




Article

HinPhish: An Effective Phishing Detection Approach Based on Heterogeneous Information Networks

Bingyang Guo ^{1,2} , Yunyi Zhang ^{1,2}, Chengxi Xu ^{1,2} , Fan Shi ^{1,2}, Yuwei Li ^{1,2}  and Min Zhang ^{1,2,*}

¹ College of Electronic Engineering, National University of Defense Technology, Hefei 230037, China; guobingyang@nudt.edu.cn (B.G.); sky_yunyi@hotmail.com (Y.Z.); xuchengxi17@nudt.edu.cn (C.X.); shifan17@nudt.edu.cn (F.S.); liyuwei23@163.com (Y.L.)

² Anhui Province Key Laboratory of Cyberspace Security Situation Awareness and Evaluation, Hefei 230037, China

* Correspondence: zhangmindy@nudt.edu.cn

Abstract: Internet users have suffered from phishing attacks for a long time. Attackers deceive users through malicious constructed phishing websites to steal sensitive information, such as bank account numbers, website usernames, and passwords. In recent years, many phishing detection solutions have been proposed, which mainly leverage whitelists or blacklists, website content, or side channel-based techniques. However, with the continuous improvement of phishing technology, current methods have difficulty in achieving effective detection. Hence, in this paper, we propose an effective phishing website detection approach, which we call HinPhish. HinPhish extracts various link relationships from webpages and uses domains and resource objects to construct a heterogeneous information network. HinPhish applies a modified algorithm to leverage the characteristics of different link types in order to calculate the phish-score of the target domain on the webpage. Moreover, HinPhish not only improves the accuracy of detection, but also can increase the phishing cost for attackers. Extensive experimental results demonstrate that HinPhish can achieve an accuracy of 0.9856 and F1-score of 0.9858.

Keywords: malicious domain detection; phishing; heterogeneous information network



Citation: Guo, B.; Zhang, Y.; Xu, C.; Shi, F.; Li, Y.; Zhang, M. HinPhish: An Effective Phishing Detection Approach Based on Heterogeneous Information Networks. *Appl. Sci.* **2021**, *11*, 9733. <https://doi.org/10.3390/app11209733>

Academic Editor: Federico Divina

Received: 16 September 2021

Accepted: 8 October 2021

Published: 18 October 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Phishing is a very common and dangerous attack campaign. With the popularization of network technology, the cost of phishing attacks has been reduced significantly. Phishers can deploy a mimic website with low cost and high efficiency by utilizing various phish kits [1]. Then, these websites convince victims to leak their sensitive credentials, such as passwords, phone numbers, bank accounts, credit card numbers, and other private information. Phishing attacks have been become one of the major cybersecurity threats. They may not only cause the victim to suffer economic losses, but can also destroy the reputation of the imitated website. According to the report published by the Anti-Phishing Working Group (APWG) [2], after experiencing a rapid growth in 2020 (doubling over the course of the year), the number of phishing attacks in January 2021 reached its peak in the APWG's records, as shown in Figure 1. The trend of phishing attacks is becoming worse.

To evade detection, attackers leverage various cloaking techniques to arm phishing pages. Oest et al. [1] analyzed the nature of server-side .htaccess filtering techniques and discovered that attackers can launch a targeted attack by customized whitelist filters. For example, when the visiting IP is from the U.S., the phishing sites will return the phishing page and, otherwise, return a normal page. Invernizzi et al. [3] studied the blackhat cloaking techniques of ten prominent cloaking services and pointed out that they hinder the detection of protection mechanisms. Zhang et al. [4] provided the first broad perspective of client-side cloaking techniques, demonstrating how an attacker can display specific content to different users through the use of client-side cloaking techniques.

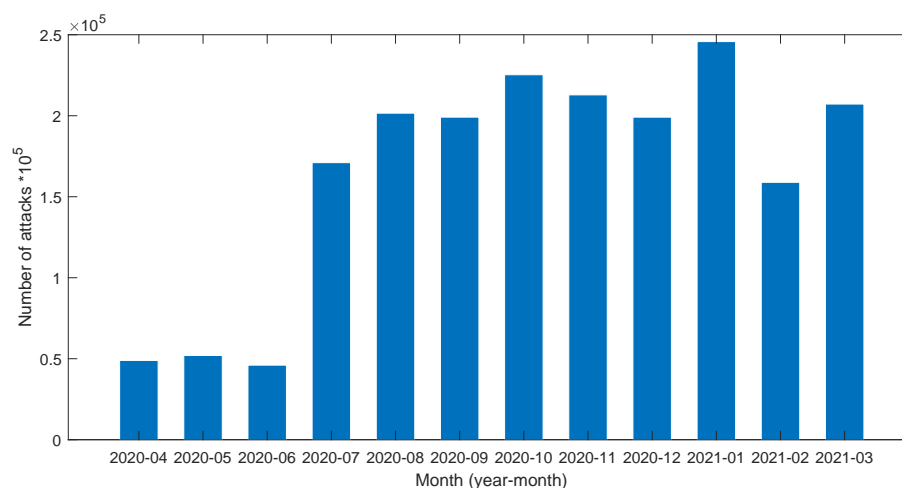


Figure 1. Trend of phishing attacks.

In recent years, researchers have developed various anti-phishing techniques to combat phishers. These techniques detect phishing attacks by utilizing different data, including URLs, page source codes, page screenshots, third-party data, and so on. For example, the basis of list-based techniques is the ability to collect and consolidate URLs. To date, researchers have focused on how to automatically collect and generate more effective lists. Visual similarity-based techniques can display the detection results more intuitively. Bozkir et al. [5] recognized the brands of phishing webpages by localizing and classifying the target brand logos involved in page screenshots through the sole use of computer vision methods in an object detection task. In addition, machine learning-based approaches are also worthy of attention, which leverage multi-dimensional features to detect phishing pages.

Nevertheless, there are still many drawbacks which remain unsolved [6] in existing phishing detection techniques. List-based techniques are incapable of discovering zero-day attacks, and visual similarity-based techniques can be bypassed by changing the location or pixels of elements in webpages. In addition, there are some drawbacks of implementation, such as the excessive computational burden, third-party dependencies, etc. The root cause of the traditional methods being bypassed is the discrete features and single focus, which can lead to a lack of attention to the relationships between phishing websites. From another perspective, approaches that significantly increase the cost of phishing attacks may effectively reduce the occurrence of attacks. In practice, we have found that many out-of-domain links usually appear on the pages of phishing websites. Although normal websites sometimes link outside of the domain, it has been observed, in a large number of analyses, that the more links outside the domain, the greater the possibility that a website is a phishing website. Our aim is to effectively prevent and reduce such attacks. Motivated by this, we propose HinPhish, with the core idea that every webpage on the Internet cannot exist independently, and will always be connected to other pages, such as forwarding pages and loading resources. Attackers may ignore these relationships when deploying fake sites, in order to reduce the cost of the attack, thus leading to implicit differences between phishing sites and normal sites. As shown in Figure 2, phishers retain the resource links of the original pages. Of course, they can modify the links to their own domain name, but this will cost more. In particular, HinPhish models use a Heterogeneous Information Network (HIN) where we naturally model the page with domain nodes, resource nodes, and design three relations among the four links: outlier (foreign link and null link), local (local link), and relative (relative link). Then, we apply a modified Authority Ranking algorithm [7] to compute the influence of different relations to each node and obtain a quantified score for every node. Finally, machine learning algorithms are applied to make use of the node score to achieve phishing detection. By integrating

semantic meaning into the model, HinPhish not only improves the detection accuracy, but also increases the cost of attacks.

In summary, we make the following contributions in this research:

- An effective representative model—HinPhish models the domains and resources links in a webpage as a HIN, in order to describe the relation of the links, which effectively preserves the deeper semantics between the links.
- An effective phishing detection approach—HinPhish applies the modified algorithm to leverage the characteristics of different link types to calculate the phish-score of the target domain in a webpage, then leverages HIN and a machine learning algorithm to detect phishing attacks, which significantly increases the evasion cost.
- An adequate experiment—to demonstrate the effectiveness and superiority of the proposed method, we conducted sufficient comparative experiments with various state-of-the-art techniques. HinPhish exhibited an excellent performance, with an accuracy of 0.9856 and F1-score of 0.9858.

The remainder of this paper is organized as follows: Section 2 describes the related work in the literature. After introducing necessary preliminaries in Section 3, we describe HinPhish's framework and the technical details of each component in Section 4. Section 5 reports the experimental results and provides a comparison of the proposed method with other methods. We discuss the limitations of the study and future work in Section 6 and summarize our work in Section 7.



