Journal of Computer Science

Review

Detection of Phishing Websites Hosted in Name Server Flux Networks Using Machine Learning

Thomas Nagunwa

School of Computing and Digital Technology, Birmingham City University, Birmingham, United Kingdom

Article history
Received: 28-02-2023
Revised: 13-08-2023
Accepted: 04-09-2023

Email: tom.nag@gmail.com

Abstract: Attackers are increasingly using Name Server IP Flux Networks (NSIFNs) to run the domain name services of their phishing websites in order to extend the duration of their phishing operations. These networks host a name server that manages the Domain Name System (DNS) records of the websites on a network of compromised machines with frequently changing IP addresses. As a result, blacklisting the machines has less impact on stopping the services, lengthening their lifespan and that of the websites they support. High detection delays and the use of fewer, lesser varied detection features limit the proposed solutions for identifying the websites hosted in these networks, making them more susceptible to detection evasions. This study suggests a novel set of highly diverse features based on DNS, network, and host behaviors for fast and highly accurate detection of phishing websites hosted in NSIFNs using a Machine Learning (ML) approach. Using a variety of traditional and deep learning ML algorithms, the prediction performance of our features was assessed in the context of binary and multi-class classification tasks. Our approach achieved optimal accuracy rates of 98.59% and 90.41% for the binary and multi-class classification tasks, respectively. Our approach is a crucial step toward monitoring NSIFN components to mitigate phishing attacks efficiently.

Keywords: Phishing Hostname, Name Server Flux Network, Machine Learning, Deep Learning, Flat and Hierarchical Classification

Introduction

Attackers commonly use legitimate or their own authoritative Name Servers (NSs) to keep DNS records of their malicious websites. These frequently consist of a single machine or a small network of machines with consistent IP addresses. This makes it simple to identify servers by their IP addresses and remove or blacklist malicious NS records to effectively shut down the services. Attackers are increasingly hosting the records on NSIFNs to avoid being blacklisted. In NSIFNs, the attacker-controlled authoritative NS (also referred to as mothership) maintains DNS records for malicious websites. The attacker compromises and manages vulnerable machines (also known as flux agents) from different networks through the NS, which also serves as the network's command and control centre and utilizes them as proxies for user DNS requests. Blacklisting the machines does not effectively stop the domain name services because they are often replaced as new machines are compromised. As a result, the usage of NSIFNs makes it impossible to shut down the services, which makes it possible for the malicious websites' operations to continue and become more impactful. In order to effectively combat persistent malicious websites, an approach for detecting the websites hosted in these networks must be effective and efficient.

This study aims to detect phishing-specific websites/hostnames hosted in NSIFNs (referred to in this study as phishing NS flux hostnames). This is due to the fact that phishing attacks are the leading cause of cybersecurity attacks worldwide. They are responsible for up to 91% of all data breaches worldwide (Sophos, 2017). The number of global data breach incidents will be significantly reduced if users are prevented from visiting phishing websites. Furthermore, according to Caglayan *et al.* (2010), flux networks hosting various malicious web services, such as spam, malware, and phishing, differ significantly from one another in terms of DNS, host, and network-related characteristics, which are often used as a source of features for the detection. As a result, a solution that aims to detect all types of malicious web services hosted in an NSIFN is likely going to be less efficient than one that targets a particular web service in a dedicated NSIFN.

Building a blacklist of phishing NS flux hostnames to make it easier to monitor networks over the long term for their effective takedown is one of the potential uses of our approach. The blacklist can specifically be used by cybersecurity stakeholders such as Internet Service Providers (ISPs) and government organizations to continuously query NSs forming flux networks hosting phishing hostnames in order to investigate and monitor them. This will result in the identification of the legitimate networks that are hosting the NS flux agents that have been compromised, information that can be used to alert the network owners to find the compromised machines in their networks, clean them up, and take security measures to prevent the machines from becoming compromised again. Additionally, solutions such as Gu *et al*. (2007); Khattak *et al*. (2015) can be used at network gateways to track the motherships in order to blacklist them, thereby shutting down the entire infrastructure of the phishing campaigns. These solutions do this by monitoring data traffic between flux agents in the local networks and their external motherships.

A few studies have proposed approaches for detecting malicious NS flux hostnames. For example, Kadir *et al*. (2012); Pa *et al*. (2015) tracked DNS-related traits of legitimate and malicious websites and their hosting networks for up to several months to find detection patterns. However, they are constrained by the following:

- They rely solely on detection features that are based on DNS-related traits. Attackers may be able to evade detection by eluding the features using simple evasion techniques
- Their monitoring (detection) period, which ranges from 3-6 months, is notably long for real-time detection. The time gives the malicious NSs the opportunity to keep giving malicious websites DNS services and do more harm before being detected
- Their approaches detect all types of malicious hostnames hosted in NSIFNs. These are less likely to be able to detect hostnames that are phishing-specific, as was already mentioned
- Their detection abilities were not thoroughly validated by using a variety of performance metrics, such as precision, recall, and false negative rates. This limits our ability to assess the techniques' overall efficacy

We propose a fast, highly accurate, and more detection evasion-resistant ML-based approach to detect phishing-specific NS flux hostnames to address the aforementioned shortcomings. The following are our contributions to this study:

1. Our approach is designed to effectively detect phishing NS flux hostnames using a novel set of highly diverse features. The set consists of 11 features across five distinct feature categories, all of which were derived from hosting networks' DNS, host, and network characteristics. This study is the first to propose all of the features
2. The problem is formulated as both binary and multi-class classification tasks to distinguish between legitimate and legitimate NS non-flux hostnames and phishing NS flux hostnames. The phishing NS flux hostnames are distinguished from the other hostnames combined into a single hostname class in the binary classification. In the multi-class classification, each of the three hostnames is identified separately, allowing the precise hostname type to be determined and providing more information to users for decision-making
3. Using flat and hierarchical classification techniques, three implementation architectures of our prediction model are proposed with the goal of determining the architecture that offers the best prediction performance
4. Compared to the related works, we use a larger number of different ML algorithms to evaluate the performance of the features. Conclusions about the overall efficacy of a feature set can be made by comparing the performance results from such a vast array of algorithms. Additionally, the performance was measured and reported using a larger variety of metrics to inform us of the all-around efficacy of the prediction model

To our knowledge, this is the first study to analyze the problem as a three-class classification task, compare flat and hierarchical classification techniques, apply Deep Learning (DL) algorithms, and evaluate the features using both traditional ML and DL algorithms.

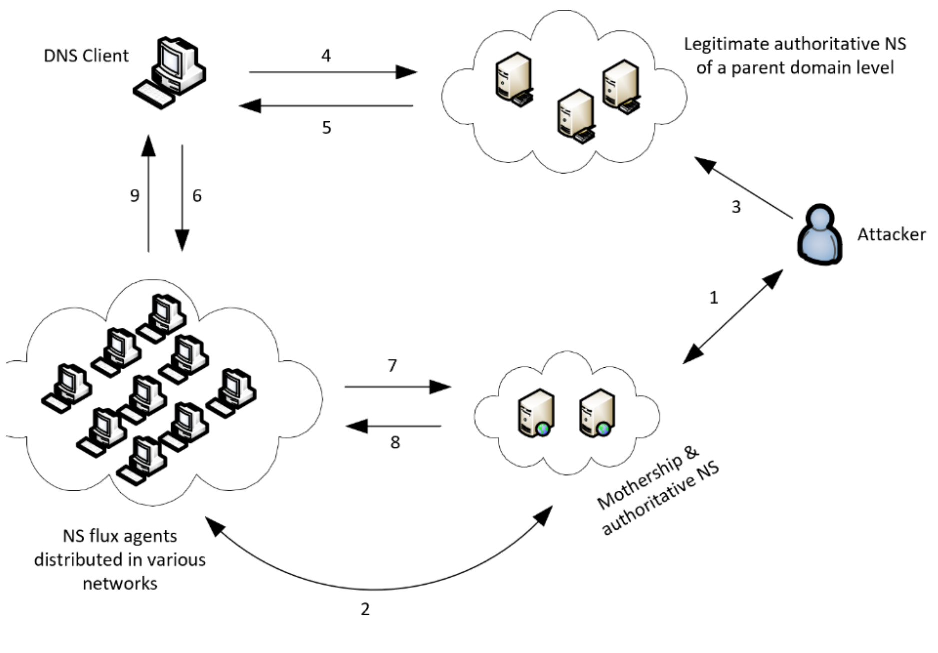
Background of Name Server Flux and Non-Flux Networks

According to studies by Salusky and Danford (2008); Konte *et al*. (2009); Kadir *et al*. (2012); Metcalf and Spring (2013); Pa *et al*. (2015), some of the authoritative NSs registering RRs of malicious web services display a fluxing behaviour in which their IP addresses change quickly. This phenomenon is known as NS IP Flux (NSIF). In NSIF Networks (NSIFNs), the NS infrastructure is made up of a mothership NS and several NSs that act as the network's flux agents. The latter serves as proxies to the mothership by sending NS queries to the mothership and returning the results to the DNS clients, while the former, which is owned and managed by an attacker, is the actual authoritative NS for one or more zones of malicious web services. The compromised machines, normally from widely dispersed networks on the internet, that are enlisted by Malwa are controlled by the attacker in the mothership are known as NS flux agents. The attacker registers the A records in the legitimate authoritative NS of a parent zone that maps hostnames to IP addresses of NS flux agents. When a DNS client queries a series of NS records, the flux agents are quickly rotated (as illustrated in Fig. 1). Attackers frequently give the NS records a shorter Time to Live (TTL) in order to make this strategy possible. The motherships are rendered invisible to users and the attackers' tracks are obscured from forensic analysis by the use of flux agents as proxies. Therefore, tracking and blacklisting the flux agents does not stop the NSIFN because the mothership continuously compromises and employs new flux agents, prolonging the existence of malicious campaigns. The agents and the mothership are frequently utilized to host multiple malicious NSs and websites, similar to NSs in malicious NS non-flux networks.

Flux agents typically come from a wide variety of distinct networks because the malware infection process for recruiting them is random. The majority of flux agents are compromised home and small office networks' Internet of Things (IoT) devices, which are often less secure and have numerous security flaws. Since a large portion of the flux agents are owned by private individuals and small businesses, their availability may change as equipment is turned off when not in use. Furthermore, when computers are cleaned of malware, agents might be lost from the NSIFN.

Table 1 summarizes the expected operational and structural variations between the three categories of networks hosting authoritative NSs (i.e., malicious NS flux networks, malicious NS non-flux networks, and legitimate NS non-flux networks) based on the descriptions above. As described in the following sections, features for differentiating hostnames hosted in each of these networks will be derived from these variations.

One of the two popular types of IP flux network services that are often used by attackers to enhance the productivity of their phishing services is NSIFNs. The other one, Fast Flux Network Services (FFSNs), is the most widely used. In FFSNs, attackers use compromised machines to act as proxies for the real web servers hosting phishing websites. These machines are frequently rotated, so blacklisting them only prevents the front end of the networks supported by the blacklisted machines from accessing the websites hosted on the real servers but not the new machines. This in turn prolongs the lifespan of the websites, which harms users more. In our recent work, Nagunwa *et al*. (2022) proposed an ML model to detect the hostnames of the websites hosted in these networks using predictors based on DNS and network characteristics.


DNS Client
4
5
Legitimate authoritative NS of a parent domain level
9
6
3
Attacker
1
7
8
Mothership & authoritative NS
NS flux agents distributed in various networks
2

1. Attacker sets up the mothership and registers DNS records of hostnames he owns in it
2. The mothership recruits and controls the compromised machines as NS flux agents
3. The agents are registered as authoritative NSs of malicious hostnames in the NS of a parent domain level
4. A DNS client queries for A records of an authoritative NS of a malicious hostname
5. The NS of the parent domain level returns A records of some of the agents
6. The client queries for A records of the hostname from the returned flux agents
7. The flux agents forward the query to the mothership
8. The mothership returns the answer to the flux agents
9. The flux agents return the answer to the client

Fig. 1: The NSIFN architecture

Table 1: Main operational and structural differences between the three types of networks hosting authoritative NSs

Key differences	Legitimate NS non-flux networks	Malicious NS flux networks	Malicious NS non-flux networks
Number of A records returned per NS record request	Small	Large	Small
TTL of NS records	Long	Short	Long
Network distribution of NSs	Small/medium number of distinct networks or one network	Large number of distinct networks	Small number of distinct networks or one network
Geographical distribution of NSs	Small distances between NS locations	Large distances between NS locations	Small distances between NS locations
Co-hosted web services	Small/medium to large number of malicious web services that are co-hosted	Large number of malicious web services that are co-hosted	Small to large number of malicious web services that are co-hosted
NS hosts	High performance servers with longer uptimes and the same OS	Standard machines with variety Operating Systems (OSs) and shorter uptimes	Machines with low to medium performance, the same OS, and longer uptime
IP matching with hosts known to host blacklisted malicious websites	There should be no matching	Large IP matching count	Small to large IP matching counts

Related Work

One of the two popular types of IP flux network services that are often used by attackers to enhance the productivity of their phishing services is NSIFNs. The other one, Fast Flux Network Services (FFSNs), is the most widely used. In FFSNs, attackers use compromised machines to act as proxies for the real web servers hosting phishing websites. These machines are frequently rotated, so blacklisting them only prevents the front end of the networks supported by the blacklisted machines from accessing the websites hosted on the real servers but not the new machines. This in turn prolongs the lifespan of the websites, which harms users more. Our recent work, Nagunwa *et al*. (2022), proposed an ML model to detect the hostnames of the websites hosted in these networks using predictors based on DNS and network characteristics.

