Evaluating the Usability of Optimizing Text-based CAPTCHA Generation

Suliman A. Alsuhibany
Computer Science Department
College of Computer, Qassim University
Buridah, Saudi Arabia

***Abstract*—A CAPTCHA is a test that can, automatically, tell human and computer programs apart. It is a mechanism widely used nowadays for protecting web applications, interfaces, and services from malicious users and automated spammers. Usability and robustness are two fundamental aspects with CAPTCHA, where the usability aspect is the ease with which humans pass its challenges, while the robustness is the strength of its segmentation-resistance mechanism. The collapsing mechanism, which is removing the space between characters to prevent segmentation, has been shown to be reasonably resistant to known attacks. On the other hand, this mechanism drops considerably the human-solvability of text-based CAPTCHAs. Accordingly, an optimizer has previously been proposed that automatically enhances the usability of a CAPTCHA generation without sacrificing its robustness level. However, this optimizer has not yet been evaluated in terms of improving the usability. This paper, therefore, evaluates the usability of this optimizer by conducting an experimental study. The results of this evaluation showed that a statistically significant enhancement is found in the usability of text-based CAPTCHA generation.**

Keywords*—*text-based CAPTCHA; usability; security; optimization; experimentation; evaluation

I. INTRODUCTION

Nowadays, several studies have been conducted for web-based services that may be exposed by some attacks using such tools. In particular, researchers tend to make some of the theoretical and practical methods not only to prevent these attacks, but also to distinguish bots from humans. One of these methods is called Human Interactive Proofs (HIPs). Where from these endeavours, a captcha [1] has been developed to resist these attacks and improve the robustness level of such systems [7].

A captcha (Completely Automated Public Turing test to tell Computer and Human Apart) has been proposed to improve the security of services and verify that a client request is submitted by individual users from online operations rather than by malicious software. It is a program that generates and grades tests that humans can pass easily, whereas computers cannot [13]. A good captcha should satisfy two main requirements: robustness and usability. The robustness aspect is its strength to defend against adversarial attacks; the usability aspect is the ease with which humans pass its challenges [5]. These aspects have attracted considerable attention in the research community (e.g. [2, 11, 10, 8, 9]). The text-based captcha is the most commonly deployed type in websites, such as *Gmail*, *eBay*, and *Facebook*, to date, with many advantages [2].

Over the past decade, the generation of captcha uses combinations of distorted characters and obfuscation techniques that humans can recognise, whereas they may be difficult for automated scripts. Recently, collapsing or Crowding Characters Together (CCT) technique has been recommended in several studies, such as in [1, 2], as the main anti-segmentation technique. Although of this, a number of character confusions that lead to unsolvable schemes by humans have been recognised in [3] which are expanded in our previous work [4]. Additionally, the accuracy and response time of solving the captcha drop drastically the human-solvability for websites that utilise this technique such as Google and Recaptcha [4, 5, 6, 8]. To overcome this concern, an optimizer that can optimise the generated text of captchas to keep the same level of security while improving the usability level for a varied character set has been proposed in [4].

In particular, the optimizer is designed to be embedded in a text-based captcha generator, and the generated text is optimised based on a set of rules which are empirically derived. These rules are then fed into a developed captcha generator with different fonts and size. Afterwards, the optimizer checks if there is any confusion for character or combination of characters, and then replaced with a set of non-confusing characters based on its position [4], and more details will be given in Section 2. However, the usability of this proposed optimizer has not yet been evaluated. Thus, this paper evaluates the usability of this optimizer and the main hypothesis H_1 is that "*The human-solvability of text-based captchas is significantly improved after using the optimizer.*"

To validate this hypothesis, an experimental study is conducted in which a text-based captcha generator that contains the optimizer is developed. The experiment focuses on the effect of collapsing mechanism on the usability of a generated scheme. A within-subject design (i.e. *prepost-test design*) experiment was used in which fifty-three subjects are participated. The results of the experiment showed that there is a statistically significant improvement after using the optimizer in terms of the accuracy and response time. So, this result supported our hypothesis.

The rest of this paper is organized as follows. Section 2 presents an overview of the optimizer. Section 3 explains the

[1] For the sake of readability, the acronym is written here in lowercase throughout this article as it is normally written in capitals

methods. Section 4 presents the results. The results are discussed in Section 5. Section 6 concludes the paper with future works.

