PJET, Vol. 2, No. 1, February 2024: 44-65

Poly Journal of Engineering and Technology

journal homepage: https://journals.bdu.edu.et/index.php/PJET/index

ORIGINAL ARTICLE

Detection of Phishing Websites Using Structural Similarities Amongst Attacks

Mgdelawit Bewketu[1], Esubalew Alemneh Jalew[2,*]

[1] Faculty of Computing, Bahir Dar Institute of Technology, Bahir Dar University, P. O. Box 26, Bahir Dar, Ethiopia

[2]ICT4D Research Center, Bahir Dar Institute of Technology, Bahir Dar University, P. O. Box 26, Bahir Dar, Ethiopia

ABSTRACT

Phishing attacks have become a prominent threat to online security with attackers constantly evolving their techniques to steal sensitive information. Traditional phishing detection methods often rely on analyzing individual features or patterns which may not capture the dynamic and evolving nature of phishing attacks. In this paper a novel approach for phishing website detection by considering the structural similarities among phishing websites. A clustering-based algorithm that can effectively group together websites exhibiting similar attack patterns is developed. For clustering we used hierarchical agglomerative and K-mean clustering algorithms whereas Longest Common Sequence (LCS) and fingerprint algorithms were employed to calculate similarities of the websites. We collected 3,588 website URLs from publicly available phishing websites repositories - Phishtank and legitimate website - Alexa. From the collected websites, HTML parsing was performed to extract relevant features that help to compute similarities. Silhouette score for internal validation and Adjusted Rand Index (ARI) for external validation of clusters were used. Hierarchical based clustering model (with Ward Linkage methods) with fingerprint similarity measure outperformed other models with Silhouette score and ARI values of 0.85, and 0.87, respectively. The proposed approach offers several advantages. First, it provides a holistic view of phishing attacks by considering the overall structural similarities instead of isolated features. Second, the clustering-based approach enables the detection of previously unknown phishing websites based on their similarity to known attack patterns. Third, it allows for the identification of emerging attack patterns traditional detection methods might not capture.

Keywords: *Cyber Attack, Phishing Detection, Document Object Model, Clustering, Similarity Measure, Cluster Validation*

DOI: https://doi.org/10.20372/pjet.v2i1.1716

Corresponding Author: Esubalew Alemneh Jalew

ICT4D Research Center, Bahir Dar Institute of Technology, Bahir Dar University, P. O. Box 26, Bahir Dar, Ethiopia

Email *esubalew@gmail.com*

1. Introduction

The rise of Internet has caused shift in peoples' lifestyle especially in the way we communicate with our customers, friends and families remotely. This gave a chance to thieves to initiate a new sort of crime that involves computers and networks. The attack which is named cybercrime is carried out by online criminals [1] in order to sneak data, deliberately disable machines, or utilize a compromised computer as a launching pad for more attacks. There are many types of cyber-attacks:malware, phishing, ransomware, denial of service, spoofing, identity-based attacks, codeinjection attacks, supply chain attacks, insider threats, *Domain Name System* (DNS) tunneling and Internet of Things (IoT)-based attacks are the most common types of cyberattacks [2]

Those who conduct cyber assaults are known as cybercriminals or threat actors [3]. They can act alone, alongside other attackers, or as part of an organized criminal groups. They attempt to uncover flaws in computer systems and exploit them to accomplish their objectives. For this study, among many varieties of cyber-attacks, we closely examined phishing attacks, which are one of the most widely spread and known type of cybercrimes [4].

Phishing attack is an attack in which customers' sensitive information is stolen via a website that mimic another legitimate website of an organization. The malicious website is designedin such a way that it has similar look and feel [4]. It is very genuine in appearance as the malicious page contains the organization's logos and other copyrighted contents. The most common types of phishing attacks are email phishing, spear phishing, whaling and pharming. addPhishing attack is a combination of technical and social dynamics which makes it even harder for users to detect it [2].

Phishing is frequently increasing type of cyber threat that practice of sending deceiving emails or other communications that appear to be from a reliable source. The intention is to steal personal information like (1) Credentials: such as password, username, credit card information, and pin number, (2) Personal Data: Name, address, and Email address, and (3) medical Data: such as treatment information and insurance claim or to infect the victim's computer with malware [5, 6]. Phishing attacks are generally easy to execute, as most of the victims are not well aware of web applications and computer networks [7]. As a result, identifying a phishing website is difficult.

Phishing attack detection approaches are classified into two as user education and software- based detection. User education-based phishing detection approach relies on increasing the user awareness onphishing websites by giving different types of trainings for end users [8]. The training is vital in creating awareness on phishing itself and on rules and regulations regarding phishing. This approach does not provide a decent solution because of factors such as users lack the motivation to educate themselves onsecurity, most users choose other tasks above security, and it is challenging to instruct individuals on how to choose the legitimate websites, and fake websites tend to migrate frequently throughout various servers.

The second phishing detection approach is a software-based approach. The approach is a technical anti-phishing solution that uses tools for the detection and prevention of phishing attacks. This software-based approach can be further divided into different categories [9]. List-based (white and black listing), deep learning based, machine learning based and heuristic-based detections as well as hybrid methods are under such category. These types of solutions are converted to some add-ons or plugins that are integrated into browsers. Many browsers now a day include security toolbars that comprises of those plugins. Examples of such plugins and toolbars are SpoofGuard of Internet Explorer plugin [10]. PhishTank SiteChecker of Mozilla Firefox add-on [11], eBay toolbar [12], and EarthLink Toolbar [13].

Although there are many techniques applied to detect and prevent phishing attacks, phishing attacks have not been stopped or even slowed down. According to numerous studies conducted by major tech organizations and anti-phishing groups phishing websites and email phishing are increasing even more. The statistics in [4] shows, monthly number of phishing attacks reported in May 2020 was 40,000 while number of attacks reported in May 2022 was almost 100,000, which is more than double of the May 2020 report. In addition, the same source showed that 3.4 billion spam emails are generated daily and Spam made up more than 48% of emails sent in 2022. This makes phishing the most predominant type of cybercrime. Figure 1 shows number of phishing attacks as reported by Anti-Phishing Working Group (APWG).

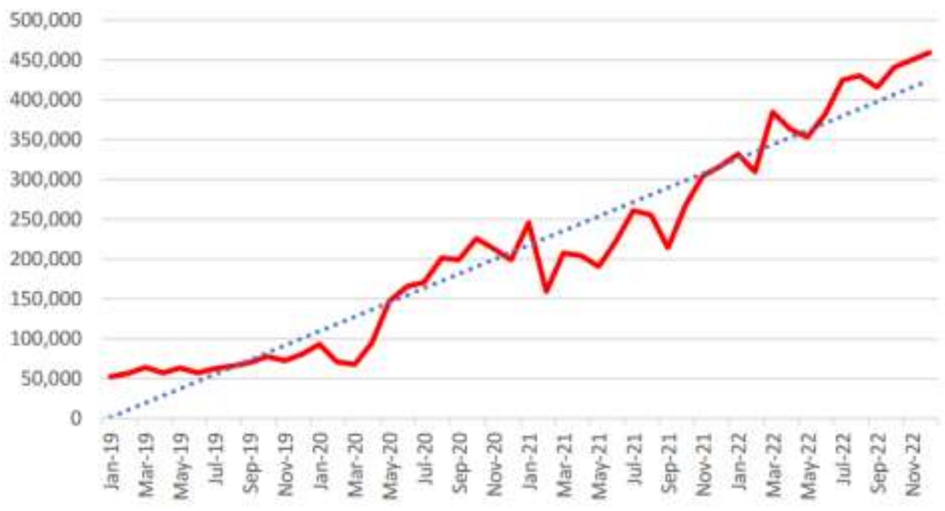


Figure 1: Monthly number of phishing attacks reported by APWG

Phishing attacks pose a significant threat to individuals, organizations, and inflicts a problem on the overall securityof online systems. These attacks continually evolve, employing various techniques to deceive users and extract sensitive information. Traditional detection methods often struggle to keep pace with the dynamic nature of phishing attacks, as they primarily rely on individual features or known patterns [14, 15]. Therefore, there is a pressing need for an innovative approach that can effectively detect phishing websites by considering the structural similarity of attacks.

