Intelligent Automation & Soft Computing
DOI:10.32604/iasc.2022.019913
Article

Tech Science Press

A Parametric Study of Arabic Text-Based CAPTCHA Difficulty for Humans

Suliman A. Alsuhibany[*] and Hessah Abdulaziz Alhodathi

Department of Computer Science, College of Computer, Qassim University, Saudi Arabia
*Corresponding Author: Suliman A. Alsuhibany. Email: salsuhibany@qu.edu.sa
Received: 01 May 2021; Accepted: 02 June 2021

Abstract: The Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA) technique has been an interesting topic for several years. An Arabic CAPTCHA has recently been proposed to serve Arab users. Since there have been few scientific studies supporting a systematic design or tuning for users, this paper aims to analyze the Arabic text-based CAPTCHA at the parameter level by conducting an experimental study. Based on the results of this study, we propose an Arabic text-based CAPTCHA scheme with Fast Gradient Sign Method (FGSM) adversarial images. To evaluate the security of the proposed scheme, we ran four filter attacks, which led to a success rate of less than 5%. Thus, we developed a defensive method against adaptive attacks which is a standard for evaluating defenses to adversarial examples. This method is ensemble adversarial training, and it gave an accuracy result of 41.51%. For the usability aspect, we conducted an experimental study, and the results showed that it can be solved by humans in a few seconds with a success rate of 93.10%.

Keywords: Arabic text-based CAPTCHA; cyber security; adversarial examples; robustness; usability

1 Introduction

With the ubiquitous spread of the internet and its use in various areas of our lives such as e-commerce and electronic services, the need to preserve security has received increasing attention, with website developers seeking to apply several forms of protection against attacks. One of these forms is a CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart), which is one of the most important types of Human Interaction Proofs (HIP) systems [1,2].

A CAPTCHA is a security measure to prevent damages and automated exploitations with abusive purposes. That is, it can identify automatically whether the user is a human or a robot program [3]. Hence, CAPTCHA designers should consider two basic aspects – robustness and usability – and seek a balance between them [4]. The robustness is the CAPTCHA's resistance against malicious attackers, and this has attracted a considerable attention in the research community (e.g., [5–8]), while the usability is the ease with which the CAPTCHA can be solved by a human [9]. However, there have been few studies evaluating the usability aspect (e.g., [3,9–12]).

Generally, there are five main categories of CAPTCHA: text, audio, image, puzzle and video [13]. The text-based is the most widely deployed and has advantages over other categories, in that a computer system can generate an unlimited number of texts at low cost without wasting web resources [2,3]. Most researchers have developed and improved their text CAPTCHAs based on the Latin script. Although an English CAPTCHA has several benefits, it also has several vulnerabilities and weaknesses, as reported in [2] and [8]. Moreover, a user who does not know English might find it difficult [9,13]. Therefore, one of the solutions is to use alternatives to the Latin script, such as the Arabic script [7,10].

In recent years, the internet usage in Arabic countries has increased as the population has grown, with 180 million individuals in 2020 living in Arabic-speaking countries, according to Internet World Stats [14]. Besides, a number of websites provides services in the Arabic language. Moreover, it is not only Arabic that uses the Arabic script, but also other languages, including Persian, Malay and Urdu. Thus, applying the Arabic script allows many users to interact with CAPTCHAs [7]. From a security point of view, the recognition of Arabic script is more difficult, as it supports different fonts and each of them differs in the design of the letters' strokes and ligatures [15]. Although of this advantage, it may be the target of various types of attack, such as artificial intelligence attacks [2].

Due to the accelerated development of artificial intelligence techniques, several CAPTCHA features against such attacks have become ineffective. However, these techniques have reduced the gap between people and machine in terms of the problem-solving process.

This paper first analyses the current Arabic text-based CAPTCHA schemes at a parameter level by conducting an experimental study. Specifically, we evaluate five different studies and websites that generate Arabic text-based CAPTCHAs. To obtain statistically meaningful results, we developed an online application to conduct this study on a diverse enough scale to explore a large interaction space. Accordingly, more than 11,000 CAPTCHAs have been solved by users through this application, and the solving time and accuracy for each CAPTCHA have been recorded. The results showed some features that influence the time and accuracy of solving CAPTCHA schemes. For instance, the cursive style has a higher solution speed than separate letters, and the position of the lines affects human performance more than the number of lines.

Based on the achieved results, and given that there are shortcomings in machine learning concerning human ability, we exploited this to generate Arabic text-based CAPTCHA schemes by applying adversarial examples to the generated scheme. Furthermore, we evaluated the security of the generated scheme in two different ways. Finally, we conducted a usability experiment to evaluate the performance of the generated scheme. The results of these evaluations demonstrated that perturbations injected into Arabic text-based CAPTCHAs can preserve the usability and improve the security of traditional text-based CAPTCHA.

The structure of this paper is organized as follows: Section 2 reviews the previous works. In Section 3, we explain the methodology. Section 4 presents and discusses the results. Finally, Section 5 concludes the paper.

2 Related Works

This section discusses the studies based on Arabic text-based CAPTCHAs. Then, a number of parametric studies based on Latin text-based CAPTCHAs are reviewed. Also, the adversarial CAPTCHA generation is discussed.

2.1 Arabic Text-Based CAPTCHA

Very few studies have been carried out on Arabic text-based CAPTCHAs [7]. Generally, most studies have been done on Arabic text-based CAPTCHAs either as typed or handwritten schemes. Although Arabic text-based handwritten schemes are beyond the scope of our paper, we expand our coverage to this area since only very few works have been reported. Accordingly, the following sections address works that have been reported on Arabic text-based handwritten and printed CAPTCHAs, respectively.

