The IntelliTweet: Unveiling Malicious Activities in Tweets through a Multifaceted Feature Analysis

by

Eric Dzeha

Thesis submitted to the University of Ottawa
in partial fulfillment of the requirements for the degree of
Master of Computer Science, Concentration in Applied Artificial Intelligence
in
Electrical and Computer Engineering
University of Ottawa
Ottawa, Ontario, Canada

Declaration of Authorship

I hereby certify that this thesis is entirely my own original work except where otherwise indicated. I am aware of the University of Ottawa regulations concerning plagiarism, including those regarding consequent disciplinary actions. Any use of the works of any other author, in any form, is properly acknowledged at their point of use.

Abstract

Social media platforms have seamlessly integrated into our daily communication, facilitating information sharing, connections and engagement for both individuals and businesses. Among these platforms, Twitter has emerged as one of the popular platforms for its rapid information dissemination and real-time interaction capabilities. However, the widespread adoption of Twitter has also attracted malicious activities such as phishing, spam, and scams, which take advantage of the platform's extensive reach to spread rapidly.

In this Thesis, we introduce "The IntelliTweet," a machine learning system designed to enhance real-time detection and classification of malicious tweets on Twitter. IntelliTweet employs a multifaceted feature approach by integrating content analysis, user profile attributes, sentiment analysis, URL analysis and term frequency-inverse document frequency (TF-IDF) techniques. This holistic methodology considers the contextual nature of tweets, as well as content-based features and user behavior patterns, to accurately distinguish malicious tweets from legitimate ones, including user-reported tweets that raise awareness about threats.

Our work began with an in-depth review of existing literature and the landscape of Twitter-centric threats, identifying shortcomings in current detection methodologies ranging from traditional assessments to machine learning classifiers. We subsequently delved into the conceptualization of IntelliTweet as well as the feature design integrating tweet metadata, user profiles, and linguistic nuances within tweets.

As part of this work, we created a database by collecting tweets in real-time directly from the Twitter stream. This database contains a mix of malicious tweets, legitimate tweets, and user-reported tweets, allowing us to analyze the interactions between user-generated warnings and responses to malicious activities on Twitter.

We conducted experiments including model selection, feature importance analysis, grid search optimization, hyperparameter tuning, and t-tests, providing a thorough evaluation of IntelliTweet's performance. Validating both binary and multiclass system configurations, IntelliTweet's precision-centric approach demonstrates reliability and significant improvements with results achieving 98.80% precision, 98.15% F1-score, and a low 0.07 false positive rate on real-world Twitter data. By prioritizing false alarm reduction and maximizing the global precision, IntelliTweet minimizes the mislabeling of legitimate users to account for the real-world implications of user misclassification such as account suspension.

IntelliTweet represents a positive step towards Twitter security and positive user experiences, contributing to cybersecurity evolution and providing valuable insights for mitigating emerging threats on the platform. We also suggest in this Thesis some future research directions, including integrating user-centric features and cross-linguistic detection, and considering real-world applications and ethical considerations. It also proposes developing a global, multilingual defense mechanism against digital threats.

Acknowledgements

I would like to express my sincere gratitude to my supervisor Professor Guy-Vincent Jourdan and the entire Cyber Range Research Team for their invaluable guidance, support, and encouragement throughout this research journey. Their expertise, insights, and unwavering commitment have been instrumental in shaping the development of this research.

I extend my heartfelt appreciation to the Foundations & Practice of Security (FPS) Symposium chairs and organization who enabled me to publish my research paper to contribute to the fight of cybercrime, enriching our research endeavors and fostering collaboration within the academic community.

I am deeply grateful to my family, Campus Rush, Transforming Life Center and all my loved ones, for their constant prayers, unwavering encouragement, understanding, and patience during the challenges and triumphs of this research endeavor. Their boundless love and support have been a constant source of strength and inspiration.

I would like to thank the examiners, Professor David Knox and Professor Paria Shirani, for taking time out of their busy schedules to evaluate and examine my thesis during the defense. Your insights, comments, and corrections were invaluable.

Finally, I acknowledge the Vector Institute and the Faculty of Engineering, University of Ottawa who have supported this research through the scholarships and research opportunities they provided.

Table of Contents

List of Tables xi

List of Figures xiii

Abbreviations xvi

1 Introduction 1

1.1 Problem Description:Twitter Threat Landscape 2

1.2 Motivation 4

1.3 Objectives 5

1.4 Publication 6

1.5 Organization 6

2 Related Work 8

2.1 Threat Models in Malicious Tweet Detection 9

2.2 Traditional Approaches to Detecting Phishing, Spam, and Scams 10

2.2.1 List-based . . . 10
2.2.2 Rule-based . . . 11
2.2.3 Machine Learning-based . . . 13
2.3 Uncovering Malicious Tweets on Twitter . . . 16
2.3.1 Common Approaches and Features Employed . . . 17
2.3.1.1 Ground Truth Datasets . . . 18
2.3.1.2 URL Redirection Effect . . . 19
2.3.1.3 Sentimental Analysis Application . . . 20
2.3.1.4 Graph-based . . . 22
2.3.1.4.1 Clustering: . . . 22
2.4 Malicious Tweet Reports on Twitter . . . 23
2.4.1 Effectiveness of Phishing Reports . . . 23
2.4.1.1 Clues in Tweets . . . 26
2.5 Current Gaps in Malicious Tweet Detection . . . 27
2.6 Conclusion . . . 28

3 The IntelliTweet **30**

3.1 Overview . . . 30
3.2 Data Collection and Characterization . . . 31
3.2.1 Crawling Twitter . . . 31
3.2.2 Ground Truth Formalization . . . 32

3.3 Feature Engineering . . . 35
3.3.1 Twitter Related Features . . . 35
3.3.2 Text classification features . . . 35
3.3.3 URL features . . . 37
3.4 System Flow of IntelliTweet . . . 37
3.5 The Binary and Multi System Setups . . . 40
3.6 Summary . . . 41

4 IntelliTweet Designs and Evaluation Results 42

4.1 Evaluation Criteria . . . 42
4.2 Feature Importance . . . 44
4.2.1 Rationale for choosing Mutual Information . . . 45
4.3 Binary System . . . 46
4.3.1 Model selection and evaluation . . . 46
4.3.1.1 Default Parameters . . . 47
4.3.1.2 Hyperparameters in Machine Learning . . . 48
4.3.1.3 Rationale for Choosing Grid Search . . . 49
4.3.1.4 Parameter Tuning . . . 49
4.3.2 IntelliTweet and TextAnalyst Evaluation: . . . 51
4.3.3 IntelliTweet compared to a Baseline approach . . . 53
4.4 Multi System Experiments and Setups . . . 56

4.4.1 Model Selection for Multi-Class Classification 57
4.4.1.1 Default Parameters Model Performance 57
4.4.1.2 Optimizing Model Parameters 58
4.4.2 TextAnalyst in MultiClass System 60
4.5 T-Test Experiment for Precision Values 62
4.5.1 Experimental Setup 64
4.5.2 Interpreting the p-test Results: 65
4.6 Summary 66

5 Previous Experiments: Paving the Path to IntelliTweet 69

5.1 Cascade System with Majority Voting 69
5.1.1 Cascade Operation 71
5.2 Online Learning Approach 73
5.2.1 Integrating Online Learning in IntelliTweet 74
5.2.2 How is the evolution by use of an online model addressed 76
5.2.2.1 A. By Incorporating External Verified Labels 76
5.2.2.2 B. By Leveraging Report Predictions for Online Learning . 77
5.2.2.3 C. Data Augmentation Using Reports 81
5.3 Summary and Key Takeaways 86

6 Conclusions **90**

6.1 Research Core Contributions 91

6.2 Some Research Ideas, Recommendations, and Opportunities 92

6.3 Future Work 93

6.4 IntelliTweet Summary 94

References **95**

APPENDICES **105**

A All Additional Images, Tables and plots **106**

A.1 Confusion Matrices for Binary System IntelliTweet and TextAnalyst Evaluation 106

A.2 Multi System Confusion Matrices for TextAnalyst Evaluation 109

List of Tables

3.1 Summary of Twitter Related Features 36

3.2 URL Features 37

4.1 Average Metrics for Models after five Repetitions of Stratified 5-fold Cross Validation (CV) 47

4.2 Average Metrics for Optimized Models after five Repetitions of Stratified 5-fold CV 50

4.3 Average of 5 folds per iteration of IntelliTweet without TextAnalyst 51

4.4 Average of 5 folds per iteration of IntelliTweet with TextAnalyst 51

4.5 Mean of confusion matrices (IntelliTweet TextAnalyst parameter ON) . . . 52

4.6 Mean of confusion matrices (IntelliTweet TextAnalyst parameter OFF) . . 52

4.7 Mean of confusion matrices of Kamel's approach 56

4.8 Mean of confusion matrices with emphasis on Non-Malicious Predictions . 56

4.9 Average Metrics for Models with default parameters after five Repetitions of Stratified 5-fold CV 58

4.10 Average Metrics for Optimized Models after five Repetitions of Stratified 5-fold CV 59

4.11 Average of 5 folds per iteration of IntelliTweet with TextAnalyst 60

4.12 Average of 5 folds per iteration of IntelliTweet without TextAnalyst 61

4.13 Mean of confusion matrices (IntelliTweet multisystem TextAnalyst parameter OFF) 61

4.14 Mean of confusion matrices (IntelliTweet multisystem TextAnalyst parameter ON) 61

4.15 Precision Values for Different Configurations 64

4.16 Paired Differences, Means, Standard Deviations, and p-values 65

5.1 Metric results from 5 fold cross validation for Initial Data, "Is This Tweet Malicious or Not?" 82

5.2 Metric results from 5 fold CV for Augmented Data "Is This Tweet Malicious or Not?" 83

List of Figures

3.1 Data Collection Process Diagram 33

3.2 IntelliTweet System Flow 39

4.1 Striking the right balance between metrics 43

4.2 Feature Importance by Mutual Information 46

4.3 Box Plot for performance evaluation 55

4.4 Multi System Box Plot for performance evaluation 63

4.5 Box plot for precision values of IntelliTweet configurations 67

5.1 Cascade Design 72

5.2 Online Learning Architecture Design 75

5.3 Sequential Training 77

5.4 Use case scenarios and live tests 79

5.5 Tweet extraction API 80

5.6 Test case 1 80

5.7 Test case 2 81

5.8 Predictions on "Is This Tweet Malicious or Not?" 83

5.9 Report Phase Predictions "Is This Tweet a Report or Not?" 84

5.10 "Is This Tweet Malicious or Not?" 85

5.11 Metrics showing the performance of systems before and after augmentation 88

A.1 Confusion Matrix Images for TextAnallyst On and Off 107

(a) Iteration 1 - TextAnallyst On 107

(b) Iteration 1 - TextAnallyst Off 107

(c) Iteration 2 - TextAnallyst On 107

(d) Iteration 2 - TextAnallyst Off 107

A.1 Confusion Matrix Images for TextAnallyst On and Off 108

(e) Iteration 3 - TextAnallyst On 108

(f) Iteration 3 - TextAnallyst Off 108

(g) Iteration 4 - TextAnallyst On 108

(h) Iteration 4 - TextAnallyst Off 108

(i) Iteration 5 - TextAnallyst On 108

(j) Iteration 5 - TextAnallyst Off 108

A.2 Confusion Matrix Images for TextAnallyst On and Off 109

(a) Iteration 1 - TextAnallyst On 109

(b) Iteration 1 - TextAnallyst Off 109

(c) Iteration 2 - TextAnallyst On 109

(d) Iteration 2 - TextAnallyst Off 109

A.2 Confusion Matrix Images for TextAnallyst On and Off 110

(e) Iteration 3 - TextAnallyst On 110

(f) Iteration 3 - TextAnallyst Off 110

(g) Iteration 4 - TextAnallyst On 110

(h) Iteration 4 - TextAnallyst Off 110

(i) Iteration 5 - TextAnallyst On 110

(j) Iteration 5 - TextAnallyst Off 110

Abbreviations

API Application Programming Interface 31, 32

APWG Anti-Phishing Working Group 1

BERT Bidirectional Encoder Representations from Transformers 14, 15

BiLSTM Bidirectional LSTM 21

CNN Convolutional Neural Networks 21

CV Cross Validation xi, xii, 47, 50, 58, 59, 64, 81, 83, 106

DL Deep Learning 15, 20, 21

DNS Domain Name System 25

FPR False Positive Rate 42, 43, 47, 50, 58, 59

kNN k-Nearest Neighbor 14, 19

LSTM Long Short-Term Memory 21

MI Mutual Information 44, 45

ML Machine Learning 11, 13, 14, 16–18, 20–22, 25, 28

OCR Optical Character Recognition 34

p-test paired t-test 63, 64

PCA Principal Component Analysis 44

RFE Recursive Feature Elimination 44

RNN Recurrent Neural Networks 21

SGD Stochastic Gradient Descent 21

SMO Sequential Minimal Optimization 13

SVM Support Vector Machine 13, 14, 17, 19, 21, 22, 47, 96

TA TextAnalyst 63

TF-IDF Term Frequency Inverse Document Frequency 12, 35, 36, 39, 45

UCI University of California, Irvine 17, 28

URL Uniform Resource Locator 5, 10–12, 14, 17–20, 22, 23, 28, 32, 37, 38, 54, 76, 78, 81

Chapter 1

Introduction

In the online world, the combination of phishing, spam, and scams poses a constant danger that affects both individuals and organizations. These attacks have become a pervasive threat, manifesting through various channels such as emails, text messages, and social media platforms.

Phishing for instance, are attacks designed to trick users into downloading malware, sharing sensitive information or personal data such as Social Security and credit card numbers, bank account numbers, login credentials, passwords, credit card numbers, or taking other actions that expose themselves or their organizations to cybercrime [30]. It may involve creating fake websites that mimic real ones to trick unsuspecting users [47]. Recent statistics reveal an increase in phishing attacks. In the second quarter of 2023, the Anti-Phishing Working Group (APWG) recorded 1,286,208 phishing attacks, representing the third-highest quarterly total in its documented history [25]. Globally, the year 2022 witnessed a record-breaking 4.7 million reported cases, reflecting a substantial 150% surge since 2019.

Scams, on the other hand, are deceptive schemes designed to defraud individuals or organizations through false promises, misleading information, or fake offers. These schemes exploit human emotions and vulnerabilities, often playing on desires for financial gain, security, or social acceptance. Scams have experienced a notable surge, with consumers losing an estimated $56 billion to online scams and fraud in 2021. However, it is important to note that the actual figure could potentially be even higher. [22].

Additionally, spam, which are characterized as unsolicited, irrelevant or unwanted messages is another concern. It floods inboxes, diminishing user experience, and may harbor malicious intent, containing links to phishing websites, malware downloads, or fraudulent schemes. Over 200 billion spam messages are sent daily, constituting nearly 85% of global email traffic [2, 34].

As individuals and businesses move their activities online, the combination of phishing, scams, and spam presents serious threats to financial, informational, and infrastructural security. Their widespread presence on digital platforms highlights the need for new detection methods to combat rapidly evolving threats in real time. In this work, a Malicious Tweet is defined as tweets that encompasses all of spam, scam and phishing attempts which poses serious threats to the user's experience on the platform.

1.1 Problem Description:Twitter Threat Landscape

Social media platforms, including Twitter, have become breeding grounds for malicious activities because they have a far-reaching effect globally, operate in real-time, and have seamless integrations across websites. With over 330 million monthly active users generating over 500 million new tweets daily [3, 5], Twitter stands as a prime target for threat

actors, who exploit its open ecosystem to propagate misinformation, perpetrate fraud, and engage in cybercrime [57].

Several factors contribute to Twitter's susceptibility to attacks. Firstly, Twitter's emphasis on brevity encourages users to condense information into short and brief messages [56]. However, this brevity may lead to the omission of vital context and source verification, making it easier for false claims or harmful links to appear genuine. Until October 2018, Twitter had a 140-character limit for tweets, which was then increased to 280 characters [36]. Additionally, Twitter's "retweet" feature enables the swift propagation of tweets across the platform, resulting in large amounts of false information being replicated. Moreover, the prevalence of shortened URLs on Twitter provides an ideal avenue for phishing content to proliferate. Threat actors exploit these shortened links to lure unsuspecting users into clicking on malicious URLs [13, 39, 43, 52]. This leads users to intermediary servers set up by attackers and potentially exposing them to harmful web pages [17].

Twitter has taken steps to address malicious activities [55]. They automatically scan URLs to flag potential threats and offer users a reporting system for suspicious URLs [42]. Although these efforts are being pursued, these malicious activities remain a persistent setback to the users of Twitter. This may be as a result of the fact that scammers and phishers are constantly evolving their tactics to avoid detection [39]. Consequently, detecting and preventing such malicious activities remains a critical challenge for both platform owners and users.

In response to these challenges, independent researchers have deployed various techniques to proactively identify threats in malicious tweet detection. These methods range from rule-based and list-based to classifier-based systems. Rule-based approaches use predefined rules or patterns and rely on conditional statements to flag prohibited content,

while blocklists and allowlists maintain repositories of known malicious and benign domains for content vetting. Classifier-based models leverage machine learning algorithms to statistically distinguish malicious from benign tweets. While these detection mechanisms have proven useful to an extent, the issue of malicious tweets persists.

Some other researchers [45, 50, 54] investigate phishing and other malicious activity through reports that have been shared on Twitter. They identify and analyze phishing attacks and raise more awareness through these reports, which have minimal interaction from other users on Twitter, especially from the targeted domains and organizations.

In this work, "a Reporter" is defined as a Twitter user who identifies potentially harmful or inappropriate content and takes action by posting about the content they have identified to notify other users or victims. Likewise, a "Report" refers to a tweet that contain information about an identified malicious activity, aimed at raising awareness among users.

1.2 Motivation

The motivation for this research arises from several key observations.

Firstly, it can be observed that the studies that investigated phishing and other malicious activity through reports shared on Twitter often receive minimal interaction from users, especially from targeted domains and organizations.

Secondly, while researchers propose using advanced machine learning techniques to improve the distinction between legitimate and malicious tweets, the introduction of reports in modern literature may interfere with existing predictions. This could be attributed to the presence of malicious URLs within reported tweets, which in turn could pose challenges for accurate prediction.

Furthermore, there may be concerns regarding the relevance and effectiveness of features employed in previous studies. Relying solely on Uniform Resource Locator (URL) detection features or user account information may limit the overall scope of features that can be utilized.

Beyond these initial observations, we were motivated to perform some text analysis, including sentimental analysis to capture the emotional tone of tweets, adding a layer of nuance to the classification process.

1.3 Objectives

In this study, our motivation suggests that relying solely on malicious URLs in a tweet may not be sufficient for accurate malicious tweet activity classification. The presence of these URLs do not always indicate malicious intent; they could be shared innocently or for awareness purposes. Our aim is to introduce a system, which we call "IntelliTweet", a machine learning-based system designed for the precise identification of malicious tweets. IntelliTweet employs a multifaceted approach by combining URL features, user profile attributes, sentiment analysis, and TFIDF-based content analysis. Our goal is to consider both the content and context of tweets, along with user behavior, to effectively distinguish between malicious and legitimate tweets, including user reports. To achieve this, we outline the following objectives:

1. **Data Collection:** Gather live tweets to construct a dataset containing malicious, legitimate tweets, and user reports.

2. **Feature Enhancement:** Introduce additional features to create a holistic feature engineering methodology that considers the nuances of tweet context and content.

The goal is to combine multiple features including text classification, Twitter user features, metadata and tweet text features to enhance model training and robustness.

3. **Model Selection:** Conduct experiments with different models to identify the most suitable one for our dataset to ensure optimal performance.

4. **Text Analysis Evaluation:** Assess the effectiveness of our contextual features, such as sentiment analysis and Term Frequencies, when integrated with conventional features.

1.4 Publication

We have published a paper out of this research:

- [21] Eric Edem Dzeha and Guy-Vincent Jourdan. "Intellitweet: A multifaceted feature approach to detect malicious tweets". In Mohamed Mosbah, Florence Sèdes, Nadia Tawbi, Toufik Ahmed, Nora Boulahia-Cuppens, and Joaquin Garcia-Alfaro, editors, Foundations and Practice of Security, pages 157-173, Cham, 2024. Springer Nature Switzerland.

- Our code and datasets have been made publicly accessible and can be found within our GitHub repository [20].

1.5 Organization

The remainder of this thesis is structured as follows: In Chapter 2 we review pertinent literature discussing methods and approaches for detecting malicious activities on Twitter.

Chapter 3 details our methodology, explaining our approach, the collection of malicious, reported, and legitimate tweets, and the analysis of relevant features; Chapter 4 presents the key experiments and discusses the results; Chapter 5 presents the initial experiments conducted and discusses the results; and in Chapter 6, we summarize our findings, draw conclusions, highlight limitations, and offer recommendations.

Chapter 2

Related Work

This chapter provides an overview of the research on detecting malicious tweets, covering various forms of misinformation, scams, phishing attempts, and spam campaigns. The discussion begins by examining different attack vectors and threat actors, including Twitter users who engage in phishing, scamming, and spamming activities, as well as Twitter bots. Additionally, to establish context, we review various methodologies, from traditional approaches to emerging techniques, that are generally used to identify malicious activities.

Also, the chapter emphasizes the importance of detecting malicious tweets on Twitter's platform and highlights recent advancements in the field. We provide insights into the strengths and weaknesses of diverse approaches and identify potential avenues for improvement. Additionally, we highlight research gaps that require attention.

2.1 Threat Models in Malicious Tweet Detection

Casanove et al [15] underscore the significance of threat models in the domain of malicious tweet detection, emphasizing that threats are defined by the combination of malicious actors (the attacker) and a means to avoid the event detection (the attack vector). This review recognizes the diverse range of malicious actors on Twitter, encompassing automated accounts (including traditional spambots, social spambots, content polluters, and fake followers) and human users (primarily trolls) [33]. Social bots, in particular, have a significant presence on Twitter especially because they create tweets faster than humans. With so many Twitter users around, hackers take advantage of these speedy bots to circulate malicious URLs [60]. Research [38] further reveals that many malicious tweets originate from ordinary Twitter users with malicious intent. Leveraging the platform's extensive reach and diverse user base, these individuals engage in activities ranging from financial gain to personal vendettas.

Casanove et al [15] also highlights that malicious activities on Twitter encompass a range of attack types, including phishing attacks, spam campaigns, and scams. Each threat represents a distinct way adversaries can exploit the Twitter platform for malicious activities.

With this foundation established, we now explore methodologies to identify broader malicious online activities.

2.2 Traditional Approaches to Detecting Phishing, Spam, and Scams

2.2.1 List-based

Historically, list-based techniques have been a fundamental weapon against phishing, spam, and scams. Organizations like the Anti-Phishing Working Group (APWG) have played a pivotal role in this battle. [1] They collect and maintain lists of known phishing URLs, which serve as references for authentication. Services like PhishTank [2] and Google Safe Browsing [3] leverage these lists by providing access to databases for swift lookups. These lists may either contain "Blocklists" featuring suspicious URLs and IP addresses) [48] or "Allowlists" that include presumed safe websites, effectively categorizing all others as potentially blocklisted [8].

In the work of Gupta et al [27], a strategy centered on allowlisting was introduced, wherein various attributes associated with legitimate websites, such as URL, IP address, and Login User Interface, were documented. If a user accesses a website that does not correspond to any entry in this list, it is flagged as potentially malicious. Also, Hong et al [28] suggested a blocklist-oriented approach in their work, which involves dissecting the URL of a suspicious webpage into distinct components and comparing them against a roster of known phishing sites. The compilation of suspicious sites is sourced from diverse outlets, including spam traps and databases of phishing emails.

List-based methods offer the advantages of cost-effectiveness and speed. However, the

[1] The APWG collects, analyzes, and exchanges lists of verified credential collection sites, like those used in phishing. If you have received a suspicious or obviously malicious email you can forward those to APWG for analysis. https://education.apwg.org/report-cybercrime/ Accessed 21 December, 2023

[2] PhishTank | Join the fight against phishing. (n.d.). https://phishtank.org/

[3] Google Safe Browsing. (n.d.). https://safebrowsing.google.com/

blocklists face the formidable challenge of maintaining global lists due to the relentless emergence of new phishing sites. Keeping these lists up-to-date often lags behind the pace of recent attacks, resulting in missed threats [14]. Contrastingly, maintaining a comprehensive global allowlist is even more challenging, given legitimate websites' vast and ever-expanding landscape. Even minor URL alterations can disrupt accurate matching within these lists [10]. Also, since malicious Twitter users often employ URL shortening services to hide harmful links within tweet text [4], list-based methods can only help to investigate such links after following through the redirection to the final URL.

