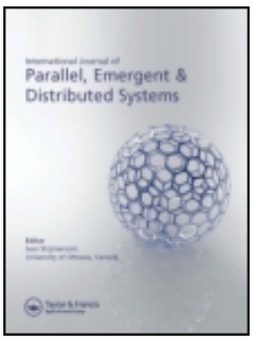

International Journal of Parallel, Emergent and Distributed Systems

ISSN: 1744-5760 (Print) 1744-5779 (Online) Journal homepage: http://www.tandfonline.com/loi/gpaa20

A segmentation method for web page analysis using shrinking and dividing

Jiuxin Cao , Bo Mao & Junzhou Luo

To cite this article: Jiuxin Cao , Bo Mao & Junzhou Luo (2010) A segmentation method for web page analysis using shrinking and dividing, International Journal of Parallel, Emergent and Distributed Systems, 25:2, 93-104, DOI: 10.1080/17445760802429585

To link to this article: http://dx.doi.org/10.1080/17445760802429585

Published online: 19 Mar 2010.

International Journal of Parallel, Emergent and Distributed Systems
Vol. 25, No. 2, April 2010, 93–104

Taylor & Francis
Taylor & Francis Group

A segmentation method for web page analysis using shrinking and dividing

Jiuxin Cao[a,b,c]*, Bo Mao[a,b,c] and Junzhou Luo[a,b,c]

[a]*School of Computer Science and Engineering, Southeast University, Nanjing 210096, China;* [b]*Jiangsu Provincial Key Laboratory of Network and Information Security, Southeast University, Nanjing 210096, China;* [c]*Key Laboratory of Computer Network and Information Integration, Ministry of Education, Southeast University, Nanjing 210096, China*

(*Received 1 August 2008; final version received 25 August 2008*)

On the basis of image processing technology and characteristics of web pages, a new web segmentation method – iterated shrinking and dividing is proposed in this paper. Dividing conditions and concept of dividing zone are introduced, based on which web page image is divided into visually consentaneous sub-images by shrinking and splitting iteratively. First, the web page is saved as image that is preprocessed by edge detection algorithm such as Canny. Then dividing zones are detected and the web image is segmented repeatedly until all blocks are indivisible. This method can be used to analyse the web pages such as detecting similar visual layout. Experiments show that the algorithm is suitable for web page segmentation, and does well in expansibility and performance.

Keywords: web analysis; dividing zone; shrinking

1. Introduction

As the development of internet, the web related applications have become one of the most significant applications of networks. Therefore, the acquisition, detection and analysis on web contents are paid more and more attentions, and web page segmentation algorithm is an important aspect of them. Precisely speaking, we use the segmentation method for phishing page detection.

The existing algorithms of web segmentation for phishing page recognition are mainly based on web source code (HTML) including template-based analysis, DOM tree analysis, tab list and so forth. However, those analyses have great limitations especially in visual similarity assessment, which is crucial for phishing page detection, because of HTML flexibility and browser interdependency. Compared with HTML, it is more effective to directly deal with the web pages shown to the user by image processing which works better in certain cases like image searching, phishing web detecting [12,13,15].

Although there are various researches on image segmentation, the web page has its own characteristics e.g. composed by blocks and complex images. Then it is the key for a segmentation algorithm to make use of those characteristics, which is exactly what we do here. The whole phishing page detection method contains two parts. One is the web page segmentation algorithm that we will show in this paper, and the other is web-matching algorithm based on Nested-EMD. Experiments show that our scheme has better

*Corresponding author. Email: jx.cao@seu.edu.cn

ISSN 1744-5760 print/ISSN 1744-5779 online

DOI: 10.1080/17445760802429585
http://www.informaworld.com

performance than other proposed image analysis methods. First, the suspicious web pages that may come from spam email testing are saved as images and sent to phishing detecting centre for further analysing. Then, the detecting centre segments the coming page image into blocks by the method proposed here. Finally, the features of the blocks that contain size, colour information and the relations between blocks are extracted. Those features compose an attribute relation graph (ARG) of the web page image. We detect the similarity of the suspicious web page with those protected ones like pages of bank and ISP by the nested earth mover's distance (NEMD) of their ARG's. If the NEMD is smaller than set value, the suspicious web pages are marked by phishing page. And the key for the detection is the segmentation method that can determine the accuracy of ARG generation. We will focus on the segmentation method here.