However, the focus of this study is on the detection of phishing hostnames hosted in NSIFNs. It is worth noting that we found a small number of studies in this domain. The reason could be that this type of flux is not as popular as FFSNs to attackers and therefore to the cybersecurity community. However, research has indicated that NSIFN deployment is likely to rise in the future. Studies on this type of flux can be divided into two categories: (1) Those that investigate the existence and behaviors of NS flux hostnames and their networks and (2) Those that propose approaches for detecting malicious NS flux hostnames. We will discuss them in that order.

The concept of NS IP flux was first defined by Salusky and Danford (2008), who also looked into how they operated and behaved. For at least a month, they kept track of the A and NS records of known malicious hostnames as well as the A records of their NSs. They consequently discovered distinctive characteristics of NSIFNs hosting the hostnames. High IP fluxing rates, IP addresses drawn from larger numbers of unique Autonomous Systems (ASs), and specific malicious activities carried out on the websites being hosted are a few of these. By infecting a honeypot with malware from a flux network, they were able to research how infected machines are recruited and then commanded by their mothership to carry out malicious activities. The study also suggested six various strategies for mitigating flux networks Konte *et al*. (2009) investigated fluxing behaviours of A records of malicious websites and the names and A records of the authoritative NSs for the zones these websites belong to. For a month, they took 3,360 hostnames out of spam emails and checked their DNS records every 5 min. They extracted 500 of the most popular websites from the Alexa ranking and kept track of similar DNS records to compare with the legitimate hostnames. While the NS records of the legitimate websites did not change over time, the study noticed that all three types of DNS records of the malicious websites did. They created 21 clusters of malicious campaigns by comparing how similar the contents of malicious websites were. After that, the study compared a number of dynamic aspects of the infrastructures housing the specific malicious and legitimate hostnames as well as the campaigns. The aspects that were compared included the rate of change of DNS records, the expansion rate of networks, the location of DNS hierarchy changes, and the topological and geographic distribution of hosts. To illustrate the NS IP flux behaviour, Metcalf and Spring (2013) tracked NS records of hostnames from a zone file of common gTLDs and those from passive data from the Security Information Exchange. Once per day, the NS records were queried and their subsequent 28 days records were gathered for examination. The number of changes of A records of the NSs, the number of changes of Asynchronous System Number (ASN) of NSs, and the distribution patterns of TTLs of NS records were all statistically analyzed in the study.

In the second category of the studies, Kadir *et al*. (2012) proposed a k-NN classifier to detect hostnames with single flux (the behaviour in which the A records of a website's hostname change frequently) and double flux (the behaviour in which the A records of a website's hostname and its NS change frequently) using seven features. The features were based on the number of A records for the hostnames and NSs for the zones they are associated with, as well as the rate at which those two records are changing. They gathered their data over the course of 3 months by monitoring DNS records of 500 legitimate and malicious hostnames with known single flux behaviors every 12 h. The results demonstrated that the classifier produced no FPR or FNR. 182 of the 250 malicious hostnames were classified as single flux while 68 were classified as double flux. The study also identified various types of IP addresses to NS mappings. By monitoring A and NS records of 50,030 malicious and legitimate hostnames for 6 months, Pa *et al*. (2015) proposed an IP address and hostname mapping technique to detect double flux hostnames. Three criteria were used to map the records as detection features. Single NS to many IP addresses, single IP address to many NSs, and single IP address to both hostname and NS. Using a threshold of 3 for each of the features to classify the NSs, above 3 being the malicious one and below 3 being the legitimate one, the technique was able to achieve a low FPR of 0.8% in detecting NS flux hostnames. The studies discussed above are tabulated in Table 2.

Detection of Phishing Name Server Flux Hostnames

Design Overview

The proposed approach uses supervised machine learning to train and build a classifier that can distinguish between legitimate and phishing NS non-flux hostnames using features extracted at a single point in time (instantaneous features). We observed the NS fluxing behavior of the set of hostnames used to train the classifier for an extended period of time in order to determine their class labels. The main steps of the approach are shown in Fig. 2 as follows:

Steps 1-2: Label the hostname classes of sets of known phishing and legitimate websites based on their NS fluxing behavior by monitoring the A records of authoritative NSs
Steps 3-4: Extraction of instantaneous feature data. A number of services are queried for each website's hostname and the features are taken out of the data that is returned to create the training dataset
Steps 5-6: A suitable ML algorithm is trained on the training dataset to create a classifier
Steps 7-8: The classifier predicts the class of a new website based on the hostname's instantaneous features

Table 2: A summary of studies related to the detection of malicious hostnames hosted in NSIFNs

Study	Strengths	Weaknesses
Salusky and Danford (2008)	Provided empirical proof of the changing behaviour of the authoritative NSs' IP addresses for the monitored malicious websites Investigated flux agents and identified their key characteristics Demonstrated how vulnerable machines are taken advantage of and enlisted to create NSIFNs. Suggested a number of methods for mitigating flux networks	An experimental approach to detect NSIFN hosted websites was not presented
Konte *et al*. (2009)	Provided empirical data on the changes to malicious websites' A records, as well as the names and A records of the websites' DNSs Examined and outlined different network behaviours, both of NSIFNs and non-NSIFNs, in relation to the types of malicious content they host	An experimental approach to detect NSIFN hosted websites was not presented
Metcalf and Spring (2013)	Provided empirical proof of the changing behaviour of the authoritative NSs' IP addresses for the monitored malicious websites	An experimental methodology to detect websites that are hosted in NSIFNs was not proposed
Kadir *et al*. (2012)	To predict malicious websites hosted in single and double-flux NSIFNs, an ML model was proposed Zero errors were produced by the model	The amount of data used was small (consisting of only 500 websites) High detection time (3 months)
Pa *et al*. (2015)	Proposed a method for detecting malicious websites hosted in double-flux NSIFNs by mapping IP addresses to hostnames Used a sizable dataset Low error rates were achieved by the model	High detection time (6 months)

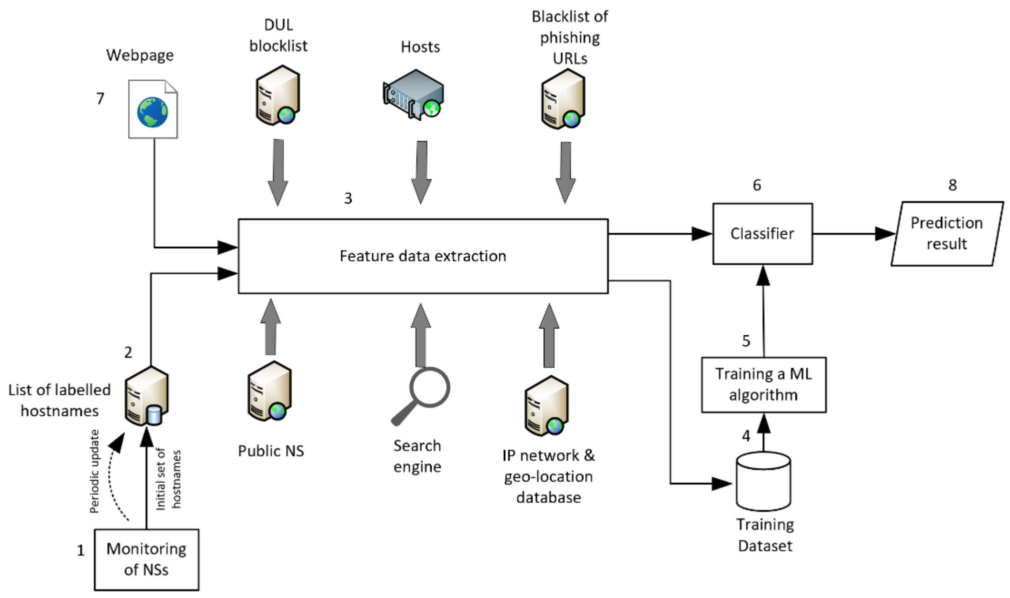


Fig. 2: The architecture of our proposed approach: Building of a classifier (components 2-6), prediction of a new hostname (components 7-8)

Monitoring of Records of NSs for Class Labelling

In order to label hostname classes for the model's training, we monitored of changes in the IP addresses of the authoritative NSs of 6,638 legitimate and 6,630 phishing websites. First, the legitimate and phishing URLs were collected from Tranco's list of the 1 million most popular websites and PhishTank's blacklist, respectively. By requesting NS records from the public NS (we used Google's DNS server), we were able to find authoritative NSs (both primary and secondary) for each URL's hostname. For a period of 3 months (23rd of July to 27th of October 2020), A records for each NS were queried and collected from the public NS every 2 h. Our trial dataset, which showed that the majority of NSs were changing their A records between 2 and 4 h intervals, guided our choice of a 2 h window. The IP addresses returned in the consecutive queries of the same NS were compared and the number of times a change was observed throughout the period was recorded. Figure 3 illustrates this process. Figure 4 depicts the distribution of IP address changes that were noticed during the monitoring period, broken down by NS type. 82.3% of legitimate hostname NSs (hereinafter referred to as legitimate NSs) did not change their IP addresses at all, 17.7% did so only once to five times and none did so more than five times. However, 31.6% of NSs for phishing hostnames (also known as phishing NSs) were found to have their IP addresses changed, with 5.4% changing their IP addresses more than five times. In contrast to legitimate NSs, some phishing NSs displayed numerous changes, with the highest number seen being 826. This study demonstrates that a significant number of phishing websites are hosted in networks deploying NS IP fluxing.

We first had to decide on a threshold number of IP address changes to distinguish between NS flux and non-flux hostnames before we could label the hostname classes for the classification tasks. Metcalf and Spring (2013) assert that legitimate NSs are not intended to display IP fluxing behavior, suggesting that changes in their IP addresses are caused by other factors such as routine server maintenance and network scaling. Due to this, we chose 5 as the threshold, which represents the maximum number of changes in legitimate NS we could find in the data we collected from the NS monitoring task described above. As a result, any phishing hostname that had more than 5 IP changes in NS was classified as a phishing NS flux hostname otherwise it was a phishing NS non-flux hostname. All legitimate hostnames were labeled as legitimate NS non-flux hostnames.

Prediction Features

Based on the differences between the hostname's networks described, we propose 26 predictive features divided into 6 categories that are likely to be helpful in differentiating phishing NS flux hostnames from legitimate and phishing NS non-flux hostnames. The features are summarised in Table 3. The rest of the features are utilized to address this problem for the first time, with the exception of one feature (feature 11), which was used by Kadir *et al.* (2012). Some of the important features of each category are described in the subsections that follow.

