National College of Ireland

Detection of Clickjacking using Convolutional Neural Network

MSc Research Project
MSc in Cybersecurity

Kishore Hariram
Student ID: 21115737

School of Computing
National College of Ireland

Supervisor: Dr. Vanessa Ayala-Rivera

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Kishore Hariram

Student ID: 21115737

Programme: MSc in Cybersecurity **Year:** 2022

Module: MSc Research Project/Internship

Supervisor: Dr. Vanessa Ayala-Rivera

Submission Due Date: 01/02/2023

Project Title: Detection of Clickjacking using the Convolutional Neural Network

Word Count: 6457 **Page Count:** 19

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Kishore Hariram

Date: 29/01/2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Detection of Clickjacking using the Convolutional Neural Network

Kishore Hariram
21115737

Abstract

A clickjacking attack is one of the most serious and dangerous vulnerabilities in modern web applications. The aim of clickjacking is for an attacker to trick the user or a victim to perform a malicious action or an activity by embedding a hidden iframe that is placed transparently over the webpage. This makes the attacker hijack a click from the victim without the victim's knowledge. Even though clickjacking has gained much attention, it is still uncertain how and to what extent an attacker may use this practice to lure a victim and get personal information. This research, therefore, suggests a method for detecting malicious URLs that are susceptible to clickjacking attacks. This model uses Convolution Neural Network (CNN) technology to detect the suspicious iframe on a website, and HTML CSS property is utilized to highlight malicious iframe on the webpage. The "Dataset for Phishing website detection from the Data in Brief [1]" is used for the detection of a malicious iframe. The performance is evaluated by detecting the malicious iframe in minimal time with Convolution Neural Network (CNN) which results in good prediction of a malicious iframe.

1. Introduction

The world changed after the pandemic leading people to use the internet nowadays. This makes the attacker a good chance for exploiting the vulnerabilities of the people. One of the widely used attacks on the victims is Clickjacking. The attacker steals the click from the victim by placing a hidden iframe in the web pages and luring the victim to click on it.[2] This makes the victim perform activities of which they are unaware. The attacks try to transfer money, post on blogs and forums, redirect users to fake social networking sites to obtain passwords, and carry out several other malicious actions that may be initiated with a single click.[2] Web apps, which increase the dynamism and user-friendliness of websites, are built from a variety of static and dynamic HTML web pages. Many websites that are provided by various sources' content are combined to build web applications. The clickjacking vulnerability is being leveraged against the victims since online applications are so widely used. The clickjacking vulnerability was discovered in 2008 by Jeremiah Grossman and Robert Hansen.[3] To conduct the clickjacking exploit against the website, the harmful script is inserted into invisible iframes that are displayed above the actual site or other website components. To make the iframe invisible and hardly perceptible, the CSS (Cascading Style Sheet) attribute's transparency level has been set to a very low value.[4]

Figure 1 shows the perfect example of a clickjacking attack. The original webpage has the payment transaction page. Over this page there is a hidden iframe is embedded mentioning to win the lottery. This leads to steal the click and transfer the amount to attacker's account. In this manner, the attacker can alter the website such that the attack can steal personal data, bank information, transfer money, etc.

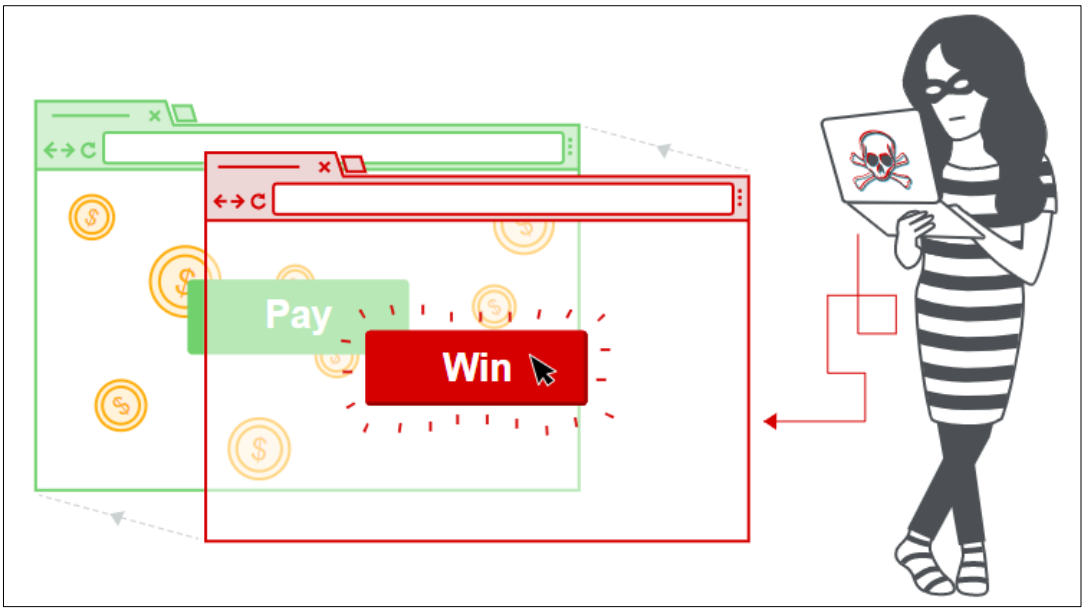

Figure 1: Clickjacking attack is performed by embedding hidden iframe in original webpage*[5]*

The intention of the adversary is completely unpredictable, which is the way in which the adversary can still use clickjacking in the execution of the exploits. The "Chameleon" virus was designed to click on adverts put on websites, deceiving the developer of the advertising by luring in the products, but in reality, the adversary was getting charged based on the number of clicks on the commercials.[6]

To address the issue mentioned above, my research question is how effectively clickjacking can be detected with minimal time using the Convolution Neural Network (CNN) Model using image processing as the clickjacking should be identified immediately before the user or the employees fall as a prey.

The goal of this research is to locate any potentially harmful connections that an adversary may have placed to a webpage to steal the victim's personally identifiable information or install malware on the user's machine. The links are extracted by a feature extraction method using a CNN model. Furthermore, a list of dangerous URLs and legitimate URLs is placed on the webpage.

The suggested approach may be employed with a variety of websites whenever malicious hyperlinks can be identified using machine learning approaches, and if clickjacking-related URLs can be identified and shown using CSS attributes. Using real-time internet factors including SSL, website traffic, domain names, and web hosting providers, this algorithm verifies URLs.

