

Style Matching CAPTCHA: Match Neural Transferred Styles to Thwart Intelligent Attacks

Palash Ray

Haldia Institute of Technology, Haldia, WB, India

Asish Bera (✉ asish.bera@pilani.bits-pilani.ac.in)

Birla Institute of Technology and Science, Pilani, Pilani Campus, Rajasthan, India

Debasis Giri

Maulana Abul Kalam Azad University of Technology, WB, India.

Debotosh Bhattacharjee



Jadavpur University, WB, India

Research Article

Keywords: CAPTCHA, Convolutional Neural Networks, Deep Learning, Neural Style Transfer, Security, Denoising, Attack

Posted Date: April 5th, 2023

DOI: <https://doi.org/10.21203/rs.3.rs-2769420/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.
[Read Full License](#)

Additional Declarations:

Competing interests: The authors declare no competing interests.

Style Matching CAPTCHA: Match Neural Transferred Styles to Thwart Intelligent Attacks

Palash Ray · Asish Bera · Debasis Giri · Debotosh Bhattacharjee

Received: 7 April 2022 / Accepted: 6 March 2023

Abstract Completely Automated Public Turing Test to Tell Computers and Humans Apart (CAPTCHA) is widely used to prevent malicious automated attacks on various online services. Text- and image-CAPTCHAs have shown broader acceptability due to usability and security factors. However, recent progress in deep learning implies that text-CAPTCHAs can easily be exposed to various fraudulent attacks. Thus, image-CAPTCHAs are getting research attention to enhance usability and security. In this work, the Neural Style Transfer (NST) is adapted for designing an image-CAPTCHA algorithm to enhance security while maintaining human performance. In NST-rendered image-CAPTCHAs, existing methods inquire a user to identify or localize the salient object (*e.g.*, content) which is solvable effortlessly by off-the-shelf intelligent tools. Contrarily, we propose a Style Matching CAPTCHA (SMC) that asks a user to select the style image which is applied in the NST method. A user can solve a random SMC challenge by understanding the semantic correlation between the content and style output as a cue. The performance in solving SMC is evaluated based on the 1368 responses collected from 152 participants through a web-application. The average solving accuracy in three sessions is 95.61%; and the average response time for each challenge per user is 6.52 seconds, respectively. Likewise, a Smartphone Application (SMC-App) is devised using the proposed method. The average solving accuracy through SMC-App is 96.33%, and the average solving time is 5.13 seconds. To evaluate the vulnerability of SMC, deep learning-based attack schemes using Convolutional Neural Networks (CNN), such as ResNet-50 and Inception-v3 are simulated. The average accuracy of attacks considering various studies on SMC using ResNet-50 and Inception-v3 is 37%, which is improved over

existing methods. Moreover, in-depth security analysis, experimental insights, and comparative studies imply the suitability of the proposed SMC.

Keywords CAPTCHA · Convolutional Neural Networks · Deep Learning · Neural Style Transfer · Security · Denoising · Attack.

1 Introduction

CAPTCHA: Completely Automated Public Turing Test to Tell Computers and Humans Apart relies on solving the hard Artificial Intelligence (AI) problem [66]. The hardness of a CAPTCHA entails the difficulty of developing an automated algorithm to solve the query with a higher success rate, *i.e.*, a computer program can solve a challenge correctly with a higher probability. User authentication is validated through a random challenge to access the system in this human-machine interaction. On the contrary, an intelligent automated program, aka *bot*, can crack or bypass the hardness of CAPTCHA and access the system deliberately. Over the decades, several variations of CAPTCHAs (*e.g.*, textual, image, audio, cognitive, adversarial, visual reasoning, etc.) have been developed to thwart different categories of *bot* attacks [1, 7, 15, 30, 37, 57, 63, 68, 70, 75]. State-of-the-art methods have emphasized the security and robustness of underlying algorithms. However, the strengths of CAPTCHA (*e.g.*, text-CAPTCHAs) can easily be undermined by deep neural networks [61, 69, 72, 82]. To overcome the limitations of text-CAPTCHAs, image-based CAPTCHAs are considered as a suitable alternative for further security enhancement [63]. Nevertheless, few methods have applied real-time user verification, or biometric authentication [5], [4] etc. independently, or in conjunction with liveness detection [7, 65], spoofing detection [7], [8], to improve security.

