IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 13, Issue 1 (Jul. - Aug. 2013), PP 67-72
www.iosrjournals.org

Captcha Recognition and Robustness Measurement using Image Processing Techniques

Ramya T[1], Jayasree M[2]
[1]*(Computer Science Department, Government Engineering College, Thrissur, India)*
[2]*(Assistant Professor in Computer Science Department, Government Engineering College, Thrissur, India)*

Abstract : *The advances in web-based technology have revolutionized the way people communicate and share information, necessitating firm security measures. Network security prevents and monitors unauthorized access, misuse, modification, or denial of a computer network and network-accessible resources. A CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) is a standard security mechanism for addressing undesirable or malicious Internet bot programs. CAPTCHA generates and grades tests that are human solvable, but beyond the capabilities of current computer programs. Captcha prevents quality degradation of a system by automated software, protect systems that are vulnerable to e- mail spam and minimizes automated posting to blogs, forums and wikis. This paper carries out a systematic study of various Text-based Captchas and proposes the application of Forepart based prediction and Character-Adaptive Masking to break these captchas to evaluate their robustness. Captcha segmentation and recognition is based on Forepart prediction, necessity sufficiency matching and Character-adaptive masking. Different classes of captchas like simple captchas, Botdetect captchas & Google captchas are taken into consideration.*
Keywords : *Captcha, Robustness, Segmentation, Character-adaptive masking.*

I. INTRODUCTION

The proliferation of the publicly available services on the Web is a boon for the community at large. But unfortunately it has invited new and novel abuses. Programs (bots and spiders) are being created to steal services and to conduct fraudulent transactions. Such bots may be intended to

1. Make automatic registrations in service accounts like email/social network/cloud/etc and are being used to distribute stolen or copyrighted material.
2. Mimic legitimate clients to change rank of websites popularity.
3. Broadcast junk emails, post advertisements, or ask server to respond at avery high frequency.
4. Bogus comments on blogs/forums/chats.
5. Online polls are attacked by bots and are susceptible to ballot stuffing. This gives unfair mileage to those who benefit from it.

All these forms of misuses will decrease the usefulness of internet services. To prevent such abuses, it is very important to design an automatic system to differentiate between the messages of legitimate human users and non-legitimate computer bots. The Captcha was created to address these needs.

The term "CAPTCHA" was coined by Luis Von Ahn, Manuel Blum, Nicholas J. Hopper (all of Carnegie Mellon University, and John Langford (then of IBM) in 2000. A typical CAPTCHA user interface consists of two parts: a character image with noise, and an input textbox. The CAPTCHA system then asks the user to enter the string of characters that appear in a distorted form on the screen. CAPTCHAs are used because of the fact that it is difficult for the computers to extract the text from such a distorted image, whereas it is relatively easy for a human to understand the text hidden behind the distortions. Therefore, the correct response to a CAPTCHA challenge is assumed to come from a human and not an automated computer bot.

While considering the design principles of well-known CAPTCHA systems, one can see that many well-known websites such as MSN, Yahoo, Google, Badongo, Rapid Share and YouTube have been employing user interfaces similar to that of Fig. 1.

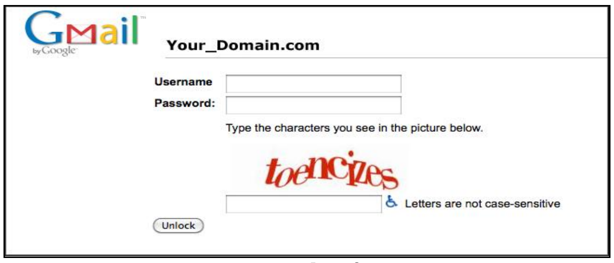

GMail by Google
Your_Domain.com
Username
Password:
Type the characters you see in the picture below.
toencizes
Letters are not case-sensitive
Unlock

Figure 1. An Example of a CAPTCHA.