There is the lack of a robust and accurate phishing website detection system that utilizes clustering techniques to group together websites exhibiting structural similarity attacks, enabling proactive identification and prevention of phishing occurrences. The objective of this research is to develop and evaluate a clustering-based approach for effective and accurate detection of phishing websites by leveraging the Hyper Text Markup Language (HTML) structural similarity of phishing attacks. Previous studies have demonstrated that the majority of phishing attacks (up to 90%) are duplicates of known attacks and are not created from scratch [16]. Hence, we propose a model that can detect phishing attack by evaluating its similarity with known attacks using different similarity measurement metrics. Since we know that most new attacks are variations of known ones, it shows that detecting such replicas is a strategic move.

Phishing detection based on structural similarity is proposed by many researchers. However, the review conducted on existing related works showed that the proposed solutions do not consider evolving nature of phishing attacks, and they have high false rates. Phishing attacks continually evolve to bypass detection mechanisms. Attackers employ new tactics, exploit emerging technologies, and employ social engineering techniques to deceive users. Existing research struggle to keep pace with these evolving phishing techniques, leading to increased false negatives and reduced effectiveness in detecting novel attacks. Moreover, the false positive rates of the existing classifier are high. That is even though the websites are legitimate the models classified them as phishing. A phishing attack is a rather new type of cybercrime, in comparison with other forms like virus and hacking. This research will contribute to the body of knowledge theoretically, practically and academically for the on-going research into phishing attack approach and detection. The subsequent sections of the paper are arranged as follows. On section two, background studies on anti-phishing strategies and their shortcomings are reviewed. Next section isdedicated to present the proposed solution – structural similarity-based phishing detectionmethod. The result of evaluation of the proposed method is presented at section four. Finally, conclusions are drawn at section five.

2. Related Works

An overview of phishing websites and detection techniques is provided in [14]. In addition, the study compares phishing detection techniques like list based, Heuristic oriented detection, visual similarity and machine learning phishing detection techniques. They stated that machine learning approach is the best out of the given detection mechanisms. Over a ten-month period, a total of 19,066 phishing assaults were discovered and over 90% of those attacks were either copies or variants of previous attacks inthe database and had a similar DOM structure [16]. The study provided new insights into the fight against phishing. They asserted and demonstrated that the generally accepted idea that phishing attempts are dealt quickly is an illusion. Over their monitoring period, the researchers observed multiple attacks. They were able to demonstrate that present preventative methods are becoming impractical and must be revised. They also make some ideas in that regard. According to the study, cluster and group-based strategies are the best solution for latest phishing attempts.

A homology detection method based on webpage clustering according to structural similarity is proposed in [17]. They collected and vectorized the structural characteristics and style attributes of webpages via HTML parser scraping tool, then assigned different weights to different features, assessed the similarity of webpages, and directed webpage clustering viathe webpage difference index. To test the new webpages against the model, they created patterns for the cluster centers and the test data, respectively and computed the similarity via bit wise comparison. Their study revealed that, when compared to existing approaches, clustering webpages could accurately discover and detect the family of phishing webpages.

The potential of employing unsupervised clustering approaches to classify emails into frauds that may be addressed together were investigated at [15]. They compared three clustering techniques with varied feature sets using a mix of contextual and semantic information collected from emails. The researchers assessed the clustering findings on real- world email datasets using a variety of internal and external validation approaches. Their finding indicated that unsupervised clustering is a potential method for scam identification and categorization.

A Natural Language Processing (NLP) model that used the Doc2Vec paradigm to treat DOMs as natural languages so that the model automatically learns structural semantics to detect phishing web pages was proposed in [9]. The researchers claimed that the existing phishing web page detection approach based on visual similarity is primarily concerned with getting visual features while ignoring the overall representation of web pages and the semantic information that HTML tags may include. The retrieved web page's DOM structure was parsed byBeautifulSoup to produce the DOM tree, and then the Doc2Vec model is used to vectorizethe DOM tree and assess semantic similarity in web pages by the distance between distinct DOM vectors. The study employed hierarchical clustering approach and were able to get a satisfying result.

Applications of similarity measures and a clustering technique to group the web pages into clusters is described in [18]. The researchers used tree edit distance measure on DOM trees ofthe webpage to measure structural similarity and Jaccard similarity metric on Cascading Style Sheet (CSS) class names to measure stylistic similarity between webpages. They demonstrated a shared near neighbor method for clustering the data, and promising result was found.

Nagaraj, Bhattacharjee, Sridhar, and Sharvani [19] stated that there was a lack of available techniques for detecting phishing activity and avoiding deception. They stated that the classification of phishing and non-phishing website contents is an important issue in any security information protocol. However, fool-proof methods have not been implemented in practice. Therefore, the aim of this study is the presentation of an effective phishing detection model for phishing website detection. The model engrosses concepts like evolving phishing techniques and zero-hour phishing and it is designed to reduce false positive rate. The model employes structural similarity-based approach and phishing is detected by comparing the similarities of known phishing attacks.

3. Structural Similarity Based Phishing Detection Model

The main objective of this study is to build a phishing attack identification model usingunsupervised learning approach. This section discusses about designing and grouping of variances of known attacks based on structural similarity using Hierarchical agglomerative and K-Mean clustering’s approach. We start by introducing the architecture of the proposed model.Then materials for data preparation and extraction, method for measuring the similarity of phishing attacks, clustering algorithm are discussed.

Proposed Architecture

In the proposed approach, instead of relating a suspicious phishing website to its target based on some defined features or extracting character of phishing and legitimate websites, we extract structural features that group the phishing websites together. Figure 2 shows the proposed approach. The following subsections explain components of the model.

Dataset preparation

The dataset here are a collection of Uniform Resource Locators (URLs), see Figure 3. For that purpose, the phishing and legitimate URLs have been collected from two major publicly available sources (PhishTank and Alexa) in order to achievediverse, well-balanced representation of their characteristics. An important aspect to consider while collecting phishing URL is the use of up-to-date URLs to capture the latest trends that phishers use nowadays in their efforts. Phishing webpages have short lifecycle.

Data Pre-processing

The two data preprocessing activities executed as part of this research are:

- *Correcting Missing Values* - The deletion of rows or columns with null values is one method of dealing with missing values. The step we take to identify missing value is by identifying empty or NULL columns rows with missing values in the column. This kind ofdefect is removed in the data cleaning process (1) calculate the number of missing valuesin the column, (2) drop rows with any missing values, and (3) Replace missing values in a column with another value since we don't want to reduce the size of our dataset.
- *Correcting Duplicate Values* - Because we used a web crawler for collecting the data, the missing and duplicated values in our dataset were very rare. Anyways, we performed the following tasks: (1) Identify rows with duplicate values in the column, (2) drop duplicate values - keep the first duplicate row and (3) add another phishing orlegit URL depending on the dropped URL (phishing or not) to keep our dataset size the same.