The CNN approach is used to predict potentially harmful URLs. The convolution neural network is rapid and efficient for binary classification. The CNN classifier has been trained using the phishing sites dataset, which has variables that are specific to dangerous behavior and yields results with minimal time. In terms of learning period, the CNN classifier beat supervised machine learning methods, according to the prediction outcome.[7]

The literature review step involves investigating, analyzing, and contrasting a range of tools, techniques, theories, and algorithms—from non-machine learning to machine learning—in order to choose the strategy that best addresses our research issue.

2. Literature Review

Several studies have been examined to study how the work has been made in the field and to check the challenges which were faced during the examinations. Throughout the research, different types of tools and techniques were used- are compared, from the ones which don't depend on machine learning to the ones which depend on.

Non-Machine Learning studies:

To study the detection of clickjacking, the author[2] has created two browser extension names “ClickIDS” and “NoScript”. The ClickIDS check for any click buttons embedded over the web content to hijack a click. The NoScript creates an alert when there is a component overlays over the web content. This study does a good job of detecting the clickjacking on the clickable actions, but one of the major drawbacks of this approach is that it fails to identify assault on non-clickable component. Most important fact is NoScript extension is browser-dependent. The author Krishna Chaitanya T.[8] identify the challenges from the author[2] as identifying the clickable elements in a translucent website. To overcome that author used Java and CSS were used to create the Chrome extension with the same origin. This plugin is put on the assault webpage so that it can be tested for real-time accuracy. The opaque Iframe that is included on the webpage is recognized by this plugin, but because it is merely a NoScript imitation, it does not have all of NoScript's features and may not be entirely useful for identifying additional clickjacking attempts.

Although same origin policy is implemented, clickjacking attack was carried out. To overcome this issue author [9] suggested a method a that help in identifying clickjacking by specifying statistical analysis on the page’s back end. In the first, a hyperlink is placed haphazardly on the page as there are no other page elements, and in the second, unsuccessful and successful clicks are recorded. A further optimization technique called BUCKETISATION saves all signups as a single click and aggregates them into buckets for data analysis. The buckets on social networking platforms may be used as a user's identifier when utilizing the Follow and Like symbols. The limitation in the method was identified as this can’t be used in complicated applications.

K.Joylin Bala also conducted a study on identifying clicking parameters.[10] Use of 3 elements (Tracking elements, Detection elements and action elements) the clickjacking attack was detected. The research proposes that before loading the page, the monitoring element stores and tracks the clickable component’s parameter. Post the page rendering, the clickable parameter are verified for the webpage display. if there is any change in verification of legitimate click, the system creates an alarm. The critics question the ability of identifying the clickable elements are not visible.

The author made a browser based mitigation[3] to avoid clickjacking attacks on attempting to associate using Facebook plugins. Two Chrome Browser extensions called Zscaler Likejacking Prevention (Zscaler) and Cursor Spoofing and Clickjacking Prevention are created only when a victim clicks the "like" or "follow" buttons on a website (CSCP). Any web plugins that are hidden but nonetheless exist on the webpage are identified by Zscaler. The challenge was it can be used in social media websites.

In order to identify drive-by downloading assaults, the webpage's JavaScript code is dynamically evaluated by the creator Marco Cova [11]. Drive-by-download attacks are online application attacks. The plan is to search for and check over JavaScript code. The development

of the JSAND tool allows for the detection of malicious JavaScript code based on characteristics, the numbers of which are evaluated using machine learning techniques and anomaly testing processes from reliable websites. The accuracy is assessed over more than 140,000 locations. Nevertheless, a lot of clickjacking assaults use in-depth dynamic simulation of the websites. In terms of identifying clickjacking assaults, this study is not noteworthy.

The study's author [12] devised the "Prophiler" technique to analyze the website on a larger scale. Web-based page content and URL-based attributes are extracted from the HTML pages that the web search engine has gathered using the image retrieval technique. The process is really inspirational, but it takes time because the result is verified twice.

We used a machine learning technique to identify clickjacking attacks as a result of how effectively this strategy works for detecting clickjacking attempts on Facebook.

Machine Learning Studies:

Youngsang Shin [13] either classifies the URL as spam or as legitimate. The URLs are extracted from the post's comments. The SVM classifier will put the URL in one of many categories in accordance with the URL types it has collected. This method imposes limitations on the retrieval of links from comments.

The author [14] of this work offers a machine learning method for locating phishing websites. Based on the author, categorization may be done using machine learning in a trained, untrained, or semi-trained manner. The harmful URL is discovered and utilized as feed into the machine learning algorithm based on the characteristic gathered and translated into input vector. The model is trained using data gathering from blacklisted URLs, and depending on this learning, it decides if a URL is safe or harmful. The training machine learning approach is used throughout the evaluation procedure. By taking attributes from the URL and separating them into four distinct subcategories, successful classification is accomplished.

Dharmaraj Patil. conducted research. Using several classifier algorithms,[15] discovered a hazardous website. Several trained machine learning algorithms were used to separate the links into the several assaults. Due to the scale of the website and the range of techniques and tools utilized throughout the feature extraction process, this technique poses a difficulty.

Sha_ Ahmed [16] identified the malicious site in real-time using trained and untrained machine learning techniques. The data is gathered using Alexa and the dataset from the phishing tank. The URLs are separated into phishing and real URLs after being gathered from their respective sources and converted to vector space representations. Comparatively speaking, the SVM performed the best of all the algorithms.

Anjali B. Sayamber [17] compiled data on safe URLs, safe URLs, and phishing from a number of sources. Comparing this model's performance against that of SVM and Bayesian networks. The Nave Bayes algorithm performed well at classifying the URLs into separate groups like spam, safe, and phishing. Article [17] gathers hazardous URLs from jwspamspy, phishing tank, and DNS-BH in contrast to paper [16], which solely used data from the phishing tank, which only contains phishing data. The quality of the models in the publications [16] [17] reveals varied findings as a result, depending on the mixture of datasets used.

The URLs are collected from several databases where they are classified as great and prohibited. Links are also retrieved from the prior works' classification [15] [16] [17]. The

author has proposed a technique to extract online material from an HTML webpage [15]. The Html document is changed, the features are extracted, and a rule is created to gather the useful information from the Html document in the Document Object Model tree.

The author gave several tools and libraries for data crawling in the work [18], where she explored various web scraping approaches for capturing HTML websites by converting web-based unstructured data into structured data.

The researcher created the malicious URL and the safe URL for the article using the data from the DMZ and the phishing tank [19]. Python libraries are applied in order to retrieve the data from the URL. The MATLAB Neural Network is used in conjunction with a number of taught machine learning algorithms to categories the URL. The output of the neural network and the supervised machine learning are both examined. The precision of the decision tree supervised machine learning approach is 96.18 percent, outperforming other supervised learning algorithms. Over 98.16 percent of the accuracy of the MATLAB neural network. The MATLAB neural network has fared well in comparison.