In light the above listed abuses and much more, a need was for a facility that checks users and allows access to services to only human users. It was in this direction that such a tool like CAPTCHA was created.

The main objective of this paper is to conduct a systematic study of Captchas like Google captcha, Complex captcha and Bot detect Captcha and to measure their robustness against attacks by breaking them using Forepart based prediction and Character Adaptive Masking.

The rest of the paper is organized as follows: Related work about CAPTCHA is given in section 2. Section 3 describes the sequential steps involved in the system design and the characteristics of the three classes of Captchas: Complex Captchas, Google Captchas and Bot Detect Captchas that are considered here. Section 4 gives implementation details of the segmentation algorithm used for breaking CAPTCHA. The testing results are given in section 5. Finally section 6 concludes the paper.

II. RELATED WORK

Nowadays, CAPTCHAs are designed in the toughest possible way that prevents any algorithm to break them but still significant work has been done to break these. In 2003, Mori and Malik [4] proposed a shape matching algorithm to break EZ-Gimpy and Gimpy CAPTCHAs. They achieved a success rate of 92% in case of EZ-Gimpy and 33% in case of Gimpy. CAPTCHAs. Chellapillas' et al [5] attacked a number of early CAPTCHAs using machine learning algorithms, and they achieved 4.89% success on an early version of Google's CAPTCHA (around year 2004). In 2007 Ahmad Salah-El-Ahmad, Jeff.Yan and Mohomad Tayara described different types of captcha & the need of captcha in the real time environment [3]. In 2011, Ahmad S, Jeff Yan and Tayara proposed a novel attack that is applicable to a whole family of text CAPTCHAs that build on top of the popular segmentation-resistant mechanism of "crowding character together" for security [1]. Jiqiang Song, Zuo Li, Michael R. Lyu Shijie Cai discussed about Recognition of Merged Characters in [2]. This method utilizes the information obtained from the forepart of merged characters to predict candidates for the leftmost character, and then applies character-adaptive masking and character recognition to verifying the prediction.

III. SYSTEM DESIGN

3.1 Sequential steps involved in the design

- Pre-processing – A set of standard techniques is applied to prepare each challenge image.
- Character segmentation –Character segmentation is carried out using Forepart based prediction, necessity-sufficiency matching and Character Adaptive Masking.
- Character Recognition – after obtaining single character sub images, they can be recognized using a training set or by inputting to an Optical Character recognition system.

3.2 Preprocessing

In preprocessing step the Captcha image is up sampled initially. Up sampling enlarges the image, increases its pixel details, and thus smoothes the embedded text. Up sampling is done by resizing the original image to a standard size of 300*100. Next we binarize the resized image. Binarizing process is done via standard thresholding method: all the pixels with a color value above a heuristically predetermined threshold is converted to black & those below it to white.

Thresholding is then done because the background colors are sufficiently darker or lighter than the text characters. Thresholding is the simple act of partitioning pixels into two groups, text and background, by a value such as color or magnitude/brightness. Pixels matching values above or below a threshold are labeled text or background. Thresholding is used to help segment characters and find character pixels.

3.3 Captcha Characteristics

3.3.1 Class 1: Captcha with Complex background

The most efficient technique used to confuse the segmentation is to add random noise to the image. To de-noise captchas many techniques have been proposed over the years. Text characters are drawn in captchas in a variety of ways. Captchas consisting of untransformed text generated from bitmapped fonts placed randomly or statically are easy to identify. Often text pixels are easy to identify because they share a similar set of properties, distinct from the background. This property is used here to distinguish text from background. The algorithm for Class1 consists of preprocessing, character segmentation and character recognition as described above.

