A NEW REAL-TIME APPROACH FOR WEBSITE PHISHING DETECTION BASED ON VISUAL SIMILARITY

by

OMID ASUDEH

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

MASTERS OF SCIENCE IN COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT ARLINGTON

May 2016

To my family

I would like to thank all of my family members for their endless help and especially my parents and my brother who made it possible for me to do my education.

ACKNOWLEDGEMENTS

First, I would like to thank my adviser Dr. Matthew Wright for his countless help during my master's studies. I appreciate all of his contributions of time, helpful ideas, and funding that helped me a lot toward achieving my educational and research objectives. I would also like to thank Dr. Farhad Kamangar, my committee member, and his students for the assistance and ideas that they have provided to me for my thesis. Finally, I want to thank Dr. Gergely Zaruba for his endless help and support even before I join UTA. I have learned a lot from him both academically and personally, and during the Data Modeling and Analysis course which has helped me a lot in all of my research activities.

April 22, 2016

ABSTRACT

A NEW REAL-TIME APPROACH FOR WEBSITE PHISHING DETECTION BASED ON VISUAL SIMILARITY

OMID ASUDEH, M.Sc.

The University of Texas at Arlington, 2016

Supervising Professor: Matthew Wright

Phishing attacks cause billions of dollars of loss every year worldwide. Among several solutions proposed for this type of attack, visual similarity detection methods have a good amount of accuracy. These methods exploit the fact that malicious pages mostly imitate some visual signals in the targeted websites. Visual similarity detection methods usually look for the imitations between the screen-shots of the web-pages and the image database of the most targeted websites. Despite their accuracy, the existing visual based approaches are not practical for the real-time purposes because of their image processing overhead. In this work, we use a pipeline framework in order to be reliable and fast at the same time. The goal of the framework is to quickly and confidently (without false negatives) rule out the bulk of the pages that are completely different with the database of targeted websites and to do more processing on the more similar pages. In our experiments, the very first module of the pipeline could rule out more than half of the test cases with zero false negatives. The mean and the median query time of each of the test cases is less than 5 milliseconds for the first module.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 What Is Phishing?

Over the past years, several definitions are proposed for the phishing attack. Some of the definitions [1] are narrow-domain targeting a specific type of the phishing attacks with more details, while some others [2] propose more general definitions covering broader types of attacks with fewer amounts of details. Since the focus of this work is more on the phishing websites, we define phishing websites as the fraudulent web pages imitating the appearance of the well-known websites in order to fool the users to reveal their information, either sensitive information or non-sensitive information that can be abused later.

1.2 Motivation

United states business losses hundreds of million dollars, if not billions, each year caused by phishing attacks [3]. Anti-Phishing Working Group (APWG) reported 123,972 unique phishing attacks worldwide over millions of fake web pages on the second half of the year 2014. Based on their report, the median uptime of a phishing web page is 10 hours and 6 minutes while the average is three times worse. The top-ten target companies have been constantly attacked even more than a thousand times per month [4]. The PhishTank's status information shows over a half a million valid phishing web-pages for the year 2015 [5]. Only for February 2016 the ScamWatch shows almost $180,000 money loss in less than 1800 reports while only %1.4 of them are reports with financial loss [6]. These alerts demonstrate that cyber criminals are

doing their job excellently, and now it is our turn to make some countermeasures in order to mitigate this huge amount of loss.

1.3 An Overview of the Existing Solutions

The phishing detection solutions are generally divided into two major categories: human-based approaches and machine-based approaches.

Although there have been a lot of efforts in the training of the users against the phishing web-pages [1, 7, 8, 9, 10], unfortunately, several studies have shown that human is not a good line of defense against this type of attack. Dhamija et.al. (2006) [11] showed 20 web-pages to 22 participants and asked them to classify them into phishing and non-phishing groups. One of the key findings of their experiment was that well-designed phishing web-pages could fool %90 of the users. In fact, on average users could correctly distinguish only %60 of the times. Another study on 2015 that does a similar experiment [12] shows only six percent increase in the ability of the users in specifying the malicious web pages. The success rate of the users in their experiment was %64, implying that after almost a decade of efforts on human training, the gain was only four percent.

The machine-based detection methods fall mostly in one of the following four classes: Non-content based methods or those that make the decision based on the information that is not a part of the web-page's content, such as the URL and the host WHOIS information [13, 14]; content-based methods or those relying on the information in the content of the web pages including texts, images, and also HTML, java scripts, and CSS codes [2, 15, 16]; those trying to find the fraudulent websites using visual similarity and image processing methods [17, 18, 19], and finally, the hybrid methods, which apply a combination of the previous three methods [20]. Next, we briefly discuss each of these four classes.

In the Non-content based scenario one approach is to apply blacklisting [21, 22] and white listing [23, 24]. The blacklisting methods usually have a database including the URLs of the fraudulent websites (usually in the server), and if the user asks for a URL that matches the blacklist, the browser will generate an active warning. For the whitelisting case, only the pages that are already authorized by the user are allowed to be visited, and the browser will generate a warning once the user asks for a URL that is not in the whitelist. Although blacklisting and whitelisting are fast methods, making them good candidates for a real-time performance, they have not the power of prediction, and therefore, they cannot detect zero-day attacks. Moreover, whitelisting methods suffer from usability issues. Another scenario is to exploit the information laying in the URL and its host to make a vector of features and use it in order to train a classifier. An example of the URL feature extraction is to check the URL length or the existence of the special characters inside the URL. Also, the host's features can include its IP address, owner's name, and its location. One of the good researches done in this area is by Ma et.al. (2009) that makes use of the online learning on the lexical and host-based features of the URL. They reported more than 95% accuracy and proposed their work for the real-time usages [13].

In content-based methods, features are extracted from the visible text and the code of the website. These features are usually used for training a machine learning classification model. Although at the first glance this approach might seem as a reliable and fast method making it a good candidate for the real-time purposes, the attackers can circumvent it using several heuristics [25, 17]. Next, we provide two of these heuristics that are mentioned by Chen et.al. (2010) [17]. An attacker can use images instead of the code when constructing a fake web page. This makes it difficult for the classifiers to decide since they cannot get the information they need from the images. Another example is injecting invisible characters in the code in a way that

makes it difficult for the tokenizers to get the words that are going to be shown by the browser after the code is compiled.

Visual similarity methods use what exactly the users see as their source of the decision. They usually use the screenshots taken from the web pages and exploit image processing methods to figure out if they are pretending other famous web pages. Although these approaches promote the accuracy, most of them are computationally expensive eliminating them from being a candidate for the real-time purposes.

Finally, the hybrid approaches try to apply a combination of the previous methods in an attempt to strike a balance between accuracy and performance.

1.4 Our Solution and Goal

We propose a multi-step framework that aims to be real-time and reliable at the same time. The detection engine of the framework is made of a pipeline of smaller classifiers. The goal of the very early classifier is to confidently rule out the bulk of pages that are not phishing attacks. The key feature of this stage is to minimize the false negative rate (if any) as much as possible and simultaneously to be as fast as possible. All of the suspicious pages will be passed to the next classifiers for more inspections. The next classifier should be more accurate while it may become slower. This will continue until the last classifier that is the most sophisticated and accurate one that may have not a good time performance. In our framework, we use visual similarity methods in order to achieve these objectives.

We provide a summary of the related works in chapter 2. Chapter 3 discusses the proposed framework. Chapter 4 includes the experiments that we have done and their results. We have the conclusion and the overall discussion on the potential issues of our approach in chapter 5. Finally, chapter 6 includes the perspective of the future works.

CHAPTER 2

RELATED WORKS AND TECHNICAL BACKGROUNDS

2.1 Related Works

The strategy of using the visual appearance of the websites as a measure of understanding the possible imitations was first proposed by Liu et. al. [26, 27, 28], and in fact several other research groups have already worked on the area [25, 19, 17, 20]. We discuss these works and some other related works in this section.

In 2005 Liu et. al. [26, 27, 28] defined the similarity of the web pages as the combination of three metrics namely: block level, layout level, and overall style similarities. For block level similarity both suspicious and legitimate pages are segmented to *salient* blocks. Then the most similar blocks of the true page within the suspicious page are considered as a match, and the block similarity is defined as the weighted average of the matched blocks.The layout similarity aims to exploit the objects' location cues, and it is defined as the number of the matched blocks that has the same position divided by the total number of the blocks. Finally, for the overall style similarity they defined some style features such as font family, background color, and line spacing, and then the similarity is defined as the normalized correlation coefficient of the histograms of the two web pages on these features. Later in 2006, they have improved the previous work by replacing the similarity measure with the Earth Mover's Distance [25].

In 2009 Chen et.al. [19] proposed a method to detect phishing websites using screen-shots of the web-pages. They exploited Harris-Laplacian [29] corner detection method in order to get the key points of the screes-shots. Then they used Contrast

Context Histograms to describe each key point as a vector. They considered both number and spatial distribution of the key points (using K-means clustering approach) for the purpose of matching. The paper reported %95-%99 of accuracy with %0.1 false positive rate. One of the important limitations of their work is that their method cannot tolerate the rotation and scaling of the objects and logos in the web page in an attempt to keep the framework fast enough.

T.-C. Chen et.al (2009) [17] try to assess the web page similarity from the human point of view. They argue that human has a specific perception of the structure and the layout of the web page and do not interpret the individual elements in a page. Based on their discussion, human collapse many features into one impression, or super signal, and use it during the decision-making process. They exploit Normalized Compression Distance as an inverse measure of the pages similarity and have reported %95 true positive rate with less than %1.7 false positive for their large scale experiment.