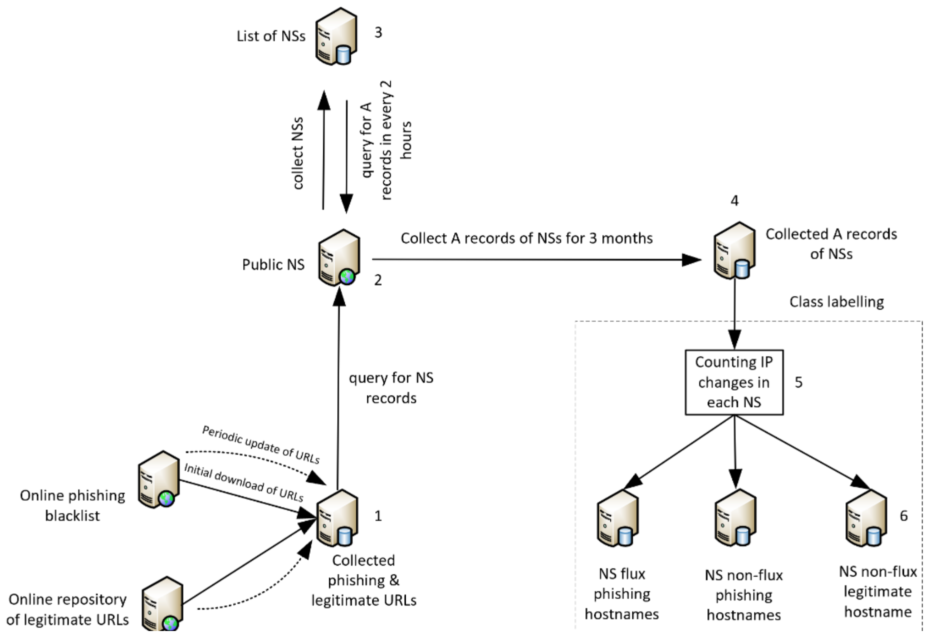

List of NSs
3
collect NSs
query for A records in every 2 hours
4
Collected A records of NSs
Public NS
2
Collect A records of NSs for 3 months
Class labelling
Counting IP changes in each NS
5
query for NS records
Periodic update of URLs
Initial download of URLs
Online phishing blacklist
1
Collected phishing & legitimate URLs
Online repository of legitimate URLs
6
NS flux phishing hostnames
NS non-flux phishing hostnames
NS non-flux legitimate hostname

Fig. 3: Monitoring of A records of NSs for hostname class labelling

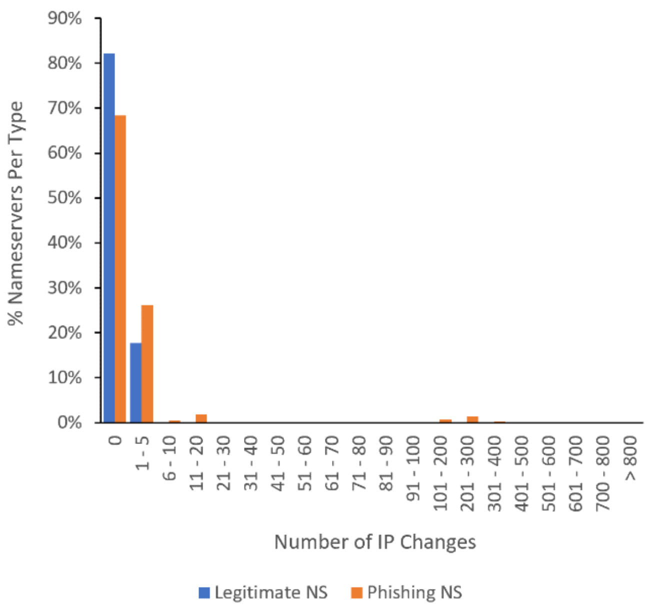


Fig. 4: Distribution of counts observed IP address changes for authoritative NSs by NS type

Temporal Features

Round Trip Time (RTT): This is the average time it takes for each NS to respond with an acknowledgement that corresponds for the DNS query of the hostname of a particular website. A traceroute command is used to calculate the time.

DNS response time: This is the amount of time it takes for the DNS server to respond to a request for NS records.

Authoritative TTL for NS records: This is the maximum amount of time allowed for caching NS records for a hostname in the authoritative NS.

Spatial Features

Geographical and network distances: Using the traceroute command, we find the IP addresses of intermediate hops on the path to each NS of a given hostname. Features 5 and 6 are then generated using the hop IP addresses. An average geographic distance is calculated to obtain feature 10 using the geographic coordinates of a user's IP address and the NS hosts of a hostname. We get the coordinates from the Geolite2 database maintained by MaxMind.

Table 3: A list of proposed features for predicting phishing NS IP flux hostnames

Feature #	Category	Feature name
1	Temporal	Average Round Trip Time (RTT)
2		DNS response time for NS records
3		Average uptime of NSs
4		TTL of NS records
5	Spatial	Average number of hops between user and NSs
6		Average number of unique hops' countries between user and NSs
7		Average number of unique hops' continents between user and NSs
8		Number of unique countries hosting the NSs
9		Number of unique continents of NSs
10		Average geo-distance between user and NSs
11	DNS	Average number of unique A records per NS per single lookup
12		Average number of co-hosted websites per NS
13		At least one NS with a dynamic IP address
14	Network	Number of unique subnets of NSs
15		Number of unique networks of NSs
16		Number of unique ASNs of NSs
17		Number of unique AS organizations of NSs
18	Host	Ratio of available (up) NSs
19		Number of unique OSs of NSs
20		The most common OS
21		Installed with a web server
22		At least one NS uses a proxy IP address
23	Reputation	Total number of times IP addresses of all NSs of a hostname appear in a list of IP addresses of blacklisted phishing websites
24		Average number of times IP addresses of all NSs of a hostname appear in a list of IP addresses of blacklisted phishing websites
25		Ratio of NSs of a hostname whose IP addresses match with those of phishing websites that have been blacklisted
26		Registrar of NS records

DNS Features

Characteristics of Co-hosted Websites. We search for websites that are co-hosted on a computer that is identified by each NS of a hostname's IP address. Bing search engine was used to conduct the search with the search command "IP: W.X.Y.Z". We count the number of co-hosted websites from the search results and extract their URLs to generate feature 12.

Network Features

Network characteristics: In order to generate features 14-17, we extract network identity data from an IP geolocation database (IP2 location) for each IP address of NSs of a given hostname. The data includes the subnet, network, and Autonomous System Number (ASN). For instance, for feature 14, the number of distinct NS subnets per hostname is counted after identifying the subnet of each NS.

Host Features

Upstate of hosts: Using a host scanning tool (Nmap), we scan each NS of a given hostname to determine its state of availability The ratio of NSs in the "up" state is then calculated as feature 18.

Host's operating system: We scan each NS of a given hostname with Nmap to determine its Operating System (OS). Then, as features 19 and 20, we count the number of distinct OSs and also identify the most common OS among NSs for each hostname.

Host's webserver software: We check the response to an HTTP header request to see if a web server software is installed on each NS of a given hostname. Thus, feature 21 is generated.

Hosts with proxy IP addresses: We investigate into whether at least one NS for a specific hostname is located in a database of known public proxy IP addresses (we use the IP2Proxy database). This generates feature 22.

Reputation Features

IP Addresses shared with other malicious hostnames: To generate features 23-25, we identify IP addresses of NSs of a given hostname that correspond to IP addresses of blacklisted phishing URLs gathered over the past 3 months. For instance, in feature 23, we tally the total number of IP addresses of all NSs of each hostname that have matched in the database.

Domain registrar: To derive feature 26, we identify the registrars of the NS records for the hostnames of websites.

Results

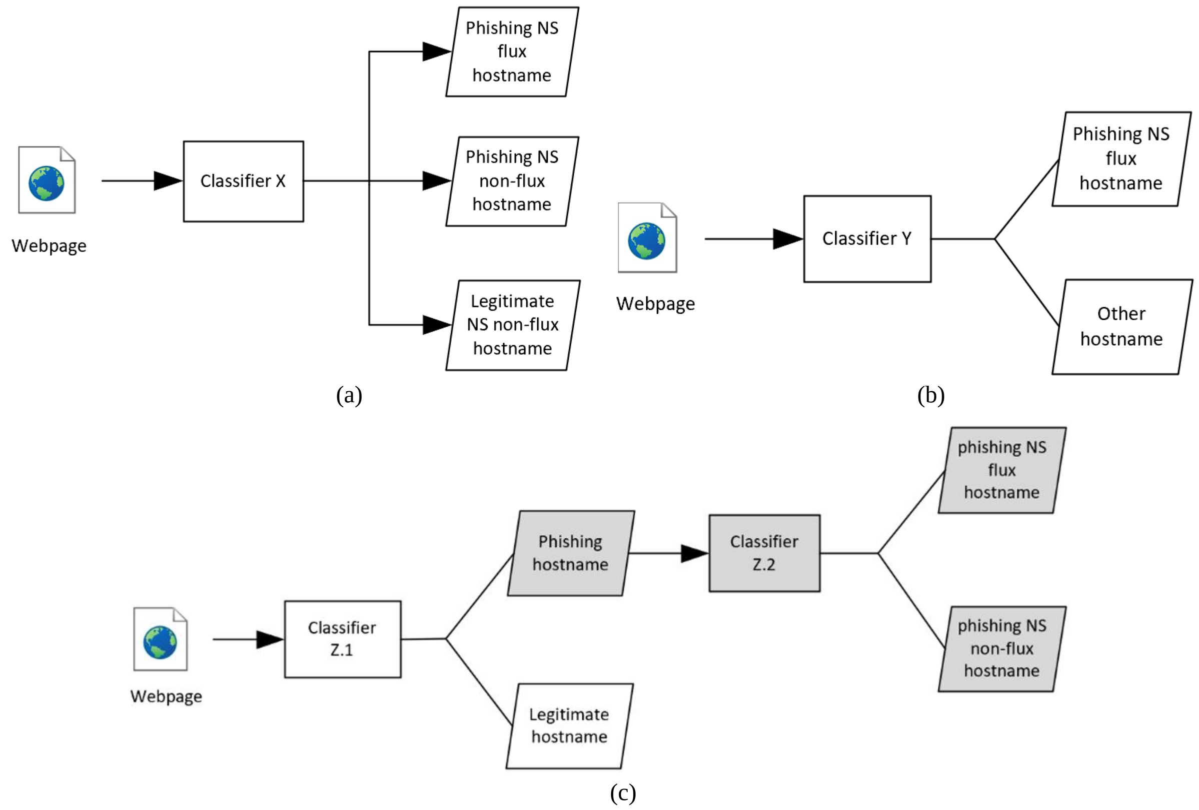
A number of experiments were run in order to develop a prediction model that assesses the effectiveness of the features in predicting phishing NS flux hostnames. First, using flat and hierarchical classification techniques, we designed three architectures for the model. We then identified binary and multi-class classifiers building each architecture. We ran two sets of experiments to assess the prediction performance of each classifier for each architecture. 8 traditional ML and 3 DL algorithms were used in the first and second experiments, respectively. For each classifier, the best set of features for the prediction task was determined. The overall performance of each architecture in predicting the flux hostnames was calculated by combining the classifiers' individual performances within each architecture. The best-performing architecture for the model is finally determined by comparing the performances of the various architectures. The classifiers and the architectures were evaluated using 8 common ML performance metrics. All experiments were run using Python and Jupyter hosted on Google's Colab platform. The host machine was a Windows 10 with 16 GB memory and Intel's i7 CPU.

Binary and Multi-Class Classification Based Architectures

Using flat and hierarchical classification techniques (Kowsari *et al*., 2017; Silla and Freitas, 2011; Weiss, 2020), 3 architectures for our proposed prediction model were taken into consideration for differentiating phishing NS flux hostnames from phishing and legitimate NS non-flux hostnames (Fig. 5a-c and Table 4). While the second (Architecture Y) uses a binary classification to separate phishing NS flux hostnames from the rest of the class as a whole, the first (Architecture X) performs a single-step three-class classification in which each hostname class is distinguished individually. The third (Architecture Z) uses a hierarchical approach to separate phishing hostnames from legitimate hostnames first, followed by the separation of phishing NS flux hostnames from phishing NS non-flux hostnames. We implement the approach using a Local Classifier per Parent Node (LCPN) technique (Silla and Freitas, 2011). In this technique, a binary or multi-class classifier is built at each parent node of the hierarchy to categorize the parent's child nodes. The node classifiers are trained using different datasets and selected features, which results in different prediction performances. When a new instance is predicted, it is predicted using a series of classifiers, from top to bottom, where the outputs of the parent classifiers are used as the inputs for the child classifiers. The best-performing ML algorithm is independently determined at each node using a selective classifier approach.

Building Training Datasets

To create a training dataset, the features discussed were taken from the labeled hostnames (discussed). Classifiers X, Y, Z.1, and Z.1 were trained using the entire dataset, as shown in Table 5, whereas classifier Z.2 had all legitimate hostnames removed from the dataset.