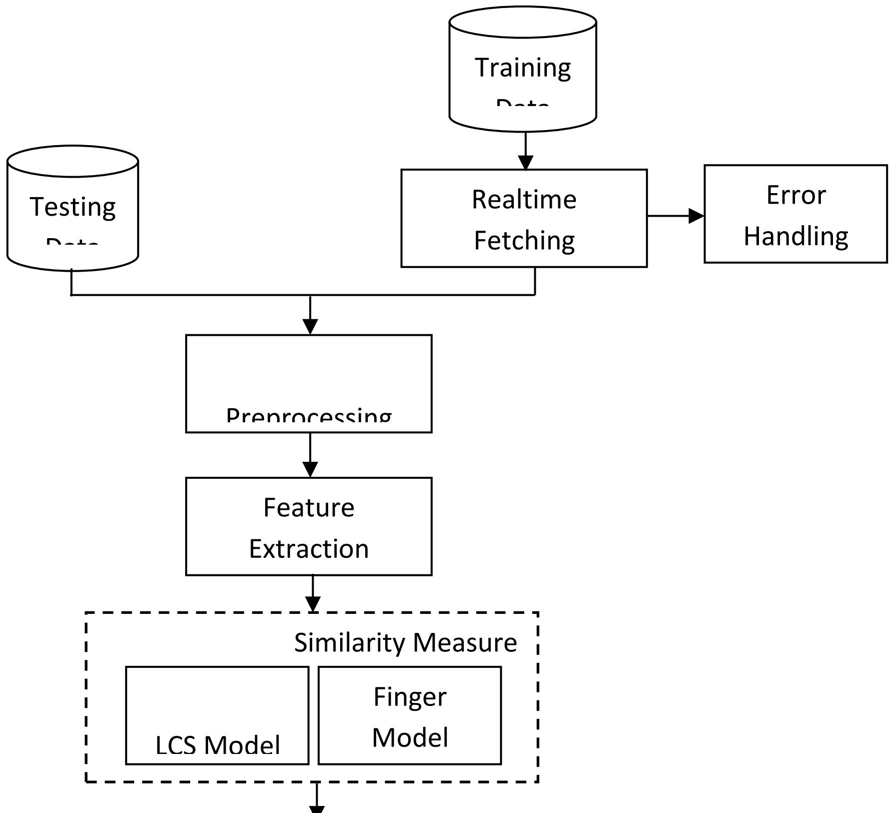
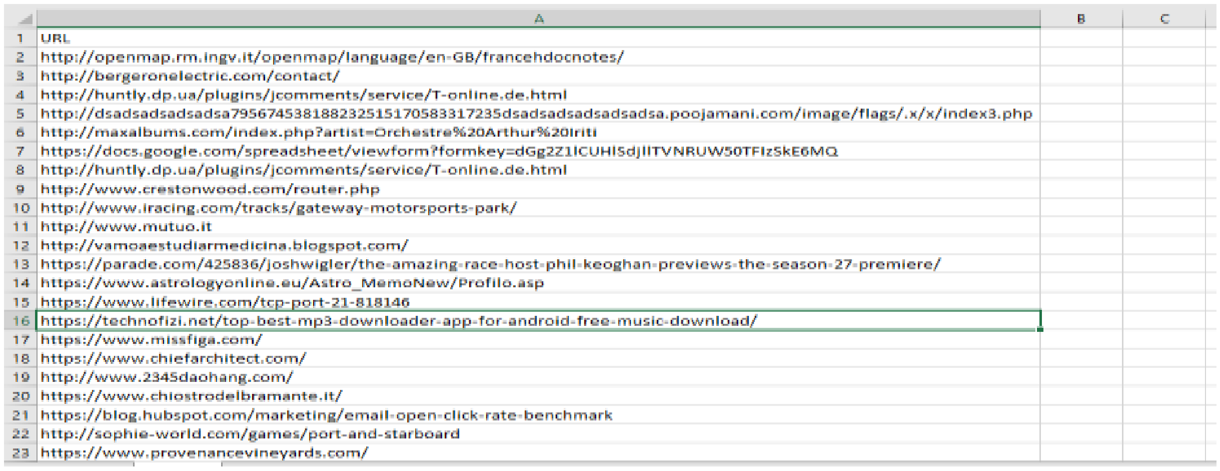


Figure 2: The Architecture of the proposed model



	A	B	C
1	URL		
2	http://openmap.rm.ingv.it/openmap/language/en-GB/francehdocnotes/		
3	http://bergeronelectric.com/contact/		
4	http://huntly.dp.ua/plugins/jcomments/service/T-online.de.html		
5	http://dsadsadsadsadsa795674538188232515170583317235dsadsadsadsadsadsa.poojamani.com/image/flags/.x/x/index3.php		
6	http://maxalbums.com/index.php?artist=Orchestre%20Arthur%20Iriti		
7	https://docs.google.com/spreadsheet/viewform?formkey=dGg2Z1lCUHlSdjlITVNRUW50TFIzSkE6MQ		
8	http://huntly.dp.ua/plugins/jcomments/service/T-online.de.html		
9	http://www.crestonwood.com/router.php		
10	http://www.iracing.com/tracks/gateway-motorsports-park/		
11	http://www.mutuo.it		
12	http://vamoaestudiarmedicina.blogspot.com/		
13	https://parade.com/425836/joshwigler/the-amazing-race-host-phil-keoghan-previews-the-season-27-premiere/		
14	https://www.astrologyonline.eu/Astro_MemoNew/Profilo.asp		
15	https://www.lifewire.com/tcp-port-21-818146		
16	https://technofizi.net/top-best-mp3-downloader-app-for-android-free-music-download/		
17	https://www.missfiga.com/		
18	https://www.chiefarchitect.com/		
19	http://www.2345daohang.com/		
20	https://www.chiostrodelbramante.it/		
21	https://blog.hubspot.com/marketing/email-open-click-rate-benchmark		
22	http://sophie-world.com/games/port-and-starboard		
23	https://www.provenancevineyards.com/		

Figure 3: Sample Dataset

In summary, the overall process of data-preprocessing involves sequential application of source code filter, source code parsing, HTML filter and HTML parsing. Both training and testing datasets pass through this pipeline.

Web scraping

Web scraping involves using algorithms to extract underlaying HTML code. An algorithm, which will scrape first the source code then the DOM structure of each URLs in real time wasdeveloped. DOM is a tree like structure of a collection of HTML tags that made up the webpage. Look at an example DOM structure of a webpage at Figure 4.

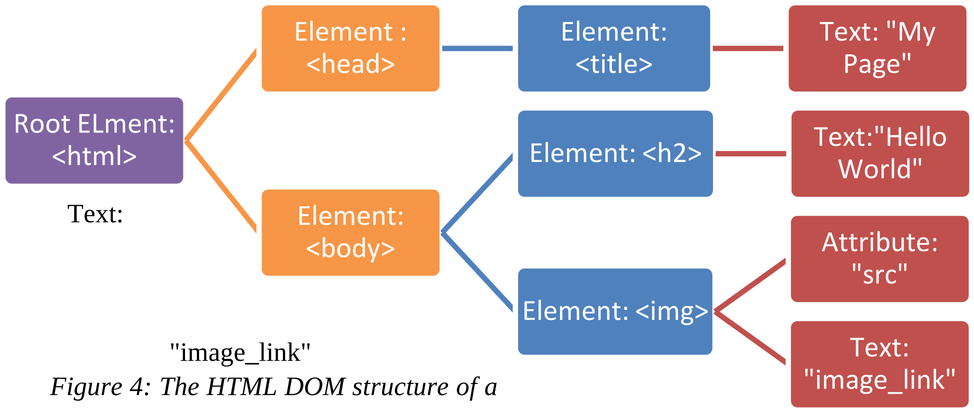


Figure 4: The HTML DOM structure of a webpage

There are a number of python libraries that assist in perfuming web scraping. We used *lxml*, and *BeautifulSoup* together for DOM manipulation and for parsing respectively. The libraries are chosen because of their effective performance and efficiency. The web scraping process of the model is shown on Figure 5.