To accurately predict website messages, Andrew H. Sung and colleagues [26] use a range of machine learning algorithms, including SVM, K-means, neural networks, and the Self-Organizing Map model. Several tests are conducted using various models to confirm the model's accuracy. The accuracy percentage of the neural network model used to identify phishing emails is above 97.99 percent.

Hüseyin Gökal [4] used the ELM classifier for the classification step. Extreme machine learning classification is a type of neural network. This classifier categorizes the malicious URL based on 30 different extracted properties. The SVM and Naive Bayes As a consequence, the ELM neural network is compared against supervised machine learning techniques. In comparison to this technique, the ELM has shown the best performance with a prediction performance of 95.34 percent. The verification result, which is the percentage of data that properly approximates all of the dataset's data, is used to gauge the model's performance.

The Cursor Spoofing and Click Jacking Prevention (CSCP) browser plugin is suggested by the paper's author, Ubaid Ur Rehman[3]. The CSCP Google Chrome extension provides web-based protection against clicking on the sensitive user interface that is embedded. Threats to pointer and visual integrity are protected from by the addon. The CSCP has a success rate of 56% to 67% in preventing the recently proposed and current Clickjacking assault.

The author of this research, Yasin Sönmez [4], describes the traits of phishing assaults, and we provided a classification model to classify phishing attacks. This technique consists of components for classification and feature extraction from webpages. In the feature extraction process, we have provided precise rules for extracting phishing features, and these criteria have been applied to gather features. These characteristics were identified by using SVM, NB, and ELM. The ELM had the highest accuracy rating and used six different activation modes.

4. Research Methodology

The below Clickjacking detection diagram is implemented in this research to detect iframe with malicious hyperlinks which lead to a clickjacking attack possible.

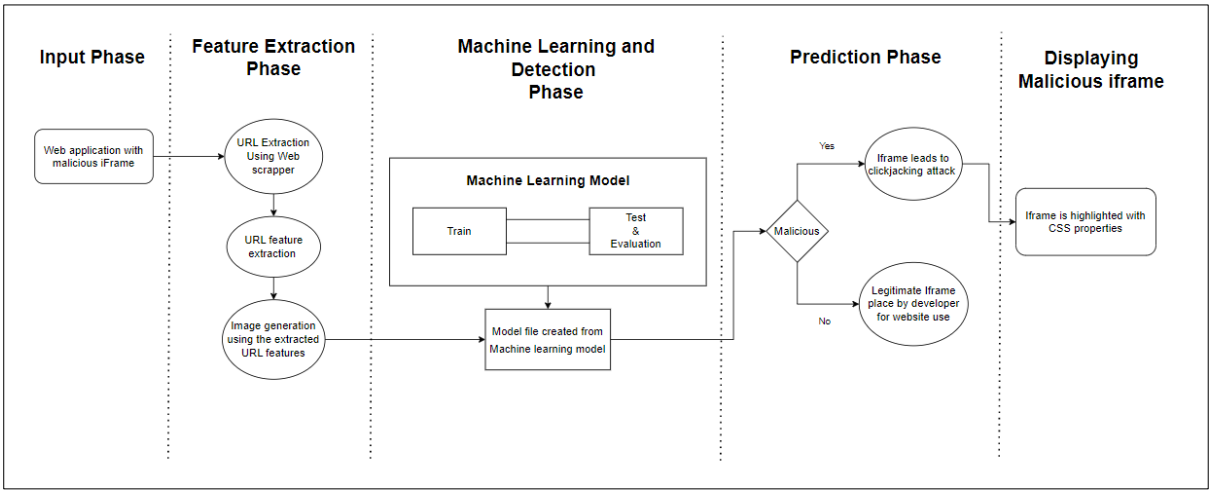

Figure 2: Flow Diagram

As seen in Figure 2, the system is divided into 5 phases namely the Input phase, Feature extraction Phase, Machine learning and Detection Phase, Prediction Phase, and the Displaying Malicious iframe. The website with the embedded URLs is given as the input. The URLs in the webpage are extracted using a URL extractor. Further, the URLs are identified using the URL feature extraction and converted into images that will go as input into the model file from the machine learning model. Prediction is made using the image from the feature extraction phase and malicious iframe are further displayed on the webpage.

The attacker's ultimate aim is to target the victim using a social engineering method. The attacker tricks victim to redirect or visit malicious domain or website with different methods.[5] One method includes using double framing or hiding a hazardous link in a picture to deceive the visitor into viewing the phishing site. In turn, the user is led to the malicious website when they click on an iframe that carries a malicious website.[17]

This research provides information on how a client-side clickjacking attack may be identified. The major goal is to find these dangerous URLs on the website and to make users to them using two techniques, namely Convolutional Neural Network and HTML's CSS feature.

The vulnerability of Clickjacking is detected using Convolution Neural Networks (CNN). Why, then, CNN? CNN can handle a wide range of unstructured data types. Numerous malicious URLs are sent into the network to evaluate CNN's accuracy fast and depending on the research. Other neural networks, in comparison, divide operations into several nodes, which may take a long time to complete to achieve accuracy. For processing 2D input, CNN's architecture, which uses a 2D convolution layer, is perfect. A 2D convolution layer produces a single output by splitting a 2D matrix into several smaller 2D matrix parts. Other neural networks often just employ one decision tree.

4.1 Extraction of malicious URL

A Web scraper is a tool used to collect all the URLs on the webpage. Other methods of data collection exist, such as API (Application Interface Programming), yet this method does not use API because not all web apps use it. The web scraping technique is used to parse HTML webpages, which entails removing the necessary data from the HTML page, such as the URL, content, keywords, email, etc. [18]

4.2 Malicious URL extraction and Pre-Processing of the data

Once we have collected all of the hyperlinks in the HTML, we can identify the various types of properties of these URLs (acquired via site scraping). Below is a list of the site attributes that need to be obtained.

Feature Pre-Processing is the method in which the data is transformed into a number vector space before the information is given as input to the machine learning model.

4.2.1 Features of the URL depending on the malicious webpage scripts: [4]

- **Request URL:** This property determines whether the website contains items from different domains like, videos and images.
- **URL with anchor:** By this functionality, we can able to check whether the website's domain name and the <a> tag matches.
- **Hyperlinks on the HTML tags such as script, link, media:** This function is used to identify all the tags on the website belongs to the same domain.
- **Handler of the Server Form:** This feature checks to determine if a domain name given to SFH is the same as the domain name of another webpage.
- **Email Information transmission:** If a webpage provides client-side or server-side scripting like "mailto()" or "mail()," it may be assumed that it is malicious.
- **Malicious URL:** Webpage's information is retrieved from the database of the WHOIS.

