflection attacks.	0.79
T0871	APIs vulnerable to adversary attacks.	749	API allows dangerous function access.	0.90	749	Unrestricted dangerous function in API.	0.86
T0872	Removing evidence of system presence.	1293	Single data source lacks security.	0.83	1295	Product leaks sensitive debugging information.	0.81
T0873	Malware can infect project files.	511	Software disrupts program or environment.	0.84	511	Code designed to disrupt software.	0.81
T0874	API functions vulnerable to attacks.	749	Unsafe API function not restricted properly.	0.84	589	API function causes portability and security issues.	0.82
T0877	PLC input/output table vulnerability.	1278	Hardware data recoverable via microscopy.	0.80	1278	Data recoverable from hardware images.	0.80
T0878	Adversaries target protection function alarms.	11	Debugging reveals system vulnerabilities.	0.84	1320	Risk of disabling signal alerts.	0.82
T0879	Adversaries cause damage to infrastructure.	511	Code disrupts software's operation intentionally.	0.82	508	Non-replicating code targets one system.	0.80
T0880	Adversaries may compromise safety systems.	267	Privilege can cause unintended danger.	0.84	435	Integration error causes system weaknesses.	0.82
T0881	Adversaries stop services for sabotage.	511	Software contains disruptive code.	0.85	589	API function causes portability issues.	0.82
T0882	Adversaries seek operational information valuable.	205	Product reveals important differences to observers.	0.81	435	Interaction errors cause system weakness.	0.78
T0883	Adversaries infiltrate industrial environments remotely.	537	Attacker exploits unhandled exceptions errors.	0.85	537	Unhandled exceptions allow unauthorized access.	0.84
T0884	Adversaries can use connection proxy.	444	Product acts as intermediary HTTP agent.	0.85	444	HTTP agent with limited interpretation.	0.82
T0885	Adversaries bypass firewalls through ports.	5	Network data vulnerable to attack.	0.83	605	Multiple sockets cause port conflicts.	0.80
T0886	Adversaries exploit remote services.	697	Security software produces incorrect comparisons.	0.81	351	Software allows insecure behavior.	0.80
T0887	Adversaries target RF communication in distributed environments.	514	Information transfer via unintended path.	0.79	326	Weak encryption used for sensitive data.	0.78
T0888	Adversary seeks system information targeting.	385	Covert channels convey information through behavior.	0.83	385	Secret channels convey protected information.	0.81
T0889	Adversaries can modify controller programs.	269	Flaw in software privilege management.	0.83	269	Software lacks proper privilege management.	0.81
T0890	Adversaries exploit software for privileges.	250	Software creates high-level weaknesses.	0.88	250	Software operates at high privilege.	0.86
T0891	Hardcoded credentials provide unauthorized access.	1391	Product has weak credentials vulnerability.	0.84	537	Unhandled exceptions lead to cybersecurity breach.	0.82
			SBERT Sim Avg	0.83		BERT Sim Avg	0.81

triplet networks to learn sentence representations and encode them in high-dimensional vector spaces [123]. This enables SBERT to capture semantic relationships between sentences, allowing it to find similarities that may not be evident with other approaches. BERT, on the other hand, is better suited for capturing syntactic relationships between words, and its ability to capture semantic relationships is limited [135].

Another reason why SBERT may have performed better than BERT and TFIDF is due to its training on large amounts of data. This allows SBERT to learn the underlying meaning of words and phrases, even if they are not explicitly related. In contrast, TFIDF only considers the frequency of individual words in a document, which can make it less effective in measuring semantic similarity between documents [134].

Our results suggest that SBERT is the most effective approach for measuring similarity between MITRE ATT&CK ICS Techniques and CWEs, outperforming both BERT and TFIDF. Therefore, we used SBERT to conduct further experiments and analysis.

Table 6.3: Case Studies

Case Study	Description	CWE	Description
GE CIMPLICITY	Path traversal vulnerability	CWE-22	Improper Limitation of a Pathname to a Restricted Directory
Siemens WinCC	Unauthenticated remote code execution	CWE-284	Improper Access Control
Schneider Triconex 3008	Arbitrary code execution	CWE-119	Improper Restriction of Operations within the Bounds of a Memory Buffer

6.3.2 Enhancing Security Weakness Identification in ICS with NLP and Frequent CWE Itemsets: Case Studies

We use the case studies presented in Table 6.3 to evaluate our approach for identifying ICS Techniques and Common Weakness Enumeration (CWE) that co-occur in the context of an attack using natural language processing techniques and frequent itemsets.

Table 6.4: Relationship between the ICS Alerts, ICS Advisories, CWEs, and MITRE ATT&C ICS Techniques found during the web scraping process

Technique	Description	Alert	Advisory	CWE
T0803	Adversaries can block command messages.	ICS-ALERT-14-281-01B	ICSA-14-023-01 ICSA-14-329-02D	CWE-22 CWE-284
T0804	Adversaries may block reporting messages.	ICS-ALERT-14-281-01B	ICSA-14-023-01 ICSA-14-329-02D	CWE-22 CWE-284
T0807	Adversaries exploit command-line interfaces.	ICS-ALERT-14-281-01B	ICSA-14-023-01 ICSA-14-329-02D	CWE-22 CWE-284
T0816	ICS devices can be disrupted.	ICS-ALERT-14-281-01B	ICSA-14-023-01 ICSA-14-329-02D	CWE-22 CWE-284
T0819	Weaknesses in internet-facing software exploited.	ICS-ALERT-14-281-01B	ICSA-14-023-01 ICSA-14-329-02D	CWE-22 CWE-284
T0820	Exploitable software vulnerabilities enable evasion.		ICSA-18-107-02	CWE-119
T0821	Adversaries modify controller tasking.		ICSA-18-107-02	CWE-119
T0822	External services allow network access.	ICS-ALERT-14-281-01B	ICSA-14-023-01 ICSA-14-329-02D	CWE-22 CWE-284
T0823	GUI enhances machine execution capabilities.	ICS-ALERT-14-281-01B	ICSA-14-023-01 ICSA-14-329-02D	CWE-22 CWE-284
T0834	Adversaries access OS functions directly.		ICSA-18-107-02	CWE-119
T0843	Adversaries transfer user program to controller.		ICSA-18-107-02	CWE-119
T0845	PLC program upload security risks.		ICSA-18-107-02	CWE-119
T0846	Adversaries gather system information.		ICSA-18-107-02	CWE-119
T0849	Masquerading disguises malicious content's identity.	ICS-ALERT-14-281-01B	ICSA-14-023-01 ICSA-14-329-02D ICSA-18-107-02	CWE-22 CWE-284 CWE-119
T0853	Adversaries exploit scripting languages, dangers.	ICS-ALERT-14-281-01B	ICSA-14-023-01 ICSA-14-329-02D ICSA-18-107-02	CWE-22 CWE-284 CWE-119
T0855	Unauthorized commands pose control system risk.	ICS-ALERT-14-281-01B	ICSA-14-023-01 ICSA-14-329-02D	CWE-22 CWE-284
T0857	Updating firmware for modern devices.	ICS-ALERT-14-281-01B	ICSA-14-023-01 ICSA-14-329-02D ICSA-18-107-02	CWE-22 CWE-284 CWE-119
T0858	Adversaries exploit controller operating modes.		ICSA-18-107-02	CWE-119
T0859	Credential access enables adversary access.	ICS-ALERT-14-281-01B	ICSA-14-023-01 ICSA-14-329-02D	CWE-22 CWE-284
T0865	Spearphishing with malware attachment explained.	ICS-ALERT-14-281-01B	ICSA-14-023-01 ICSA-14-329-02D	CWE-22 CWE-284
T0867	Adversaries transfer files for sabotage.	ICS-ALERT-14-281-01B	ICSA-14-023-01 ICSA-14-329-02D	CWE-22 CWE-284
T0868	Adversaries gather PLC information states.		ICSA-18-107-02	CWE-119
T0869	Adversaries exploit common protocols for control.		ICSA-18-107-02	CWE-119
T0871	APIs vulnerable to adversary attacks.		ICSA-18-107-02	CWE-119
T0872	Removing evidence of system presence.		ICSA-18-107-02	CWE-119
T0874	API functions vulnerable to attacks.		ICSA-18-107-02	CWE-119
T0880	Adversaries may compromise safety systems.		ICSA-18-107-02	CWE-119
T0884	Adversaries can use connection proxy.	ICS-ALERT-14-281-01B	ICSA-14-023-01 ICSA-14-329-02D	CWE-22 CWE-284
T0885	Adversaries bypass firewalls through ports.		ICSA-18-107-02	CWE-119
T0886	Adversaries exploit remote services.	ICS-ALERT-14-281-01B	ICSA-14-023-01 ICSA-14-329-02D	CWE-22 CWE-284
T0890	Adversaries exploit software for privileges.		ICSA-18-107-02	CWE-119