On the other side, style transfer is widely explored by computer vision researchers [33]. The convolutional neural networks (CNN) have shown great success in artistic Neural Style Transfer (NST) between the various classes of content and style images retrospectively [22]. NST is proliferated in diverse and interesting applications such as image super-resolution [18], geometric warping [42], video [52], CAPTCHA [13], image steganography [44], restorable arbitrary NST [43], etc. Recently, a framed-based arbitrary video style transfer method is proposed by aligning cross-domain features with input videos leveraging multi-channel correlation [35].

In general, a content image (I_c) and a style image (I_s) both are fed into a deep network (N) to produce a stylized image (I_t) as an outcome of an NST, which is illustrated at top-row of Fig. 1. A research direction is targeted to enhance the controllability in stylization tasks [11], stability [28], robustness

P. Ray
Department of Computer Science and Engineering, Haldia Institute of Technology, Haldia, WB, India.
E-mail: palash.ray@gmail.com

A. Bera
Department of Computer Science and Information Systems, Birla Institute of Technology and Science, Pilani, Pilani Campus, Rajasthan, India.
E-mail: asish.bera@pilani.bits-pilani.ac.in

D. Giri
Department of Information Technology, Maulana Abul Kalam Azad University of Technology, WB, India.
E-mail: debasis.giri@hotmail.com

D. Bhattacharjee
Department of Computer Science and Engineering, Jadavpur University, WB, India.
E-mail: debotosh@ieee.org

Style Matching CAPTCHA (SMC)

Butterfly Pattern/Style Stylized Butterfly



Click on Correct Pattern From Each Row



TIME 00:05 SUBMIT Refresh

Fig. 1: Style Matching CAPTCHA (SMC) using the Neural Style Transfer (NST). Top: a styled output image of a butterfly leveraging a standard NST method is shown. Bottom: a schematic user interface posing an SMC challenge. A user requires to match (by clicking on the correct style/pattern) with the input style image (style image-grid in the middle) for each of the three rows by understanding the semantic correlation between the input content (Kingfisher) at the left side and the stylized output at the right side.

[71], computational speed-up [34], [11], etc. Another direction investigates the suitability of applying NST in other broader areas [33], [60].

The discrimination between humans and *bots* based on CAPTCHA leveraging NST has been explored in recent times [10, 13]. Substantially, an NST-based technique uses I_t to generate a CAPTCHA challenge Q using an algorithm H . If the given query Q is solved by a human correctly, then the respondent is permitted/accepted, otherwise rejected. The objective remains the same as identifying the object(s) of interest in the query image. The human vision can recognize and solve the challenge despite adversarially perturbed stylized contents, but, it is vulnerable to attacks. Modern optical character recognition (OCR) tools (for text-CAPTCHA), AI-based object detectors (object detection from image-CAPTCHA), and other off-the-shelf sophisticated vision-based tools can undermine the underlying strengths of an NST algorithm easily, shown in Fig. 3.a. Even adding more visual difficulties with balancing the usability, and introducing complex patterns or illumination variations can not hinder the localization of content or main object using the gradient weighted class activation maps (Grad-CAM) [55], shown in Fig. 3.b. Also, object detection in NST using Faster RCNN [51] (in Fig.3. d) Shows a higher success rate. To overcome this issue, we propose that instead of recognizing or locating salient object(s) in the stylized image, correct matching with the corresponding style image might be an alternative solution for NST-based CAPTCHA design.