2.1.1 Arabic Handwritten Text-Based CAPTCHAs

The first study on generating Arabic handwritten CAPTCHAs is introduced by Alsuhibany and Parvez in [16]. In particular, they developed a novel approach by exploiting some Optical Character Recognition (OCR) operations on Arabic sub-words. The results of this study indicated that the developed approach has increased the difficulty of a lexicon-based attack, as well as achieved a high accuracy in terms of usability. Moreover, Aldosari and Al-Daraiseh in [17] proposed a novel multilingual handwritten text-based CAPTCHA and Arabic was one of these adopted languages.

2.1.2 Arabic Printed Text-Based CAPTCHAs

The first Arabic printed text-based CAPTCHA was proposed by Shahreza et al. in [18], which created a scheme consisting of Arabic/Persian letters. This study is extended in [15] by containing meaningless words and added lines to distort the image background to further complicate character recognition for machines. Besides, Shahreza et al. in [19] applied random meaningless Persian words to improve the resistance of Nastaliq CAPTCHA against attacks. Yaghmaee et al. in [20] proposed three different types of schemes using special features of the Persian alphabet which can be used efficiently in the Arabic language too. Furthermore, Shahreza in [21] applied typed-text Persian CAPTCHA for recognizing spam SMS messages. Khan et al. [22] proposed a method of embedding Arabic text for CAPTCHA by extending the work in [15]. The robustness level of this method is evaluated against OCR engines.

There are also applications and websites that provide Arabic typed-text CAPTCHA, such as the BotDetect CAPTCHA generator [23] that presents distinct schemes and languages including Arabic. Additionally, ArCaptcha [24] is an Arabic scheme application that shows random noises with a fixed number of separate characters.

Alsuhibany et al. in [7] evaluated the robustness of several Arabic printed text-based CAPTCHAs. The results of this evaluation demonstrated that the existing schemes and websites that rely on Arabic CAPTCHA are vulnerable to automated attacks.

2.2 Latin Text-Based CAPTCHA

In this section, English/Latin CAPTCHA parameters are defined by reviewing the studies that showed the impact of these parameters on the usability aspect. Also, the adversarial CAPTCHA scheme is discussed.

2.2.1 Parametric Studies of Text-Based CAPTCHAs

Chellapilla et al. in [1] carried out a study to examine the effectiveness of specific parameters in a laboratory setting on 76 participants. Also, the study in Bursztein et al. [25] presented the closest to our methodology, whereby the authors evaluated a selected comprehensive list of security features, then designed new CAPTCHA schemes. This study examined the impact of features in isolation and systematically analyzed how the feature combination affected the accuracy and solving time.

2.2.2 Adversarial Text-Based CAPTCHAs

This section gives an overview of adversarial examples and reviews adversarial text-based CAPTCHA generators.

The neural networks algorithm has achieved an excellent performance in classifying different types of data such as images. However, a recent work in [26] has discovered that a small and specific perturbation added to the categorized items is intended to misclassify the machine learning model. This may be imperceptible to the human eye that called adversarial examples. Thus, the objective of the adversary is misclassifying or reducing the confidence of the target model prediction by affecting the integrity of the classifier output. For example, it changes the classification of one of the input items to any class that differs from the original category [27].

Besides, adversarial samples have the property of transferability, in that adversarial samples created from a specific model can affect another model, even if they have different architectures [28]. There are several algorithms for generating adversarial examples. For instance, the Fast Gradient Sign Method (FGSM) proposed in [29], and the Universal Adversarial Perturbations Method (UAPM) developed in [30]. Adversarial examples are an interesting phenomenon in terms of improving the robustness of neural networks, which has been an active area of research, as they cause security threats for some intelligent devices [31]. However, the current methods can mitigate the attack only to some extent by retraining neural networks directly on adversarial samples, which is a simple and effective way for a model to learn to classify them correctly [28].

Adversarial examples can be categorized into two classes of threat model: black-box and white-box attacks. White-box attacks have complete knowledge of the model, while black-box attacks have no or little knowledge of the model, and can be further classified into adaptive and non-adaptive. Attackers try several sophisticated methods against adversarial examples, while being aware of potential defenses. There are types of defensive techniques against an adaptive attack, such as ensemble adversarial training, that can be adopted to improve their attacks, unlike non-adaptive attack, in which the crafting adversarial samples are straightforward [27,28].

Recently, adversarial examples of machine learning models have been a research-challenging area of CAPTCHAs. Osadchy et al. in [32] suggested that adding adversarial noise resists removal attempts on samples. Accordingly, they evaluated the scheme in terms of usability and security, with the results showing that this scheme offers a high level of security and good usability compared to current CAPTCHAs. In [31], the authors mainly focused on testing the effect of adversarial examples on three types of CAPTCHA, including the text-based type. The result showed that adversarial examples have a positive impact on the robustness level. Moreover, Shi et al. in [28] proposed a framework for CAPTCHA generation that integrates different image pre-processing techniques, methods and attacks. The results of this study showed that the adversarial approach can improve the CAPTCHA security while maintaining usability.

Therefore, too little attention has been given to Arabic CAPTCHAs. To the best of our knowledge, we are the first to address studying Arabic CAPTCHAs in terms of the parameter level as well as introducing adversarial Arabic examples.

3 Methodology

This section describes the methodology of our study. In particular, it is divided into three consecutive steps. The first step is conducting an experimental study to evaluate the usability of the state-of-the-art Arabic CAPTCHA schemes, since the robustness has been studied in [7]. The second step is deriving and designing a CAPTCHA model based on the results of step one. Then, the robustness and the usability effectiveness of the derived scheme are tested. Each step will be described in the following sections.

3.1 Evaluating the State-of-the-Art

This evaluation aims to measure the usability of existing Arabic CAPTCHA schemes. Specifically, it is to understand how parameterizations affect the completion time and human solving accuracy by conducting an experimental study. The next section explains the setup and the procedure of the experiment.

3.1.1 Experimental Setup

The experiment involves subjects solving a set of Arabic text-based CAPTCHAs that have been collected from several sources. In this section, we describe our targeted schemes, participants, and the developed system.