3.3.2 Class 2: Bot Detect Captcha

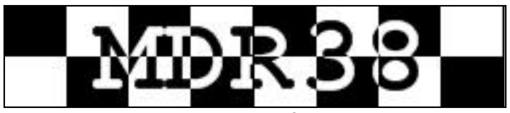


Figure 2. An Example of Bot Detect Captcha

T

he next class is the Bot Detect Captcha. The Bot Detect CAPTCHA is a chess scheme mainly because of its design. This scheme is effectively like embedding characters within a chess board. As each character is divided into a number of components, either black or white, and all the characters are mixed and connected with the chess board, the designers expect that it is hard for automated programs to extract and recognize the embedded characters. Firstly each chess box is detected. If the majority of pixels in a box are black, then reverse all the pixels that are originally black to white, and the pixels that are originally white, if any, to black. If the majority of pixels in the box are white, then do nothing.

3.3.3 Class 3: Google Captcha

Class 3 consists of Google captchas. Google uses only 2 colors to create captchas: red, green or blue for the text & white for the background. Thickness of characters varies from portion to portion. Global Warping is also applied on text. String length varies between 5&8 characters and only lowercase letters are used. Multiple font typefaces & styles (Bold, Italics, regular) are used. Initially preprocessing is done and for segmentation Forepart Based segmentation method [2] is used. Some of the Google captchas are shown in figure 3.

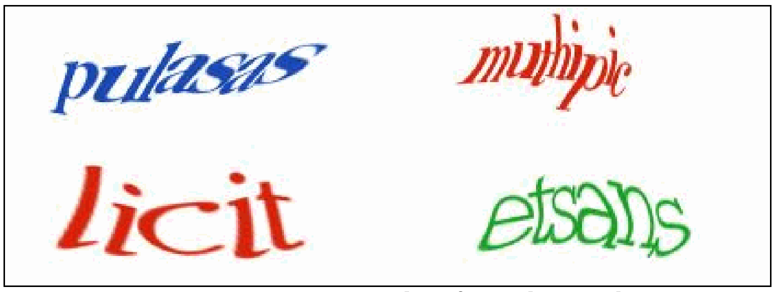


Figure 3. Some examples of Google Captcha

IV. SEGMENTATION ALGORITHM

The segmentation algorithm employed for these three classes of captcha is Forepart based (FP) Segmentation algorithm. The FP-based segmentation algorithm [2] consists of two steps: forepart prediction and character-adaptive masking. The former selects a group of candidates by forepart analysis, while the latter screens out the bitmap using the nonrectangular mask adaptive to each candidate and then recognizes the segmented bitmap by the NSM algorithm. Therefore, the merged characters can be segmented with the highest recognition probability.

4.1 Forepart Prediction and Analysis

After horizontally scanning a document image, one can obtain the baseline position and the height of each text line, which is the vertical distance between the baseline and the top of the text line. Denoting the height of text line by H_l, the term "forepart" means the leftmost $H_l /4$ wide part of the input image of merged characters. For a prototype bitmap, it is nearly the left half of the bitmap. The forepart prediction is based on three reliable forepart features, i.e., baseline-related feature, forepart height feature, and forepart boundary feature. The baseline-related feature indicates whether the forepart of a character has components under the baseline, which takes value "*true*" or "*false*." The forepart height feature indicates whether the forepart occupies the full height above the baseline, which also takes value "*true*" or "*false*." The forepart boundary feature is a little more complex, defined as follows:

$$B = \{B(i) | i \text{ is the row index of a bitmap and } \\ baseline \le i \le baseline + H_l\}$$

B(i) is the number of white points before the first black point in the row . Considering the input scanned bitmap may contain noises, it can be revised to be the number of points before the first black segment whose width is not less than the minimum stroke width. The difference between two forepart boundary features B_1 and B_2 is defined as

$$|B_1 - B_2| = \max_{baseline \le i \le baseline + H_l} [|B_1(i) - B_2(i)|]. \qquad \ldots\ldots.(1)$$