Afrooz et.al (2011) [20] apply a hybrid approach that is a combination of the content based and visual similarity based methods. They first make a profile database of the legitimate web pages. Profiles stores several features including SSL certificates, URL and the contents related to the sites appearance such as HTML files and extracted features of the logo. Their approach separates the classification problem for the offline and online purposes. For the offline situations, given a new web page, they suggest image matching with all of the legitimate profiles in the background using the SIFT image keypoint matching algorithm [30]. However, for the online purposes, a two-level approach is used. The first level does the content based similarity check and finds the most similar legitimate page. Then at the second level, image matching will be done with the most similar page. In their experiments, they have reported %90 of accuracy with less than %0.5 of false positives. The downside of their work

for the online part is that the first level is obviously vulnerable to countermeasures that have previously discussed in the introduction section.

2.2 Technical Backgrounds

In this section we review the necessary technical backgrounds and notation that is used in the rest of this work.

2.2.1 Histogram Similarity Metrics

Calculating the similarity of two histograms is one of the important ingredients of our experiments. Therefore, here we review some of the common techniques for the calculation of histograms' similarity.

2.2.1.1 Cosine Similarity

Suppose we have two histograms A and B each with n bins. If we consider the histograms as two vectors and the bin values of the histograms as the values of the different dimensions of the vectors, then we can define cosine similarity measure (the cosine of the angle between the two vectors) as the following:

$$\text{Cosine similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

The cosine similarity of two vectors is a value ranging from 0, representing the exact opposition, to 1, representing the exact similarity of the two vectors.

2.2.1.2 Histogram Intersection

Another important histogram similarity measure, which is used in the literature of content-based image retrieval [31], is called the *histogram intersection* method.

Suppose H(i) be the histogram of an image, where i represents the i-th bin in the histogram, then the normalized histogram I is defined as I(i) = $\frac{H(i)}{\sum_i H(i)}$. Now let Q(i) be the normalized histogram of a query image, then the similarity of Q, and I is defined as follows:

$$Similarity(Q, I) = \sum_i \min(Q(i), I(i))$$

Here, also the similarity values ranges between 0 and 1.

2.2.2 Performance Measurement

There are several performance metrics for assessing the results of the classifiers [32]. However, in this work we use only the false negative and false positive rates in order to evaluate the results. Consider the general problem, given a set of web-pages, a classifier is going to predict phishing and benign pages. We will have the following cases:

- True positives (TP) = the phishing pages that are correctly predicted as phishing
- False negatives (FN)= the phishing pages that are mistakenly predicted as benign
- False positives (FP) = the benign pages that are mistakenly predicted as phishing
- True negatives (TN) = the benign pages that are correctly predicted as benign

Table 2.1 illustrates all of these cases as a *Confusion Matrix* . False negative and false positive rates are defined as the following:

$$\text{False Negative Rate} = \frac{FN}{TP + FN}$$

$$\text{False Positive Rate} = \frac{FP}{TN + FP}$$

Next, we provide an example to clarify these concepts. Suppose we have a model

Table 2.1. Confusion Matrix

	Predicted as phishing	Predicted as benign
Actually phishing	True positive	False negative
Actually benign	False positive	True negative

that can predict phishing and non-phishing pages, and we want to test it. We have a test set of 10 phishing and 10 benign pages, and the model have resulted the confusion matrix shown in the table 2.2.

Table 2.2. Confusion Matrix example

	Predicted as phishing	Predicted as benign
Actually phishing	5	5
Actually benign	3	7

Then the false positive and false negative rates can be calculated as the following:

$$\text{False Negative Rate} = \frac{5}{5+5} = 0.5$$

$$\text{False Positive Rate} = \frac{3}{7+3} = 0.3$$

CHAPTER 3

PROPOSED FRAMEWORK

We are proposing a new framework that aims to minimize the false negatives and false positives rates and also be fast enough for the real-time purposes. In this chapter, first we talk about the overall proposed framework, and then we discuss its practical aspects.

3.1 The Big Picture

One important fact is that, based on the statistical information published about the targets of the phishing attacks [4, 5], the focus of the most of the phishing attacks is on some specific brands. At the end of the second half of the year 2014, APWG reported that Apple, PayPal, and Taobao.com together were the target of %54 of the world's phishing attacks while the next seven brands were the target for %23 of the attacks. This means that almost %80 of the phishing attacks were on the top-10 brands.

A very general framework used in several previous Anti-phishing solutions including all of the works discussed in the related works section [17, 28, 19, 20] is that intuitively, we can have a database of the most-targeted legitimate pages, and given a new web page, we can compare it with the legitimate ones and decide if it is phishing or not. However, this setting alone is not appropriate when exploiting visual similarity methods since, as previously discussed in the introduction chapter, the task of similarity checking of the images is computationally heavy and makes it difficult

for these class of methods to be good candidates for the real-time scenarios. We are going to convert this framework into a faster and more robust one.

Another important fact is that although we have a lot of phishing web pages, the number of benign pages on the Internet are much more than the malignant ones. It means that almost all of the pages that users visit every day are not fraudulent pages. However, obviously, this cannot alleviate the danger of the attacks.

Based on the aforementioned facts, we grow the idea of our framework next. If we had a method that could rule out most of the benign web pages that a user visits every day very quickly in an early stage and could inspect the remaining suspicious pages in a second deep checking stage, it could help us in improving the overall time performance. For example, suppose the users' web browser has a plugin that monitors the appearance of the pages that they visit. If the plugin can rule out %80 of the pages in 1 millisecond, but it takes 10 seconds for other %20 pages to be verified by the plugin, the amortized decision-making time for the plugin is nearly 2 seconds.

Although quickness is one objective of the early stage (we call it Shallow Detection phase), another very important feature of this stage is that it should not mistakenly rule out a phishing page as a benign one, or in other words, *the false negative rate of the early phase should approach to zero*, otherwise, it would lead the user to a dangerous scenario by allowing him to surf the phishing web page. Therefore, the Shallow Detection phase should not consider the web page as benign unless it is pretty *confident* in its decision. This approach may lead the Shallow Detector to generate a high false positive rate, but it is acceptable for this step since the false positives are going to be inspected deeply in the next phase. To summarize, the Shallow Detection phase has two important objectives: first, it is going to rule out a good portion of the benign web pages very fast, and second, its false negative rate has to approach to zero.

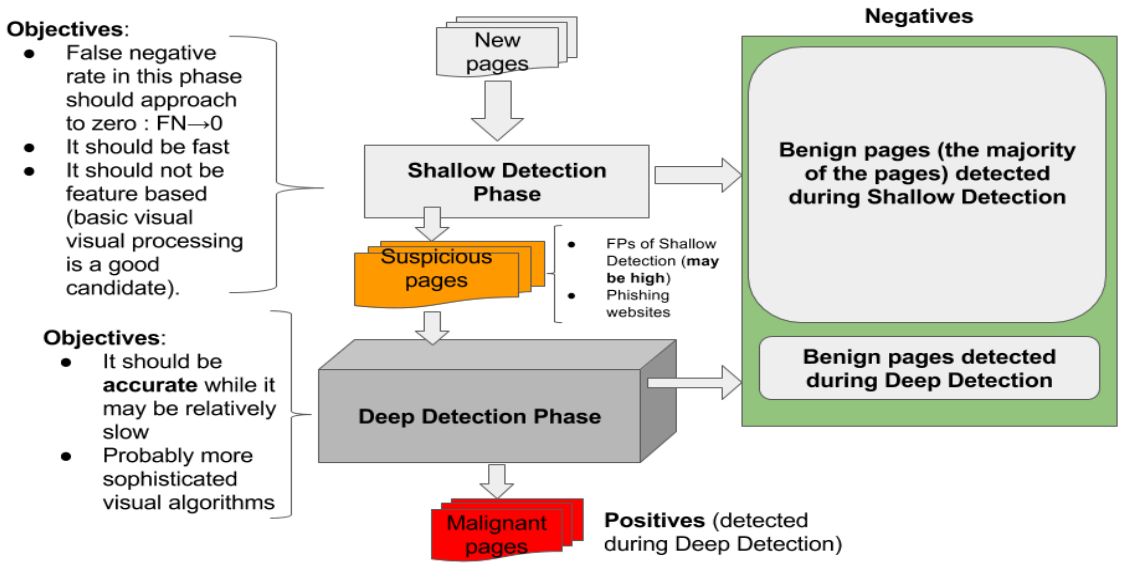

Figure 3.1. The Overall Proposed Framework.

The suspicious pages from the Shallow Detection phase will be passed to the Deep Detection phase. Since the inputs of this phase are potentially including phishing attacks, Deep Detector should be pickier and more accurate in its inspections. More accuracy means more computations that will result in a slower decision-making process, but since the number of times the Deep Detector is responsible for the decision making is much less than the Shallow Detector, the overall framework's time performance would be good enough for the real-time purposes. In fact, the Deep Detection phase can also exploit the same technique we used previously. It may include a chain of detectors each one a bit more accurate but slower than the previous one. The goal of such architecture is that once the overall system is confident enough that a page is benign, it should rule it out and no more computation is needed. To sum up, the objective of the Deep Detection phase is to be more accurate in its inspections and predictions in order to mitigate both the false positive and the false negative rates.

The figure 3.1 illustrates a big picture of the overall proposed framework discussed in this section.

3.2 The Proposed Framework in Practice

In this work, our focus is more on the Shallow Detection phase, since it is the most important part of the proposed framework. At the first glance, the content-based information including the texts, code, and the images of the web pages seems interesting for the Shallow Detection phase since the analysis of such information can be done quickly satisfying the first objective of the Shallow Detection phase. As mentioned in the Related Works chapter, Afrooz et.al. [20] used such information to find the most similar legitimate profiles in the database and then applied the SIFT image matching method [30] only on these profiles, in an attempt to decrease the number of the times that their method have to run the time-consuming SIFT algorithm. However, as discussed and exemplified in the introduction chapter, content-based methods are severely vulnerable to the heuristics that attackers use to circumvent them. Therefore, if we apply the content-based methods for the Shallow Detection phase, the attackers can simply make pages that can fool the detector to classify them as benign pages. Consequently, the detector will have a high false negative rate. This is in an obvious contradiction with the second objective of the Shallow Detection phase.