Targeted schemes. So far, we have observed five different Arabic CAPTCHA sources that have been proposed by three research papers [15,20] and [22] and two websites [23,24]. However, we have excluded some of the papers' schemes reviewed in Section 2, which used special Persian letters that are not included in the Arabic letters (e.g., [19]).

The first targeted scheme is the proposal for Persian/Arabic CAPTCHA in [15], and we found only one sample available. This scheme used six letters to form a meaningless word in a cursive style. Also, it uses a background color that differs from the foreground, in addition to several straight lines with a distinct color. It is important to note that this scheme was not evaluated previously in terms of the usability aspect.

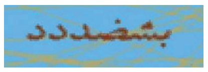
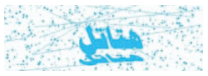
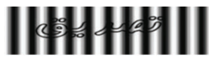
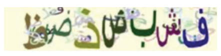
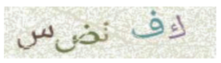
The second scheme [22] applied some parameters that differ according to the security level. These parameters are several font sizes with different types, as well as different distortion types added, such as random lines and dots. We selected eight samples that fit our experiment, such as those that do not contain explanatory symbols and are human-readable.

The third scheme [20] presented three different styles, but we tested only two of them that contain only Arabic letters. We chose one sample of the first style that appeared as a meaningless word, and four samples from the second style that mainly consisted of Persian/Arabic words. Further, it used a special pattern with dark, bright bands in the background. Accordingly, the total of the samples collected from this scheme is five.

The fourth scheme is the BotDetect website, which generates separate Arabic characters as CAPTCHA text with different styles [23]. To support our experiment, we have selected an arbitrary eight samples to be evaluated.

The last targeted scheme is ArCaptcha [24], which applied salt and pepper features. For the usability evaluation step, we have collected eight different samples that consist of meaningless-content separated characters with random font size from 14 to 24. Tab. 1 shows a sample from each evaluated scheme.

Table 1: Arabic text-based sources and examples

Scheme	Source type	Example	Reference
1	Research paper		[15]
2	Research paper		[22]
3	Research paper		[20]
4	Website		[23]
5	Website		[24]

Participants. We recruited 382 participants. However, due to unwanted or incorrect data such as writing an English letter mistakenly, 371 participants were involved in this experiment, of differing gender, ages, level of education and technical skills. The distribution is shown in Tab. 2.

System. In order to allow as many users as possible to solve our targeted samples, a system was developed using the Google App Script platform, which enables users to build and publish a website

[33]. The system consisted of 33 pages, the first page being for the participant's demographics. The second page includes a set of instructions in addition to an illustrative example. Then, the experiment has 30 Arabic text-based CAPTCHA images in which the user is asked to recognize and type the shown characters of each image, sequentially. Note that each single sample was displayed in a separate web interface to simulate a real CAPTCHA service. Importantly, the participant cannot move to the next page if the input field is empty. Moreover, the system does not give any hints about the solution, nor whether the typed letters are correct or not. Furthermore, the user is able to estimate his or her solving progress by looking at a counter that illustrates the number of challenges that have been solved so far and how many are still remaining. Once all CAPTCHAs are solved, a thanks page will appear stating that the task has been completed, and this page is the last page.

Table 2: Participants' information in the usability evaluation

Gender	Female			Male		
	194			177		
Age	[< 16]	[16–25]	[26–35]	[36–45]	[> 45]	
	14	71	150	88	48	
Educational level	Intermediate school	High school	Diploma	B.S.	M.S.	Ph.D.
	30	39	41	219	36	6
Technical background	Yes			No		
	308			63		

3.1.2 Procedure

In this section, we explain how the experiment was run in terms of the participants' instructions, and the data gathered.

At the beginning, we offered a survey asking about demographic information. Then, each participant was informed that 30 Arabic text-based CAPTCHAs would be displayed sequentially that were collected from previous studies or websites, as discussed in the previous section. Also, there was an example in which the user is instructed to type the shown characters of this example. Once the user moves to the first page of solving the shown Arabic CAPTCHA, he is then asked to recognize and write the letters as shown and send them by clicking on the submit button. Finally, the participant is instructed to avoid any distractions and to focus on the task.

The data analysis in this study was for 371 participants. Moreover, the characters of each submitted test and the time taken to solve these tests are recorded by the system; consequently, the total number of CAPTCHA texts is 11,130.

3.1.3 Experimental Results

This section presents the evaluation result of measuring the accuracy and the solving time. We also present the results of the features individually.

For the accuracy, we have compared the submitted answers by participants with the original text. Although we have ignored the spacing between texts like the ArCaptcha website [24], the accuracy of the BotDetect website [23] was affected when we added them. Based on the results gleaned during this step, Tab. 3 demonstrates the usability issues that have been observed.

Table 3: The usability issues in our targeted schemes

	Usability issues
Scheme [15]	• Repeating the same letter more than twice leads to ignoring the rest or adding more.
Scheme [22]	• Using confusing letters like أ and ؤ. • Applying similar consecutive letters (e.g., ي and ب) leads to three confusion states appearing: switching between letters, ignoring one of them or writing both positions as the same letter. • Presenting inappropriate fonts, for example PT Bold Mirror, which causes the word to be typed twice.
Scheme [20]	• Appling similar consecutive characters such as ق and ـف. • Offering broad lines in an inappropriate place. For instance, a separate letter which leads to a connected character like ـد is written as ـحـ.
Scheme [23]	• Adding distortion is similar to the Arabic diacritics, such as in mass, dots and bubble styles. • Some letters that consist of two parts, such as ك, are not suitable for a Black Circles style, and the reason is that people do not recognize the different parts, like Hamza. Some characters are incorrectly positioned, causing them to appear as another letter. For example, if the character ر came in the upper left of the image, users thought it was ب or ل.
Scheme [24]	• Appling confusing characters such as ـهـ and أ. • Utilizing random lines that pass through the letters at an improper location.