2.2.2 Rule-based

Rule-based approaches offer an alternative strategy for identifying phishing, spam, and scams. These systems use predefined rules or patterns to flag potentially malicious content [9, 37]. Basnet et al. [9] proposed a novel rule-based phishing detection method inspired by the Snort intrusion detection system. Their approach involved manually crafting rules based on common patterns and heuristics found in phishing sites, with the aim of creating an interpretable and easily updatable system. The authors defined 15 rules, informed by an analysis of phishing sites and existing literature, which examined various aspects such as URLs, page content, links, and form usage. Testing these rules using Decision Tree and Logistic Regression classifiers yielded impressive results, with the decision tree achieving 99% accuracy and the logistic regression reaching 97.88%. Their study highlights the importance of continuous rule and training data updates, offering interpretability and competitive accuracy compared to existing Machine Learning (ML) approaches on the authors' dataset.

Also in the work of Catherine [16], an association rule mining approach is used to

detect phishing websites, focusing on URL analysis as an alternative to downloading webpage content or relying on search engines. The central concept revolves around identifying distinctive features within phishing URLs that can be mined and translated into classification rules. The authors defined ten heuristics, including factors such as the number of slashes, presence of special characters, and keywords, as input features for mining. The dataset comprised phishing URLs from PhishTank and legitimate URLs from five other sources. The Predictive Apriori algorithm was employed for association rule mining on the URL features. The implementation involved PHP for feature extraction and MySQL for dataset storage. The results demonstrated the efficacy of the algorithm in uncovering patterns/rules that distinguish phishing URLs. Sample rules, such as those involving TLS and keywords in the path, were provided, and screenshots illustrated a search and verification interface alerting users to potential phishing sites.

The CANTINA study [61], conducted by Zhang et al., introduced a content-based approach that harnessed Term Frequency Inverse Document Frequency (TF-IDF) and Google search engine queries. Their methodology meticulously crafted a unique lexical signature consisting of the top TF-IDF weighted words extracted from suspect webpages, seamlessly integrating the webpage's domain name into this signature. They queried the Google search to determine whether the top search results corresponded to the page's domain. Their method effectively reduced false positives and improved accuracy by incorporating domain names into the analysis. The results of the evaluation showcase the effectiveness of the CANTINA approach. The basic TF-IDF variant achieved an impressive 94% true positive rate, though this was accompanied by a notable 30% false positive rate. However, introducing the domain name into the analysis significantly mitigated false positives, reducing them to 10%, albeit with a trade-off in accuracy, which dipped to 67%. Remarkably, incorporating the "zero results mean phishing" heuristic was a game-changer, catapulting

the accuracy to a commendable 97% while maintaining low false positives.

While heuristic-driven and very valuable, rule-based methods are not immune to false positives and may require fine-tuning. They have limitations in adapting to the dynamic nature of Twitter-based phishing attacks.

2.2.3 Machine Learning-based

In response to these challenges, researchers have turned to Machine Learning-based methods. Machine Learning-based methods have emerged as powerful tools in the fight against phishing. Korkmaz et al.'s [39] research demonstrated the promise of ML algorithms in creating robust phishing detection systems. Their methodology hinged on utilizing eight distinct machine-learning algorithms with three diverse datasets. The outcomes underscored the remarkable efficiency of ML in enhancing phishing detection, yielding commendable levels of accuracy.

Similarly, in a recent study, Ghosh et al. [24] showcased the current prowess of ML algorithms in spam detection. Their exploration of 13 different classifiers, including Adaptive Booster, Artificial Neural Network, Bootstrap Aggregating, Decision Table, Decision Tree, J48, K-Nearest Neighbor, Linear Regression, Logistic Regression, Naïve Bayes, Random Forest, Sequential Minimal Optimization (SMO), and Support Vector Machine (SVM), revealed exceptional results. Among these classifiers, the Random Forest classifier outperformed others, achieving an accuracy of 99.91% and 99.93% for the Spam Corpus and Spambase datasets, respectively. The Naïve Bayes classifier didn't perform as well as the other 12 classifiers. It showed lower accuracy, with 87.63% for the Spam Corpus dataset and 79.53% for the Spambase dataset.

Jain and Gupta [32] introduced a client-side anti-phishing method enriched by ML.

Their approach involved the identification of nineteen critical website characteristics gleaned from source code and URLs. This method yielded exceptional results, boasting a true positive rate of 99.39% and an overall detection accuracy of 99.09%.

In the work proposed by [6], a phishing detection and prevention approach using a SVM algorithm, specifically focusing on URL analysis. URL features, such as length, presence of suspicious words, and the use of IP addresses instead of domain names, are extracted to train SVM models for automated categorization of URLs into phishing or legitimate. The methodology involved collecting a dataset from PhishTank, and multiple ML models, including SVM, k-Nearest Neighbor (kNN), and logistic regression, were trained and evaluated using 10-fold cross validation. The SVM model achieved the highest accuracy at 96.4%, with precision at 96.5% and recall at 98.0%, outperforming the other models. The high precision indicates a low false positive rate, demonstrating the effectiveness of SVM for automated phishing URL detection based on URL features and substantial improvements over prior literature.

Sunjay et al. [40] delved into the global menace of spam emails, constituting around 50% of all emails, prompting research into ML-based spam detection. A proposed framework, integrating linguistic and URL-based features, demonstrated high accuracy, detecting spam emails at 89.2% and phishing emails at 97.7%.

Similarly, Guo et al. [26] have investigated spam detection utilizing Bidirectional Transformers and ML Classifier Algorithms. They introduced an efficient approach to spam detection that relies on a pretrained bidirectional encoder representation from the transformer (Bidirectional Encoder Representations from Transformers (BERT)) model in conjunction with ML classifier algorithms. The methodology involved feeding email texts into the BERT model to obtain feature representations capturing text semantics and salient features. These BERT-based features were subsequently input into four different classi-

fier algorithms—support vector machines, k-nearest neighbors, random forest, and logistic regression—to classify feature vectors as either ham (legitimate) or spam emails. The reported results indicated that logistic regression demonstrated the best performance among the classifiers, achieving precision, recall, and F1 scores of 97.86%, 97.83%, and 97.84%, respectively, on the Enron-Spam dataset. Similarly, on the second dataset, logistic regression achieved top results with 95.95% precision, 96% recall, and 95.92% F1 score. The consistent high performance of logistic regression across both datasets underscored its effectiveness for spam detection based on BERT feature extraction. The AUC metric and ROC curves further demonstrated the robust classification ability of the proposed approach, thereby validating the utility of BERT for text representation and emphasizing logistic regression's efficacy in identifying spam emails.

Ghanem et al. [23] proposed a novel Deep Learning (DL) architecture called Contextualized Bi-directional Long Short-Term Memory (CBLSTM) to detect spam messages on social media. Their approach focused on overcoming limitations of traditional word embedding models like word2vec and GloVe in handling short, noisy texts. The key aspect was using deep contextualized word representations from ELMo to capture semantics and account for polysemy instead of fixed word vectors. Their methodology employed ELMo embeddings to represent tweets and YouTube comments, which were fed into a Bi-LSTM network with two dense layers and dropouts, optimized for binary classification. For three datasets - Twitter, YouTube comments, and SMS spam - they achieved high accuracy of 97.8%, 95.21%, and 99.30%, respectively. Ablation studies demonstrated the contribution of the Bi-LSTM architecture and ELMo embeddings. Comparative results against standard feature extraction methods and glove/fastText embeddings validated the superior performance of their approach. The character-level capability of ELMo also overcame out-of-vocabulary issues. Overall, their novel architecture effectively leveraged recent ad-

vances in contextualized representations to push the state-of-the-art for social media spam detection.

The interesting highlights from the systematic review of Safi et al. [51] on phishing website detection techniques shed light on the landscape of various approaches. The prevalence of ML techniques, particularly the Random Forest Classifier, is evident, with 57 studies applying them. This survey emphasizes the significance of ML in addressing the sophisticated nature of phishing attacks and underscores the need for continuous exploration of innovative and holistic solutions to combat evolving threats in the digital realm.

Traditional approaches, while once effective, are struggling to keep pace with the evolving tactics of malicious actors. Despite their promise, ML-based methods face limitations, such as the need for continuous model updates and representative datasets for model training. As the digital threat landscape continues to evolve, so too does the vigilance of cybersecurity professionals in safeguarding the digital realm from these multifaceted adversaries.

Following this broad survey of methodologies for identifying malicious activities, we now narrow our attention directly to the Twitter platform itself and explore the progression of research focused specifically on countering the emerging threats of malicious tweets.

2.3 Uncovering Malicious Tweets on Twitter

As social media has exploded in popularity, online harms have migrated alongside users to these platforms. Some researchers rely on Twitter as an important information source to study current threats. In the world of Twitter, much like how we scrutinize website features and attributes to flag potential phishing sites, the task of detecting phishing and

spam takes a parallel route. It entails a systematic assessment of Twitter content and user behavior, with the ultimate goal of identifying potential scams and phishing endeavors.

2.3.1 Common Approaches and Features Employed

Djaballah et al. [19] in 2015 devised a pioneering study with a 3-step approach combining phishing URL blacklist checking, ML-based URL classification, and user account analysis to detect phishing attacks on Twitter. The approach's foundational step involved scrutinizing tweeted URLs against the PhishTank phishing blacklist. URLs that find a match on this blacklist are promptly flagged as potential phishing threats. Surviving URLs underwent ML-based analysis, utilizing 25 selected URL features and three ML classifiers: Logistic Regression, SVM, and Random Forest, which were trained using the ML Phishing University of California, Irvine (UCI) dataset. The last step was to examine Twitter user accounts responsible for these tweets. This multi-pronged approach leveraging blacklist filtering, ML classifiers, and user account analysis achieved 95% accuracy, surpassing previous benchmarks. The inclusion of URL blocklist checking, serving as a preprocessing sentinel, emerges as a pivotal measure for reducing false negatives. Furthermore, the methodology introduces innovative discriminative features, including insights into IP addresses and accounting for Twitter-specific risks like shortened links commonly used to disguise phishing sites. In their user account analysis, the authors introduced classifying Twitter accounts into malicious or legitimate categories, offering insights into the identity of malicious users.

Even though this study takes a thorough approach, using datasets from the UCI focused on general phishing websites, that aren't Twitter-related raises concerns about how applicable the findings are in the real world. It is clear that the methodology's strength

still needs to be tested with actual Twitter data to see how well it holds up in real-world situations.

While phishing detection is critical to Twitter security, spam activity poses significant challenges. Aggarwal et al. in PhishAri [1] highlight Twitter's historically high spam URL clickthrough rate, surpassing that of email spam, underscoring the urgency of the issue. PhishAri addresses this problem in real time by integrating URL features with Twitter-specific attributes, including tweet length, hashtags, mentions, content, and user characteristics like account age, tweet count, and follower-to-followee ratio. These Twitter-specific features complement URL-based ones, and when combined with ML classification techniques, they achieve a 92.52% accuracy in detecting phishing tweets.

Meanwhile, the work of Inuwa-Dutse et al. [31] presented a technique for spam-posting account detection on Twitter. Their approach distinguishes between spam and non-spam social media posts, emphasizing features independent of historical tweets. These features encompass user-related attributes, account characteristics, and engagement patterns. The method demonstrates efficacy, robustness, and real-time deployment potential, enhancing the validity of research data.

2.3.1.1 Ground Truth Datasets

Chen et al.'s [17] extensive research also focuses on understanding and mitigating spam on Twitter. Their approach to collecting data enabled them to gather 600 million public tweets over several months. This extensive dataset served as the foundation for their investigation. To establish ground truth, they leveraged Trend Micro's Web Reputation Service for labeling spam. They tagged 6.5 million tweets as spam, with an additional 6 million tweets serving as non-spam references, creating a valuable resource for researchers. Chen

et al. extracted 12 lightweight features encompassing various aspects of tweet content and user account characteristics, including the number of hashtags, URLs, mentions, follower counts, and more. The authors employed six classification algorithms, including a Random Forest, SVM, Naive Bayes, kNN, Decision Tree, and Bayes Network. The purpose was to assess the performance of these algorithms in identifying spam tweets. Notably, the choice of a diverse set of algorithms adds depth to the evaluation, allowing for insights into the strengths and weaknesses of each approach. They conducted experiments on four sampled datasets, encompassing 5,000 and 50,000 tweets each. This experiment allowed them to explore the impact of class imbalance as well as the effects of varying tweet selection methods, which is often a challenge in spam detection. Both cross-validation and train-test methodology were utilized to evaluate classifier performance using metrics of accuracy, precision, recall, and F1 score.

Their findings revealed that classifiers, on average, achieved F1 scores of over 90% when handling balanced datasets with a 1:1 ratio of spam to non-spam instances compared to imbalanced datasets with a 1:19 ratio. Among the classifiers tested, Random Forest emerged as the standout performer, boasting an impressive 95% accuracy when applied to continuous tweets. In contrast, Naive Bayes exhibited a propensity for false positives, while SVMs struggled when faced with imbalanced data.

2.3.1.2 URL Redirection Effect

Our attention now shifts to the WarningBird system introduced by Lee and Kim [41]. This system offers an innovative approach to detecting suspicious URLs on Twitter by analyzing correlations in URL redirect chains. This method addresses the limitations of prior techniques that rely on landing page content analysis and proves robust against conditional redirections used to evade detection.

The authors collect Twitter data with web links and check each link to see where it leads. Instead of looking at the final pages, which might hide bad stuff, they focus on finding links that are often shared in a row. This helps them find chains of links controlled by the same group with limited resources.

Specifically, they introduce the notion of an "entry point URL" as the most frequent URL within a set of correlated chains. By analyzing features of these entry points and associated chains, like length, frequency, initial URLs, and landing URLs, they can classify suspicious URLs. Additional features based on tweet metadata, like timestamps and account information, further enhance accuracy.

Their findings revealed high accuracy and low false positive rates in distinguishing benign and malicious URLs using Twitter data. The system showed robustness against conditional redirections compared to conventional content-based approaches. Processing performance benchmarks also demonstrate near real-time capabilities.

2.3.1.3 Sentimental Analysis Application

Building upon the valuable insights derived from Chen et al.'s and other spam detection methodologies, we now turn our attention to the research conducted by Rodrigues et al. [49]. Their study offers a comprehensive perspective, encompassing both spam detection and sentiment analysis on Twitter, leveraging a blend of classical ML and DL models.

In their pursuit, Rodrigues et al. embarked on data collection, amassing an assortment of Twitter data and SMS messages to form the foundation of their training datasets. For the specific task of spam classification, they harnessed a dataset comprising 5,572 SMS messages categorized as "ham" (non-spam) or spam. Simultaneously, they employed a more substantial dataset for sentiment analysis, incorporating 31,015 tweets, each labeled

with a positive, negative, or neutral sentiment. The initial phase of their methodology entailed essential data preprocessing, encompassing activities like filtering and tokenization. Moreover, they employed well-established feature extraction techniques, including TF-IDF and Bag of Words models.

Rodrigues et al. conducted a comprehensive exploration in pursuing model optimization, incorporating a spectrum of ML models spanning classical and DL approaches. This extensive experimentation involved Decision Tree, Logistic Regression, Multinomial Naive Bayes, SVM, Random Forest, and Bernoulli Naive Bayes models skillfully applied to the SMS dataset for binary spam classification. Furthermore, they evaluated the performance of models such as Stochastic Gradient Descent (SGD), SVM, Logistic Regression, Random Forest, and Naive Bayes for multiclass sentiment analysis. Taking a step further into the realm of DL, they ventured into the territory of Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), Bidirectional LSTM (BiLSTM), and 1D Convolutional Neural Networks (CNN) architectures.

The outcomes of their study were indeed promising. For instance, the Multinomial Naive Bayes model showcased an accuracy rate of 97.78% in the spam classification context. In contrast, LSTM models achieved a validation accuracy of 73.81% for sentiment analysis. However, amidst these notable achievements, their research is not devoid of limitations. The scope of their comparative analysis, especially concerning the DL models, could be broadened for a more robust evaluation. Additionally, a notable avenue for future research is the exploration of the synergy between spam detection and sentiment analysis. In this area, Rodrigues et al.'s study primarily treats these tasks in isolation without delving into the potential connections between them. Exploring how text sentiment and spam probability interact could yield valuable insights into the nature of malicious tweets, potentially leading to improved detection accuracy. Thus, a promising opportunity exists

to bridge this gap and develop more holistic, integrated approaches in future research.

2.3.1.4 Graph-based

In the context of ML techniques for identifying malicious tweets, it is worth examining how graph-based approaches can contribute to this task. Wang's [59] work sheds light on this aspect as he focused on detecting spam accounts on Twitter. He adopted a unique approach by modeling Twitter as a directed graph that captures follower/friend relationships. In addition to this innovative approach, they introduced novel content and graph-based features derived from Twitter's spam policies. These features included considerations like reputation, duplicate tweets, mentions, links, and hashtags.

To evaluate the effectiveness of their approach, Wang's team collected a dataset comprising 25,000 users and 500,000 tweets. They subjected this dataset to several classifiers, including Naive Bayes, SVM, decision trees, and neural networks. Among the classifiers, Naive Bayes emerged as the top performer, achieving the best F-score at 91.7%. The Bayesian model showcased 89% precision when applied to the entire dataset.

While this research possesses distinct strengths, such as the introduction of innovative graph-based features and a comprehensive comparison of multiple classifiers, it comes with its limitations. Notably, the focus primarily lies on account features, and the research does not delve deeply into the content of tweets. Additionally, the dataset size may be considered somewhat limited, and the study doesn't thoroughly assess the generalizability of its findings over time.

2.3.1.4.1 Clustering: Some other researchers, like Federico Concone et al. [18], used URL inspection and tweet clustering to identify common behaviors among spammers and

legitimate users. Their clustering technique achieves approximately 80% accuracy in labeling accounts. Similarly Jeong et al [35]. proposed a two-phase approach called Phishing Detector for Twitter (PDT), which combines a density-based clustering algorithm, DBSCAN, with the DerTIA algorithm used in detecting attacks on recommender systems.

2.4 Malicious Tweet Reports on Twitter

Phishing detection on Twitter is not limited to URL-based or network features. The public reporting of phishing attacks has garnered attention in the dynamic Twitter landscape. Some research efforts reviewed below focus on educating users about identifying and avoiding phishing scams on Twitter by sharing reports and raising awareness.

2.4.1 Effectiveness of Phishing Reports

In 2021, Saha Roy et al. [50] investigated an approach of harnessing crowdsourced phishing reports on Twitter as a varied source of information on potential threats. The authors explored the dynamics of users sharing phishing attack reports on Twitter and shed light on user behavior and content-sharing patterns.

The authors deployed an innovative data collection methodology that harnessed the Twitter API to curate tweets about phishing attacks. These tweets were identified through specific keywords and indicators, including the presence of obfuscated URLs (“hxxp” and “hxxps”) and the usage of the “#phishing” hashtag. They implemented automated tracking systems to continuously monitor various attributes of each unique phishing URL, such as its status, detections by anti-phishing tools, interactions (including retweets and comments),

and the involvement of organizational accounts. This enabled them to amass a robust dataset for their analysis.

They performed a comparative analysis and revealed that Twitter reports on phishing attacks outperformed established feeds like OpenPhish and PhishTank in several critical aspects. Notably, Twitter reports offered more extensive contextual details, including screenshots and information about the targeted organizations. This section underscores the substantial value of utilizing Twitter as a rich, real-time threat intelligence source. Unfortunately, analysis unveiled a surprising trend of limited engagement, with Only 13.8% of these reports receiving comments. A key finding of the study was the lower rate of false positives in Twitter reports (11%) compared to PhishTank (20%). Also, in their findings, Twitter reports exhibited faster coverage of emerging threats, providing a timelier response to evolving phishing attacks.

The paper's key insight comes from an observation regarding reports from targeted organizations. When these organizations interacted with the reports, it correlated strongly with faster removal of harmful URLs. This finding underscores the influential role of target organizations in mitigating threats. Longitudinal tracking of reported URLs revealed that 31% of them remained active after a week, signifying persistent threats. Moreover, at least 27% of the URLs had fewer anti-phishing detections compared to those from other sources. This implies limitations in the coverage of anti-phishing tools, emphasizing the need for enhanced cybersecurity measures.

In a parallel exploration of Twitter's potential as a source of threat intelligence, Nakano et al. [45] introduced the innovative "Canary in Twitter Mine" system. Their research focused on collecting and analyzing phishing attacks reported by both experts and non-experts on Twitter, providing valuable insights into the value of non-expert reports in discovering new phishing URLs and attack details.

The authors present CrowdCanary, a system with two primary components. First, it collects data by utilizing pre-selected security keywords and automatically extracting co-occurrence keywords to pinpoint pertinent tweets. Employing both image and text analysis, the system extracts candidate URLs/domains, which are then screened for formatting and age. Second, CrowdCanary classifies reports by extracting numerous features from tweets and employing a random forest classifier trained on 20,000 manually labeled tweets to identify phishing attack reports with an accuracy rate of 95%. Over three months of operation, the system amassed a substantial dataset of 19 million tweets, identified 38,000 phishing reports, and extracted 35,000 URLs. Notably, the extracted URLs were cross-referenced with antivirus engines on VirusTotal. Users who shared reports were categorized as experts or non-experts based on account traits.

The results revealed that CrowdCanary achieved a high detection rate for malicious URLs (90%), surpassing other Twitter threat data systems (57-59%). Furthermore, the authors observed that expert accounts shared 15,000 reports, while 9,000 non-experts shared 18,000. Interestingly, non-expert reports contained more dynamic Domain Name System (DNS)/short URLs and included details such as brand names, contributing 31,000 previously unseen malicious URLs. The paper highlights several system strengths, such as its innovative hybrid approach, successful evaluation on a large three-month Twitter dataset, and the insights gained from categorizing and comparing experts and non-experts.

However, the paper also acknowledges some limitations, including its focus on English and Japanese tweets, potential accuracy degradation over time due to evolving tactics, and the exclusion of attacks involving redirects. Additionally, ethical concerns related to large-scale Twitter data collection were raised, and the paper did not delve into user interactions with the reports to understand their impact.

Despite certain limitations, the hybrid ML approach showcased impressive accuracy in

identifying malicious phishing URLs, and the analysis of expert versus non-expert reporting behaviors provided fresh insights that could inform the design of more comprehensive threat data collection systems.

2.4.1.1 Clues in Tweets

Spam activity on Twitter is not limited to tweets alone; it extends to Short Message Service (SMS) spam. Considering the realm of SMS spam, Tang et al. [54] introduced SpamHunter, a framework for detecting and collecting SMS spam messages reported on Twitter. This innovative approach bridged the gap between Twitter and SMS spam.

The SpamHunter framework is constructed upon a robust foundation, encompassing tweet collection, image detection, classification, and text recognition modules. These components work harmoniously to identify tweets that report SMS spam and subsequently extract pertinent messages. The authors conducted their data collection process between 2018 and 2021, focusing on English tweets that featured spam images or relevant hashtags.

The analysis conducted by the research team encompassed a deep dive into the contents of the identified messages, the underlying infrastructure of spammers, and the intricacies of cross-language spam campaigns. Moreover, the study also explored the interactions and behaviors of users who reported SMS spam.

The performance of SpamHunter achieved a precision rate of 95% and a recall rate of 87% in the classification of spam-reporting tweets. Over four years, it amassed a collection of 21,918 SMS spam messages spanning 75 different languages. This dataset now stands as the largest public repository of SMS spam messages. The findings of the study unveiled several significant insights into SMS spam. Among them were the prevalence of fraudulent messages, identifying localized spam trends, and detecting spammers resorting to dynamic

DNS services. Equally significant was the revelation that the majority of users reporting SMS spam did so only once, and their interactions with the targeted companies garnered minimal responses.

Despite its successes, SpamHunter has certain limitations. It primarily focuses on tweets in the English language that contain specific keywords indicative of spam. This scope limitation, while essential for precision, may restrict the coverage of the framework in detecting SMS spam across various languages and regions.

2.5 Current Gaps in Malicious Tweet Detection

While the reviewed literature offers valuable insights into detecting malicious tweets on Twitter, several gaps and challenges persist that must be addressed to stay ahead of malicious actors in the dynamic digital landscape.