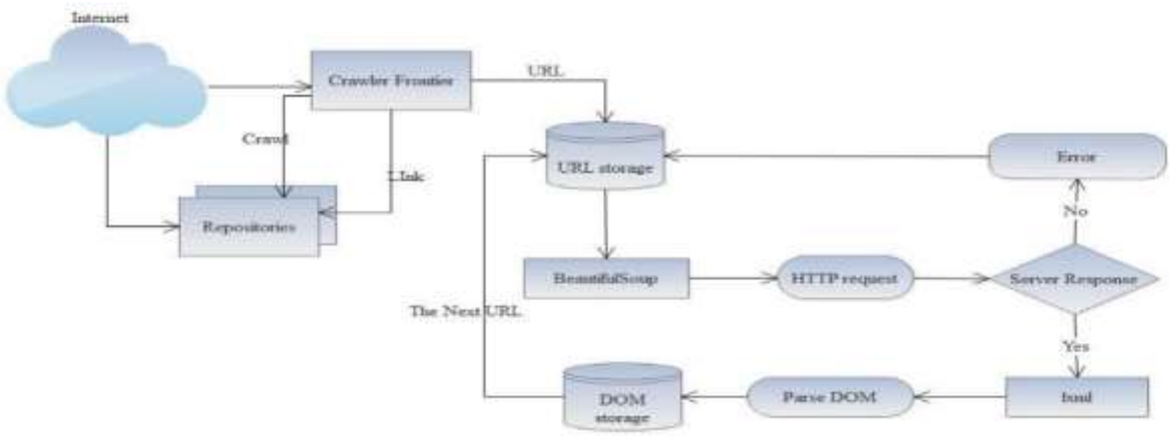


Figure 5: The web scraping process of the model

Dimensionality Reduction

Unsupervised learning employs a fundamental technique known as dimensionality reduction. It minimizes information loss by picking a smaller, unique set of variables that capture the most significant parts of the original features [20]. As a result, we decreased the number of random variables in a problem by identifying a group of important variables. According to HTML.com there are 132 HTML tags. Out of those attributes, 107 basic tags are selected by excluding common attributes that are applied to almost all HTML elements whether phishing or not, such as <html>, <head>, <title>, <body> [16].

Pairwise Similarity Metrics

Pairwise similarity refers the relationship between two DOMs of websites. Obviously, webpages are represented by a set of HTML tags and comparison of websites implies comparisons of tags the two websites are composed of. Hence, the tags of the two pages under comparison can be divided intothree categories: (1)

tags that are found in both DOMs, (2) tags that are found in just one of the twoDOMs, and (3) tags that are not included in either of the two DOMs. According to this division a page similarity measurement metrics have to fulfil five properties [21].

1. The presence of a tag in either page is more important than the absence of a tag.
2. The similarity measurement value should be symmetrical, meaning; the order inwhich the webpages are feed to the process should not affect the similarity result.
3. The similarity value should increase when the common sequence tags found in bothpages increases.
4. The similarity value should decrease when the common sequence tags found in both pages decrease.
5. Two pages are least similar to each other, if there is no common tag between them.

To group similar data points into the same clusters, similarity measures are very important components used by distance-based clustering algorithms [22]. Two types of similarity measures are used to calculate the similarity of the webpages: Longest Common Sequence (LCS) and fingerprint algorithms.

LCS Based Similarity

LCS checks each of the subsequence elements in the DOM to see if they are the same in value. As a webpage is represented as a tree data structure (i.e., DOM), it is straightforward to measuresimilarity of pages by comparing the DOMs. One common DOM metric for pairwise similarity measure is Tag Sequence algorithm, which represents and calculates the longest contiguous elements between DOMs to find the closeness of pages to each other.

In the pairwise similarity metric, the greater the similarity of two objects, the greater the value of the measure. The proposed algorithm has a time complexity of $(m * n)$, where n is the size of the first DOM and m is the size of the second DOM. Sequence matcher will first calculate the similarity between two pages and then output pairwise similarity values for each of the DOM structures in the directory corresponding to each webpage. The similarity values will be a number between 0.0 (no similarity) and 1.0 (identical page structures.). Algorithm 1 shows the LCS Algorithm.

Algorithm 1: An Algorithm to calculate LCS based Similarity

Input: DOM1, DOM2

Output: Similarity measure

```
1.  LCS (DOM1, DOM2):
2.      m = length (DOM1)n
3.      = length (DOM2)
4.      dp = create_matrix (m+1, n+1)for
5.      i = 1 to m:
6.          for j = 1 to n:
7.              if DOM1[i-1] == DOM2[j-1]:
8.                  dp[i][j] = dp[i-1][j-1] + 1
9.              else:
10.                 dp[i][j] = max(dp[i-1][j], dp[i][j-1])
11.     return dp[m][n]
12. function LCS (DOM1, Dom2):
13.     lcs_length = LCS(DOM1, DOM2)
14.     ratio = lcs_length / max(length(DOM1), length(DOM2))
15.   return ratio
```

Fingerprint based Similarity

The fingerprint of a webpage refers to a unique identifier or representation of the webpage based on its characteristics, features, or attributes. A webpage fingerprint can be generated by utilizing the HTML structure and extracting relevant features. The fingerprints are used to identify near- duplicate content across different websites or pages. The most important aspect of fingerprint extraction is assigning tag vector and tag weight. Tag vectorization means assigning a corresponding vector (number) to a given tag in the DOM. The tag elements are equivalent to the numbers provided by World Wide Web (WWW) Consortium. For the tag vectorization process, *eigenvector* was used. To construct the eigenvector (a mathematical representation of the webpage's structure), we require an algorithm that allocates an id to each branch of the DOM tree. We then determine the weight of each tag. Each tag's weight is affected by its position in the DOM tree. Assigning HTML tags, a weight based on their position is a crucial method for preventing the model from being overwhelmed by false tags. Algorithm 2 constructs the eigenvector of a webpage's DOM tree by assigning a weight to each node based on its position and attributes, and then summing up these weights in the eigenvector.

Algorithm 2: Construct Eigenvector (fingerprint)

Input: *DOM of a webpage*

Output: *eigenvector of a DOM*

```
SET dom_tree TO dom_tree  // the extracted html structure of a webpage
SET dimension TO dimension // The number of defined corpora of HTML tagsSET
node_info_list TO [ ] // tags in every tree structure
SET initial_weight TO 1  // Tags weight based on their position in the treeSET
depletion_value TO 0.6[1]
SET dom_eigenvector TO {}.fromkeys(range(0, dimension), 0)
DEFINE FUNCTION get_eigenvector():
  for node_id in range(1, size of dom_tree + 1):node
        = dom_tree.get_node(node_id)
        node_feature = create_feature(node)
        feature_hash =
        feature_hash(node_feature)
        node_weight = calculate_weight(node, node_id, feature_hash)
        construct_eigenvector (feature_hash, node_weight)
    return dom_eigenvector
   DEFINE FUNCTION construct_eigenvector(feature_hash, node_weight):
        feature_hash  =  feature_hash % self.dimension
dom_eigenvector[feature_hash] += node_weight
```

Here is an overview of how the eigenvector is constructed:

1. The algorithm iterates over all the nodes in the webpage's DOM tree. The range of theloop is from 1 to the size of the tree, which assumes that the root node has an ID of 1.
2. For each node, the method calls another method called create_feature [23]., which generates a feature vector based on the node's attributes and position in the DOM tree.This feature vector represents the node's structural properties.
3. The method then generates a hash code for the feature vector using another algorithm called feature_hash [23]. This hash code is used to map the feature vector to an index in the eigenvector.
4. The method calculates a weight for the current node using another method called calculate_weight [23]. This weight represents the node's importance in the webpage's overall structure.
5. The *construct_eigenvector* function updates the eigenvector by adding the node'sweight to the appropriate index in the vector. attribute_hash is the index at which the value is being updated and it is incremented by the node_weight value. The modulo operator (%) ensure that the attribute_hash value does not exceed the dimension of the eigenvector (self.dimension).
6. Once all the nodes have been processed, the algorithm returns the complete eigenvector.

Algorithm 3 defines a function called calculated_similarity that calculates the structural similarity between two websites based on their eigenvectors, which are mathematical representations of the websites' structures.

[1] depletion_value is a scaling factor to reduce the importance of elements further down the document hierarchy. We assigned value a higher than 0.5 as a weight to top-level elements as they have higher importance on the DOM tree thatwill gradually decrease the weight as it move down the hierarchy.