Webpage
Classifier X
Phishing NS flux hostname
Phishing NS non-flux hostname
Legitimate NS non-flux hostname
(a)
Webpage
Classifier Y
Phishing NS flux hostname
Other hostname
(b)
Webpage
Classifier Z.1
Phishing hostname
Legitimate hostname
Classifier Z.2
phishing NS flux hostname
phishing NS non-flux hostname

(c)

Fig. 5: (a) Architecture X (flat classification architecture); (b) Architecture Y (flat classification architecture); (c) Architecture Z (hierarchical classification architecture)

Table 4: Classifiers that are useful for the prediction of phishing NS flux hostnames in each architecture

Classifier	Classifier description	Classification type
Classifier X	Classifies hostnames into three classes: Legitimate NS non-flux hostnames, phishing NS flux hostnames, and phishing NS non-flux hostnames	Multi-class classification
Classifier Y	Classifies hostnames into two classes: Phishing NS flux hostnames and other hostnames (combination of phishing NS non-flux hostnames and legitimate NS non-flux hostnames)	Binary classification
Classifier Z.1	Classifies hostnames into two classes: Phishing and legitimate hostnames	Binary classification
Classifier Z.2	Classifies hostnames into two classes: Phishing NS flux and phishing NS non-flux hostnames	

Table 5: Training dataset created for each classifier

Classifier	Hostname class labels	Class size	Dataset size
Classifier X	Phishing NS flux hostnames	326	13,268
	Phishing NS non-flux hostnames	6,304	
	Legitimate NS non-flux hostnames	6,638	
Classifier Y	Phishing NS flux hostnames	326	13,268
	Other hostnames	12,942	
Classifier Z.1	Phishing hostnames	6,630	13,268
	Legitimate hostnames	6,638	
Classifier Z.2	Phishing NS flux hostnames	326	6,630
	Phishing NS non-flux hostnames	6,304	

Prediction Results

The prediction performances of the classifiers are reported using a variety of evaluation metrics. These include Accuracy, False Positive Rate (FPR), False Negative Rate (FNR), Precision, Recall, F1-score, ROC curve and Area Under Curve (AUC) (Brownlee, 2014; 2018; Müller and Guido, 2023). They are defined as follows:

$$Accuracy = TP + TN / TP + TN + FP + FN$$

$$FPR = FP / FP + TN$$

$$FNR = FN / FN + TP$$

$$Precision = TP / TP + FP$$

$$Recall = TP / TP + FN$$

$$F1 - score = 2 * Precision * Recall / Precision + Recall$$

The counts of True Positives (TP), False Positives (FP), True Negatives (TN) and False Negatives (FN) are used to calculate the performance metrics mentioned above. Keep in mind that an instance is defined as TP if it is positive and classified as such. It is FP if the instance is negative but is classified as positive. A negative instance classified as negative is TN and if it is classified as positive, it is called FN. A positive instance of this problem is the phishing NS flux hostname. In the following subsections, we present and compare the results of the classifiers for both ML and DL experiments.

Prediction Results of Traditional Machine Learning Algorithms

Eight traditional ML algorithms, namely Logistic Regression (LR), k-Nearest Neighbour (k-NN), Decision Tree (DT), Naive Bayes (NB), Support Vector Machine (SVM), Artificial Neural Network (ANN), Random Forest (RF) and Gradient Boosting (GB), were used to evaluate the proposed features (Brownlee, 2016; Chauhan, 2020; Dickson, 2019; Donges, 2021; JavaTpoint, 2021; Müller and Guido, 2023; Nicholson, 2019; Ray *et al*., 2017; VanderPlas, 2023; Yiu, 2019). Other researchers have successfully applied these algorithms to a variety of classification problems in cybersecurity (Al-Garadi *et al*., 2018; Apruzzese *et al*., 2018; Jiang and Li, 2017; Li, 2018; Xin *et al*., 2018). First, the best predictive set of features for each classifier was selected using the backward feature elimination method (Sillipo and Maarit, 2019).

In order to identify the best-performing algorithm and its best performance, we applied ML algorithms to the best feature set for each classifier. The stratified cross-validation technique (k-fold where k is 10) (Brownlee, 2023) was applied to the algorithms to obtain the average scores. The best classifier was then tuned using a Random Search method (Worcester, 2019) to achieve its optimal performance. Figure 6 displays how well each classifier's ML algorithms performed for all ROC curve threshold values. The results of the four top-performing algorithms for each classifier are presented in Tables 6a-b. The findings show that in each classifier, RF produces the best performance across the majority of metrics. Overall, classifier Y's RF produced the best performance, with the tuned hyperparameters and their values listed in Table 6c.

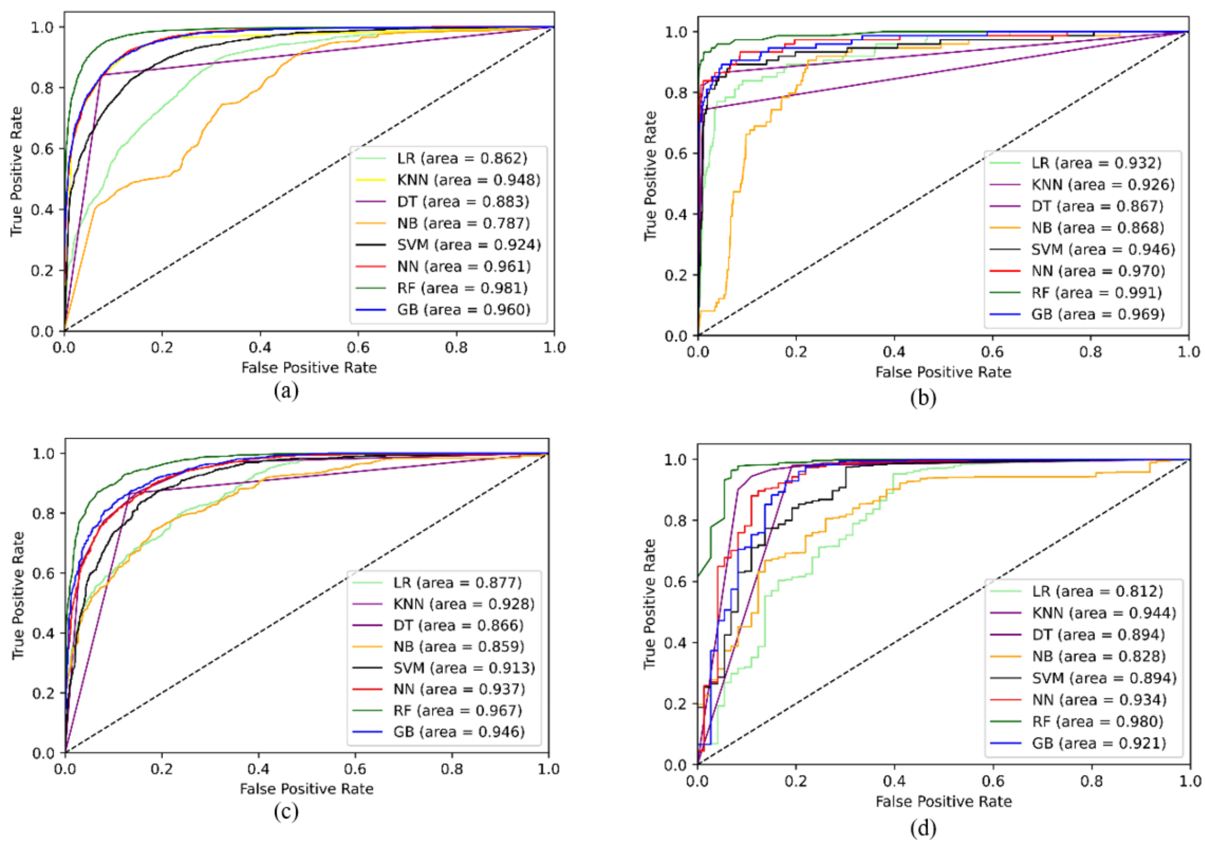


Fig. 6: (a) Classifier X's ROC curves of the traditional ML algorithms; (b) Classifier Y's ROC curves of the traditional ML algorithms; (c) Classifier Z.1's ROC curves of the traditional ML algorithms; (d) Classifier Z.2's ROC curves of the traditional ML algorithms

Table 6a: Prediction results of top four best-performing traditional ML algorithms for classifiers X and Y

		Classifier X				Classifier Y		
Algorithm	Acc. (%)	FPR (%)	FNR (%)	F1-score	Acc. (%)	FPR (%)	FNR (%)	F1-score
KNN	84.29	9.34	16.31	0.85	97.21	2.56	9.50	0.98
DT	89.67	6.34	13.21	0.90	97.53	1.04	7.17	0.99
RF	90.41	5.92	12.41	0.90	98.59	0.76	5.29	0.99
GB	82.74	10.24	18.33	0.83	97.97	2.64	7.01	0.97

Table 6b: Prediction results of top four best performing traditional ML algorithms for classifiers Z.1 and Z.2

		Classifier Z.1				Classifier Z.2		
Algorithm	Acc. (%)	FPR (%)	FNR (%)	F1-score	Acc. (%)	FPR (%)	FNR (%)	F1- score
KNN	85.43	15.29	13.85	0.85	94.95	14.72	4.55	0.96
DT	86.52	12.93	14.03	0.87	96.52	23.31	2.46	0.97
RF	90.19	10.29	9.34	0.90	98.42	11.17	0.57	0.98
GB	86.67	13.77	12.88	0.87	96.89	19.94	2.24	0.97

Table 6c: Optimal RF hyperparameters' values for classifier Y

Parameter	Description	Value
n_estimators	Number of trees	800
max_features	Max number of features considered for splitting a node	Auto
max_depth	Max number of levels in each decision tree	32
min_samples_split	Min number of data points placed in a node before the node is split	5
min_samples_leaf	Min number of data points allowed in a leaf node	2
bootstrap	Method for sampling data points	False

Prediction Results of Deep Learning Algorithms

In these experiments, we use the Fully Connected feedforward Deep Neural Networks (FC-DNN), the Long Short-Term Memory (LSTM) and the one-dimension Convolutional Neural Network (1D CNN) to evaluate the performance of the features (Al-Garadi *et al*., 2018; Apruzzese *et al*., 2018; Berman *et al*., 2019; Dertat, 2017; Kiranyaz *et al*., 2021; Moolayil, 2019; Nguyen, 2018; Phi, 2018; Verma, 2019). First, the datasets for LSTM and 1D CNN were converted into time series (with time step = 1), a standard input data format for the algorithms. The Random Search method was then used to fine-tune the DL algorithms. We first identified important tuning hyperparameters and their wide range of values for assessment (indicated in Table 7). We also attempted to tune using multiple hidden layers, but we discovered that only one hidden layer was necessary to produce the best results for each algorithm. The performances were not boosted by more layers. The classifiers were then built using the determined optimal values of all the hyperparameters. The results in Tables 8a-b demonstrate that LSTM outperformed the other algorithms in classifiers Y and Z.2, while FC-DNN performed best in classifiers X and Z.1. Overall, LSTM in classifier Y outperformed the other algorithms in all four classifiers, achieving the best performance. The three network architectures of the top-performing classifier (classifier Y) and their tuned hyperparameters are shown as examples in Figs. 7a-c.