The solving time was measured by recording the time from when a sample was shown until the user submitted the answer. Fig. 1 gives an overall view of the solving times for all schemes. Specifically, the fourth scheme takes the longest time, with an average of 25 s. The fastest was the second scheme, with an average time of around 15 s.

Character sets. The most accurate models which took the least time to solve contained meaningful words. However, this feature has threatened the robustness of the scheme, as mentioned in Kaur et al. [34]. Luckily, the difference in solving time between meaningful and meaningless was slight.

Certainly, confusing characters have reduced the accuracy of the solution and increased the time the user spends on solving the CAPTCHA test. Hence, there is a relationship between the accuracy, time to solve, and confusing characters. However, a study in Khan et al. [22] did not consider all confusing characters; therefore we have observed new confusing characters, as shown in Tab. 4.

As mentioned in Alsuhibany [35], some English characters were affected by increasing the distortion, and thus led to mis-writing. It also happened with Arabic characters, such as ذض or زض, which can resemble تض ,نض or لض, and these letters (س، ش، ص) also have this issue.

Cursive style. Since Arabic letters connect in writing to form a word, we can divide the printed Arabic text-based CAPTCHAs into two categories: connected characters called cursive style, and separate characters like English letters. For cursive schemes, users are slightly faster and substantially more accurate. Fig. 2a shows the relationship between accuracy and continuous (i.e., cursive style) versus separate letters and Fig. 2b illustrates the time needed to solve each category.

The reasons for the high accuracy in the cursive schemes are explained as follows. These schemes contain neither collapse nor overlapping as distortion. Also, some of these schemes contain meaningful words which had the highest accuracy. In terms of time taken in the non-cursive schemes, people were divided into three groups according to their way of writing texts. The first group wrote a space between all the letters. The second group wrote all the letters continuously. The last group wrote the letters between collapses or overlaps in a continuous manner as in CAPTCHA texts, and the rest separately. Therefore, adding spaces in texts increase the solution time, and may perhaps be the reason why more time is spent in solving this type.

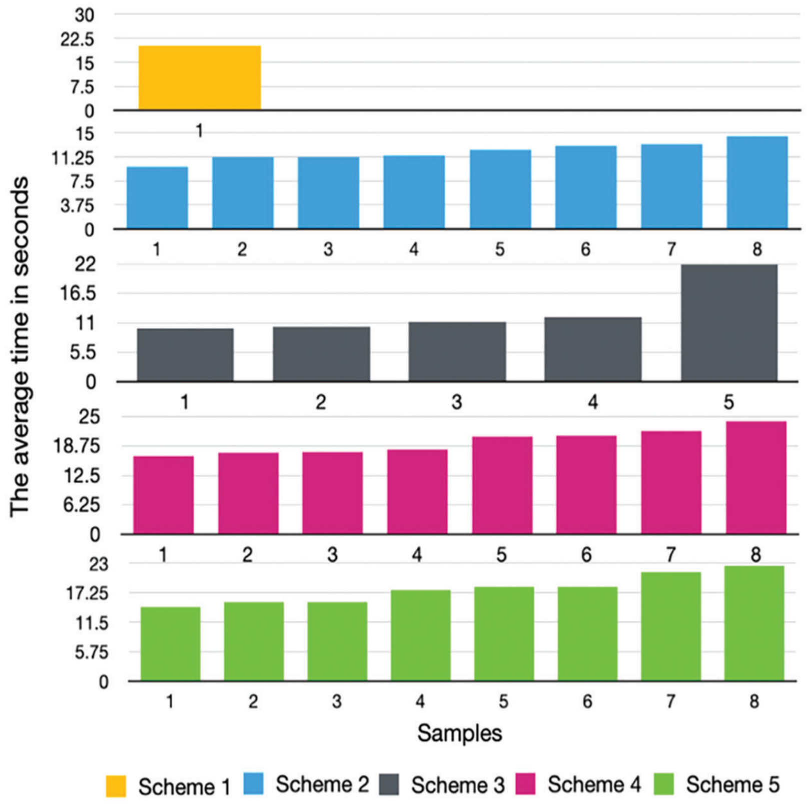
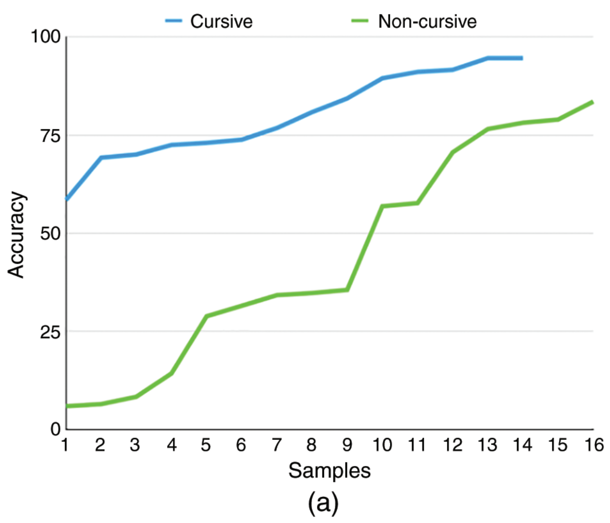


Figure 1: Average solving time of schemes

Table 4: Observed confusing characters

Original letter	Replaced by	Or	Original letter	Replaced by
أ	ا	ر	ر	ز
ث	ت	ن	ش	س
خ	ج	ح	ص	ض
د	ذ	ح	ض	ص
ذ	ل	د	ط	ظ
ذ	ن	ت	ف	ق
ز	ذ	ر	ق	ف
ت	ن		ك	ل
ح	خ		م	ج
ؤ	ق		ه	هـ