Algorithm 3: Pages similarity calculation

Input: *dom1_eigenvector, dom2_eigenvector, dimension*

Output: *Similarity Measure*

```
DEFINE FUNCTION calculated_similarity( dom1_eigenvector, dom2_eigenvector, dimension):a = 0 ;
    b = 0
    for i in range(dimension):
        a += dom1_eigenvector[i] - dom2_eigenvector[i]if
        dom1_eigenvector[i] and dom2_eigenvector[i]:
            b += dom1_eigenvector[i] + dom2_eigenvector[i]
    similarity = abs(a) / b
    return similarity
```

Here is a breakdown of what the algorithm does:

1. The function initializes two variables which are used to calculate the similarity score,a and b, to zero.
2. The function then uses a loop to iterate over dimension number of elements in the dom1_eigenvector and dom2_eigenvector arrays.
   For each element, the function subtracts the corresponding value in dom2_eigenvectorfrom the value in dom1_eigenvector and adds the result to a. This step calculates the difference between the two eigenvectors.
3. The function also checks if both dom1_eigenvector[i] and dom2_eigenvector[i] are non-zero. If they are, the sum of the two values is added to b.
   This step ensures that only non-zero values are used in the similarity calculation.
4. Then, it calculates the similarity score by dividing the absolute value of a by b.
5. Finally, the function returns the similarity score.

Clustering Model

Hierarchical Agglomerative Clustering Algorithms

The Hierarchical Agglomerative Clustering (HAC) is the most common type of hierarchical clustering used to group objects in clusters based on their similarity. The hierarchical algorithm is set to work ina bottom-up style. The algorithm initially, treats each item as a single-element cluster (leaf). At each phase of the method, the two most comparable clusters are joined into a new larger cluster (nodes). This method is repeated until all points belong to a single large cluster (root). The result is a tree-based representation of the objects called dendrogram. Figure 6 shows how hierarchical agglomerative works using dendrogram.

The HAC algorithm will excel to group DOMs that are very similar to at least one otherDOM in the cluster. No matter what sequence the webpages are introduced to the model, this has the benefit of consistently providing the same result for the same input [20]. This is a huge benefit in our situation since the database will change when new phishing websites are found, causing it to grow over time.

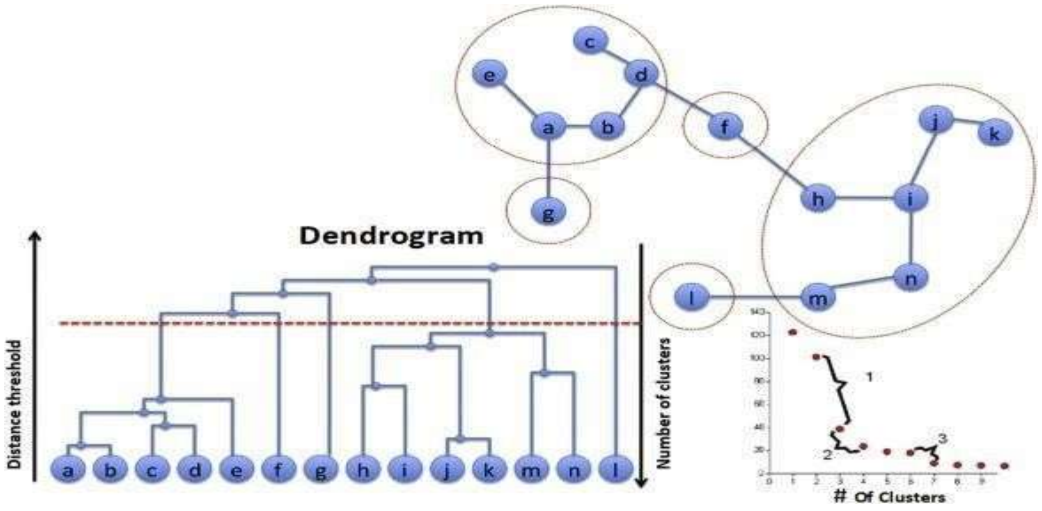


Figure 6: Visualization of HAC

The HAC algorithm first constructs the finest partition by treating all webpages (their DOM structure) as a cluster. Then it computes similarity matrixes. Then it finds the two webpages with the closest distance by performing distance metric. It put those two webpages into one cluster if the distance condition (threshold) is satisfied. Then it computes the distance between the new groups and obtain a reduced distance matrix. This process is repeated until all possible webpages are agglomerated as one cluster. Algorithm 4 displays a pseudocode that describes how HAC algorithm works.

Algorithm 4: Agglomerative Definition of a Hierarchical Clustering Structure

Input: a set of vectors data being clustered, a linkage method {ward, average, complete}[2], threshold value that will stop the clustering process once reached

Output: a set of clusters of the given vectors data

```
1. function hierarchical_agglomerative_clustering (data, linkage_method, threshold):clusters =
2.     initialize_clusters(data)
3.     similarity_matrix = compute_similarity_matrix(data) while
4.     threshold_not_met(similarity_matrix, threshold):
5.         (cluster_i, cluster_j) = find_most_similar_clusters(similarity_matrix, linkage_method)
6.         merged_cluster = merge_clusters(cluster_i, cluster_j)
7.         clusters = update_clusters(clusters, merged_cluster)
8.         similarity_matrix = update_similarity_matrix(similarity_matrix, merged_cluster)return
9. clusters
```

The above algorithm works as follows:

1. *Initialize clusters:* Initialize each data point as a separate cluster.
2. *Compute initial similarity matrix:* Calculate the similarity matrix based on the similarity metric used (LCS, fingerprint-based similarity) for pairwise comparisons between data points.
3. *Main loop:* Iterate until the stopping criteria are met.
4. *Find most similar pair of clusters based on linkage method:* Identify the pair of clusters with the highest similarity score according to the specified linkage method (ward linkage, complete linkage, average linkage).
5. *Merge the two clusters:* Combine the data points from the two clusters into a single merged cluster.

[2] Linkage methods determines how the similarities between clusters are calculated

6. *Update clusters:* Update the set of clusters by removing the two merged clusters and adding the new merged cluster.
7. *Update similarity matrix:* Recompute the similarity matrix to reflect the updated clusters. This typically involves updating the similarity scores between the merged cluster and the remaining clusters.
8. *Check stopping criteria:* Evaluate if the stopping criteria are met. This could be based on the desired number of clusters, a threshold for similarity score, or any otherdefined stopping condition.
9. *Return clusters:* Once the stopping criteria are met, return the final set of clusters.

K-means Algorithm

Besides HAC, we have used the famous clustering algorithm K-means. K-means is a centroid-based clustering algorithm in which each data point is assigned to a cluster based on its distance from a centroid. Clustering can be used to detect and group together phishing pages that share similar layouts, HTML structures, or other structural properties. Grouping structurally similar phishing pages together is considered a clustering problem because it involves unsupervised learning, identification of structural similarities, dimensionality reduction, and anomaly detection [17]. The main purpose of clustering structurally similar webpages is to organize and group webpages that exhibit similar structural characteristics. This can provide a benefit of identifying the criminal groups behind the attack as the same phishing attack organizations seek to develop structurally similar websites, it becomes easy to navigate and investigate similar assaults and even the group behind them. Clustering techniques provide a valuable approach to detecting and categorizing phishing pages basedon their structural properties, serving in the identification of phishing threats.

4. Results

Before presenting the result of the research, in the Evaluation Setup subsections, topics like the data collection and preparation as well as experimental and parameter setups are presented.

Evaluation Setup

Dataset collection and preparation*:* The phishing URLs come from a website called PhishTank [phishtank.org]. PhishTank is acommunity that allows people to submit, verify, track, and share phishing website. It has been utilized for most phishing attack-based researches as source to fetch phishing websites. As a result, it is a trustworthy source of accurate information. 3,600 links in general are gathered.We also compiled a database of non-phishing websites in order to compare the results of our clustering technique on phishing sites and legal sites. We utilized Alexa, which analyses online traffic and produces lists of the most popular websites, for this purpose. We assumed that all of the websites listed on Alexa.com were real. We chose 500 authentic site addresses at random from Alexa's top 100,000 online sites. In order to collect the above stated URLs of both legitimate and phishing websites we used web crawling techniques.

Experimental setup: To conduct experiments, we used a machine with 45 GB of RAM and a GPU A4000. To build the model, we used the Python programming language along with the BeautifulSoup, lxml, HTMLParser, difflib, request, NumPy, and fcluster libraries. We conducted the experiments using LCS and fingerprint as similarity metrics on the Hierarchical (HAC) and centroid model (K-Mean). To do these experiments, we have prepared two different repositories from the given dataset: URL-DOM pair data for the LCS method and URL-page fingerprint pair data for the fingerprint method. Finally, we compare the models by observing the Dendrogram.