II. An Overview of the Optimizer

This section highlights the optimizer that has been proposed in [4]. That is, the optimizer aims to improve the usability of text-based captcha without interfering with their robustness level. In particular, there are three important characteristics that the optimizer can exhibit. These are: optimising the generated text based on a set of rules; refining the optimised text; and positioning the optimised character [4]. Fig. 1 shows the proposed design of the optimizer. Each of the optimizer's characteristics is explained below.

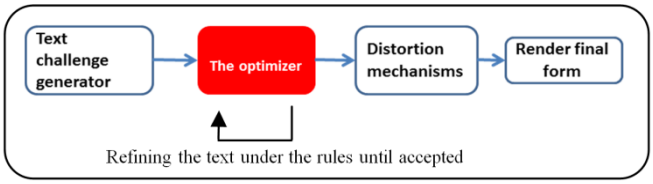

Fig. 1. The design of the optimizer [4]

1) The optimization rules: To make the captcha more robust against attacks, different distortion methods have been deployed, for example, CCT, random arcs, overlapping characters and random angled connected line. Specifically, for CCT, captchas appear to be more difficult even for the human. By increasing the level of distortion, a number of confusing letters such as "bl" can resemble "lol" or "ld". Furthermore, a captcha generator is built that can produce these confusing character, for example, when the characters touch or overlap with each other. By analysing these, the optimization rules are constructed. Hence, the optimization rules are collected empirically. Moreover, as will be seen later, the confusing character is replaced with a suitable non-confusing character based on its position [4].

The non-confusing characters are a set of characters that are developed empirically by both the confusion matrix and the generator. As shown in Fig. 2, the substitution process is accomplished by replacing a confusing character with one of a series of non-confusing characters. However, the position of the confusing character is considered where replacing confusion characters with a random character from non-confusion characters set can result in another confusion character [4]. More details are in the next section.

c is any character in the alphabet C, $c \in C$

S is a series of n characters, $S = \{ c_1, ..., c_n \}$, $n \geq 1$, $c_i \in C$

$\boldsymbol{S_R}$ is a series of characters that replace confusing characters, $S_R \subset S$.

Rule: $S_1 \rightarrow S_2$, $S_1 \in S_R$, $S_2 \in S$

Fig. 2. The general rule of replacing [4]

2) Refining the optimised text: As shown, optimising the generated text operates by replacing a character or combination of characters that caused the confusion with non-confusing characters. This, however, may cause a new character confusion. For example, in the case of the "cl" rule, replacing "l" with "m" can resemble "am". Thus, the step of refining the optimised text can simply reduce the possibility of a new character confusion that may occur depending on the position of the character as shown in Fig. 3. It is important to note that the algorithm will be terminated when replacing the confusing character with a non-confusing character that will not effect the remaining characters (*Refine= True* in Fig. 3). In other words, the termination of the refining step occurs when generated text is free from all possible confusing characters [4]. The position of the optimised character is detailed in the next section.

***GT**: Generated Text.*
***CC**: Character Confusion.*

```
While (Refine==False)
{
    Check GT;
    If there is CC then replacing character;
    Else Refine=True;
}
```

Go to the next phase (i.e. distortion mechanism).

Fig. 3. Illustration of the optimizer's algorithm [4]

3) The position of the optimised character: The position of the replaced character is important in the process of optimization. There are three possible positions in the process of replacing characters. Firstly, the optimised character is the first character of the text, so it should only check the effect of the second character on the optimised character. Secondly, the optimised character is in the middle. Therefore, it should check both the effect of the right character and the left character on the optimised character. Finally, the optimised character is the last character, and it should check only the effect of the previous character to the last character. The rules of these positions are presented in Fig. 4 [4].

c and S are as in Fig. 2.

Text of CAPTCHA $= \{ P_1, ..., P_i, ..., P_m \}$, i is the position of the character, $P_i \in C$

$\boldsymbol{S_R}$ is a series of characters that replace confusing characters, $S_R \subset S$

$\boldsymbol{S_c}$ is a series of characters that cause confusion, $S_c \subset S$.

Rule of position 1 : $S_R \rightarrow P_1$, $P_{i+1} \notin S_c$

Rule of position 2 : $S_R \rightarrow P_i$, $P_{i-1} \notin S_c$, $P_{i+1} \notin S_c$

Rule of position 3 : $S_R \rightarrow P_m$, $P_{m-1} \notin S_c$

Fig. 4. Rules of the positions [4]

III. Methods

A controlled laboratory experiment in which participants were asked to solve a set of generated captchas after and before

using the optimizer is conducted. The aim of this experiment is to evaluate the usability of the proposed optimizer in [4] and has been highlighted in the previous section. The following sections present the setup and the procedure of the experiment.