4.2.2 Address bar attribute

- **URL with IP:** The website uses the IP instead of a domain name, as most of the malicious website comes with the IP.
- **URL length**: Describes the length of the URL, and check any unusual size of the URL.
- **Use of Shortening tool for URL:** This check whether any use of any tools used for shortening the URL.
- **Web content uses any '@' symbol:** Checks any use of '@' symbol in the web content.
- **Web content uses any '//' symbol:** Checks any use of '//' symbol in the web content.
- **Domain name used any '- 'symbol:** Check any use of '-'symbol in the web content.
- **Multiple subdomain use:** Check if there are any multiple subdomains used in the parent domain.
- **Site is hosted with HTTPS:** Check whether the domain has a SSL certificate.
- **Use of non-regular ports:** Check any use of unusual ports or service used.

4.2.3 Domain Attributes

- **Top Level Domain (TLD):** Check the website using top level domains.
- **Time To Live (TTL):** Checks the age of the domain persistence.
- **DNS Records:** Check the domain's DNS records.
- **Website's Traffic:** The traffic of the domain is examined whether it has any malicious content or not.

4.2.4 Extraction of Attributes from HTML and JavaScript resources:

- Click of the mouse verification.
- Personalization of the status bar.
- Unusual trigger of Popups.
- Unusual redirection to a different website.
- Website is redirected to different domain using an iframe.

4.3 Convolutional Neural Network

Convolution neural network technology is currently the most valued, highly used type of artificial intelligence because it is more accurate than other intrusion neural networks in detecting intrusions. This approach uses a computing model that incorporates many convolutional layers with additional deep learning layers, which may or may not be fully connected. Some of the essential elements employed by CNN to allow flexible and automated learning of spatial information hierarchy are convolution layers, pooling layers, and fully connected layers. CNN employs fewer parameters and makes fewer computations as a result of parameter pooling.

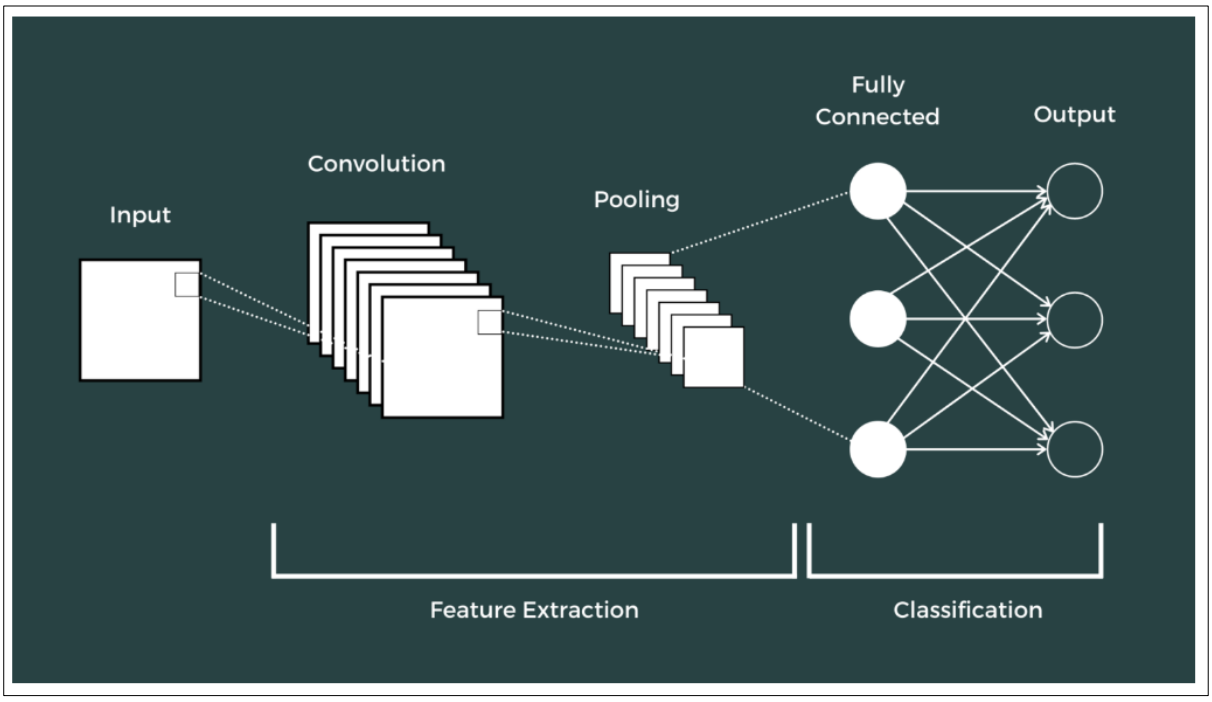

Figure 3: Processing the input in CNN model

When comparing CNNs to other neural networks, CNNs have the benefit of being able to recognize data without human assistance or input. Since the CNN method can analyze a large quantity of unstructured data, it is more frequently utilized in natural language processing and image analysis. [6]

4.4 Highlighting the feature of malicious hyperlink in website

This section of the research makes any malicious URL that have been identified and are being used for clickjacking visible on the website. As the machine learning system detects the dangerous URL hyperlinks, a list of these dangerous web links is provided. Attackers may use the opacity value to attack by taking advantage of the HTML element and the Iframe attribute. [20]

5. Convolution Neural Network Core Architecture

The data is pre-processed through a number of phases. Below is a description of how the pre-processing happens.