Figure 2. Example of a phishing site.

2. Related Work

2.1. Anti-Phishing Techniques

Researchers have proposed many anti-phishing techniques to counter phishing attacks. These technologies can be divided into three categories, including list-based techniques, site content-based techniques, and side channel-based techniques.

2.1.1. List-Based Techniques

List-based techniques were the first approach used to discover phishing. They use approved or unapproved resource lists (e.g., URLs, domains, digital certificates, and so on) to match phishing or normal webpages. The list of approved resources constitutes a whitelist [8,9], while the list of unapproved resources is called a blacklist [10,11].

Whitelist technology detects phishing websites by matching the similarity between the target website and the resources in the list. If the similarity between the target website and the legitimate resource is less than a certain threshold, it is considered a phishing website; however, some legitimate websites are often identified as phishing websites as they have not been entered into the whitelist, resulting in a high rate of false positives.

On the other hand, blacklist technology detects the similarity between the target website and the resources in the blacklist. If it is greater than a certain threshold, the target

website is considered to be a phishing website. In a system using this strategy, the creator of a malicious website can simply change some characters or add some confusion to bypass detection. In addition, blacklist technology is often powerless against the latest phishing websites, as they are not included in the blacklist.

List-based techniques served as a simple and effective method in the early days of this technology. However, as the network environment changed and attack techniques improved, they gradually became infeasible.

2.1.2. Site Content-Based Techniques

Site content-based phishing detection techniques [12–14] separately extract some common attributes (e.g., URL, text, and image) in the website and then use heuristic or machine learning methods for phishing detection. Although there is no guarantee that all phishing websites have these extracted attributes, there is a certain amount of under-reporting. These technologies can detect the latest phishing attacks that list-based technology cannot. Moreover, some excellent attributes can also lead to a satisfactory under-reporting ratio.

Extracting features from URLs is a commonly used method in this category of techniques. Adebowale et al. [15] and Zhang et al. [16] detected phishing websites by counting the number of characters, hyphens, redirects, numbers, and other information. Similarly, other researchers [17,18] have detected phishing websites by judging the semantics of URLs [19]; for example, they checked whether the URLs contained an IP address, brand name, https protocol tag, special characters (?, @, *, -, /), and specific words.

Extracting features from source code is another technique that is often used. Researchers identify the legitimacy of websites by extracting features from webpage elements, including keywords, tags, and images [9,20–23]. In addition, some works have attempted to find a breakthrough, in terms of comparing the visual similarity [24] between suspicious and normal websites. With the development of deep learning technology, this kind of research has made significant progress [25] in recent years. Finally, another line of work [18,26,27] attempts to detect phishing websites based on the relationships between hyperlinks, for instance, by analyzing the proportion of foreign links, broken links, common URLs, and null links. Our work, similarly, focuses on relationship of hyperlinks. The difference is that we regard the relationships between links as the different edges in HIN and construct the HIN model of link objects to discover complex semantic information.

2.1.3. Side Channel-Based Techniques

Side channel-based techniques integrate the various data provided by other services for phishing detection. Commonly used data include DNS records [28], WHOIS, and search engine information. The WHOIS [27,29] involves the age of domains, the registered date and expiry date of domains, and so on. However, after the implementation of the GDPR (General Data Protection Regulation [30]), most WHOIS information is no longer available. Search engine-based techniques [31–34] collect information from different search engines to detect phishing. At the same time, the low PageRank value of a webpage is often thought of as evidence of a phishing website [27,29]; however, these techniques tend to send a large amount of query requests, causing a certain network and data processing burden.

2.2. Heterogeneous Information Networks

Over the few last years, an increasing number of researchers have begun to consider these interconnected, multi-typed data as heterogeneous information networks [35], and have developed analysis approaches by leveraging the rich semantic meaning of structural types of objects and links in the networks. Compared with homogeneous networks, HINs can combine different types of objects and their interactions. Thus it can achieve a better performance in various fields including information fusion, similarity measurement, link prediction, recommendation systems, and other data mining tasks [36]. As for the cybersecurity field, Hindroid [37], for the first time, used a heterogeneous information

network to identify Android malware. HinCTI [38] established a cyberthreat intelligence model based on a heterogeneous information network, in order to identify and analyze the types of cybersecurity threats. HinDom [39] integrates the relationships between DNS client, domain, and IP address into a heterogeneous information network, and calculates the similarity of different meta-paths by transductive classification methods in order to detect malicious domain names. Following a similar research line, our work further explores the usefulness of heterogeneous information networks for phishing detection.

3. Preliminaries

3.1. Heterogeneous Information Networks

Systems in the real world are always composed of different types of components with complex interactions. However, many traditional data analysis and mining studies have ignored the differences between different objects and relationships, leading to the loss of some semantic information. In recent years, more and more researchers have modeled target systems as heterogeneous information networks (HINs) [40], which can retain more comprehensive semantic and structural information, supporting higher-level semantic mining. A HIN is defined as follows:

Definition 1 (Heterogeneous Information Network [40,41]). An information network is a directed graph $G = (V, E)$ with an object type mapping $\varphi : V \rightarrow \mathcal{A}$ and a link type mapping $\psi : E \rightarrow \mathcal{R}$. Each object $v \in V$ belongs to one particular object type in the object type set $\mathcal{A} : \varphi(v) \in \mathcal{A}$, and each link $e \in E$ belongs to one particular relation type in the relation type set $\mathcal{R} : \psi(e) \in \mathcal{R}$. The information network is called a **heterogeneous information network** if the number of object types satisfies $|\mathcal{A}| > 1$, or if the number of relationship types satisfies $|\mathcal{R}| > 1$; otherwise, it is a **homogeneous information network**.

In order to understand the complex objects and relationship types in heterogeneous information networks more clearly, the network schema is used to describe the structure of the network at the model level.

Definition 2 (Network Schema [40,41]). The network schema is a meta-template for an information network $G = (V, E)$, with the object type mapping $\varphi : V \rightarrow \mathcal{A}$ and the link type mapping $\psi : E \rightarrow \mathcal{R}$. It is denoted as $T_G = (\mathcal{A}, \mathcal{R})$. An information network following a network schema is called a **network instance** of the network schema. For a link type R connecting object type S to object type T (i.e., $S \xrightarrow{R} T$), S and T are the source object type and target object type of the link type R , which can be denoted as $R.S$ and $R.T$, respectively. The inverse relation R^{-1} holds naturally for $T \xrightarrow{R^{-1}} S$.

Figure 3 shows a simple HIN model considering a few websites. It contains three types of objects: Webpage (W), Domain (D), and Resource (R). There are different relationships between different types of objects: the webpage of a website is built on a domain; a webpage can be linked to other webpages through hyperlinks; and webpages also use other resources, such as pictures, videos, and JavaScript, to enrich themselves and implement their functions. In summary, a HIN is a special information network that contains many types of objects and relationships. The network schema specifies the structural constraints of the HIN, and the network instance contains all the information about the objects and their relationships in the whole network.

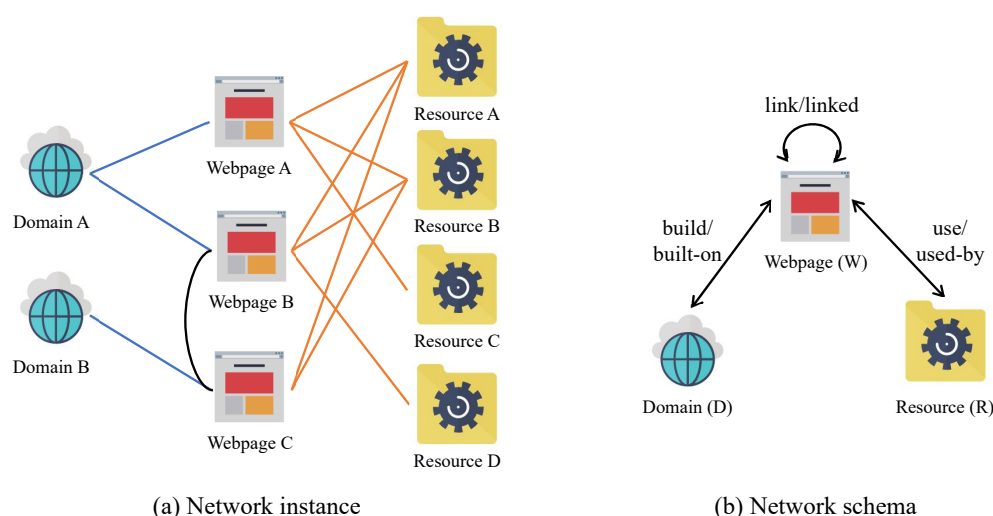


Figure 3. Example of a HIN instance (a) and its network schema (b).