This paper is organised as follows. Related works are presented in Section 2. Then a new web segmentation algorithm is proposed in Section 3 and the results of the experiment are given in Sections 4 and 5. Finally, Section 6 concludes the whole paper.

2. Related works

In the field of the web analysis, a lot of jobs have been done. Bar Yossef [1] proposed a method to improve the search engine by detecting template in the web page; Lin and Ho [11] divided the web page into several blocks according to the table labels. However, both of the methods only directed at certain web station and were not portable. Cai [2,3] proposed a vision-based page segmentation which approached the problem from a different angle – visual characteristic, and made a full use of characteristics such as the size and style of types, background colours, and blank areas, meanwhile laid down the rules to divided page into blocks, which could serve the complex pages well. But the drawn artificially rules are always indistinct, so it is difficult to keep the consistency of these rules. Some others focus on the DOM trees. Gupta [8] masked off ads by counting the number of linked/non-linked words, it needs the manpower to adjust the parameter for the best results and also cannot deal with images. Finn [6] treated the HTML document as the list composed of characters and labels, and extract the words from the character set. However it is not suitable for the complex web pages but only those in the character set. Kovacevic [10] divided the web pages into head, foot, right, middle, and left areas, and its defect is the low adaptation. Feng [5] proposed a framework of web page analysis with coordinate trees, which can divide the pages by space relationships and their locations, but it also relied on the rules and cannot deal with the frame labels as well as the pages with CSS.

Meanwhile, to some extent the web page segmentation is similar with document segmentation in which a lot of efforts have been made. Those can be divided into two types – top–down and bottom–up. The top–down is faster but needs some prior knowledge, and the representation is project profile cutting (PPC)[14], while the bottom–up does not need the prior knowledge but slower. Besides, the segmentation of document is based on its textures [9], which is independent of the text area shape but more complex and time consuming.

The algorithm proposed in this paper satisfies the special needs of web page segmentation. Because of the web page features such as composed by blocks, divided by background, the segmentation based on saddle point in PPC cannot deal with the complex web page well. Therefore we propose a new conception – dividing zone that can be defined by prior knowledge (the features of web) meanwhile the shrinking technology is employed for better result.

3. Iterated dividing and shrinking

Based on web pages features, our algorithm is composed of three steps – web to image, image pre-processing and dividing.

3.1 Features of Web page

By analysing the visual characteristics of web page, it is easy to find out that there are following two general features.

(a) Web page is composed by blocks;
Every visible element in HTML is displayed in its rectangle area. Labels, images, tables even flashes are all possessed of the basic parameters: width and length. So with a rectangle area, we can locate any visible element in web pages.
(b) The visible elements of web page are separated by background space;
If there is no space between two visible elements, they may be considered as one by people, which dose not handicap our understanding from visual aspect.

Based on those two features, we can divide the web page into sub-images (basic blocks) according to the dividing zones which are the spaces between blocks.

3.2 Web page pre-processing

First, web pages are converted into images, and then we get the binary images from detecting the edges of these web images.

The dominant browsers such as IE and Firefox provide the APIs to save web page as image. Also there are several commercial programs like ViewPage (http://viewpage.maxthon.com/) Htmltojpg that can do the same job. In this paper, we make use of the system library function (Microsoft.mshtml.dll) to fulfil convert from html to image. This system library is used to parse the HTML, and is employed by IE, Outlook.

The initial web images are always complex and usually in RGB format, so we will simplify them by boundary detecting. Canny [4] is employed for its validity. Compared with Roberts, Prewitt and Sobel, Canny algorithm enjoys the virtues of higher positioning precision and better signal to noise ratio. From the results of experiments (see Section 5), we can find that the Canny algorithm is much more suit for web page image pre-processing. The result of the pre-processing is a binary image with 0 for background and 1 for boundary.

3.3 Dividing and shrinking

The coordinate of two-dimensional image was shown in Figure 1, and let $f(x,y)$ donate the input binary image. The related definitions go as follows.

DEFINITION 1. SUB-IMAGE. The sub-image $g = \{\text{initP}, \text{size}, f(x,y)\}$, in which $\text{initP} = (x_0, y_0)$, is the initial point, $\text{size} = \{w, h\}$ represents the width and height, $f(x, y)$ is the original image of g, and $x \in [x_0, x_0 + w), y \in [y_0, y_0 + h)$, as shown in Figure 1; especially the $f(x, y)$ itself can be represented as $g_0 = \{\text{initP}_0, \text{size}_0, f(x,y)\}$, in which $\text{initP}_0 = (0, 0)$, $\text{size}_0 = \{\text{width}, \text{height}\}$.