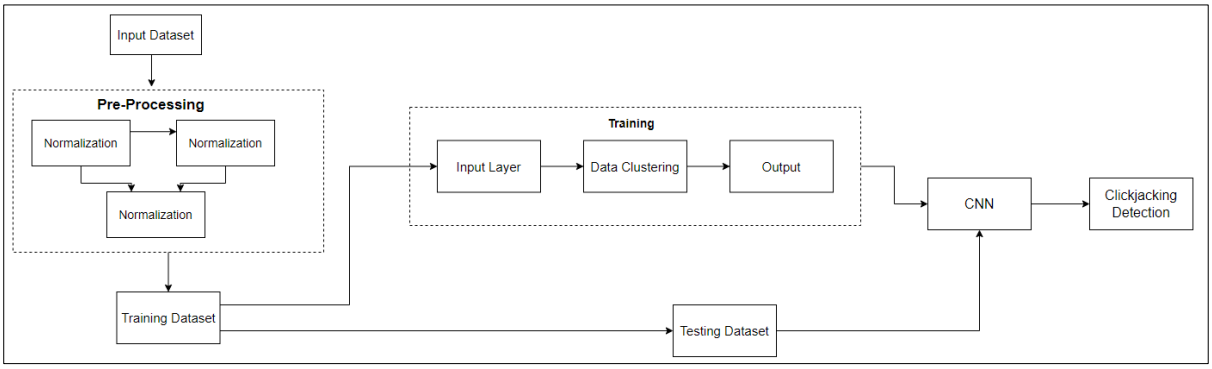

Figure 4: CNN Model

Initially, the dataset is sent for the preprocessing phase where the dataset undergoes normalization. In normalization, the dataset is filtered and removes the void values in the dataset are. If the dataset is not filtered properly there is a chance the CNN slows down the learning process. Once normalization is done the dataset is later split into 2 as test and train datasets. The training dataset is first clustered depending on the features and then produces the output which will later be converted into images. The image from the test and train phases will be sent into the CNN model and the detection is made based on the images. After evaluating the result, the performance of the model is checked.

- **Dataset Description**: The characteristics from the sets of website URLs that make up the data are included. Overall, there are 111 features in the data, 96 of which are taken directly from website addresses, and the other 15 features are taken directly from unique Python code. The publicly available phishing dataset " Dataset for Phishing website detection from the Data in Brief [1]" is used to train the algorithm to identify whether a hyperlink is genuine or phishing for study purposes.
- **Dataset Training:** The data set is trained on the Phishing website detection dataset. Where the dataset is split into 60% training and 40% for testing. The model learns the data and performs accordingly to predict the result based on the extracted URL features.

6. Implementation

The many methods and instruments utilized to carry out this research are covered in this section. The majority of the modules, including those for machine learning, web scraping, feature extraction, and URL prediction, utilized Python. The range of libraries and frameworks that Python provides is one of its greatest advantages. A static web page is created using HTML code for this research, and a few iframes are added to it. Since the opacity of these iframes is set to "0," the victim won't be able to see them.
The web contents are specifically URLs extracted from the webpage URLs using a web scraping technique. The URLs are taken extracted from the webpage using Python. Python has developed several libraries that use BeautifulSoup and request to extract URLs from webpages. The library for the Python programming language is quite helpful and significant since it imports all of the related functions with those libraries simply by loading it. The get() method is utilized to generate a get request to the webpage for the information and to obtain the server's

response. Some libraries don't come with preloaded libraries. The command "pip install LIBRARY NAME" can be used to install it.
The feature extraction method is carried out which takes the input one at a time of all the URLs from the web page. The output is then converted to a dataset and converted to the image which is given as input to the CNN for the prediction of whether the URL is malicious or not.
The python libraries like pandas, NumPy, sklearn, and Tensorflow were used in Convolutional Neural Networks. In handling the array of data, the NumPy works the best. To eliminate the "null values" and "NA" values the data cleaning process is carried out in the data preprocessing phase.
Once the preprocessing of the data is completed, the dataset is divided into 2 train and test data with 60% and 40% respectively. The model is trained on the training dataset and used for future prediction. The epochs for the model were set to 20 rounds with a batch size of 200. At end of CNN training, the model file is created which will be further used for URL predictions. The images created from feature extraction are given as input into the model file to classify the link as malicious or not.
Once the URL is predicted as malicious, the border of the iframe is highlighted by increasing the opacity to the maximum level to show the iframe is malicious and there is a chance of a clickjacking attack on the webpage.

7. Evaluation

There are several matrices used for determining the result such as accuracy, precision, F1 score, and recall which are shown below. [21]

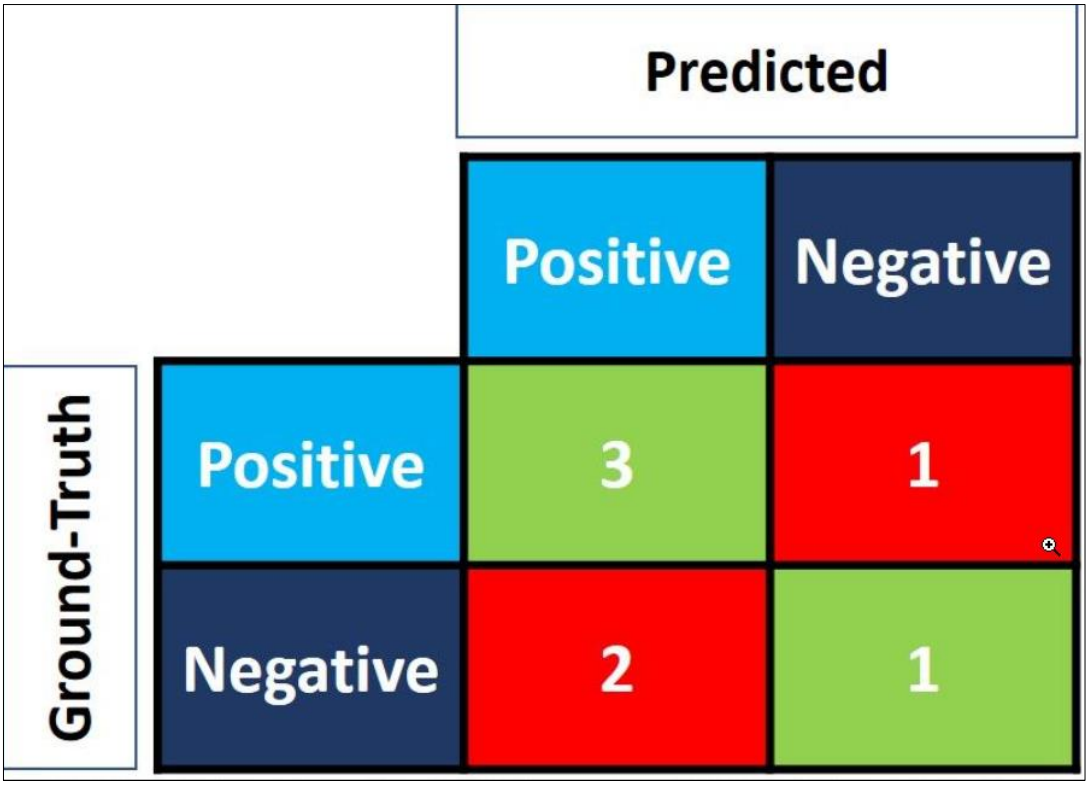

Figure 5: Accuracy prediction

The matrix has four components represents four parameters which determines the model is accurate or inaccurate. Each component has 2 variables:

- True or False
- Positive or Negative

In summary, First word always will be False when the prediction is incorrect. If not the model results as true. The objective is to maximize the True(True Positive and True Negative) and minimize the False(False Positive and False Negative). The above matrix mentions the below:

- **Top Left (True Positive**): Number of time the model predicted the Positive sample as Positive?
- **Top Right (False Negative):** Number of times the model predicted the Positive sample as Negative?
- **Bottom-Left (False Positive):** Number of times the model predicted the Negative sample as Positive?
- **Bottom-Right (True Negative):** Number of times the model predicted the Negative sample as Negative?