3.2. Machine Learning Classification

Machine learning classification is a supervised learning method in which a computer program learns from a training set and makes predictions on new data based on the learned "knowledge". The basis of machine learning classification is the classification algorithm, which determines the efficiency and effect of classification. In this section, we review several classic classification algorithms that are often used for phishing detection.

Naïve Bayes (NB) classification is a probabilistic machine learning method. It is not only straightforward, but also powerful. It has been considered preferable in many applications, such as for the classification of text, detection of spam emails, and prediction of disease. It is based on Bayes's theorem, which states the assumption of independence among predictors. Equation (1) demonstrates how to calculate the conditional probability:

$$P(C_i|x_1, x_2, \dots, x_n) = \frac{P(x_1, x_2, \dots, x_n|C_i) * P(C_i)}{P(x_1, x_2, \dots, x_n)}, (1 < i < k) \quad (1)$$

where $P(C_i)$ is the prior probability, which represents the probability that samples belong to the i th category and $P(x_1, x_2, \dots, x_n)$ is named the probability of the evidence. $P(x_1, x_2, \dots, x_n|C_i)$ is the probability of the evidence, given that the samples belong to C_i . $P(C_i|x_1, x_2, \dots, x_n)$ is the posterior probability, which is the probability that samples belong to C_i , according to the given evidence.

Support Vector Machine (SVM) is a classifier that represents the training data as points in space, separated into different categories by a gap which is as wide as possible. New points can then be added to the space, predicting which category they fall into and which space they will belong to. It uses a subset of training points in the decision function, which makes it memory-efficient and highly effective in high-dimensional spaces.

Random Forest (RF) is an ensemble learning method for classification, regression, and similar processes. It operates by constructing a multitude of decision trees during the training stage and outputs the class as the mode of the classes, classification, or mean prediction (regression) of the individual trees. A random forest is a meta-estimator that fits a number of trees on various subsamples of data sets and then uses an average to improve the accuracy in the model's predictive nature. Due to its forest structure, the instabilities of the individual decision trees can be mitigated.

4. HinPhish

The core idea of HinPhish was motivated by the observation that every webpage on the Internet cannot exist independently; it will always be connected with other pages or resources, such as forwarding pages, loading resources, and so on, for a variety of reasons. For normal sites, the relationships among links in the pages are cohesive; namely,

the domain names of links in pages are the same or related to the visited domain name. However, phishers may ignore this relationship when deploying mimic sites, in order to reduce the cost of attacks. So, for phishing sites, there are often unrelated domain names, leading to a divergent relationship among the links in pages.

Based on the above observations, our work assumes that different types of relations among domain names influence the importance of domain names in the proposed HIN model. Here, we name the importance score the phish-score. According to different purposes, we divide links into two types of nodes: domain nodes and resource nodes. Further, for the four categories of links, we designed three relationships: outlier, local, and relative. In addition, we apply a modified algorithm to leverage the character of different link types to calculate the phish-score of the node objects in the webpage. Finally, the node attributes in HIN are used as input to machine learning algorithms, in order to detect phishing attacks.

As shown in Figure 4, there are four main components in the proposed HinPhish model: Information Extraction, Model Construction, HinPhish-score Computation, and Website Classification. After extracting links from webpages (step 1), a HIN model is built (step 2). Then, we calculate the feature matrix (step 3). Based on these features and label information, the machine learning classifier classifies the website set (step 4). Finally, HinPhish outputs the final detection results (step 5). In the following, we introduce each component in detail.

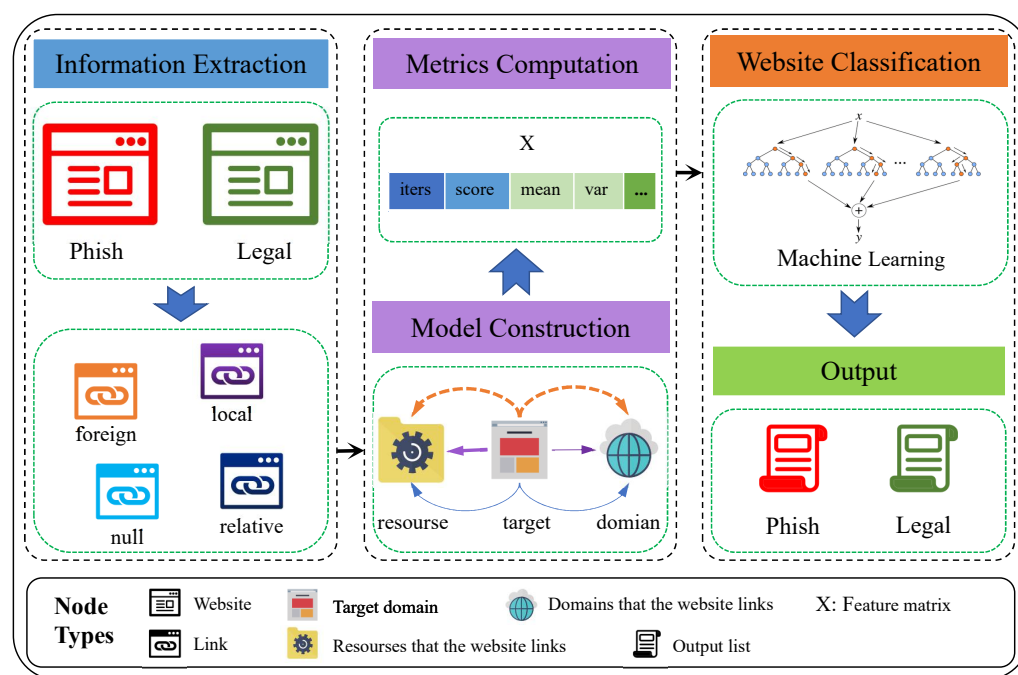


Figure 4. System architecture of HinPhish.

4.1. Information Extraction

The underlying data of HinPhish come from webpages. At present, the vast majority of webpages use hyperlinks to connect to other webpages or sites. This allows the webpages to load resources (e.g., pictures, CSS) and communicate with other hosts (e.g., submitting forms) from places other than themselves, as well as allowing users to easily access other websites. Phishing websites use these characteristics to deceive users, in order to achieve the purpose of stealing information. The sensitive information submitted by the users on the phishing website will be sent to the attacker's server. In the source code, it often appears that there will be more foreign links or null links on the website.

Therefore, we first use a crawler to obtain the HTML source code of websites. Then, we extract all the links from the source code. These links are derived from four types of

tags in the page: (i) the <a> tag is used to set a hypertext link, which connects to a new page; (ii) the “src” attribute of the tag is used to indicate the source of an image; (iii) the <link> tag defines the resource reference address; and iv) the “src” attribute of the <script> tag is used to indicate the location of an external script. According to the source of these links, we classify the <a> tags as domain objects, and other tags as resource objects.

4.2. Model Construction

After obtaining the links in the webpages, we use implementation-defined rules (Equation (2)) to divide the links into four categories: local links, foreign links, null links, and relative links. The label of the website—namely, “Phish” or “Legal”—is determined by the implementation of the given data set.

$$\text{Link Type} = \begin{cases} \text{null link} & \text{link is empty,} \\ \text{relative link} & \text{domain(link) is empty,} \\ \text{local link} & \text{domain(link) = domain(target_URL),} \\ \text{foreign link} & \text{domain(link) \neq domain(target_URL),} \end{cases} \quad (2)$$

where “target_URL” is the website’s URL to be detected and “domain()” refers to extracting the hostname.

Figure 5 shows two examples of link relationship networks extracted from real websites. The left is a phishing website, while the right is a legitimate website. The number on the edges represents the number of links. From the color and number of lines, we can see that there are obvious differences in the link types between phishing and legitimate websites: phishing websites have more foreign links (orange), while legitimate websites tend to have more local (purple) and relative (dark blue) links.