We have generated a Style Matching CAPTCHA (SMC) underlying NST to enhance security. In existing methods, a user is asked to select the style-transferred regions according to a given description [10, 13, 63]. In contrast, our proposed SMC uses human vision to identify the style I_s used in the generic stylization process along with the content I_c to render a stylized output I_t . A user is requested to select the style I_s by visualizing content I_c and its textural rendering I_t by perceiving the semantic correlation between them. The

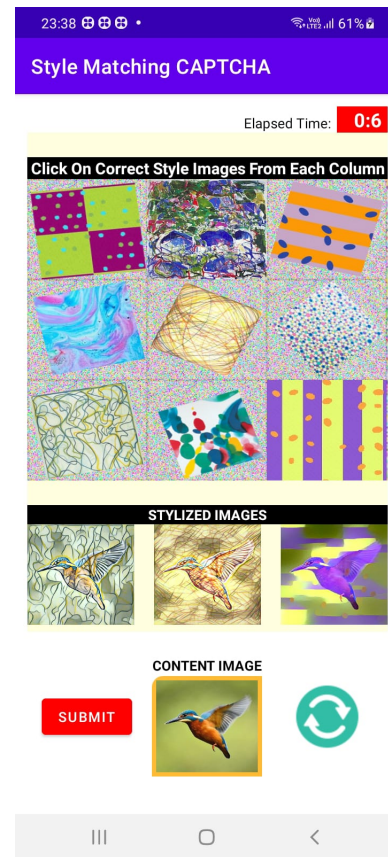


Fig. 2: A Smartphone Application namely, SMC-App is developed for user verification to enhance usability using SMC Algorithm. It follows mainly three easy steps (from bottom to top): a) visualize the content image (bottom-centre); b) understand the NST-rendered three stylized output images (middle); c) apply visual understanding/ cognition to relate and localize the input style images (vertically/column-wise) within 3×3 image-grid.

SMC utilizes human vision in solving a random CAPTCHA challenge Q_{SMC} as a style-matching task. We demonstrate that it could be a difficult problem for the intelligent agents by interchanging the query for style-matching, *i.e.*, match with input style/pattern other than the content/object in focus (see Fig. 1). Interestingly, it is easily solvable by human users, whereas it is still a challenging task for automated tools and simulated attacks.

To guard against *bot* attacks, random source image selection with a broader range of variations (*i.e.*, data augmentation) is followed. Dynamic data augmentation offers additional randomness to design adversarial perturbed content in a stylized representation. In addition, user responses are collected in the same order of appearances of the stylized images according to serial repetition. Correct matching maintains a higher similarity score and correlation between an ordered pair of (I_c, I_t) for neural style matching in SMC. The proposed method is conceptualized in Fig.4.

According to human perception, this challenge can be effortlessly solved by humans, but, difficult for the *bots*. Modern computer programs based on deep learning can recover the content from stylized images (Fig.3.a-b). However, restoring an exact style or pattern from the stylized image is tough for current deep architecture with minimal training samples within a stipulated time, according to the best of our knowledge. We emphasize the limited properties and benefits of the current CAPTCHA design using deep networks. The Grad-CAM [55] can focus on the central part of a pattern where a significant degree of variations are involved (Fig.3.c). Thus, it may not be able to match the correct style from a partial pattern. To explore further in this direction, an attack scheme based on deep learning is simulated using the ResNet-50