7.1 Calculations of CNN

7.1.1 Accuracy

Calculating the proportion of accurate predictions to all other predictions uses this information.

$$Accuracy = \frac{TRUE_{Positive} + TRUE_{Negative}}{TRUE_{Positive} + TRUE_{Negative} + FALSE_{Positive} + FALSE_{Negative}}$$

7.1.2 Precision

The precision is used to determine the accuracy of a classified sample that is positive. Precision is calculated using both the total number of samples classified as positive and the number of positive samples (either correct or incorrect).

$$Precision = \frac{TRUE_{Positive}}{TRUE_{Positive} + FALSE_{Positive}}$$

7.1.3 Recall

Divided by the percentage of positives that were correctly identified as positive, the total number of three positive samples is used to calculate recall.

$$Recall = \frac{TRUE_{Positive}}{TRUE_{Positive} + FALSE_{Negative}}$$

7.1.4 F-Measure

It is a periodic technique for recall and precision.

$$\text{F- Measure} = 2 \frac{(Precision \times Recall)}{(Precision + Recall)}$$

- **True Positive**: Total number of samples identified correctly.
- **True Negative**: Total number of clickjacking class correctly identified
- **False Positive**: Total number of samples incorrectly identified.
- **False Negative**: Total number of samples incorrectly identified.

7.2 Case Studies

The main aim of this study is to get a better prediction result of whether the URLs embedded with the iframe over the website are malicious or not. This study is made by using the trial-and-error method by giving different combinations of values to the CNN model and seeing how it performs and produces the result. The model is given with 4 test cases with 2 cases having less train value and 2 cases with standard values for the train. In this study, it was found that case 4 gives a better prediction of URLs.

Study 1:

The model to check how the CNN computes the dataset and gives the result. I gave the values to the model for training as below.

Test = 80, Train= 20, epoch=40 and batch =200

From this combination of data, the model gave the below train result.

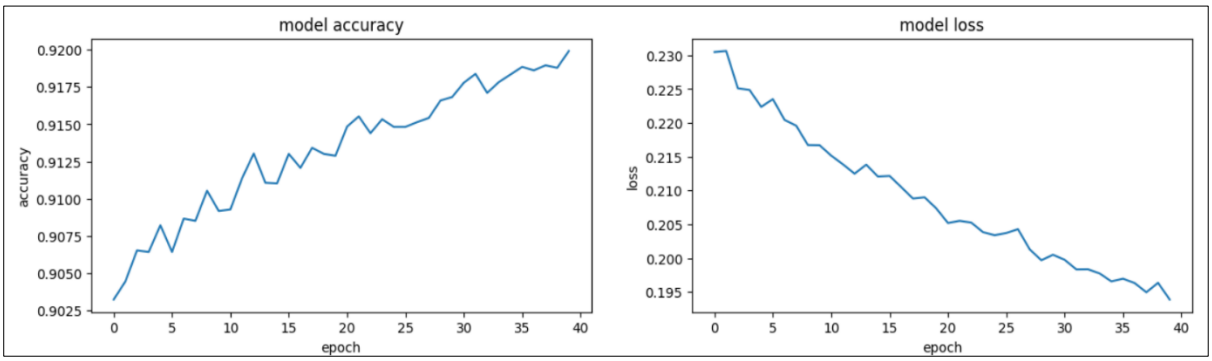

Figure 6: Study 1

The above Fig.6 shows although there is good accuracy achieved the model faces lots of data loss and failed to predict the malicious URLs. This is because the dataset was trained with little data for the training and lots of data loss on training the data.

	url	phishing
0	http://reneerojanaro.com/	0
1	https://pastehtml.com/view/b2i243gkw.html	0
2	https://octopus-prediction.com/	0
3	https://www.dgvaishnavcollege.edu.in/	0
4	http://nihahaw5.beget.tech/	0

Figure 7: Case 1 prediction

Fig.7 shows the prediction results of the URLs in websites. The model failed to predict the URLs as it was undertrained.

Study 2:

The model was given with the below values for training for the case 2.

Test =80, Train=20, epoch=40 and batch=300

From this combination of values, the CNN gave results as below.

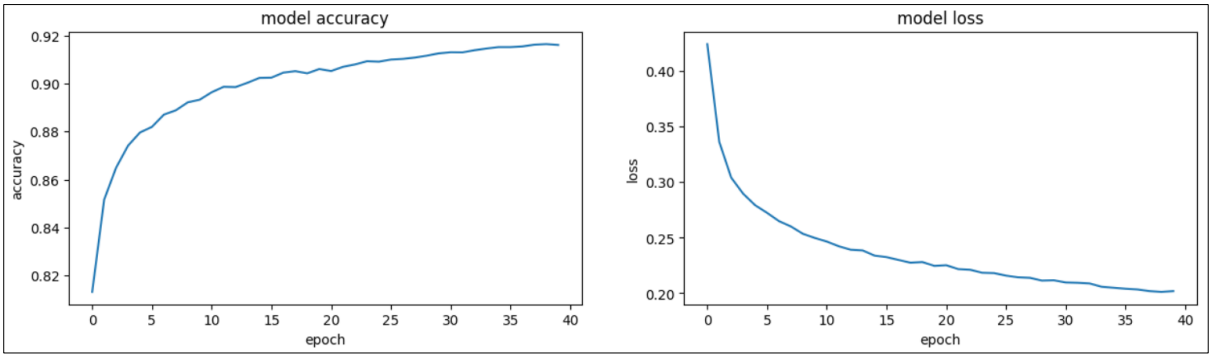

Figure 8: Study 2

The above Fig.8 shows that the model gives good accuracy and the data loss is also less in this combination. The model is given with a 300-batch size for processing the dataset. Even though the accuracy is good and data loss is less the model is undertrained as there was not enough data allocated for training but still this combination of the values used to train the CNN gives a prediction result of finding 1 URL as phishing as shown in Fig.9. This is because the model is given with fewer data to train and the model went under trained.

	url	phishing
0	http://detkreimeier-fotografie.de/	0
1	https://pastehtml.com/view/b2i243gkw.html	1
2	https://octopus-prediction.com/	0
3	https://www.dgvaishnavcollege.edu.in/	0
4	http://nihahaw5.beget.tech/	0

Figure 9: Case 2 Prediction

As from the result of case 2, I thought of giving a standard value in study 3.