6.3.2.1 Case Study: General Electric (GE) Proficy human-machine interface/supervisory control and data acquisition (HMI/SCADA) - CIMPLICITY application

Researchers amisto0x07 and Z0mb1E of Zero Day Initiative (ZDI) identified two vulnerabilities in the General Electric (GE) Proficy human-machine interface/supervisory control and data acquisition (HMI/SCADA) - CIMPLICITY application[9]. This software is a widely-used Client/Server-based HMI/SCADA application that allows operators to monitor and control industrial processes in real-time, across multiple industries. It is commonly used in critical infrastructure such as power plants, oil and gas refineries, and water treatment plants. GE released security advisories, GEIP13-05 and GEIP13-06, to inform customers about these vulnerabilities.

The vulnerabilities identified by the attackers could be exploited remotely and affect multiple GE Intelligent Platforms products, including Proficy HMI/SCADA - CIMPLICITY, Version 4.01 to 8.2, and Proficy Process Systems with CIMPLICITY. Specifically, CIMPLICITY CimWebServer does not check the location of the shell files, which may allow an unauthenticated user to load shell code from a remote location instead of the default local directory. The impact of these vulnerabilities depends on various factors unique to each organization, such as the operational environment, architecture, and product implementation.

Based on our data scraped from the MITRE ATT&CK website, it appears that the weakness in the GE CIMPLICITY application is related to a path traversal attack (CWE-22). This vulnerability allows an attacker to bypass system restrictions and gain unauthorized access to files or directories outside the intended scope of access. The analysis using our approach indicates similarities between CWE-22 and six techniques, T0803, T0804, T0807, T0849, T0857, and T0859, which can all be associated with path traversal attacks and used to conduct them. For instance, attackers may block command messages that

[9]https://www.cisa.gov/news-events/ics-advisories/icsa-14-023-01

prevent the traversal of a restricted directory (T0803) and gain access to unauthorized files. Attackers may also hide their access by blocking reporting messages that indicate a path traversal has occurred (T0804). Attackers can use command-line interfaces (CLIs) to execute path traversal attacks (T0807) by changing the current working directory to a parent directory and specifying a path that traverses into a restricted directory. Moreover, attackers may use masquerading (T0849) to camouflage a malicious file as a legitimate file located within a restricted directory. They may also exploit the firmware update feature to modify the system's behavior to allow path traversals (T0857) by creating a malicious firmware update that bypasses directory restrictions and installing it on the system. In the same way, attackers may steal credentials (T0859) through phishing, keylogging, or brute force attacks to gain access to files outside the intended directory.

On the other hand, the results of our analysis indicate that while the data scraped from the MITRE ATT&CK website suggests an association between the T0816, T0819, T0822, T0823, T0853, T0855, T0865, T0867, T0884, and T0886 techniques and the CWE-22 vulnerability, our approach did not reveal any similarities between them. For example, T0816 exploits the built-in functionalities of a device to shut it down, while T0819 exploits vulnerabilities in internet-facing software to gain initial access to a network. Neither of these techniques relies on a path traversal vulnerability like CWE-22.

Similarly, T0822 uses remote services like VPNs to gain initial access to a network, and T0823 aims to gain access to a machine through its GUI interface. Although these techniques could potentially be used with a path traversal vulnerability, our analysis found that they do not inherently involve such a vulnerability. T0853 involves using scripting languages to execute arbitrary code, and T0855 involves sending unauthorized command messages to control system assets. However, neither technique relies on a path traversal vulnerability. Also, T0865 uses social engineering tactics to convince targets to open attachments, while T0867 exploits weaknesses in file-sharing protocols to transfer files between systems. These techniques could also potentially be used with a path traversal vulnerability, but they do not inherently involve such a vulnerability. T0884 involves using a

connection proxy to direct network traffic, while T0886 exploits weaknesses in remote access mechanisms such as RDP or SSH. Like the other techniques, neither of these exploits relies on a path traversal vulnerability. Instead, they exploit weaknesses in the system's defenses and access mechanisms.

We assume that the information on the MITRE ATT&CK website was manually linked by an expert, and our methodology was unable to identify the similarities between the T0816, T0819, T0822, T0823, T0853, T0855, T0865, T0867, T0884, and T0886 techniques and the CWE-22 weaknesses using only the textual information provided on the MITRE websites.

6.3.2.2 Case Study: Siemens SIMATIC WinCC, PCS7, and TIA Portal Vulnerabilities

Siemens identified two vulnerabilities in the SIMATIC WinCC application, according to an updated advisory from the National Cybersecurity and Communications Integration Center (NCCIC) [10]. The affected products include SIMATIC WinCC, SIMATIC PCS 7, and TIA Portal V13. These vulnerabilities could allow for unauthenticated remote code execution, which means that an attacker could exploit the vulnerabilities remotely without authentication.

SIMATIC WinCC is a supervisory control and data acquisition (SCADA) system, while PCS7 is a distributed control system (DCS) integrating SIMATIC WinCC. TIA Portal is engineering software for SIMATIC products. These products are used in several sectors, including chemical, energy, food and agriculture, and water and wastewater systems, and are primarily used in the United States and Europe with a small percentage in Asia.

Our approach and the data scraped from the MITRE ICS Techniques revealed similarities between the CWE-284 weakness category (Improper Access Control) and several MITRE ICS Techniques, including T0804, T0807, T0816, T0819, T0822, T0857, T0859, and T0886. As we mentioned earlier, the vulnerabilities identified in the SIMATIC WinCC

[10]https://www.cisa.gov/news-events/ics-advisories/icsa-14-329-02d

application allow attackers to perform unauthenticated remote code execution, meaning that they can exploit the vulnerabilities remotely without authentication, which is related to CWE-284.

Attackers can gain unauthorized access to systems and execute commands or actions without proper authentication or authorization by leveraging weaknesses similar to CWE-284, as specified in T0807 and T0857. They can also exploit weaknesses in internet-facing software or remote services to bypass access controls designed to protect against external and internal threats, gaining initial access to a network or moving between assets and network segments, as we can read in T0819, T0822, and T0886.