One key gap is the susceptibility of existing feature-based machine learning classifiers to social engineering techniques and obfuscation methods, which pose challenges for classifiers in accurately identifying deceptive content. This may necessitate researching into building resilience by revisiting the kind of features used, the depth of the features, and incorporating broader relationship data beyond individual tweets. Another challenge is ensuring real-world applicability by validating against contemporary Twitter data rather than synthetic or outdated samples. Constant updates and maintenance are crucial given the dynamic nature of malicious tactics on the platform. Limited access to annotated datasets from prior work hampers reproducibility and collective progress. Moreover, developing real-time detection systems for timely threat mitigation and reducing both false positives (non-malicious content misidentified as threats) and false negatives (malicious

content missed by the classifier) are essential for enhancing the overall accuracy of detection approaches. Minimizing false positives improves precision, while reducing false negatives improves recall. Addressing these gaps through innovative research represents a significant stride toward robust malicious tweet detection.

2.6 Conclusion

In conclusion, the persistent challenge of combating malicious tweets and evolving phishing attacks demands a proactive and adaptable approach. Despite extensive literature, the continuous evolution of tactics and techniques employed by malicious actors on Twitter raises questions about the relevance and effectiveness of previous features and detection systems. Some prior research efforts, such as those outlined in [1], reported low metric scores, and Sharma et al. [53], records that systems like "PhishAri" are obsolete due to Twitter's evolving authentication requirements. Furthermore, certain works have predominantly focused on ML techniques, analyzing only specific aspects such as URLs [18] or user account information [31] , potentially limiting the overall effectiveness of the detection systems. Another noteworthy approach discussed in [19] combines URL analysis with tweet content and blacklists; however, its reliance on datasets from the UCI raises concerns about its real-world applicability. The UCI Machine Learning phishing dataset [44] is incompatible with our approach due to its lack of live tweet access, essential for our text analysis features. Additionally, existing works in this field [18, 31, 35, 50] do not provide access to their datasets for further evaluation.

Traditional and emerging approaches offer valuable insights, but addressing the identified gaps is a step in the right direction, for staying ahead of malicious actors in this dynamic digital environment. Researchers must consider multilingual detection, adapt to

evolving tactics employed by malicious actors, and develop methods for handling manipulated media within tweets. Additionally, understanding the broader context of tweets, incorporating user-centric detection, and developing real-time response systems are crucial for enhancing Twitter security while addressing ethical considerations surrounding user privacy and consent. Researchers and practitioners must collaborate to develop innovative, robust, and ethical detection methods to safeguard users and the integrity of social media platforms.

Chapter 3

The IntelliTweet

This chapter outlines our strategy to improve the detection and mitigation of malicious tweets on Twitter. We detail the real-time generation of our dataset and provide an overview of the Intellitweet system architectures.

3.1 Overview

In response to the challenges discussed in section 2.5 paragraphs, we introduce IntelliTweet, a comprehensive solution that aims to enhance the detection and mitigation of malicious tweets. To address the issue of real-world dataset applicability, our approach involves the collection of real-time Twitter data directly from the Twitter stream, to build a dataset that can be used by other researchers. This data encompasses not only the content of tweets but also metadata, providing a more comprehensive and up-to-date foundation for our system.

IntelliTweet offers a multifaceted approach to tackling malicious tweets. In this overview,

we delve into the formulation of the problem through both binary and multisystem representations by considering the prediction of reports. These dual representations enables us to consider the system's adaptability and robustness in response to evolving threats. Additionally, we explore the various components that constitute the IntelliTweet system, including data collection and characterization, data engineering and processing, feature engineering, selection, and visualization, and the overall system flow.

3.2 Data Collection and Characterization

3.2.1 Crawling Twitter

For this research, we built our dataset by collecting live tweets. We made use of the Twitter Public Streaming Application Programming Interface (API) [1] which offers samples of public data flowing through Twitter around the world in real-time. We collected over 20k samples of Tweets across the months of April 2022 to November 2022. For each batch of 1k tweets obtained, a cleaning procedure is executed to eliminate stopwords, punctuations, emojis, and to stem the tweets. The API has a limiting factor in terms of the number of queries per hour and the API can get only 1% of all the public tweets flowing through Twitter. However, with the help of its Filter function, we utilized keywords such as phishing, scam, spam, threat, bit.ly, btc, bitcoin, free, click here, avoid, required, hxxp to search for relevant data.

In the related works [18, 19, 35] we observe that while it is possible to use Twitter to send spam and other messages without using URLs, the vast majority of spam and other malicious messages on the platform contain URLs. We concur that among the thousands

[1] Twitter API documentation https://developer.twitter.com/en/docs/twitter-api

of malicious tweets manually inspected, there were only a few tweets without URLs found to be malicious. In [39], the authors reported that malicious actors primarily use embedded URLs to make it easier to direct victims to their external sites to achieve their goals, such as phishing, scams, and malware downloading. As a result, we focus on collecting tweets with embedded URLs.

Figure 3.1 below summarizes our data collection process.

3.2.2 Ground Truth Formalization

Each extracted tweet from the Twitter API contains extensive information including: user ID, username, user biography, user screen name, user URL, user account creation data, tweet text, tweet creation time, tweet's unique ID, the language of the tweet, number of tweets of a user, number of favorites, number of followers, number of mentions, number of following, number of retweets, bounding box of the location (geolocation), and the application that sent the tweet. Some of this information that was relevant to our research has been used to develop a set of features for efficiently classifying the data from Twitter which is discussed in subsection 4.2.

For our ground truth, we took the tweet text and user account into consideration. Malicious users on Twitter usually use URL shortening services to shorten long malicious URLs and embed these URLs into the tweet text [1]. As such, we execute a function that follows all the redirections of URLs in the tweet until we obtain the long URL of the shortened one, and this offers us additional information. We then employ the use of VirusTotal [2] to scan the full URLs and provides a scan report [58]. If the scan reports the URL as malicious, the corresponding tweet is categorized as such. In some special cases at

[2]VirusTotal Home https://www.virustotal.com/gui/home/upload

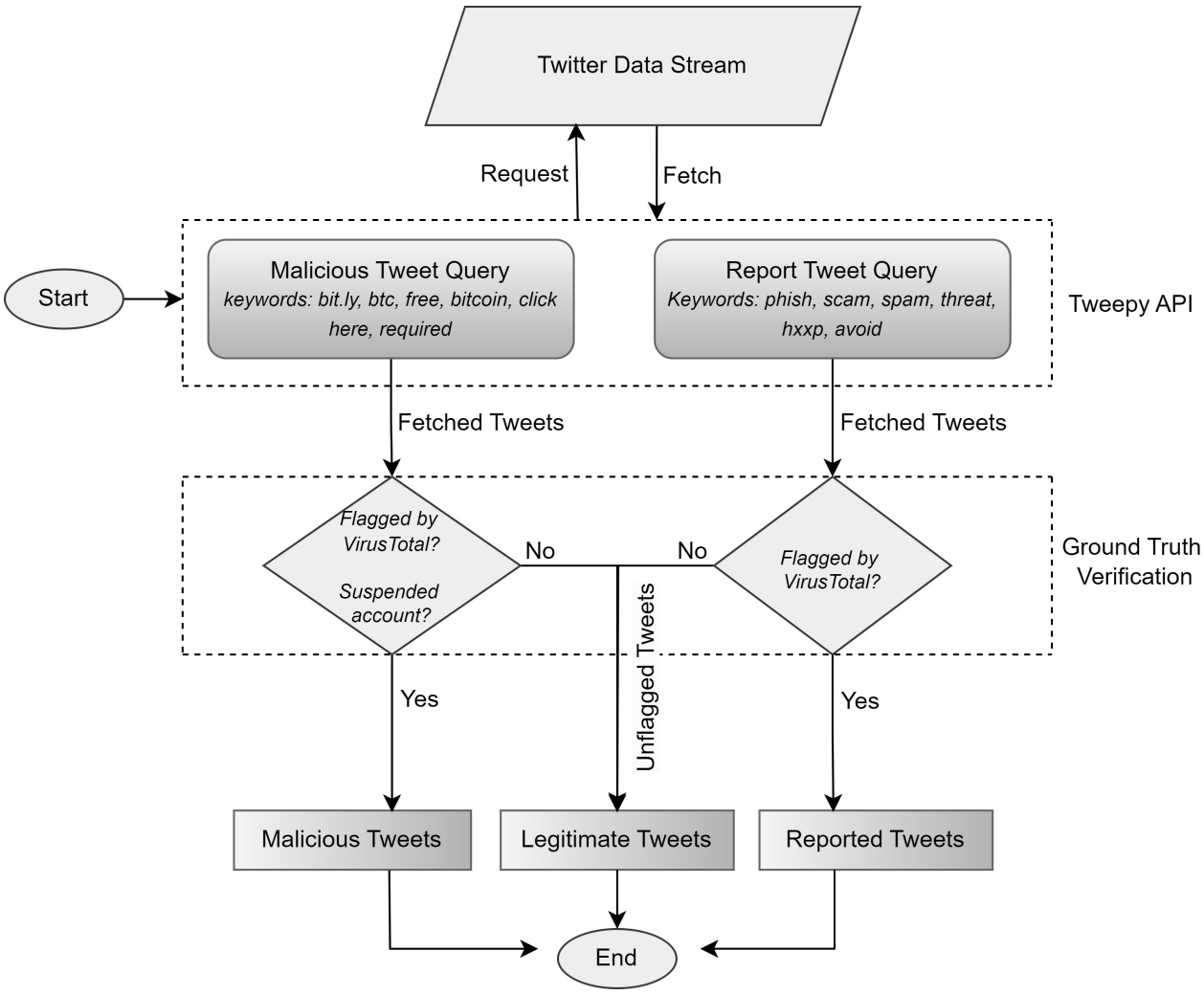


Figure 3.1: Data Collection Process Diagram

the time of labelling, we realized Twitter has suspended some of the accounts we fetched and we have labeled these users as malicious also.

In the case of Reports, tweets are typically geared towards reporting malicious content and as such they usually have embedded in them hashtags like #phishing, #spam, and #scam to create awareness for vulnerable users. Also the work of Sayak Roy [50] addresses that some Reporters try to defang the URLs to prevent users from accidentally visiting the malicious links. For example, they may replace 'http'/'https' with variants of 'hxxp/hxxps' [50] or employ other techniques. We therefore collected and labeled reports by searching for tweets using the keywords phishing, scam, spam, hxxp, avoid, threat. After processing these tweets, we utilized VirusTotal to check if the extracted URLs were flagged as malicious, since reporters reports malicious URLs. In cases where VirusTotal did not flag them, we conducted manual inspections to determine if the tweet warranted labeling as a report. This manual inspection process involved examining a small subset, approximately 2% of the total reported tweets, from the report collection. We also collected some tweets that have linked media attachments. We considered only images and screenshots for media and made use of an Optical Character Recognition (OCR) engine to extract the content of images and combine it with the main tweet before labelling. For Legitimate Tweets, we collect and label them if there is no indication of malicious activity reported, no flags from VirusTotal, and no suspension of user accounts. From the pool of tweets meeting these criteria, we randomly selected some for our dataset.

3.3 Feature Engineering

3.3.1 Twitter Related Features

For our Twitter related features, we used the Tweepy API to extract useful information from the unstructured Twitter data. These features included both Twitter user features as well as features from the raw text of tweets. Specifically, we analyzed the number of Twitter users who follow the user (follower count), the number of Twitter users who are followed by the user (following count), the number of days since the creation of the account (age of account), the number of tweets the user has marked as favorite (user favourite count), the number of lists that the user is subscribed to (lists count), the length of the tweet message (message length), the number of times a tweet has been retweeted (retweet count), the URL count, the number of times a tweet has been marked as a favorite (tweet favorite count), the number of user mentions in a tweet (mention count), the number of hashtags in a tweet (hashtag count), and the count of digits in a tweet (digit count). These features, extracted from the available metadata and also utilized by other researchers in previous studies, are left to our models to determine their significance in our analysis.

3.3.2 Text classification features

TextAnalyst is a module within IntelliTweet that incorporates contextual features to capture intricate text patterns. This parameter can be toggled on or off to include textual features, such as polarity, subjectivity, sentiment, and TF-IDF measures of the raw tweet text during model training. Disabling it reverts to standard features.

BerTweet, as proposed by Dat Quoc et al. [46], is the first publicly available large-scale pre-trained language model for English Tweets. BerTweet was trained on a vast dataset

comprising approximately 850 million Tweets (16 billion word tokens). We applied transfer learning and utilized the pre-trained BerTweet to assess and return tweet sentiment, polarity, and subjectivity values, taking into account language usage and emotional intensity.

- **Subjectivity:** Subjectivity quantifies the amount of personal opinion and factual information contained in the text as a float between 0 and 1. A higher subjectivity value suggests that the text contains personal opinions rather than factual information.

- **Polarity:** Polarity measures the sentiment's direction, ranging from -1 (negative) to 1 (positive). It considers the tone, emotions, and judgments expressed in the tweet. A polarity value of 1 signifies a positive statement, while -1 indicates a negative statement.

- **TF-IDF:** Additionally, we apply TF-IDF analysis to assess the significance of terms within tweets. TF-IDF calculates a numerical value for each term in a tweet, indicating its importance relative to the train dataset for our system. These values serve as features for our classification model.

Table 3.1: Summary of Twitter Related Features

USER FEATURES	TWEET FEATURES	TEXT ANALYSIS FEATURES
Follower count	Message length	TF-IDF
Following count	Retweet count	Subjectivity measure
Age of account	Tweet Favourites count	Polarity measure
User Favourites count	URL count	Sentiment
Lists count	Hashtag count	
	Mention count	
	Digit count	

3.3.3 URL features

From each tweet that we collected, we extracted the embedded URLs within the text. As underscored by previous studies [7,11,29], the nature and characteristics of URLs included in tweets can serve as significant indicators of potential malicious intent. To further analyze these links, we scrutinized each URL and assessed redirections to track subsequent paths upon clicking the link.

To facilitate the featurization process, we drew inspiration from the methodology outlined by Kamel et al. [19] as outlined in their study. The approach they described captures various aspects of URL behavior and characteristics. These attributes include the URL's relationship to the address bar, its impact on HTML content and JavaScript execution, WHOIS lookup results, and domain-specific attributes. The resulting feature set, as detailed in Table 3.2, was used for our analysis.

Table 3.2: URL Features

Having IP Address	Website Traffic	URL of Anchor
Having "@" Symbol	On mouseover	Website Forwarding
Prefix/Suffix in Domain	DNS Record	IFrame, Favicon
Request URL	Disabling Right Click	Age of Domain
Multi Sub Domains	PageRank	Pop-up Window
Domain Registration Length	Redirect	Google Index

3.4 System Flow of IntelliTweet

The general internal system flow of our solution is illustrated in Figure 3.2. In the utilization of IntelliTweet, Tweets that are collected from the stream undergo a cleaning process, eliminating stopwords, punctuations, emojis, and stemming the content. The feature ex-

tractor then separates the URLs from the main tweet text for feature extraction. The system visits each URL to collect the raw data (webpage's HTML content, JavaScript, WHOIS lookups, and domain-specific information) associated with it. It then encodes and converts these raw features into a usable format.

In the process of URL encoding, features are assessed based on their presence or absence in a URL. For example, the "Having_IP_Address" feature checks for the presence of an IP address in the URL and returns 1 if found, -1 if absent. Similarly, the "Having_@_Symbol" feature identifies the "@" symbol in the URL and returns 1 if found, -1 if absent.

It is important to note that four of the URL features incorporate counting mechanisms in their encoding process. Firstly, the "Having_Sub_Domain" feature counts the number of subdomains within a URL, assigning -1 when none are found, 0 for a single subdomain, and 1 for multiple subdomains. Secondly, the "Redirect" feature counts the redirections a URL experiences and classifies them as -1 for none, 0 for one or two, and 1 for more than two. The "URL_of_Anchor" feature evaluates the ratio of external to total links on a webpage, returning values of -1 for a ratio $<$ 0.31, 0 for 0.31 $<=$ ratio $>=$ 0.67, and 1 for a ratio higher than 0.67. Lastly, the "Request_URL" feature calculates the count of external domains in the resources loaded by a webpage and returns -1, if the threshold is $<$ 0.22, 0 if 0.22 $<=$ threshold $>=$ 0.61 and 1 for a threshold higher than 0.61. These features offer a nuanced analysis of URLs by incorporating quantitative assessments into their encoding.

After encoding the URLs, the feature extractor then extracts the metadata of the tweet, including the user account features (following count, follower count, age of the account, user favorites count, list count) and full-text features (message length, retweet count, favorites count, URL count, hashtag count, mention count). The processed tweet text obtained from the extractor is subsequently fed into the text analyzer. This analyzer is trained to discern

various aspects of the tweet, including its sentiment, subjectivity, polarity, and TF-IDF score. This information is stored as additional features. All of these are then combined as one set of feature vectors and the classifier provides the result.

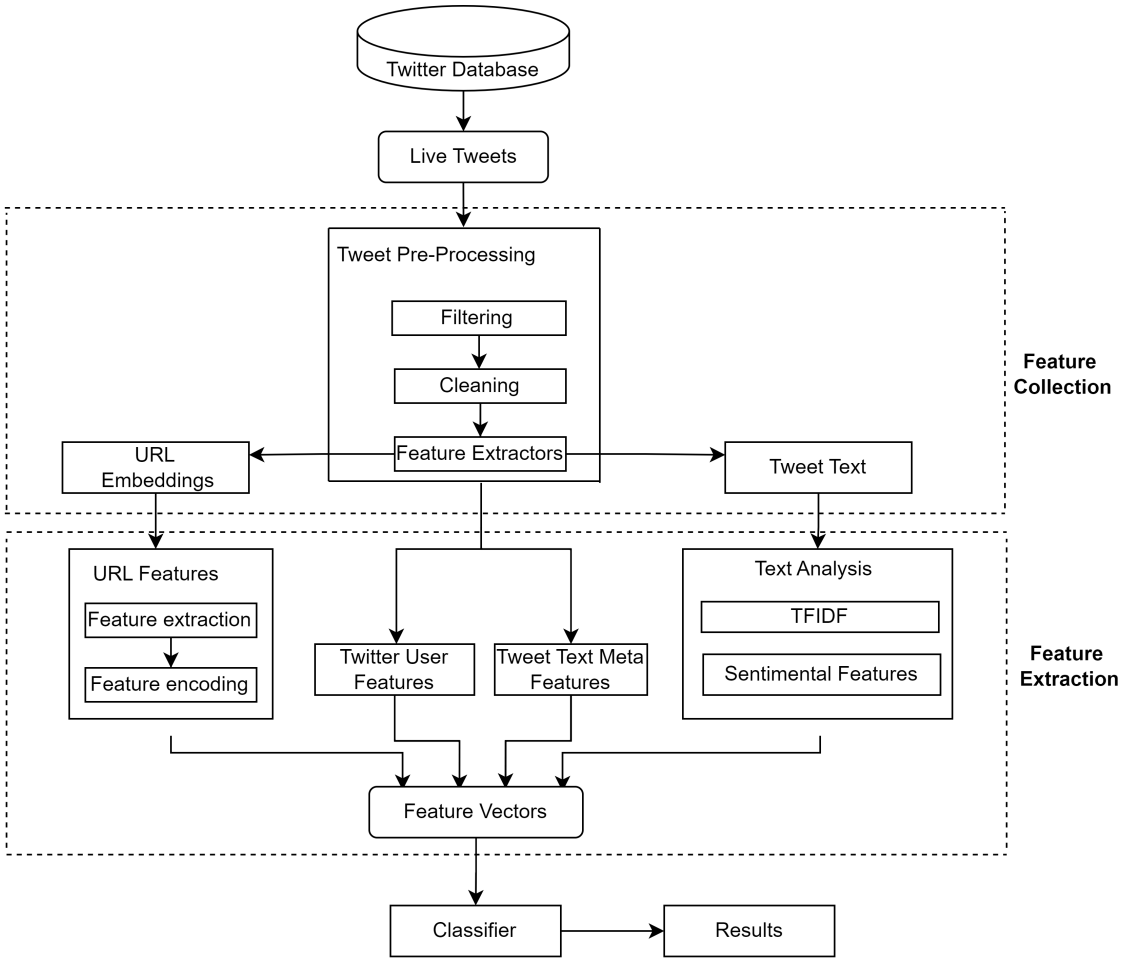


Figure 3.2: IntelliTweet System Flow

3.5 The Binary and Multi System Setups

IntelliTweet currently functions as a binary classifier, categorizing tweets as either malicious or legitimate. This binary approach assumes that user reports are authentic and primarily serve to raise awareness of potential scams or phishing links. In an ideal scenario, these reports are treated as legitimate tweets with good intentions.

However, the introduction of the multi-class system setup prompts a reconsideration of this assumption, acknowledging the presence of user reports as a unique label in their own right, leading to a nuanced perspective.

To address this nuance, we have established a distinct setup for IntelliTweet – a multi-class classification system. This transformation enables IntelliTweet to distinguish between legitimate tweets, malicious tweets, and user reports by re-labeling tweets in the dataset, specifically identifying reports, and subsequently retraining models for multi-class predictions.

For a comprehensive evaluation of the model's performance under each setup, we conduct various experiments, detailed in Chapter 4, and present the corresponding results. This dual-system approach allows us to capture the nuances of both binary and multi-class scenarios, providing a holistic perspective on IntelliTweet's effectiveness in identifying and categorizing tweets on Twitter.

By combining content and context-based features, real-time data, and innovative representations, IntelliTweet aims to address the dynamic and multifaceted challenges posed by malicious activities on Twitter. This multifaceted approach is used to offer a significant advancement in the field of malicious tweet detection, ultimately contributing to a safer and more secure online environment.

3.6 Summary

IntelliTweet is presented as a comprehensive system to enhance the detection and mitigation of malicious tweets on Twitter.

The system incorporates a multifaceted approach with 2 main configurations, binary and multi-class representations. Numerous features are extracted including text, user, tweet and URL-based, selected using mutual information scores. The system flow involves data collection/cleaning, feature extraction/selection, text analysis via BERTweet and final classification.

As a component of developing IntelliTweet, a significant contribution entails the creation of a real-time Twitter dataset extracted directly from the live stream. This dataset is openly shared with fellow researchers for their utilization [20].

In the next chapter, we will present experiments of all the configurations of IntelliTweet, including variants that were initially implemented but failed. We will showcase the results obtained using the experimental design outlined in this chapter 3, evaluating IntelliTweet's effectiveness in enhancing malicious tweet detection.

Chapter 4

IntelliTweet Designs and Evaluation Results

In this chapter, we outline the different experimental setups conducted as part of the development of IntelliTweet. We detail our evaluation criteria and present the outcomes obtained for each experiment as per our methodology. We explore various configurations and their results, shedding light on the evolution of IntelliTweet.

4.1 Evaluation Criteria

In assessing IntelliTweet's effectiveness in identifying malicious tweets, we undertook various experiments to validate its performance. Our primary goal for this system is to minimize False Positive Rate (FPR) while maintaining a high level of precision. This focus is important, as IntelliTweet helps in identifying malicious users, potentially leading to actions like content removal or account blocking by Twitter. We don't want our system to

detect a legitimate user as malicious since their account can be suspended. In erring on the side of caution, we prioritize avoiding false positives (incorrectly identifying a malicious tweet) over false negatives (missing a malicious tweet).

Moving forward, in formulating an appropriate evaluation criteria, we had to balance precision and recall. Improving precision by reducing false positives may have the trade-off of decreasing recall, potentially resulting in missed detections, thereby impacting the overall performance of the system. Conversely, prioritizing recall might increase false positives. Through continuous fine-tuning and adjustment in an iterative feedback loop, we arrived at an optimal configuration that reflects IntelliTweet's objectives.

Considering how these trade-offs impact the overall effectiveness, we assessed a variety of metrics, including accuracy, recall, precision, false positive rate, and F1-score, to ensure our models are well-balanced across all metrics. The trade-offs were carefully calibrated, with a focus on precision and maintaining a low FPR, aligning with our criteria for the real-world implications of user misclassification.