Overall Prediction Results of the Architectures

We discovered that the tuned RF outperformed the DL algorithms in all metrics by comparing the results of the ML and DL algorithms for each classifier. Next, we consider all the architectures proposed in section 5.1 to identify the one that offers the best performance for predicting the phishing NS flux hostnames. The overall performances of the architectures were first assessed. In order to achieve the overall accuracy at the leaf class (in this case, the phishing NS flux hostname), we used the method proposed by Kowsari *et al*. (2017), which takes the accuracy of the child classifier as a fraction of the accuracy of its parent classifier. We calculate the misclassification errors at the leaf class by adding the errors of the parent and child classifiers since the LCPN hierarchical approach propagates misclassification errors from top to bottom (Silla and Freitas, 2011). The accuracy and error rates are calculated using the following formulas:

$$Overall\ (leaf)\ accuracy = Accuracy\ of\ a\ parent\ classifier * Accuracy\ of\ a\ child\ classifier$$

$$Overall\ (leaf)\ error\ rate = Error\ rate\ of\ a\ parent\ classifier + Error\ rate\ of\ a\ child\ classifier$$

Table 7: Hyperparameters values evaluated for tuning each DL algorithm

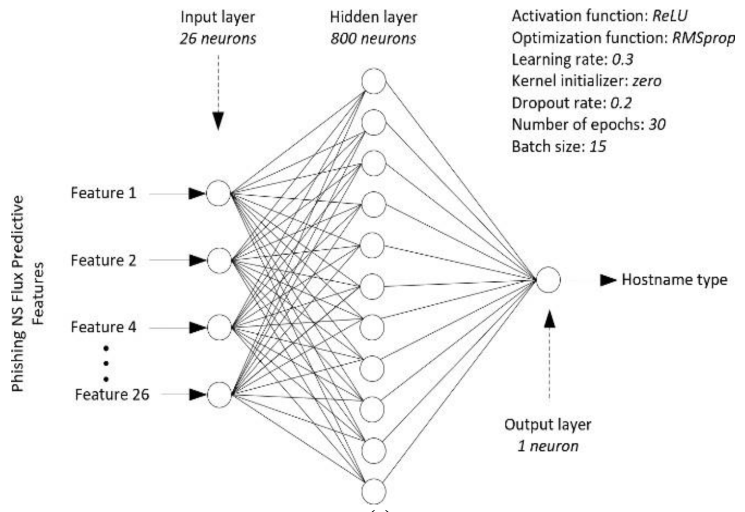
Hyperparameter	Range of evaluated values
Number of neurons in dense layers of FC-DNN/ memory units in a hidden layer of LSTM/ filters in convolution layer of CNN	10, 30, 50, 80, 100, 150, 200, 300, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000, 2200, 2400, 2800, 3000
Activation functions optimization algorithms	Relu, tanh, linear, soft plus, soft sign, softmax SGD, Adam, Adamax, Nadam, RMSprop, Adagrad, Adadelta
Learning rates	0.001, 0.01, 0.1, 0.2, 0.3
Kernel initializers	Uniform, glorot_uniform, he_normal, he_uniform, lecun_uniform, normal, zero, glorot_normal
Dropout rates	0.1, 0.2, 0.3, 0.4, 0.5
Batches	15, 30, 50, 70, 90, 110, 130, 150
Epochs	10, 30, 60, 90, 120, 150, 180, 210, 240, 270, 300

Table 8a: Prediction results of DL algorithms for classifiers X and Y

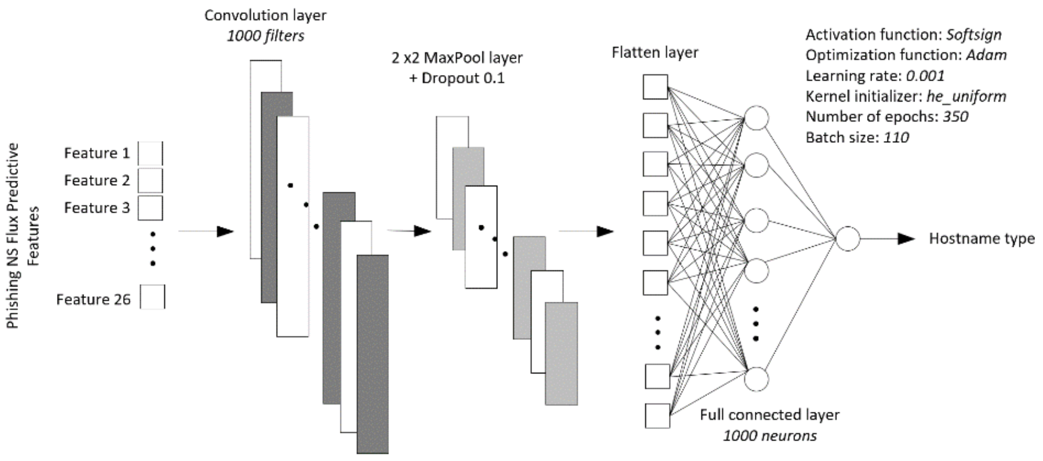
		Classifier X				Classifier Y		
Algorithm	Acc. (%)	FPR (%)	FNR (%)	F1-score	Acc. (%)	FPR (%)	FNR (%)	F1-score
FC-DNN	85.59	9.01	17.13	0.86	97.84	0.44	21.89	0.98
LSTM	81.94	11.59	26.09	0.82	98.51	0.25	16.60	0.98
CNN	74.39	16.74	28.23	0.74	98.40	0.42	21.49	0.98

Table 8b: Performance results of the evaluated DL algorithms for classifiers Z.1 and Z.2

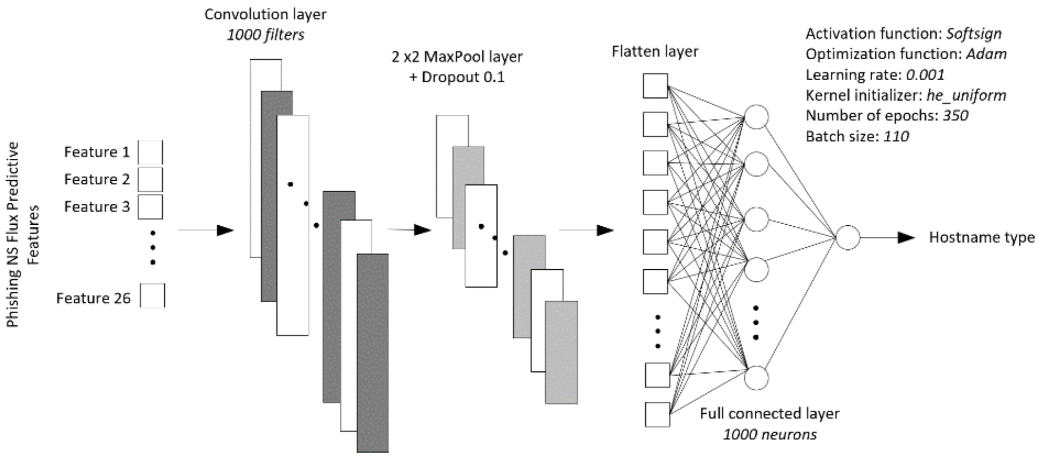
		Classifier Z.1				Classifier Z.2		
Algorithm	Acc. (%)	FPR (%)	FNR (%)	F1-score	Acc. (%)	FPR (%)	FNR (%)	F1-score
FC-DNN	86.43	12.20	15.55	0.86	95.77	0.89	23.01	0.96
LSTM	85.25	14.39	24.03	0.86	97.54	0.65	18.63	0.98
CNN	78.91	14.92	19.43	0.79	97.40	0.71	21.05	0.98



(a)



(b)



(c)

Fig. 7: (a) FC-DNN architecture of classifier Y with the optimal hyperparameters' values; (b) LSTM architecture of classifier Y with the optimal hyperparameters' values; (c) 1D CNN architecture of classifier Y with the optimal hyperparameters' values

The best performance of each architecture is summarized in Table 9. The RF performances of classifiers Z.1 and Z.2 are combined for architecture Z using the aforementioned formulas. Following architecture X in terms of performance across all metrics was architecture Y. As a result, we draw the conclusion that architecture Y is the most effective one for implementing the model

Analysis of Prediction Performance and Feature Importance Ranking

The effectiveness of feature categories and specific features on the best-performing classifier (classifier Y) is examined in this section. Based on feature importance weights, the feature selection method resulted in the classifier's top 11 features (Fig. 8), all of which are third-party-based and are being proposed for the first time for this problem.

The most important feature is feature 21 (Table 3 for feature numbers), which determines whether web server software is installed on the NS hosts. According to the data we gathered, Fig. 9a demonstrates that nearly 70% of NS hosts of the phishing NS flux hostnames have web servers installed, 40% of NS hosts of the phishing NS non-flux hostnames, and less than 5% of NS hosts of the legitimate NS non-flux hostnames. Even after combining the two NS non-flux hostnames, their total percentage is still less than half of the percentage in the NS flux hostnames (Fig. 9b). Assuming equal numbers of samples of each hostname type in the dataset, then if only this feature is known, a hostname with a 'Yes' value is about 1.6 times more likely than not to be a phishing NS flux hostname in the three-class case and about 3.3 times more likely to be a phishing NS flux hostname in the two-class case. In contrast to legitimate NSs, which are typically hosted on dedicated servers, it is plausible that the majority of NSs of phishing hostnames are co-hosted with malicious websites. Attackers may have gained access to a small percentage of legitimate NSs with web servers to host their malicious web services. The average number of websites that are co-hosted in a hostname's NS servers is the second-ranked feature (feature 12). Its data distributions in Figs. 10a-b, which show a correlation with those in the prior feature, support our suspicion. We believe attackers host multiple malicious websites in addition to the NS application on their servers for two reasons: (1) These websites expect to receive a lower volume of user requests than the established legitimate websites, which reduces the need for more computing resources; and (2) Attackers prefer to use a small number of resources to launch numerous attacks in order to maximize their return of investment.

Table 9: Prediction performances of the architectures

Architecture	Acc. (%)	FPR (%)	FNR (%)	Classification type
X	90.41	5.92	12.41	Multi-class
Y	98.59	0.76	5.29	Binary
Z	88.76	21.46	9.91	Multi-class

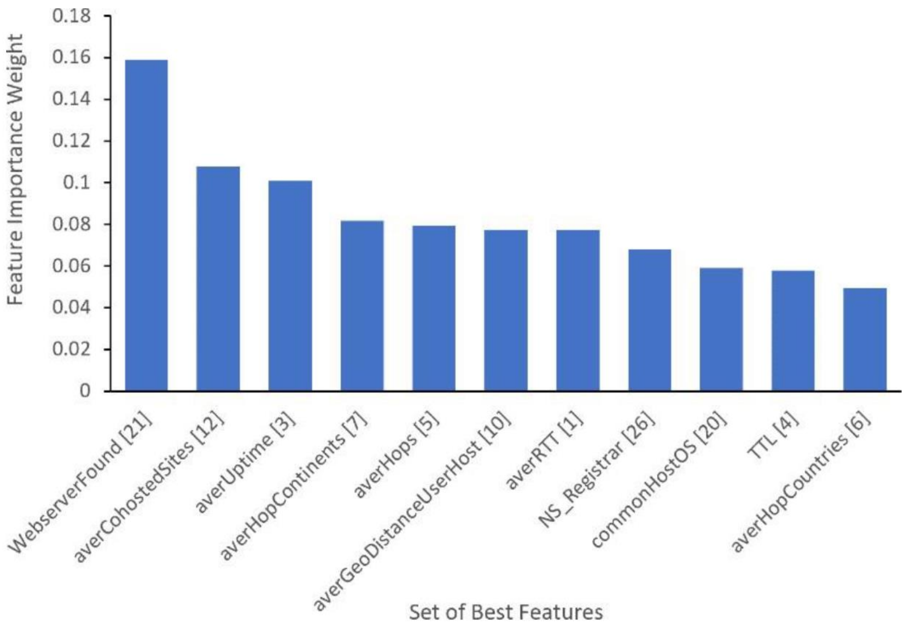
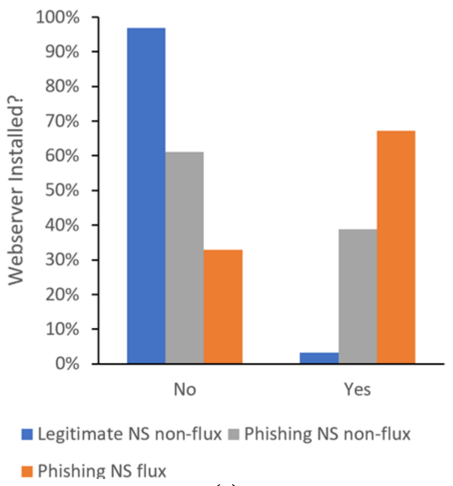
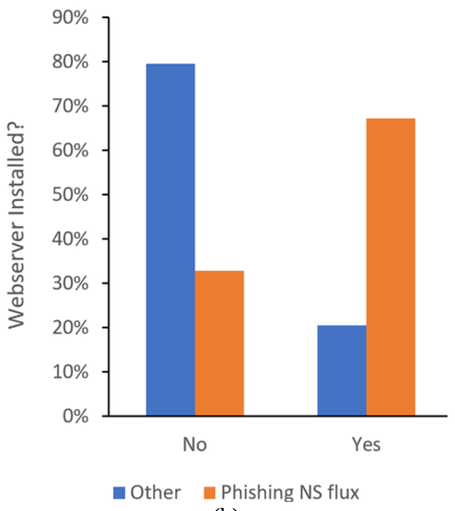


Fig. 8: The ranking of best features of classifier Y by importance weights. Numbers in the brackets represent numbers of the features as indicated in Table 2

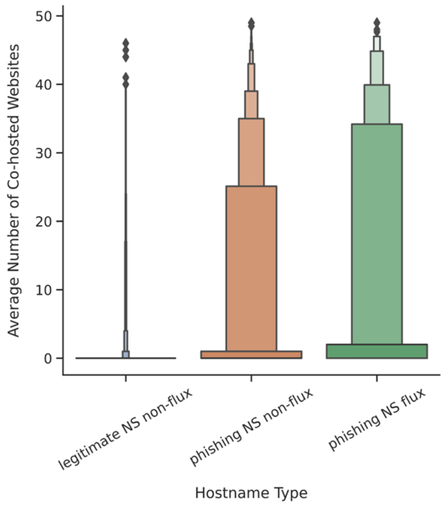


(a)

