A new method of comparing webpages

Hao Jiang, CS657

Fall, 2013

Abstract

Webpage comparison compare the similarity of two webpages. It can be useful in areas such as distinguishing phishing website and making personal recommendation. Most of the previous work on webpage comparison focus on visual comparsion using image processing technique, which is not good at extracting information from the text in the webpage. Moreover, visual comparison cannot tell the content correlation between the webpages. Our method takes both the visual similarity and text keywords into account, generating a more reliable result.

1 Introduction

Website similarity indicate how similar two webpages look from the user perspective. It includes two parts: visual similarity and content similarity. Visual similarity basically means whether the two websites looks the same. This includes the color, layout and image used, etc. Content similarity indicates whether the two webpages talk about the same subject.

The comparison of webpage similarity can be very useful in automatic detection of phishing websites, which tries to resemble original websites, in order to cheat user of their secret. Phishing websites are easy to setup and thus big in amount. An effective and accurate webpage comparison result allows automatic phishing websites to be detected and identified as fast as possible.

Webpage comparison result can also be used to cluster webpages, which is valuable in personalized browsing recommendations. Recommendation may be more accurate if they are similar to the websites the target users visit frequently.

In this paper, we develop a novel method that quantify the similarty of webpages. In other words, we "computes" a number that indicate the difference of webpages. The smaller the difference is, the more similar two webpages are. Our method compare two websites based on both their visual similarity and correlations of text content.

2 Previous Work

Different methods are used to compare the similarity of two webpages. We categorize these methods into two types: structure-based comparison and visual based comparison. We will look at some of the work belongs to these types.

We are also interested in doing clustering of webpages. In this section we also introduce some previous work on this aspect.

2.1 DOM Tree Comparison

Rosiello et al. in [5] talk about a method that compare the similarity of the webpages DOM tree. They will compare the tags contained in the tree, and tries to extract the regularity from sub-tree structures. They also introduce penalties to nodes that are different on the DOM tree, to avoid possible interference introduced by the attacker.

This method is easy to implement and efficient. It is also effective against phishing websites that directly copy the content of original websites. But the disadvantage is also obvious. First, the appearence of a webpage cannot be uniquely defined by its DOM tree structure. Thus an attacker could easily avoid such detection by using different HTML tags to generate webpages that look similar. In addition, comparing DOM tree helps little about extracting the correlation of text content of webpages. The difference between the DOM trees of two webpages also tells us little about the clus-

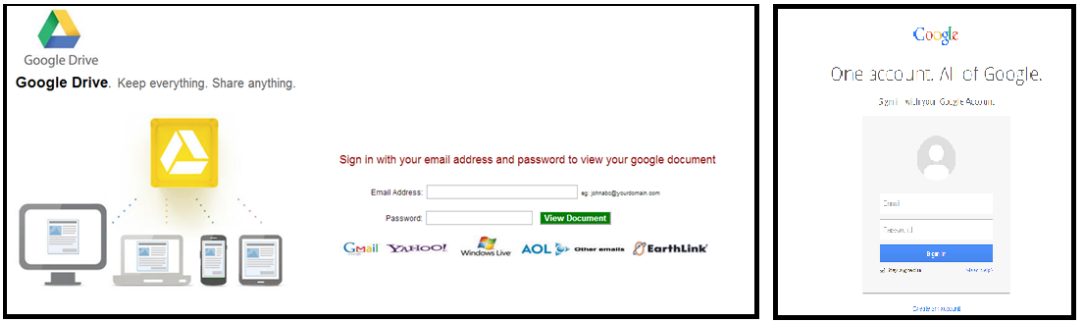

Figure 1: A phishing website of Google Drive

tering.

2.2 Visual Similarity

Another type of comparison is based on the visual similarity of two webpages. More specifically, the screenshots of webpages are captured using techniques such as headless browsers. Comparison between these images are then conducted using various image processing techniques.

Visual comparison methods overcome some of the disadvantage of DOM-based methods as it focuses on comparing the visual output of the webpage, which is the same scene seen by the end users. This method is especially efficient against phishing websites, which will most of the time looks the same as the original website.