Study 3:

The model is given with the below values for training.

Test= 30, Train=70, epoch = 10 and batch=100

From the combination of the result, the result from the CNN gives below.

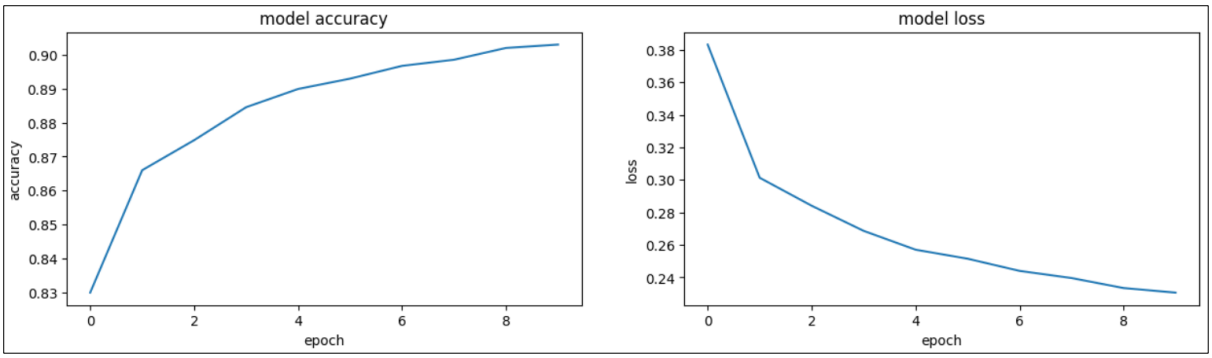

Figure 10: Study 3

From Fig.10, the CNN model gives an accuracy of prediction of almost 90% which is less than the study of 1 and 2 and there is not much of a data loss but the model doesn't predict the URLs where only 1 URL was predicted as it was undertrained. This case went undertrained even though the model is given with sufficient data for training but the amount of data per batch and the number of iterations given for the model was less. This is the reason the model didn't learn well.

	url	phishing
0	http://detkreimeier-fotografie.de/	0
1	https://pastehtml.com/view/b2i243gkw.html	1
2	https://octopus-prediction.com/	0
3	https://www.dgvaishnavcollege.edu.in/	0
4	http://nihahaw5.beget.tech/	0

Figure 11: Case 3 prediction

From this result, it understood that the model needs more data to train and takes more iteration for learning to predict the URLs. Based on this understanding the case study 4.

Study 4:

The model is trained with the below combination of values.

Test = 40, Train= 60, Epoch = 40 and Batch = 200

From the combination value, CNN gave the results below.

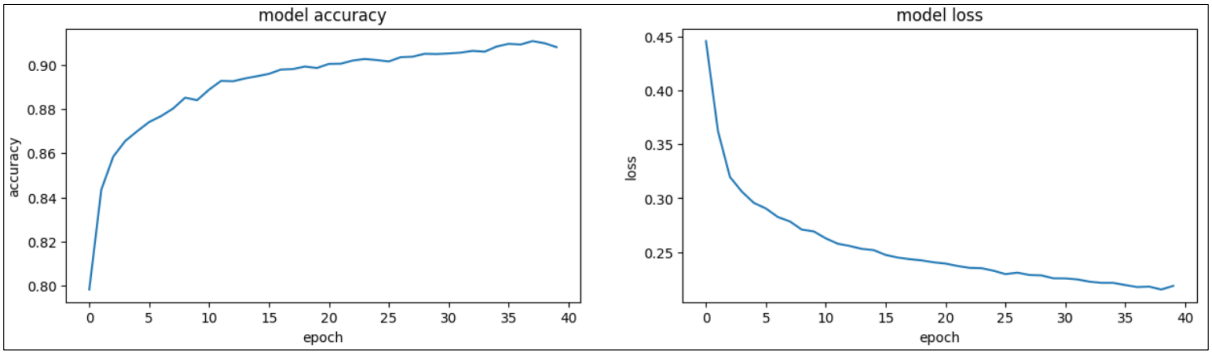

Figure 12: Study 4

From this result, the model gave an accuracy of prediction of 91% which was less compared to cases 1 and 2 but gave a very good prediction of the URLs as shown in the below results.

	url	phishing
0	http://detkreimeier-fotografie.de/	1
1	https://pastehtml.com/view/b2i243gkw.html	1
2	https://octopus-prediction.com/	1
3	https://www.dgvaishnavcollege.edu.in/	0
4	http://nihahaw5.beget.tech/	0

Figure 13: Case 4 prediction

The model was trained well as the model was given sufficient data for training.

8. Discussion

The Convolutional Neural Network is compared with different test cases which was discussed in the above section. From the evaluation results of the above test cases, the accuracy came to 91% within very less time period of 56 seconds.

9. Conclusion and Future Works

The attacker takes advantage of the malicious URLs as the best piece of leverage to get control on the victim's system by making the user to get victimized. This leads to the clickjacking attacks by the attacker to get information about the user, redirect to different website, and collect PII data. This research is made to detect the clickjacking attacks which uses malicious link using the Convolution Neural Network model and highlight on the web page. Once the implementation is carried out, the iframe with malicious link is identified successfully using the aid of Convolutional Neural Network and then the opacity is maximized to highlight on the webpage. All over accuracy obtained from the CNN module was 91% and time required to identify is 0.5 seconds. So overall the provided solution is effective to find the malicious iframe.

In most of the real time scenarios, more clickjacking attacks are performed using the advertisements in the websites by embedding a malicious iframe over the webpage. Detection of advertisement link on an iframe can be made for the future work. For the real time, the advertisement link can be hosted by embedding in a webpage using a dedicated server and can be detected using the machine learning model. Furthermore, these can also be detected using browser extension which can detect the clickjacking attacks.

Reference:

[1] G. Vrbančič, I. Fister, and V. Podgorelec, "Datasets for phishing websites detection," *Data in Brief*, vol. 33, Dec. 2020, doi: 10.1016/j.dib.2020.106438.

[2] M. Balduzzi, M. Egele, E. Kirda, D. Balzarotti, and C. Kruegel, "A solution for the automated detection of clickjacking attacks," in *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, New York, NY, USA, Apr. 2010, pp. 135–144. doi: 10.1145/1755688.1755706.