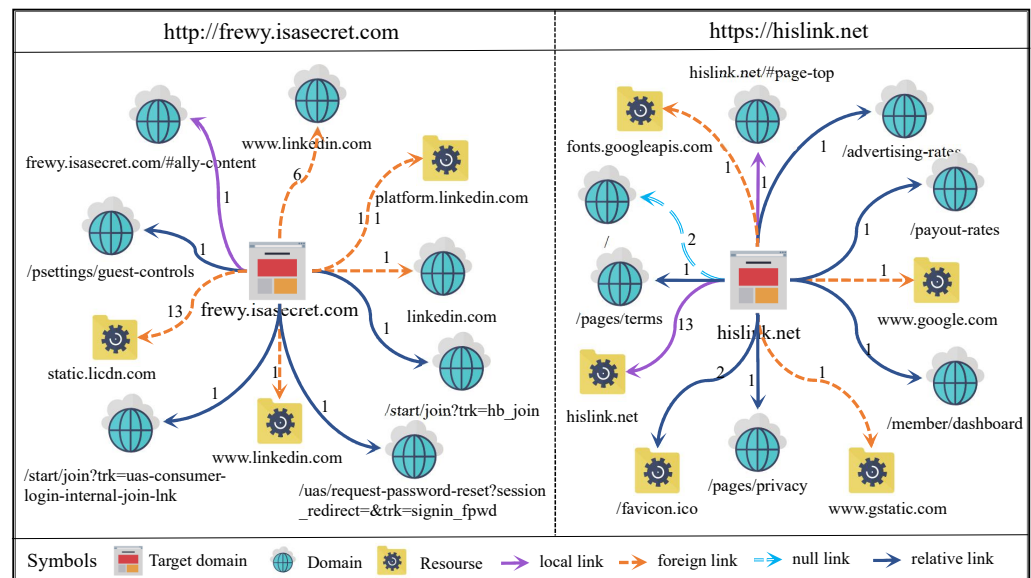


Figure 5. Instances of different websites.

As shown in Figure 6, HinPhish constructs each webpage as a HIN, based on the extracted information. Although their network instances are different, their network modes are the same, including the target domain, other domains, and resources. The target domain, other domains, and resource objects are connected by three different relationships. It is worth noting that, in this work, we believe that foreign links and null links are important features of phishing websites. Therefore, we denote these two types of links as outlier relations. This is also reflected in the subsequent relation matrices.

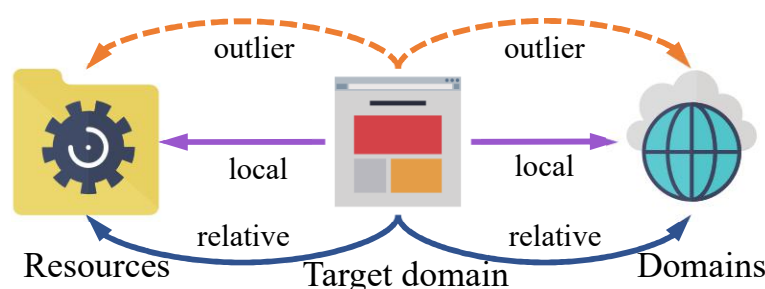


Figure 6. HIN schema in HinPhish.

4.3. HinPhish-Score Computation

Different types of objects and relationships play different roles in the network. The connection-based importance calculation method spreads the importance from different objects through a connection [7], while the PageRank algorithm believes that the ranking of a website is closely related to the rankings of its backlinks [42]. Similarly, we found that such characteristics exist in HINs, but we focused on the relation types among links, rather than the number of links. After extending it to the HIN, which contains a variety of objects and relations, we can assess the influence of different relations on node attributes by assigning different weights. Therefore, considering the fact that phishing websites tend to have more foreign links and null links, we clarified the basic compliance of score propagation in HinPhish. In our work, we name this the phish-score. Specifically, a local relation will increase the phish-score of a target domain, an outlier relation will seriously reduce the phish-score of a target domain, and a relative relation will slightly weaken the phish-score of a target domain. Among them, the increase or decrease is determined by the positivity or negativity of the adjacent matrix element, and the degree of increase or decrease is determined by the absolute value of the element. Based on such considerations, we formulated an adjacency matrix that is different from the previous ones. A description of the structure of the relation matrix is provided in Table 1.

Table 1. Definition and description of the relation matrices.

Matrix	Element	Relation	Description
D	$D(i, j)$	local	if domains i and j have k local links, $D(i, j) = D(j, i) = 2k$; otherwise, $D(i, j) = D(j, i) = 0$
		outlier	if domains i and j have m outlier links, $D(i, j) = D(j, i) = -2m$; otherwise, $D(i, j) = D(j, i) = 0$
		relative	if domains i and j have n relative links, $D(i, j) = D(j, i) = n$; otherwise, $D(i, j) = D(j, i) = 0$
R	$R(i, j)$	local	if domain i and resource j have k local links, $R(i, j) = R(j, i) = 2k$; otherwise, $R(i, j) = R(j, i) = 0$
		outlier	if domain i and resource j have m outlier links, $R(i, j) = R(j, i) = -2m$; otherwise, $R(i, j) = R(j, i) = 0$
		relative	if domain i and resource j have n relative links, $R(i, j) = R(j, i) = n$; otherwise, $R(i, j) = R(j, i) = 0$

In order to distinguish the effects of different types of relationships on the transfer of scores, in the adjacency matrix, we set the weight of the outlier to a negative number. Similarly, in order to amplify the character, we give each local and outlier a weight of “2” or “−2” instead of “1”. This is significantly different from the traditional HIN adjacency matrix construction method.

Based on the previous analysis, we have made it clear that different objects have different importances, and their importance will spread along different relationships. We classify the objects into two types—domain objects (the target domain is also included) and

resource objects—and formulate an iterative calculation method for importance scores. It should be noted that, under normal circumstances, we believe that there is no relationship between resource objects, such that the importance of resource objects is not transferred to each other. Therefore, the calculation formulas for the importance of domain objects and resource objects will be different. The formula is described as:

$$\overrightarrow{DS}^{(k+1)}(i) = \alpha \sum D(i, j) * \overrightarrow{DS}^k(j) + (1 - \alpha) \sum R(i, j) * \overrightarrow{RS}^k(j), \quad (3)$$

$$\overrightarrow{RS}^{(k+1)}(j) = \sum R^T(j, i) * \overrightarrow{DS}^k(i), \quad (4)$$

where \overrightarrow{DS} and \overrightarrow{RS} are vectors composed of the importance scores of the two types of nodes, $\overrightarrow{DS}(i)$ is the importance score of domain i , the superscript number $k + 1$ represents the result of the iterative calculation round $k + 1$, D and R represent the relationship matrices, R^T is the transpose of R , and the parameter α (with a value of 0.6 in our experiments (which will be discussed in detail in Appendix A)) is used to control the weights of the values propagated by the two types of nodes. In each round of iteration, the two types of objects are calculated separately, until the calculation results converge, and the convergent value is considered the final importance score. Based on the previous analysis, we designed the HinPhish phishing website detection index calculation algorithm as Algorithm 1.

Algorithm 1. HinPhish-score Computation

Input: Matrices D, R ; $\alpha = 0.6$

Output: $iters, Dom_score : \overrightarrow{DS}, Res_score : \overrightarrow{RS}$

```

1: // Initialization
2: for  $i$  in  $len(D)$  do
3:    $\overrightarrow{DS}(i) = 1/len(D)$ 
4: for  $j$  in  $len(R)$  do
5:    $\overrightarrow{RS}(j) = 1/len(R)$ 
6:  $iters = 0$ 
7: while  $iters < T$  do // The maximum number of iterations
8:    $iters = iters + 1$ 
9:    $tmp\_DS = \overrightarrow{DS}.copy()$ 
10:  if  $tmp\_DS - \overrightarrow{DS} < 10^{-7}$  then
11:    Break
12:   $\overrightarrow{DS} = \alpha * tmp\_DS * D + (1 - \alpha) * \overrightarrow{RS} * R$ 
13:   $\overrightarrow{RS} = \overrightarrow{DS} * R^T$ 
14:  // Normalization
15:   $sum\_D = \overrightarrow{DS}.sum()$ 
16:   $sum\_R = \overrightarrow{RS}.sum()$ 
17:  for  $i$  in  $len(D)$  do
18:     $\overrightarrow{DS}(i) = \overrightarrow{DS}(i)/sum\_D$ 
19:  for  $j$  in  $len(R)$  do
20:     $\overrightarrow{RS}(j) = \overrightarrow{RS}(j)/sum\_R$ 
21: return  $iters, \overrightarrow{DS}, \overrightarrow{RS}$ 

```

4.4. Website Classification

According to the proposed algorithm, we can calculate the phish-score of all domains and resources in the website. We believe that abnormal links will lead to an imbalance in the overall score, so we also recorded the mean and variance as features. The number of iterations indicates the speed of calculation convergence and reflects the complexity of the relationship in the website, which we also recorded. In addition, the number of

different types of objects and links are also important features that we considered. In short, the features we obtained are as described in Table 2.