Moreover, attackers may leverage denial-of-service attacks to prevent proper access to reporting systems or device control systems, as read in T0804 and T0816. They may also steal credentials using credential access techniques to gain unauthorized access to resources and systems that rely on those credentials, as read in T0859.

On the other hand, our approach did not initially identify any similarities between the CWE-284 and the techniques T0803, T0823, T0849, T0853, T0855, T0865, T0867, and T0884. However, upon analyzing the data scraped from the MITRE ATT&CK ICS Techniques, we discovered that there were indeed similarities. We found that two distinct groups of these techniques can be used to conduct an attack using the CWE-284. The first group includes three techniques: blocking a command message (T0803), accessing a GUI interface (T0823), and using masquerading to disguise a malicious file (T0849). These techniques may or may not be specific to exploiting weaknesses in the control system. The second group includes five techniques: using scripting languages to execute arbitrary code (T0853), sending unauthorized command messages (T0855), spearphishing (T0865), transferring files between systems (T0867), and using a connection proxy (T0884). These techniques are directly related to exploiting a vulnerability in the control system. Our analysis revealed that sending unauthorized command messages (T0855) is a crucial approach for an attacker to carry out a successful attack using CWE-284. However, it's important to note that the specific techniques used in an attack depend on the attacker's goals. For

instance, if the objective is to gain access to credentials, spearphishing (T0865) may be a more effective technique than blocking a command message (T0803).

Our approach was unable to detect similarities between the CWE-284 and the techniques T0803, T0823, T0849, T0853, T0855, T0865, T0867, and T0884, which we attribute to the limited information in the technique descriptions. It is possible that certain details that could potentially reveal similarities were not included in the descriptions, leading to our approach's inability to identify them.

Also, we believe that the data presented on the MITRE website was curated manually by experts, which could account for the discrepancies in our findings. It is possible that the experts responsible for collecting the data identified similarities that were not apparent in the technique descriptions alone.

Despite these limitations, our approach identified similarities for eight techniques (T0804, T0807, T0816, T0819, T0822, T0857, T0859, and T0886) that could potentially be used to conduct an attack using the CWE-284. Our approach provides valuable insights into the techniques used in attacks targeting control systems, which could prove useful for future research and development of more effective mitigation strategies and security measures.

6.3.2.3 Case Study: Schneider Electric Triconex Tricon. Improper Restriction of Operations within the Bounds of a Memory Buffer

Schneider Electric is a multinational corporation that specializes in energy management and automation solutions. One of its products is the Triconex Tricon Model 3008, which is a safety instrumented system used in industrial control systems (ICS). This system is designed to protect against potentially hazardous situations that may occur during industrial processes.

Recently, it was discovered that the Triconex Tricon Model 3008 contains vulnerabilities that could be exploited by attackers. Specifically, the vulnerabilities are related to an improper restriction of operations within the bounds of a memory buffer[11]. This could

[11]https://www.cisa.gov/news-events/ics-advisories/icsa-18-107-02

allow attackers to remotely access and take control of the system, potentially leading to arbitrary code execution or system shutdown.

Schneider Electric has released an updated advisory regarding these vulnerabilities. This advisory is a follow-up to a previous advisory titled ICSA-18-107-02 Schneider Electric Triconex Tricon (Update A) that was published in 2018. The updated advisory warns that HatMan malware specifically targets these vulnerabilities and could be used to exploit the system.

We analyzed the similarities between the CWE-119 vulnerability and the techniques that attackers could use to exploit it and gain access to sensitive areas of the system (Table 6.4). By employing our approach and analyzing data scraped from the MITRE ATT&CK ICS Techniques, we identified similarities between CWE-119 and five different techniques: T0820, T0834, T0857, T0874, and T0890.

Two of the identified techniques, T0820 and T0874, involve exploiting software vulnerabilities to overwrite buffers with malicious code and execute it with elevated privileges, closely related to CWE-119. Another technique, T0834, involves bypassing security controls and accessing sensitive areas of a system by directly interacting with the native OS API. Although the T0857 technique is not directly related to CWE-119, it can exploit other vulnerabilities, such as outdated or unpatched firmware. Lastly, the T0890 technique also targets software vulnerabilities to elevate privileges and gain access to systems, making it closely related to CWE-119. Our results suggest that attackers can use these techniques to exploit CWE-119 and gain access to ICS systems.

However, there was a discrepancy between our results' similarities and the data scraped from the MITRE website. Our approach was not able to identify similarities between the CWE-119 and the T0821, T0843, T0845, T0846, T0849, T0853, T0858, T0868, T0869, T0871, T0872, T0880, T0885 techniques. However, we have classified the techniques based on their potential relevance to CWE-119.

Our findings show that some of the techniques could potentially be used to exploit a vulnerability in a controller, while others may not be directly related. The techniques

that could potentially be related to CWE-119 include T0821, T0843, T0845, T0849, T0858, T0868, T0869, T0871, T0872, and T0880. These techniques involve modifying the behavior of a controller, exploiting vulnerabilities, or compromising safety systems. Therefore, these techniques should be considered when designing security measures to prevent CWE-119 attacks. On the other hand, some techniques may not be directly related to CWE-119, such as T0846 and T0853. These techniques involve gathering information or executing code, but they do not directly exploit a vulnerability in a controller. Nevertheless, they could potentially be used in combination with other techniques to exploit a weakness.

It is important to note that our classification of these techniques as potentially related or not related to CWE-119 does not imply their guaranteed success in exploiting a vulnerability. The effectiveness of these techniques will depend on the specific implementation of the controller and the security measures in place. Our method could not find similarities between CWE-119 and several techniques (T0821, T0843, T0845, T0846, T0849, T0853, T0858, T0868, T0869, T0871, T0872, T0880, T0885), possibly due to limited information in their descriptions. The data on the MITRE website was manually curated by experts, which could explain the discrepancies in our findings. Despite these limitations, our approach found similarities in five techniques (T0820, T0834, T0857, T0874, and T0890) that could potentially be used in an attack with CWE-119. This provides valuable insights into attack techniques and could aid future research in developing better mitigation strategies and security measures.

6.3.2.4 Frequent CWE Itemsets

In this work, we aimed to investigate the exploitation of different CWEs using one or more MITRE ATT&CK ICS Techniques. To achieve this, we employed the FPgrowth algorithm to compute frequent CWE itemsets, as described in Section 6.2.3. The results of our analysis are presented in Table 6.5a and Table 6.5b. Table 6.5a displays the support values and itemsets for two different CWEs (CWE-22 and CWE-284), that were identified and used in our case studies. The support values in Table 6.5a represent the proportion