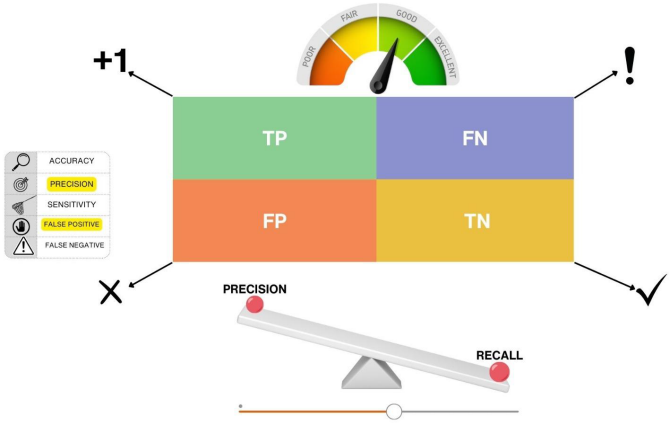


Figure 4.1: Striking the right balance between metrics

4.2 Feature Importance

In the pursuit of understanding the critical features driving IntelliTweet's classification capabilities, several feature selection strategies were considered. These strategies include Recursive Feature Elimination (RFE), Principal Component Analysis (PCA), Mutual Information (MI) and Tree-based methods, specifically the Random Forest Feature Importance. Each approach offers distinct advantages and limitations.

- **Recursive Feature Elimination:** is a backward feature selection method commonly used with machine learning algorithms such as Random Forests. It iteratively removes the least significant features until the optimal subset of features is identified, based on model performance metrics.
- **Principal Component Analysis:** is a dimensionality reduction technique that transforms the original feature space into a lower-dimensional space while preserving the most important information. PCA is particularly useful when dealing with high-dimensional data and can help identify significant components while preserving variance in the dataset.
- **The Random Forest Feature Importance:** assess feature importance by evaluating the contribution of each feature to the overall predictive performance of the model. Features with higher importance scores are considered more influential in the classification task.
- **Mutual Information:** assesses the relationship between two variables by measuring the amount of information gained about one variable through the other. MI quantifies how much information one feature provides about the target variable. It calculates

the reduction in uncertainty regarding the target variable given the knowledge of a feature.

4.2.1 Rationale for choosing Mutual Information

Mutual Information emerged as the preferred method in our analysis due to its ability to capture both linear and non-linear relationships between features and target variables. Unlike some other methods that assume linear relationships, MI offers a more flexibility making it well-suited for the complex and diverse data landscape present in Twitter datasets. Moreover, MI accommodates non-parametric data and handles categorical variables effectively, aligning with the heterogeneous nature of Twitter data.

Our feature importance analysis using Mutual Information scores revealed key insights into the nuanced interplay of all the various features used in IntelliTweet. Notably, features such as TF-IDF, the number of tweets, polarity, subjectivity, account age, follower count, following count, page rank, user favorite count, web traffic, the absence of hashtags, domain registration length, retweet count, right-click occurrences, DNS records, pop-up events, and redirects emerged as the most influential contributors to the classification task. These findings underscore the multifaceted nature of malicious tweet detection, where a combination of content, contextual, user-related, URLs and engagement-based features significantly impacts the model's ability to distinguish between different tweet categories.

Mutual Information scores in Figure 4.2 provides a visual representation of these influential features, highlighting TF-IDF as the highest contributor. This underscores the pivotal role of textual content in distinguishing tweet categories, particularly in malicious tweet detection. This integrated analysis provided us with valuable insights for refining and enhancing our model, laying the foundation for a robust and effective system.

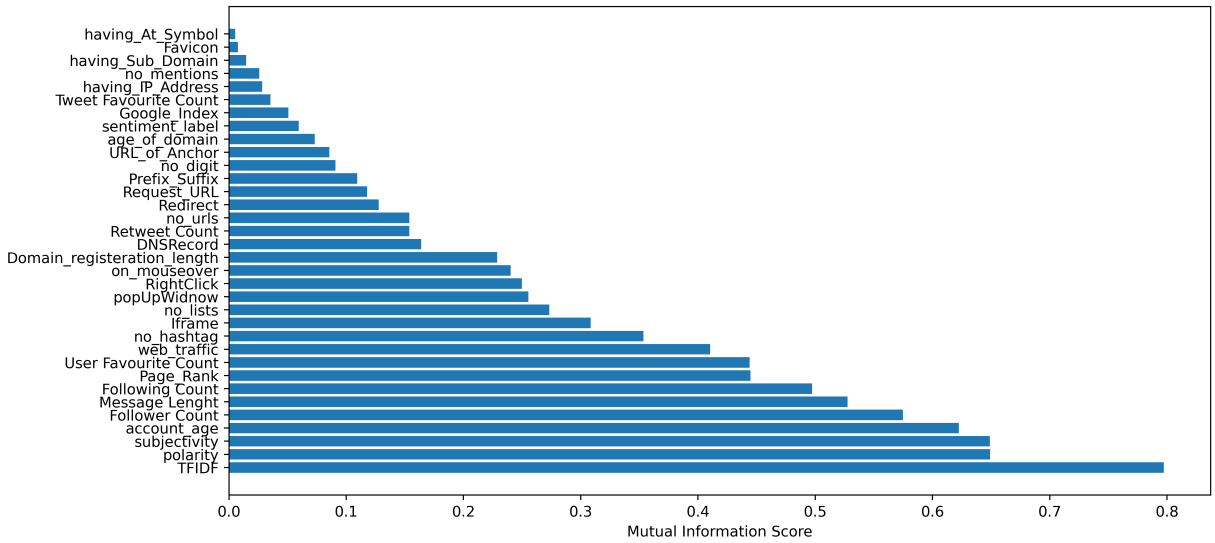


Figure 4.2: Feature Importance by Mutual Information

4.3 Binary System

4.3.1 Model selection and evaluation

To select the optimal model for our malicious tweet classification system, we conducted an experiment evaluating several models on a dataset of 6,000 tweets. This dataset contained an even distribution of 2,000 malicious tweets, 2,000 legitimate tweets, and 2,000 tweets reporting malicious content that we categorized as legitimate data to establish a legit legitimate-to-malicious ratio of 2:1. Our strategy emphasizes comprehensive evaluation based on minimizing false positives and optimizing global weighted average precision, not merely maximizing accuracy.

4.3.1.1 Default Parameters

We initially constructed all models using default parameters from scikit-learn. Specifically, the random forest used 100 estimators, Gini criterion, random state of 0, and other defaults; the decision tree had Gini splitting and best splits; the SVM had C=1.0, RBF kernel, etc. We also assessed the logistic regression classifier, bagging decision tree, and gradient boosting classifier with their respective default parameters.

To ensure a robust evaluation, we employed a stratified 5-fold CV methodology, repeating it five times across the dataset. This approach provided a thorough and reliable assessment of the models. Table 4.1 shows the results obtained for the performance of the evaluated models.

Table 4.1: Average Metrics for Models after five Repetitions of Stratified 5-fold CV

Classifier	Class	Precision(%)	Recall(%)	F1(%)	FPR
Random Forest	Malicious	97.6643	**99.915**	98.7658	**0.085**
	Legitimate	**99.8259**	95.1300	97.3662	
Decision Tree (DT)	Malicious	92.8118	98.4350	95.4392	1.565
	Legitimate	96.0659	83.7600	88.7001	
Bagging DT	Malicious	94.2004	98.9700	96.4674	1.03
	Legitimate	97.6932	87.2900	91.7925	
Gradient Boosting	Malicious	96.8346	99.5150	98.1153	0.485
	Legitimate	98.9985	93.1500	95.7472	
Support Vector	Malicious	**98.9236**	99.1000	**99.0111**	0.900
	Legitimate	98.1975	**97.8400**	**98.0157**	
Logistic Regression	Malicious	98.1388	97.4600	97.7974	2.54
	Legitimate	94.9948	96.3000	95.6398	

Using default parameters, the Random Forest achieved the top performance. Precision reached approximately 99.8% for legitimate tweets, and 97.7% for malicious tweets. Recall stood approximately at 95.1% for legitimate, and 99.9% for malicious and a false positive

rate of 0.085. The high metrics confirms Random Forest can differentiate between the classes despite their nuances.

Among alternatives, the Gradient Boosting and Support Vector Machine also performed well with 96-99% metric values across all classes but there is still room for improvement on the models by tuning hyperparameters.

4.3.1.2 Hyperparameters in Machine Learning

Hyperparameters are attributes or properties of a machine learning or deep learning model that serve two main purposes. They aid in decision-making during training and inference, and configuring the model's architecture to influence its operation. These parameters act as knobs that allow us to fine-tune the behavior of machine or deep learning classifiers. Various methods are employed for this purpose:

- **Manual Search:** A domain expert manually explores hyperparameter settings, adjusting them based on the observed performance until satisfactory metrics are achieved.
- **Grid Search:** This method involves defining a search space for each hyperparameter. The performance metric is computed for all possible combinations of hyperparameters, and the best configuration is selected. Grid search is chosen for our use case; the subsequent section explains the rationale behind this choice.
- **Random Search:** In random search, the search space and the total number of iterations are specified. Models are trained and evaluated using random hyperparameter combinations, and the best-performing settings are chosen.

- **Bayesian Optimization:** Unlike grid and random search, Bayesian optimization employs a statistically guided approach to search for optimal hyperparameters, avoiding the exhaustive consideration of all possible combinations.

4.3.1.3 Rationale for Choosing Grid Search

Grid search is selected for our hyperparameter optimization process due to its simplicity and comprehensiveness. By systematically exploring the entire search space, grid search ensures that no configuration is overlooked, making it suitable for our use case where a thorough examination of hyperparameter combinations is desired.

4.3.1.4 Parameter Tuning

With the results obtained from the default parameters, our decision at this point leaned towards the Random Forest model due to its promising performance. However, to establish further confidence and optimize the model's parameters for the selected features, we conducted a grid search. This advanced step aimed to fine-tune the models and identify the most suitable combination of hyperparameters for our dataset. The grid search ensured that each algorithm was fine-tuned to its best configuration, contributing to a fair and comparative evaluation. We tested various parameters specific to each algorithm, such as the learning rate for gradient boosting or the regularization parameter for logistic regression, the number of trees in the forest, the maximum depth of the tree, and other relevant parameters.

The key enhancements during parameter tuning involved:

- Increasing estimators from 100 to 270 for more diverse trees

- Maximum features considered per split expanded to log2
- Using Gini impurity for node splits as suited for classification
- And set the random_state to 0 for reproducibility

The subsequent performance metrics, including precision, recall, F1-measure, accuracy, and false positive rate in table 4.2 reflect the outcomes of this detailed optimization process for the entire ensemble of models in our malicious tweet classification system.

Table 4.2: Average Metrics for Optimized Models after five Repetitions of Stratified 5-fold CV

Classifier	Class	Precision(%)	Recall(%)	F1(%)	FPR
Random Forest	Malicious	97.7445	**99.9300**	98.8156	**0.07**
	Legitimate	**99.8566**	95.3100	97.4840	
Decision Tree (DT)	Malicious	93.1953	98.3450	95.6203	1.655
	Legitimate	95.9853	84.8700	89.5616	
Bagging DT	Malicious	94.2004	98.9700	96.4674	1.03
	Legitimate	97.6932	87.2900	91.7925	
Gradient Boosting	Malicious	96.2464	99.7550	97.9228	0.245
	Legitimate	99.4840	91.8200	95.2265	
Support Vector	Malicious	**98.9236**	99.1000	**99.0111**	0.900
	Legitimate	98.1975	**97.8400**	**98.0157**	
Logistic Regression	Malicious	98.1388	97.4600	97.7974	2.54
	Legitimate	94.9948	96.3000	95.6398	

Among the models tested, our selection criteria led us to choose the Random Forest Classifier. After the grid search, we fine-tuned the Random Forest model with the optimal specific parameters identified through this process. The refined Random Forest model exhibited exceptional performance, achieving a global weighted average precision of 98.80% (standard deviation 0.0008), recall of 97.62% (standard deviation 0.0017), F1-measure of 98.15% (standard deviation 0.0014), accuracy of 98.39% (standard deviation 0.011), and a

low false positive rate of 0.07. This meticulous optimization through grid search ensures that the selected model operates with the most effective configuration for our specific malicious tweet detection task, contributing to its robust performance in real-world scenarios.

4.3.2 IntelliTweet and TextAnalyst Evaluation:

As indicated earlier in section 3.3.2, TextAnalyst is a module in IntelliTweet, which can be toggled on or off to enhance model training using sentiment analysis and TF-IDF text vectorization to capture intricate text patterns. We evaluated the predictive improvement from these advanced natural language processing techniques through stratified repeated cross-validation experiments summarized in Tables 4.3 and 4.4.

Table 4.3: Average of 5 folds per iteration of IntelliTweet without TextAnalyst

Metrics	Iteration					Global Average ± Std Dev
	1	2	3	4	5	
Accuracy	0.9848	0.9730	0.9822	0.9803	0.9768	0.9794± 0.0046
Recall	0.9779	0.9604	0.9744	0.9711	0.9660	0.9700 ± 0.0069
Precision	0.9883	0.9806	0.9859	0.9853	0.9825	0.9845 ± 0.0030
F1	0.9826	0.9676	0.9795	0.9772	0.9733	0.9761 ± 0.0058

Table 4.4: Average of 5 folds per iteration of IntelliTweet with TextAnalyst

Metrics	Iteration					Global Average ± Std Dev
	1	2	3	4	5	
Accuracy	0.9852	0.9822	0.9837	0.9838	0.9847	0.9839 ± 0.0011
Recall	0.9781	0.9736	0.9761	0.9759	0.9773	0.9762 ± 0.0017
Precision	0.9888	0.9869	0.9875	0.9882	0.9886	0.9880 ± 0.0008
F1	0.9830	0.9793	0.9813	0.9814	0.9824	0.9815 ± 0.0014

The stratified cross-validation experiments demonstrate the value added by IntelliTweet's

TextAnalyst module. With TextAnalyst enabled, gains were observed across the board in accuracy, recall, precision, and F1-score compared to the base model without text analysis techniques. Notably, accuracy climbed from 97.94% to 98.39% when activating TextAnalyst. Recall saw a rise from 97.00% to 97.62% which translates to a slightly better detection of positive malicious cases. Precision also increased up from 98.45% to 98.80%, ensuring reliable malicious tweet identification while minimizing false positives. Finally, the F1 metric balances these precision and recall gains, reflected in its climb from 97.61% to 98.15%. The stratified methodology and the consistent standard deviations across iterations indicate stable, reliable gains. Overall, these slight increases demonstrate the tendency of TextAnalyst to positively impact performance.

The confusion matrix results in Tables 4.5 and 4.6 provide further insight into IntelliTweet's improvements with TextAnalyst, by breaking down the types of predictions across the stratified 5-fold cross validation with five repetitions for both cases of IntelliTweet with Text analyst on and off.

Table 4.5: Mean of confusion matrices (IntelliTweet TextAnalyst parameter ON)

		Predicted	
		Malicious	Non-Malicious
Actual	Malicious	1906.2	93.8
	Non-Malicious	2.8	3997.2

Table 4.6: Mean of confusion matrices (IntelliTweet TextAnalyst parameter OFF)

		Predicted	
		Malicious	Non-Malicious
Actual	Malicious	1883	117
	Non-Malicious	6.4	3993.6

On average, false positives, where the model incorrectly flags legitimate tweets as malicious, dropped from 6.4 without text analysis down to 2.8 with TextAnalyst. This marks an approximately 56% reduction in false positives, supporting TextAnalyst's effectiveness in preventing unnecessary mislabeling of benign tweets. Meanwhile, false negatives improved slightly from 117.0 to 93.8. Thus, the precision gains from TextAnalyst predominantly stem from reducing false alarms rather than missing actual tweets. This aligns well with IntelliTweet's goal of maintaining high accuracy for regular tweets to avoid unjust penalties for users. Notably, while false positives decreased, true positives remained consistently high at 19064.2, and true negatives also increased.

The confusion matrices of the iterations of the stratified 5 fold cross validation can be found in Appendix A.1.

4.3.3 IntelliTweet compared to a Baseline approach

After reviewing various literature sources, we selected the methodology introduced by Kamel Djaballah et al. [19] as our baseline solution. This choice stemmed from several key factors. Firstly, Kamel's method provided the dataset used for training URLs and incorporated a blacklist approach, augmenting the filtering process to reduce false positives, particularly for phishing URLs. Secondly, their methodology addressed the challenge of shortened URLs, which might bypass traditional detection systems. Additionally, they examined the user posting the tweet, adding another layer of analysis.

Their approach comprised three distinct steps. Firstly, they verified the presence of URLs contained in a tweet against a database of malicious URLs (blocklisting). Subsequently, if the URLs did not match any entries in the database, they proceeded to the

second step. This involved employing supervised machine learning to classify the URL, and consequently the tweet, into one of two categories: phishing or legitimate. This classification was based on a predefined set of features and a predictive model applied to URLs. In the third step, they introduced a user analysis mechanism to lessen the spread of phishing links on Twitter.

We conducted an experiment to compare TextAnalyst parameter ON and OFF, and the baseline method used in Kamel's [19] on our dataset.

The results in Figure 4.3 reveal that IntelliTweet demonstrates stability and outperforms the baseline across both states - with and without TextAnalyst enabled. This is evidenced by the median values consistently being higher across all metrics. The baseline exhibits reduced variability, as indicated by small box sizes, implying more consistent predictive capability. When TextAnalyst is disabled, variability slightly increases, notably in the recall but remains relatively tight in the other metrics. Enabling TextAnalyst compensates for this reduced variability and restores consistency in all metrics.

It is worth noting that the approach in [19] does not account for reports within their dataset. Instead, their machine learning approach primarily focuses on detecting the presence of malicious URLs and blocklists to predict phishing tweets. Their user account analysis was only an extra verification step. Consequently, when we tested the approach in the work of Kamel [19], as a baseline on our dataset, the system's performance was relatively poor. The confusion matrix shown in Table 4.7 highlights a significant number of false positives. This outcome can be attributed to the fact that Kamel's approach, when applied to our dataset, is unable to differentiate reports from other types of tweets. Table 4.8 offers a potential explanation for this observation, proposing that Kamel's system labels reports as malicious when blocklisted URLs are present. Notably, a substantial portion of the false positive predictions (2077.4) is due to misclassifying reports, which is a clear error.

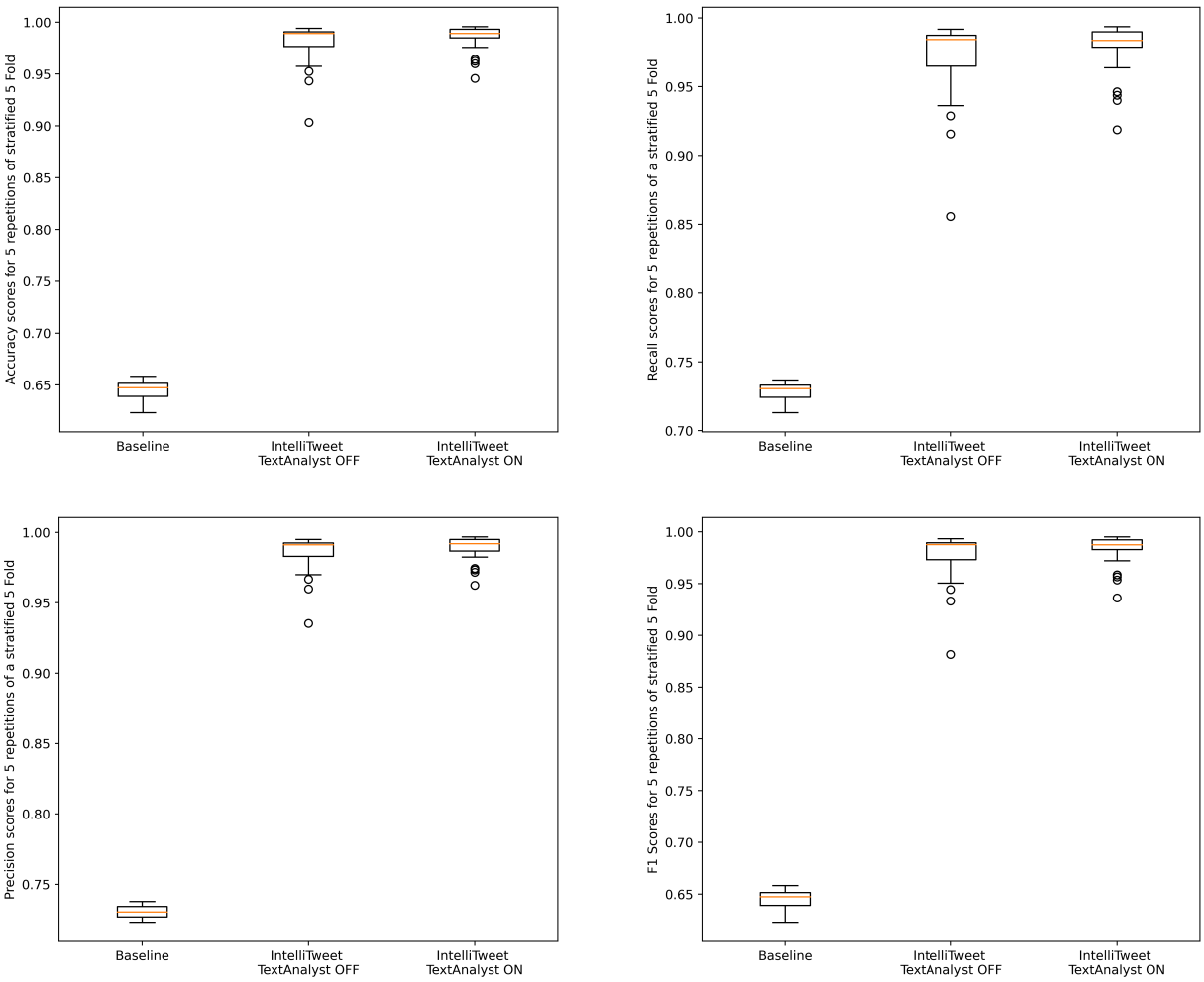


Figure 4.3: Box Plot for performance evaluation

However, even less obvious false positives originating from legitimate tweets (231.2) remain relatively high compared to the 2.8 false positives observed in IntelliTweet. This comparison again futher highlights the performance of IntelliTweet particularly in minimizing false positives and handling the nuanced nature of user reports more effectively.

We conducted runtime analysis comparing the system with and without the TextAnalyst

Table 4.7: Mean of confusion matrices of Kamel's approach

		Predicted	
		Malicious	Non-Malicious
Actual	Malicious	1954.0	46.0
	Non-Malicious	2077.4	1922.6

Table 4.8: Mean of confusion matrices with emphasis on Non-Malicious Predictions

		Predicted	
		Malicious	Non-Malicious
Actual	Legitimate	231.2	1768.8
	Report	1846.2	153.8

component, along with the baseline system. Including text features enhances malicious tweet detection but increases runtime to 2,753 milliseconds, compared to 2,327 milliseconds without this feature. The baseline system achieved a runtime of 1,902, which is efficient but yielded poorer metric results with a precision of 73.12%. TextAnalyst's advantage lies in the improved false positive rate, allowing users to customize IntelliTweet by activating or deactivating the feature according to their needs.

4.4 Multi System Experiments and Setups

As described in Chapter 3.5, the multi-system configuration enables IntelliTweet to distinguish between legitimate tweets, malicious tweets, and user reports.

We adapted the binary dataset to support multi-class classification by re-labeling reported tweets, while keeping malicious and legitimate labels intact. With the new 3-class labeled data, IntelliTweet was re-trained for multi-class classification. The primary tuning

required was configuring the terminal node setup. Instead of binary classification, nodes were now predicting one of three labels. We optimized the multi-class entropy criteria for node splitting to achieve maximum information gain between the classes, using the same features shown in Figure 4.2 for model training.

4.4.1 Model Selection for Multi-Class Classification

With the dataset now expanded to three classes through report re-labeling, we re-assessed model selection for multi-class classification. Similar to the binary system, we initially utilized default parameters in the training of the models. Table 4.9 below shows the results.

4.4.1.1 Default Parameters Model Performance

The default parameterized models indicate promising capability for distinguishing between legitimate tweets, malicious tweets, and user reports even before parameter tuning. Once again, the random forest achieved the top performance across metrics, demonstrating effective multi-class learning. Recall exceeded 90% across all tweet types, nearing 99.2% for Malicious posts. Precision also scored strongly in the Legitimate and Reports with approximately 99%. This reliably minimizes false positives during classification. The high F1-scores in the 94 -96 % range showcase the balance between precision and recall. Notably, the false positive rate increased to 0.095% compared to the binary system. This increased specificity, results from modeling additional nuances between classes instead of just binary legitimate/malicious splits.

The decision tree and some other classifiers struggled with balancing performance across classes. Precision, recall and F1 lagged in the report class.