[3] U. U. Rehman, W. A. Khan, N. A. Saqib, and M. Kaleem, "On Detection and Prevention of Clickjacking Attack for OSNs," in *2013 11th International Conference on Frontiers of Information Technology*, Dec. 2013, pp. 160–165. doi: 10.1109/FIT.2013.37.

[4] Y. Sönmez, T. Tuncer, H. Gökal, and E. Avcı, “Phishing web sites features classification based on extreme learning machine,” in *2018 6th International Symposium on Digital Forensic and Security (ISDFS)*, Mar. 2018, pp. 1–5. doi: 10.1109/ISDFS.2018.8355342.
[5] “What is Clickjacking? Tutorial & Examples | Web Security Academy.” https://portswigger.net/web-security/clickjacking (accessed Dec. 07, 2022).
[6] “Difference between ANN, CNN and RNN,” *GeeksforGeeks*, Jun. 28, 2020. https://www.geeksforgeeks.org/difference-between-ann-cnn-and-rnn/ (accessed Dec. 07, 2022).
[7] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, “Convolutional neural networks: an overview and application in radiology,” *Insights Imaging*, vol. 9, no. 4, Art. no. 4, Aug. 2018, doi: 10.1007/s13244-018-0639-9.
[8] T. Krishna Chaitanya, H. Ponnapalli, D. Herts, and J. Pablo, “Analysis and Detection of Modern Spam Techniques on Social Networking Sites,” in *2012 Third International Conference on Services in Emerging Markets*, Dec. 2012, pp. 147–152. doi: 10.1109/ICSEM.2012.28.
[9] “New Research and Progress Against Clickjacking at the W3C,” *The Security Practice*. https://www.thesecuritypractice.com/the_security_practice/2012/05/new-research-and-progress-against-clickjacking-at-the-w3c.html (accessed Dec. 06, 2022).
[10] “Bala, K., 2022. EFFECTIVE APPROACH TO DETECT CLICKJACKING ATTACKS.” Accessed: Dec. 02, 2022. [Online]. Available: https://www.semanticscholar.org/paper/EFFECTIVE-APPROACH-TO-DETECT-CLICKJACKING-ATTACKS-Bala-Raj/873defae4652d90659eeddaaaa252b8d3bff0326
[11] M. Cova, C. Kruegel, and G. Vigna, “Detection and analysis of drive-by-download attacks and malicious JavaScript code,” in *Proceedings of the 19th international conference on World wide web*, New York, NY, USA, Apr. 2010, pp. 281–290. doi: 10.1145/1772690.1772720.
[12] D. Canali, M. Cova, G. Vigna, and C. Kruegel, “Prophiler: a fast filter for the large-scale detection of malicious web pages,” presented at the Proceedings of the 20th international conference on World wide web, Mar. 2011, p. 197. doi: 10.1145/1963405.1963436.
[13] Y. Shin, S. Myers, M. Gupta, and P. Radivojac, “A link graph-based approach to identify forum spam,” *Security and Communication Networks*, vol. 8, Jan. 2015, doi: 10.1002/sec.970.
[14] D. Sahoo, C. Liu, and S. C. H. Hoi, “Malicious URL Detection using Machine Learning: A Survey.” arXiv, Aug. 21, 2019. doi: 10.48550/arXiv.1701.07179.
[15] “[PDF] Survey on Malicious Web Pages Detection Techniques | Semantic Scholar”, Accessed: Dec. 06, 2022. [Online]. Available: https://www.semanticscholar.org/paper/Survey-on-Malicious-Web-Pages-Detection-Techniques-Patil-Patil/1f67b724614a50f90688d1db450eed7916d6dad8
[16] S. Ahmed, “Real time detection of malicious webpages using machine learning techniques,” doctoral, London Metropolitan University, 2015. doi: 10/1/Shafi%20Ahmed%20-%20PhD%20Full%20Thesis.pdf.
[17] A. B.Sayamber and A. Dixit, “Malicious URL Detection and Identification,” *International Journal of Computer Applications*, vol. 99, pp. 17–23, Aug. 2014, doi: 10.5120/17464-8247.
[18] “An Overview On Web Scraping Techniques And Tools | International Journal on Future Revolution in Computer Science & Communication Engineering.” https://www.ijfrcsce.org/index.php/ijfrcsce/article/view/1529 (accessed Dec. 07, 2022).
[19] “Categorization of Phishing Detection Features and Using the Feature Vectors to Classify Phishing Websites - ProQuest.”

https://www.proquest.com/openview/7bad68b82d74ae70c5089bf66302f709/1?pq-origsite=gscholar&cbl=18750 (accessed Dec. 07, 2022).
[20] H. Selim, S. Tayeb, Y. Kim, J. Zhan, and M. Pirouz, "Vulnerability Analysis of Iframe Attacks on Websites," in *Proceedings of the The 3rd Multidisciplinary International Social Networks Conference on SocialInformatics 2016, Data Science 2016*, New York, NY, USA, Aug. 2016, pp. 1–6. doi: 10.1145/2955129.2955180.
[21] "Accuracy, Precision, and Recall in Deep Learning," *Paperspace Blog*, Oct. 12, 2020. https://blog.paperspace.com/deep-learning-metrics-precision-recall-accuracy/ (accessed Dec. 06, 2022).
[22] M. Riva, "Batch Normalization in Convolutional Neural Networks | Baeldung on Computer Science," Oct. 29, 2020. https://www.baeldung.com/cs/batch-normalization-cnn (accessed Dec. 06, 2022).
[23] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights Imaging*, vol. 9, no. 4, Art. no. 4, Aug. 2018, doi: 10.1007/s13244-018-0639-9.
[24] U. U. Rehman, W. A. Khan, N. A. Saqib, and M. Kaleem, "On Detection and Prevention of Clickjacking Attack for OSNs," in *2013 11th International Conference on Frontiers of Information Technology*, Dec. 2013, pp. 160–165. doi: 10.1109/FIT.2013.37.
[25] K. Nethra, J. Anitha, and G. Thilagavathi, "WEB CONTENT EXTRACTION USING HYBRID APPROACH.," *ICTACT Journal On Soft Computing*, vol. 4, no. 2, 2014.
[26] R. Basnet, S. Mukkamala, and A. H. Sung, "Detection of Phishing Attacks: A Machine Learning Approach," in *Soft Computing Applications in Industry*, B. Prasad, Ed. Berlin, Heidelberg: Springer, 2008, pp. 373–383. doi: 10.1007/978-3-540-77465-5_19.