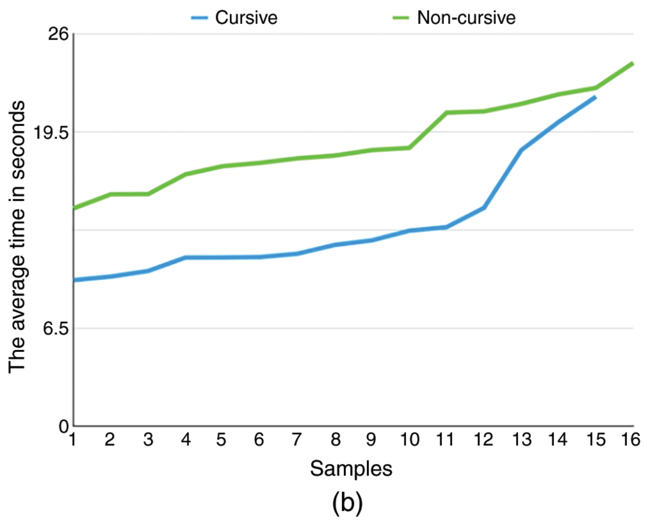


Figure 2: Accuracy and solving time *vs*. cursive and non-cursive schemes (a) illustrates the relationship between accuracy and cursive style versus separate letters, while (b) illustrates the time needed to solve each category

Number of characters. We could not determine precisely the effect of the number of characters on usability due to the detection of confusing new characters. The appropriate length of a CAPTCHA text should not exceed nine characters in order not to affect the usability, but must be no fewer than four characters in order not to become susceptible to attack. Consequently, depending on the required security level, CAPTCHA designers can adjust the length of characters without harming users' accuracy unless there are confusing characters. However, there will be additional time demands.

Lines. Fig. 3 demonstrates the relationship between accuracy and solution time with a number of random lines in the background. We observed that humans are affected by wavy lines more than straight lines, and the number of straight lines did not hurt individual performance directly. Moreover, the position of lines affects accuracy more than the number of lines. For instance, a random line that crosses a letter can cause an incorrect answer, such as where a line passes through the top of the character ر, and the user sees the upper part as a point, then writes it as ز, or vice versa. Furthermore, a line crossing over one of the main character points and covering it may cause the user to fail to recognize the correct character or write it as a diacritic; for example, ث can resemble ن ,ت or تَ.

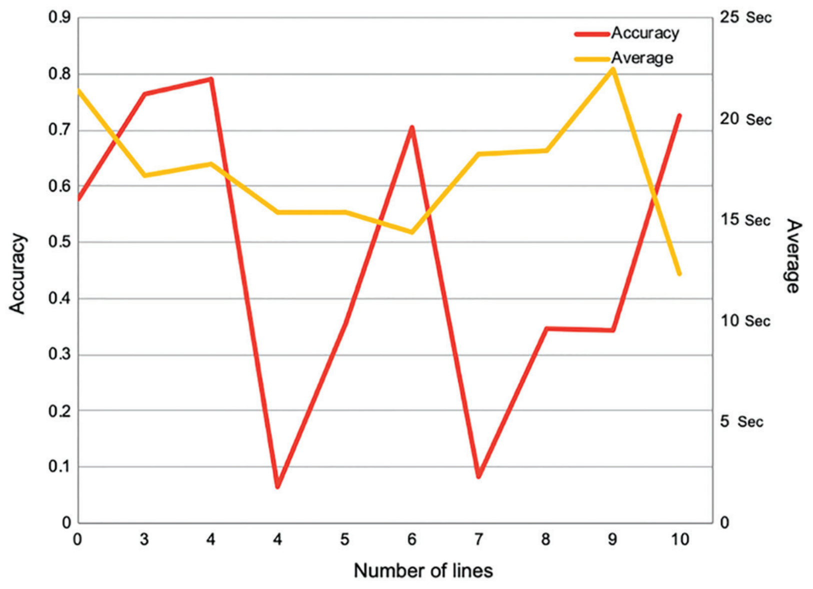


Figure 3: Accuracy and solving time *vs*. the number of lines

Font size. Fig. 4 shows that the font size had a very minimal effect on the solving time. Therefore, although the font size did not impact on the usability aspect, this might need more investigation and would be the subject of one of our future works.

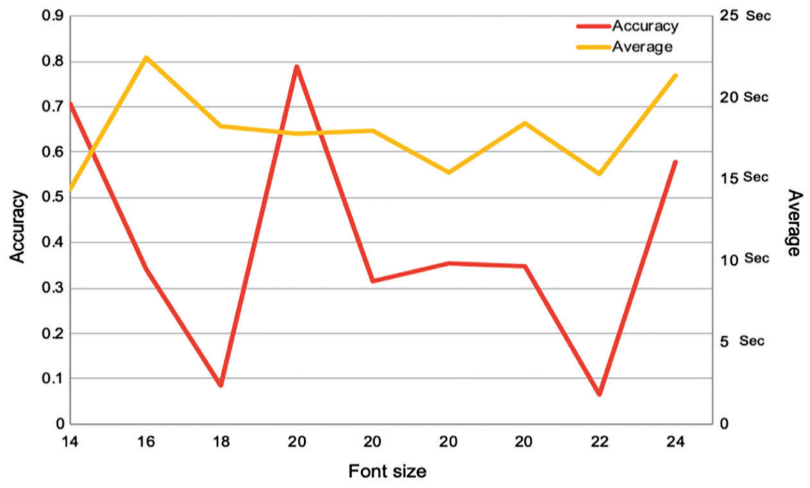


Figure 4: Accuracy and solving time *vs*. font size

Dots and bubbles. Several Arabic letters contain dots and differ according to the number and position of points. Unlike English, the letters that have dots are only i and j, as shown in Tab. 5. Accordingly, the results showed that human performance suffers when random dots are added as a noise in the background. In particular, the worst accuracy of a solution we received was 5.91%, which was from a sample that contained a bubble, leading to a confusing character. Also, very few people (31.54%) correctly answered a dots model. This may be due to random dots having the same size, shape and color as the main character's points.