Table 4.9: Average Metrics for Models with default parameters after five Repetitions of Stratified 5-fold CV

Classifier	Class	Precision(%)	Recall(%)	F1(%)	FPR
Random Forest	Malicious	89.8453	**99.2100**	94.2328	
	Legitimate	**99.8028**	94.4600	96.9543	**0.095**
	Report	**99.2776**	93.9300	96.5191	
Decision Tree (DT)	Malicious	75.9715	91.8300	82.7800	
	Legitimate	93.6731	85.2800	88.6430	3.035
	Report	91.3020	78.1900	83.1813	
Bagging DT	Malicious	79.1085	96.0600	86.5807	
	Legitimate	97.6455	86.2200	91.0845	1.025
	Report	96.8312	86.3700	91.1776	
Gradient Boosting	Malicious	91.8339	97.9600	94.7542	
	Legitimate	99.3673	95.7900	97.4916	0.310
	Report	98.1779	94.9600	96.5329	
Support Vector	Malicious	**94.2894**	96.2600	**95.2603**	
	Legitimate	98.0576	**98.2400**	**98.1462**	0.975
	Report	97.9886	**95.7400**	**96.8482**	
Logistic Regression	Malicious	92.6522	91.2600	91.9376	
	Legitimate	94.3611	96.5600	95.4436	2.89
	Report	94.3521	93.5200	93.9232	

4.4.1.2 Optimizing Model Parameters

We then refined and optimized the models for the multi-class problem through grid search and cross-validation. Similarly to the binary system the number of estimators was 270, log2 for maximum features considered per split, gini impurity for node splits, random state of 0 and 2 for the minimum leaf samples to prevent underfitting.

The Table 4.10) presents the results after optimization.

The comprehensive grid search to fine-tune models tailored them to better performance on the three-class tweet classification task. The optimized random forest continued to ex-

Table 4.10: Average Metrics for Optimized Models after five Repetitions of Stratified 5-fold CV

Classifier	Class	Precision(%)	Recall(%)	F1(%)	FPR
Random Forest	Malicious	90.7852	**99.2000**	94.7657	
	Legitimate	**99.8349**	95.4200	97.5211	**0.08**
	Report	**99.2593**	94.2500	96.6803	
Decision Tree (DT)	Malicious	75.8253	91.6300	82.5495	
	Legitimate	93.4188	84.2100	87.6565	3.125
	Report	91.1433	78.4500	83.5653	
Bagging DT	Malicious	79.1085	96.0600	86.5807	
	Legitimate	97.6455	86.2200	91.0845	1.025
	Report	96.8312	86.3700	91.1776	
Gradient Boosting	Malicious	90.4574	98.4800	94.2211	
	Legitimate	99.7360	94.4200	96.8968	0.125
	Report	98.2837	94.4400	96.3066	
Support Vector	Malicious	**95.2374**	95.8500	**95.5368**	
	Legitimate	98.5007	**98.0400**	**98.2657**	0.75
	Report	97.2597	**97.0600**	**97.1564**	
Logistic Regression	Malicious	92.6522	91.2600	91.9376	
	Legitimate	94.3611	96.5600	95.4436	2.89
	Report	94.3521	93.5200	93.9232	

hibit top performance across all categories. It balances high precision and recall to maximize F1 scores. The FPR decreased to 0.08 indicating specificity in avoiding false alarms. Overall, optimized Random Forest achieves robust performance for multi-class malicious tweet detection. Notably, precision exceeded 99% on legitimate tweets and reports, preserving IntelliTweet's emphasis on minimizing false positives. Precision also surpassed 90% on malicious tweets - reliably detecting each class. Recall reached 94-99% across the board, successfully identifying the majority of test cases. The high F1 scores confirm the balance between precision and recall.

4.4.2 TextAnalyst in MultiClass System

Similar to the binary system presented in Section 4.3.2, we tested the effect of TextAnalyst in the multiclass domain. The stratified cross-validation presented in Sections 4.11 4.12 experiments demonstrate the value provided by text analysis techniques on top of base features for differentiating between multiple tweet types.

Across the board improvements were observed in accuracy, recall, precision and F1 scores when activating the TextAnalyst module, highlighting its benefits for enhanced comprehension of tweets' contextual meaning. Specifically, precision received a slight boost of 0.65% from 95.98% without TextAnalyst to 96.63% when enabled. Recall and accuracy also climbed by over 0.76%, leading to more reliable detection of all case types with fewer false alarms.

By enhancing the content and contextual understanding of TextAnalyst, the model better distinguishes between classes - malicious tweets aim to mislead, user reports raises awares and call out violations, while legitimate posts behave normally. The quantifiable metrics showcase this - higher precision ensures reliable identification and separation.

Table 4.11: Average of 5 folds per iteration of IntelliTweet with TextAnalyst

Metrics	Iteration					Global Average ± Std Dev
	0	1	2	3	4	
Accuracy	0.9617	0.9560	0.9678	0.9630	0.9660	0.9629 ± 0.0046
Recall	0.9617	0.9560	0.9678	0.9630	0.9660	0.9629 ± 0.0046
Precision	0.9649	0.9615	0.9698	0.9664	0.9687	0.9663 ± 0.0033
F1	0.9620	0.9563	0.9681	0.9634	0.9663	0.9632 ± 0.0045

The confusion matrix results provide further confirmation of TextAnalyst's benefits in improving multi-class classification performance. By breaking down prediction types, we gain additional insight.

Table 4.12: Average of 5 folds per iteration of IntelliTweet without TextAnalyst

Metrics	Iteration					Global Average ± Std Dev
	0	1	2	3	4	
Accuracy	0.9582	0.9473	0.9572	0.9595	0.9542	0.9553 ± 0.0048
Recall	0.9582	0.9473	0.9572	0.9595	0.9542	0.9553 ± 0.0048
Precision	0.9615	0.9550	0.9607	0.9632	0.9586	0.9598 ± 0.0031
F1	0.9585	0.9476	0.9575	0.9598	0.9547	0.9556 ± 0.0049

The confusion matrices below in Tables 4.13 and 4.14 summarize respectively the average predictions across the stratified 5 fold cross validation with 5 repetitions for both cases of the multisystem with Text analyst on and off.

Table 4.13: Mean of confusion matrices (IntelliTweet multisystem TextAnalyst parameter OFF)

		Predicted		
		Malicious	Report	Legitimate
Actual	Malicious	1891.2	1.2	107.6
	Report	2.2	1863.6	134.2
	Legitimate	7.8	15.4	1976.8

Table 4.14: Mean of confusion matrices (IntelliTweet multisystem TextAnalyst parameter ON)

		Predicted		
		Malicious	Report	Legitimate
Actual	Malicious	1908.4	0.4	91.2
	Report	1.0	1885.0	114.0
	Legitimate	2.2	13.8	1984.0

Notably, false positives on the legitimate tweet class significantly dropped from 7.8 without TextAnalyst down to 2.2 with text analysis enabled. This 72% relative decrease in false alarms aligns with IntelliTweet's priority of avoiding incorrectly penalizing benign users. At the same time, false negatives on actual malicious tweets barely climbed from 107.6 to 91.2. So the precision improvements predominantly arose from slashing false positives rather than missing detections.

Furthermore, while the false positive metric on the user report class slightly dropped, true positives on reports increased from 1863.6 to 1885.0. So the model seems to slightly shift priority based on TextAnalyst's signals to maximize performance on the two classes with real-world actions - malicious and legitimate.

The confusion matrices of each iteration are highlighted in the Appendix A.2 .

Additionally we rerun the setup in Section 4.3.3: TextAnalyst ON and TextAnalyst OFF on our multisystem dataset, and compared to Kamel's approach [19]

The results in Figure 4.4 reveal: More Intellitweet consistently outperforms the baseline, as evidenced by the higher median values across all metrics. When IntelliTweet's TextAnalyst parameter is enabled, variability slightly increases from when it is disabled. Nevertheless, both states of IntelliTweet exhibit stability, with minimal differences in the median lines.

4.5 T-Test Experiment for Precision Values

Upon initial observation, the multi system and the binary system appeared similar with no significant difference. To delve deeper, we conducted a T-test to ascertain if there existed a statistical difference between the different configurations of IntelliTweet, focusing on binary

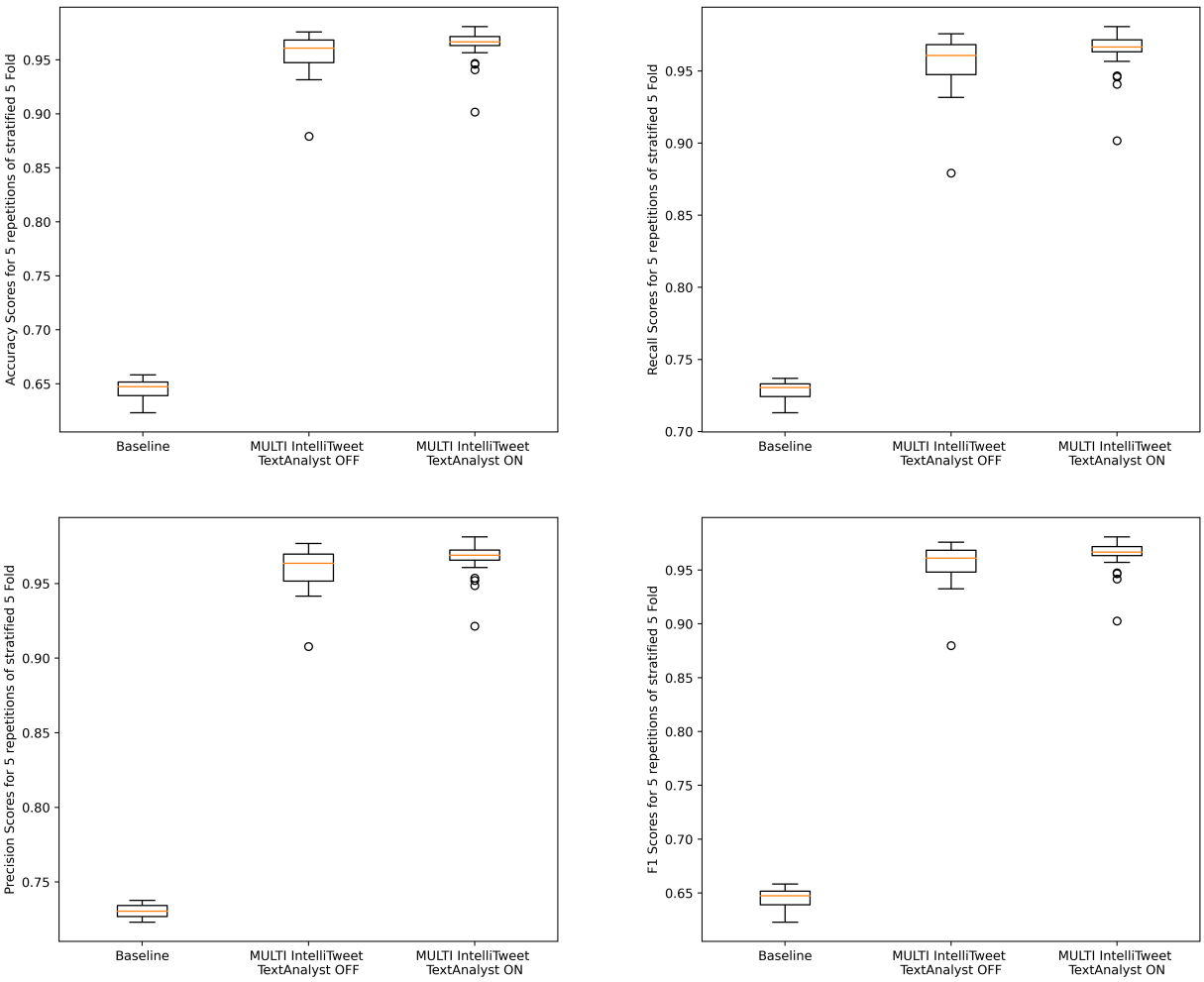


Figure 4.4: Multi System Box Plot for performance evaluation

and multi-system setups with and without TextAnalyst (TA). The paired t-test (p-test) assesses whether the means of two paired groups are statistically different from each other.

Before proceeding with the t-test, we verified that our data met the necessary assumptions:

Normality: We checked the normality of the precision value distributions for each

configuration using the Shapiro-Wilk statistical tests. The precision values followed an approximately normal distribution, satisfying the normality assumption.

No outliers: We examined the precision value distributions for the presence of outliers, which can significantly influence the results of the t-test. After careful examination, we confirmed that there were no outliers present in our data.

After confirming that our data met the assumptions of the t-test, we proceeded with the analysis. Adhering to our primary goal of minimizing false positives and maximizing overall precision, we used precision values, performed and evaluated a two-sided p-test to gauge differences in mean precision values across these configurations.

4.5.1 Experimental Setup

We collected the precision values for each configuration over the stratified 5-fold CVwith five iterations. The average precision values were recorded for each iteration, and the mean and standard deviation were calculated for analysis.

Table 4.15: Precision Values for Different Configurations

Index	Binary w/o TA	Binary w/ TA	Multi w/o TA	Multi w/ TA
0	0.9883	0.9888	0.9615	0.9649
1	0.9806	0.9869	0.9550	0.9615
2	0.9859	0.9875	0.9607	0.9698
3	0.9853	0.9882	0.9632	0.9664
4	0.9825	0.9886	0.9586	0.9687
avg	0.98452	0.9880	0.9598	0.966260
stddev	0.00301	0.000791	0.003152	0.003279

Table 4.16: Paired Differences, Means, Standard Deviations, and p-values

Index	Binary w/o TA minus(-) Binary w/ TA	Binary w/ TA minus(-) Multi w/o TA	Multi w/o TA minus(-) Multi w/ TA	Multi w/ TA minus(-) Binary w/o TA
0	-0.0005	0.0273	-0.0034	-0.0234
1	-0.00630	0.0319	-0.0065	-0.0191
2	-0.00160	0.0268	-0.0091	-0.0161
3	-0.00290	0.0250	-0.0032	-0.0189
4	-0.00610	0.0300	-0.0101	-0.0138
avg	-0.00348	0.0282	-0.00646	-0.01826
stddev	0.002348	0.002447	0.002836	0.003228
p-value	0.041386	0.000021	0.010371	0.000348

4.5.2 Interpreting the p-test Results:

The significance level set for our analysis was 0.05, implying that tests resulting in less than 95% confidence would be rejected. With a degree of freedom of 4 (one less than the 5 folds used), we formulated our null hypothesis that the models under consideration were similar, while the alternate hypothesis posited that the models differed.

The p-values derived from the test serve as indicators of the significance of differences between the systems. A low p-value, typically below the significance level (0.05 in our case), suggests strong evidence against the null hypothesis, signifying significant differences among the models concerning precision.

We computed the paired differences between precision values for each pair of configurations: Binary without TA vs. Binary with TA, Binary with TA vs. Multi without TA, Multi without TA vs. Multi with TA, and Multi with TA vs. Binary without TA. Subsequently, using the scipy [1] library, we calculated the mean, standard deviation, and p-values for each paired difference.

[1]Scipy Library v1.12.0 Manual

In our analysis, the obtained p-values were 0.041386, 0.000021, 0.010371 and 0.000348 for the pairs: "Binary without TA - Binary with TA", "Binary with TA - Multi without TA", "Multi without TA - Multi with TA", and "Multi with TA - Binary without TA", respectively. All these p-values fell below the significance level of 0.05, indicating compelling evidence to reject the null hypothesis and assert that the models exhibit differences.

The results imply that the inclusion of TextAnalyst influences model precision, and there exist performance disparities between binary and multi-system setups, irrespective of TextAnalyst's presence.

4.6 Summary

We presented a comprehensive overview of the experiments conducted during IntelliTweet's development, providing insight into the iterative refinement process. We began by establishing an evaluation criteria aligned with IntelliTweet's goals of minimizing false positives while maximizing precision. Our methodology emphasized robust evaluation through stratified cross-validation across multiple model configurations.

After assessing various classifiers, we selected and optimized a Random Forest model using grid search. Comparisons to a baseline Twitter phishing detection approach demonstrated IntelliTweet's performance.

Feature analysis via Mutual Information highlighted the multifaceted predictors driving detection, showcasing the prominence of TF-IDF, user details, text polarity, page rank, and engagement metrics in distinguishing tweet types, thereby emphasizing the value of a hybrid feature set.

We expanded the dataset and adapted the model for multi-class classification across

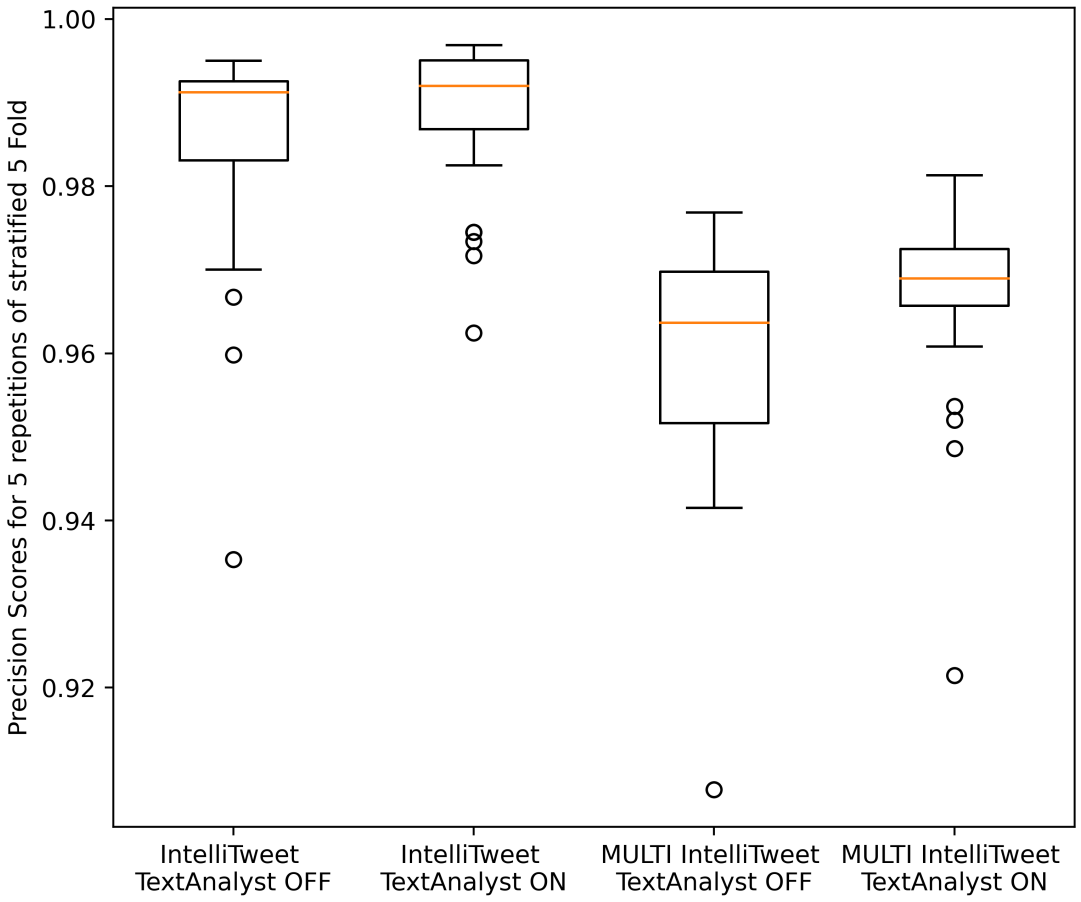


Figure 4.5: Box plot for precision values of IntelliTweet configurations

malicious tweets, legitimate tweets, and reports revisiting assumptions of report authenticity. Similar evaluation criterion confirmed IntelliTweet's effectiveness despite the different setup. Additional experiments using the paired t-test revealed statistically significant differences between the Multi-system and Binary system, emphasizing the importance of understanding and optimizing each configuration's unique characteristics.

Through comprehensive experiments following established practices, we systematically matured IntelliTweet into an accurate system exhibiting reliable precision. The evaluations provide confidence in its real-world viability for enhanced malicious tweet detection on Twitter.

Chapter 5

Previous Experiments: Paving the Path to IntelliTweet

This chapter delves into the preliminary experiments conducted in the journey towards building IntelliTweet. We discuss the variants initially implemented, including those that failed and the reasons behind their shortcomings. We extract insights from these experiments and propose avenues for future work to capitalize on these insights in further refining and enhancing IntelliTweet.

5.1 Cascade System with Majority Voting

In the initial stages of our work, a cascade system comprising three compartments was devised: the URL Detection System, the Intent Recognition System, and the User Analysis System. Each compartment served a distinct purpose. The URL Detection System was trained to identify malicious URLs, the Intent Recognition System aimed to predict the

intent of a tweet (malicious, reported, or Legitimate), and the User Analysis System focused on studying user accounts to detect suspicious patterns.

The User Account Detection compartment was trained using a 5k-random large ground truth dataset obtained from the work of Chen et al. [17] to predict whether an account is suspicious or not. The 5k ground truth dataset was split into training and testing sets, with 25% of the data reserved for testing and the remaining data shuffled for training purposes. Various classifiers were trained on this training data, and the results obtained on the test set were as follows:

Classifier	**Accuracy**	**Recall**	**Precision**	**F1**
Random Forest	0.8692	0.8575	0.8843	0.8707
Logistic Regression	0.6072	0.4159	0.6971	0.5210
Support Vector Machine	0.6556	0.7593	0.6385	0.6937
Gradient Boosting	0.836	0.8411	0.8398	0.8405
Decision Tree Bagging	0.8132	0.7975	0.8318	0.8143

The Intent Recognition component utilized a pre-trained BERT model (BertForSequenceClassification from the "bert-base-uncased" checkpoint) and a corresponding BERT tokenizer for text preprocessing. The model was trained for 3 epochs using the AdamW optimizer with a learning rate of 2e-5 and a batch size of 32. The training data was a combination of datasets from Kaggle and Twitter, following the approach proposed by Rodrigues et al. [49]. The Twitter dataset consisted of 3,000 tweets, categorized into 1,000 malicious, 1,000 user-reported, and 1,000 non-malicious tweets. Additionally, the Kaggle dataset, initially containing 5,572 rows labeled as 4,825 non-spam and 747 spam messages, was relabeled as malicious and non-malicious to align with the training approach. Over the course of the 3 training epochs, the model's performance improved steadily, achieving an accuracy of 99.25% by the end of training.

To validate the performance of the Intent Recognition component, a test was conducted using 600 samples of real tweet data. The results were as follows:

- Spam: 165/200 correctly classified
- Report: 186/200 correctly classified
- Legitimate: 200/200 correctly classified

The URL Detection System was trained using the Public UCI Machine Learning phishing dataset [44], which comprises instances from PhishTank, MillerSmiles, and Google. Various classifiers were tested, Random Forest, Logistic Regression, and Support Vector Machine (SVM) were tested, with Random Forest achieving the best results with all metrics slightly above 90%.

5.1.1 Cascade Operation

The cascade system operates as follows: when there is an incoming tweet, all three compartments make predictions, and a majority voting mechanism is employed with a 2-vote requirement for a final prediction. The possible combinations for a final prediction are:

- Malicious URL + Suspicious User Account + Malicious Intent → Malicious
- Malicious URL + Suspicious User Account + Clean Intent → Malicious
- Malicious URL + Clean User Account + Malicious Intent → Malicious
- Malicious URL + Clean User Account + Clean Intent → Clean
- Non-Malicious URL + Suspicious User Account + Malicious Intent → Malicious

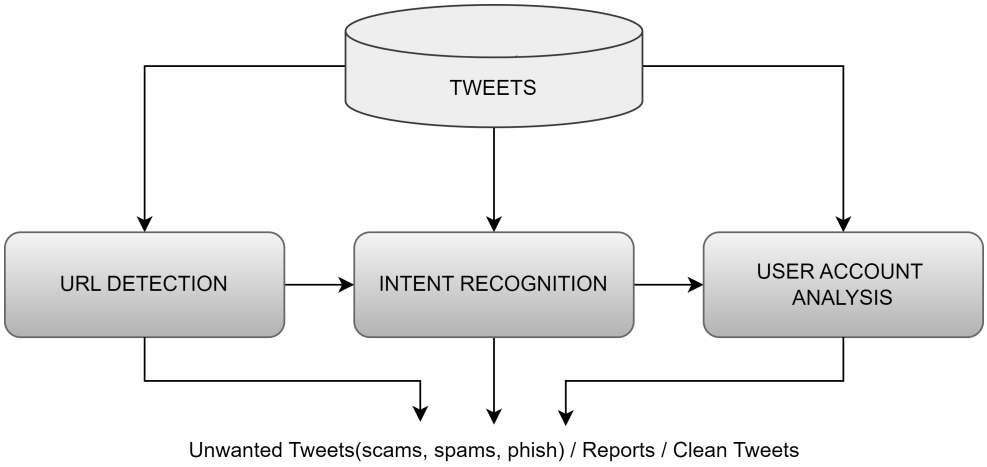


Figure 5.1: Cascade Design

- Non-Malicious URL + Suspicious User Account + Clean Intent → Clean
- Non-Malicious URL + Clean User Account + Malicious Intent → Clean
- Non-Malicious URL + Clean User Account + Clean Intent → Clean
- Malicious URL + Suspicious User Account + Report Intent → Report
- Malicious URL + Clean User Account + Report Intent → Report
- Non-Malicious URL + Suspicious User Account + Report Intent → Report
- Non-Malicious URL + Clean User Account + Report Intent → Report

Remark: We chose to discontinue the cascade system because the various combinations for final predictions became confusing. Moreover, the equal weighting in the majority voting process proved impractical. These challenges prompted us to reconsider our approach and seek a more straightforward and effective solution for accurate predictions.