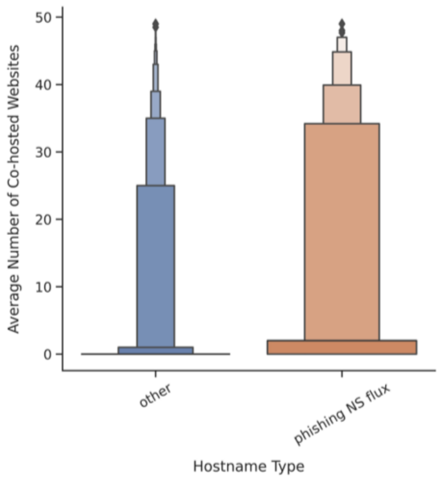


(b)

Fig. 9: (a) Percentage of hostnames for each class whose NSs have web servers installed; (b) Percentage of classifier Y hostnames whose NSs are installed with web servers



(a)

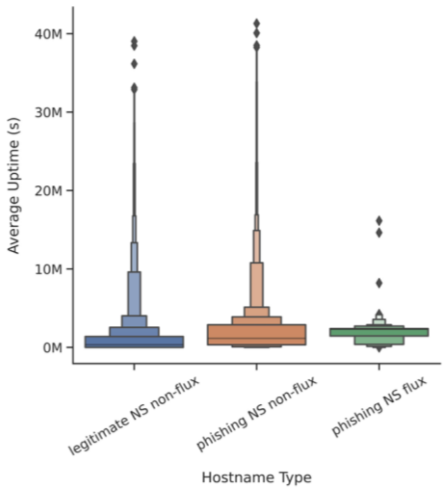


(b)

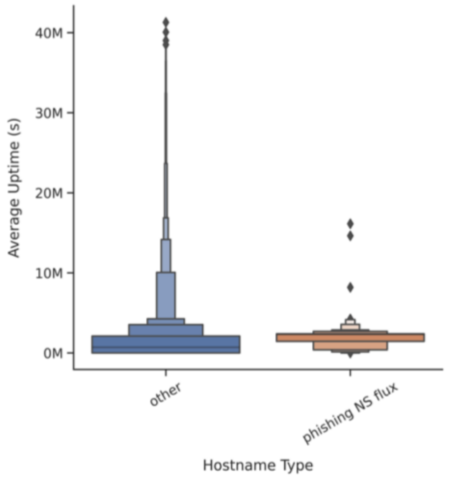
Fig. 10: (a) Distribution of average counts of distinct websites shared in the hostname's NSs for each hostname class; (b) Distribution of average counts of distinct websites shared in the hostname's NSs for the two classes of hostnames in classifier Y

The average uptime of NS hosts of a hostname (feature 3), which is ranked in the third position, is the highest-ranked temporal feature. According to Fig. 11a data distribution, the median uptime of the hosts of NSs for phishing NS flux hostnames is higher than that of the other classes but the distribution is much more concentrated at low values. Very few phishing NS flux hostnames have average uptimes above 5 million seconds while a sizeable minority of other hostnames have uptimes above this value and some exceed 30 million seconds. It appears that a subject with a very low average uptime is probably a legitimate NS non-flux hostname, one with a medium average uptime is probably a phishing NS non-flux hostname and one with a high average uptime is probably a phishing NS flux hostname. The binary classification case in Fig. 11b shows a similar pattern.

The fifth-ranked feature is the number of network hops between a user and the NS hosts of the same hostname (feature 5). Its data distribution in three and two class classifications are shown in Figs. 12a-b, respectively. Based on the median numbers and density distributions, NS hosts for phishing NS flux hostnames are typically farther away from the user in termas of network hops than those for the other two classes of hostnames, with legitimate NS non-flux hostnames having the lowest median number. Other spatial features ranked at positions 4, 6, and 11 show comparable patterns. Therefore, our data supports previous studies that NS flux agents are more geographically dispersed than NSs of non-flux networks. This is consistent with the hypothesis that the malware in charge of the flux networks infects vulnerable computers in a random manner from a large number of networks.

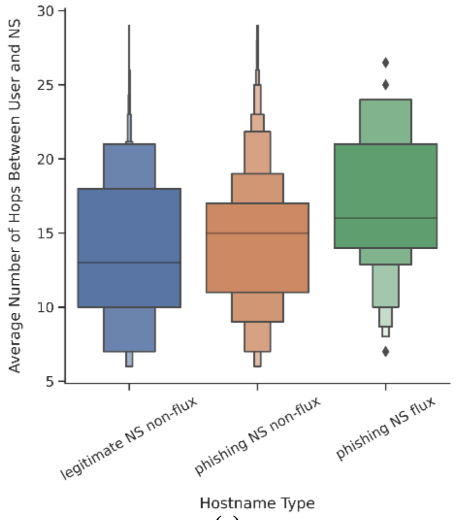


(a)

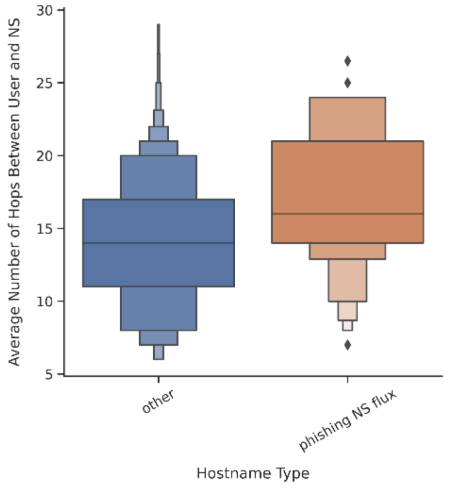


(b)

Fig. 11: (a) Distribution of hostname NSs' average uptime for each class; (b) Distribution of hostname NSs' average uptime in classifier Y



(a)



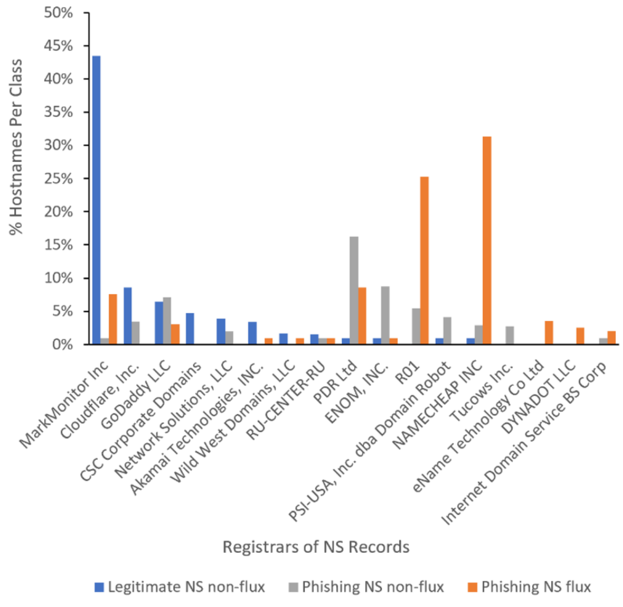
(b)

Fig. 12: (a) The three hostname classes' respective averages for the number of network hops between users and NS hosts; (b) The two hostname classes' respective averages for the number of network hops between users and NS hosts in classifier Y

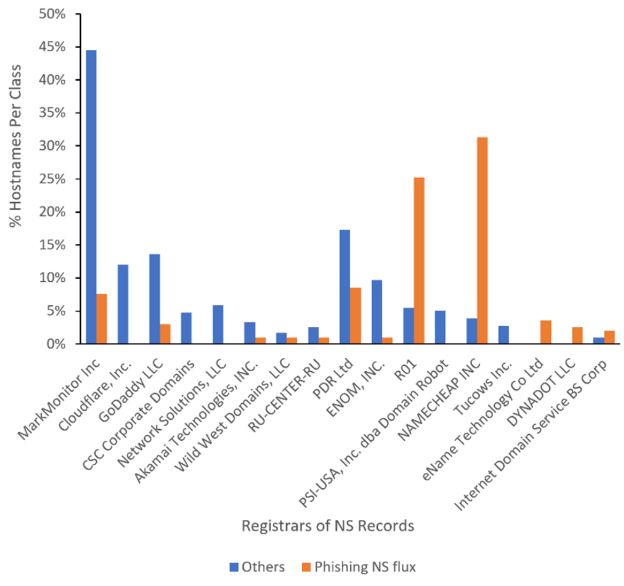
The only reputation-based feature in the ranking, Registrar of NS Records (feature 26), is in eighth place. Figures 13a-b, some of the registrars including Namecheap Inc., R01, PDR Ltd, eName Technology Co. Ltd, Dynadot LLC, and Internet Domain Service BS Corp register a high proportion of NS records for both phishing NS flux and non-flux hostnames. This implies that some registrars take less stringent measures to verify record owners at the time of registration and to keep an eye on how the records are used, which makes it easier for attackers to exploit their platforms.

The TTL of the NS Records feature (feature 4) is ranked at position 10 in the list. The distributions of its data are shown in Figs. 14a-b. While the majority of NS non-flux hostnames have TTLs of 86400 and 3600 sec, the majority of phishing NS flux hostnames have TTLs of 600 sec followed by 86400 and 7200 sec. In order to maintain the fluxing behavior of the NS networks, it is expected that NS flux hostnames use short TTLs to make sure that frequently changed NS records are returned to users when they query from their authoritative NSs instead of the cached records.

Additionally, the best feature subset of classifier Y's composition and each feature category's performance contribution were examined. A categorization is shown in Table 10. The best feature set has the most temporal and spatial-based features. While none of the network features were included in the set, DNS, host, and reputation-based features each contributed slightly less than half of their proposed features. This is in line with how well each category performs when run with the tuned classifier Y shown in Fig. 15. The best accuracy, the lowest FPR, and the second-lowest FNR were produced by spatial features. While producing slightly lower accuracies and FPRs than spatial features, temporal and DNS features produced the highest FNRs. Conversely, the reputation category yielded the lowest accuracy and FNR but the highest FPR. None of the categories produced the best performance across all three metrics.

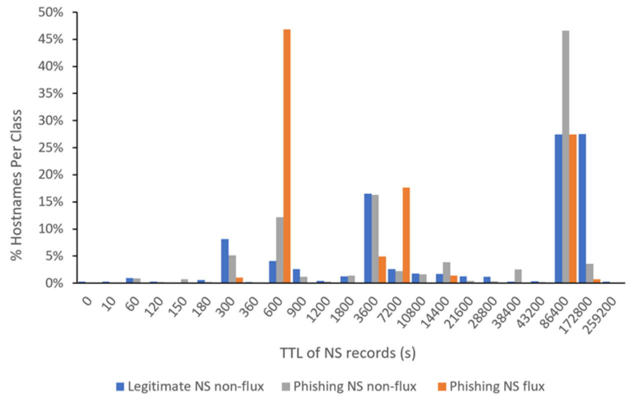


(a)

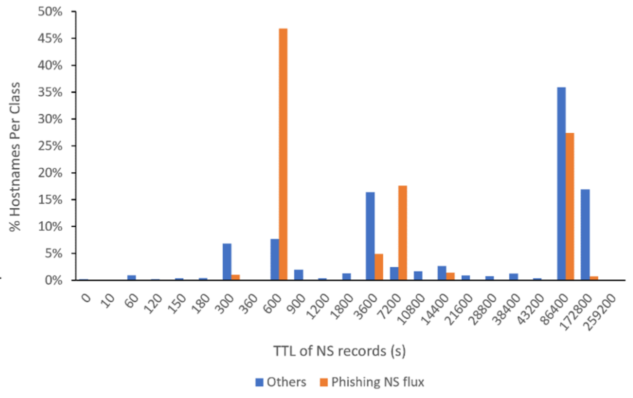


(b)

Fig. 13: (a) Distribution of NS record registrars among the three hostname classes; (b) Distribution of registrars of NS records of the two classes of hostnames in classifier Y



(a)

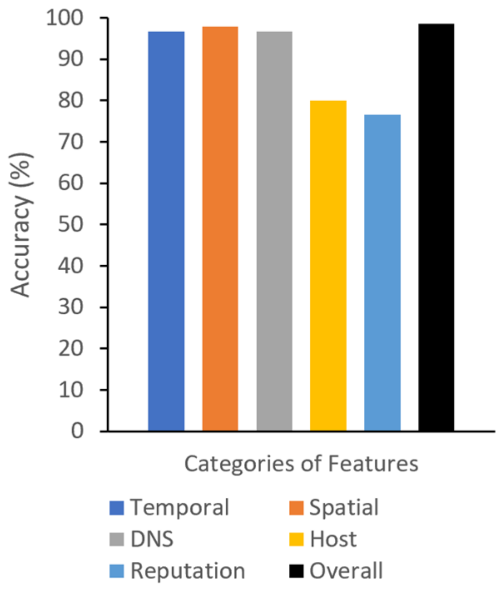


(b)