We are going to use exactly the same information that a user sees in his web browser, and that is the cause pushing us to exploit the visual similarity of the web-pages as a measure for detecting phishing websites. In general, we should have a complete database of the visual characteristics of the most targeted legitimate pages, then given a new page, the Shallow Detector should do a quick query on the database and rule the page out if it is confident that the page is not imitating any of

the brands in the database, otherwise, it should pass the page to the next level for more inspections. However, image processing operations can easily become very time consuming in a way that makes them useless and in contrast with the first objective of the Shallow Detection phase. Therefore, it is crucial that we can achieve our goals with lightweight image processing operations. Next, we will discuss the way we are going to approach this challenge.

3.2.1 Object Detection

In practice, users do not memorize every single piece of information in a web-page for the purpose of checking its authenticity in the future. Instead, they only have a general perception of the big signals within the page [17] that helps them making fast decisions. Similarly, when an algorithm is going to decide a new web page, instead of processing the whole screenshot of the page, that is indeed computationally heavy, as a pre-processing stage, it should locate the big signals inside the web page and then do the further analysis only on those signals. Ideally, logos are the most important signals in a web page, since users usually trust the logos of the brands. So, we should implement a method for extracting the logos from a page. Unfortunately, the existing object detection algorithms are computationally expensive; so we are going to make a simple logo detector that can satisfy the time restrictions. In practice, usually the background of an image, especially in the web-pages, has the dominating color/intensity in the whole image. Therefore, one simple idea is to extract the intensity histogram of the image, and then, consider the largest bin as the background and other bins, which we call it *objects' spectrum* here, as the objects in the image. Then, we make a binary image called *digest image* in the following manner: 1. Segment the original image into three by three pixel (9 pixels) cells. 2. for each cell, If the number of pixels with the intensity belong to the objects' spectrum is more than half of the

pixels in the cell, set all of the related pixels in the digest image to one, otherwise to zero. The figure 3.2 - middle column illustrates two examples of the digest image.

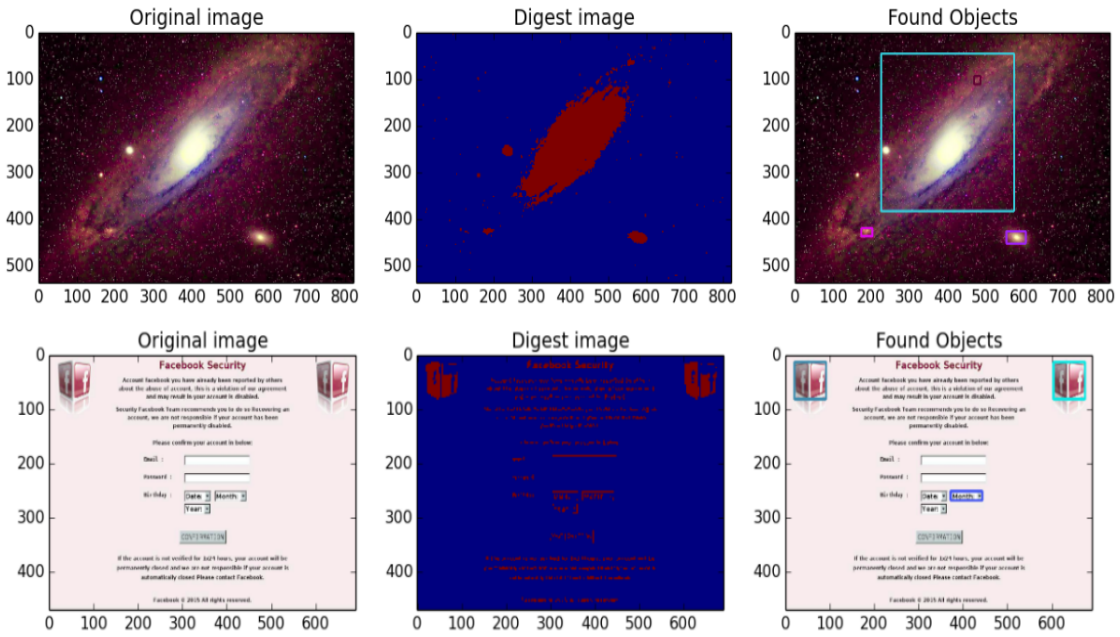

Figure 3.2. Object Detection.

Once the algorithm found the digest image, it should consider the groups of pixels that are adjacent to each other as a single object. In order to achieve this goal, the algorithm start traversing the digest image pixel by pixel until reaching to the first pixel with the value one (recall that the digest image is a binary image) and generates an initial tight rectangle surrounding the pixel. Then it grows the initial rectangle in all the directions that have adjacent pixels with the value of one. The growth of the rectangle continues until the whole solid object becomes inscribed in the rectangle. After one pass over the digest image, the algorithm will find a set of rectangles surrounding the solid objects tightly in the image. The set of rectangles may include many small rectangles, and also several rectangles each covering one part

of objects. The algorithm should filter the small rectangles and also should merge the rectangles that are beside each other. Finally, the algorithm will end up with a list of rectangles surrounding the important objects (figure 3.2 - left column).

3.2.2 Image Retrieval by Using the Orientation Histograms of the Edges

After doing logo detection, the problem will reduce to the problem of searching a new logo in a database of the legitimate logos. In other words, we store the logos of the legitimate brands in a database. Given a new page, the logo detection algorithm will extract its logos. Then the Shallow Detector should rule out the page if the logos of the page are not imitating the logos in the database, otherwise, the new page should be passed to the next phase for more inspections. This scenario will push us to the area of content-based image retrieval. There are several algorithms for accurate object detection such as SIFT [30], but their computational overhead is high, making them inappropriate for our purpose. On the other hand, recall that the task of Shallow Detector is not proving the similarity, but instead, it should rule out the pages that are not similar to those in the database quickly and confidently (without false negatives). For the purpose of image comparison, an algorithm should have a comparable representation of the images. The more information this representation includes, the more accurate the comparison result will be, but it also will be more computational. Consequently, the image representation should only have the necessary information that suffices the comparison's needs. One classic method for the content-based image retrieval is to represent the image with its histograms of color/intensity. Histogram comparison methods are quite fast methods satisfying the performance objective of the Shallow Detector. The only remaining requirement is whether it can satisfy the second objective of the Shallow Detection phase. Our pilot experiments have shown that although intensity histograms may seem a good measure of the image similar-

ity, but they are not accurate enough for a confident demonstration of dissimilarity. In fact, there may be two images that have two completely different color/intensity histograms, but they both are representing the same objects. Figure 3.3 illustrates an example of the similar images with different intensity histograms. The reason is

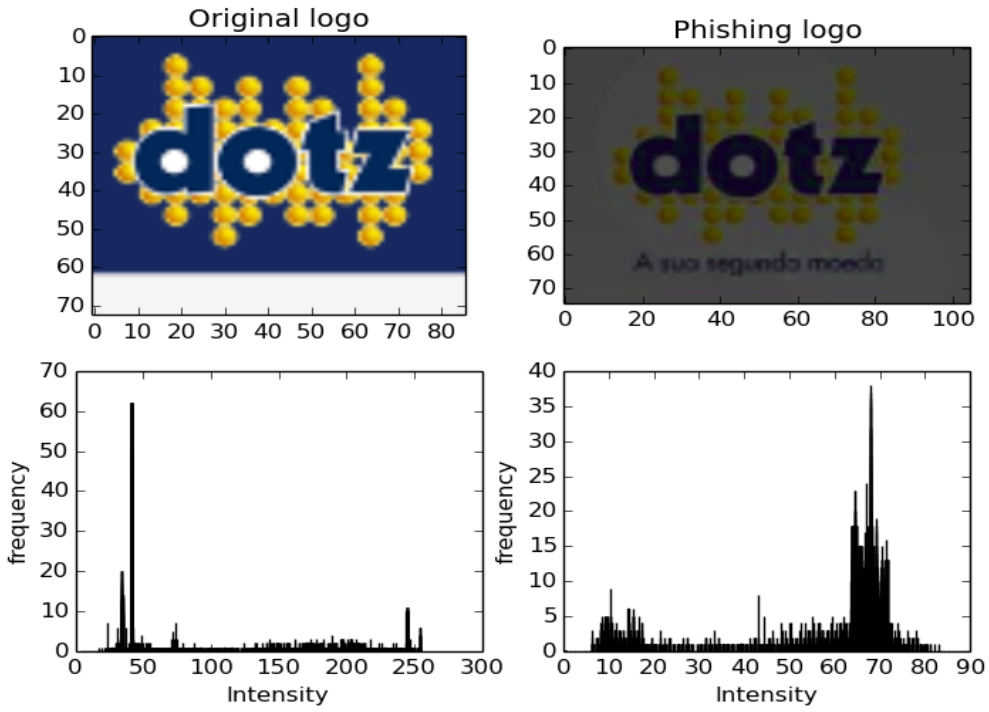

Figure 3.3. Similar images with completely different intensity histograms.

that this type of histograms can only represent the intensity-based information, and they do not convey any information about the structure of the objects. Next, we are going to introduce a method that can capture some information about the structure the objects, and simultaneously be quick enough for the Shallow Detection phase.

