Recognition Based Segmentation of Connected Characters in Text Based CAPTCHAs

Rafaqat Hussain, Hui Gao, Riaz Ahmed Shaikh, Shazia Parveen Soomro
School of Computer Science and Engineering,
University of Electronics Science and Technology of China
Chengdu 611731, China
e-mail: 201414060105@std.uestc.edu.cn; huigao@uestc.edu.cn; riaz.shaikh@salu.edu.pk; shazosoomro@gmail.com

***Abstract*—Text based CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) is the most widely used mechanism adopted by numerous popular web sites in order to differentiate between machines and humans, however due to extensive research carried out by computer vision researchers, it is now a days vulnerable against automated attacks. Segmentation is the most difficult task in automatic recognition of CAPTCHAs, therefore contemporary Text based CAPTCHAs try to combine the characters together in order to make them as segmentation resistant against these attacks as possible. In this research, we have found vulnerabilities in such CAPTCHAs, a novel mechanism, i.e. the recognition based segmentation is applied to crop such connected characters, a sliding window based neural network classifier is used to recognize and segment the connected characters. Experimental results have proved 95.5% recognition success rate and 58.25% segmentation success rate on our dataset of tmall CAPTCHAs, this algorithm is further tested on two other datasets of slightly different implementations and promising results were achieved.**

Keywords-CAPTCHA; intelligent character recognition; computer vision; image processing; pattern recognition; machine learning

I. INTRODUCTION

In order to protect the valuable resources from unauthorized access, CAPTCHAs (Completely Automated Public Turing Test to tell Computers and Humans Apart) were introduced. Since its introduction, a friendly war began between CAPTCHA designers and Computer vision researchers. In response to attacks designed by these vision researchers the designers have tried to defend their CAPTCHAs against these automated attacks over the years. The pioneers in the field of CAPTCHA design, Von Ahn et. al [1] build the idea on the basis of hard AI problems and they believed that this is always a win-win game because if their CAPTCHAs will be broken then it will be a one step forward in the field of AI and otherwise a security mechanism can be provided to stop the spam. They encouraged the research community to solve their CAPTCHAs, which was also a motivation for us to work in this field. Breaking CAPTCHAs can help to identify the problems in the current design; it can also help to improve the computer vision algorithms related with Intelligent Character Recognition (ICR).

Various types of CAPTCHAs are being developed including Text, image, audio, and video but text based CAPTCHAs are still popular and most widely used due to their easy implementations [2]. The initial idea was to distort characters in a way that they become extremely hard to solve by pattern recognition programs and still usable by humans as shown in Fig. 1. No matter how much the individual characters are distorted even then the machine learning programs can be trained to identify the individual characters with higher accuracy [3], the space between characters can be exploited to segment the characters. Contemporary text based CAPTCHAs are therefore mainly based on the problem of segmentation rather than recognition [4] as shown in Fig. 2. Clearly it can be observed in Figure 2 that the CAPTHCA on the left contains 4 characters and all characters are connected with each other while the CAPTCHA shown on the right is an MSN CAPTCHA [5] with all connected and broken characters. Segmentation of these types of connected characters is a challenging task both in CAPTCHAs and handwritten recognition and/or intelligent character recognition.

Figure 1. An E-Z Gimpy CAPTCHA

Figure 2. Hard to solve CAPTCHAs based on segmentation resistance of connected characters.

Several segmentation attacks have been designed by discovering fatal design errors in the previous CAPTCHA implementations. These weaknesses were exploited by algorithms like Color Filling Segmentation (CFS) [4], shape contexts [6], Projection method [7] and Machine learning techniques [8] etc.

978-1-5090-1781-2/16/ $31.00 ©2016 IEEE

There are typically three main steps in solving the text based CAPTCHAs including preprocessing, segmentation and recognition. Depending on the type of CAPTCHA, preprocessing aims to make CAPTCHA image easy to analyze and includes the steps like gray scale conversion, binary conversion and noise removal, the next step is the segmentation of characters, i.e. to locate the individual characters in the whole word, segmentation aims to find 'where' the character is rather than finding 'what' the character is. The last step is to recognize these segmented characters. Research has shown that once the characters are properly segmented then the recognition is a trivial task and according to the research by Chellapilla et al. the computers have outperformed humans in this step [3]. The most difficult step in this regard is the segmentation of connected or merged characters. Therefore this research work focuses on the segmentation part. The rest of the paper is organized as follows: Section II discusses the related work, Section III provides the discussion on the preprocessing step, section IV provides the discussion of our recognition based segmentation attack, Section V provides the results and discussion of our method and it further discusses about the problems found in the said steps, Section VI provides the conclusion and future work.