5.2 Online Learning Approach

In the context of machine learning, online learning (also known as incremental learning, online training, or real-time learning) refers to a training paradigm where models learn and update themselves in real-time as new data becomes available one at a time, without needing to retrain on all data from scratch [12]. This is in contrast to traditional batch learning, where the model is trained on a fixed dataset. Some key aspects of online learning are:

- **Sequential Learning:** Models are trained iteratively on individual data samples or mini-batches rather than full datasets. This allows the model to adapt as new data arrives sequentially.
- **Real-time Updates:** Online learning allows models to adapt to changes in the data distribution over time. When a new data point arrives, the model makes a prediction, compares it to the true label/value, and updates its parameters based on the error. This is often done using optimization algorithms like stochastic gradient descent.
- **Efficiency:** Online learning algorithms emphasize computational efficiency to large datasets as they avoid the need to process the entire dataset in each iteration.
- **Scalability:** Online learning can handle streaming data and can scale to large datasets that might not fit into memory. This scalability is particularly valuable in scenarios where data is continuously generated.
- **Memory Efficiency:** Online learning is memory-efficient as it does not require storing the entire dataset in memory. The model updates its parameters based on the most recent observations.

This technique is commonly applied and well-suited for domains, such as natural language processing, fraud detection, real-time prediction systems, recommendation systems, and other applications where the data is generated and updated over time or the model needs to stay current and adapt to changes in the environment.

5.2.1 Integrating Online Learning in IntelliTweet

In the initial stages of our work, we introduced a shift in IntelliTweet's learning approach from batch to online. This adjustment allows the model to update in real-time as tweets continuously flow into the system. Instead of training once on a fixed historical dataset, IntelliTweet utilizes a sliding window that retains the n most recent samples, dynamically training the model on this continuous influx of data. This approach is advantageous as it enables IntelliTweet to adapt to emerging features used by malicious actors. Verified malicious tweets appearing in the stream are processed in real-time, and their features are added to the sliding window.

Figure 5.2 shows the architectural design of the online approach. In this design, when a new data point arrives, we get the model to make a prediction then we compare it to the true label/value, and finally update the model based on the error. The features extracted from the incoming data are stored in a feature store. Consequently, the model incrementally updates itself with the latest examples of malicious behavior, focusing on learning new threats while gradually forgetting outdated patterns in the data sink.

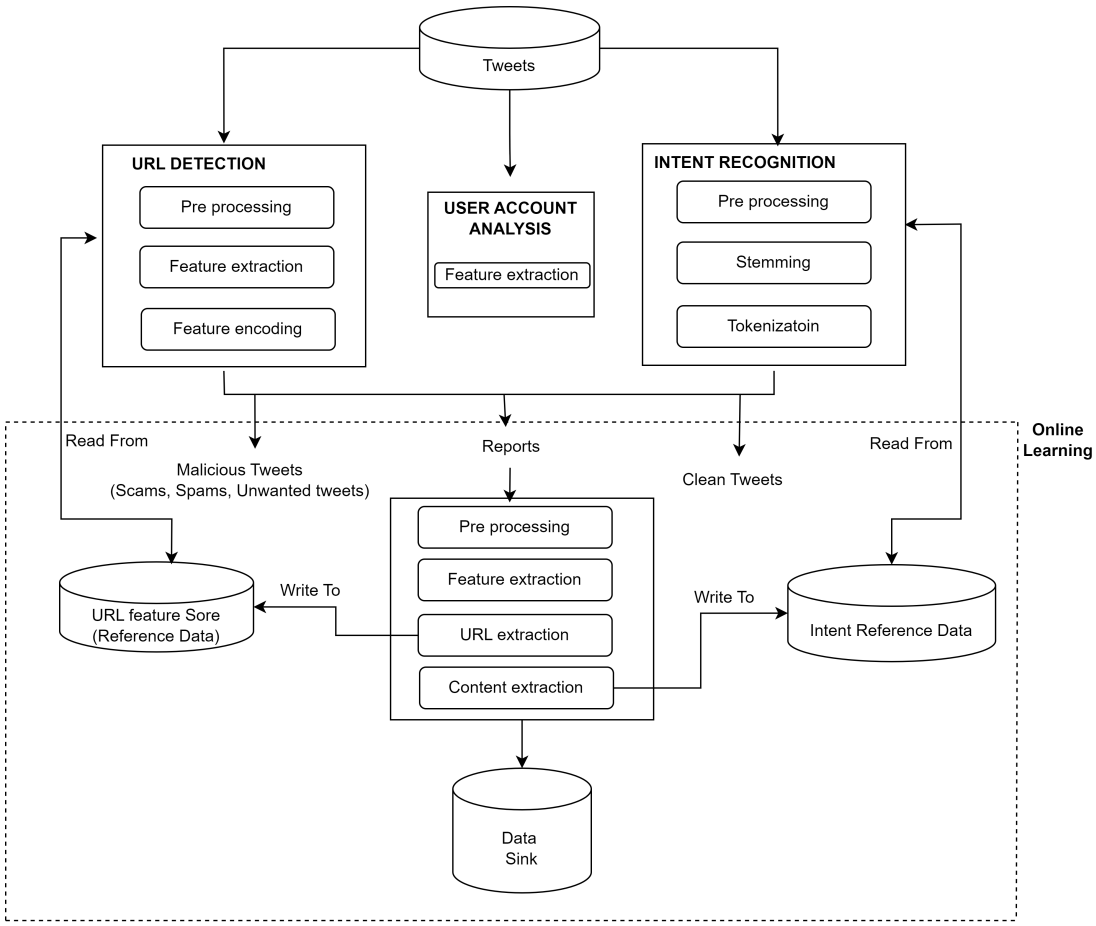


Figure 5.2: Online Learning Architecture Design

5.2.2 How is the evolution by use of an online model addressed

5.2.2.1 A. By Incorporating External Verified Labels

Assumption: Antiphishing services such as Virustotal, Phishtank, Openphish, Google Safe Browsing, etc., do not have access to all malicious URLs but have partial knowledge of some. By learning the techniques and extracting patterns in the form of features from these partial ground truth URLs, with an Online Model, we can continuously learn and generalize these URL obfuscation techniques to identify new URLs.

Our online stream integrates feeds of confirmed malicious URLs from external sources like PhishTank, VirusTotal, and Google Safe Browsing. Extracting features from these verified examples fine-tunes the URL detection model to emerging threats. For instance, when VirusTotal flags a new phishing URL, we capture its features and add them to the stream window, training our model to recognize these malicious indicators. This form of online learning, coupled with external verification sources, is designed to provide a preliminary level of confidence in real-time malicious tweet detection. The classifier adapts flexibly to the ever-changing social media threat landscape.

To test this approach, we conducted a simple experiment. Initially, we statically trained a random forest model using some URL data, achieving baseline metrics of accuracy, recall, precision, and F1 slightly above ∼90%. We then implemented an online learning strategy, configuring the random forest as an online learning model with the warm_start parameter set to True. This setup allowed each subsequent fit call to add a new set of trees to the ensemble, processing URLs sequentially through a sliding window of 1000 samples.

As seen in Figure 5.3, metrics consistently improved from batch levels as the model incrementally updated itself on the latest threats. This small experiment only begins to

demonstrate online learning's potential to keep pace with emerging attacks by continually remodeling on contemporary data.

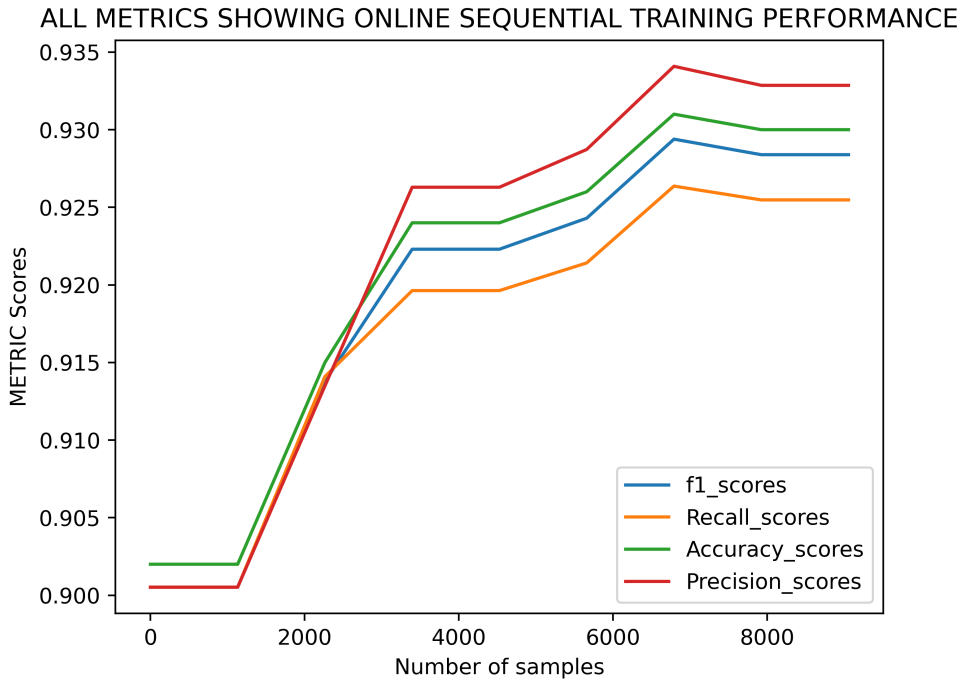


Figure 5.3: Sequential Training

While far from definitive, this simple study provided some promising evidence that online training can enable models to reliably adapt detection capability to new attacks.

5.2.2.2 B. By Leveraging Report Predictions for Online Learning

In our subsequent exploration of the online learning framework, we leveraged report predictions to enhance the model's predictive capabilities continuously. This involved the following key steps:

- **Sequential Availability of Predicted Reports:** Predicted reports were system-

atically presented in a sequential order. Each report prediction was strategically utilized to update the existing model, refining the best predictor for future data at each step.

- **Relearning from Malicious Messages in Reported Snapshots:** Extracting valuable insights from reported snapshots containing malicious messages, we extracted the text content within these snapshots with identified malicious messages. This extracted content, labeled as malicious tweets, was reintroduced into the system, allowing the model to relearn and adapt to evolving malicious communication.

- **Processing Reported URLs for Enhanced Generalization:** We processed reported URLs within the tweets of reporters, extracting techniques and patterns as features. These features were reintegrated into the system to enhance generalization, especially in overcoming URL obfuscation techniques employed by malicious actors.

The snapshots in Figure 5.4 depict a live test scenario where we received malicious text messages on a mobile phone, took screenshots of the messages, and tweeted them as reports to warn others of such malicious messages.

The idea behind leveraging reports for online learning of malicious tweets involves extracting reported tweets, including their texts, keywords, hashtags, and URLs. Since reports may contain malicious URLs, we extract these URLs and run a classification on them. Figures 5.6 and 5.7 show URLs from the reports predicted as malicious. These extracted features are reintegrated into the system, allowing the model to relearn and adapt to the evolving nature of malicious communication.

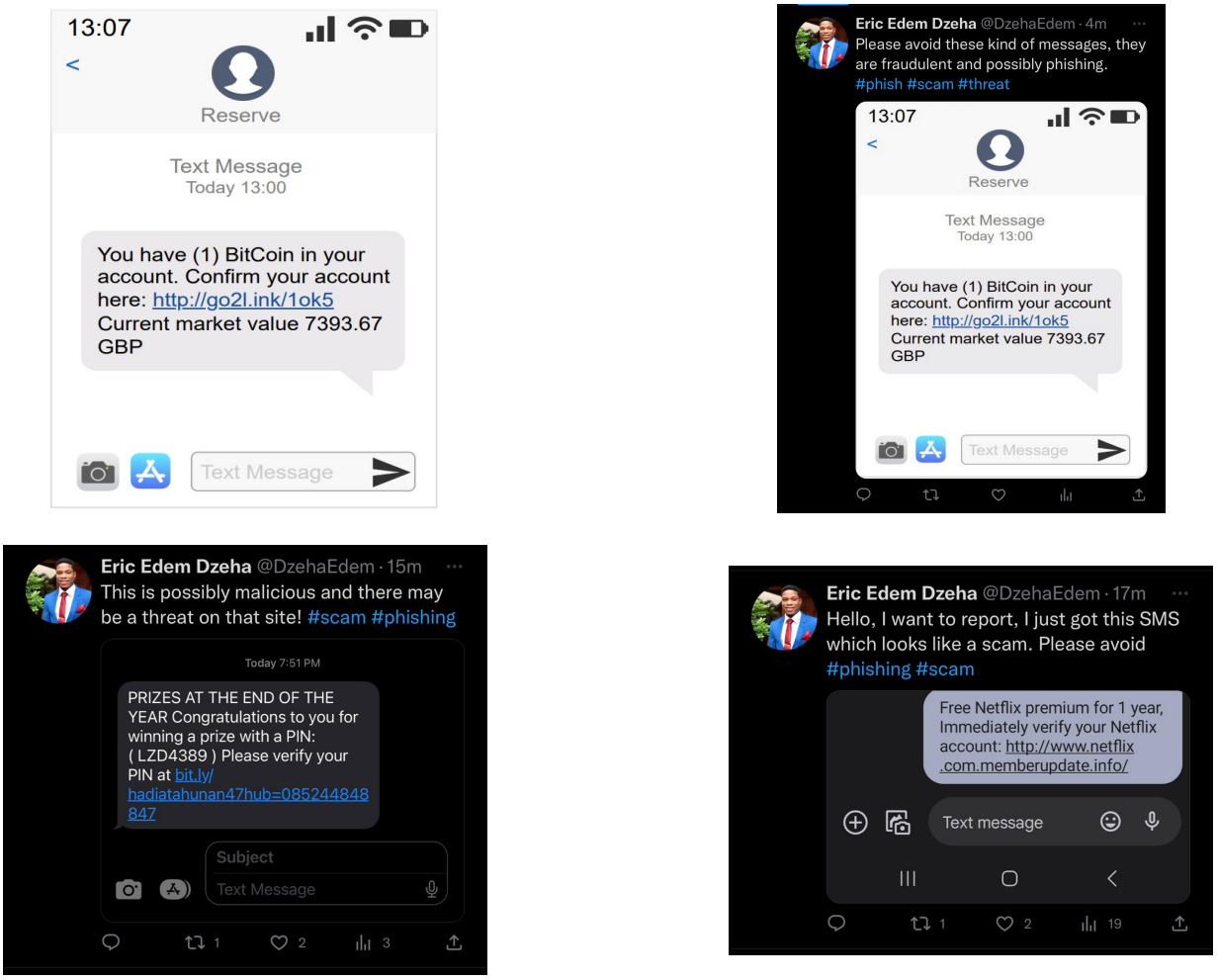


Figure 5.4: Use case scenarios and live tests

Challenges of the Online Configuration: We encountered several challenges when implementing the online configuration strategy. Firstly, storing and utilizing all historical data was challenging due to its continuous increase, requiring infinite running time for our experiments and storage. Additionally, the system constantly required mechanisms to identify and avoid processing the same rows in the streaming data to prevent redundancy. Moreover, our online setup relied on the cascade system, which itself was halted, presenting

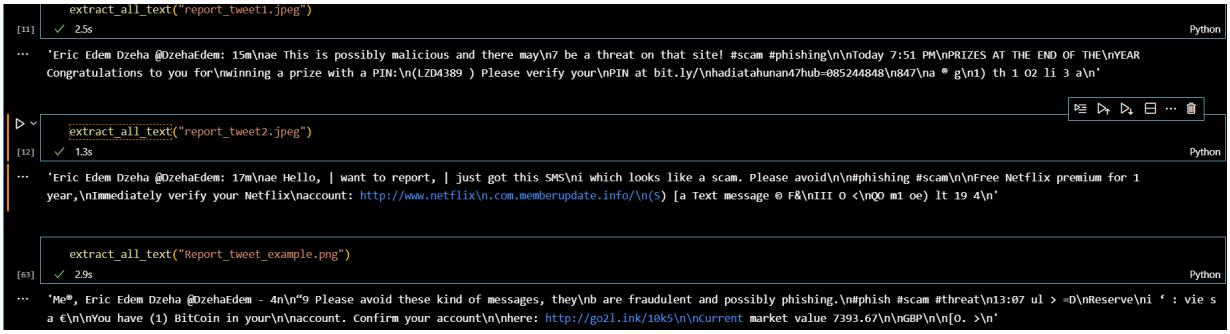


Figure 5.5: Tweet extraction API

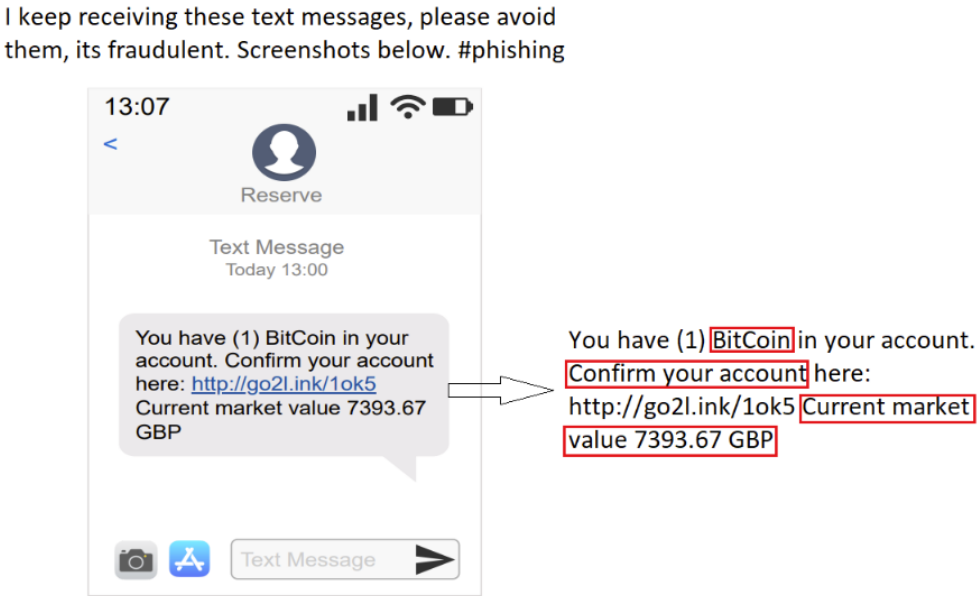


Figure 5.6: Test case 1

Test Case: You have (1) BitCoin in your account. Confirm your account here: https://promoark.org/btc/ Current market value $3000.00 → **Malicious**

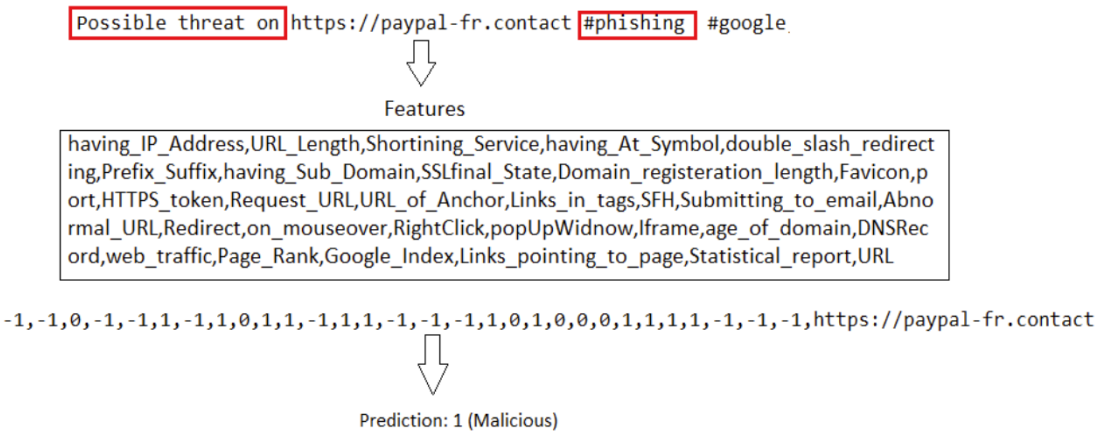


Figure 5.7: Test case 2

Test Case: You stand a chance to win 5000 dollars, click here for more details: https://bit.ly/Giveaway
-1,-1,1,-1,-1,-1,-1,1,0,1,1,-1,1,1,-1,-1,-1,1,0,1,0,0,0,1,1,1,1,-1,-1,-,https://bit.ly/Giveaway → **Malicious**

further limitations. There was also the concern that concept drift could become a problem as the underlying patterns in the data change over time. These challenges prompted us to pause the implementation of online learning and earmark it for future work. We emphasize the need for effective strategies to manage data volume, adapt to concept drift, and handle repeated rows in the processing stream.

5.2.2.3 C. Data Augmentation Using Reports

In an effort to enhance our model's understanding of malicious tweet features, we conducted an experiment with a 5-fold CV, involving the augmentation of data through insights gained from reports. The objective was to create synthetic malicious tweets by extracting insights from reported content. For instance, taking the reported tweet in Figure 5.7 as an example, we used the same URL features but appended new malicious user account features to the URL and crafted new tweet texts. The intention was to simulate a scenario where a

malicious actor attempts phishing, effectively creating our own set of malicious test cases and datasets.

Our baseline solution involved a binary classification of malicious tweets on our initial dataset. The experimental setup comprised two phases:

Phase 1: We initially predicted reports in the initial dataset, transforming the problem into a classification of report or not, rather than a binary classification of malicious or non-malicious tweets. Subsequently, we augmented the predicted reports, engineering them into malicious tweets. Maintaining the same test set in the 5-fold cross-validation, we retrained the system.

Phase 2: Following the retraining of the system with the augmented dataset, we reverted to a binary classification of malicious tweets. This allowed us to assess if the new malicious tweets, engineered based on insights from the reports, contributed to the system's performance. It's important to note that the test set remained consistent across all setups for all 5 folds.

The results of the binary classification in Phase 2 were then compared to the baseline solution to evaluate the impact of data augmentation using insights from reports.

Table 5.1: Metric results from 5 fold cross validation for Initial Data, "Is This Tweet Malicious or Not?"

Fold	Accuracy	Recall	Precision	F1
1	0.9896	0.9916	0.9852	0.9888
2	0.9879	0.9903	0.9828	0.9864
3	0.9237	0.8849	0.9477	0.9078
4	0.9340	0.9011	0.9530	0.9211
5	0.8264	0.7368	0.8948	0.7641
Avg	0.9323	0.9009	0.9527	0.9135

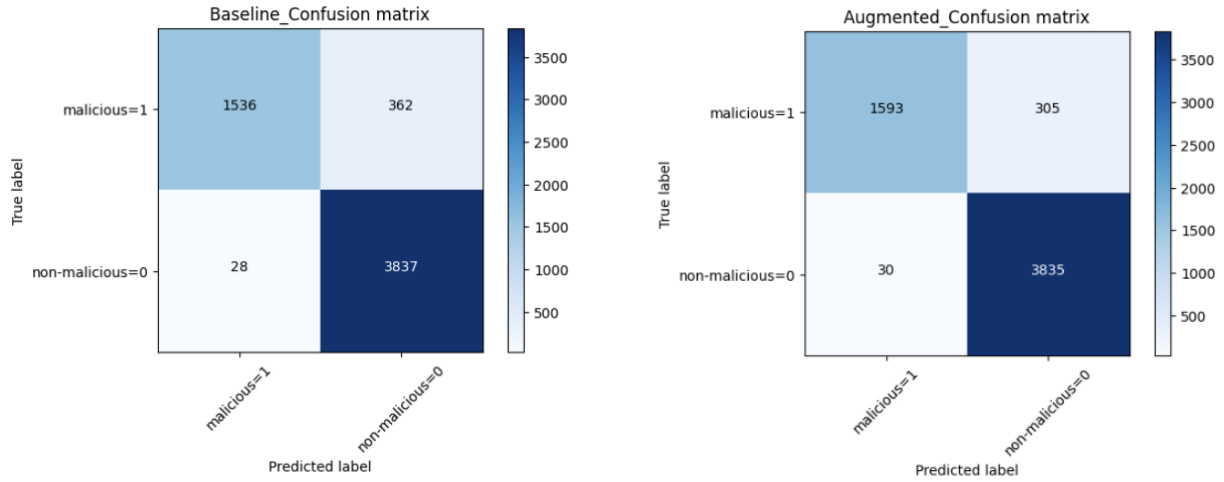


Figure 5.8: Predictions on "Is This Tweet Malicious or Not?"