Table 5: Letters that have dots in Arabic and English languages

	Number	Samples
Arabic characters	15	ق ن، ي ب، ت، ث، ج، خ، ذ، ز، ش، ض، ظ، غ،ف،
English characters	2	i, j

3.2 The Derived Scheme: Arabic Adversarial CAPTCHA

Based on the results in the previous section, this section presents the system architecture of the proposed Arabic Adversarial CAPTCHA Generator (AACAPTCHA), as shown in Fig. 5. The detail of this system is described in the following.

First of all, we needed a dataset in order to generate adversarial samples. Thus, we collected 3,606 samples from two different schemes: BotDetect [23] and ArCaptcha [24]. Then, pre-processing was applied to these samples using CAPTCHA Solving Software (GSA Captcha Breaker), which is a commercial software [36]. Afterwards, the characters were segmented using a segmentation algorithm developed in [7]. Consequently, we used a dataset containing 15,400 hand-labelled letters from 28 categories with approximately 550 items from each category.

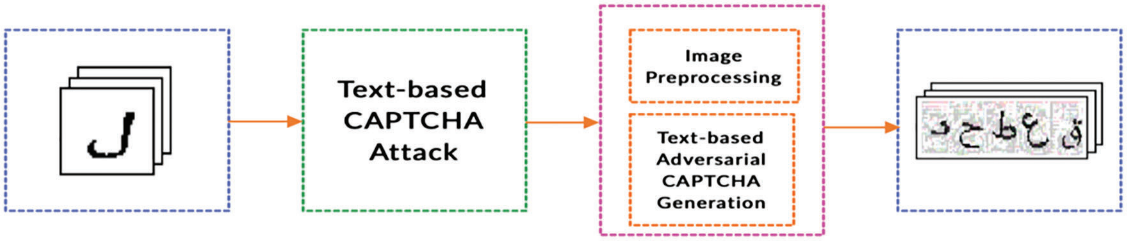

Text-based CAPTCHA Attack
Image Preprocessing
Text-based Adversarial CAPTCHA Generation

Figure 5: Overview of AACAPTCHA system

Then, we implemented a Convolutional Neural Network (CNN) machine learning algorithm as text-based CAPTCHA attacks. It has two functions in our system which provides essential model information for generating adversarial samples and is utilized to evaluate the robustness of CAPTCHAs against real attacks. Accordingly, we applied this algorithm in two phases: without image processing and with implementing a median filter as an image pre-processing technique for CAPTCHA security to remove the noise in an image, which is used in [32]. Thus, we used FGSM, which is widely used and computationally efficient to generate adversarial examples which it calculates the noise as follows Eq. (1):

$$x' = P(x + \epsilon \times sign(\nabla x \ J(x,y))) \tag{1}$$

where x is the original clean image, ε is a constant that controls the amount of noise, J is the cost used to train the neural network and y is the label for x [29]. The last step is generating CAPTCHA samples, which randomly contain 4–9 characters. It is important to note that we have taken into consideration here in the generation step the rules that were obtained in the previous experiment (discussed in Section 3.1).

The adversarial with epsilon 0.3 can fool a neural network as it reduces accuracy from 95.12% to 3.04%. Moreover, the noise can be removed by pre-processing, where the model becomes more powerful along with image filtering, as the correct recognition rate is increased to 4.78%. Finally, to overcome the challenges facing samples against image processing, we apply a median filter that fits this algorithm during the generation process as presented in [32] in order for the noise to be resistant to the filter attacks. As a result, the success rate is reduced to 0.33%.

3.3 Evaluating the Derived Scheme

This section presents the evaluation of the proposed AACAPTCHA scheme in terms of security and usability, respectively.

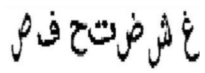
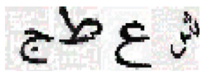
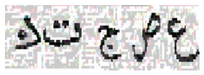
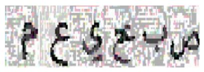
3.3.1 Security evaluation

We utilize two different attack techniques that potential attackers could use against the system. In particular, we evaluate an attack which aims to return the image to its original class by utilizing image-processing techniques such as filters. Ensemble adversarial training is one of the most important options against adversarial CAPTCHAs, and the trained model won the first round of NIPS 2017 in a competition on defenses against adversarial attacks. This method increases the model training data with examples crafted on other pre-trained static models [37].

3.3.2 Usability Evaluation

For the usability aspect, we used a pre-post experimental design [38] for measuring the results before and after embedding perturbation to Arabic letters. Thus, we evaluated 24 samples from four categories, each category having six examples with different CAPTCHA lengths of between 4 and 9 characters, which is the appropriate length range observed in the previous evaluation experiment. These categories differ from each other according to the epsilon value, from 0% (non-adversarial versions) to 30% (as presented in [31]). Tab. 6 shows some examples.

Table 6: Samples used in this usability experiment

Scheme	Epsilon (%)	Example
1	0	
2	10	
3	20	
4	30	

It is worthwhile noting that the same system explained in Section 3.1.1 is used here in this experiment. However, the number of pages is different as we used 28 pages, due to the number of samples, in addition to the introduction, the questionnaire and the thanks pages.

We tested the proposed AACAPTCHA with 413 participants. Participants were requested to provide information about their gender, age, educational level and whether they had a technical background before starting the test. Tab. 7 illustrates the user statistics. The sample yielded 9,912 answered tests (i.e., 413 participants * 24 samples).

Table 7: User statistics in the proposed AACAPTCHA scheme usability experiment

Gender	Female			Male		
	303			110		
Age	[< 16]	[16–25]	[26–35]	[36–45]	[> 45]	
	22	135	105	92	59	
Educational level	Intermediate school	High school	Diploma	B.S.	M.S.	Ph.D.
	25	49	49	244	38	8
Technical background	Yes			No		
	331			82		

4 Results and Discussion

In our security experiment, we used filter attacks that aimed to remove samples' noise generated by the FGS method. In particular, we applied four filters on a set of 5,083 adversarial examples, representing 30% of our dataset. These filters are: Median, Gaussian, Minimum and Maximum. The accuracy rate of each filter is presented in Tab. 8. Therefore, the success rate did not exceed 5%.