Recall that the missing part of the puzzle was some information about the structure of the images. Edges are the key elements in an image that can convey brief

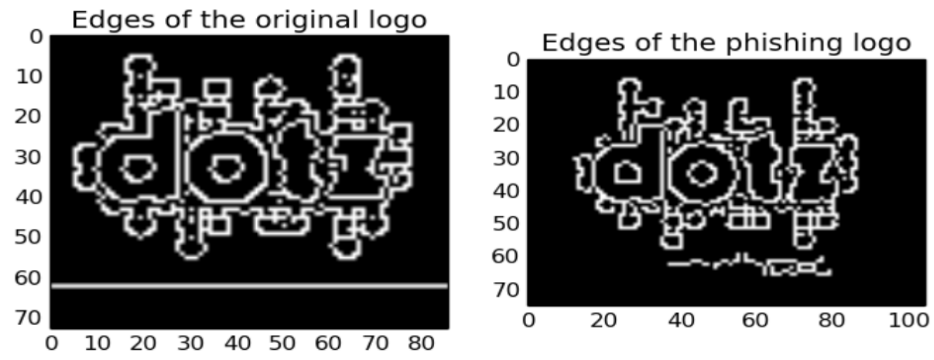

Figure 3.4. Canny Edges.

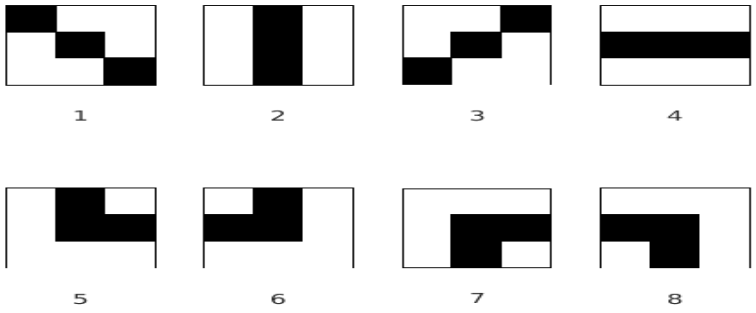

Figure 3.5. 8 major cases that happens in an edge image.

but important information about the structure of the objects in the image. Figure 3.4 shows the edges computed by Canny algorithm [33] for the previous example. As you see, the images of the edges share pretty similar information. However, since the edge images are binary images, calculating simple histograms of them are fruitless. Here we exploit a simple technique before using the histograms. One important feature of the edge images is that if we zoom in and look at each 3 by 3 pixels cell, we understand that there are only a few cases including 8 major cases. Figure 3.5 illustrates these major cases. We exploit this property in order to make an image called *derivative image*, which is going to store the information about the structure of the objects in the image. In order to make the derivative image, the algorithm sweeps a 3 by 3

pixels window over the image checking the specific case that happened in each pixel and then places the case label (a number between 1 to 8) at the same location in the derivative image. Finally, if we extract the histogram of the derivative image, we can have good information about the structure of the objects. In fact, the histogram will have 8 bins giving the frequency of each of the cases, and hopefully, the comparison of the histograms of the derivative images will give us a good measure of the structure similarity of the images. The second row of the figure 3.6 illustrates the derivative images (note that there is a difference in the brightness of the pixels in these images each for one of the 8 specific cases), and the third row shows the histograms of the derivative images.

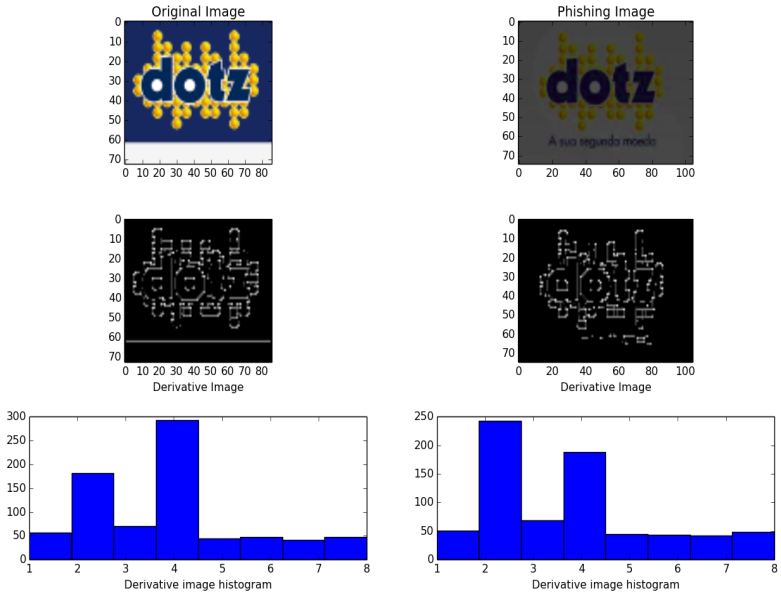

Figure 3.6. Derivative image histogram.

Another idea is that we can break the derivative image into, say four, separate square regions and calculate the histogram of each region separately. Then if we

want to compare two images we can calculate the pairwise similarity of these regions and we can get an overall similarity measure by averaging these values. We expect that this might increase the accuracy in the cost of more computations. Figure 3.7 illustrates the segmentation idea, and in this case the overall similarity is calculated as follows:

$$\text{Overall Similarity(A,B)} = \frac{\sum_{i=1}^{i=4} similarity(A_i, B_i)}{4}$$

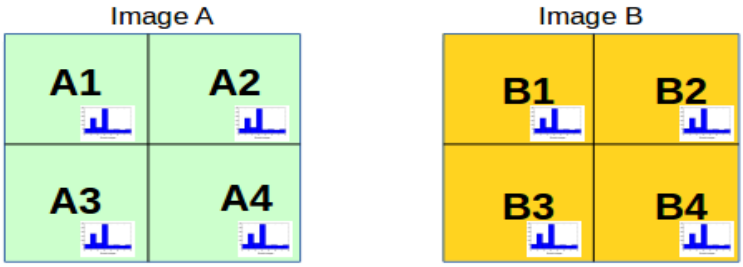

Figure 3.7. Similarity calculation for the segmented derivative image.

3.2.3 Image Retrieval by Histograms of Gradients

Sadly, our pilot experiments on exploiting the histograms of the edge orientations, that have been discussed in the previous section, have shown that if we want to get zero false negatives, the false positive ratio will increase dramatically up to about %85. Therefore, we can only rule out %15 percents of the test cases confidently. These results lead us to use a more informative representation of the images. In fact, we use the same intuition but apply a more accurate approach. Dalal et.al. [34] has firstly introduced the Histograms of Gradients as an image representation method in a human detection context. Instead of using only the edge orientations, given the

image matrix as I(x,y), we calculate the actual gradient vectors of the image pixels in both X and Y directions as follows:

$$\nabla I(x,y) = \begin{pmatrix} I_x \\ I_y \end{pmatrix} = \begin{pmatrix} \frac{\partial I(x,y)}{\partial x} \\ \frac{\partial I(x,y)}{\partial y} \end{pmatrix}$$

Where $\frac{\partial I(x,y)}{\partial x} = I(x+1) - I(x-1)$ and $\frac{\partial I(x,y)}{\partial y} = I(y+1) - I(y-1)$. Figure 3.8 illustrates the gradient images in both X and Y directions. As an example, suppose we want to calculate the gradient values for the pixel with the value 172 in the figure 3.9. Then $\frac{\partial I(x,y)}{\partial x} = 205 - 136 = 69$ and $\frac{\partial I(x,y)}{\partial y} = 182 - 162 = 20$.

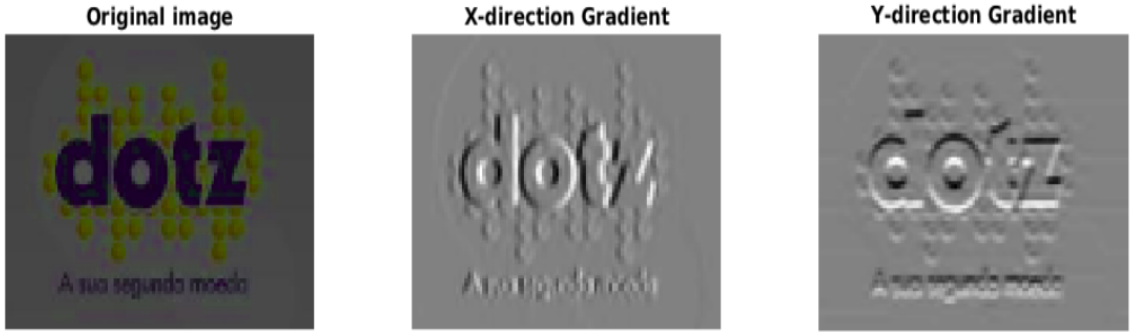

Figure 3.8. Gradient images in X and Y directions.

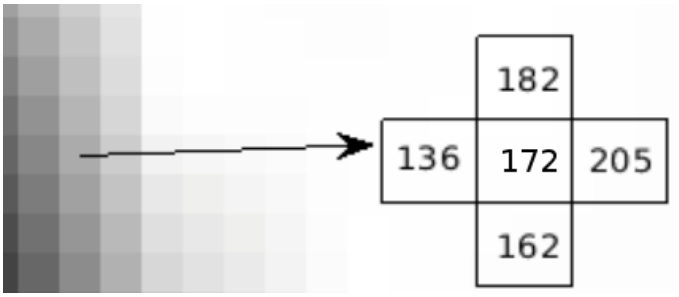

Figure 3.9. An example of pixels that contribute in the value of a specific pixel gradients.

Therefore, we will end up with a two dimensional gradient vector for each pixel of the image. We can also get the magnitude and direction of these vectors:

$$\begin{cases} \text{Gradient magnitude} = |\nabla I(x,y)| = \sqrt{I_x^2 + I_y^2} \\ \text{Gradient direction} = \theta = \arctan(\frac{I_y}{I_x}) \end{cases}$$

Then we make a histogram with 9 bins on the gradient directions of the vectors where the gradient magnitude specifies the contribution of each vector in the bins' values. Using the histogram of gradients as the image representation, we could achieve zero percent of false negative and less than % 47 percet of false positive rates.