The sub-images are used to locate the blocks in web page. Based on features of web page we have summarised in 3.1, any element in a web page can be constricted in a rectangle area that is represented by a sub-image.

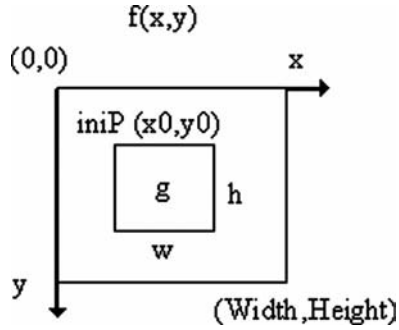

Figure 1. The coordinates of the sub-image.

DEFINITION 2. REAL SUB-IMAGE. The real sub-image g' is the minimal rect-zone which contains the all non-zero pixels of a sub-image g. Following conditions need to be satisfied.

1. $\forall(x,y)((x \in ([x_0, x_0') \bigcup (x_0' + w', x_0 + w)) \wedge \quad y \in [y_0, y_0 + h)) \rightarrow (f(x,y) = 0))$

2. $\exists s(s \in [y_0, y_0 + h) \rightarrow f(x_0', s) = 1)$
   $\exists s(s \in [y_0, y_0 + h) \rightarrow f(x_0' + w', s) = 1)$

3. $\forall(x,y)((y \in ([y_0, y_0') \bigcup (y_0' + h', y_0 + h)) \wedge x \in [x_0, x_0 + w)) \rightarrow (f(x,y) = 0))$

4. $\exists v(v \in [x_0, x_0 + w) \rightarrow f(v, y_0') = 1)$
   $\exists v(v \in [x_0, x_0 + w) \rightarrow f(v, y_0' + h') = 1)$

in which, g' is the real sub-image of g, $g' = \{(x_0', y_0'), \{w', h'\}, f(x, y)\}$, $g = \{(x_0, y_0), \{w,h\}, f(x, y)\}$, $x_0 < x_0'$, $y_0 < y_0'$, $x_0 + w > x_0' + w'$ and $y_0 + h > y_0' + h'$.

Because there is certain space formed by blank between elements in web page, we have to eliminate the space to get the valid part of the block that is represented by real sub-image.

DEFINITION 3. SHRINKING. The process getting the real sub-image from its sub-image is called shrinking, expressed as $g' = \text{shrink}(g)$. In which g is the sub-image and g' is the real sub-image.

In our implementation, the shrinking not only gets real sub-image from sub-image, but also deal with the rectangle boundaries. Because in many web pages elements have boundaries that could prevent the further dividing to the areas inside the boundaries, the shrink function has to do something to get into those boundaries. Besides detecting real sub-image, the shrink function identifies the boundaries in the edges of the sub-image and gets rid of them. Since we are only care about the valid parts of the web page, it is not necessary to record those broken boundaries.

DEFINITION 4. DIVIDING ZONE. Dividing zone $d = \{g, \text{attr}\}$, in which g is the sub-image that the dividing zone belongs to, and attr is the attribute of the zone. The attr can be adjusted according to the specific applications, and decides how to divide the sub-image. The definition of attr is given in Definition 5.

Dividing zone represents the blank area between valid blocks of the web page. Every dividing zone is related and belongs to a sub-image in which it exists. In another word, the dividing zone is a rectangle blank area, which runs through certain sub-image and could split this sub-image into two. The sub-image may have more than one dividing zones, so it is a problem that how to select the best one. The answer lays in the attributes of dividing zone.

DEFINITION 5. ATTRIBUTES OF DIVIDING ZONE. The dividing zone attribute attr = {hv, gd, r}, in which hv $\in \{0, 1\}$ represents the zone is horizontal or vertical, gd is the sub-image that the zone occupies, and $r \in [0, 1]$ is the shrinking rate given in Definition 6. Especially gd has to meet some conditions called dividing conditions, for example gd must be the background.