Table 6.5: Frequent Itemsets where CWE-22 and CWE-284 appears in itemsets of length 2

a) Frequent CWE itemsets			
CWE-22		**CWE-284**	
Support	**CWEs**	**Support**	**CWEs**
0.46	(22)	0.47	(284)
0.46	**(22,23[e])**	0.34	(284,22)
0.41	(24,22)	0.46	**(284,23[e])**
0.36	(26,22)	0.34	(24,284)
0.39	(27,22)	0.36	(27,284)
0.38	(28,22)	0.32	(284,28)
0.34	(32,22)	0.39	**(284,36[f])**
0.33	(33,22)	0.42	(284,39)
0.33	(34,22)	0.30	(284,62)
0.33	(35,22)	0.37	(73,284)
0.42	(36f,22)	0.36	(284,78)
0.45	(22,39)	0.36	(96,284)
0.32	(40,22)	0.30	(114,284)
0.37	(61,22)	0.41	(116,284)
0.45	(73,22)	0.33	(284,117)
0.32	(74,22)	0.30	(121,284)
0.36	(77,22)	0.43	(250,284)
0.37	(78,22)	0.34	**(306[d],284)**
0.41	(96,22)	0.37	(384,284)
0.36	(114,22)	0.30	(441,284)
0.43	(116,22)	0.41	(284,494)
0.41	(117,22)	0.43	(497,284)
0.33	(121,22)	0.34	**(732[c],284)**
0.32	(138,22)	0.32	(284,807)
0.34	**(184[g],22)**	0.39	(284,828)
0.41	(250,22)	0.33	(841,284)
0.34	(284,22)	0.37	(940,284)
0.34	**(306[d],22)**	0.30	(284,941)
0.38	(384,22)	0.37	(1249,284)
0.34	(444,22)	0.34	(1329,284)
0.45	(22,494)		
0.46	(497,22)		
0.41	**(732[c],22)**		
0.32	(22,807)		
0.33	(822,22)		
0.41	(828,22)		
0.33	**(862[a],22)**		
0.34	**(22,863[b])**		
0.36	(940,22)		
0.36	(1249,22)		
0.30	(1259,22)		
0.30	(1268,22)		
0.30	(1293,22)		
0.30	(221,302)		
0.32	(1329,22)		

b) Manual Verification		
CWE-284	**CWE-22**	**CWE-119**
[a]CWE-862	CWE-1003	CWE-1003
[b]CWE-863	CWE-1305	CWE-1305
[c]CWE-732	CWE-1340	CWE-1340
[d]CWE-306	CWE-209	CWE-700
CWE-286	CWE-79	CWE-252
CWE-923	**[e]CWE-23**	CWE-476
CWE-1008	CWE-434	CWE-126
CWE-1340	**[d]CWE-306**	CWE-839
[e]CWE-23	CWE-20	CWE-129
	[f]CWE-36	CWE-125
	[g]CWE-184	CWE-122
	CWE-182	CWE-1339
	CWE-602	CWE-190
	CWE-180	CWE-193
	CWE-174	CWE-131
	CWE-59	
	CWE-243	
	CWE-95	
	CWE-621	
	CWE-790	

of transactions that contain the corresponding CWEs. For instance, the first row of Table 6.5 shows that the itemset containing only CWE-22 appeared in 46% of transactions. Our approach identified possible attacks that exploit the weaknesses of CWE-22, CWE-284, and CWE-119 using the technique T0857 (see Table 6.4).

Table 6.5b shows the CWEs related to the three CWEs we used in our case studies. We verified this relation manually by checking the official page of the CWE-22[12], CWE-119[13], and CWE-284[14]. Additionally, Table 6.5a and Table 6.5b are connected using colors and superscripts (a,...,g).

The CWEs that appeared together in the transactions are highlighted in green and include CWE-22 and CWE-284. Other CWEs were identified using our approach and by manually checking the official CWEs page. The colors in the table represent these different CWEs. For instance, CWE-306 appeared in the transactions using our NLP approach, and we verified that it is related to CWE-284 and CWE-22 on the official CWEs page. This validation step allowed us to confirm that our approach effectively identified the main and related weaknesses.

The superscripts used to annotate CWEs in Table 4a indicate that they were discovered using our frequent CWE itemset approach. Each superscript in Table 6.5a corresponds to an entry in Table 6.5b. For example, in the CWE itemset (22,23) of Table 6.5a, CWE-23 was identified as an exploit that can occur with CWE-22. Table 6.5b confirms that CWE-23 is related to CWE-284 and CWE-22. Therefore, we can validate that our approach identifies new weaknesses by using CWE itemsets. This analysis helps identify which CWEs can be exploited with specific techniques or sets of techniques. For example, Table 6.4 shows that CWE-22 can be exploited with techniques such as T0803, T0804, T0849, and T0857. Further analysis of our results in Table 6.5a reveals that our approach identified frequent occurrences of CWE-22 with other CWEs, including CWE-23, CWE-184, and CWE-306. Notably, some of these CWEs were derived from our frequent CWE itemset

[12]https://cwe.mitre.org/data/definitions/22.html
[13]https://cwe.mitre.org/data/definitions/119.html
[14]https://cwe.mitre.org/data/definitions/284.html

mining. Therefore, we can infer that CWE-23, CWE-184, and CWE-306 can also be exploited with the techniques T0803, T0804, T0849, and T0857.

These results confirm the effectiveness of our approach in detecting possible attack scenarios involving multiple CWEs and techniques.

It is worth noting that CWE-119 does not appear in Table 6.5 because the minimum support used in the FPgrowth algorithm was set to 30%, and CWE-119 does not appear in many transactions. Nevertheless, our results demonstrate the utility of frequent itemset mining in identifying the co-occurrence of different attacks using CWEs and revealing potential attack scenarios on ICS.

6.4 Conclusions and Future Work

The three case studies presented in this work demonstrate the importance of identifying and addressing vulnerabilities in industrial control systems (ICS). Our proposed approach, which computes the similarity between MITRE ATT&CK ICS Techniques and CWEs using natural language processing techniques, provides valuable insights into the exploitation of several CWEs using ICS Techniques, allowing for more effective threat analysis and the development of more robust security measures.

The use of frequent itemset mining in the case studies further illustrates the approach's efficacy in identifying potential attack scenarios involving multiple CWEs. While the methodology has limitations and requires expert knowledge, it holds significant promise for enhancing security weakness identification in ICS and improving critical infrastructure's protection against cyber attacks.

Future work in this area could benefit from expanding the dataset used for our analysis, which would provide a more comprehensive understanding of the relationships between MITRE ATT&CK ICS Techniques and CWEs. A larger dataset could lead to more accurate and relevant insights into the effectiveness of natural language processing techniques in identifying similarities between attack techniques and weaknesses.

In addition, further exploration of other natural language processing techniques beyond SBERT, BERT, and TFIDF could be valuable. We could evaluate the effectiveness of other techniques, such as GPT-4, to determine their potential utility in this context.

Another area for future work is incorporating expert knowledge and domain-specific information to enhance the accuracy and relevance of the results. Expert knowledge could help refine the selection of relevant techniques and vulnerabilities, as well as provide additional context for the analysis. This approach could also involve leveraging additional sources of data, such as threat intelligence feeds or data from the ICS community, to enrich our analysis and provide further insights.

Chapter 7

Concluding Remarks

The primary objective of this dissertation was to examine effective methods of analyzing and predicting emerging threats. To achieve this, we combined theories and models from both computer science and epidemiology. This integration enabled us to identify and predict potential threats.

An important contribution of this research is the development of a framework for detecting infectious disease outbreaks. This approach uses Google Trends data and disease-specific symptoms to offer a practical and effective way to detect outbreaks of several infectious diseases. In a subsequent project, we found that integrating human mobility data with the traditional SEIR epidemiological model has significant implications for simulating disease control strategies.

Another important aspect of our project is the development of an approach to detect malware activity using our AVERTs datasets and public search data. The results are promising, indicating that it is possible to identify unusual malware activity by analyzing public search data.

These findings have significant practical implications. Policymakers, public health officials, and security experts can use the proposed frameworks and corresponding studies to make informed decisions and take appropriate measures in response to emerging threats. Furthermore, the proposed frameworks and corresponding studies lay the foundation for future research in these areas.

References

[1] D. N. Obinna, "Solidarity across borders: A pragmatic need for global covid-19 vaccine equity," *The International Journal of Health Planning and Management*, vol. 37, no. 1, pp. 21–29, 2022.