3.3 Discussion on the Novelty of the Work

As also mentioned in the related works section, Afrooz et.al.'s detection method [20] is completely different from the approach we proposed here, since although their two-level method for online detection improves the time performance, they have not mentioned any efforts on decreasing the false negatives and consequently, the first part of their approach is vulnerable to heuristic attacks, which is discussed in the introduction chapter.

Islam et.al. [35] also propose a multi-tier phish detection method. Their approach is completely different from our framework. They use a three-level voting method in order to decide new pages. Each level has a separate classifier. If both two first classifiers labeled the new page as phishing, overall, it will be considered as phishing; if both voted a page as non-phishing, the overall result is non-phishing. In the case that one classifier labeled the page as phishing and another labeled it as non-phishing, a third classifier will make the last decision.

Viola et.al (2001) [36] have used a similar approach in the area of the real-time object detection, more specifically for the face detection purposes, but their approach is different from ours in practice.

CHAPTER 4

EXPERIMENTS AND RESULTS

4.1 Data sets

We use two major data sets in our experiments. The first one is the *valid phishing logo data set*, which includes 132 logos extracted from valid phishing pages in PhishTank. More than 1000 screenshots of the valid phishing pages, during Jan, 1st to Feb, fifth 2016, have checked by human and 132 logos were extracted (logos are not unique; for instance 8 different phishing logos targeting Facebook are among the data set). The second data set is *the legitimate logo data set*, which includes 168 legitimate logos of 65 mostly targeted brands. Monthly status of the PhishTank was reviewed for all 12 months of the year 2015 and the top 20 targeted companies was selected, added up to the most targeted brands list of APWG that was published in 2014. We also added most targeted companies among the 1000 valid phish screen-shots noticed in the phishing data-set. APPENDIX A includes the list of these brands. Then the legitimate logos of these brands were extracted from their official website by the human. Totally, 168 logos are in this dataset (some of the brands has more that one logo).

4.2 Finding a setting that minimizes False Negatives

In this experiment for each similarity threshold (ranging from 0.1 to 0.95, with a step of 0.02), and also for each different segmentation (1,4,9,16, and 25), we compared the whole valid-phish-logo data set with the whole legitimate-logo data set. For each setting, we calculated the false negative and false positive rates, as discussed in the

section 2.2, and plotted them in a three-dimensional plot illustrated in the figure 4.1. As mentioned in the previous section, the legitimate data set includes 65 brands. We made a map of these brands with 65 entries where the keys are the brand names and the values are the list of the legitimate logos of each of the brands. In fact, for each similarity threshold, each phishing logo is queried in the map and if it is similar to at least one of the logos of a brand, it will be considered as a malicious logo imitating the brand. For the purpose of similarity measurement between two logos, the algorithm calculates the gradient images of the logos, as discussed in the section 3.2.3, and then for each segmentation setting, it calculates the similarity using the histogram intersection method, which is explained in the Backgrounds (2.2) section. If the similarity was less than the similarity thresholds, two logos will be considered as not similar, otherwise, they will be considered as similar to each other. In an obvious contradiction with what we expected, in the section 3.2.2 about the effect of increasing the number of segments on similarity measurement accuracy, based on the results we got in this experiment, segmenting the logos and calculating more histograms do not improve the accuracy. As you see in 4.1, the false negative ratio was less when we just considered the whole logo as a single segment. Another important observation is that in the case we had only one segment, with a similarity threshold of 0.83 the false negative ratio is zero and the false positive rate is tolerable, less than %47. Therefore, based on this experiment, the best setting is to use only one segment with the similarity threshold of 0.83.

We stored the time taken for each malicious logo to be queried in the legitimate database in the best setting. Figure 4.2 illustrates the distribution these times, which is mostly less than 5 milliseconds (the average is 4.5 and the median is 4.1 milliseconds).

These results shows that our comparison method, in the best setting, fits appropriately with the objectives of the Shallow Detection phase. Querying the legitimate data set is fast enough for the first objective, and also the false negative rate is zero, which is exactly what the second objective asks us for.

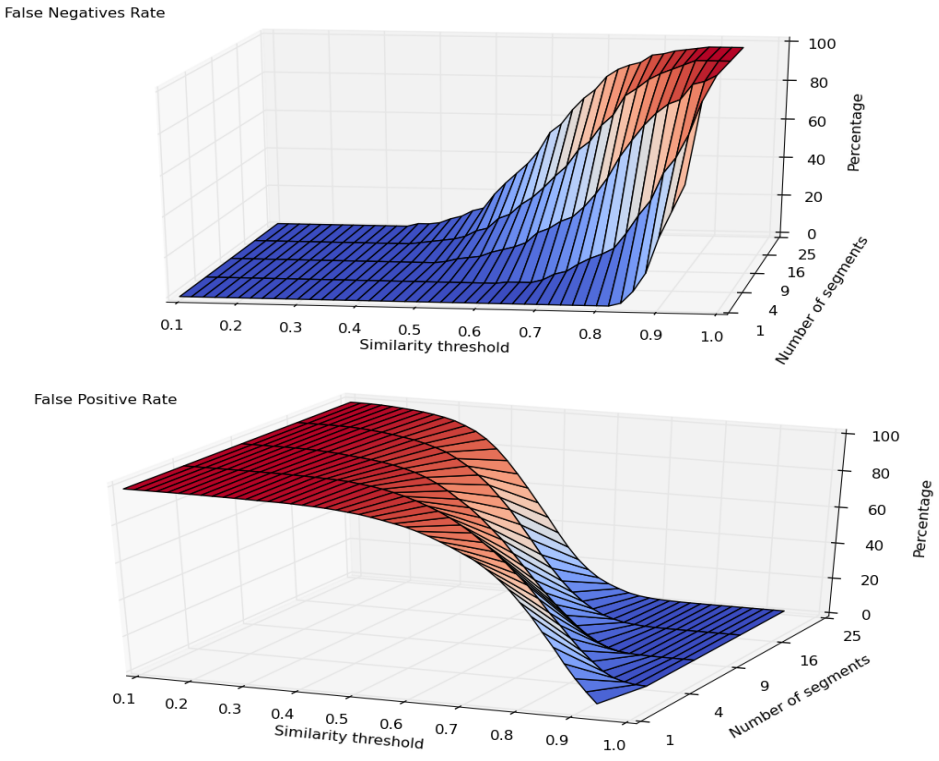

Figure 4.1. False negative (up) and false positive (down) rates for different similarity thresholds and different number of segments.

4.3 Exploring the effects of noise, scaling and rotation in the proposed similarity measure

In this set of experiments we are going to explore the effects of noise, scaling, and rotation on the similarity measure that is based on histograms of gradients of the logos. For these experiments, we have selected 32 different legitimate logos and

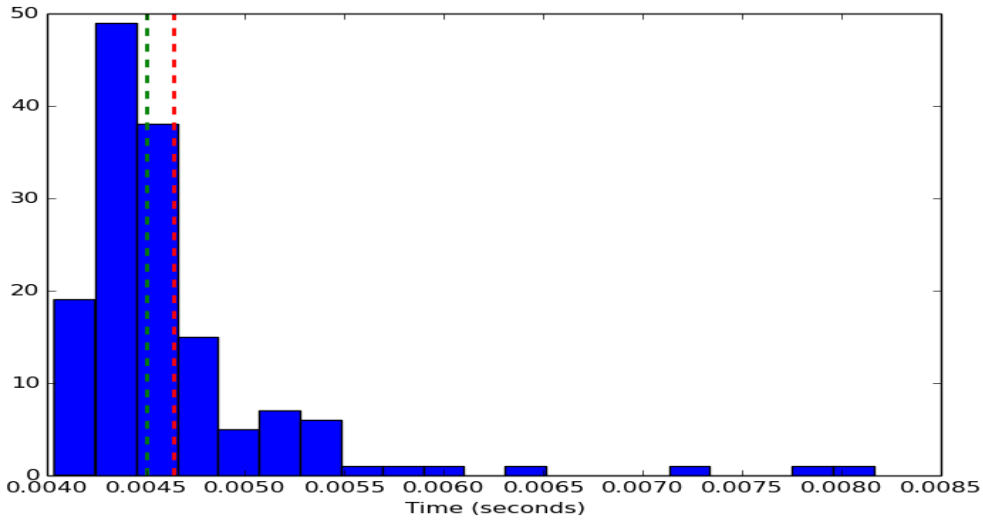

Figure 4.2. The distribution of the time used for querying a malicious logo in the legitimate data set.

in separate experiments, added them several degrees of noise, scaling and rotation. Next, we discuss each of these experiments.

4.3.1 Effect of noise on the false negative rate

For this experiment we added from 10 up to 80 percent, with a step of 10 percent, of salt and pepper noise (figure 4.3) to 32 images and then compared them with the original images. when the HOG similarity was less that 0.83, the number of false negatives was incremented by one. Figure 4.4 illustrates the effect of noise on the false negative rate.