These attributes are about the information of dividing zone selection. First, the zone has the feature of transfixion. Therefore, there are two kinds of zones – horizontal and vertical. If the sub-image under segmentation is too slim, then its horizontal dividing zones are priority choice. Second, the dividing zone occupies certain area (gd), and the bigger one is priority. Third, we introduce the shrink rate to do farther selection, because different blocks of the web page have the different width. In the whole selection, we compare the attributes of each zone, and choice the best one as the dividing zone of the sub-image.

DEFINITION 6. SHRINKING RATE. Let g' donate the real sub-image of g, and the area of g' to be s. When g was divided into $g1$ and $g2$ by $d = \{g', \text{attr}\}$, let the area of $g1'$ and $g2'$ the real sub-images of $g1$ and $g2$ to be $s1$ and $s2$, then the shrinking rate of attr equals to $1 - (s1 + s2)/s$.

Based on the definitions above, the algorithm is described as follows.

Start:
SG = {g0}, EG = {} //Initially, SG the set of the sub-images contains only one element – the original image, EG is the set of the end sub-image that cannot be divide any more.
Loop:
If SG = = Φ // quit condition
    Exit;
Else
    g = get(SG); //get a sub-image from SG
    SG = SG − g;
End
g′ = shrink(g);
If isDivisible(g′) = = FALSE //g′ cannot be divided any more
    EG = EG + g′; //put it into the EG
Else
    d = findD(g′); //find out the best dividing zone of g′
    g1, g2 = divide(d); //divide g′ into g1 and g2
    SG = SG + {g1, g2}; //and send them to the SG
End
Goto Loop;
End Start.

The keys of this algorithm are the separability judgment function – isDivisible(), and dividing zone searching function – findD(), which both relate to the dividing zone attribute mentioned in Definition 5. In this paper, the dividing conditions are simple. First is transfixion. The dividing zone should have the same width or height with the sub-image, in other words it has to intersect the sub-image. Second is consistency. The dividing zone should be composed by the same pixels or almost the same, which means that the zone has to be the background.

In order to find out those dividing zones, we first locate the dividing line which is the line in sub-image and satisfies dividing conditions (transfixion and consistency). For the better robustness, it is allowed that there are different pixels other than background in the dividing line, and the number and distribution of those pixels are calculated. If that number is bigger than a set value (2% of total pixels of the line in our implementation), the line is eliminated from set of dividing line. Then the nearby dividing lines with same distribution of non-background pixels are merged and become the candidate dividing zone.

When there are several candidate zones in a sub-image, we compare them by size (the bigger the better), the shrinking rate (the higher the better), *gd* (boundary first), and *hv* value (horizontal first), then select the best one as the dividing zone.

By the dividing conditions, we can judge whether a sub-image *g* would be divided any more by following judgments.

1. If the area of *g* is smaller than a given value, then the sub-image is not divisible.
2. If there is no dividing zone detected, then the sub-image is not divisible.

Detecting the dividing zone can be divided into horizontal and vertical searching. Take the horizontal searching for example (the vertical is similar), and let the sub-image to be *g*.

```
Start:
If the height of g less than a certain value
      return null.
Else
   scan every row of g
   If the row is background (over 98% pixels are 0s)
      mark the row as dividing line,
   End
End
merge the interfacing background dividing lines to be candidate zone;
find the best horizontal dividing zone from candidate zones;
return the best zone.
End Start.
```

We can use the similar method to find the best vertical dividing zone, then compare them and get the best one as the dividing zone of *g*.

The time complexity of this algorithm is related to dividing conditions. Assume the original image was divided into n parts then the expected scan times is O(logn), because a part is divided into two each time. When the dividing zone searching complexity equals to O (A) in which A is the area of the image equals to width $\times$ heigth, the whole complexity of the algorithm is O(Alogn).

Meanwhile, it is noted that the half of the dividing zones could be reused by the dividing of next around. More specifically, if the sub-image *g* is divided by a horizontal

zone into $g1$ and $g2$, then its vertical dividing zones are also the dividing zones of $g1$ or $g2$. With this advantage, we can save half time of the dividing zone detection.

4. Pre-processing results

We compare the results of different edge detection algorithms like Sobel, Roberts, Prewitt and Canny, which are showed in Figure 2. It is clear that the Canny is much more suitable for our application.

From the results we can find that Canny method could deal with the low contrast grade areas such as yellow part on white background as shown in Figure 2. Also the details are better in Canny than other edge detection methods. Therefore Canny is selected as our pre-processing method.