II. Related Work

Currently there is no comprehensive universal segmentation algorithm which can pass all CAPTCHA systems. Depending on the particular type of CAPTCHAs the researchers have developed methods to pass those tests accordingly. Mori and Malik [4] used shape contexts to solve the EZ-gimpy and GIMPY images with a success rate of 92% and 33% respectively. Chellapilla et al [8] used the machine learning techniques to break number of early CAPTCHAs and achieved 4.89 % of success on the early version of Google CAPTCHA back in 2004. Huang et al [7] proposed a projection based segmentation algorithm to break early MSN CAPTCHA of 2008. They observed that clutter found in the said CATPCHA had flatter and smaller horizontal projection than the projection of the normal characters on x-axis. By exploiting these design flaws they achieved a segmentation success rate of 55.13%. In 2011 Ahmad et al attacked a mechanism known as CCT (acquired by Google in their home design and reCAPTCHA), they used pattern based detection of characters and achieved 46% segmentation accuracy with overall success rate of 33%. In a study conducted by Chen et al. [9], they achieved an average recognition rate of 81.05% on CAPTCHAs containing noisy lines and points. They used the probability pattern framework to recognize the said CAPTCHAs. Huang et al. [10] proposed two segmentation methods known as projection and separation of middle axis point. Using these methods they removed the line clutters and identified the warping characters. Sakottas et al. [11] proposed a technique called Template Matching Correlation (TMC) technique to analyze the robustness of text based CAPTCHAs, this technique consists of conversion of image into binary, removing the noise, segmentation and recognition methods. They discovered by their simulations that robustness is improved when the image is distorted by noise background and font skew from 0.3 to 0.4. Starostenko et al. [12] offered a method to break the reCaptcha of version 2012, they have used three color bar codes to segment the characters, and have also proposed an SVM classifier to recognize the characters achieving a segmentation success of 56.3% and recognition success of 95% on reCaptcha of 2012. Chandavale and Sapkal [13] presented an interesting study on the usability and security of CAPTCHAs, they revealed that using the colors actually reduces the security instead of improving it and it also irritates the users at times, the use of noise also occasionally affects the usability and security instead of improving it. Zhang and Wen [14] presented a fuzzy matching method to crack a code from the picture; they have proposed two mask creation methods for measuring the similarity of characters with the help of theses masks.

III. Pre Processing

In this work, initially we collected 300 samples of tmall [15] (a popular shopping website in China) CAPTCHAs which consists of 4 characters where all the characters are connected with each. The following preprocessing operations were performed:

(a) Gray Scale conversion: RGB images of tmall CAPTCHAs are converted to gray scale by eliminating the hue and saturation information while retaining the luminance as shown in Fig. 3 (left).

(b) Binary Conversion: The obtained gray scale image is further converted to binary image by using Otsu threshold mthod as shown in Fig. 3 (Right image)

(c) Cropping from Edges: The obtained binary image is further cropped from edges in order to remove the white space on all the 4 sides of it. The cropped binary image is shown in Fig. 4 (Right). This step reduces the size of the image so not only faster processing can be achieved but it also simplifies the process of segmentation as discussed in Section 4.

Figure 3. Gray scale image (left), binary image (Right)

Figure 4. Original binary image (Left), cropped binary image (Right)

IV. Recognition Based Segmentation

Although segmentation is usually performed prior to the recognition process but in our case segmentation is based on the recognition. Therefore we will first discuss the recognition of individual characters and then we will discuss the recognition based segmentation of whole CPATCHAs images.

A. Recognition of Cropped Characters

We have trained a neural network (Artificial neural network with back propagation) on manually cropped characters. This neural network is trained on 300 CAPTCHA images, i.e. 1200 characters (As each CAPTCHA image consists of 4 characters). Fig. 5 shows few samples of this corpus of characters.

Figure 5. Few samples of cropped charcters from our corpus of training dataset.

Although possible number of classes are 62 (26*2 letters each capital and lower case and 10 digits) but here in tmall CAPTCHAs the digits 0 and 1 while characters I, L and O are not used. Therefore our dataset reduced to 54 classes.

All characters are normalized to a fixed size of 25x25 patterns and consequently represented by a feature vector of 625 values. Matlab 8 is used to train the neural network having 625 input values and 54 output values (Number of classes). After training on different hidden layers and multiple trials of training we have achieved an accuracy of 96.5 %, 95.2% and 94.4 on training data, validation data and test data respectively.