J.White et al. talks about using the hash value of images in [7]. In the paper, the authors also introduce their work of capturing suspcious links from Twitter message, which is also very interesting. But here we only mention about the method they used to compare webpage content. They first capture the screenshot of the webpage using , calculate the MD5 hash of the image file and evaluate the image difference using Hamming Distance between two hash values. With their experiment result, the authors claims that adding a small change to the original image will also lead a small increase to the Hamming Distance.

A.Fu et al. talk about their work in [3] of using Earth's Mover Distance to evaluate the difference of two webpages. The Earth Mover's Distance (EMD) [6] is a measure of the distance between two probability distributions over a region D. If we think an image file as a distribution of different colors, then the EMD between two images is defined to be the minimal amount of work to move the colors in order to make two distribution the same. EMD describes the difference between two images, so it should also be a good candidate of comparing webpage screenshots.

All the work are effecient against a special kind of phishing websites. However, in practice we also found some examples that can avade the detection easily. Figure 1 shows an example phishing webpage of Google Drive we got from phishtank.com, a website that collects phishing website from volunteers. This webpage doesn't look like the actual Google Drive login page, which use Google's Single-Sign-On and is also shown in the same Figure. We cannot omit the possibility that this webpage can still lure some users that are not so familiar with Google's login page. However, as it doesn't look like the actual login page, the methods we mentioned above will fail to capture this phishing website.

2.3 Website Clustering

In [1], Bohunsky et al. describes their work of using the visual structure to do webpage clustering. Instead of doing comparison to the entire picture, they split the webpage into small rectangle areas which they call "visual boxes". By comparing on the visual box structure of a webpage, they tries to tell the correlation of two webpages and do the clustering accordingly.

However, during the process of converting everything into images, the text information contained in the website is lost, which means two web-

Figure 2: Challenge to Visual Similarity

pages that has similar layout but total different text content may be categorized as similar. Figure 2 shows such an example. The images are from two different news websites. They all have a title picture and an abstract. Dispite the visual similarity, the content they talk about may be totally different things. This shows that relying only on visual layout to do clustering may introduce high rate of false positive.

3 System Description

By the analysis to previous work, we found that visual similarity is an effective tool of phishing website detection, and it is immune to attacking efforts that creates a phishing website that is slight different to the original one. To overcome the problem we have mentioned in the previous section, we also introduce the comparison of the webpage's text content. That is to say, besides making sure two webpages look the same, we are also going to make sure they are talking about the same thing. In this section, we will introduce our method for webpage comparison.

3.1 Overview

Our method is based on comparing both the visual feature and the text feature of the webpages. We are trying to figure out the "feature" that best describe a webpage. In other words, we are looking for parts that make this webpage different from others.

We will make use of a headless browser library to capture both a screenshot of the target webpage, and the text included in it. With the screenshot image, we use a partitioning algorithm to detect the visual layout of the entire page. We also try

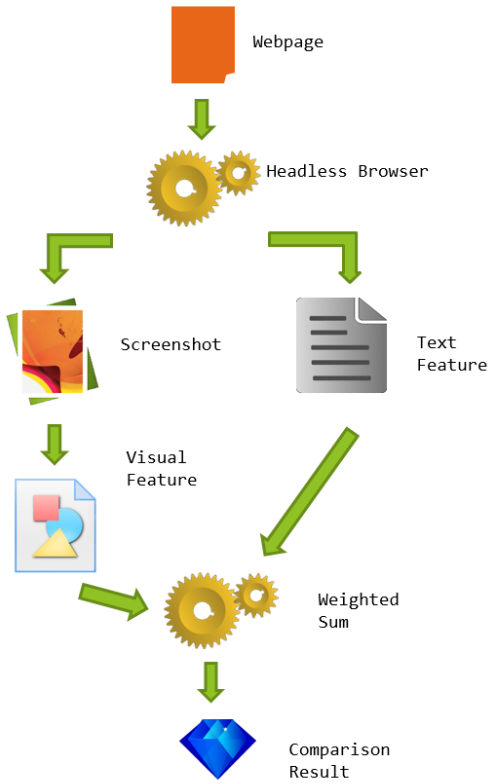

Figure 3: System Process

to find the piece of image that is "unique" to this webpage, which we hope can be used to identify it. The text information we extract from the page will also include the CSS style of the text, which helps us describe the type of text

With the feature extracted from original webpage, we can do comparisons between them. We then calculate a weighted sum of the comparison result. This is the comparison result.