5. Experiment results

We implement the algorithm in Matlab, and employ some library functions such as canny. More than 20 web pages are tested. Figure 3 gives some of the results.

The platform of the experiment is a PC with PIV CUP, 512M RAM, and XP OS. We set the stopping area to be 100 that means if the area of sub-image is less than 100, then the sub-image is not divisible. Finally program costs about 3 s for the images above. Time cost can be reduced if candidate-dividing zone is reused in the process. Compared with the web

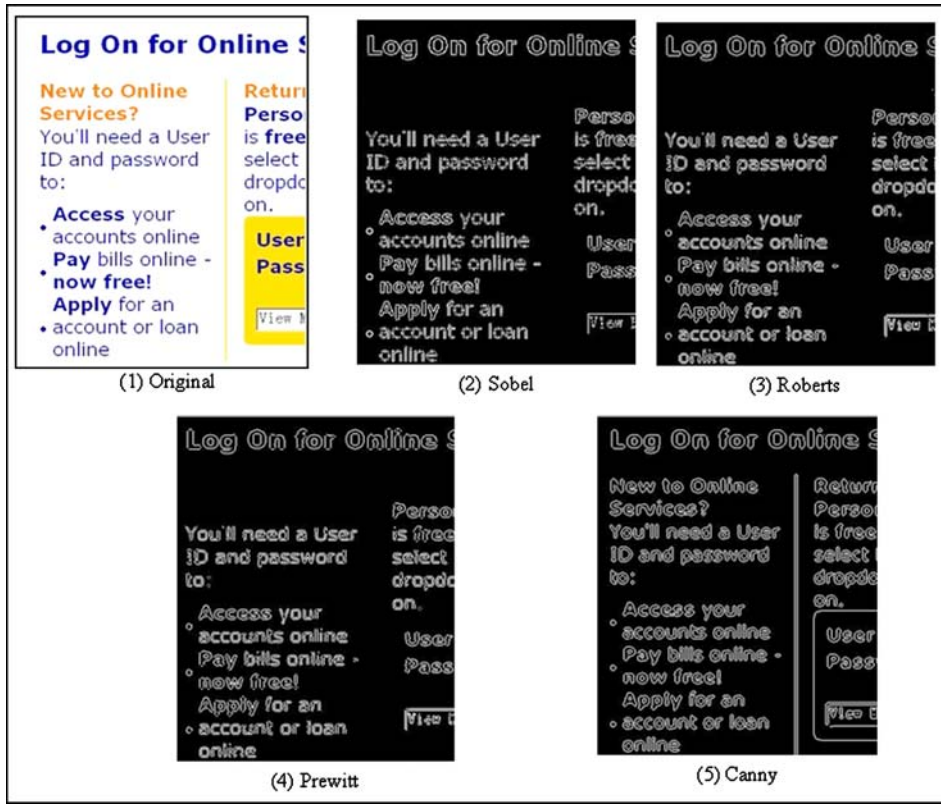

Figure 2. Results of pre-processing.