B. Segmentation with Trained Neural Network

Once the neural network is trained on individual cropped characters, now an algorithm is developed to recognize and ultimately segment the characters with neural confidence. This is demonstrated in Fig. 6. All the steps of this algorithm are explained below:

a. Input CAPTCHA: The image containing CAPTCHA text is feed into the system.
b. Crop Edges: As discussed in Section III, the images are cropped from the edges in order to remove the white space surrounding it.
c. Calculate window size: After cropping the image from edges, the width of the image is calculated. In order to find the average character size (avg_cs), it is divided by total number of characters in the whole image (here the number is 4 in our case of tmall CAPTCHA).
d. Calculate sub windows size: Sub windows are the windows extracted to crop individual character. We have cropped every character on three locations, i.e. first sub window is equal to round of average character size divided by 0.25, i.e. round (avg_cs/0.25), second sub window is exactly equal to the average character size while the third sub window is equal to round of average character size multiplied by 0.25, i.e. round (avg_cs/0.25).
e. Send feature vectors to ANN: After extracting the sub windows, each sub window is normalized as discussed above in this section; the resultant feature vector of 625 values (25x25 patterns) is sent to the neural classifier already trained on such character corpus. The classifier matches each window pattern with pre-defined patterns of characters and consequently sends the results in terms of percentage of confidence.
f. Save scores: The results received from classifier are stored for each sub window. The location in terms of coordinates of these sub windows is also stored for future reference as a cut point.
g. Calculate highest score: The highest score is the score of a sub window containing highest percentage of confidence returned by the classifier. The coordinate location of this window is retrieved and the main window is cropped from this location. The remaining main window (after cropping first character) is stored for later extraction of remaining characters.
h. Next character: After extracting every character, it is verified that it is the last character or still there are remaining characters, in case of remaining characters the process is repeated from step ii, otherwise the obtained characters are displayed.
i. Output characters: The characters retrieved with highest confidence are displayed in this final step.

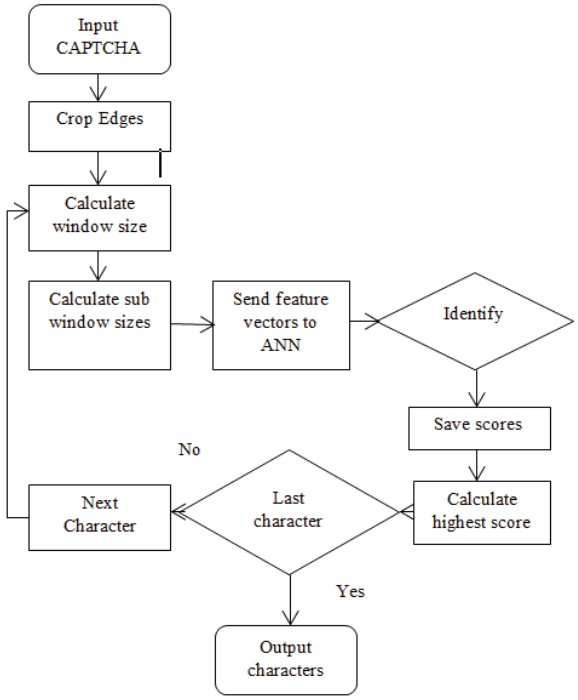

Figure 6. Recognition based segmentation algorithm.

C. *Experiments on Other Datasets*

In order to check the generic nature of our algorithm we tested it on JCAPTCHA [16] and Microsoft CAPTCHA [15], JCAPTCHA is an open source library to develop and integrate CAPTCHAs in your web sites, we created 500 samples of 5 characters each as shown in Fig. 7 and performed similar preprocessing, training and recognitions steps as performed on tmall CAPTCHA without any modifications. In order to test Microsoft CAPTCHAs we downloaded 250 images of Microsoft CAPTCHAs from their new account registration page [5]. Microsoft CAPTCHA is based on connected and broken characters; it consists of 6 to 11 characters with a reduced number of classes of 23 different characters instead of 62. With a slight variation in our algorithm we achieved an ample access on both of these CAPTCHA implementations. The combined results of these CAPTCHAs along with tmall CATPCHA are summarized in Table I.

Figure 7. Samples of JCAPTCHA images.

TABLE I. RESULTS OF ALL TESTED CAPTCHAS

Type of CAPTCHA	Success recognition rate (SRR) %	success segmentation rate (SSR)%	Dataset (No. of samples)
tmall	95.5	58.25	300
JCAPTCHA	97.25	63.5	500
Microsoft	72.33	30.5	250

V. RESULTS AND DISCUSSIONS

In this section we discuss the quantitative analysis of the effectiveness of our algorithm. As shown in Table 1, Using dataset of only 300 images of 4 characters each, we obtained Success Recognition Rate (SRR) of 95% on individual characters and success segmentation rate (SSR) of 58.25% on whole CAPTCHA images containing 4 characters each. These results improved on increasing number of samples (500 samples) in training data in case of JCAPTCHA with SRR as 97.25% and SSR 63.5%, while using 250 samples as training data in Microsoft CAPTCHA we achieved SRR of 72.3 while SSR as 30.5. This is a high success to beat a CAPTCHA and a serious fatal error to be considered in the designing of an effective CAPTCHA. However few highly overlapping characters were not correctly segmented by our algorithm and in case of Microsoft CAPTCHAs, training on broken characters also suffered from lower accuracy in recognition and segmentation as compared to tmall and JCAPTHCAs.