[2] M. Lehto, "Cyber-attacks against critical infrastructure," in *Cyber Security: Critical Infrastructure Protection*. Springer, 2022, pp. 3–42.

[3] L. M. Stolerman, L. Clemente, C. Poirier, K. V. Parag, A. Majumder, S. Masyn, B. Resch, and M. Santillana, "Using digital traces to build prospective and real-time county-level early warning systems to anticipate covid-19 outbreaks in the united states," *Science Advances*, vol. 9, no. 3, p. eabq0199, 2023.

[4] D. M. Morens, G. K. Folkers, and A. S. Fauci, "The challenge of emerging and re-emerging infectious diseases," *Nature*, vol. 430, no. 6996, pp. 242–249, jul 2004.

[5] A. R. Lab, *The color of coronavirus: Covid-19 deaths by race and ethnicity in the us*. APM Research Lab, 2020.

[6] J. Wildman, "Covid-19 and income inequality in oecd countries," *The European Journal of Health Economics*, vol. 22, no. 3, pp. 455–462, 2021.

[7] C. Pelat, P.-Y. Bolle, B. J. Cowling, F. Carrat, A. Flahault, S. Ansart, and A.-J. Valleron, "Online detection and quantification of epidemics." *BMC Medical Informatics and Decision Making*, vol. 7, p. 29, oct 2007.

[8] S. C. Wieland, J. S. Brownstein, B. Berger, and K. D. Mandl, "Automated real time constant-specificity surveillance for disease outbreaks." *BMC Medical Informatics and Decision Making*, vol. 7, p. 15, jun 2007.

[9] M. Frisn and E. Andersson, "Semiparametric surveillance of monotonic changes," *Sequential analysis*, vol. 28, no. 4, pp. 434–454, oct 2009.

[10] B. Miller, H. Kassenborg, W. Dunsmuir, J. Griffith, M. Hadidi, J. D. Nordin, and R. Danila, "Syndromic surveillance for influenzalike illness in ambulatory care network." *Emerging Infectious Diseases*, vol. 10, no. 10, pp. 1806–1811, oct 2004.

[11] S. H. Heisterkamp, A. L. M. Dekkers, and J. C. M. Heijne, "Automated detection of infectious disease outbreaks: hierarchical time series models." *Statistics in Medicine*, vol. 25, no. 24, pp. 4179–4196, dec 2006.

[12] C. Marshall, N. Best, A. Bottle, and P. Aylin, "Statistical issues in the prospective monitoring of health outcomes across multiple units," *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, vol. 167, no. 3, pp. 541–559, aug 2004.

[13] M. Kulldorff, F. Mostashari, L. Duczmal, W. Katherine Yih, K. Kleinman, and R. Platt, "Multivariate scan statistics for disease surveillance." *Statistics in Medicine*, vol. 26, no. 8, pp. 1824–1833, apr 2007.

[14] A. K. Johnson and S. D. Mehta, "A comparison of internet search trends and sexually transmitted infection rates using google trends." *Sexually Transmitted Diseases*, vol. 41, no. 1, pp. 61–63, jan 2014.

[15] S. Cho, C. H. Sohn, M. W. Jo, S.-Y. Shin, J. H. Lee, S. M. Ryoo, W. Y. Kim, and D.-W. Seo, "Correlation between national influenza surveillance data and google trends in south korea." *Plos One*, vol. 8, no. 12, p. e81422, dec 2013.

[16] V. Gianfredi, N. Bragazzi, M. Mahamid, B. Bisharat, N. Mahroum, H. Amital, and M. Adawi, "Monitoring public interest toward pertussis outbreaks: an extensive google trends–based analysis," *Public Health*, vol. 165, pp. 9–15, 2018.

[17] A. Walker, C. Hopkins, and P. Surda, "Use of google trends to investigate loss-of-smell–related searches during the covid-19 outbreak," in *International forum of allergy & rhinology*, vol. 10, no. 7. Wiley Online Library, 2020, pp. 839–847.

[18] S. B. Omer, P. Malani, and C. Del Rio, "The covid-19 pandemic in the us: a clinical update," *Jama*, vol. 323, no. 18, pp. 1767–1768, 2020.

[19] G. Onder, G. Rezza, and S. Brusaferro, "Case-fatality rate and characteristics of patients dying in relation to covid-19 in italy," *Jama*, vol. 323, no. 18, pp. 1775–1776, 2020.

[20] R. E. Jordan, P. Adab, and K. Cheng, "Covid-19: risk factors for severe disease and death," 2020.

[21] A. Atkeson, "What will be the economic impact of covid-19 in the us? rough estimates of disease scenarios," National Bureau of Economic Research, Tech. Rep., 2020.

[22] D. W. Berger, K. F. Herkenhoff, and S. Mongey, "An seir infectious disease model with testing and conditional quarantine," National Bureau of Economic Research, Tech. Rep., 2020.

[23] Y. Zhou, R. Xu, D. Hu, Y. Yue, Q. Li, and J. Xia, "Effects of human mobility restrictions on the spread of covid-19 in shenzhen, china: a modelling study using mobile phone data," *The Lancet Digital Health*, vol. 2, no. 8, pp. e417–e424, 2020.

[24] H. Wang, A. Ghosh, J. Ding, R. Sarkar, and J. Gao, "Heterogeneous interventions reduce the spread of covid-19 in simulations on real mobility data," *Scientific reports*, vol. 11, no. 1, pp. 1–12, 2021.

[25] D. Parker and O. Pianykh, "Mobility-guided estimation of covid-19 transmission rates," *American journal of epidemiology*, vol. 190, no. 6, pp. 1081–1087, 2021.

[26] X. Hou, S. Gao, Q. Li, Y. Kang, N. Chen, K. Chen, J. Rao, J. S. Ellenberg, and J. A. Patz, "Intracounty modeling of covid-19 infection with human mobility: Assessing spatial heterogeneity with business traffic, age, and race," *Proceedings of the National Academy of Sciences*, vol. 118, no. 24, 2021.

[27] S. Chang, E. Pierson, P. W. Koh, J. Gerardin, B. Redbird, D. Grusky, and J. Leskovec, "Mobility network models of covid-19 explain inequities and inform reopening," *Nature*, vol. 589, no. 7840, pp. 82–87, 2021.

[28] S. Wu, X. Fan, L. Chen, M. Cheng, and C. Wang, "Predicting the spread of covid-19 in china with human mobility data," in *Proceedings of the 29th International Conference on Advances in Geographic Information Systems*, 2021, pp. 240–243.

[29] Z. Zheng, Z. Xie, Y. Qin, K. Wang, Y. Yu, and P. Fu, "Exploring the influence of human mobility factors and spread prediction on early covid-19 in the usa," *BMC Public Health*, vol. 21, no. 1, pp. 1–13, 2021.

[30] M. A. Bhouri, F. S. Costabal, H. Wang, K. Linka, M. Peirlinck, E. Kuhl, and P. Perdikaris, "Covid-19 dynamics across the us: A deep learning study of human mobility and social behavior," *Computer Methods in Applied Mechanics and Engineering*, vol. 382, p. 113891, 2021.

[31] H. Holm, "Signature based intrusion detection for zero-day attacks:(not) a closed chapter?" in *2014 47th Hawaii international conference on system sciences*. IEEE, 2014, pp. 4895–4904.

[32] W. Liu, P. Ren, K. Liu, and H.-x. Duan, "Behavior-based malware analysis and detection," in *2011 first international workshop on complexity and data mining*. IEEE, 2011, pp. 39–42.