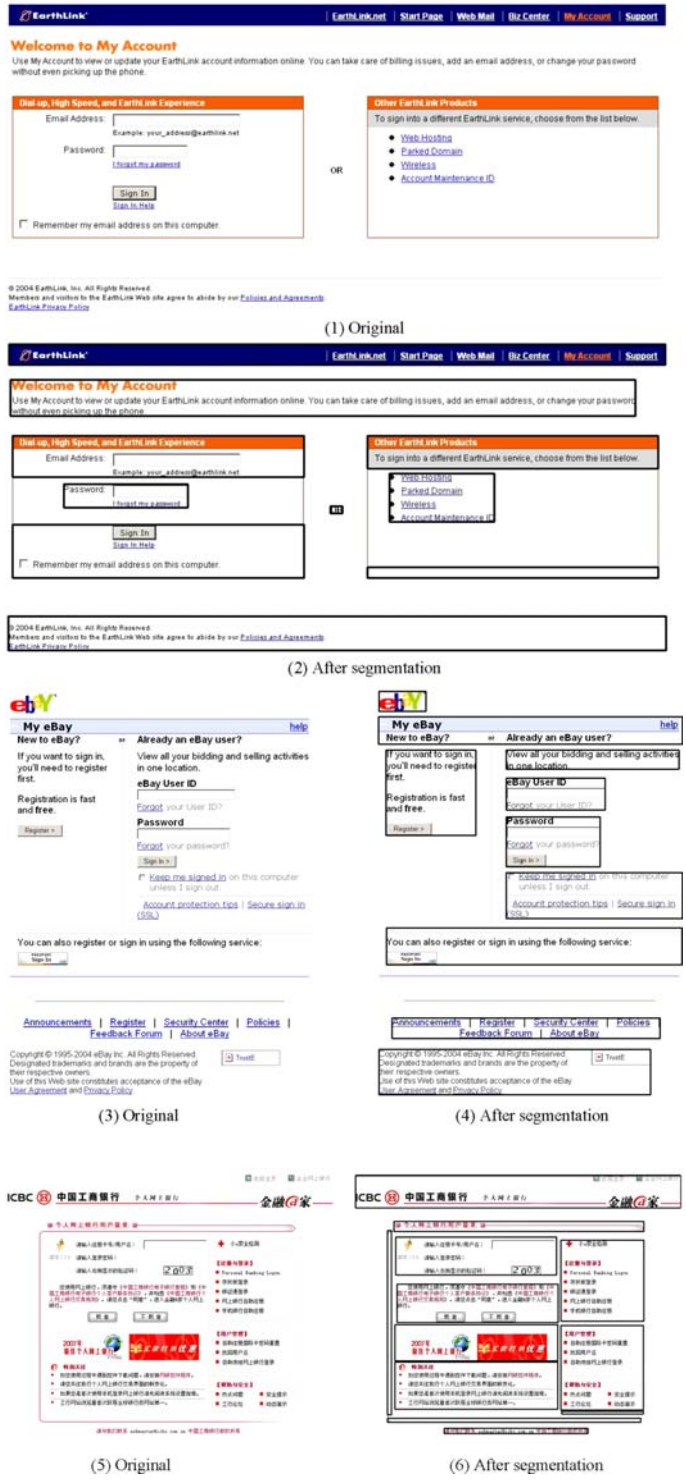

Figure 3. Segmentation results. (1) Original (2) after segmentation (3) original (4) after segmentation (5) original (6) after segmentation.

page transport time, this delay is acceptable. Meanwhile, the whole analysis of phishing page is implemented in the server (phishing analysis center) and is off-line, so the common users will not be influenced. We implement our segmentation algorithm in a real page analysis system to detect web phishing, the experiments show that our method not only can segment the web page precisely in visual aspect, but also satisfy the real time requirement for the phishing detection.

We can see that the algorithm is very accurate for the web page segmentation. If a web page complies with the two features we mentioned in 3.1 and the layout of its blocks is common, our method could deal well with it. Most of the web pages satisfy our requirement, while there are certain indivisible situations. In Figure 4, an example of special layout is given. In this case, the two features of 3.1 are satisfied, but there is no dividing zone in it (no rectangle area of background runs though the web page). Therefore, this web page cannot be divided by our method further more, and this is what we should work on next. However, it is rarely to confront with web pages share the layout as shown in Figure 4, so our method could work effectively in most cases.

To test the efficiency of the page segmentation method, we use it in our phishing page detection approach, and compare that with Fu's [7] approach by Matlab. Fu's method also makes use of image process and EMD, but without segmentation. The test phishing web page data is supplied by Liu [12] in his homepage. Among Liu's data, two phishing web pages aimed at eBay and one at EarthLink, ICBC, Wells Fargo, US Bank, and Washington Mutual Bank, respectively. The correspondent six true target web pages are also collected for comparison. In the remaining part of this section, we denote a true web page by adding the prefix 't-', e.g., t-eB stands for the true web page of eBay, and a phishing web page by adding the prefix 'f-', e.g., 'f-IC' refers to the Phishing webpage targeting at ICBC.

Tables 1 and 2 show the results of Yu's EMD and our approach among real and phishing web pages, respectively. It is shown that both algorithms work well except for EarthLink, because the phishing web page of Earthlink is not similar to the real one. However, the results also show ours is much more robust than Yu's approach.