Table 5.2: Metric results from 5 fold CV for Augmented Data "Is This Tweet Malicious or Not?"

Fold	Accuracy	Recall	Precision	F1
1	0.9835	0.9864	0.9772	0.9815
2	0.9922	0.9935	0.9890	0.9912
3	0.9653	0.9474	0.9754	0.9596
4	0.9392	0.9103	0.9542	0.9279
5	0.8290	0.7408	0.8960	0.7685
Avg	0.9419	0.9157	0.9584	0.9258

Discussion of Results: From the Figure 5.8 we observed that the augmented data slightly enhances the model's ability to predict malicious tweets. However, this improvement comes at the expense of a marginal increase in false positive predictions.

Continuing our efforts to enhance the effectiveness of reports, we incorporated sentiment analysis features into the experiment. This addition involved extracting the polarity score, subjectivity score, and sentiment labels from the tweet texts and incorporating them as

supplementary features. This objective aimed to provide deeper contextual insight and further refine our predictions, particularly with a focus of minimizing false positives.

As illustrated in Figure 5.9, the introduction of sentiment analysis in the report phase of the experiment, contributed to a notable reduction in false positives, decreasing from 135 to 107. Building on this advancement, we conducted a final evaluation of the augmented data, incorporating sentiment analysis into the entire process. The results, shown in Figure 5.10, demonstrated that the augmented dataset, now enriched with sentiment analysis, restored the earlier increase in false positives. Notably, while false positives decreased, the overall count of malicious predictions also saw a reduction, from 1,593 to 1,580.

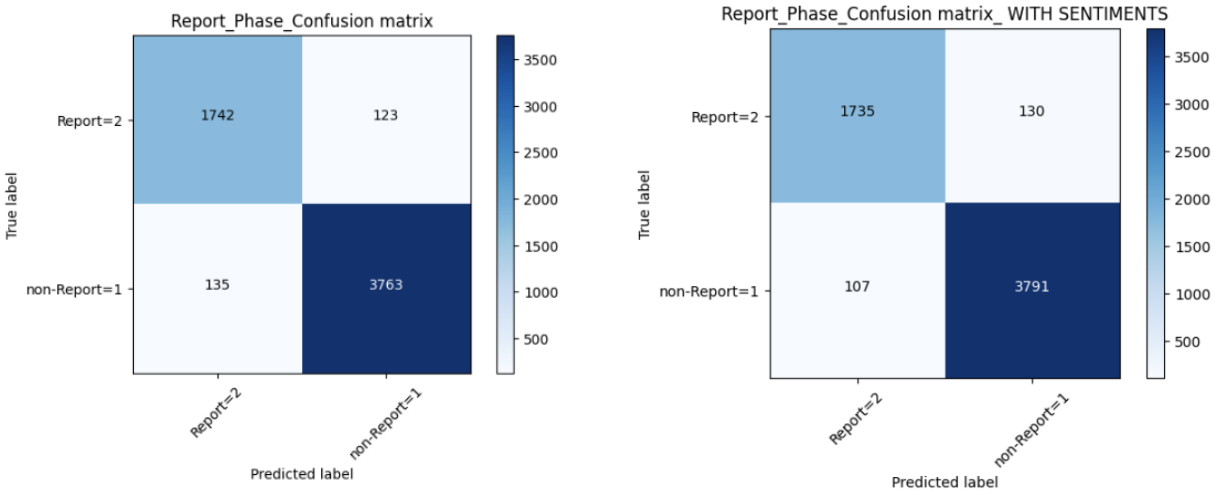


Figure 5.9: Report Phase Predictions "Is This Tweet a Report or Not?"

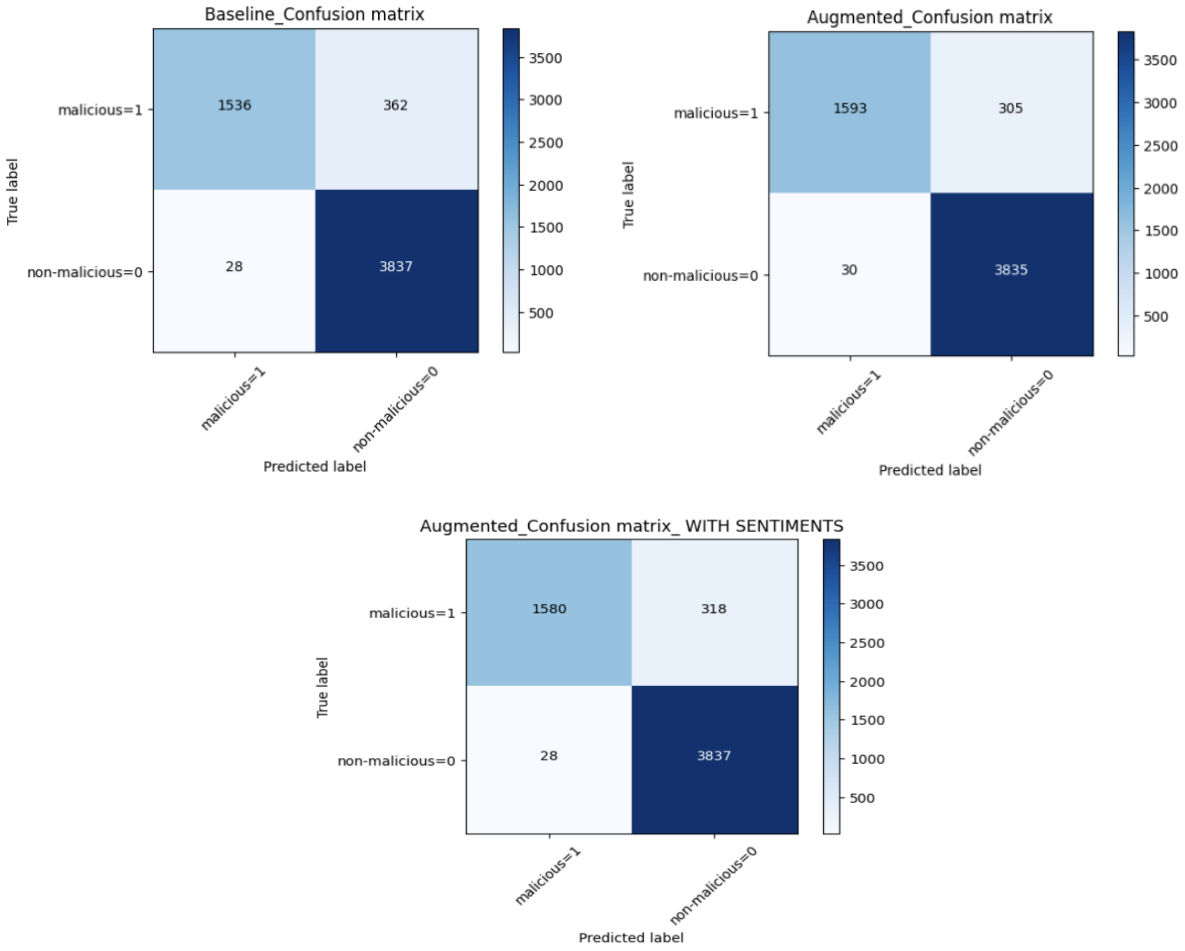


Figure 5.10: "Is This Tweet Malicious or Not?"
Initial Data, Augmented data, Augmented data + Sentiments

While recognizing the potential of the augmentation step we temporarily halted our exploration of augmentation techniques, stressing the need for further investigation. We conclude by stating that although the augmentation step holds promise, confidence in its stability with using only reports is not assured. Future work is planned, both in refining the augmentation approach and exploring the broader realm of online learning, as it holds substantial value if implemented correctly.

5.3 Summary and Key Takeaways

The initial experiments documented in this chapter provided valuable insights and lessons for the development of the final IntelliTweet system. The key takeaways from these early explorations are:

The online learning approach aimed to equip IntelliTweet with the capability to adapt to emerging threats by continuously learning from new data streams. Leveraging a sliding window approach enabled the system to undergo continuous model updates, showcasing scalability, efficiency, and adaptability to evolving data distributions. The simple experiment with the Random Forest model demonstrated promising results, with metrics consistently improving as the model incrementally updated itself on the latest threats. The integration of external verified labels from sources like VirusTotal and leveraging report predictions showed potential for continuous model improvement and enhanced the model's understanding, but requires further investigation. Despite initial promise, practical implementation encountered challenges in managing data volume, handling repeated rows, and mitigating concept drift.

Key Takeaways:

- Transitioning from a batch to an online learning paradigm facilitated the adaptation to dynamic data influx.
- Online learning enables real-time adaptation to evolving data, promoting scalability and efficiency.
- Integration of external verified labels and report predictions shows potential for continuous model improvement.

- Further research is needed to refine data management techniques such as approximate nearest neighbor algorithms or reservoir sampling, to handle the continuous influx of data.
- Strategies to detect and adapt to concept drift, where the underlying patterns in the data change over time, should also be explored.
- Exploring methods to handle repeated rows in the streaming data and avoid redundant processing is a vital step to be addressed.
- Further exploring the potential of incorporating external verified labels and leveraging reports for online learning.

Data Augmentation Considerations: The data augmentation experiment aimed to enhance IntelliTweet's understanding of malicious tweet features by creating synthetic malicious tweets from reported content. While the augmented data showed some improvements in precision and F1-score compared to the initial data, the differences were relatively small. Additionally, the inclusion of sentiment analysis did not significantly enhance the performance further. Although the augmentation step showed potential, relying solely on reports for augmentation may not be efficient or provide substantial value. The similarities in performance metrics across the experiments as summarized in Figure 5.11 indicate that the augmented data did not significantly improve IntelliTweet's ability to detect malicious tweets compared to the initial data, suggesting the need for additional strategies beyond report augmentation to achieve substantial enhancements in model performance.

Future Improvements:

- Future work should explore more robust augmentation techniques, such as leveraging generative models or sampling techniques on the specific task of generating synthetic

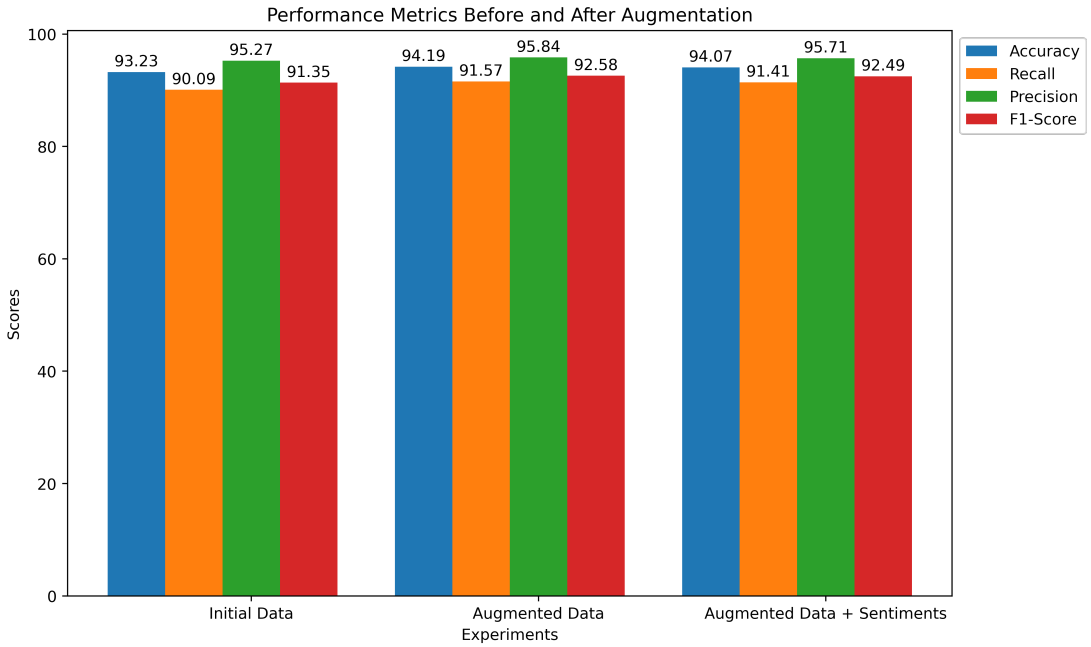


Figure 5.11: Metrics showing the performance of systems before and after augmentation

texts of malicious tweets.

- Exploring alternative augmentation techniques that complement the insights gained from reports could prove useful.

The cascade system relied on a majority voting mechanism across three compartments including URL Detection, Intent Recognition, and User Analysis. While individual classifiers demonstrated satisfactory performance in classifying tweets during training, the system's overall effectiveness was compromised by an oversimplified majority voting approach that failed to consider the varying importance of each compartment's prediction across different scenarios.

Failures and Future Improvement:

- Despite promising results in individual components, a simplification of the prediction process for a more straightforward and effective solution was lacking.
- Equal weighting across compartments proved impractical.
- This experience highlighted the need to explore some ensemble techniques with weighted voting mechanisms or alternative approaches to integrate multiple compartments effectively that will simplify the prediction process to avoid confusion and enhance system performance.

While the results of these initial experiments were not conclusive enough for immediate integration into IntelliTweet, they provided valuable insights and guidance for future research directions. Addressing the identified challenges, refining the proposed techniques, and exploring new avenues for enhancing IntelliTweet's performance will be crucial in developing a robust and adaptive system capable of mitigating emerging threats on the Twitter platform. As we navigate the complexities of enhancing IntelliTweet, these insights serve as guiding principles for refining its capabilities and ensuring robust performance in real-world scenarios.

Chapter 6

Conclusions

In this thesis, we present the culmination of our research efforts in addressing the evolving threat landscape on Twitter. The pervasive threat of phishing, scams, and spam across digital channels necessitates innovative detection mechanisms to safeguard individuals and organizations. Social media platforms like Twitter, with their global reach and real-time nature, have become prime targets for malicious activities. In response to these challenges, researchers have developed various approaches to detect and mitigate malicious tweets.

We present IntelliTweet, an innovative system leveraging machine learning and natural language processing to enhance the detection of malicious tweets on Twitter. Driven by the limitations in existing threat detection mechanisms, IntelliTweet embraces a multifaceted approach across users, content, and context to accurately unveil threats in real-time.

Our research began with a review of existing literature, which shed light on the limitations of current detection methods and emphasized the necessity for a more nuanced approach.

6.1 Research Core Contributions

In this thesis, we provided the following main contributions:

- **Feature Enhancement:** Our IntelliTweet system introduces a holistic feature engineering methodology that considers the nuances of tweet context and content. By integrating multiple features including text analysis, user metadata, URL attributes, and sentiment analysis, IntelliTweet offers a multifaceted approach to malicious tweet detection.
- **Domain-Specific Dataset:** Our work leverages a meticulously curated real-time dataset sourced directly from Twitter, encompassing a diverse array of malicious, legitimate, and user-reported tweets. This dataset serves as a valuable resource for future research endeavors in the field.
- **Publication and Accessibility:** Our findings have been disseminated through a peer-reviewed conference paper, contributing to the academic discourse on Twitter security. Additionally, we have made our codebase and datasets publicly available on GitHub [20], promoting transparency, reproducibility, and collaboration within the research community.
- **False Alarm Reduction:** IntelliTweet prioritizes the reduction of false positives, mitigating the real-world implications of misclassifying benign users. By incorporating user-reported tweets and authenticating user reports, IntelliTweet minimizes the mislabeling of legitimate awareness-raising content.

Overall, our research endeavors have resulted in several core contributions that fill in some research gaps to advance the state-of-the-art, setting a step for further advancements in cybersecurity research.

6.2 Some Research Ideas, Recommendations, and Opportunities

Our exploration of malicious tweet detection on Twitter has unveiled some avenues we would like to recommend to other researchers:

- **Contextual Cues:** Further research into understanding the broader context surrounding tweets is recommended. Examining factors like user behaviors, account automation levels, temporal patterns, and network interactions can provide valuable signals related to emerging bot activity and potential threats enabling the identification of evolving malicious tactics and patterns.
- **Integration of Sentiment Analysis:** Incorporating sentiment analysis into malicious tweet detection offers opportunities to understand the interplay between text sentiment and spam probability.
- **User Engagement Analysis:** Exploring the impact of user interactions with malicious tweet reports on threat mitigation can inform strategies for enhancing user engagement and response mechanisms.
- **Ethical Considerations:** As researchers collect and analyze user-generated content on social media platforms, addressing ethical considerations regarding user privacy and consent is recommended for maintaining trust and integrity.

6.3 Future Work

In our future research endeavors, we will focus on several key areas to further enhance IntelliTweet's capabilities and address emerging challenges in malicious tweet detection.

Firstly, limitations in data collection and potential biases underscores the imperative to diversify assessments on a large scale to effectively address evolving real-world threats. Future endeavors will address these concerns and be benchmarked against recent advancements in the field.

Secondly, we recognize the importance of extending malicious tweet detection capabilities beyond English-language threats. We aim to explore linguistic contexts and extend our approach to other languages, providing comprehensive protection against global malicious activities.

Moreover, we plan to conduct real-world deployment and testing of IntelliTweet to gain insights into its performance over time and its impact on mitigating malicious activities on Twitter. This real-world deployment will help validate IntelliTweet's effectiveness and identify areas for further improvement.

Finally, we intend to complete the implementation of the Online Learning component that we initiated. This will enable IntelliTweet to adapt in real time to evolving threats and dynamic social media environments. Although we encountered challenges with the Online Learning setup our documented outcomes provided valuable insights for future iterations. Moving forward, we will prioritize addressing the issues encountered and optimizing these configurations to enhance IntelliTweet's capabilities.

6.4 IntelliTweet Summary

Comparative experiments highlight IntelliTweet's capabilities in countering URL obfuscation and significantly reducing false positives. The system demonstrates consistent performance across key metrics like 98.80% precision, 98.15% F1-measure, and 0.07 false positive rates based on real-world Twitter data. The consideration of user reports further empowers IntelliTweet to avoid mislabeling legitimate awareness-raising content.

Overall, IntelliTweet represents a vital step toward enhanced Twitter security and positive user experiences, providing valuable insights for mitigating emerging threats on the platform.

References

[1] Anupama Aggarwal, Ashwin Rajadesingan, and Ponnurangam Kumaraguru. Phishari: Automatic realtime phishing detection on twitter. In *2012 eCrime Researchers Summit*, pages 1–12, 2012. doi:10.1109/eCrime.2012.6489521.

[2] Md Tofael Ahmed, Mariam Akter, Md Saifur Rahman, Maqsudur Rahman, Pintu Chandra Paul, Miss Parvin, Almas Hossain Antar, et al. Deep neural network based spam email classification using attention mechanisms. *Journal of Intelligent Learning Systems and Applications*, 15(04):pages 144–164, 2023.

[3] Zulfikar Alom, Barbara Carminati, and Elena Ferrari. A deep learning model for twitter spam detection. *Online Social Networks and Media*, 18:100079, 2020. URL: https://www.sciencedirect.com/science/article/pii/S2468696420300203, doi:10.1016/j.osnem.2020.100079.

[4] Yazan Alshboul, Raj Nepali, and Yong Wang. Detecting malicious short URLs on Twitter. *AMCIS 2015 Proceedings*, June 2015. URL: https://aisel.aisnet.org/amcis2015/ISSecurity/GeneralPresentations/19.

[5] Anon. Twitter usage statistics - internet live stats 2022. Available online: https://www.internetlivestats.com/twitter-statistics/. Accessed on 29 December 2023.

[6] M Arivukarasi, A Manju, R Kaladevi, Shanmugasundaram Hariharan, M Mahasree, and Andraju Bhanu Prasad. Efficient phishing detection and prevention using support vector machine (SVM) algorithm. In *2023 IEEE 12th International Conference on Communication Systems and Network Technologies (CSNT)*, pages 545–548, 2023. doi:10.1109/CSNT57126.2023.10134735.

[7] Nureni Ayofe Azeez, Oluwadamilola Atiku, Sanjay Misra, Adewole Adewumi, Ravin Ahuja, and Robertas Damasevicius. Detection of malicious urls on twitter. In *Advances in Electrical and Computer Technologies: Select Proceedings of ICAECT 2019*, pages 309–318. Springer, 2020.

[8] Nureni Ayofe Azeez, Sanjay Misra, Ihotu Agbo Margaret, Luis Fernandez-Sanz, et al. Adopting automated whitelist approach for detecting phishing attacks. *Computers & Security*, 108:102328, 2021.

[9] Ram B Basnet, Andrew H Sung, and Quingzhong Liu. Rule-based phishing attack detection. In *International conference on security and management, Las Vegas, NV*, 2012.

[10] Amine Belabed, Esma Aïmeur, and Amine Chikh. A personalized whitelist approach for phishing webpage detection. In *2012 Seventh International Conference on Availability, Reliability and Security*, pages 249–254, 2012. doi:10.1109/ARES.2012.54.

[11] Simon Bell, Kenny Paterson, and Lorenzo Cavallaro. Catch me (on time) if you can: Understanding the effectiveness of twitter url blacklists. *arXiv preprint arXiv:1912.02520*, 2019. arXiv:1912.02520.

[12] András A Benczúr, Levente Kocsis, and Róbert Pálovics. Online machine learning in big data streams. *arXiv preprint arXiv:1802.05872*, 2018. arXiv:1802.05872.

[13] Habiba BOUIJIJ and Amine BERQIA. Machine learning algorithms evaluation for phishing urls classification. In *2021 4th International Symposium on Advanced Electrical and Communication Technologies (ISAECT)*, pages 01–05, 2021. doi:10.1109/ISAECT53699.2021.9668489.

[14] Jian Cao, Qiang Li, Yuede Ji, Yukun He, and Dong Guo. Detection of forwarding-based malicious urls in online social networks. *International Journal of Parallel Programming*, 44:163–180, 2016. doi:10.1007/s10766-014-0330-9.

[15] Olivier de Casanove and Florence Sèdes. Malicious human behaviour in information system security: Contribution to a threat model for event detection algorithms. In *Foundations and Practice of Security*, pages 208–220, Cham, 2023. Springer Nature Switzerland.

[16] Ogunmodimu Dupe Catherine. An intelligent rule based phishing website detection model. *Iiardpub. Org*, 5(2):10–19, 2019.

[17] Chao Chen, Jun Zhang, Xiao Chen, Yang Xiang, and Wanlei Zhou. 6 million spam tweets: A large ground truth for timely twitter spam detection. In *2015 IEEE International Conference on Communications (ICC)*, pages 7065–7070, 2015. doi:10.1109/ICC.2015.7249453.

[18] Federico Concone, Giuseppe Lo Re, Marco Morana, and Claudio Ruocco. Assisted labeling for spam account detection on twitter. In *2019 IEEE International Conference on Smart Computing (SMARTCOMP)*, pages 359–366. IEEE, 2019.

[19] Kamel Ahsene Djaballah, Kamel Boukhalfa, Zakaria Ghalem, and Oussama Boukerma. A new approach for the detection and analysis of phishing in social networks: the case of twitter. In *2020 Seventh International Conference on Social Networks Analysis, Management and Security (SNAMS)*, pages 1–8, 2020. doi:10.1109/SNAMS52053.2020.9336572.

[20] Dzeha, Jourdan Eric Edem, and Guy-Vincent. eric-edem/The_intellitweet: A Multifaceted Feature Approach to Detect Malicious Tweets. URL: https://github.com/eric-edem/The_IntelliTweet.