Table 8: Filters examined in the filter attacks

Filter type	Median	Gaussian	Minimum	Maximum
Accuracy rate	0.11%	3.24%	4.18%	2.15%

For ensemble adversarial training, we used three different models to generate adversarial examples and train an ensemble model. Thus, if we did not craft examples, then the accuracy was 35.43%; otherwise, the accuracy was 41.51%. Fortunately, it is difficult for attackers to discover the models or methods applied in generating an adversarial CAPTCHA, which indicates an interesting reduction in the practical impact of this training.

Based on the collected data, Tab. 9 shows the main results of the participants' performance in the schemes. Average accuracy measures the average successful solution to the challenges' schemes and average time estimates the average time required for all the users to complete the corresponding tasks.

Table 9: Usability results of the proposed AACAPTCHA

	Scheme 1	Scheme 2	Scheme 3	Scheme 4
Average time	16.30 s	15.07 s	15.98 s	18.17 s
Accuracy average	383.83	384.5	377.5	337.17
Accuracy rate	92.94%	93.10%	91.40%	81.64%

We observed that the average time for all classes converges, with a slight difference. Although the time for solution was not entirely accurate, as there are differences between them, fortunately the variance was not large. The lowest average time was epsilon 10%, while the highest was epsilon 30%. Also, we got the same results in accuracy, with a success rate of 93.10% and 81.64%, respectively. As expected, the solving speed and difficulty of CAPTCHAs deteriorated when the epsilon value was increased; this obviously impacts on the human solvability. However, it takes more time for machine learning recognition. Hence, this implies that there is a trade-off between security and usability.

After completing the given tasks, we asked the participants three questions about AACAPTCHA. The first question was about the level of readability of the scheme containing perturbation characters. The second question was whether it took time for them to recognize letters in the proposed scheme. Finally, we asked them whether the proposed scheme affected whether they could write letters correctly or not. As a result, 65.4% of the participants found the proposed scheme easy and 20.1% found it very easy. This indicates that the adversarial example does not make a text-based CAPTCHA more difficult from the users' point of view. Also, 41.9% said that the proposed scheme did not take much time to solve, while 46.5% answered the opposite, with the rest neutral. Moreover, 46.5% of solvers replied that adding a perturbation would affect their accuracy of solution. However, by looking at Tab. 9, we can see that there is no difference between schemes 2 and 3 and the original samples in both accuracy and speed of solution, though the second scheme surpasses the third one a little bit in the usability aspect. This may be a good indication for applying the adversarial example on Arabic text-based CAPTCHAs.

Finally, AACAPTCHA can be seamlessly integrated with current text-based CAPTCHA systems. However, it might be interesting to investigate applying more algorithms for generating adversarial examples.

5 Conclusion and Future Works

As a first study to support the usability evaluation of Arabic schemes, we have conducted a comprehensive experiment including all existing Arabic text-based CAPTCHAs. The experiment verified that the human solvability of these schemes demonstrated how each feature varies and how this can impact on the solution time and accuracy. Based on the results of this experiment, we studied how to design an Arabic adversarial CAPTCHA scheme, which is the first attempt in this field. This scheme is evaluated in terms of security and usability aspects. The results of this evaluation were very encouraging.

Since the results showed that the adversarial technique increases the security of regular Arabic text-based CAPTCHA, an extensive evaluation in terms of both security and usability will be carried out as one of our future works.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] K. Chellapilla, K. Larson, P. Simard and M. Czerwinski, “Designing human friendly human interaction proofs (HIPs),” in *ACM Conf. on Human Factors in Computing Systems - CHI 2005*, Portland, Oregon, USA, pp. 711–720, 2005.

[2] M. Moradi and M. Keyvanpour, “CAPTCHA and its alternatives: A review,” *Security and Communication Networks*, vol. 8, no. 12, pp. 2135–2156, 2015.

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

[4] K. A. Kluever and R. Zanibbi, “Balancing usability and security in a video CAPTCHA,” in *SOUPS, 2009 - Proc. of the 5th Sym. On Usable Privacy and Security*, CA, USA, 2009.

[5] J. Yan and A. Salah, “CAPTCHA security a case study,” *IEEE Security & Privacy Magazine*, vol. 7, no. 4, pp. 22–28, 2009.

[6] R. Hussain, H. Gao, R. A. Shaikh and S. P. Soomro, “Recognition based segmentation of connected characters in text based CAPTCHAs,” in *Proc. 2016 8th IEEE Int. Conf. on Communication Software and Networks. ICCSN 2016*, Beijing, China, pp. 673–676, 2016.

[7] S. A. Alsuhibany, M. T. Parvez, N. Alrobah, F. Almohaimeed and S. Alduayji, “Evaluating robustness of Arabic CAPTCHAs,” in *2017 2nd Int. Conf. Anti-Cyber Crimes, ICACC 2017*, Abha, Saudi Arabia, pp. 81–86, 2017.

[8] C. Tangmanee, “Effects of text rotation, string length, and letter format on text-based CAPTCHA robustness,” *Journal of Applied Security Research*, vol. 11, no. 3, pp. 349–361, 2016.

[9] J. Yan and A. S. El Ahmad, “Usability of CAPTCHAs or usability issues in CAPTCHA design,” in *Proc. SOUPS 2008. 4th Symp. Usable Privacy and Security*, Pittsburgh, PA, USA, pp. 44–55, 2008.

[10] C. A. F. Avouris, N. M. Avouris and A. G. Voyiatzis, “On the necessity of user-friendly CAPTCHA,” in *Proc. Int. Conf. on Human Factors in Computing Systems*, Vancouver, BC, Canada, pp. 2623–2626, 2011.

[11] S. I. Swaid, “Usability measures of text-based CAPTCHA,” *Journal of the Academic Business World*, vol. 7, no. 2, pp. 63–66, 2013.