A. Experiment Setup

The experiment involves subjects to solve a set of captchas that are generated by the developed generator. The experiment design, participants, system, variables, and materials are explained in this section.

1) Experiment Design: We use a within-subject design, which means that each participant is assigned to all of the following experimental sessions. Session 1 represents *solving captchas before using the optimizer*, while Session 2 represents *solving captchas after using the optimizer*.

2) Users: Fifty-three participants were recruited for this experiment, 40 male and 13 female. The mean age of the participants was approximately 25 years. More than half of participants came from technical background (31), whereas the remaining came from non-technical backgrounds. Participants from technical category included university students from science and engineering, while the non-technical category came from social science disciplines.

3) System: A captcha generator is developed by using a Java programming language that embeds the proposed optimizer. This generator produces challenges with different types of text which includes all the confusion characters that presented in [4]. In addition to these confusion characters, we discovered a new set of confusion characters as shown in Table 1.

TABLE I. A NEW SET OF CONFUSION CHARACTERS AND THEIR OPTIMIZATION RULES

Characters	Problem	Enhancement
“ck”	It can resemble “ok” or “ak”	Replace character “c” with character “w”
“cn”	It can resemble “on”	Replace character “c” with character “z”
“cp”	It can resemble “op” or “qo”	Replace character “c” with character “z”
“lo”	It can resemble “b” or “p”	Replace character “l” with character “z”
“pl”	It can resemble “ld” or “lq”	Replace character “l” with character “w”
“rl”	It can resemble “nl”	Replace character “l” with character “g”
“rp”	It can resemble “np”	Replace character “r” with character “w”
“ol”	It can resemble “d” or “q”	Replace character “o” with character “w”

Furthermore, in order to enable the users to solve the generated captcha schemes, a Graphical User Interface (GUI) is developed by using a Java Applet, as shown in Fig. 5. The main goal is that the participants are asked to recognise the letters that are generated by the developed generator, and submit them by pressing on the submit bottom. In case a participant presses the submit bottom before writing the presented letters or leaves the box of the letters, which enables the user to write the recognised letters in, empty, then a warning message is appeared.

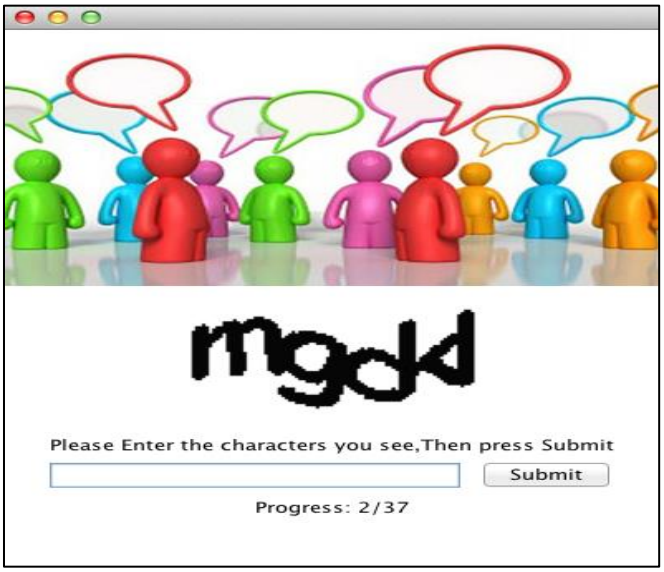

Fig. 5. The developed GUI

4) Variable: The main independent variable of this experiment is the optimization technique. The accuracy of solving the generated captcha and the response time (i.e. the time consumed) to solve the generated captcha are the dependent variables.

5) Materials: stimulus and rational: The stimulus material provided to participants consisted of a set of generated schemes before and after applying the optimization algorithm. The subjects were asked to solve this set sequentially. The same set was assigned to all subjects, rather than generating different sets. There were several reasons for this. First, different sets may be of different schemes, making the measurement and comparison of participant’s answer a difficult task. Second, different sets might be applied because the generator is developed by the author. This would again introduce biases that are difficult to control. Finally, using the same set of captchas for everyone affected experimental control over unanticipated biases.

B. Procedure

In this section, the way the experiment was run is explained, i.e. instructions to participants, tasks, and the data collected.