Based on the collected URLS in the datasets, web scraping is performed for DOM structure. However, some phishing webpages from the dataset could not be reached in real- time for variety of reasons (websites taken down, returned a 403-level HTTP error code, were empty or failed to load in our scraper). The above stated reason changes the size of our dataset from 4,100to 3,588. Overall (500 legitimate URLs and 3,088 phishing URLs) all the unresponsive pages were phishing URLs.

In this experimental study, we trained the models on a raw DOM data to identify patterns and tocluster similar data into unspecified number of groups. We conducted the first experiment by using the Dom of 3,588 webpages. We were able to achieve a good result as those phishing attacks have structural similarity with each other by way of their similarity value showed.

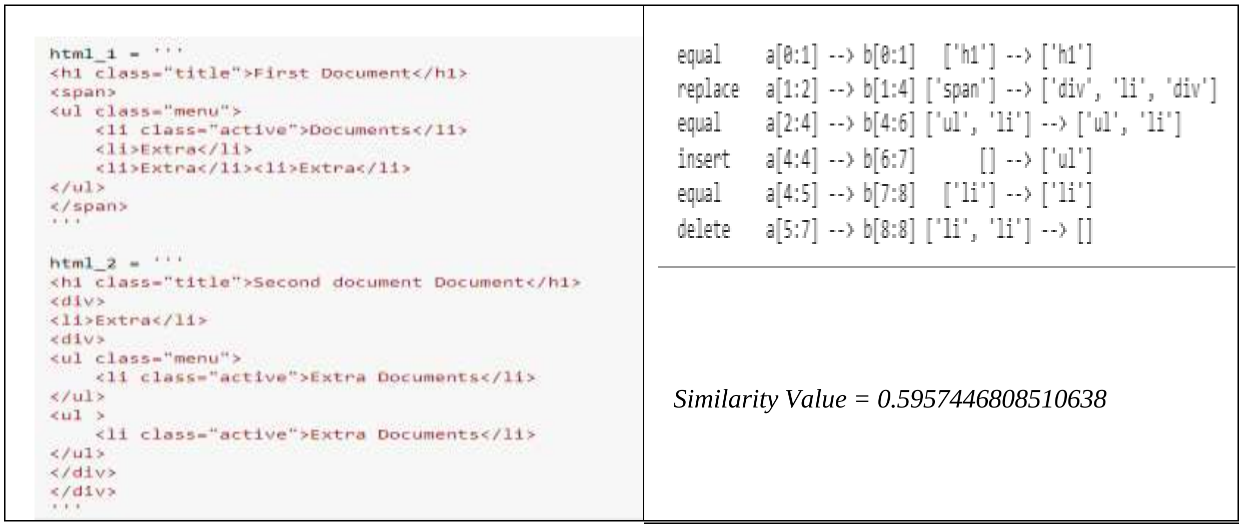
Parameter Selection: The parameters for both fingerprint and LCS algorithms are shown on Table 1. as similarity metrics.

Table 1: Evaluation Models

K-Mean Model		**HAC Model**	
Parameters	**Value**	**Parameters**	**Values**
Random-state	42	Threshold	0.2 - 0.6 inclusive
Init	10	Distance Metrics	Euclidean
Tol	*1e-4*	Linkage Method	Ward, Average, Complete
Max-iter	300		

Experimental Results

The sequence matcher has four functions (insert, equal, delete, and replace) while comparing two values. Figure 7 bellow shows how those functions operate on two given DOMs. Because it is a pairwise similarity calculation it will produce a (n*n) matrix value in this case (1000*1000) that will be used as an input for the clustering algorithm.



```
html_1 = '''
<h1 class="title">First Document</h1>
<span>
<ul class="menu">
    <li class="active">Documents</li>
    <li>Extra</li>
    <li>Extra</li><li>Extra</li>
</ul>
</span>
'''

html_2 = '''
<h1 class="title">Second document Document</h1>
<div>
<li>Extra</li>
<div>
<ul class="menu">
    <li class="active">Extra Documents</li>
</ul>
<ul >
    <li class="active">Extra Documents</li>
</ul>
</div>
</div>
'''
```

```
equal    a[0:1] --> b[0:1]   ['h1'] --> ['h1']
replace  a[1:2] --> b[1:4] ['span'] --> ['div', 'li', 'div']
equal    a[2:4] --> b[4:6] ['ul', 'li'] --> ['ul', 'li']
insert   a[4:4] --> b[6:7]       [] --> ['ul']
equal    a[4:5] --> b[7:8]   ['li'] --> ['li']
delete   a[5:7] --> b[8:8] ['li', 'li'] --> []
```

Similarity Value = 0.5957446808510638

Figure 7: Example of similarity calculation of two webpages

Figure 8 below shows the pairwise similarity value between 1,000 phishing webpages. The x-axis represents the webpages while the y-axis represents the similarity value. The similarity valuesare numbers between 0.0 and 1.0, inclusive, where 0.0 indicates no similarity and 1.0 indicates identical page structures. Here the figure just shows that the dataset contains websites of various similarity level.

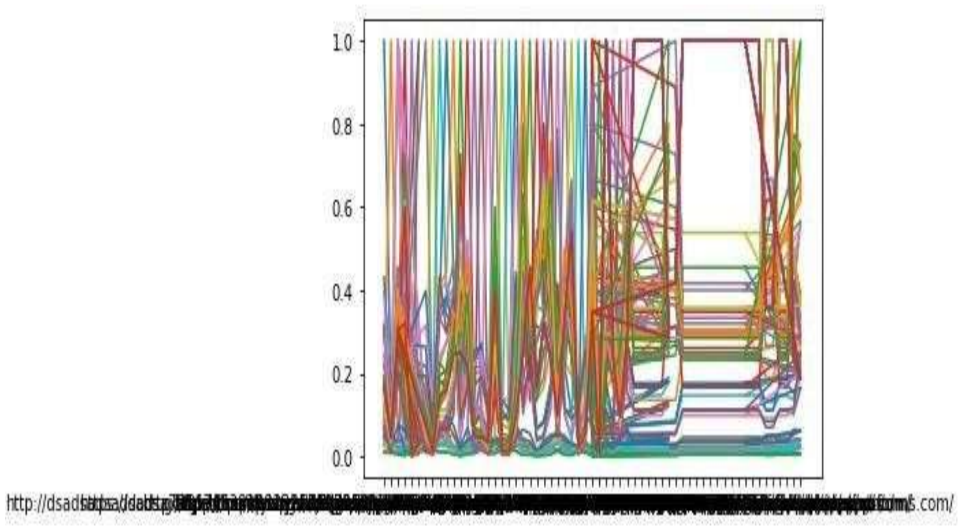


Figure 8: Pairwise similarity value of the attacks

The dendrogram bellow, Figure 9, shows a tree-based representation of the attacks by the agglomerative algorithm. The model clustered the 3,588 attacks into 174 clusters having a size variety of 1 to 94. Most of those URL's were gathered in February 5 - 6, 2022. Not only this cluster but also most of the phishing attacks that were grouped together are mostly collected in consecutive days. One of the reasons that we conducted the experiment is in the hopes that the algorithm finds hidden pattern in the attacks and group them accordingly. Nevertheless, to observe the group behavior, we manipulated the distance threshold value, which then change the number of clusters formed by the model. The final number of clusters described above was returned by applying a threshold value of 0.3.