VI. CONCLUSION AND FUTURE WORK

In this reasearch work, an artificial neural network with backpropagation is trained to identify the individual characters first and then this trained classifier is used to segment the connected characters in text based CAPTCHAs, characters with highest neural confience are extracted from the CAPTCHA images by using variable size increasing windows. The aim of this research was to verfiy the robustness of CAPTCHAs contianing connected characters and/or find a way to solve the problem of intelligent character recognition. The resluts obtained by using this recognition based segmentation method are promising and these results encourage further research on bigger datasets, better results can be achieved by adding more and more samples in the training data. Overlapping characters and other types of resistance techniques can be validated as a future work.

REFERENCES

[1] L. V. Ahn, M. Blum, N.J Hopper, and J Longford, "Captcha: Using hard AI problems for security," In Advances in Cryptology—EUROCRYPT, vol. 2656, pp. 294-311, Springer 2003.

[2] J. Yan and A. S. El Ahmad, "Usability of captchas or usability issues in captcha design," In Proceedings of the 4th symposium on Usable privacy and security, SOUPS '08, pages 44–52, New York, USA, 2008.

[3] K. Chellapilla , K. Larson , P. Simard, and M. Czerwinski, "Computers beat humans at single character recognition in reading based human interaction proofs," In Proceedings of the Third Conference on E-Mail and Anti-Spam 2005.

[4] J Yan and A. S. El Ahmad. "A Low-cost Attack on a Microsoft CAPTCHA," 15th ACM Conference on Computer and Communications Security (CCS'08). Virginia, USA, ACM Press. pp 543-554, Oct. 2008.

[5] https://signup.live.com (Last accessed on 10 March 2016)

[6] G. Mori and J. Malik. "Recognising Objects in Adversarial Clutter: Breaking a Visual CAPTCHA," IEEE Conference on Computer Vision and Pattern Recognition (CVPR'03), Vol. 1, pp.134-141, June 2003.

[7] S. Y. Huang, Y. Lee. K., Bell, and Z.H. Ou. "A projection-based segmentation algorithm for breaking MSN and YAHOO CAPTCHAs, " In proceedings of the world congress on Engineering, London, U.K. July 2008.

[8] K. Chellapilla and P. Simard, "Using machine learning to break visual human interaction proofs (HIPs)," Advances in Neural Information Processing Systems, Vol. 17, pp. 265-272, 2004.

[9] C. J. Chen, Y Wei, and W. P. Fang, "A Study on Captcha Recognition," In 10th international conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), Kitakyushu, pp. 365-398, Aug. 2014.

[10] S. Y. Huang, Y. K. Lee, G. Bell and Z. H. Ou, "An efficient segmentation algorithm for CAPTCHAs with line cluttering and character warping," Multimedia Tools and Applications, vol. 48, pp 267-289, June 2010.

[11] P. Sakkatos, , W. Theerayut, V. Nuttapol, and P. Surapong, "Analysis of text-based CAPTCHA images using Template Matching Correlation technique," In Information and Communication Technology, Electronic and Electrical Engineering (JICTEE), 2014 4th Joint International Conference on, pp. 1-5. IEEE, 2014.

[12] O. Starostenko, C. Cruz-Perez, F. Uceda-Ponga, & V. Alarcon-Aquino, "Breaking text-based CAPTCHAs with variable word and character orientation," Pattern Recognition, vol. 48, issue 4, pp. 1101-1112, Sep. 2014.

[13] A.A. Chandavale, A. Sapkal, "Security Analysis of CAPTCHA," In Recent Trends in Computer Networks and Distributed Systems Security, vol. 335, pp. 97-109, Springer Berlin Heidelberg, 2012.

[14] H. Zhang, and X. Wen. "The Recognition of CAPTCHA Based on Fuzzy Matching," Foundations of Intelligent Systems. vol. 227, pp. 759-768, Springer Berlin Heidelberg 2014.

[15] https://www.tmall.com (Last accessed on 10 March 2016)

[16] http://jcaptcha.sourceforge.net/ (Last accessed on 4 January 2016)