[21] Eric Edem Dzeha and Guy-Vincent Jourdan. Intellitweet: A multifaceted feature approach to detect malicious tweets. In Mohamed Mosbah, Florence Sèdes, Nadia Tawbi, Toufik Ahmed, Nora Boulahia-Cuppens, and Joaquin Garcia-Alfaro, editors, *Foundations and Practice of Security*, pages 157–173, Cham, 2024. Springer Nature Switzerland. doi:10.1007/978-3-031-57537-2_10.

[22] Federal Trade Commission (FTC). Consumer sentinel network data book, February 2022. URL: https://www.ftc.gov/system/files/ftc_gov/pdf/CSN%20Annual%20Data%20Book%202021%20Final%20PDF.pdf.

[23] Razan Ghanem and Hasan Erbay. Spam detection on social networks using deep contextualized word representation. *Multimedia Tools and Applications*, 82(3):3697–3712, 2023.

[24] Argha Ghosh and A Senthilrajan. Comparison of machine learning techniques for spam detection. *Multimedia Tools and Applications*, pages 1–28, 2023.

[25] Anti-Phishing Working Group. Phishing activity trends reports. Available online: https://docs.apwg.org/reports/apwg_trends_report_q2_2023.pdf. Accessed on 27 December 2023.

[26] Yanhui Guo, Zelal Mustafaoglu, and Deepika Koundal. Spam detection using bidirectional transformers and machine learning classifier algorithms. *Journal of Computational and Cognitive Engineering*, 2(1):5–9, 2023.

[27] Brij B Gupta, Nalin AG Arachchilage, and Kostas E Psannis. Defending against phishing attacks: taxonomy of methods, current issues and future directions. *Telecommunication Systems*, 67:247–267, 2018.

[28] Jiwon Hong, Taeri Kim, Jing Liu, Noseong Park, and Sang-Wook Kim. Phishing url detection with lexical features and blacklisted domains. *Adaptive autonomous secure cyber systems*, pages 253–267, 2020.

[29] Sameera Horawalavithana, Ravindu De Silva, Mohamed Nabeel, Charitha Elvitigala, Primal Wijesekara, and Adriana Iamnitchi. Malicious and Low Credibility URLs on Twitter during the AstraZeneca COVID-19 Vaccine Development, February 2021. arXiv:2102.12223 [cs] version: 1. URL: http://arxiv.org/abs/2102.12223.

[30] IBM. What is phishing? Available online: https://www.ibm.com/topics/phishing. Accessed on 29 December 2023.

[31] Isa Inuwa-Dutse, Mark Liptrott, and Ioannis Korkontzelos. Detection of spam-posting accounts on twitter. *Neurocomputing*, 315:496–511, 2018.

[32] Ankit Kumar Jain and Brij B Gupta. Towards detection of phishing websites on client-side using machine learning based approach. *Telecommunication Systems*, 68:687–700, 2018.

[33] Amelia M Jamison, David A Broniatowski, and Sandra Crouse Quinn. Malicious actors on twitter: A guide for public health researchers. *American journal of public health*, 109(5):688–692, 2019.

[34] Francisco Jáñez-Martino, Rocío Alaiz-Rodríguez, Víctor González-Castro, Eduardo Fidalgo, and Enrique Alegre. Classifying spam emails using agglomerative hierarchical clustering and a topic-based approach. *Applied Soft Computing*, 139:110226, 2023.

[35] Se Yeong Jeong, Yun Sing Koh, and Gillian Dobbie. Phishing detection on twitter streams. In *Trends and Applications in Knowledge Discovery and Data Mining: PAKDD 2016 Workshops, BDM, MLSDA, PACC, WDMBF, Auckland, New Zealand, April 19, 2016, Revised Selected Papers 20*, pages 141–153. Springer, 2016.

[36] Amir Karami, Morgan Lundy, Frank Webb, and Yogesh K. Dwivedi. Twitter and research: A systematic literature review through text mining. *IEEE Access*, 8:67698–67717, 2020. doi:10.1109/ACCESS.2020.2983656.

[37] Mahmoud Khonji, Youssef Iraqi, and Andrew Jones. Phishing detection: A literature survey. *IEEE Communications Surveys & Tutorials*, 15(4):2091–2121, 2013. doi:10.1109/SURV.2013.032213.00009.

[38] Yutaro Kokubun and Akihito Nakamura. Analysis of malicious urls on twitter. In *2018 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 1285–1288, 2018. doi:10.1109/CSCI46756.2018.00248.

[39] Mehmet Korkmaz, Ozgur Koray Sahingoz, and Banu Diri. Detection of phishing websites by using machine learning-based url analysis. In *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pages 1–7, 2020. doi:10.1109/ICCCNT49239.2020.9225561.

[40] Sanjay Kumar, Azfar Faizan, Ari Viinikainen, and Timo Hamalainen. Mlspd - machine learning based spam and phishing detection. In *Computational Data and Social Networks*, pages 510–522, Cham, 2018. Springer International Publishing.

[41] Sangho Lee and Jong Kim. Warningbird: A near real-time detection system for suspicious urls in twitter stream. *IEEE transactions on dependable and secure computing*, 10(3):183–195, 2013.

[42] Sreekanth Madisetty and Maunendra Sankar Desarkar. A neural network-based ensemble approach for spam detection in twitter. *IEEE Transactions on Computational Social Systems*, 5(4):973–984, 2018. doi:10.1109/TCSS.2018.2878852.

[43] Samuel Marchal, Kalle Saari, Nidhi Singh, and N. Asokan. Know your phish: Novel techniques for detecting phishing sites and their targets. In *2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS)*, pages 323–333, 2016. doi:10.1109/ICDCS.2016.10.

[44] Rami Mohammad and Lee McCluskey. Uci machine learning repository. Available online: https://archive.ics.uci.edu/dataset/327/phishing+websites. Accessed on 29 April 2022.

[45] Hiroki Nakano, Daiki Chiba, Takashi Koide, Naoki Fukushi, Takeshi Yagi, Takeo Hariu, Katsunari Yoshioka, and Tsutomu Matsumoto. Canary in twitter

mine: Collecting phishing reports from experts and non-experts. *arXiv preprint arXiv:2303.15847*, 2023.

[46] Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. BERTweet: A pre-trained language model for English tweets. In Qun Liu and David Schlangen, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 9–14, Online, October 2020. Association for Computational Linguistics. URL: https://aclanthology.org/2020.emnlp-demos.2, doi:10.18653/v1/2020.emnlp-demos.2.

[47] Jason R. C. Nurse. Cybercrime and you: How criminals attack and the human factors that they seek to exploit, October 2018. URL: http://dx.doi.org/10.1093/oxfordhb/9780198812746.013.35, doi:10.1093/oxfordhb/9780198812746.013.35.

[48] Routhu Srinivasa Rao, Tatti Vaishnavi, and Alwyn Roshan Pais. Catchphish: detection of phishing websites by inspecting urls. *Journal of Ambient Intelligence and Humanized Computing*, 11:813–825, 2020.

[49] Anisha P Rodrigues, Roshan Fernandes, Adarsh Shetty, Kuruva Lakshmanna, R Mahammad Shafi, et al. Real-time twitter spam detection and sentiment analysis using machine learning and deep learning techniques. *Computational Intelligence and Neuroscience*, 2022.

[50] Sayak Saha Roy, Unique Karanjit, and Shirin Nilizadeh. Evaluating the effectiveness of phishing reports on twitter. In *2021 APWG Symposium on Electronic Crime Research (eCrime)*, pages 1–13, 2021. doi:10.1109/eCrime54498.2021.9738786.

[51] Asadullah Safi and Satwinder Singh. A systematic literature review on phishing website detection techniques. *Journal of King Saud University - Computer and Information Sciences*, 35(2):590–611, 2023. URL: https://www.sciencedirect.com/science/article/pii/S1319157823000034, doi:10.1016/j.jksuci.2023.01.004.

[52] Maria Sameen, Kyunghyun Han, and Seong Oun Hwang. Phishhaven—an efficient real-time ai phishing urls detection system. *IEEE Access*, 8:83425–83443, 2020. doi:10.1109/ACCESS.2020.2991403.

[53] Nilesh Sharma, Nishant Sharma, Vishakha Tiwari, Shweta Chahar, et al. Real-time detection of phishing tweets. In *4th Int Conf Comput Sci Eng Appl*, pages 215–27, 2014.

[54] Siyuan Tang, Xianghang Mi, Ying Li, XiaoFeng Wang, and Kai Chen. Clues in tweets: Twitter-guided discovery and analysis of sms spam. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 2751–2764, 2022.

[55] Twitter. About unsafe links. Available online: https://help.twitter.com/en/safety-and-security/phishing-spam-and-malware-links. Accessed on 28 December 2023.

[56] Twitter. Counting characters. Available online: https://developer.twitter.com/en/docs/counting-characters. Accessed on 28 December 2023.

[57] Wisdom Umeugo. Cybercrime awareness on social media: A comparison study. *International Journal of Network Security & Its Applications*, 15(2):23–35, 2023.

[58] VirusTotal. Api overview. Available online: https://docs.virustotal.com/docs/api-overview. Accessed on 6 April 2022.

[59] Alex Hai Wang. Don't follow me: Spam detection in twitter. In *2010 International Conference on Security and Cryptography (SECRYPT)*, pages 1–10, 2010.

[60] Kalpana Wani, Aditya Patil, Sumedha Mukherjee, and Sushmita Sarkar. Malicious twitter bot detector. In *2021 4th Biennial International Conference on Nascent Technologies in Engineering (ICNTE)*, pages 1–6, 2021. doi:10.1109/ICNTE51185.2021.9487674.

[61] Yue Zhang, Jason I Hong, and Lorrie F Cranor. Cantina: a content-based approach to detecting phishing web sites. In *Proceedings of the 16th international conference on World Wide Web*, pages 639–648, 2007.

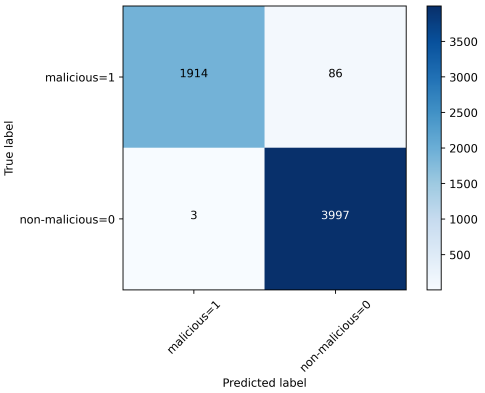
APPENDICES

Appendix A

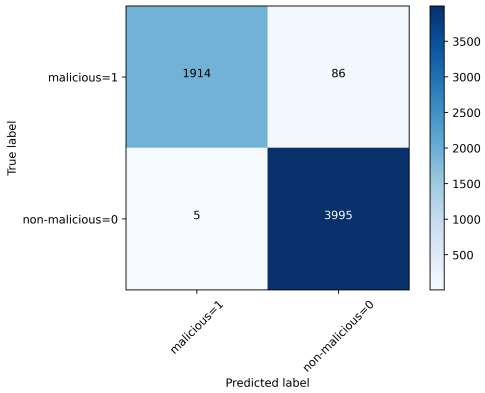
All Additional Images, Tables and plots

A.1 Confusion Matrices for Binary System IntelliTweet and TextAnalyst Evaluation

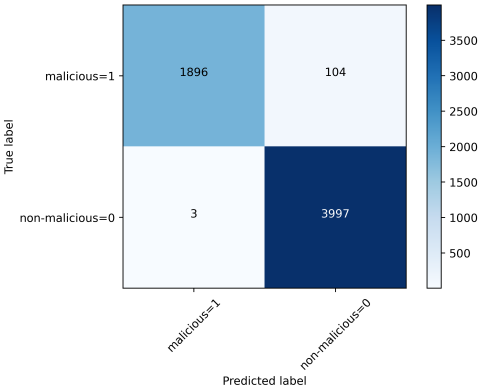
The confusion matrices in Section A.1 below summarize the results of all the individual iterations of the stratified 5-fold CV for the Binary System TextAnalyst On and Off experiment.



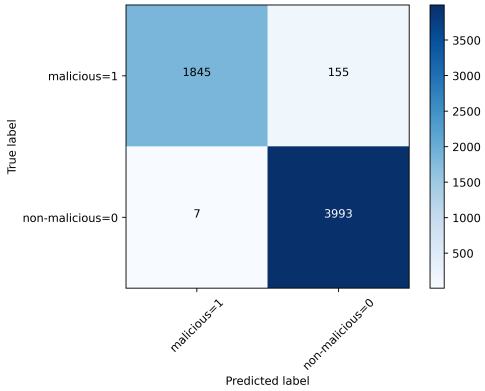
(a) Iteration 1 - TextAnallyst On



(b) Iteration 1 - TextAnallyst Off

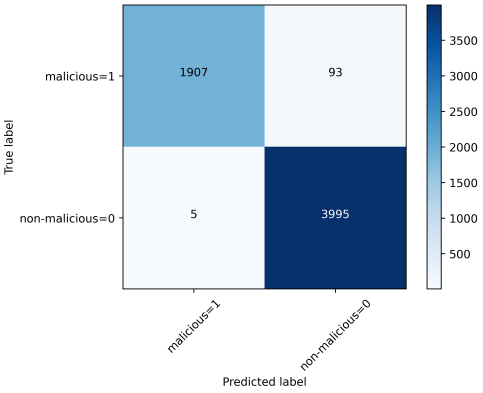


(c) Iteration 2 - TextAnallyst On

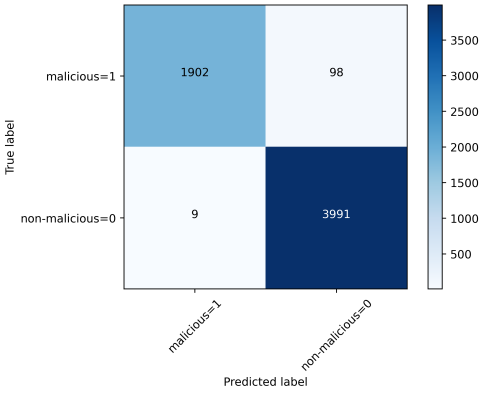


(d) Iteration 2 - TextAnallyst Off

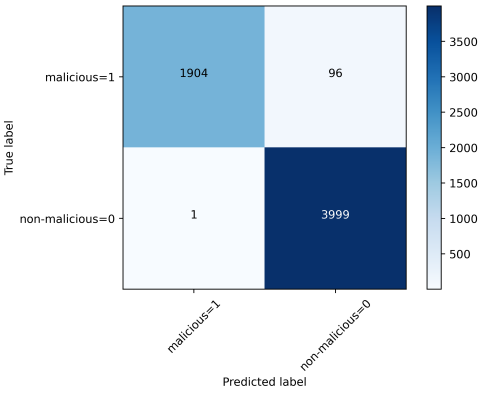
Figure A.1: Confusion Matrix Images for TextAnallyst On and Off



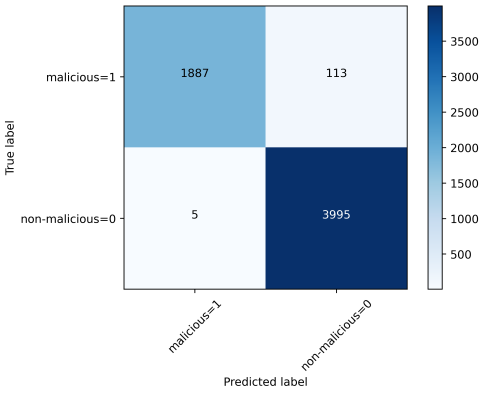
(e) Iteration 3 - TextAnallyst On



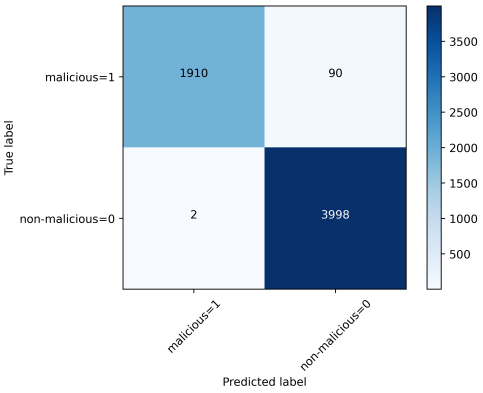
(f) Iteration 3 - TextAnallyst Off



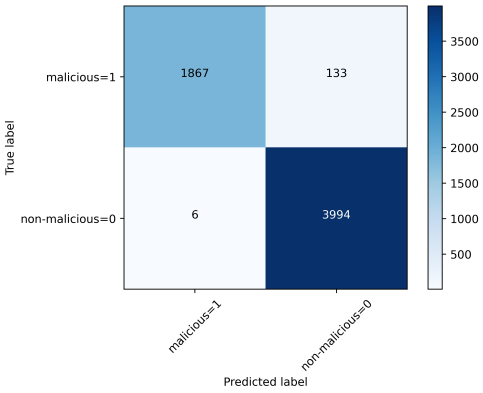
(g) Iteration 4 - TextAnallyst On



(h) Iteration 4 - TextAnallyst Off



(i) Iteration 5 - TextAnallyst On

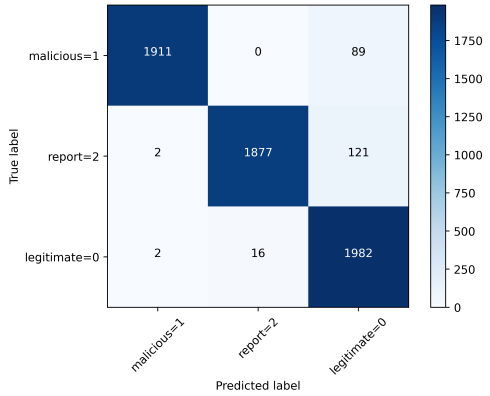


(j) Iteration 5 - TextAnallyst Off

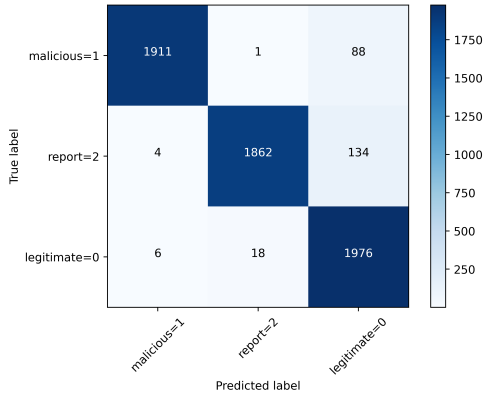
Figure A.1: Confusion Matrix Images for TextAnallyst On and Off

A.2 Multi System Confusion Matrices for TextAnalyst Evaluation

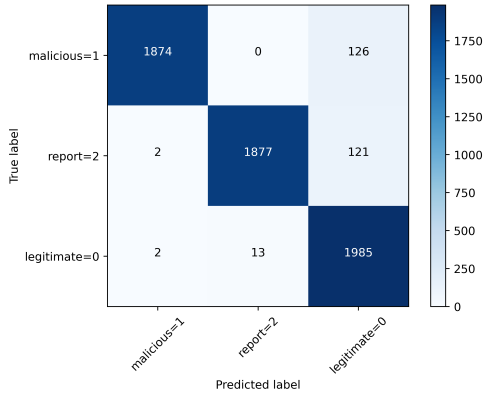
The confusion matrices in Section A.2 highlight the results of each iteration for the Multisystem System TextAnalyst evaluation.



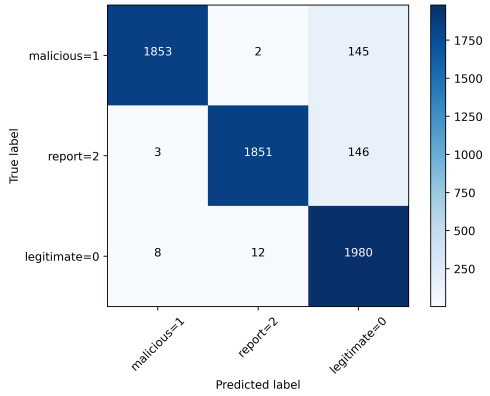
(a) Iteration 1 - TextAnallyst On



(b) Iteration 1 - TextAnallyst Off

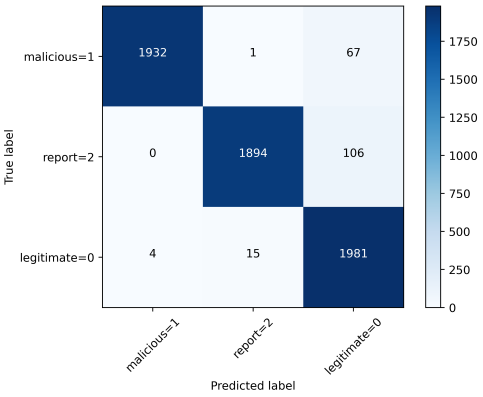


(c) Iteration 2 - TextAnallyst On

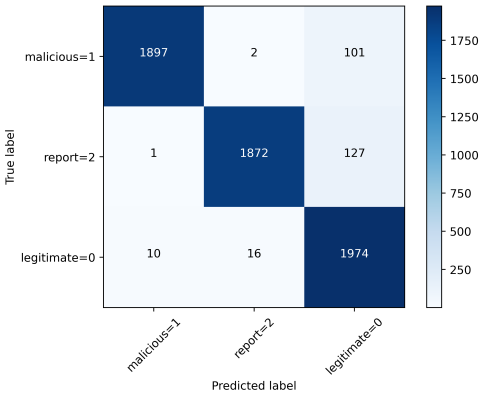


(d) Iteration 2 - TextAnallyst Off

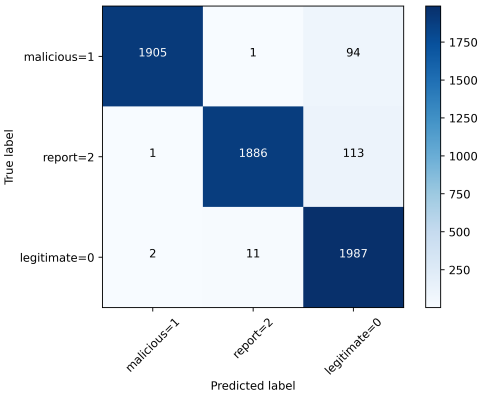
Figure A.2: Confusion Matrix Images for TextAnallyst On and Off



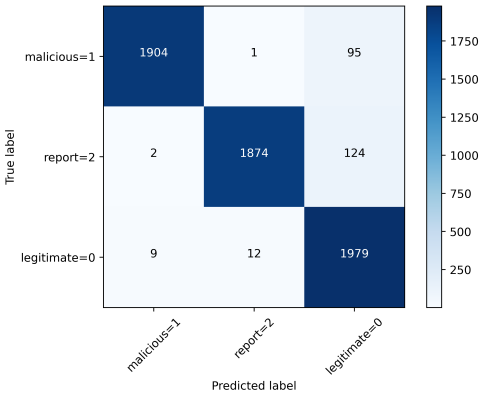
(e) Iteration 3 - TextAnallyst On



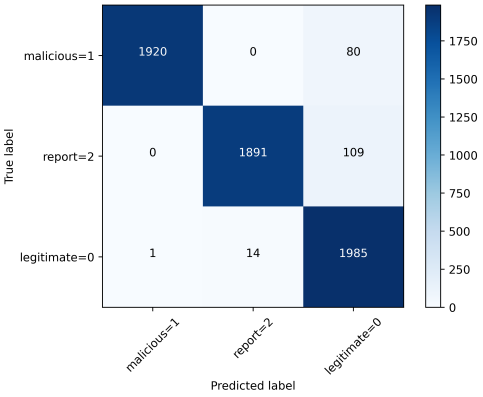
(f) Iteration 3 - TextAnallyst Off



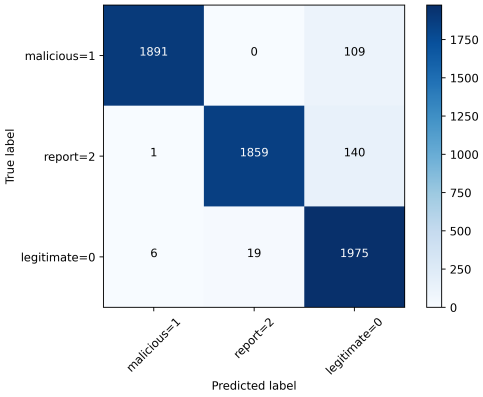
(g) Iteration 4 - TextAnallyst On



(h) Iteration 4 - TextAnallyst Off



(i) Iteration 5 - TextAnallyst On



(j) Iteration 5 - TextAnallyst Off

Figure A.2: Confusion Matrix Images for TextAnallyst On and Off