[33] I. Burguera, U. Zurutuza, and S. Nadjm-Tehrani, "Crowdroid: behavior-based malware detection system for android," in *Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices*, 2011, pp. 15–26.

[34] M. Sun, X. Li, J. C. Lui, R. T. Ma, and Z. Liang, "Monet: a user-oriented behavior-based malware variants detection system for android," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 5, pp. 1103–1112, 2016.

[35] H. S. Galal, Y. B. Mahdy, and M. A. Atiea, "Behavior-based features model for malware detection," *Journal of Computer Virology and Hacking Techniques*, vol. 12, no. 2, pp. 59–67, 2016.

[36] R. Mosli, R. Li, B. Yuan, and Y. Pan, "A behavior-based approach for malware detection," in *IFIP International Conference on Digital Forensics*. Springer, 2017, pp. 187–201.

[37] S. Ravi, N. Balakrishnan, and B. Venkatesh, "Behavior-based malware analysis using profile hidden markov models," in *2013 International Conference on Security and Cryptography (SECRYPT)*. IEEE, 2013, pp. 1–12.

[38] N. Peiravian, *Data Mining Heuristic–Based Malware Detection for Android Applications*. Florida Atlantic University, 2013.

[39] S. Treadwell and M. Zhou, "A heuristic approach for detection of obfuscated malware," in *2009 IEEE International Conference on Intelligence and Security Informatics*. IEEE, 2009, pp. 291–299.

[40] M. Zakeri, F. Faraji Daneshgar, and M. Abbaspour, "A static heuristic approach to detecting malware targets," *Security and Communication Networks*, vol. 8, no. 17, pp. 3015–3027, 2015.

[41] P. Khodamoradi, M. Fazlali, F. Mardukhi, and M. Nosrati, "Heuristic metamorphic malware detection based on statistics of assembly instructions using classification

algorithms," in *2015 18th CSI International Symposium on Computer Architecture and Digital Systems (CADS)*. IEEE, 2015, pp. 1–6.

[42] F. Song and T. Touili, "Efficient malware detection using model-checking," in *International Symposium on Formal Methods*. Springer, 2012, pp. 418–433.

[43] P. Battista, F. Mercaldo, V. Nardone, A. Santone, and C. A. Visaggio, "Identification of android malware families with model checking." in *ICISSP*, 2016, pp. 542–547.

[44] F. Song and T. Touili, "Pommade: pushdown model-checking for malware detection," in *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering*, 2013, pp. 607–610.

[45] ——, "Model-checking for android malware detection," in *Asian Symposium on Programming Languages and Systems*. Springer, 2014, pp. 216–235.

[46] F. Martinelli, F. Mercaldo, V. Nardone, A. Santone, and G. Vaglini, "Model checking to detect the hummingbad malware," in *International Symposium on Intelligent and Distributed Computing*. Springer, 2019, pp. 485–494.

[47] C. Bernardeschi, F. Mercaldo, V. Nardone, and A. Santone, "Exploiting model checking for mobile botnet detection," *Procedia Computer Science*, vol. 159, pp. 963–972, 2019.

[48] M. Sewak, S. K. Sahay, and H. Rathore, "An investigation of a deep learning based malware detection system," in *Proceedings of the 13th International Conference on Availability, Reliability and Security*, 2018, pp. 1–5.

[49] D. Zhu, H. Jin, Y. Yang, D. Wu, and W. Chen, "Deepflow: Deep learning-based malware detection by mining android application for abnormal usage of sensitive data," in *2017 IEEE symposium on computers and communications (ISCC)*. IEEE, 2017, pp. 438–443.

[50] J. Saxe and K. Berlin, "Deep neural network based malware detection using two dimensional binary program features," in *2015 10th international conference on malicious and unwanted software (MALWARE)*. IEEE, 2015, pp. 11–20.

[51] M. K. Alzaylaee, S. Y. Yerima, and S. Sezer, "Dl-droid: Deep learning based android malware detection using real devices," *Computers & Security*, vol. 89, p. 101663, 2020.

[52] N. Lu, D. Li, W. Shi, P. Vijayakumar, F. Piccialli, and V. Chang, "An efficient combined deep neural network based malware detection framework in 5g environment," *Computer Networks*, vol. 189, p. 107932, 2021.

[53] V. Sihag, M. Vardhan, P. Singh, G. Choudhary, and S. Son, "De-lady: Deep learning based android malware detection using dynamic features," *Journal of Internet Services and Information Security (JISIS)*, vol. 11, no. 2, pp. 34–45, 2021.

[54] K. Vahedi and K. Afhamisisi, "Cloud based malware detection through behavioral entropy," in *2021 IEEE International Conference on Big Data (Big Data)*. IEEE, 2021, pp. 6046–6048.

[55] D.-W. Kim, Y.-T. Jo, and J.-M. Kim, "Cloud-based malware qr code detection system," *Journal of the Korea Institute of Information and Communication Engineering*, vol. 25, no. 9, pp. 1227–1233, 2021.

[56] L. Zheng and J. Zhang, "A new malware detection method based on vmcadr in cloud environments," *Security and Communication Networks*, vol. 2022, 2022.

[57] J. M. Arif, M. F. Ab Razak, S. R. T. Mat, S. Awang, N. S. N. Ismail, and A. Firdaus, "Android mobile malware detection using fuzzy ahp," *Journal of Information Security and Applications*, vol. 61, p. 102929, 2021.

[58] R. Casolare, C. De Dominicis, G. Iadarola, F. Martinelli, F. Mercaldo, and A. Santone, "Dynamic mobile malware detection through system call-based image repre-

sentation." *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl.*, vol. 12, no. 1, pp. 44–63, 2021.

[59] S. Wei, Z. Zhang, S. Li, and P. Jiang, "Calibrating network traffic with one-dimensional convolutional neural network with autoencoder and independent recurrent neural network for mobile malware detection," *Security and Communication Networks*, vol. 2021, 2021.

[60] C. Wang, Z. Zhao, F. Wang, and Q. Li, "A novel malware detection and family classification scheme for iot based on deam and densenet," *Security and Communication Networks*, vol. 2021, 2021.

[61] J. Abawajy, A. Darem, and A. A. Alhashmi, "Feature subset selection for malware detection in smart iot platforms," *Sensors*, vol. 21, no. 4, p. 1374, 2021.

[62] K. Priyadarsini, N. Mishra, M. Prasad, V. Gupta, and S. Khasim, "Detection of malware on the internet of things and its applications depends on long short-term memory network," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–12, 2021.

[63] J. Jeon, B. Jeong, S. Baek, and Y.-S. Jeong, "Hybrid malware detection based on bi-lstm and spp-net for smart iot," *IEEE Transactions on Industrial Informatics*, 2021.

[64] R. Li, Q. Li, J. Zhou, and Y. Jiang, "Adriot: An edge-assisted anomaly detection framework against iot-based network attacks," *IEEE Internet of Things Journal*, 2021.

[65] B. Edwards, T. Moore, G. Stelle, S. Hofmeyr, and S. Forrest, "Beyond the blacklist: modeling malware spread and the effect of interventions," in *Proceedings of the 2012 New Security Paradigms Workshop*, 2012, pp. 53–66.

[66] G. Wang, J. W. Stokes, C. Herley, and D. Felstead, "Detecting malicious landing pages in malware distribution networks," in *2013 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2013, pp. 1–11.