Table 2. Features and descriptions.

Features	Description	Range
score	The phish-score of the target domain	$[-1, 1]$
iters	Iterative rounds for fractional convergence	$[1, 500]$
dom_local_link	The number of local links to a domain	$[0, +\infty)$
dom_foreign_link	The number of foreign links to a domain	$[0, +\infty)$
dom_null_link	The number of null links to a domain	$[0, +\infty)$
dom_relative_link	The number of relative links to a domain	$[0, +\infty)$
res_local_link	The number of local links to a resource	$[0, +\infty)$
res_foreign_link	The number of foreign links to a resource	$[0, +\infty)$
res_null_link	The number of null links to a resource	$[0, +\infty)$
res_relative_link	The number of relative links to a resource	$[0, +\infty)$
dom_mean	Average of all domain object phish-scores	$[-1, 1]$
dom_var	Variance of all domain object phish-scores	$[0, 1]$
mean	Average of all object phish-scores	$[-1, 1]$
var	Variance of all object phish-scores	$[0, 1]$

After obtaining these attributes, HinPhish inputs them as features into classifiers. The pre-trained classifier classifies the samples based on these features, and outputs a “Phish list” and “Legal list”.

5. Experiments Evaluation and Comparing

5.1. Data Sets

To evaluate the performance of our approach, we chose the newest phishing data set published by Phishpedia [25] as the experimental data set. Table 3 shows the details of the data set. The raw phishpedia data set contains 29,496 phish samples and 30,649 benign samples, including page HTML source code, page screenshots, basic URL information, and the location coordinates of the logo. The phish samples were collected from URLs provided by Openphish [43]. The benign samples were collected from the top-rank Alexa list [44]. It should be noted that, when dealing with the raw data, we found that there were many invalid data, which lacked page HTML source code or a page screenshot. Therefore, we cleaned the data without affecting the integrity of the data set. Moreover, duplicate samples were discarded, in order to avoid too many repeated features polluting our data. We carried out our experiments using the phishpedia-cleaned data set.

Table 3. Details of the used data set.

	Benign Sample	Phish Sample
Invalid data	8397	453
Duplicate data	0	3765
Phishpedia-origin	30,649	29,496
Phishpedia-cleaned	20,556	20,949

5.2. Evaluation Metrics

To quantitatively evaluate the performance of HinPhish and other methods of phishing detection, we used several classic machine learning evaluation metrics. In our data set, we set phishing websites as positive (Phish) and legitimate websites as negative (Legal). The specific metrics and descriptions are given in Table 4.

Table 4. Metrics involved in performance evaluation of phish detection methods.

Metrics	Description
TP	# of websites correctly classified as Phish in test websites
TN	# of websites Correctly classified as Legal in test websites
FP	# of websites Incorrectly classified as Phish in test websites
FN	# of websites Incorrectly classified as Legal in test websites
Accuracy	$(TP + TN) / (TP + TN + FP + FN)$
Precision	$TP / (TP + FP)$
Recall	$TP / (TP + FN)$
F1_Score	$2 \times Precision \times Recall / (Pre + Recall)$
ROC	A curve plots the TPR vs. FPR at different classification thresholds
AUC	The area under the ROC curve, which indicates how much the model is capable of distinguishing between classes.

5.3. Experiment 1: Evaluation of HinPhish with Different Algorithms

In this experiment, we used the metrics of node attributes in HIN as features and evaluated different classification algorithms to determine the most efficient algorithm. We tested three classification algorithms: Naïve Bayes (NB), Support Vector Machine (SVM), and Random Forest (RF). In the cases of randomly selecting 10%, 20%, 30%, 40%, and 50% of the test set, the accuracy and F1-score were recorded. In addition, we also performed five- and ten-fold cross-validations, and the corresponding results are also given. The detailed results are shown in Table 5.

Table 5. Detection results with different algorithms and test set ratios.

Test Set Ratio	Metrics of Each Method					
	NB		SVM		RF	
	Accuracy	F1-Score	Accuracy	F1-Score	Accuracy	F1-Score
10%	0.8853	0.8943	0.9388	0.9402	0.9854	0.9856
20%	0.8846	0.8941	0.9358	0.9378	0.9844	0.9847
30%	0.8856	0.8948	0.9338	0.9360	0.9846	0.9848
40%	0.8846	0.8940	0.9332	0.9355	0.9839	0.9842
50%	0.8853	0.8947	0.9316	0.9342	0.9826	0.9829
5-cross	0.8776	0.8877	0.9487	0.9494	0.9848	0.9850
10-cross	0.8774	0.8876	0.9493	0.9501	0.9856	0.9858

In general, RF performed the best, SVM followed, and NB performed the worst. In the terms of the training set and test set, 10-fold cross validation performed best. This fully reflects the effectiveness of the random forest algorithm and ten-fold cross-validation in reducing overfitting and improving accuracy. Therefore, we took the performance of the random forest algorithm in the case of ten-fold cross-validation as the final result of our proposed method.

We reproduced several state-of-the-art and classic phishing website detection methods on our data set and compared them with the proposed approach. Their performances are provided in Table 6 and are described in detail in the following subsections.

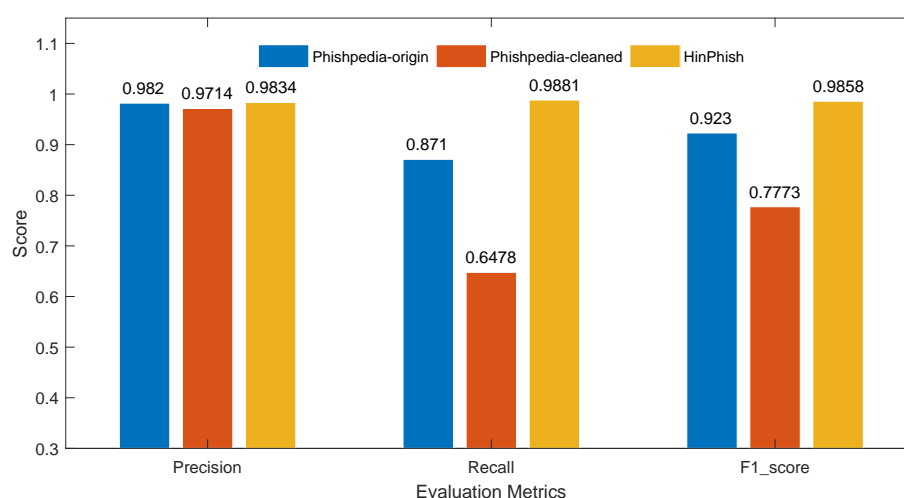
Table 6. Descriptions and performances of different approaches.

Approach	Algorithm	Performance			
		Accuracy	Precision	Recall	F1-Score
phishpedia (cleaned data)	NN	0.8121	0.9714	0.6478	0.7773
LDP	Heuristic	0.7816	0.8815	0.6525	0.7499
Jail-phish	Heuristic	0.8667	0.8692	0.8875	0.8783
HHP	Heuristic	0.9835	0.9788	0.9882	0.9835
Desai et al. [45]	RF	0.9154	0.9546	0.8740	0.9125
Shahrivari et al. [46]	RF	0.9797	0.9772	0.9834	0.9803
Shirazi et al. [18]	GB	0.9510	0.9429	0.9608	0.9518
HinPhish	RF	0.9856	0.9834	0.9881	0.9858

¹ NN, Neural Network. ² GB, Gradient Boosting.

5.4. Experiment 2: Comparison of HinPhish with Visual Similarity-Based Methods

To lure victims to trust phish pages, the pages generally utilize the logo of the original site to achieve visual consistency. Visual Similarity-based Methods check the visual consistency to detect and identify phish sites. Phishpedia is a hybrid deep learning system, which has been applied to different neural network models to discover logos on pages and compare the detected identity logo with logos in the target brand list. Moreover, Phishpedia has designed a new simple adversarial defense technique to counter some of the well-known gradient-based adversarial attacks. We downloaded the source code and data set off Phishpedia and tried to reproduce it. However, we encountered some problems. The data set problem is discussed in Section 5.1. We found that the source code did not run successfully, which may have been caused by an update of the dependence package or for other reasons. We fixed the problems but could not achieve the detection effect described in the original text. This was an unexpected result, as we were sure to use the same data, same model, and same threshold. To provide a more comprehensive evaluation, we provide the original experimental results in Figure 7.