We compare the ratio of phishing and non-phishing similarity of two algorithms. Let Sim(web1, web2) is the similarity distance between web1 and web2. The smaller Sim(web1, web2) is, the more similar they are. The ratio of $\mathrm{Sim}(t - \mathrm{web}_i, f - \mathrm{web}_i)$ and $\mathrm{Sim}(t - \mathrm{web}_i, \mathrm{f} - \mathrm{web}_j)$ and $i \neq j$ can reflect the accuracy of the algorithm, since there are several phishing pages we make use of the worst and average ratio. Formula (1) shows the worst distance ratio Riworst, and Formula (2) shows the average distance ratio R_i^{avg}, in

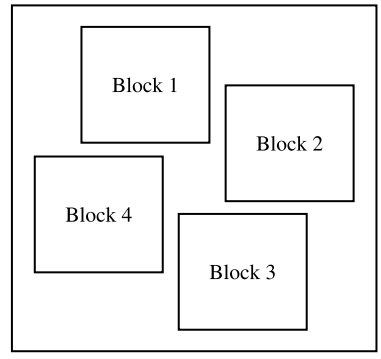

Figure 4. Special indivisible layout.

Table 1. Yu's EMDs among real and phishing web pages.

	t-eB	*t-EL*	*t-IB*	*t-WF*	*t-USB*	*t-Wt*
f-eB1	**0.0041**	**0.0292**	0.065	0.0432	0.0196	0.0256
f-eB2	**0.0048**	0.0294	0.0643	0.0434	0.0203	0.0249
f-EL	0.0187	**0.0293**	0.0609	0.0561	0.0248	0.0143
f-IC	0.0591	0.0633	**0.003**	0.0664	0.0566	0.0589
f-WF	0.0424	0.0571	0.0672	**0.0121**	0.0419	0.0559
f-USB	0.0172	**0.0240**	0.0596	0.0413	**0.0017**	0.0228
f-Wt	0.0293	**0.0231**	0.0597	0.0614	0.0299	**0.0095**

Table 2. Nested EMD among real and phishing web pages.

	t-eB	*t-EL*	*t-IB*	*t-WF*	*t-USB*	*t-Wt*
f-eB1	**0.0151**	0.2044	0.3483	0.1472	0.3458	0.2383
f-eB2	**0.0032**	0.2051	0.3232	0.1452	0.3395	0.2405
f-EL	0.1985	**0.1989**	0.4257	0.0820	0.3490	0.2449
f-IC	0.3219	0.4168	**0.0010**	0.4599	0.2155	0.4210
f-WF	0.1414	**0.1343**	0.4516	**0.0135**	0.2706	0.1685
f-USB	0.3370	0.3393	0.2153	0.2720	**0.0052**	0.3354
f-Wt	0.2470	0.2642	0.4280	0.1777	0.3387	**0.0125**

which $f - \text{web}_i^k$ is the kth phishing page aimed web_i, and $i \neq j$ and $\text{web}_i \neq$ Earthlink. Figure 5 shows the worst distances ratio of the two algorithms, and Figure 6 shows the average distance ratio. We can find our scheme is better than Fu's EMD in both accuracy and robustness of similarity detections.

$$R_i^{\text{worst}} = \frac{\max\left(\text{Sim}\left(t - \text{web}_i, f - \text{web}_j\right)\right)}{\min\left(\text{Sim}\left(t - \text{web}_i, f - \text{web}_i^k\right)\right)} \tag{1}$$

$$R_i^{\text{avg}} = \frac{\text{avg}(\text{Sim}(t - \text{web}_i, f - \text{web}_j))}{\text{avg}(\text{Sim}(t - \text{web}_i, f - \text{web}_i^k))} \tag{2}$$

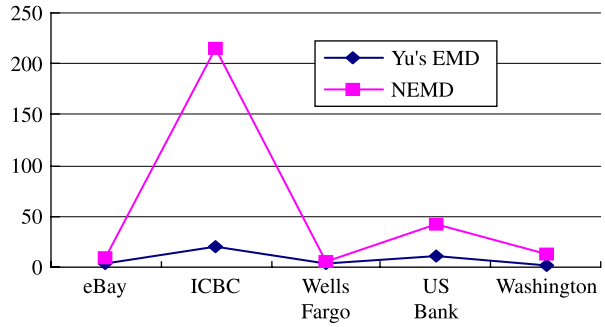

Figure 5. Worst distances ratio.