4.3.2 Effects of scaling on the false negative rate

In a similar experiment, we scaled 32 logos with a scaling factor of 0.9 to 0.1 (scaling factor of 0.1 means the size of the logo scaled to 0.1 of the original size). The figure 4.5 shows a logo scaled in this scaling factor range. The scaled logos

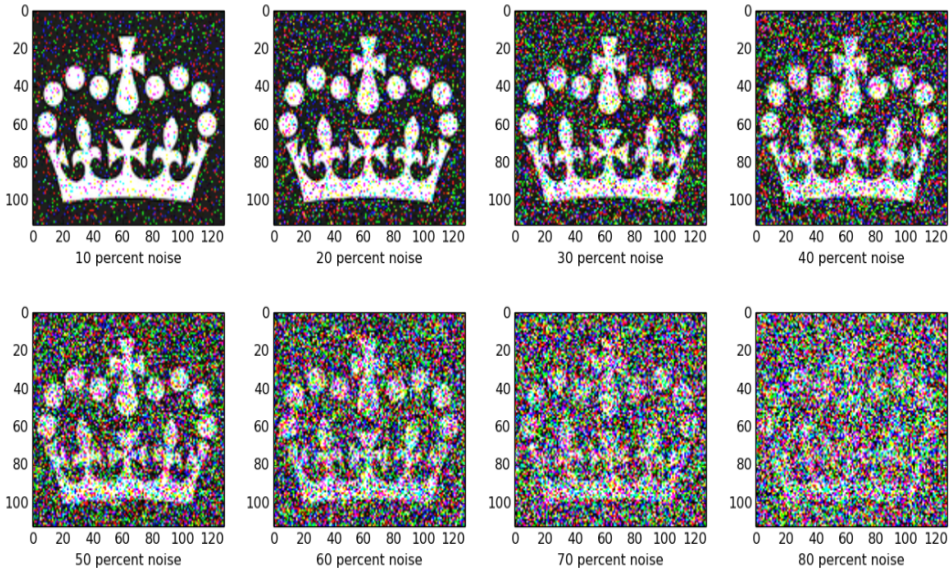

Figure 4.3. Adding noise to the logos.

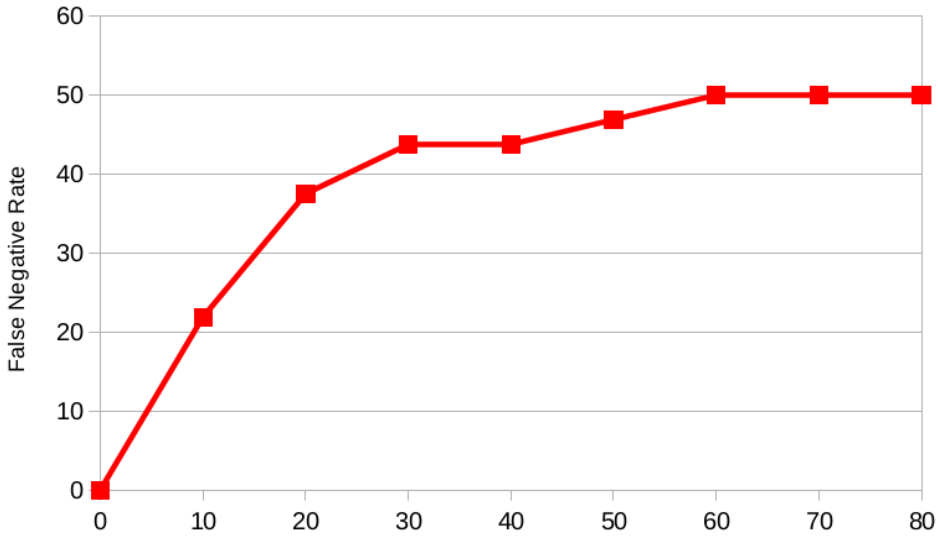

Figure 4.4. Effect of noise on the false negative rate.

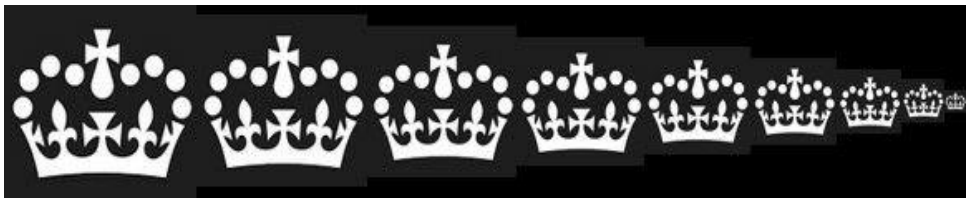

Figure 4.5. Scaling the logos, scaling factor 0.9 to 0.1.

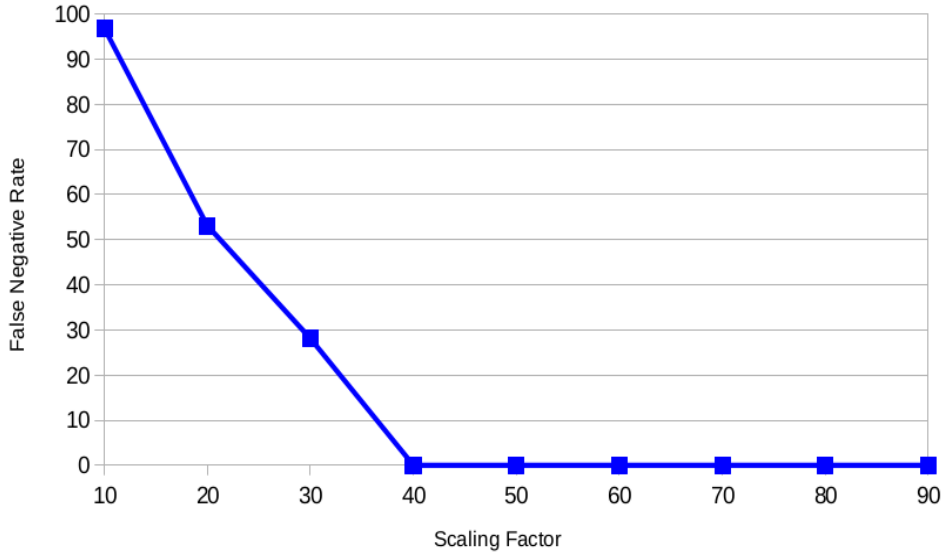

Figure 4.6. Effects of scaling on the false negative rate.

are compared with the original logos and if the HOG similarity was less than 0.83, the number of false negatives was incremented by one. The figure 4.6 illustrates the results of this experiment.

4.3.3 Effects of rotation on the false negative rate

Finally, we have rotated 32 logos from 15 to 180, with a step of 15 degrees, (figure 4.7) degrees and compared them with the original logos. If the HOG similarity was less than 0.83 the number of false negatives was incremented by one. The figure 4.8 shows the effect of rotation on the false negative rate.

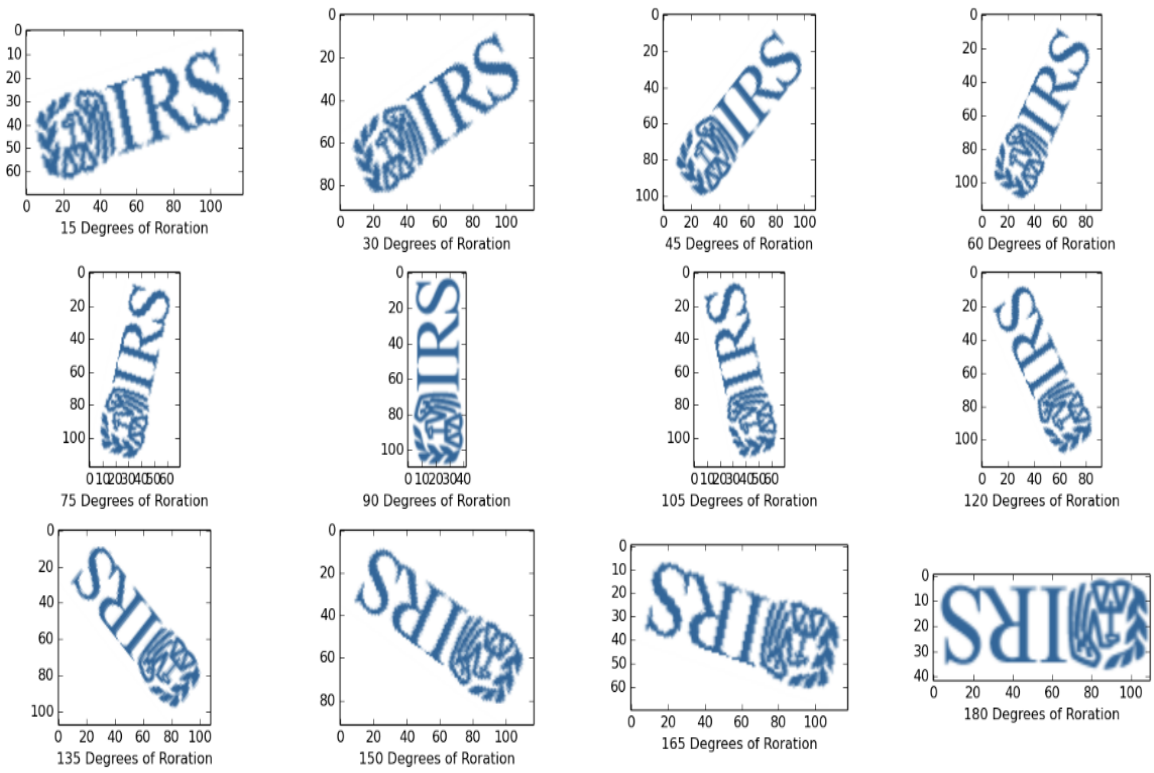

Figure 4.7. Adding rotation to the logos.