[67] F. Howard and O. Komili, "Poisoned search results: How hackers have automated search engine poisoning attacks to distribute malware," *Sophos Technical Papers*, pp. 1–15, 2010.

[68] T. Yagi, N. Tanimoto, T. Hariu, and M. Itoh, "Investigation and analysis of malware on websites," in *2010 12th IEEE International Symposium on Web Systems Evolution (WSE)*. IEEE, 2010, pp. 73–81.

[69] ——, "Enhanced attack collection scheme on high-interaction web honeypots," in *The IEEE symposium on Computers and Communications*. IEEE, 2010, pp. 81–86.

[70] L. Invernizzi, P. M. Comparetti, S. Benvenuti, C. Kruegel, M. Cova, and G. Vigna, "Evilseed: A guided approach to finding malicious web pages," in *2012 IEEE symposium on Security and Privacy*. IEEE, 2012, pp. 428–442.

[71] G. Stringhini, C. Kruegel, and G. Vigna, "Shady paths: Leveraging surfing crowds to detect malicious web pages," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, 2013, pp. 133–144.

[72] C. Bansal, P. Deligiannis, C. Maddila, and N. Rao, "Studying ransomware attacks using web search logs," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020, pp. 1517–1520.

[73] A. Kuppa, L. Aouad, and N.-A. Le-Khac, "Linking cve's to mitre att&ck techniques," in *Proceedings of the 16th International Conference on Availability, Reliability and Security*, 2021, pp. 1–12.

[74] K. Kanakogi, H. Washizaki, Y. Fukazawa, S. Ogata, T. Okubo, T. Kato, H. Kanuka, A. Hazeyama, and N. Yoshioka, "Tracing cve vulnerability information to capec attack

patterns using natural language processing techniques," *Information*, vol. 12, no. 8, p. 298, 2021.

[75] O. Grigorescu, A. Nica, M. Dascalu, and R. Rughinis, "Cve2att&ck: Bert-based mapping of cves to mitre att&ck techniques," *Algorithms*, vol. 15, no. 9, p. 314, 2022.

[76] F. ¨O. S¨onmez, "Classifying common vulnerabilities and exposures database using text mining and graph theoretical analysis," *Machine Intelligence and Big Data Analytics for Cybersecurity Applications*, pp. 313–338, 2021.

[77] E. Hemberg, J. Kelly, M. Shlapentokh-Rothman, B. Reinstadler, K. Xu, N. Rutar, and U.-M. O'Reilly, "Linking threat tactics, techniques, and patterns with defensive weaknesses, vulnerabilities and affected platform configurations for cyber hunting," *arXiv preprint arXiv:2010.00533*, 2020.

[78] B. Ampel, S. Samtani, S. Ullman, and H. Chen, "Linking common vulnerabilities and exposures to the mitre att&ck framework: A self-distillation approach," *arXiv preprint arXiv:2108.01696*, 2021.

[79] X. Zhou, J. Menche, A.-L. Barabsi, and A. Sharma, "Human symptoms-disease network." *Nature Communications*, vol. 5, p. 4212, jun 2014.

[80] CDC, "Data finder - health, united states - products," 2020. [Online]. Available: https://www.cdc.gov/nchs/hus/contents2018.htm#Table_010

[81] PubMed, "Medline database of references and abstracts on life sciences and biomedical topics." 2021. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/

[82] G. E. A. P. A. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, p. 20, jun 2004.

[83] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[84] S. Dendamrongvit and M. Kubat, "Undersampling approach for imbalanced training sets and induction from multi-label text-categorization domains," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2009, pp. 40–52.

[85] L. Zhang, K. Tang, and X. Yao, "Log-normality and skewness of estimated state/action values in reinforcement learning." *Advances in Neural Info. Proc. Sys.*, p. 1804, 2017.

[86] I. Guyon *et al.*, "A scaling law for the validation-set training-set size ratio," *AT&T Bell Laboratories*, vol. 1, no. 11, 1997.

[87] G. M. Foody, A. Mathur, C. Sanchez-Hernandez, and D. S. Boyd, "Training set size requirements for the classification of a specific class," *Remote Sensing of Environment*, vol. 104, no. 1, pp. 1–14, 2006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0034425706001234

[88] V. Popovici, W. Chen, B. D. Gallas, C. Hatzis, W. Shi, F. W. Samuelson, Y. Nikolsky, M. Tsyganova, A. Ishkin, T. Nikolskaya *et al.*, "Effect of training-sample size and classification difficulty on the accuracy of genomic predictors," *Breast Cancer Research*, vol. 12, no. 1, pp. 1–13, 2010.

[89] C. Molnar, *Interpretable machine learning. A guide for making black box models explainable*. Leanpub, 2020.

[90] E. Choi, M. T. Bahadori, J. A. Kulas, A. Schuetz, W. F. Stewart, and J. Sun, "Retain: an interpretable predictive model for healthcare using reverse time attention mechanism," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 2016, pp. 3512–3520.

[91] T. Mori, "Superposed naive bayes for accurate and interpretable prediction," in *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2015, pp. 1228–1233.

[92] J. Zurada, "Could decision trees improve the classification accuracy and interpretability of loan granting decisions?" in *2010 43rd Hawaii International Conference on System Sciences*. IEEE, 2010, pp. 1–9.

[93] D. L. Weller, T. M. Love, and M. Wiedmann, "Interpretability versus accuracy: A comparison of machine learning models built using different algorithms, performance measures, and features to predict e. coli levels in agricultural water," *Frontiers in artificial intelligence*, vol. 4, 2021.

[94] T. A. Plate, "Accuracy versus interpretability in flexible modeling: Implementing a tradeoff using gaussian process models," *Behaviormetrika*, vol. 26, no. 1, pp. 29–50, 1999.

[95] Y. Deng, X. Liu, C. Xin, and W. Jia, "An interpretable classifier with linear discriminant analysis based on afs theory," in *2019 Chinese Control Conference (CCC)*. IEEE, 2019, pp. 7583–7588.

[96] C. Ferri, J. Hernndez-Orallo, and R. Modroiu, "An experimental comparison of performance measures for classification," *Pattern recognition letters*, vol. 30, no. 1, pp. 27–38, jan 2009.

[97] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[98] L. D. Manzanares-Meza and O. Medina-Contreras, "Sars-cov-2 and influenza: a comparative overview and treatment implications," *Boletín médico del Hospital Infantil de México*, vol. 77, no. 5, pp. 262–273, 2020.

[99] C. Arnold, "Could covid delirium bring on dementia?" *Nature*, vol. 588, no. 7836, p. 22–24, 2020.

[100] F. S. Mirfazeli, A. Sarabi-Jamab, A. Jahanbakhshi, A. Kordi, P. Javadnia, S. V. Shariat, O. Aloosh, M. Almasi-Dooghaee, and S. H. R. Faiz, "Neuropsychiatric manifestations of covid-19 can be clustered in three distinct symptom categories," *Scientific Reports*, vol. 10, no. 1, pp. 1–9, 2020.

[101] P. L. Andrews, W. Cai, J. A. Rudd, and G. J. Sanger, "Covid-19, nausea, and vomiting," *Journal of Gastroenterology and Hepatology*, vol. 36, no. 3, pp. 646–656, 2021.

[102] M. P. Caan, C. T. Lim, and M. Howard, "A case of catatonia in a man with covid-19," *Psychosomatics*, vol. 61, no. 5, p. 556, 2020.