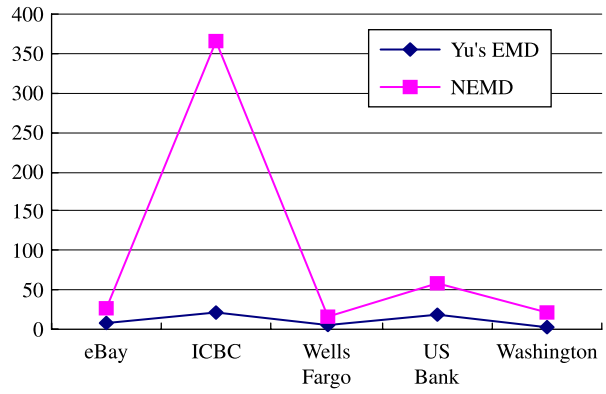

Figure 6. Average distances ratio.

6. Conclusions

In this paper, a new web page segmentation algorithm based on iterated shrinking and dividing is proposed for the better web analysis. The algorithm can be used in phishing web detection, visual similar web page retrieve and so on. The experiments show that our method can divide web pages into the parts similar in vision quickly and precisely meanwhile independent from the HTML code.

In the future, more attention should be paid into the definition of the dividing conditions, and image pre-processing for the better result.

Acknowledgements

This work was supported in partly by National Technologies Research and Development Program during the 10th Five-Year Plan Period of China under Grants No. 2005BA115A01, National Science and Technology Support Project under Grants No. 2006BAH02A24.

The authors also would like to thank the anonymous reviewers for their thorough reading and valuable comments.

References

[1] Z. Bar-Yossef and S. Rajagopalan, *Template detection via data mining and its applications*, in *Proceedings of the 11th International Conference on World Wide Web (WWW)*, 2002.

[2] D. Cai, X. He, W.-Y. Ma, J.-R. Wen, and H. Zhung, *Organizing www images based on the analysis of page layout and web link structure*, in *IEEE International Conference on Multimedia and Expo (ICME)*, 2004.

[3] D. Cai, S. Yu, J. Wen, and W.-Y. Ma, *VIPS: A vision-based page segmentation algorithm*, Tech. Rep. MSR-TR-2003-79, (2003).

[4] J. Canny, *A computational approach to edge detection*, IEEE Trans. Pattern Anal. Mach. Intell. 8(6) (1986), pp. 679–698.

[5] H.M. Feng, B. Liu, Y.M. Liu, Y. Fang, and G. Song, *Framework of web page analysis and content extraction with coordinate trees*, J Tsinghua Univ (Sci & Tech) 45(S1) (2005), pp. 1767–1771.

[6] A. Finn, N. Kushmerick, and B. Smyth, *Fact or fiction: content classification for digital librarie*, in *Joint DELO-NSF Workshop on Personalisation and Recommender Systems in Digital Libraries*, Dublin, (2001).

[7] A.Y. Fu, L. Wenyin, and X. Deng, *Detecting phishing web pages with visual similarity assessment based on earth mover's distance (EMD)*, IEEE Trans. Dependable Security Comput. 3(4) (2006), pp. 301–311.

[8] S. Gupta, G. Kaiser, D. Neistadt, and P. Grimm, *DOM based content extraction of HTML documents*, in *Proceedings of 12th world wide web conference (WWW)*, Budapest, Hungary, (2003).
[9] A.K. Jain and S. Bhattacharjee, *Text segmentation using Gabor filters for automatic document processing*, Mach. Vision Appl. 5(3) (1992), pp. 169–184.
[10] V.N. Keith and F. Carsten, *Applying gestalt principles to animated visualizations of network data*, in *Proceedings of the Sixth International Conference on Information Visualisation (IV'02)*, IEEE Computer Society Press, 2002.
[11] S-H. Lin and J.-M. Ho, *Discovering information content blocks from web documents*, in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (SIGKDD'02)*, (2002).
[12] W. Liu, H. Guanglin, X. Liu, M. Zhang, and X. Deng, *Phishing web page detection*, in *Proceedings of Eighth International Conference on Documents Analysis and Recognition*, (2005).
[13] W. Liu, X. Deng, G. Huang, and A.Y. Fu, *An anti-phishing strategy based on visual similarity assessment*, IEEE Internet Comput. 10(2) (2006), pp. 58–65.
[14] G. Nagy, S. Seth, and S.D. Stoddard, *Document analysis with an expert system*, in Pattern Recognition Practice, Elsevier Science Publishers, North-Holland, 1986.
[15] L. Wenyin, G. Huang, L. Xiaoyue, Z. Min, and X. Deng, *Detection of phishing web pages based on visual similarity*, in *Proceedings of the 14th International World Wide Web Conference*, (2005).