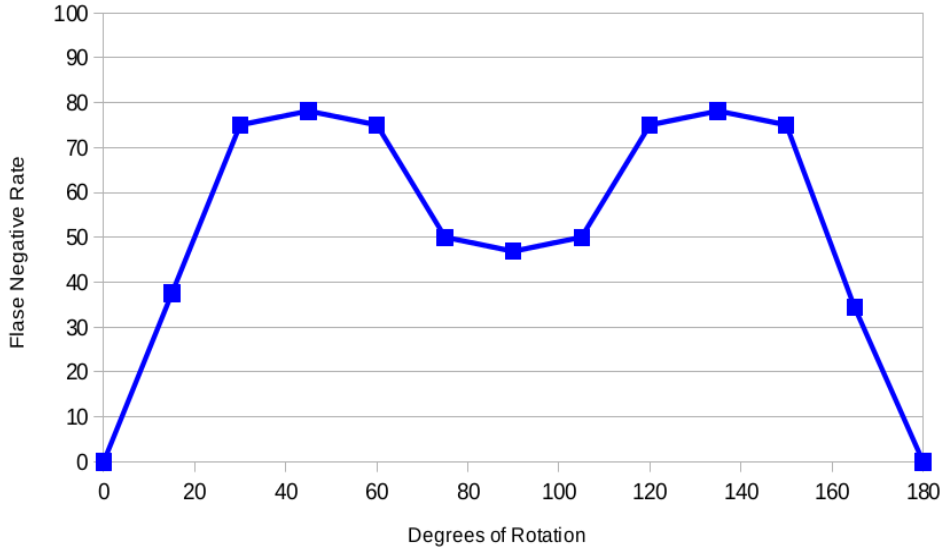

Figure 4.8. Effect of rotation on the false negative rate.

4.3.4 Discussion on the results of this set of experiments

A quick look on the figure 4.8 is enough to notice that the similarity measure based on the logos' histograms of the gradients is sensitive to rotation. A 15 degrees rotation is enough to generate about %40 of false negatives, and that is because rotation can dramatically change the histogram of gradients of an image. Increasing the false negative rate is against the second objective of the Shallow Detector. In order to mitigate this issue, we added the intensity histograms to the similarity measure. In this case, two logos will be considered as not similar if the HOG similarity is less than 0.83 *and* the similarity of their intensity histograms is less than 0.9. We have repeated the previous experiments with the new similarity measure. This can filter the effect of the rotation completely. The results are illustrated in the figure 4.9.

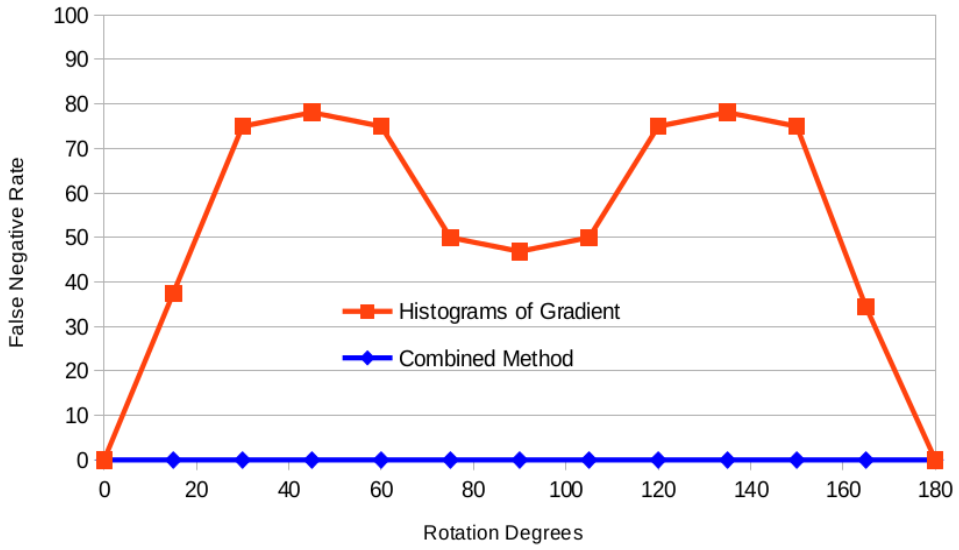

Figure 4.9. Effect of rotation on the combined similarity measure.

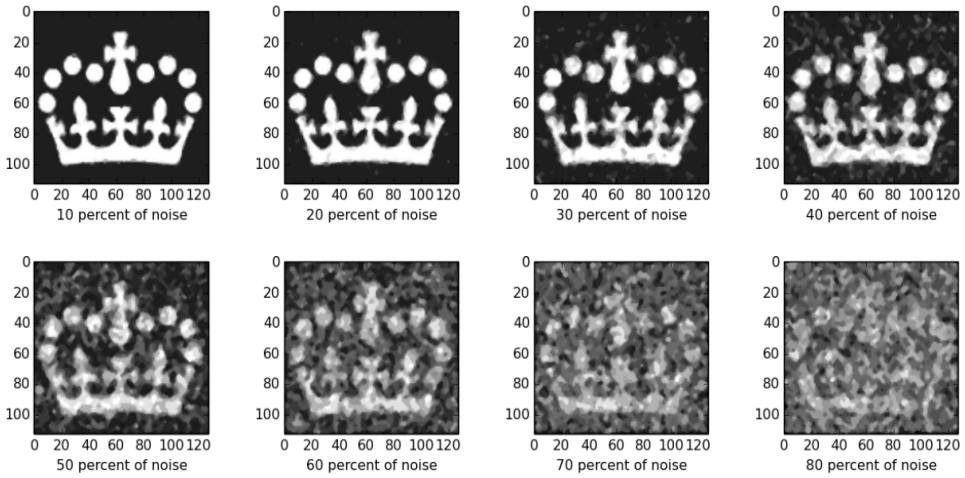

Figure 4.10. Noise removal using median filtering.

Adding 10 percent of noise have also caused more than 20 percent false negatives. We used a noise removal technique in order to mitigate the effects of the noise on the false negative rate. Several methods have been proposed for noise removal. We have used a method called *median filtering* in which each pixel will be replaced by the median intensity value of its neighbor pixels in a three by three pixels block. Figure 4.10 illustrates the results of median filtering on the previous example mentioned in the figure 4.3. We repeated the experiment after applying the noise removal process. Figure 4.11 shows the improvement of false negative rate after the noise removal. The noise removal process had an average time overhead of two milliseconds.

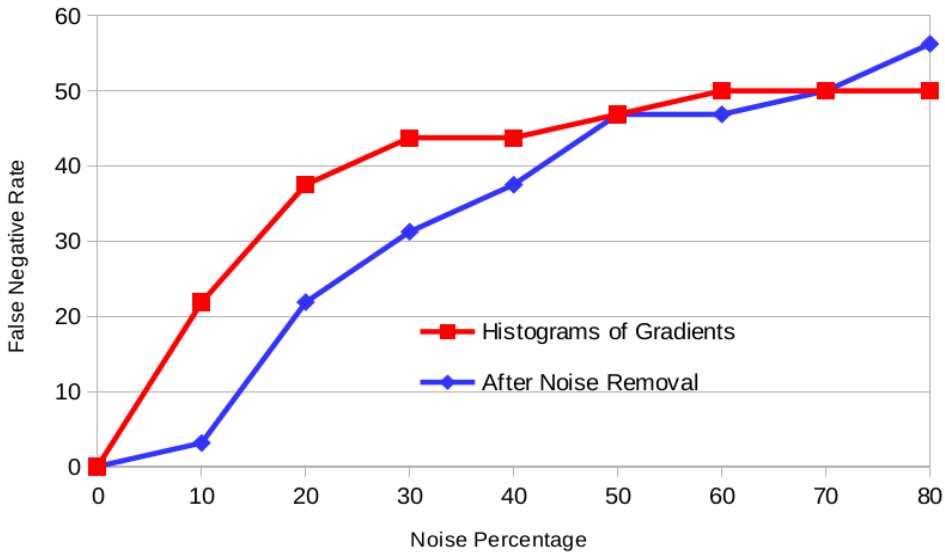

Figure 4.11. False negative rate after noise removal.

CHAPTER 5

DISCUSSION AND CONCLUSION

5.1 Discussion

In this section, we discuss the overall issues regarding the proposed framework and the results of the experiments.

One of the most important assumptions that we have done in our work was that the phishing web pages visually imitate the target web pages. During the analysis of the phishing pages, in order to make our data sets, we noticed few pages that asked for the credentials of a specific brand without having any visual signal or similarity to the brand they were attacking. The method we proposed for the Shallow Detector cannot distinguish the phishing pages that has no visual imitations. However, the chance that the users be victims of such traps can be considered low since the users also don't have any source of trust and cannot see any visual similarity to do their unconscious usual similarity validation, and consequently, they will be suspected of such an attack. Another important thing that should be considered is that during our experiments for the Shallow Detector, we assumed that we have a perfect logo detector. In fact, as discussed in the section 4.1 the logo data sets are selected and cropped by the human. Therefore, the performance of the simple object detector that has proposed in the section 3.2.1 can affect the overall results directly. Thus, a need to a relatively fast and accurate logo detector can be seen in the project. One can raise the question of why these data sets are used and how we can know that it is going to work well on the other data sets. Unfortunately, there is no standard data set of visual information about the phishing attacks and the most reliable source of

such information is the PhishTank, which we have extracted our data randomly from it. Finally, users may be worried about the privacy issues regarding the detector itself since it is going to take a screen shot of any page that they are visiting. This can be mitigated by doing the majority of the processing part on the client-side.

5.2 Conclusion

In this work, we have proposed a real-time visual similarity based approach for the website phishing detection. Although the existing visual similarity based solutions are not appropriate for the real-time purposes, the framework that we have proposed tries to be fast enough for the online scenarios and also to be accurate and reliable at the same time. The proposed framework has two important phase: Shallow Detection and Deep Detection phases. The goal of the Shallow Detector is to confidently, without false negatives, and quickly rule out the bulk of the pages that are completely different from the database of the legitimate websites. The more similar pages are passed to the Deep Detector, which is pickier and more accurate, for more inspections. Since the Shallow Detector has an important job in the overall framework, in this work, we have discussed its practical aspects in details. We exploited the fact that the malicious websites usually imitate the visual signals and especially the logos of the actual web pages. In our experiments, we queried 132 phishing logos in a database of 168 legitimate logos from 65 brands. For the purpose of similarity measurement we used Histograms of Gradients as the logo representation and compared the logos' histograms using the *histogram intersection* approach. The results for the Shallow Detector shows zero false negative and %47 false positive rates.The mean and the median query time was less than 5 milliseconds. It means that the Shallow Detector has achieved its goal by ruling out %53 percent of the test cases without any false negatives and quick enough for the real-time purposes.