1) Instruction: Subjects were instructed to solve the presented captchas by writing their letters as they are appeared. The subjects were instructed that there are two

sessions, and 20 minutes break between these sessions [2]. Subjects were instructed that there are 37 captchas that will be presented in each session, sequentially. These 37 captchas were chosen in order to reflect all possible confusion characters that mentioned in [4] as well as the discovered set in Table 1. Subjects were told that if they needed a break during a session, they were to do so after they had solved all sessions' captchas. Subjects were able to gauge their progress by looking at a counter at the middle of the screen which showed how many captcha had been solved so far and how many yet remained. Subjects were admonished to focus on the task and to avoid distractions, such as talking with the experimenter, while the task was in progress.

2) Takes: The experiment was conducted in a controlled laboratory environment to avoid any distractions and collect the desired data without any biases. After every captcha sent by the participant, the system is not giving information about the recognition made whether the submit letters of a generated captcha are correct or not[3], and once all captchas are solved, a notice message that the task is done and thank you for the participation is shown. Finally, the participant was asked to fill a short survey/questionnaire about his or her experience.

3) Collected data: The letters of each submitted captcha and the time taken to solve it are recorded by the system.

IV. Results

In the experimental study, all participants successfully completed their tasks. The following discusses the hypothesis regarding the accuracy of solving captchas before and after using the optimization algorithm, the response time, the accuracy vs. response time and solvability of captchas.

A. Testing Hypothesis: Does the solvability of text-based captcha improved?

The average accuracy of solving the presented captchas before and after using the optimization algorithm is shown in Fig. 6. It can be seen that the accuracy of solving captchas in session 2 had significantly enhanced compared with session 1. This indicates that there are implications of applying the optimization algorithm. In particular, in session 1, the accuracy was less than 60%, and was more than 90% in session 2. This signifies that the accuracy in session 1 was far less than in session 2, possibly because eliminating the confusion characters, which are presented in session 1's samples. The statistical significance of this will now be discussed.

Table 2 compares the solvability in two sessions, with respect to the accuracy before applying the optimizer (left column) as well as after applying the optimizer (right column). For both, average (Avg.), standard deviation (SD), minimum (Min) and maximum (Max) values are provided. It was found that the average accuracy before using the optimizer was %57.54, while it was %95.7 after using the optimizer. A t-test yields a result of t=-23.37, p<0.0001, indicating that the difference between session 1 and session 2 is indeed statistically significant.

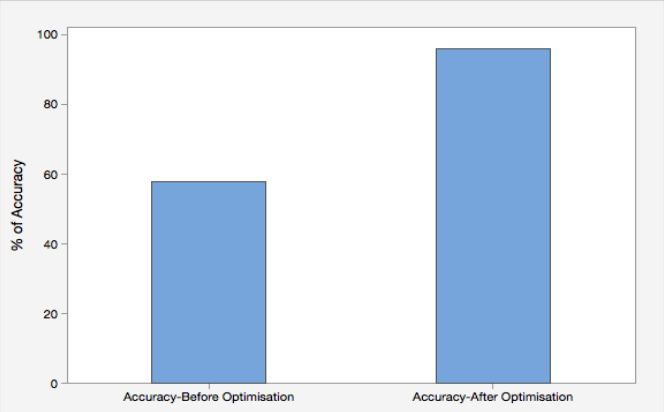

Fig. 6. The average accuracy of solving captchas before and after using the optimzation algorithm

TABLE II. Accuracy of Both Before and After using the Optimizer

Accuracy before using the optimizer (Session 1)					Accuracy after using the optimizer (Session 2)				
N	Avg.	SD	Min	Max	N	Avg.	SD	Min	Max
53	%57.5	10.8	34.0	77	53	%95.7	4.8	83.0	100

This result validates the hypothesis H_1 that the human-solvability of text-based captcha is significantly improved after using the optimizer.

B. Response Time

With respect to the response time, there was found to be a significant difference as shown in Table 3. That is, the average response time before using the optimization algorithm was 8.72 minutes, while it was 5.20 minutes in session 2. A t-test yields a result of t=6.42, p<0.0001, indicating that the time consumed for responding in session 1 is significantly higher than that in session 2. In other words, a statistically significant difference is found in the response time variable.