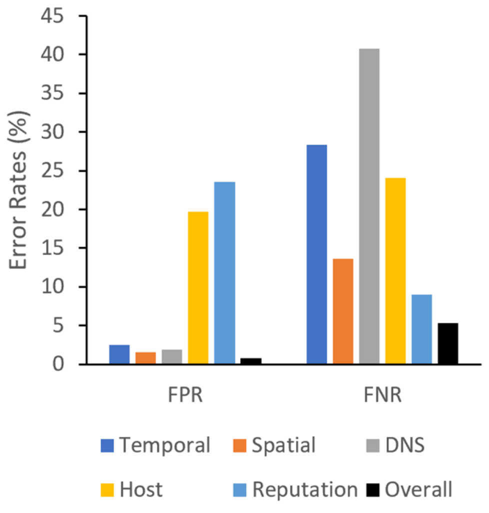
Fig. 14: (a) TTL distribution for NS records for each of the three hostname classes; (b) TTL distribution for NS records for each of the two hostname classes in classifier Y

Table 10: Composition of categories of classifier Y's best feature set

Feature category	Tally of full features	Tally of best features	Best features # (# from Table 5.12)
Temporal	4	3	1, 3, 4
Spatial	6	4	5, 6, 7, 10
DNS	3	1	12
Network	4	0	-
Host	5	2	20, 21
Reputation	4	1	26



(a)



(b)

Fig. 15: (a) Comparison of the classifier Y's feature category accuracy rates; (b) Comparison of classifier Y's feature category error rates

Detection Time

To evaluate the efficiency of our best classifier (classifier Y) in identifying phishing NS flux hostnames in real-time, the runtimes of its three phases (feature extraction, dataset training, and prediction) were measured (Table 11). The retrieval of feature data from their sources accounts for the majority of the 5.1 sec detection time (sum of the first two phases in Table 11). The breakdown of times of activities for extracting the best features is shown in Fig. 16. The strongest predictor, an HTTP header request to check the presence of a web server, takes the longest. The second longest time was spent on host scanning using Nmap to extract uptime and common OS features (at the third and ninth positions in Fig. 8), while the quickest time was spent on NS records queries to extract TTL (10th position). There is little room for reducing the detection time by removing the least significant features from the set because the strongest predictors took the longest to extract. We also measured the detection times for classifiers X and Z in a similar manner and they were found to be 5.4 and 6.3 sec, respectively. The reason for the slight variations in the times between the three classifiers is primarily due to their distinct sets of best features, which has an impact on the overall time required to extract the features.

Model Evaluation Using New Data

Using a new dataset compiled over a distinct time period, the performance consistency of the best classifier was examined. Between January 10 and March 15, 2021, a total of 1,413 legitimate and 1,398 phishing websites were collected for the new testing dataset. When tested on the new dataset, Classifier Y achieved accuracy rates of 97.18%, FPR rates of 1.85%, and FNR rates of 6.09%, which are only marginally worse than those recorded with the training dataset. The differences in accuracy, FPR, and FNR were 1.41, 1.09 and 0.8%, respectively. In order to learn more about the performance consistency of the classifier, including whether it is caused by a change in the strategies used by phishers over time, we intend to carry out additional validation tests using additional new datasets amassed in additional separate periods in the future. Despite the variations, the testing results are still acceptable and fall within the range reported in the majority of works in the field of IP flux network detection.

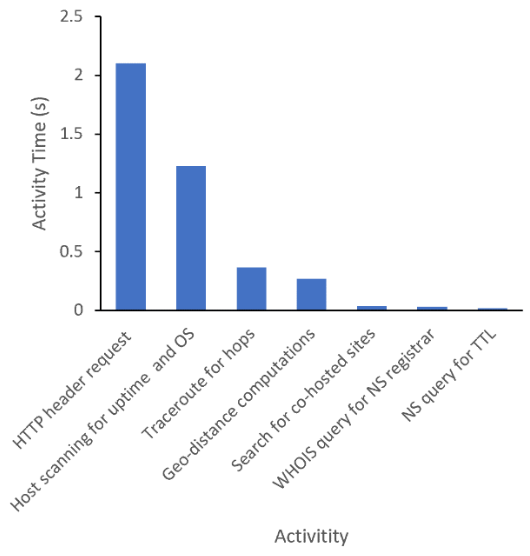


Fig. 16: Distribution of feature extraction times by activities

Table 11: Classifier Y's runtimes for the prediction of phishing NS flux hostnames

Phase	Time (s)
Feature extraction per webpage	5.0571
Prediction per webpage	0.0001
Detection time	5.0572
Training the dataset	11.0100

Discussion

Comparison with Related Works

In this section, we contrast our research with previous studies that propose approaches for detecting phishing NS flux hostnames. Only two such works, Kadir *et al*. (2012); Pa *et al*. (2015), were found, as explained Key aspects of the comparison are summed up in Table 12. The detection time is the primary distinction between our work and the other works. While the other two studies collected feature data over longer periods of time, resulting in detection times of 3 and 6 months respectively, our work used feature data collected at a single point in time, yielding the 5.1 sec detection time. The other proposed approaches give fluxing NSs under investigation sufficient time to continue serving phishing websites and therefore not suited for real-time detection.

The diversity of features used is the other difference. While our work used 5 different feature categories, the other two works only used DNS-based features for the prediction. As we have previously noted, compared to using a large number of categories, the use of a small number of feature categories increases the risk of detection evasion. For instance, the total number of NSs' unique IP addresses and their fluxing rates throughout the observation period were used as the features by Kadir *et al.* (2012). Some of the legitimate NSs also change their IP addresses, as shown in our data though at a lower rate than phishing NSs do. In order to increase the error rates thus decreasing the effectiveness of detection, the attacker may opt to reduce the fluxing rates and/or the number of unique IP addresses of phishing NSs to a range similar to those of legitimate NSs.

The time interval between consecutive DNS queries for A records of NSs during the NS monitoring phase is another notable difference. Kadir *et al*. (2012) queried the A records once every 12 h, whereas Pa *et al*. (2015) did not specify their time interval. The records used in our study were monitored every 2 h for the reason outlined. Our study is likely to have captured more precise information about the NS fluxing behaviors due to our shorter time interval than Kadir *et al*. (2012) work and as a result, should be able to make better prediction.

Applications, Limitations and Future Work

Our proposed model has a number of applications. Protecting users from visiting phishing websites whose DNS records are hosted in fluxing NSs is one of the applications. This can be done, for example, by incorporating the model in a web browser or network gateway application to filter out any phishing websites that users are attempting to access. In an effort to achieve a latency suitable for a potential real-time application, the current detection time can be decreased by extracting features concurrently rather than sequentially (the current implementation). Before extracting the rest of the features in parallel, the query of NS records to identify names of NSs and then the query of their A records to identify IP addresses would have to be performed first. With this approach, the detection time will drop to 2.1 sec, the longest time to extract a feature, as indicated in Fig. 16. As recommended by MachMetrics (2021), this time is feasible for real-time protection of users.

Table 12: Comparison of some of the related works with our work

Work	Feature # and categories	Classification type	Data size (URLs)	Evaluation algorithms	Detection time	Performance
Kadir *et al*. (2012)	7 DNS features	Binary classification (NS IP flux hostnames versus benign hostnames)	500	k-NN	3 months	FPR = 0% FNR = 0%
Pa *et al*. (2015)	3 DNS features	Binary classification (NS IP flux hostnames versus non-NS IP flux hostnames)	50,030	Mappings of IP and hostnames of NSs	6 months	FPR = 0.8%
Our work	3 temporals, 4 spatial, 1 DNS, 2 host and 1 reputation features	Binary classification (phishing NS IP flux hostnames versus phishing non-NS IP flux hostnames)	13, 268	LR, k-NN, DT, NB, SVM, ANN, RF, GB, FC-DNN, LSTM, 1D CNN	5.1 sec	Acc = 98.59% FPR = 0.76% FNR = 5.29% Prec. = 0.99 Recall = 0.98 F1 = 0.99 AUC = 0.99
		Multi-class classification (phishing NS IP flux hostnames versus phishing NS IP non-flux and legitimate NS IP non-flux hostnames)			6.3 sec	Acc = 90.41% FPR = 5.92% FNR = 12.41%

Note that the Kadir *et al*. (2012) model, despite having a lengthy detection time, produced no prediction errors. Their model and ours can be used in parallel to improve the overall prediction task by combining their accurate detection with our model's fast detection capability. In this approach, our model will provide instant detection of most phishing NS flux hostnames with a few errors while their model will provide a more accurate detection in the long run. This will lessen the number of phishing NS flux hostnames that our model will misclassify and allow the hostnames to continue to operate before being detected by the second model.

Our model can be used to build a blacklist of phishing NS flux hostnames, as was mentioned in the introduction. We are not aware of any blacklists of NS flux hostnames that are currently in existence. In this manner, the model would be fed with a continuous stream of hostnames from different sources, including user emails, network traffic, and databases of legitimate and phishing websites. The blacklist would be updated to include hostnames identified as phishing NS flux websites. The blacklist can then be applied in a number of ways to offer end users real-time protection. A web browser plug-in and a cloud anti-malware suite with its clients installed at the end users are two examples of such applications. For real-time applications, a blacklist approach has shown to be effective in other cybersecurity-related fields (Chen *et al*., 2014; Kordestani and Shajari, 2013) fields. For further research into phishing NSIFNs and the development of various tools to address networks and the web services they host, security researchers, vendors, and authorities may find the blacklist to be a useful resource.

Lastly, classifier Z.2, which has produced a relatively good performance, can serially be combined with a model, such as that proposed by Nagunwa *et al*. (2020), that distinguishes phishing websites from legitimate ones with a high prediction performance. In this integration, the latter would be used to first detect phishing websites and the ones that were found would then be fed into the former model to determine whether the inputs were phishing NS flux hostnames or phishing NS non-flux hostnames. This can be helpful when security professionals need to differentiate between the two phishing hostnames in order to implement the right countermeasures to the attacks at the network level. For instance, for phishing NS flux hostnames, this would require an approach described in the introduction section. For phishing NS non-flux hostnames, phishing NSs can be directly identified through querying A records of NSs of the hostnames and then blacklisting the returned IP addresses.

We argue that in addition to technological approaches, effective mitigation of phishing attacks can be accomplished by integrating non-technical strategies. Having strict anti-phishing or cybercrime laws is one of them. In response to the shortcomings of general criminal laws in combating crimes in cyberspace, many nations around the world have begun to adopt specific national laws for cybercrimes (Mehta *et al*., 2022). The cross-border nature of phishing attacks, however, may limit individual countries' efforts to combat the attacks because national laws may differ in their interpretations of cybercrimes and their scope, resulting in varying degrees of legal action against attackers. This might encourage attackers to move their operations to nations with lax laws. In order to address all types of cross-border cybercrimes equally, we call for the establishment of international cybercrime laws.

The main drawback of our model is that all 11 of its best features came from third-party services. The performance of the model could be significantly impacted by the inability to connect to the services or by a lack of the requested data. Our analysis of the data revealed that 8 of the 11 features had missing value percentages ranging from 1.6-31%. Only one feature had 31% and one feature also had the second-highest percentage (17.9%). However, the data distribution of our dataset suggests that under typical conditions, it is less likely to have many features with high percentages of missing values. Even with the missing values, our model was still able to achieve a good performance.

Attackers may be able to combine different types of fluxing behaviors, such as IP flux, domain flux, NS IP flux, and NS name flux in the same network, as observed by Salusky and Danford (2008); Kadir *et al*. (2012); Yadav *et al*. (2012). In the future, we aim to extend our work by investigating the extent to which attackers use some or all of these behaviors concurrently. Additionally, we intend to investigate any potential approaches for detecting hostnames hosted in domain flux and NS name flux.

A number of ML models have been shown to be vulnerable to adversarial ML attacks in recent years, such as those described in the MITRE Adversarial Threat Landscape for Artificial-Intelligence Systems (ATLAS) knowledge base (MITRE, 2021). Attackers can employ these techniques to introduce corrupted data samples into the training data or slightly tamper with some of the benign training data samples in order to reduce the classification rates of the models (Boesch, 2021; Martins *et al*., 2020). It is advised that the developer evaluate the model against these potential attacks through various experiments in order to assess the confidence level of a model in resisting the attacks. We aim to undertake this evaluation for our proposed model.