CHAPTER 6

FUTURE WORKS

Up to now our focus was on the Shallow Detector because of its importance in the overall performance of the framework. The goal of the project is to make a web browser plugin that can protect users against the phishing attacks in a real-time manner. However, before accomplishing such a goal, we should improve the logo detector in the Shallow Detection phase since as discussed in the discussion section (5.1), the accuracy of the object detector can affect the overall performance of the framework. At the same time, we are going to apply and test several more sophisticated detectors as the elements of the Deep Detection phase in order to decrease the false positive rate. When all of these have been done, we should test the whole framework as a black box to see how it actually perform in the real world scenarios.

APPENDIX A

The List of Most Targeted Brands

Table A.1. The List of Most Targeted Brands

PAYPAL	BANCO DO BRASIL	IMPOTS
EBAY	BANCO ESTADO	MICROSOFT LIMIA
IRS	BANCO FALABELLA	MAY BANK
AOL	BANCO VOTORANTIM	MICROSOFT
APPLE	BARCLAY	MSN
FACEBOOK	BING	NAB
GOOGLE	BANK OF AMERICA	NAT WEST
BRADESCO	BRADESCO	OFFICE
JPMORGAN CHASE	CAIXA	OFFICE365
YAHOO	CAIXA BANK	ONE DRIVE
WELLS FARGO	CANADA REVENUE	OUTLOOK
AZUL	CAPITAL ONE	POPFAX
ITAU	CENTURY 21	RESONA
WALMART	CHASE	SAFRA
CIELO	CIELO	SICREDI
NATWEST BANK	CIMB	SINIORPEOPLE MEET
AMAZON.COM	CITI	SKYPE
BARCLAYS BANK PLC	CRAIGLIST	SMBC
BANCO DE BRASIL	DIGICEL	STANDARD BANK
FNB	DOTZ	MICROSOFT SURFACE
ADOBE	DROPBOX	SWED BANK
LCL	EDF	TAM
MSN	FACEBOOK SECURITY	TESCO BANK
ABSA BANK	FIRST NATIONAL BANK	USSA
ALIBABA	GMAIL	WALMART
ALLEGRO CADENCE	GOOGLE DOCS	WELLS FARGO
AMERICAN EXPRESS	GOOGLE DRIVE	WEST PAC
ANZ	GOOGLE SHEETS	WINDOWS10
APPLE MUSIC	GOOGLE SLIDES	WINDOWS LIVE
APPLE ID	GOV.UK	XBOX
ASSURANCE MALADIE	HOTMAIL	
HSBC	YAHOO	

REFERENCES

[1] E. Kirda and C. Kruegel, "Protecting users against phishing attacks," *The Computer Journal*, vol. 49, no. 5, pp. 554–561, 2006.

[2] C. Whittaker, B. Ryner, and M. Nazif, "Large-scale automatic classification of phishing pages." in *NDSS*, vol. 10, 2010.

[3] T. Kitten, "FBI Alert: Business Email Scam Losses Exceed $1.2 Billion," http://www.bankinfosecurity.com/fbi-a-8506/op-1, 2015.

[4] G. Aaron and R. Rasmussen, "Global Phishing Survey : Trends and Domain Name Use in 2H2014," *APWG*, no. May, pp. 1–38, 2014. [Online]. Available: http://docs.apwg.org/reports/APWG_Global_Phishing_Report_2H_2014.pdf

[5] "Phishtank status," http://www.phishtank.com/stats.php, accessed: 2016-03-12.

[6] "Scamwatch status," http://scamwatch.gov.au/types-of-scams/attempts-to-gain-your-personal-information/phishing, accessed: 2016-03-12.

[7] I. Kirlappos and M. A. Sasse, "Security education against phishing: A modest proposal for a major rethink," *IEEE Security & Privacy*, no. 2, pp. 24–32, 2011.

[8] S. A. Robila and J. W. Ragucci, "Don't be a phish: steps in user education," in *ACM SIGCSE Bulletin*, vol. 38, no. 3. ACM, 2006, pp. 237–241.

[9] P. Kumaraguru, S. Sheng, A. Acquisti, L. F. Cranor, and J. Hong, "Teaching johnny not to fall for phish," *ACM Transactions on Internet Technology (TOIT)*, vol. 10, no. 2, p. 7, 2010.

[10] S. Sheng, B. Magnien, P. Kumaraguru, A. Acquisti, L. F. Cranor, J. Hong, and E. Nunge, "Anti-phishing phil: the design and evaluation of a game that teaches

people not to fall for phish," in *Proceedings of the 3rd symposium on Usable privacy and security.* ACM, 2007, pp. 88–99.

[11] R. Dhamija, J. D. Tygar, and M. Hearst, "Why phishing works," in *Proceedings of the SIGCHI conference on Human Factors in computing systems.* ACM, 2006, pp. 581–590.

[12] M. Alsharnouby, F. Alaca, and S. Chiasson, "Why phishing still works: user strategies for combating phishing attacks," *International Journal of Human-Computer Studies*, vol. 82, pp. 69–82, 2015.

[13] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Identifying suspicious urls: an application of large-scale online learning," in *Proceedings of the 26th annual international conference on machine learning.* ACM, 2009, pp. 681–688.

[14] S. Bin, W. Qiaoyan, and L. Xiaoying, "A dns based anti-phishing approach," in *Networks Security Wireless Communications and Trusted Computing (NSWCTC), 2010 Second International Conference on*, vol. 2. IEEE, 2010, pp. 262–265.

[15] N. Chou, R. Ledesma, Y. Teraguchi, J. C. Mitchell, *et al.*, "Client-side defense against web-based identity theft." in *NDSS*, 2004.

[16] Y. Zhang, J. I. Hong, and L. F. Cranor, "Cantina: a content-based approach to detecting phishing web sites," in *Proceedings of the 16th international conference on World Wide Web.* ACM, 2007, pp. 639–648.

[17] T.-C. Chen, S. Dick, and J. Miller, "Detecting visually similar web pages: Application to phishing detection," *ACM Transactions on Internet Technology (TOIT)*, vol. 10, no. 2, p. 5, 2010.

[18] M.-E. Maurer and D. Herzner, "Using visual website similarity for phishing detection and reporting," in *CHI'12 Extended Abstracts on Human Factors in Computing Systems.* ACM, 2012, pp. 1625–1630.

[19] K.-T. Chen, J.-Y. Chen, C.-R. Huang, and J.-Y. Chen, "Fighting phishing with discriminative keypoint features," *Internet Computing, IEEE*, vol. 13, no. 3, pp. 56–63, 2009.

[20] S. Afroz and R. Greenstadt, "Phishzoo: An automated web phishing detection approach based on profiling and fuzzy matching," in *Proceedings of the Semantic Computing (ICSC), 2011 Fifth IEEE International Conference*, 2009.

[21] "Netcraft toolbar," http://toolbar.netcraft.com, accessed: 2016-03-12.

[22] P. Prakash, M. Kumar, R. R. Kompella, and M. Gupta, "Phishnet: predictive blacklisting to detect phishing attacks," in *INFOCOM, 2010 Proceedings IEEE*. IEEE, 2010, pp. 1–5.

[23] Y. Cao, W. Han, and Y. Le, "Anti-phishing based on automated individual white-list," in *Proceedings of the 4th ACM workshop on Digital identity management*. ACM, 2008, pp. 51–60.

[24] Y. Wang, R. Agrawal, and B.-Y. Choi, "Light weight anti-phishing with user whitelisting in a web browser," in *Region 5 Conference, 2008 IEEE*. IEEE, 2008, pp. 1–4.

[25] A. Y. Fu, L. Wenyin, and X. Deng, "Detecting phishing web pages with visual similarity assessment based on earth mover's distance (emd)," *Dependable and Secure Computing, IEEE Transactions on*, vol. 3, no. 4, pp. 301–311, 2006.

[26] L. Wenyin, G. Huang, L. Xiaoyue, X. Deng, and Z. Min, "Phishing web page detection," in *Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on*. IEEE, 2005, pp. 560–564.

[27] L. Wenyin, G. Huang, L. Xiaoyue, Z. Min, and X. Deng, "Detection of phishing webpages based on visual similarity," in *Special interest tracks and posters of the 14th international conference on World Wide Web*. ACM, 2005, pp. 1060–1061.

[28] W. Liu, X. Deng, G. Huang, and A. Y. Fu, "An antiphishing strategy based on visual similarity assessment," *IEEE Internet Computing*, vol. 10, no. 2, p. 58, 2006.

[29] K. Mikolajczyk and C. Schmid, "Indexing based on scale invariant interest points," in *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, vol. 1. IEEE, 2001, pp. 525–531.

[30] D. G. Lowe, "Object recognition from local scale-invariant features," in *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, vol. 2. Ieee, 1999, pp. 1150–1157.

[31] A. K. Jain and A. Vailaya, "Image retrieval using color and shape," *Pattern recognition*, vol. 29, no. 8, pp. 1233–1244, 1996.

[32] D. M. Powers, "Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation," 2011.

[33] J. Canny, "A computational approach to edge detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, no. 6, pp. 679–698, 1986.

[34] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 886–893.

[35] R. Islam and J. Abawajy, "A multi-tier phishing detection and filtering approach," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 324–335, 2013.

[36] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1. IEEE, 2001, pp. I–511.

BIOGRAPHICAL STATEMENT

Omid Asudeh was born in Kashmar, Iran, in 1990. He received his B.S. degree from Qom University of Technology, Iran, in 2013 and his M.S. degree from The University of Texas at Arlington in 2016 both in Computer Science. He is going to continue his PhD in the same major at Ohio State University.