**Figure 7.** The experimental results with Phishpedia and HinPhish.

5.5. Experiment 3: Comparison of HinPhish with Search Engine-Based Methods

Phishing detection approaches based on search engines leverage the site reputations provided by various search engines. Generally, search engines prefer to collect more information about normal sites than phish sites. In this respect, the three approaches

we chose each had certain advantages. LDP [32] is a search engine-based lightweight phishing detection system, which only chooses the top-level domain and page title as search keywords, having the advantages of fast response and less redundant checking, among others. Jail-phish [34] advances the search keywords, through a dynamic generation algorithm designed for search keywords. Moreover, Jail-phish has the ability to detect phishing sites hosted on compromised servers. The difference of HHP [47] is that HHP applies advanced search engine search syntax, allowing it to collect more accurate site data. In addition, two decision strategies for HHP were also proposed, in order to detect phishing sites hosted on compromised servers. Table 6 shows the performance of these approaches on the phishpedia-cleaned data set. As can be seen, the effects of LDP and Jail-phish did not meet our expectations, which may have been caused by the unstable site data provided by the search engines. However, HHP could deal with this issue through use of the advanced search syntax. Therefore, among these three methods, HHP had the best performance. In Figure 8, we show a histogram of the performance of these three methods and HinPhish. From the figure, we can see that HinPhish still had a comparative advantage, compared with the most advanced search engine-based methods.

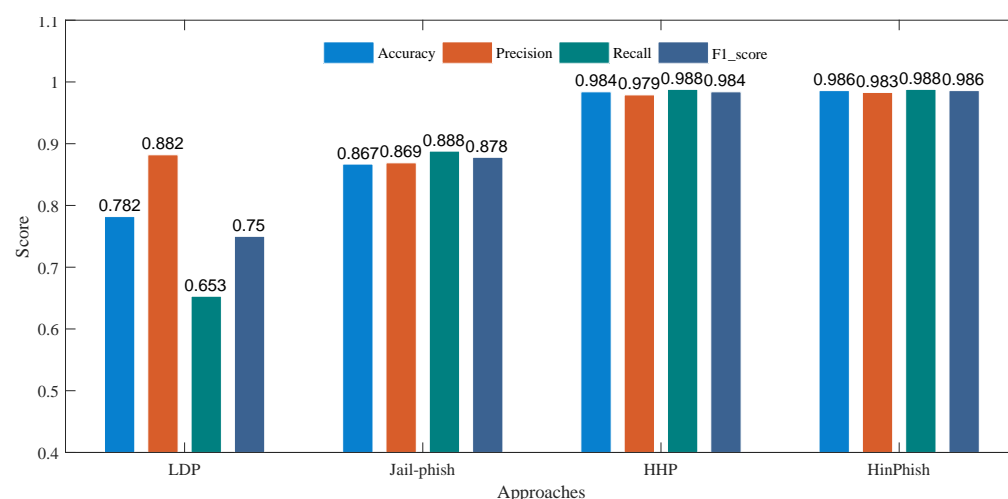


Figure 8. Comparison of HinPhish with Search Engine-based Methods.

5.6. Experiment 4: Comparison of HinPhish with Machine Learning-Based Methods

As mentioned earlier, many studies have applied different features, including URLs, webpages, or features provided by third-party services, in order to detect phishing websites by machine learning algorithms. In this section, we reproduce three representative works [18,45,46]. In order to better compare with the proposed method, we used the same data set. It is worth noting that some webpages were no longer accessible, especially given that the survival time of phishing websites is typically very short. Thus, we extracted the webpage features from the already saved webpage files in order to better ensure the consistency of data characteristics. Table 6 lists the performances of the considered methods.

The three works all used a few typical URL features and page features, such as domain length, https token, and hyperlinks in the page. In addition, [45,46] also integrated the characteristics of some third-party services, such as WHOIS, PageRank, and Google Index. They differed from HinPhish, in that [45,46] processed the features into binary features according to statistical laws, while HinPhish retains non-binary features. From the experimental results, it can be seen that all of the methods achieved satisfactory detection results. On our data set, among the three methods, that of [45] was the worst-performing, with an accuracy of 0.9154, while [46] had an accuracy of 0.9797. Our proposed approach performed slightly better, achieving an accuracy of 0.9856. In addition, we also provide the ROC curves of the considered methods and HinPhish, as shown in Figure 9. As they all showed an excellent performance, the curves quickly rose to a very high true positive rate. We carried out a ten-fold cross validation in all the experiments, so the ROC curves

are very smooth. Among all the methods, HinPhish had the highest AUC value, indicating that it had the best performance. This further confirms our previous conclusion.

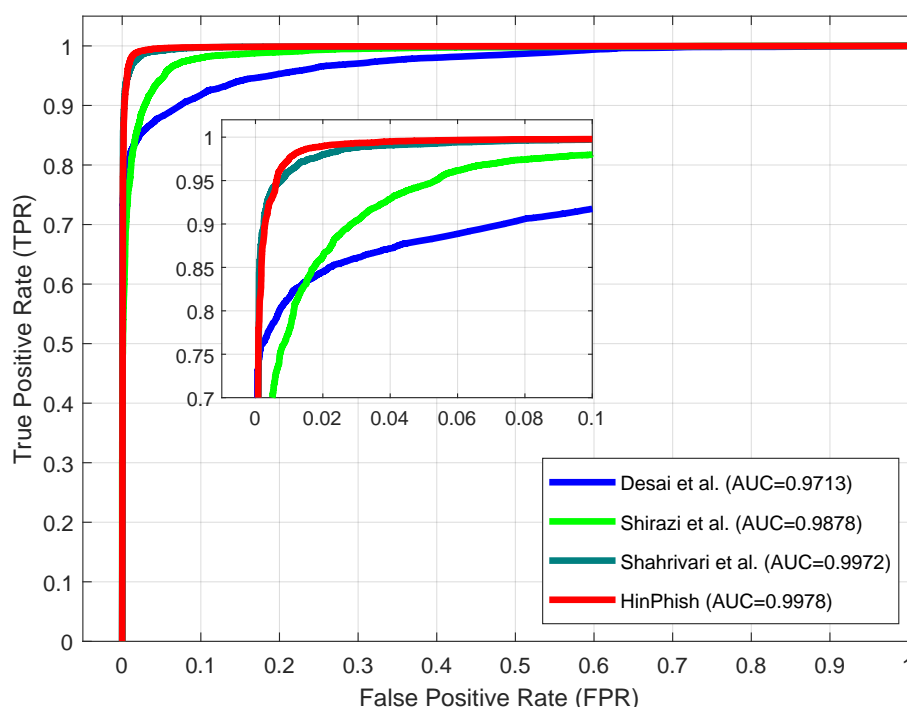


Figure 9. ROCs of Machine Learning-based Methods and HinPhish.

It is worth mentioning that Shirazi et al. [18] proposed that adding URL length as a feature can significantly improve the detection accuracy, reaching more than 0.99. We observed a similar situation on our data set; that is, consideration of the URL length can improve the accuracy of classification. Further exploration found that such results may have been caused by the bias of the data set. In Figure 10, we show the frequency curves of the URL lengths of the legitimate and phishing websites in our data set. In addition, the 500,000 URLs obtained from the pages of Alexa top 10 K revealed the frequency distribution of URL length of more general legitimate websites, marked as “Extend” in the image. Obviously, in our data set, there was a large difference between the URL length distribution of legitimate websites and phishing websites, which also explains why using the URL length as a feature significantly improved the accuracy. At the same time, we also observed a large difference in URL length distribution between our legal websites and more general legal websites. In fact, the difference in URL length between legal websites and phishing websites is not so obvious. In other words, there is a deviation between our data set and general actual websites, in terms of URL length. Therefore, we did not incorporate the implementation of [18], considering the URL length, into our comparison.

The considered machine learning-based methods can achieve quite good performances; however, they also have some insurmountable drawbacks. First, these methods need to acquire some features from third-party services. This requires a large number of query requests, which may increase the network burden. Even worse, these features have become increasingly difficult to acquire, due to an increased awareness of privacy protection. Moreover, for these detection methods using clear features, attackers can make targeted changes to bypass detection. In contrast, HinPhish only extracts links from the webpage and only needs to obtain the HTML code of the webpage to start working. Under these circumstances, attackers need to modify the links to their own domain names in order to bypass detection; if they do this, it will significantly increase their costs.