Conclusion

An ML-based model for predicting phishing hostnames hosted in NS flux networks has been proposed in this study. The model is based on 11 features that are all new to this problem and are divided into five categories. Three model implementation architectures based on binary and multi-class classification approaches were proposed and evaluated using eight conventional ML and three DL algorithms. The findings indicated that the binary classification-based architecture, which distinguishes phishing NS flux hostnames from legitimate NS non-flux hostnames and phishing NS non-flux hostnames combined as a single class, was the model's most accurate architecture. The two-class classification-based architecture was found to be more accurate than the three-class multi-class classification-based architecture, which is useful in identifying a particular hostname type among the three hostnames.

Investigation into the importance of the proposed features for prediction in the best-performing architecture revealed that the strongest predictors are those in the spatial and temporal categories, while network-related features have no bearing on prediction. The proposed model has produced high detection performance comparable to other similar works in the literature, indicating that our novel features are as effective as the existing ones. Our approach, in contrast to the existing ones, has achieved fast detection for real-time applications, has employed more diverse features to improve resistance to detection evasions, and has approached the problem as a three-class classification problem, providing a more pragmatic solution to the problem. Additionally, the results were reported using a broader range of performance metrics, affirming the solution's reliability.

It is important to note that the nature of the dataset used to train the model in this study has an impact on the performance obtained. Attackers are likely to vary configurations of their flux networks over time, which could lead to variations in the efficacy of some detection features and even the model itself. We believe that in order to maintain a high level of detection performance, it is crucial to continuously observe the behaviors of the flux networks and re-evaluate the model using fresh datasets that are collected on a regular basis.

Acknowledgment

I am appreciative of everyone who I had the opportunity to collaborate with on this research paper.

Funding Information

This research was partly funded by Birmingham City University in Birmingham, United Kingdom.

Ethics

I certify that this article has not already been published somewhere else. No conflicts of interest are disclosed by the authors.

References

Al-Garadi, M., Amr, M., Al-Ali, A., Du, X., & Guizani, M. (2018). A survey of machine and deep learning methods for Internet of Things (IoT) security. *arXiv preprint arXiv*: 1807.11023. https://doi.org/10.1109/COMST.2020.2988293

Apruzzese, G., Colajanni, M., Ferretti, L., Guido, A., & Marchetti, M. (2018). On the effectiveness of machine and deep learning for cyber security. 10th *International Conference on Cyber Conflict* (CyCon), 29 May-1 June 2018, IEEE, Estonia, pp. 371-390. https://doi.org/10.23919/CYCON.2018.8405026

Berman, D., Buczak, A., Chavis, J., & Corbett, C. (2019). A Survey of Deep Learning Methods for Cyber Security. *Information*, *10*(4): 122. https://doi.org/10.3390/info10040122

Boesch, G., (2021). What Is Adversarial Machine Learning? Attack Methods in (2021). Viso. https://viso.ai/deep-learning/adversarial-machine-learning/

Brownlee, J. (2014). Classification Accuracy is Not Enough: More Performance Measures You Can Use. https://machinelearningmastery.com/classification-accuracy-is-not-enough-more-performance-measures-you-can-use

Brownlee, J. (2016). Logistic Regression for Machine Learning. https://machinelearningmastery.com/logistic-regression-for-machine-learning/

Brownlee, J. (2023). Machine Learning Mastery with Python. Discover the fastest growing platform for professional machine learning with step-by-step tutorials and end-to-end projects. https://machinelearningmastery.com/machine-learning-with-python/

Brownlee, J. (2018). How to Use ROC Curves and Precision-Recall Curves for Classification in Python.

Caglayan, A., Toothaker, M., Drapaeau, D., Burke, D., & Eaton, G. (2010). Behavioral Patterns of Fast Flux Service Networks. 43rd *Hawaii International Conference on System Sciences*, pp: 1-9. https://doi.org/10.1109/HICSS.2010.81

Chauhan, N. (2020). Decision tree algorithm, explained. https://www.kdnuggets.com/2020/01/decision-tree-algorithm-explained.html

Chen, Y. S., Yu, Y. H., Liu, H. S., & Wang, P. C. (2014, August). Detect phishing by checking content consistency. In *Proceedings of the 2014 IEEE 15th International Conference on Information Reuse and Integration (IEEE IRI 2014)* (pp. 109-119). IEEE. https://doi.org/10.1109/IRI.2014.7051880

Dertat, A. (2017). Applied deep learning-part 4. *Convolutional Neural Networks*. Towards Data Science, 26. https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

Dickson, B. (2019). What are Artificial Neural Networks (ANN)? https://bdtechtalks.com/2019/08/05/what-is-artificial-neural-network-ann/

Donges, N. (2021). A complete guide to Random Forest Algorithm. https://builtin.com/data-science/random-forest-algorithm

Gu, G., Porras, P. A., Yegneswaran, V., Fong, M. W., & Lee, W. (2007, August). Bothunter: Detecting malware infection through ids-driven dialog correlation. In *USENIX Security Symposium* (Vol. *7*, pp. 1-16). https://doi.org/doi/10.5555/1362903.1362915

JavaTpoint. (2021). Decision tree classification algorithm. https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm

Jiang, C. B., & Li, J. S. (2017). Exploring Global IP-Usage Patterns in Fast-Flux Service Networks. *J. Comput.*, *12*(4), 371-379. https://doi.org/10.17706/jcp.12.4.371-380

Kadir, A. F. A., Othman, R. A. R., & Aziz, N. A. (2012, August). Behavioral analysis and visualization of fast-flux DNS. In *2012 European Intelligence and Security Informatics Conference* (pp. 250-253). IEEE. https://doi.org/10.1109/EISIC.2012.36

Khattak, S., Ahmed, Z., Syed, A. A., & Khayam, S. A. (2015). BotFlex: A community-driven tool for botnet detection. *Journal of Network and Computer Applications*, 58: 144-154. https://doi.org/10.1016/j.jnca.2015.10.002

Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M., & Inman, D. J. (2021). 1D convolutional neural networks and applications: A survey. *Mechanical Systems and Signal Processing*, *151*: 107398. https://doi.org/10.1016/j.ymssp.2020.107398

Konte, M., Feamster, N., & Jung, J. (2009). Dynamics of online scam hosting infrastructure. Proc. *International Conference on Passive and Active Network Measurement*, *Springer*, pp. 219-228. https://doi.org/10.1007/978-3-642-00975-4_22

Kordestani, H., & Shajari, M. (2013, May). An entice resistant automatic phishing detection. In *The 5th Conference on Information and Knowledge Technology* (pp. 134-139). IEEE. https://doi.org/10.1109/IKT.2013.6620052

Kowsari, K., Brown, D. E., Heidarysafa, M., Meimandi, K. J., Gerber, M. S., & Barnes, L. E. (2017, December). Hdltex: Hierarchical deep learning for text classification. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)* (pp. 364-371). IEEE. https://doi.org/10.1109/ICMLA.2017.0-134

Li, J. H. (2018). Cyber security meets artificial intelligence: A survey. *Frontiers of Information Technology & Electronic Engineering*, *19*(12): 1462-1474. https://doi.org/10.1631/FITEE.1800573

MachMetrics. (2021). Average Page Load Time in 2021. https://machmetrics.com/speed-blog/average-page-load-time-in-2021/

Martins, N., Cruz, J. M., Cruz, T., & Abreu, P. H. (2020). Adversarial Machine Learning Applied to Intrusion and Malware Scenarios: *A Systematic Review IEEE Access*, *8*: 35403-35419. https://doi.org/10.1109/ACCESS.2020.2974752

Mehta, N., Sanghavi, P., Paliwal, M., & Shukla, M. (2022, November). A Comprehensive Study on Cyber Legislation in G20 Countries. In *International Conference on Advancements in Smart Computing and Information Security* (pp. 3-23). Cham: Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-23095-0_1

Metcalf, L. B., & Spring, J. M. (2013). Passive Detection of Misbehaving Name Servers. *Carnegie-Mellon Univ Pittsburgh Software Engineering Inst*. http://resources.sei.cmu.edu/asset_files/TechnicalReport/2013_005_001_65284.pdf

MITRE. (2021). MITRE Adversarial Threat Landscape for Artificial-Intelligence Systems. https://atlas.mitre.org/

Moolayil, J. (2019). Learn Keras for Deep Neural Networks (1st Ed., 10.1007/978-1-4842-4240-7). Apress, California, U.S. https://doi.org/10.1007/978-1-4842-4240-7

Müller, A., & Guido, S. (2023). Introduction to Machine Learning with Python (1st Ed.). O'Reilly Media, California, U.S. ISBN: 10-9781449369897.

Nagunwa, T., Kearney, P., & Fouad, S. (2022). A machine learning approach for detecting fast flux phishing hostnames. *Journal of Information Security and Applications*, *65*, 103125. https://doi.org/10.1016/j.jisa.2022.103125

Nagunwa, T., Naqvi, S., Fouad, S., & Shah, H. (2020). A framework of new hybrid features for intelligent detection of zero-hour phishing websites. In *International Joint Conference: 12th International Conference on Computational Intelligence in Security for Information Systems (CISIS 2019) and 10th International Conference on European Transnational Education (ICEUTE 2019) Seville, Spain, May 13th-15th, 2019 Proceedings 12* (pp. 36-46). Springer International Publishing. https://doi.org/10.1007/978-3-030-20005-3_4

Nguyen, M. (2018). Illustrated Guide to LSTM's and GRU's: A step by step explanation. Retrieved September 10, 2019. https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21

Nicholson, C. (2019). A beginner's guide to neural networks and deep learning. *Retrieved January*, *30*, 2020. https://wiki.pathmind.com/neural-network

Pa, Y. M. P., Yoshioka, K., & Matsumoto, T. (2015). Detecting malicious domains and authoritative name servers based on their distinct mappings to IP addresses. *Journal of Information Processing*, *23*(5), 623-632. https://doi.org/10.2197/ipsjjip.23.623

Phi, M. (2018). Illustrated Guide to LSTM's and GRU's: A step by step explanation. *Towards Data Science*, *9*. https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21

Ray, S., Bansal, S., Gupta, A., Gupta, D., & Shaikh, F. (2017). Understanding Support Vector Machine algorithm from examples (along with code). *Analytics Vidhya*, *13*, 19. https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/

Salusky, W., & Danford, R. (2008). Know your Enemy: Fast-flux service networks. *The Honeynet Project*.

Silla, C. N., & Freitas, A. A. (2011). A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, *22*, 31-72. https://doi.org/10.1007/s10618-010-0175-9

Sillipo, R. A., & Maarit, W. (2019). 3 New Techniques for Data-Dimensionality Reduction in Machine Learning. https://thenewstack.io/3-new-techniques-for-data-dimensionality-reduction-in-machine-learning/

Sophos. (2017). Don't take the bait Sophos. https://www.cygnussystems.com/wp-content/uploads/2017/08/dont-take-the-bait.pdf

VanderPlas, J. (2023). Python Data Science Handbook (1st Ed.). O'Reilly Media. https://ae.oreilly.com/Python_Data_Science_Handbook_ch1

Verma, S. (2019). Understanding 1d and 3d convolution neural network| KERAS. *Medium, Towards Data Science*, *1*. https://towardsdatascience.com/understanding-1d-and-3d-convolution-neural-network-keras-9d8f76e29610

Weiss, N. (2020). Hierarchical Classification with Local Classifiers: Down the Rabbit Hole. *Towards Data Science Jan*, *20*.

Worcester, P. (2019). A comparison of grid search and randomized search using scikit learn. *Medium. Noteworthy-the Journal Blog*.

Xin, Y., Kong, L., Liu, Z., Chen, Y., Li, Y., Zhu, H., ... & Wang, C. (2018). Machine learning and deep learning methods for cybersecurity. *IEEE Access*, *6*, 35365-35381. https://doi.org/10.1109/ACCESS.2018.2836950

Yadav, S., Reddy, A. K. K., Reddy, A. N., & Ranjan, S. (2012). Detecting algorithmically generated domain-flux attacks with DNS traffic analysis. *IEEE/Acm Transactions on Networking*, *20*(5), 1663-1677. https://doi.org/10.1109/TNET.2012.2184552

Yiu, T. (2019). Understanding random forest. *Towards Data Science*, *1*, 1-11. https://towardsdatascience.com/understanding-random-forest-58381e0602d2