[12] D. Brodić and A. Amelio, “Exploring the usability of the text-based CAPTCHA on tablet computers,” *Connection Science*, vol. 31, no. 4, pp. 430–444, 2019.

[13] W. K. Abdullah Hasan, “A survey of current research on CAPTCHA,” *International Journal of Computer Science & Engineering Survey*, vol. 7, no. 3, pp. 1–21, 2016.

[14] Internet World Stats, Internet Usage in the Middle East, 2019. [Online]. Available: https://web.archive.org/web/20190516191014/, https://www.internetworldstats.com/stats5.htm.

[15] M. H. Shirali-Shahreza and M. Shirali-Shahreza, “Persian/Arabic baffletext CAPTCHA,” *Journal of Universal Computer Science*, vol. 12, no. 12, pp. 1783–1796, 2006.

[16] S. A. Alsuhibany and M. T. Parvez, “Secure Arabic handwritten CAPTCHA generation using OCR operations,” in *Proc. Int. Conf. on Frontiers in Handwriting Recognition ICFHR*, Shenzhen, China, vol. 0, pp. 126–131, 2016.

[17] M. H. Aldosari and A. A. Al-Daraiseh, “Strong multilingual CAPTCHA based on handwritten characters,” in *2016 7th Int. Conf. on Information and Communication Systems ICICS 2016*, Irbid, Jordan, pp. 239–245, 2016.

[18] M. Hassan and S. C. Engineering, “Persian/Arabic Captcha,” *International Journal on Computer Science and Information Systems*, vol. 1, no. 2, pp. 63–75, 2006.

[19] M. H. S. Shahreza and M. S. Shahreza, “Advanced nastaliq CAPTCHA,” in *2008 7th Int. Conf. on Cybernetic Intelligent Systems CIS 2008*, London, UK, pp. 1–3, 2008.

[20] F. Yaghmaee and M. Kamyar, “Introducing new trends for persian captcha”, *Journal of Electrical and Computer Engineering Innovations*, vol. 4, no. 2, pp. 119–126, 2016.

[21] M. S. Shahreza, “Verifying spam SMS by Arabic CAPTCHA,” in *2nd IEEE Int. Conf. of Information Communication Technology*, Damascus, Syria, pp. 78–83, 2006.

[22] B. Khan, K. Alghathbar, M. K. Khan, A. AlKelabi and A. AlAjaji, “Cyber security using Arabic CAPTCHA scheme,” *International Arab Journal of Information Technology*, vol. 10, no. 1, pp. 76–85, 2013.

[23] Captchacom, “BotDetect CAPTCHA demo – features,” 2021. [Online]. Available: https://captcha.com/demos/features/captcha-demo.aspx.

[24] M. Anini, “عربية كابتشا - عركابتشا | Arabic CAPTCHA,” 2021. [Online]. Available: http://arcaptcha.anini.me.

[25] E. Bursztein, A. Moscicki, C. Fabry, S. Bethard, J. C. Mitchell *et al.,* “Easy does it: More usable CAPTCHAs,” in *Proc. Conf. on Human Factors in Computing Systems*, Ontario, Canada, pp. 2637–2646, 2014.

[26] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan *et al.,* “Intriguing properties of neural networks,” in *2nd Int. Conf. on Learning Representations ICLR 2014*, Banff, Canada, pp. 1–10, 2014.

[27] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay and D. Mukhopadhyay, “Adversarial attacks and defences: A survey”, vol. 6, pp. 1–31, 2018.

[28] C. Shi, X. Xu, S. Ji, K. Bu, J. Chen *et al.,* “Adversarial CAPTCHAs,” *IEEE Transactions on Cybernetics (Early Access)*, pp. 1–16, 2019.

[29] I. J. Goodfellow, J. Shlens and C. Szegedy, “Explaining and harnessing adversarial examples,” in *3rd Int. Conf. on Learning Representations ICLR 2015*, San Diego, CA, USA, pp. 1–11, 2015.

[30] S. M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi and P. Frossard, “Universal adversarial perturbations,” *Proc. 30th IEEE Conf. Computer vis Pattern Recognition CVPR 2017*, vol. 2017-Janua, Honolulu, HI, USA, pp. 86–94, 2017.

[31] Y. Zhang, H. Gao, G. Pei, S. Kang and X. Zhou, “Effect of adversarial examples on the robustness of captcha,” in *Proc. 2018 Int. Conf. on Cyber-Enabled Distributed Computing and Knowledge Discovery*, Zhengzhou, China, pp. 1–10, 2019.

[32] M. Osadchy, J. Hernandez-Castro, S. Gibson, O. Dunkelman and D. Perez-Cabo, “No bot expects the deepcaptcha! introducing immutable adversarial examples, with applications to captcha generation,” *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 11, pp. 2640–2653, 2017.

[33] Google company, “Google Apps Script,” 2021. [Online]. Available: https://developers.google.com/apps-script.

[34] K. Kaur and S. Behal, “Designing a secure text-based CAPTCHA,” *Procedia Computer Science*, vol. 57, no. November, pp. 122–125, 2015.

[35] S. A. Alsuhibany, “Evaluating the usability of optimizing text-based captcha generation,” *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 8, pp. 164–169, 2016.

[36] GSA - Softwareentwicklung und Analytik GmbH, “GSA Captcha Breaker,” 2020. [Online]. Available: https://www.gsa-online.de/product/captcha_breaker/.

[37] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh *et al.,* “Ensemble adversarial training: Attacks and defenses,” in *6th Int. Conf. on Learning Representations ICLR 2018*, Vancouver, Canada, pp. 1–22, 2018.

[38] D. M. Dimitrov and P. D. Rumrill, “Pretest-posttest designs and measurement of change,” *Work-A Journal of Prevention Assessment & Rehabilitation*, vol. 20, no. 2, pp. 159–165, 2003.