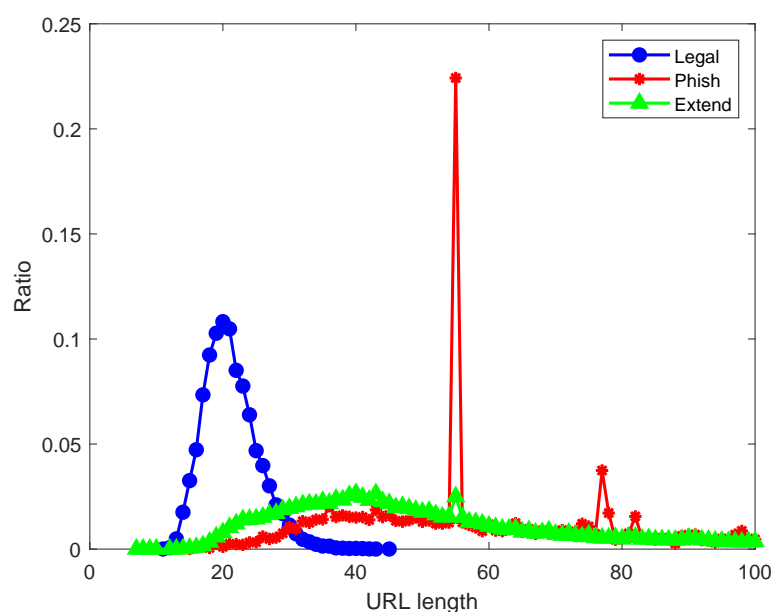


Figure 10. Ratios of websites with different URL lengths.

6. Limitations and Future Work

HinPhish achieved phishing attack detection on our data set by constructing a HIN of links on a webpage, then applying a specific algorithm. However, there may still be some potential problems for its application in the targeted environment. We discuss the limitations of the system and our future work in this section.

First, richer information is required. There are still a small number of websites that have too few connections (i.e., only one or two links, or even no links). In this case, HinPhish may exhibit a poor performance. In order to solve this problem, we plan to further crawl the secondary or tertiary links of the webpage to obtain more link relationships, in order to enrich our HIN.

Second, better parameter settings should be obtained. As mentioned in the article, in HinPhish, we merge foreign and null links into the outlier relation; that is, we do not distinguish between them. In addition, when constructing the relationship matrix, we only provide an empirical manner of setting the weights. In theory, there may exist an optimal setting method, which can further improve the detection accuracy. Although this was not the focus of this article, we will consider using statistics or other machine learning methods to find better parameter setting methods in the future.

Third, experiments with larger data sets and that detect real websites should be conducted. In our experiment, our sample size was only about 40,000 (including about 20,000 phishing websites and about 20,000 legitimate websites). In the future, we intend to verify the proposed method on more data sets, in order to enhance its generalization ability. In addition, we are also considering applying this method to real, unmarked websites, in order to assist security personnel in their analyses.

7. Conclusions

In this paper, we proposed an effective phishing domain name detection approach based on HIN, named HinPhish. HinPhish extracts various link relationships from webpages and constructs a HIN model of domains and resource objects. By applying a modified algorithm, HinPhish leverages the characteristics of different relations to effectively calculate the phish-score of each node object. Comparisons with existing techniques demonstrated that our approach exhibits better performance. In addition, the semantic information of objects and relationships based on HIN increases the cost of bypassing for phishers. In the future, we plan to improve the construction of HIN and verify the practicability of the proposed approach on a real-world data set of larger scale.

Author Contributions: Conceptualization, B.G. and M.Z.; methodology, B.G., Y.Z. and C.X.; software, B.G. and Y.Z.; validation, C.X., Y.Z. and F.S.; formal analysis, M.Z.; investigation, B.G.; resources, M.Z. and Y.L.; data curation, Y.Z. and F.S.; writing—original draft preparation, B.G. and Y.Z.; writing—review and editing, C.X., M.Z. and Y.L.; supervision, M.Z.; project administration, M.Z.; B.G. and Y.Z. contributed equally to this work. All authors read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Acknowledgments: We would like to sincerely thank the reviewers for their insightful comments that helped us improve this work.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Section 4.3 mentioned that the alpha value we used in the experiment was 0.6. Our choice is based on an experience first: the importance of domain nodes should be higher than resource nodes, because directly linking to foreign domains is more harmful than obtaining resources from foreign domains. Therefore, we selected several alpha values not less than 0.5 during the experiment and found that it reached a relatively high point at 0.6. As the focus of our work is not hyperparameter optimization, and the setting value has reached the verification effect of the method, we did not use mathematical methods to optimize this value.

However, it would be interesting to see the effect of different choices, so we chose several different alpha values to test its effect on detection performance. Specifically, we selected six different values of 0.1, 0.3, 0.5, 0.6, 0.7, 0.9, and listed their performances under the conditions of random forest algorithm and 10-fold cross-validation in Table A1. Among them, the highest value of each evaluation index is marked in bold.

Table A1. Detection performance under different “alpha” values.

α	Accuracy	Precision	Recall	F1_Score
0.1	0.9849	0.9828	0.9873	0.9850
0.3	0.9852	0.9831	0.9876	0.9853
0.5	0.9852	0.9834	0.9874	0.9854
0.6	0.9856	0.9834	0.9881	0.9858
0.7	0.9855	0.9835	0.9878	0.9856
0.9	0.9853	0.9832	0.9877	0.9855

It is seen that the performance first increased and then decreased with the increase in α , and the best performance was at 0.6, which is consistent with the value we used previously.

References

- Oest, A.; Safei, Y.; Doupé, A.; Ahn, G.J.; Wardman, B.; Warner, G. Inside a phisher’s mind: Understanding the anti-phishing ecosystem through phishing kit analysis. In Proceedings of the 2018 APWG Symposium on Electronic Crime Research (eCrime), San Diego, CA, USA, 15–17 May 2018; pp. 1–12.
- APWG. Phishing Activity Trends Report. Available online: https://docs.apwg.org/reports/apwg_trends_report_q1_2021.pdf (accessed on 11 August 2021).
- Invernizzi, L.; Thomas, K.; Kapravelos, A.; Comanescu, O.; Picod, J.M.; Bursztein, E. Cloak of visibility: Detecting when machines browse a different web. In Proceedings of the 2016 IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, 22–26 May 2016; pp. 743–758.
- Zhang, P.; Oest, A.; Cho, H.; Sun, Z.; Johnson, R.; Wardman, B.; Sarker, S.; Kapravelos, A.; Bao, T.; Wang, R.; et al. CrawlPhish: Large-scale Analysis of Client-side Cloaking Techniques in Phishing. In Proceedings of the IEEE Symposium on Security and Privacy, San Francisco, CA, USA, 23–27 May 2021.