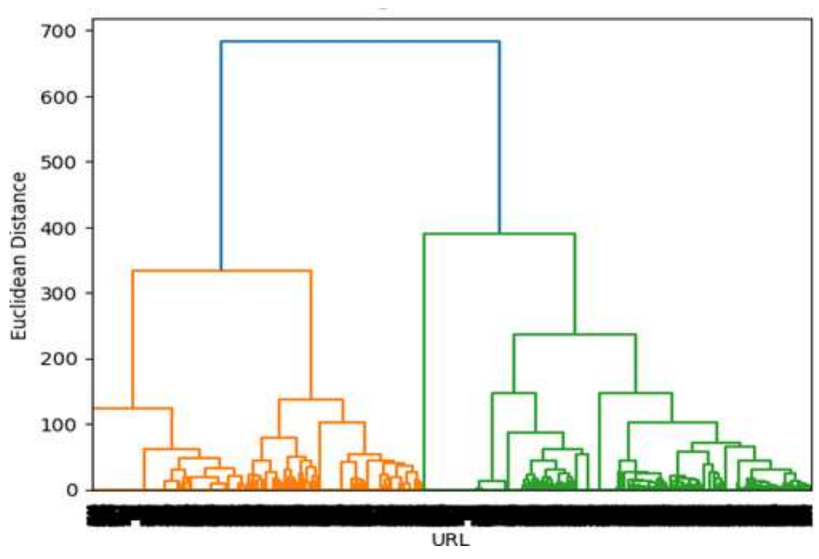


Figure 9: Dendrogram of HAC-Ward model for fingerprint method

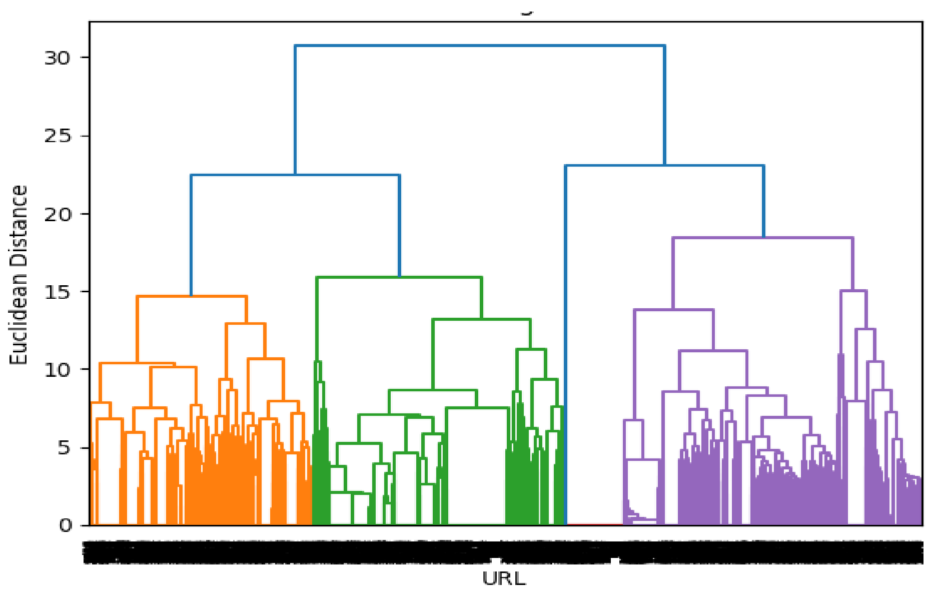


Figure 10: Dendrogram of HAC-Ward model for LCS method

Figure 10 shows the dendrogram using LCS method. The model created 293 clusters of attacks having a size variety of 1 to 76 inclusive. Objects in the same cluster are supposed to have similar value and closer distance as oppose to objects in different cluster. That means having cohesive bottom and detached structure while moving to the top in the dendrogram [20]. Out of the three-methods ward linkage showed a better adhesive result between pages thancomplete linkage.

Clustering Validation

Among the four clustering validation methods [1, 3], two methods were employed to validate the clustering results: internal validation and external validation. To evaluate the goodness clustering, internal validation uses the internal information of the clustering process. External validation involves comparing the results of a cluster analysis to an externally known result.

Internal Validation: Internal validation measures the inter variance and intra variance of the training data [2]. Only the information used for clustering is utilized for internal validation. In general, valuesindicating homogeneity and/or separation are generated as part of the referred to internal validation process. The quality of clusters is measured using Silhouette Score [24]. The measure attempts to characterize how similar a data point is to other data points in its cluster relativeto data points that are not in its cluster. This is aggregated across all data points to producethe score for an overall clustering. The score value is between -1 and 1 inclusive. A value closer to -1 indicates improper clustering, while a value closer to +1 indicates that each cluster is extremely tightly packed. Four experiments were conducted and the experimental values are shown on Table 2. The table shows the number of clusters created together with the minimum and maximum size of clusters, and Silhouette scores among others.

Table 2: Silhouette Score of the Fingerprint and LCS-based model

Model	**Fingerprint-based model**		**LCS-based model**	
	#Cluster [min size, max size]	**Silhouette Score**	**#Cluster [min size, max size]**	**Silhouette Score**
HAC-complete	261 [1, 59]	0.77	385[1, 67]	0.74
HAC-ward	174 [1, 94]	0.85	293 [1, 76]	0.81
HAC-average	221[2, 104]	0.82	412 [2, 88]	0.80
K-means	190 [150, 300]	0.80	270 [250, 450]	0.77

In case of K-means based model calculation of Silhouette can't be obtained readily as the number of clusters is required as an input. However, the numbers of clusters are unknown because we do not have prior knowledge regarding the number of sites with similar structure or the number of attack replicas [25]. To tackle this issue, we used the number of clusters generated by the three preceding models (174, 221, 261) as a guide. Therefore, we have devised a function that iterates through various k values, where k represents the number of clusters, ranging from 150 to 300 with increments of 20. After we iterate through different cluster numbers, we calculated the Silhouette score for each number of clusters and picked the one with the highest value. Here we have seen that 190 clusters have the highest score for Silhouette with the value 0.80

External Validation: External validation indices are employed in this research to validate the proposed clustering approach and compare it to current strategies. In these situations, we prepare a dataset of a size 450 for which we know the ground truth and determine if our clustering approach can yield clustering solutions that are comparable to it. Table 3 shows the results of clustering validation of the clustering models as measured in terms of Adjusted Rand Index (ARI) [26].

Table 3: External Clustering Model Validation Evaluation Results

Model	**Adjusted Rand Index**	
	Fingerprint based similarity	**LCS based similarity**
HAC-complete	0.80	0.74
HAC-ward	0.87	0.79
HAC-average	0.85	0.72
K-Means	0.86	0.74

The ARI is a measure of how similar two clustering's are. It does this by looking at all pairs of samples and counting the ones that are expected to be in the same or different clusters comparedto the actual clustering's. ARI may take on values that range from -1 to 1. Being closer to one isdesirable, whereas being closer to negative one is undesirable. HAC-Ward has a high score in both fingerprints based and LCS based measures. We would anticipate this based on the table above. We used 350 webpages (300 phishing and 50 legitimate sites) to test the models in this case.

5. Discussions

The main purpose of this study is to conduct experiment on detecting phishing websites via structural similarity between attacks. Different experiments were conducted using LCS and fingerprint as similarity metrics. The best accuracy score was obtained using fingerprint-based similarity metrics on HAC-ward model. Generally, fingerprint-based models have better Silhouette scores and accuracy have relatively minimum time than LCS based model. Even when we compared with memory, fingerprint-based models are best: it takes less memory than LCS-based models.

The main limitation of the LCS based model is it is easily affected by dummy tags. Dummy tags are tags that are injected into the code with a solely purpose of changing the DOM structure of the webpage without changing the appearance of the site. This kind of tags are inserted by the phisher to fool the malicious detector. An attacker may be able bypass in the LCS based model by injecting numerous fake tags (such as <table> and <div>) to decrease the similarity between attacks. The fingerprint-based model has overcome this potential vulnerability. The model assigns weight to each identifier. Therefore, we could specify a lighter weight for the dummy tags. This tag weight varies by the position of the tag on the DOM tree and the number of child nodes under it. Therefore, the modification to the fake identifier has minimal effect on the final similarity metrics.

In the experiment, the model grouped the attacks based on their DOM structure. Moreover, the dendrogram is cohesive to fragmented moving bottom to top. This shows that the webpages have close distance value with each other that proves there is indeed structural similarity between attacks. In order to enhance the cluster model, we tested it with various threshold values and calculated its Silhouette score to determine the optimal threshold value. We were able to observe that the optimal threshold value is 0.3 with the highest Silhouette score for both metrics.