TABLE III. Response Time of Both Before and After using the Optimizer

Response Time (min) before using the optimizer (Session 1)					Response Time (min) after using the optimizer (Session 2)				
N	Avg.	SD	Min	Max	N	Avg.	SD	Min	Max
53	8.72	3.6	4.2	18.0	53	5.20	1.5	2.80	8.98

C. Accuracy vs. Response Time

In this section, we would like to see whether there is an inverse relationship between the accuracy and response time. In particular, a correlation between the accuracy and response time in which an increase in the value of the accuracy results in a decrease in the value of response time or vice versa. Therefore, by looking at users' performance, an obvious trade-off between the accuracy and the response time can be observed. For example, before using the optimization

[2] This is to avoid any unnecessary confounding factor biasing the results (at the cost of starting the next session immediately).
[3] This is to evaluate the accuracy of each participant's respond by comparing it with the generated one.

algorithm, presenting captchas that include some character confusions lead to decreasing the accuracy, and this directs to increase the response time, as shown in Fig. 7 and Fig.8.

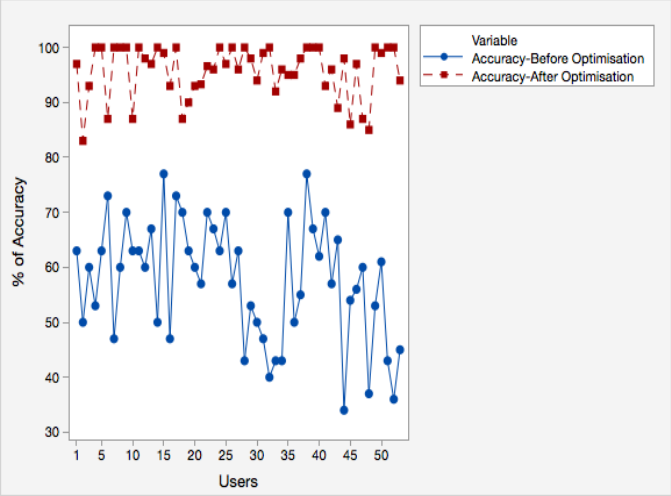

Fig. 7. The accuracy of solving captchas before and after using the optimzation algorithm by all users

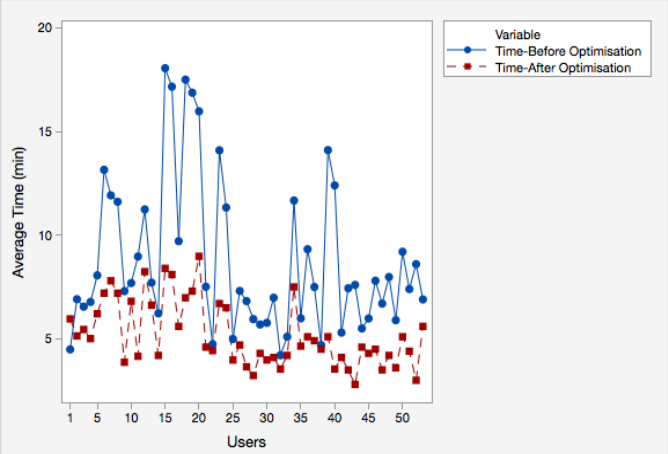

Fig. 8. The average response time of solving captchas before and after using the optimzation algorithm by all users

D. Solvability of captchas

Qualitative data were collected, in the form of a survey, to get such feedback about the generated captchas for both before (i.e. session 1) and after (i.e. session 2) using the optimization algorithm. In particular, we were looking for the difficulty level of the generated captchas regarding the solvability.

In session 1, most of the participants, 89% (47 out of 53), stated that the generated captchas were annoying due to the confusion of some characters, for example, “o” and “c.” On the other hand, 11% (6 out of 53) indicated that the generated captchas were not easy but solvable.

In session 2, 93% (49 out of 53) pointed out that the generated captchas were usable; the remaining 4 participants (7%) found that the generated captchas were not easy but solvable.

V. Discussion

The experimental study provides statistically significant evidence that the human-solvability of text-based captcha is significantly improved after using the optimizer. In particular, Table 2 shows that the accuracy of solving captchas in session 2 was significantly higher than in session 1. Accordingly, the main objective of this experiment, namely “solvability improved” is established. In other words, the result of the experiment does support the hypothesis.

Interestingly, the experimental study also gives statistically significant indication that the response time before using the optimization algorithm was higher than after using the optimization algorithm, as shown in Table 3. This result shows an inverse correlation between the accuracy and response time, as demonstrated in Fig. 7 and Fig. 8. However, by looking at Fig. 8, several users took almost the same response time in both sessions, and this also is confirmed by the survey’s results. The possible explanation can be that users may solve presented captchas as fast as possible, but without taking care of their level of typing accuracy. This can be shown obviously in the accuracy results in Fig. 7.