Doing clustering of webpages is fairly easy using the comparison result. We can define a threshold value for different scenarios, then use traditional clustering algorithms. We have chosen some algorithms as candidate and done a comparison between them.

Figure 3 shows how the system works. We will describe the detail of each step in the following sections.

3.2 Visual feature extraction

The visual feature extraction contains two parts of work: detecting visual layout and extract featural pieces. We examine the visual layout info

```
function EXTRACT(DOMNode* current)
    Tree ← EmptyTree
    for all BlockNode bn in current do
     Tree append EXTRACT(bn)
    end for
    return Tree
end function
```

Figure 4: Visual Layout Algorithm

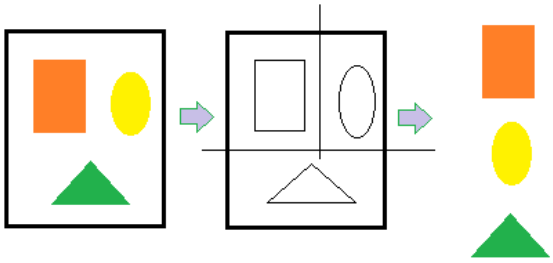

Figure 5: Visual Feature Extraction

from the DOM tree, and associate the corresponding screen area to it. More specifically, we start from the <BODY>element, and execute the algorithm shown in Figure 4.

To get the featural pieces, we first collect all <IMG>tags contained in the html page. For each image file retrieved from the tag, we then try to split it into small pieces separated by backgound color. This is done via two steps. First, we process the image using a simple Edge Detection algorithm, which will change the background color into black, and leaves only the edge of the object. We then try to find a wide enough black strip that separate the image into two parts. This process is done both horizontally and vertically, and is repeated until no further split can be done. The split result is output as separate images. Each of these small image is taken as a fetural piece. Figure 5 shows the process.

3.3 Visual feature comparison

We now need do comparison to the visual features extracted in the previous step. For the layout information, we compare them based on the overlap of two layouts. The more overlap there is, the higher the similarity will be. Assume (U_i, U_j) are layout block pairs from the two layout structure

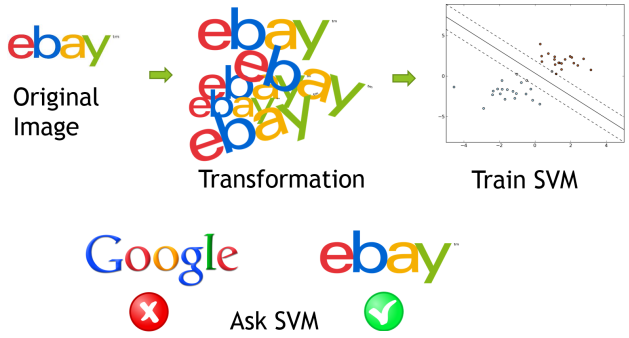

Figure 6: SVM Classification

to be compared, $Overlap(U_i, U_j)$ is the overlapped area. The similarity from visual layout is calculated as following:

$$S_{U_i,U_j} = \frac{1}{2} \sum_{U_i,U_j} \|Overlap(U_i, U_j)\|^2 \left(\frac{1}{\|U_i\|} + \frac{1}{\|U_j\|}\right)$$

We now need to deal with the featural pieces extracted from the previous step. Given two pieces, we first want to know whether they are visually the same. For example, a phishing website to eBay may hold an eBay logo that is of the different size. This is easy for human to tell that they are the same logo with different size. But image comparison technique we mentioned about in the previous section cannot tell that easily. We make use of Support Vector Machine to deal with this problem, as is shown in Figure 6. For a given image, we apply transformations such as scaling and displacement to it to generate a training set. We didn't do rotation because no rotation to original image had been found in our observation to the phishing websites. The training set is then used to train a binary classification SVM. We then use the SVM to tell whether the target image piece is similar to the original one or not. If SVM says that the two images are not similar, we need to calculate how different they are. We use the EMD between these two images directly as the difference between them. Pele proposed a variation of EMD in [4], which is called FastEMD. In our method we make use of this algorithm instead of the traditional EMD due to time efficiency. The result is then normalized to a number between [0, 1].