We believe that this study will enhance theoretical, practical, and academic understanding within the ongoing research on phishing attack methods and detection. Theoretically, the hypothesis *most phishing attacks are gemination of known attacks* is proven to be true. Implementation of a model that comprises of various algorithms to detect phishing attacks is a practical contribution. The model can be implemented as an add-on or plugin and added to the security toolbars of browsers. Obviously, the theoretical and practical contributions of the study will benefit researchers and academicians as they are continuously chasing after new findings and concepts.

6. Conclusion

This paper presented the result of a research conducted to design and implement a phishing website detector using structural similarities between attacks to solve the problem of relying on comparing phishing sites with the target site or extracting features of phishing websites. Two types of algorithms (LCS and fingerprint) were employed to measure similarity of webpage structures. Hierarchical Agglomerative Clustering (HAC) and K-mean were used as clustering algorithms. The models can detect phishing websites which are replicas of previously known attacks and are stored in the dataset. In addition, the method is able to identify deceitful sites that appear differently but have similar website structures. The proposed fingerprint-based HAC using Ward model is an optimal model in terms ofaccuracy, and Silhouette score. The model has a better accuracy (0.81) and Silhouette score (0.85) in both LCS and fingerprint-based approach. The validity and accuracy are further proved using internal and external validation. Since features are defined to show the similarity of phishing instances, phishing websites in each group are close to each other, and this group represents individual attacks that are duplicate of know attacks in the form of a cluster. Our approach is likely to be very effective in use for studies that are intended to investigate phishing attacks family or the groups behind it.

The findings will contribute to the development of more effective and adaptive phishing detection systems, ultimately improving online security and user protection against phishing threats. Moreover, this research will contribute to the body of knowledge theoretically, practically and academically for the on-going research into phishing attack approach and detection. Finally, we would like to recommend integrating the proposed model with different machine learning algorithms for detecting new variances of attacks (attacks that are developed from scratch) as our model fails to detect them. In the future we plan to develop a real-time browser extension that will provide warnings when visiting potentially malicious websites usingour model.

Reference

[1]. Kumar, J., Santhanavijayan, A., Janet, B., Rajendran, B., & Bindhumadhava, B. S. (2020). Phishing Website Classification and Detection Using Machine Learning. *International Conference on Computer Communication and Informatics*. https://doi.org/10.1109/iccci48352.2020.9104161

[2] Asadullah Safi , Satwinder Singh (2024), A systematic literature review on phishing website detection techniques

[3] Huaping, Y., Chen, X., Li, Y., Yang, Z., & Liu, W. (2018). Detecting Phishing Websites and Targets Based on URLs and Webpage Links. *International Conference on Pattern Recognition*. https://doi.org/10.1109/icpr.2018.8546262

[4] Rao, R. S., & Pais, A. R. (2019). Detection of phishing websites using an efficient feature- based machine learning framework. *Neural Computing and Applications*, *31*(8), 3851–3873. https://doi.org/10.1007/s00521-017-3305-0

[5] Nguyen, L. V., Le, D., & Vinh, L. A. (2014). Detecting phishing web pages based on DOM- tree structure and graph matching algorithm. *Symposium on Information and Communication Technology*.https://doi.org/10.1145/2676585.2676596

[6] Haynes, K. D., Shirazi, H., & Ray, I. (2021). Lightweight URL-based phishing detection using natural language processing transformers for mobile devices. *Procedia Computer Science*, *191*, 127–134. https://doi.org/10.1016/j.procs.2021.07.040

[7] Li, M., Misra, M., & Atrey, P. K. (2016). A survey and classification of web phishing detection schemes. *Security and Communication Networks*, *9*(18), 6266–6284. https://doi.org/10.1002/sec.1674

[8] Aljofey, A., Jiang, Q., Qu, Q., Huang, M., & Niyigena, J. (2020). An Effective Phishing Detection Model Based on Character Level Convolutional Neural Network from URL. *Electronics*, *9*(9), 1514. https://doi.org/10.3390/electronics9091514

[9] Feng, J., MA, Zhang, Y., & Qiao, Y. (2020). A Detection Method for Phishing Web Page Using DOM- Based Doc2Vec Model. *Journal of Computing and Information Technology*, *28*(1), 19–31. https://doi.org/10.20532/cit.2020.1004899

[10] Teraguchi, N.C.R.L.Y. and Mitchell, J.C., 2004. Client-side defense against web-based identity theft. Computer Science Department, Stanford University. Available: http://crypto. stanford. edu/SpoofGuard/webspoof. pdf.

[11] Arab, M., & Sohrabi, M. (2017). Proposing a new clustering method to detect phishing websites. *Turkish Journal of Electrical Engineering and Computer Sciences*, *25*, 4757–4767. https://doi.org/10.3906/elk- 1612-279

[12] Crescenzi, V., Merialdo, P., & Missier, P. (2005). Clustering Web pages based on their structure. *Data and Knowledge Engineering*, *54*(3), 279–299. https://doi.org/10.1016/j.datak.2004.11.004.

[13] EARTHLINK, I. (2016). Earthlink toolbar.

[14] Kalaharsha, P., & Mehtre, B. M. (2021). Detecting Phishing Sites - An Overview. *ArXiv (Cornell University)*. https://arxiv.org/pdf/2103.12739v2

[15] Saka, T., Vaniea, K., & Kökciyan, N. (2022). Context-Based Clustering to Mitigate Phishing Attacks. *Proceedings of the 15th ACM Workshop on Artificial Intelligence and Security*. https://doi.org/10.1145/3560830.3563728

[16] Cui, Q., Jourdan, G., Von Bochmann, G., Couturier, R., & Onut, I. (2017). Tracking Phishing Attacks OverTime. *The Web Conference*. https://doi.org/10.1145/3038912.3052654

[17] Feng, J., MA, Qiao, Y., Ye, O., & Zhang, Y. (2022). Detecting phishing webpages via homology analysisof webpage structure. *PeerJ*, *8*, e868. https://doi.org/10.7717/peerj-cs.868

[18] Alswailem, A. K., Alabdullah, B., Alrumayh, N., & AlSedrani, A. (2019). Detecting Phishing Websites Using Machine Learning. *2019 2nd International Conference on Computer Applications & Information Security (ICCAIS)*. https://doi.org/10.1109/cais.2019.8769571

[19] Nagaraj, K., Bhattacharjee, B., Sridhar, A., & Gs, S. (2018). Detection of phishing websites using a noveltwofold ensemble model. *Technology*, *20*(3), 321–357. https://doi.org/10.1108/jsit-09-2017-0074

[20] Gowda, T., & Mattmann, C. A. (2016). Clustering Web Pages Based on Structure and Style

Similarity (Application Paper). *Information Reuse and Integration*. https://doi.org/10.1109/iri.2016.30

[21] Oghbaie, M., & Zanjireh, M. M. (2018). Pairwise document similarity measure based on present term set. *Journal of Big Data*, *5*(1). https://doi.org/10.1186/s40537-018-0163- 2

[22] Torres, G. J., Basnet, R. B., Sung, A. H., Mukkamala, S., & Ribeiro, B. M. (2009). A similarity measure for clustering and its applications. Int J Electr Comput Syst Eng, 3(3), 164-170.

[23] Megdelawit Bewketu (2023), Phishing Website Detection using Structural Similarity amongst Attacks, MSc.Thesis, Faculty of Computing, Bahir Dar Institute of Technology, Bahir Dar University.

[24] Shahapure, K. R., & Nicholas, C. (2020, October). Cluster quality analysis using silhouette score. In 2020 IEEE 7th international conference on data science and advanced analytics (DSAA) (pp. 747-748). IEEE.

[25] Yadav, J., & Sharma, M. (2013). A Review of K-mean Algorithm. Int. J. Eng. Trends Technol, 4(7), 2972-2976.

[26] Santos, J. M., & Embrechts, M. (2009, September). On the use of the adjusted rand index as a metric for evaluating supervised classification. In International conference on artificial neural networks (pp. 175-184). Berlin, Heidelberg: Springer Berlin Heidelberg.