These three features of each character are obtained from its prototype bitmap and then stored in the feature library. Denote the baseline-related feature by L , the forepart height feature by G and the forepart boundary feature by B. Correspondingly, these three features obtained from the forepart of an input bitmap are denoted by *L'*,*G'*,*B'*,respectively. Letting the initial candidate set C_0 containing all models in a feature library, one can select candidates by the following operations:

$$C_1 = \{c \in C_0 \mid L = L'\}$$
$$C_2 = \{c \in C_1 \mid G = G'\}$$
$$C_3 = \{c \in C_2 \mid | B - B'| < \varepsilon\}$$

The dimension of C_3 is much smaller than that of C_0, however, candidates like "P" and "K" cannot be distinguished. To further reduce the number of candidates, for each candidate in C_3, a quick necessity matching is performed by equation (2) to form C_4, which is the final candidate set produced by the forepart prediction.

$$C_4 = \{c \in C_3 \mid N(f_c , X) = 0\} \qquad \ldots\ldots(2)$$

4.2 Character-adaptive Masking

The forepart prediction produces a small number of candidates. The best way to verify each candidate is to screen out the bitmap of the leftmost character and recognize it. To avoid the disadvantages of vertical cutting, a character-adaptive masking algorithm is being proposed, which uses an individual mask for each character. The mask is generated from the prototype bitmap of the character. Before defining the mask, the right boundary of the character is defined. Recall the width and height of the prototype bitmap are W and H, respectively. The right boundary is defined as follows:

$$RB = \{RB(i) | i \text{ is the row index of a bitmap and } 0 \le i \le H\}$$

Where *RB(i)* is the horizontal coordinate of the rightmost black point on the row i. If the entire row is white *RB(i)*. Then, the real width of the character is

$$W_c = \max_{0 \le i \le H} [RB(i)]$$

Now we can define the character mask.

Definition 1: The mask is a binary matrix whose row index is the same as that of the prototype bitmap and whose width is W_c. The value of each point in the mask is defined by equation (3), also as shown in Figure. 4.

$$Mask\ (i,j) = \begin{cases} 1, & 0 < j \le MAX\,(RB(i), W_c/2) \\ 0, & \text{otherwise.} \end{cases} \qquad \ldots\ldots(3)$$

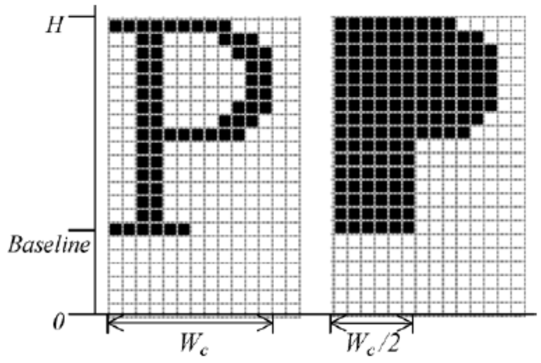


Figure 4.Character Mask of P

Notice that the right bound " MAX (RB(i) , W_c/2) is important for the sufficiency matching. For example, when the leftmost character is "B" the candidates may include "B", "P" and "L". If simply using as the right bound, the masked bitmap of "P" or "L" will also be recognized perfectly. Using the mask of a candidate to "AND" with the input bitmap pixel by pixel, one can segment the bitmap of the candidate from the remainder along the path determined by the right shape of the candidate. Character Adaptive Masking is very effective in segmenting linear and nonlinear merging types. For the overlapped type, at least the shape of the left character can be preserved.

V. IMPLEMENTATION & EXPERIMENTAL RESULTS

The system can be implemented using MATLAB R2010a. Here the input to the system is a Captcha image and output is the characters recognized from the image. Based on the accuracy of recognition robustness of Captcha is measured. A dataset of 100 images were used and a training set of 25 Captcha images were used for each classes of Captcha. Robustness is measured using the following formulae:

$$\text{Accuracy of segmentation} = \frac{\text{Correctly Segmented Characters}}{\text{Total Number of Characters}}$$

$$\text{Accuracy of recognized Characters} = \frac{\text{Correctly Recognized no. of characters}}{\text{Total no. of Characters}}$$