Finally we need to calculate the visual similarity of the two webpages. This is done by doing weighted sum of each smaller pieces. We use the

the overlapped area's relative size of each piece pair as the weight of that piece.

$$S = \sum_{U_i, U_j} \frac{\|Overlap(U_i, U_j)\|}{PageSize} S_{U_i, U_j}$$

3.4 Text Feature Extraction and Comparison

We now turn to dealing with the text information contained in the webpage. Intutively, we can first retrieve all text from the webpage, then extract keywords from it. The keyword that appears most frequently can be thought of carrying the information to be conveyed by the text.

In addition to this idea, we also notice a fact that text with different styles like size, color, weight, etc., should have different weight. Take the phishing website in Figure 1 as example, the word "Google Drive" appear only once, but it is displayed in a bold font-weight, which means it is more important than other words. Also, we notice that the words "Sign in with your email address and password" is displayed in a different color, which means it can be noticed by the end user much easier, which means it should be assigned more weight.

We take all these thoughts into account and develop a algorithm for extracting keywords from html pages. With the headless browser, we are not only able to get the text itself, but also the CSS style it is associated with. The following information are gathered together with the text itself: font-size, font-color and font-weight. We also record the type of nodes that contains the text.

For each of the text segments gathered from the webpage, we extract keywords from each of these text segments using Rapid-Automatic-Keyword-Extraction algorithm (RAKE,[2]). RAKE first split the text into words with punctuation stop and common words such as "the" and "and", then count the frequency of the words and sorts the importance of keywords by both its frequency in the text and the length of the word group it stays.

The weight of these keywords are then calculated based on the style information collected above. We do a normalization to the distribution of font text size. The bigger the font is, the higher weight is it assigned. Bold font-weight is thought to be equivalent to increasing the font-size by a factor α. When we calculate the weight of different font-color, we notice the following fact: the darker the color is, the more obvious it is to the observer. But the darkest color – the black is too common to be assign a high weight. We define the colors that have equal values in red, green and blue the "trivial" color, that is, different gradient of grays. These colors have the lowest weight of 0. In 3-D space, these colors form a line from (0,0,0) to (255,255,255). The weight of different colors is then defined to be the distance from the point represented by that color to this line. Thus we define a formula of the weight of the color based on the RGB value as following:

$$w = sin(arccos(\frac{u \cdot \mathbf{1}}{\sqrt{3}\|u\|}))\|u\|$$

The comparison between two keyword groups is done by measuring the overlap of them as well as the corresponding weight in both groups.

3.5 Clustering Algorithm

NOT DONE YET

4 Implementation

NOT DONE YET

5 Experiment

NOT DONE YET

6 Conclusion

NOT DONE YET

7 Future Work

NOT DONE YET

References

[1] Paul Bohunsky and Wolfgang Gatterbauer. Visual structure-based web page clustering and retrieval. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 1067–1068, New York, NY, USA, 2010. ACM.

[2] W.E. Cowley, N.O. Cramer, V.L. Crow, and S.J. Rose. Rapid automatic keyword extraction for information retrieval and analysis, March 6 2012. US Patent 8,131,735.

[3] Anthony Y. Fu, Liu Wenyin, and Xiaotie Deng. Detecting phishing web pages with visual similarity assessment based on earth mover's distance (emd). *IEEE Transactions on Dependable and Secure Computing*, 3(4):301–311, 2006.

[4] Ofir Pele and Michael Werman. Fast and robust earth mover's distances. In *ICCV*, 2009.

[5] Angelo P E Rosiello, E. Kirda, C. Kruegel, and F. Ferrandi. A layout-similarity-based approach for detecting phishing pages. In *Security and Privacy in Communications Networks and the Workshops, 2007. SecureComm 2007. Third International Conference on*, pages 454–463, 2007.

[6] Y. Rubner, C. Tomasi, and L.J. Guibas. A metric for distributions with applications to image databases. In *Computer Vision, 1998. Sixth International Conference on*, pages 59–66, 1998.

[7] J. S. White, J. N. Matthews, and J. L. Stacy. A method for the automated detection phishing websites through both site characteristics and image analysis. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 8408 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, May 2012.