5. Bozkir, A.S.; Aydos, M. LogoSENSE: A companion HOG based logo detection scheme for phishing web page and E-mail brand recognition. *Comput. Secur.* **2020**, *95*, 101855. [\[CrossRef\]](#)
6. Alabdan, R. Phishing Attacks Survey: Types, Vectors, and Technical Approaches. *Future Internet* **2020**, *12*, 168. [\[CrossRef\]](#)
7. Sun, Y.; Han, J.; Zhao, P.; Yin, Z.; Cheng, H.; Wu, T. Rankclus: Integrating clustering with ranking for heterogeneous information network analysis. In Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology, Saint Petersburg, Russia, 24–26 March 2009; pp. 565–576.
8. Cao, Y.; Han, W.; Le, Y. Anti-Phishing Based on Automated Individual White-List. In Proceedings of the 4th ACM Workshop on Digital Identity Management, Alexandria, WV, USA, 31 October 2008; Association for Computing Machinery: New York, NY, USA, 2008; pp. 51–60. [\[CrossRef\]](#)
9. Rosiello, A.P.; Kirda, E.; Ferrandi, F. A layout-similarity-based approach for detecting phishing pages. In Proceedings of the 2007 Third International Conference on Security and Privacy in Communications Networks and the Workshops-SecureComm 2007, Nice, France, 17–21 September 2007; pp. 454–463.
10. Prakash, P.; Kumar, M.; Kompella, R.R.; Gupta, M. Phishnet: Predictive blacklisting to detect phishing attacks. In Proceedings of the 2010 Proceedings IEEE INFOCOM, San Diego, CA, USA, 14–19 March 2010; pp. 1–5.
11. Rao, R.S.; Pais, A.R. An enhanced blacklist method to detect phishing websites. In *Lecture Notes in Computer Science, Proceedings of the International Conference on Information Systems Security, Mumbai, India, 16–20 December 2017*; Springer: Cham, Switzerland, 2017; pp. 323–333.
12. Eshete, B. Effective analysis, characterization, and detection of malicious web pages. In Proceedings of the 22nd International Conference on World Wide Web, Rio de Janeiro, Brazil, 13–17 May 2013; pp. 355–360.
13. Nguyen, L.A.T.; To, B.L.; Nguyen, H.K.; Nguyen, M.H. A novel approach for phishing detection using URL-based heuristic. In Proceedings of the 2014 International Conference on Computing, Management and Telecommunications (ComManTel), Da Nang, Vietnam, 27–29 April 2014; pp. 298–303.
14. Zhuang, W.; Jiang, Q.; Xiong, T. An intelligent anti-phishing strategy model for phishing website detection. In Proceedings of the 2012 32nd International Conference on Distributed Computing Systems Workshops, Macau, China, 18–21 June 2012; pp. 51–56.
15. Adebawale, M.A.; Lwin, K.T.; Sanchez, E.; Hossain, M.A. Intelligent web-phishing detection and protection scheme using integrated features of Images, frames and text. *Expert Syst. Appl.* **2019**, *115*, 300–313. [\[CrossRef\]](#)
16. Zhang, W.; Jiang, Q.; Chen, L.; Li, C. Two-stage ELM for phishing Web pages detection using hybrid features. *World Wide Web* **2017**, *20*, 797–813. [\[CrossRef\]](#)
17. Li, Y.; Yang, Z.; Chen, X.; Yuan, H.; Liu, W. A stacking model using URL and HTML features for phishing webpage detection. *Future Gener. Comput. Syst.* **2019**, *94*, 27–39. [\[CrossRef\]](#)
18. Shirazi, H.; Bezawada, B.; Ray, I. “Kn0w Thy Doma1n Name” Unbiased Phishing Detection Using Domain Name Based Features. In Proceedings of the 23rd ACM Symposium on Access Control Models and Technologies, Indianapolis, IN, USA, 13–15 June 2018; pp. 69–75.
19. Aljofey, A.; Jiang, Q.; Qu, Q.; Huang, M.; Niyigena, J.P. An Effective Phishing Detection Model Based on Character Level Convolutional Neural Network from URL. *Electronics* **2020**, *9*, 1514. [\[CrossRef\]](#)
20. Marchal, S.; Saari, K.; Singh, N.; Asokan, N. Know your phish: Novel techniques for detecting phishing sites and their targets. In Proceedings of the 2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS), Nara, Japan, 27–30 June 2016; pp. 323–333.
21. Chiew, K.L.; Chang, E.H.; Tiong, W.K. Utilisation of website logo for phishing detection. *Comput. Secur.* **2015**, *54*, 16–26. [\[CrossRef\]](#)
22. Chiew, K.L.; Choo, J.S.F.; Sze, S.N.; Yong, K.S. Leverage website favicon to detect phishing websites. *Secur. Commun. Netw.* **2018**, *2018*, 7251750. [\[CrossRef\]](#)
23. Alwaghid, A.F.; Sarkar, N.I. Exploring Malware Behavior of Webpages Using Machine Learning Technique: An Empirical Study. *Electronics* **2020**, *9*, 1033. [\[CrossRef\]](#)
24. Medvet, E.; Kirda, E.; Kruegel, C. Visual-similarity-based phishing detection. In Proceedings of the 4th International Conference on Security and Privacy in Communication Networks, Istanbul, Turkey, 22–25 September 2008; pp. 1–6.
25. Lin, Y.; Liu, R.; Divakaran, D.M.; Ng, J.Y.; Chan, Q.Z.; Lu, Y.; Si, Y.; Zhang, F.; Dong, J.S. Phishpedia: A Hybrid Deep Learning Based Approach to Visually Identify Phishing Webpages. In Proceedings of the 30th USENIX Security Symposium (USENIX Security 21), Virtual, Online, 11–13 August 2021.
26. Marchal, S.; Armano, G.; Gröndahl, T.; Saari, K.; Singh, N.; Asokan, N. Off-the-hook: An efficient and usable client-side phishing prevention application. *IEEE Trans. Comput.* **2017**, *66*, 1717–1733. [\[CrossRef\]](#)
27. Rao, R.S.; Pais, A.R. Detection of phishing websites using an efficient feature-based machine learning framework. *Neural Comput. Appl.* **2019**, *31*, 3851–3873. [\[CrossRef\]](#)
28. Rendall, K.; Nisioti, A.; Mylonas, A. Towards a Multi-Layered Phishing Detection. *Sensors* **2020**, *20*, 4540. [\[CrossRef\]](#) [\[PubMed\]](#)
29. Mohammad, R.M.; Thabtah, F.; McCluskey, L. Intelligent rule-based phishing websites classification. *IET Inf. Secur.* **2014**, *8*, 153–160. [\[CrossRef\]](#)
30. Voigt, P.; Von dem Bussche, A. The eu general data protection regulation (gdpr). In *A Practical Guide*, 1st ed.; Springer: Cham, Switzerland, 2017; Volume 10, p. 3152676.

31. Ramesh, G.; Krishnamurthi, I.; Kumar, K.S.S. An efficacious method for detecting phishing webpages through target domain identification. *Decis. Support Syst.* **2014**, *61*, 12–22. [\[CrossRef\]](#)
32. Varshney, G.; Misra, M.; Atrey, P.K. A phish detector using lightweight search features. *Comput. Secur.* **2016**, *62*, 213–228. [\[CrossRef\]](#)
33. Jain, A.K.; Gupta, B.B. Two-level authentication approach to protect from phishing attacks in real time. *J. Ambient Intell. Humaniz. Comput.* **2018**, *9*, 1783–1796. [\[CrossRef\]](#)
34. Rao, R.S.; Pais, A.R. Jail-Phish: An improved search engine based phishing detection system. *Comput. Secur.* **2019**, *83*, 246–267. [\[CrossRef\]](#)
35. Han, J. Mining heterogeneous information networks by exploring the power of links. In *Lecture Notes in Computer Science, Proceedings of the International Conference on Discovery Science, Porto, Portugal, 3–5 October 2009*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 13–30.
36. Shi, C.; Li, Y.; Zhang, J.; Sun, Y.; Philip, S.Y. A survey of heterogeneous information network analysis. *IEEE Trans. Knowl. Data Eng.* **2017**, *29*, 17–37. [\[CrossRef\]](#)
37. Hou, S.; Ye, Y.; Song, Y.; Abdulhayoglu, M. Hindroid: An intelligent android malware detection system based on structured heterogeneous information network. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, 13–17 August 2017*; pp. 1507–1515.
38. Gao, Y.; Xiaoyong, L.; Hao, P.; Fang, B.; Yu, P. Hincti: A cyber threat intelligence modeling and identification system based on heterogeneous information network. *IEEE Trans. Knowl. Data Eng.* **2020**. [\[CrossRef\]](#)
39. Sun, X.; Tong, M.; Yang, J.; Xinran, L.; Heng, L. Hindom: A robust malicious domain detection system based on heterogeneous information network with transductive classification. In *Proceedings of the 22nd International Symposium on Research in Attacks, Intrusions and Defenses ({RAID} 2019), Beijing, China, 23–25 September 2019*; pp. 399–412.
40. Sun, Y.; Yu, Y.; Han, J. Ranking-based clustering of heterogeneous information networks with star network schema. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, 28 June–1 July 2009*; pp. 797–806.
41. Sun, Y.; Han, J. Mining heterogeneous information networks: A structural analysis approach. *Acm Sigkdd Explor. Newsl.* **2013**, *14*, 20–28. [\[CrossRef\]](#)
42. Shen, Z.; Rehman, M.U.; Chen, W.; Liu, Y.; Liu, J.; Zhong, T. A Method based on Modified PageRank-Algorithm for Measuring and Rating Android Malwares. *Procedia Comput. Sci.* **2020**, *174*, 252–255. [\[CrossRef\]](#)
43. OpenPhish. OpenPhish. Available online: <https://www.openphish.com/> (accessed on 20 October 2020).
44. Alexa. Alexa Ranking. Available online: <https://www.alexa.com/siteinfo> (accessed on 21 October 2020).
45. Desai, A.; Jatakia, J.; Naik, R.; Raul, N. Malicious web content detection using machine learning. In *Proceedings of the 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore, India, 19–20 May 2017*; pp. 1432–1436.
46. Shahrivari, V.; Darabi, M.M.; Izadi, M. Phishing Detection Using Machine Learning Techniques. *arXiv* **2020**, arXiv:2009.11116.
47. Yunyi Zhang, S.J. Hacks Hit the Phish: Phish Attack Detection based on Hacks Search. In *Lecture Notes in Computer Science, Proceedings of the International Conference on Wireless Algorithms, Systems, and Applications, Nanjing, China, 25–27 June 2021*; Springer: Cham, Switzerland, 2021.