TABLE I. EXPERIMENTAL RESULTS CLASS1 & 2

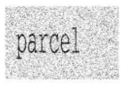
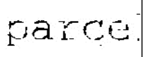
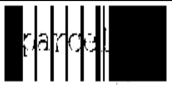
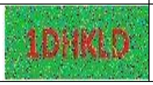
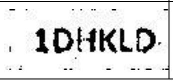
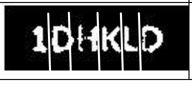
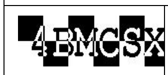
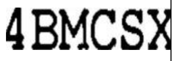
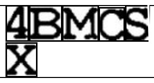
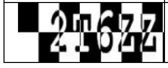
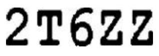
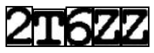
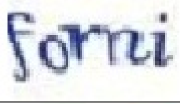
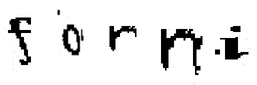
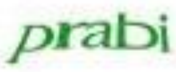
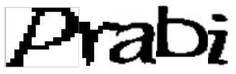
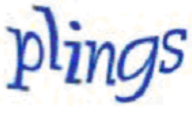
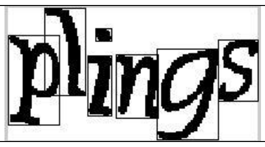
Class 1: Complex captcha			
Input Image	***Image after resizing and Noise removal***	***Segmented Image***	***Accuracy***
			100%
			100%
Class 2: Bot Detect Captcha			
			100%
			60%

TABLE II. EXPERIMENTAL RESULTS CLASS 3: GOOGLE CAPTCHA

Input Image	***Segmented Image***	***Accuracy***
		100%
		85%
		83%

VI. CONCLUSION & FUTURE SCOPE

The algorithm addressed in paper successfully breaks above mentioned text based CAPTCHAs. In the experimental results, it was found that algorithm can uniformly improve the segmentation rate over the traditional algorithm. The proposed algorithm makes novel and useful contributions to the field of CAPTCHA analysis. The future work includes evaluation of more Text-based captchas and to formulate guidelines & design principles for the generation of attack-resistant Captchas. The basic advantage of the developed system is that it can be used as a module in normal OCR, because a normal OCR program could not recognize character in distorted background. Also it can be used to rate and improve the security of various websites.

Acknowledgements

The authors wish to thank all the reviewers of the work for their valuable suggestions and advices to successfully complete the work.

REFERENCES

[1]. Technical Report:Ahmad.Salah-El-Ahmad, Jeff.Yan, Mohomad.Tayara, "The Robustness of Google CAPTCHAs" Technical report, Newcastle University, UK, 2011.

[2]. Note that the journal title, volume number and issue number are set in italics. Journal Papers:

[3]. Jiqiang Song , Zuo Li, Michael R. Lyu Shijie Cai" Recognition of Merged Characters Based on Forepart Prediction, Necessity-Sufficiency Matching, and Character-Adaptive Masking" IEEE Transactions On Systems, Man, And Cybernetics—Part B: Cybernetics, Vol. 35, No. 1, February 2005 Proceedings Papers:

[4]. J Yan and A S El Ahmad. "Breaking Visual CAPTCHAs with Naïve Pattern Recognition Algorithms", in Proc. Of the 23rd Annual Computer Security Applications Conference (ACSAC'07). FL, USA, Dec 2007. IEEE computer society. Pp 279-291.

[5]. G Mori and J Malik. "Recognising objects in adversarial clutter: breaking a visual CAPTCHA",IEEE Conference on Computer Vision & Pattern Recognition (CVPR), 2003 , IEEE Computer Society ,vol. 1 ,pp.I-134-I-141, June 18-20 ,2003.

[6]. K Chellapilla, K Larson, P Simard and M Czerwinski, "Building Segmentation Based Human-friendly Human Interaction Proofs", 2nd Int'l Workshop on Human Interaction Proofs, Springer-Verlag, LNCS 3517,2005