[103] F. Nikbakht, A. Mohammadkhanizadeh, and E. Mohammadi, "How does the covid-19 cause seizure and epilepsy in patients? the potential mechanisms," *Multiple sclerosis and related disorders*, p. 102535, 2020.

[104] B. Joob and V. Wiwanitkit, "Arthralgia as an initial presentation of covid-19: observation," *Rheumatology international*, vol. 40, no. 5, pp. 823–823, 2020.

[105] M. Zbair, A. Qaffou, F. Cherkaoui, and K. Hilal, "Bayesian inference of a discrete fractional seird model," in *Intl. Conf. on Partial Diff. Equation and App., Modeling and Simulation*. Springer, 2023, pp. 138–146.

[106] T. Rapolu, B. Nutakki, T. Sobha Rani, and S. Durga Bhavani, "A time-dependent seird model for forecasting the transmission dynamics in infectious diseases: COVID-19 a case study," in *Proc. of International. Conf. on Data Science and Applications*. Springer, 2022, pp. 423–437.

[107] H. H. Weiss, "The sir model and the foundations of public health," *Materials matematics*, pp. 0001–17, 2013.

[108] H. W. Hethcote, "The mathematics of infectious diseases," *SIAM review*, vol. 42, no. 4, pp. 599–653, 2000.

[109] "Report of the WHO-China Joint Mission on Coronavirus Disease 2019 (COVID-19)." [Online]. Available: https://www.who.int/publications-detail-redirect/report-of-the-who-china-joint-mission-on-coronavirus-disease-2019-(covid-19)

[110] A. Celmiņš, "Least squares model fitting to fuzzy vector data," *Fuzzy sets and systems*, vol. 22, no. 3, pp. 245–269, 1987.

[111] J. Lessler, A. S. Azman, H. S. McKay, and S. M. Moore, "What is a hotspot anyway?" *The American journal of tropical medicine and hygiene*, vol. 96, no. 6, p. 1270, 2017.

[112] A. M. Oster, G. J. Kang, A. E. Cha, V. Beresovsky, C. E. Rose, G. Rainisch, L. Porter, E. E. Valverde, E. B. Peterson, A. K. Driscoll *et al.*, "Trends in number and distribution of COVID-19 hotspot counties—united states, march 8–july 15, 2020," *Morbidity and Mortality Weekly Report*, vol. 69, no. 33, p. 1127, 2020.

[113] W. Dhouib, J. Maatoug, I. Ayouni, N. Zammit, R. Ghammem, S. B. Fredj, and H. Ghannem, "The incubation period during the pandemic of COVID-19: a systematic review and meta-analysis," *Systematic Reviews*, vol. 10, no. 1, pp. 1–14, 2021.

[114] J. A. Gold, L. M. Rossen, F. B. Ahmad, P. Sutton, Z. Li, P. P. Salvatore, J. P. Coyle, J. DeCuir, B. N. Baack, T. M. Durant *et al.*, "Race, ethnicity, and age trends in persons who died from covid-19—united states, may–august 2020," *Morbidity and Mortality Weekly Report*, vol. 69, no. 42, p. 1517, 2020.

[115] C. I. A. Oronce, C. A. Scannell, I. Kawachi, and Y. Tsugawa, "Association between state-level income inequality and covid-19 cases and mortality in the usa," *Journal of general internal medicine*, vol. 35, no. 9, pp. 2791–2793, 2020.

[116] ASPE, "2021 poverty guidelines," 2021. [Online]. Available: https://shorturl.at/kzYZ9

[117] C. Pipeline, "Media statement update: Colonial pipeline system disruption," 2021.

[118] I. Kilovaty, "Cybersecuring the pipeline," *Houston Law Review*, vol. 60, 2023.

[119] S. Chockalingam and C. Maathuis, "An ontology for effective security incident management," in *International Conference on Cyber Warfare and Security*, vol. 17, no. 1, 2022, pp. 26–35.

[120] J. Franklin, C. Wergin, and H. Booth, "Cvss implementation guidance," *National Institute of Standards and Technology, NISTIR-7946*, 2014.

[121] A. G. Masid, J. B. Higuera, J.-R. B. Higuera, and J. A. S. Montalvo, "Application of the sama methodology to ryuk malware," *Journal of Computer Virology and Hacking Techniques*, pp. 1–34, 2022.

[122] S. Alzahrani, Y. Xiao, and W. Sun, "An analysis of conti ransomware leaked source codes," *IEEE Access*, 2022.

[123] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," *arXiv preprint arXiv:1908.10084*, 2019.

[124] M. Steinbach, L. Ertöz, and V. Kumar, "The challenges of clustering high dimensional data," in *New directions in statistical physics*. Springer, 2004, pp. 273–309.

[125] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.

[126] R. J. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," in *Pacific-Asia conference on knowledge discovery and data mining*. Springer, 2013, pp. 160–172.

[127] M. Grootendorst, "Bertopic: Neural topic modeling with a class-based tf-idf procedure," *arXiv preprint arXiv:2203.05794*, 2022.

[128] G. E. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM SIGKDD explorations newsletter*, vol. 6, no. 1, pp. 20–29, 2004.

[129] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in neural information processing systems*, vol. 30, 2017.

[130] C. Molnar, "Interpretable machine learning: A guide for making black-box models explainable. 2019," *URL https://christophm. github. io/interpretableml-book/. Cited on*, p. 33, 2022.

[131] J. Chigada and R. Madzinga, "Cyberattacks and threats during covid-19: A systematic literature review," *South African Journal of Information Management*, vol. 23, no. 1, pp. 1–11, 2021.

[132] N. Tariq, "Impact of cyberattacks on financial institutions," *Journal of Internet Banking and Commerce*, vol. 23, no. 2, pp. 1–11, 2018.

[133] S. Perera, X. Jin, A. Maurushat, and D.-G. J. Opoku, "Factors affecting reputational damage to organizations due to cyberattacks," in *Informatics*, vol. 9, no. 1. MDPI, 2022, p. 28.

[134] Z. Yun-tao, G. Ling, and W. Yong-cheng, "An improved tf-idf approach for text classification," *Journal of Zhejiang University-Science A*, vol. 6, pp. 49–55, 2005.

[135] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[136] I. Villanueva-Miranda and M. Akbar, "Detecting malware activity using public search data," in *2022 IEEE International Conference on Big Data (Big Data)*. IEEE, 2022, pp. 2997–3006.

[137] C. Borgelt, "An implementation of the fp-growth algorithm," in *Proceedings of the 1st international workshop on open source data mining: frequent pattern mining implementations*, 2005, pp. 1–5.

Curriculum Vitae

Ismael Villanueva-Miranda is a passionate and dedicated researcher with a strong desire to make a positive impact on the world. His expertise lies in developing tools to prevent, detect, and mitigate emerging threats, with a particular focus on epidemics. Ismael has presented his research at conferences on Big Data, including IEEE Big Data and ICDM. Throughout his career, Ismael has improved his skills in retrieving and manipulating large and complex datasets, as well as implementing artificial intelligence approaches for various domains. He has demonstrated his expertise in developing frameworks for disease outbreaks, malware activity detection, and depression detection. His passion for learning and expanding his knowledge base has led him to work with climate models, meteorological models, ocean models, and health datasets, further broadening his expertise. In addition to his research, Ismael has a background in teaching and mentoring students. His experience as a university lecturer has enabled him to refine his skills in delivering engaging lectures and guiding students in their academic pursuits. He is committed to providing his students with a high-quality education and inspiring them to become future leaders in their fields.

Contact Information: ismaelvillanueva@gmail.com