Since a good captcha should satisfy the robustness and usability aspects, our paper is evaluated only the usability aspect. However, evaluating the robustness is beyond the scope of this paper. For this, a future work is required to achieve the key point of the proposed optimizer in [4] which is that the usability of captchas is improved without sacrificing their robustness level. Furthermore, the results of our paper may contribute towards the recently introduced benchmark in [12] that the generation of usable-secure text-based captcha can be improved.

VI. Conclusion and Future Work

This paper evaluates the usability of optimising captchas through a controlled experimental study. The hypothesis behind this evaluation is that the human-solvability of text-based captcha is significantly improved after using the optimizer. The rationale of this hypothesis is based on the observation that applying such distortion mechanisms that act as a defence approach against segmentation attack increases character confusions. The results of this evaluation showed that the optimization algorithm is significantly enhanced the solvability of generated captchas. Not only this, but also the optimization algorithm is significantly improved the response time of solving captchas.

Our ongoing work would be to conduct a security experiment to validate that the optimizer keeps the same level of security while improving the usability level. Furthermore, as some current approaches are using a combination of different types of characters (e.g., numbers and letters) to avoid human’s recognition confusion on text-based captchas, we would compare the results of the optimizer with this kind of settings.

Acknowledgment

We would like to thank all participants of our experiment. The author gratefully acknowledges Qassim University, represented by the Deanship of Scientific Research, on the

material support for this research under the number (3233) during the academic year 1436 AH / 2015 AD.

REFERENCES

[1] J. Yan and A. S. E. Ahmad, "A low-cost attack on a Microsoft captcha," *Proceedings of the 15th ACM conference on Computer and communications security - CCS '08*, 2008.

[2] E. Bursztein, M. Martin, and J. Mitchell, "Text-based CAPTCHA strengths and weaknesses," *Proceedings of the 18th ACM conference on Computer and communications security - CCS '11*, 2011.

[3] J. Yan and A. S. E. Ahmad, "Usability of CAPTCHAs or usability issues in CAPTCHA design," *Proceedings of the 4th symposium on Usable privacy and security - SOUPS '08*, 2008.

[4] S. A. Alsuhibany, "Optimising CAPTCHA Generation," 2011 Sixth International Conference on Availability, Reliability and Security, 2011.

[5] E. Bursztein, S. Bethard, C. Fabry, J. C. Mitchell, and D. Jurafsky, "How Good Are Humans at Solving CAPTCHAs? A Large Scale Evaluation," *2010 IEEE Symposium on Security and Privacy*, 2010.

[6] M. Motoyama, K. Levchenko, C. Kanich, D. McCoy, G. M. Voelker, and S. Savage, "Re: CAPTCHAs-Understanding CAPTCHA-Solving Services in an Economic Context," *USENIX Security Symposium*, Vol. 10, 2010, pp. 3.

[7] C. J. Hernandez-Castro, M. D. R-Moreno, and D. F. Barrero, "Using JPEG to Measure Image Continuity and Break Capy and Other Puzzle CAPTCHAs," *IEEE Internet Computing IEEE Internet Comput.*, vol. 19, no. 6, pp. 46–53, 2015.

[8] E. Bursztein, A. Moscicki, C. Fabry, S. Bethard, J. C. Mitchell, and D. Jurafsky, "Easy does it: More usable captchas," *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14*, 2014.

[9] Y.-L. Lee and C.-H. Hsu, "Usability study of text-based CAPTCHAs," *Displays*, vol. 32, no. 2, pp. 81–86, 2011.

[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," *Multimed Tools Appl Multimedia Tools and Applications*, vol. 48, no. 2, pp. 267–289, Jan 2009.

[11] G. Mori and J. Malik, "Recognizing objects in adversarial clutter: breaking a visual CAPTCHA," *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*

[12] S. A. Alsuhibany, "A Benchmark for Designing Usable and Secure Text-Based Captchas," *International Journal of Network Security & Its Applications*, vol.8, no 4, pp. 41–54, July 2016.

[13] L. Von Ahn, M. Blum, J. and Langford, "Telling humans and computers apart automatically, "*Communications of the ACM*, vol. *47*, no 2, pp.56–60, 2004.