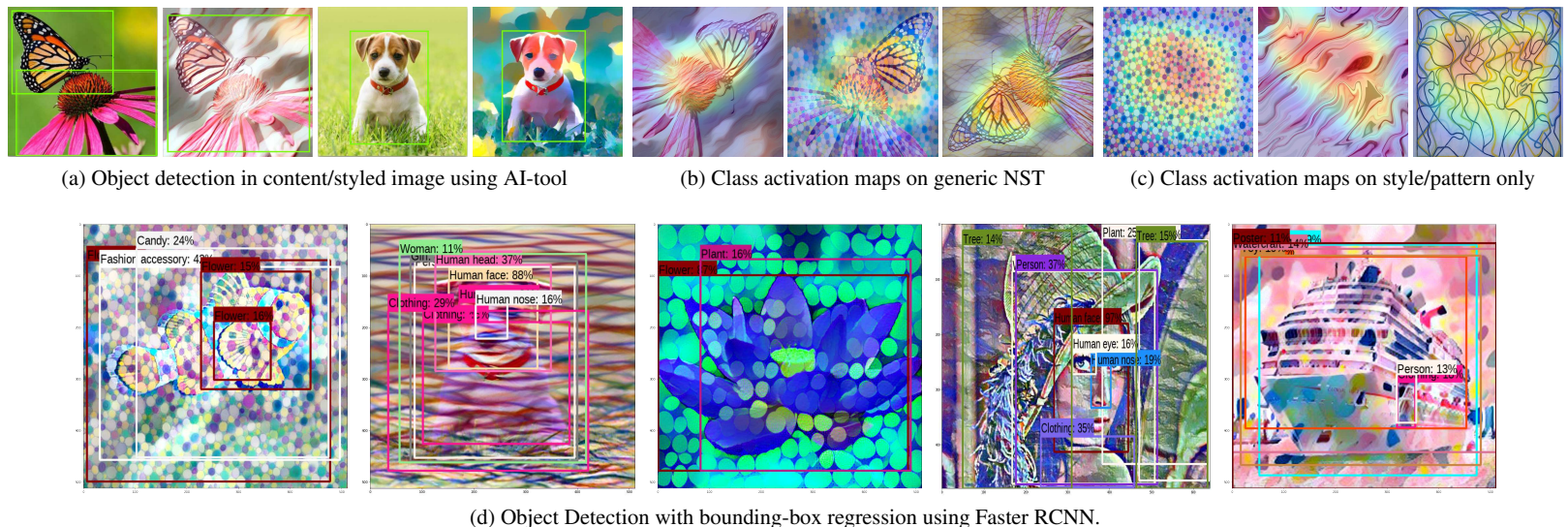


Fig. 3: (a) Success of available AI-based vision tools for object detection (surrounded with green bounding-boxes) from images which leads to a major limitation of current NST-based CAPTCHA by localizing the salient object(s) in the challenge. (b) The addition of random adversarial noise/complex distortion, and transformation/data augmentation have a futile effect of identifying the key content/object using gradient-weighted class activation maps (Grad-CAM). (c) The Grad-CAMs are centralized in the style/pattern images. (d) The content-object detection success is higher using Faster RCNN [51]. Best viewed in color.

Table 1: List of frequently used symbols in this paper

Name	Symbol	Name	Symbol	Name	Symbol	Name	Symbol
Activation Function	f	Deep network	N	Random Function	ϕ	Style-Grid	SG
Algorithm	H	Fake Style Images	I_s	Randomize Challenge	Q	Stylized Grid	TG
Attack Loss	L_{attack}	Gram matrix	G	Recognition Gap	δ	Stylized Image	I_t
CAPTCHA Challenge	Q_{SMC}	Human Population	β	Set of Content-images	C	Success Rates	α
Cognitive Constraints	U	Hyper-parameters	$\gamma; \lambda$	Set of Style-images	S	Time-interval	τ
Content Image	I_c	Intelligent Agent	η	Style Image	I_s	Total Attack Loss	L_{total}^{attack}
Content Loss	$L_{content}$	Mapping Function	M	Style Loss	L_{style}	Total Variation Loss	L_{TV}

Table 2: Study on some recent image-based CAPTCHAs. Accuracy means human accuracy (%) to solve, time (s) taken to solve, and probability implies the probability of automated attacks. † indicates the method with NST.

Year	CAPTCHA Method	Accuracy	Time	Probability
2023	Annuli CAPTCHA [76]: asks users how many circles and ovals are present in the image.	88.19	-	1.25×10^{-1}
2022	TICS CAPTCHA [31]: ask users to select images according to the given text.	94.12%	-	1.98×10^{-3}
2021	HandCAPTCHA [7]: distinguish 2 real hand images from 7 fake hands from a grid of 3×3 images.	98.55	9.67	4.7×10^{-3}
2020	StyleCAPTCHA† [10]: test asks a user to classify 10 Neural Style Transferred face images.	59.87	-	-
2019	Grid-CAPTCHA† [13]: Grid shows 9 stylized images and the user requires to select all images according to a short description.	75.04	11.83	7.5×10^{-5}
2018	Style Area CAPTCHA† [63]: Depends on semantic information understanding and pixel-level segmentation. NST generates synthetic and style transferred areas (4-7) of random shapes (e.g., rectangle, leaf, etc.).	93.1	9.73	4.9×10^{-3}
2017	Deep Learning CAPTCHA [47]: creation and addition of immutable adversarial noise on images (12) representing natural objects, shapes, textures, etc.	86.67	7.66	7.0×10^{-1}
2016	Annulus CAPTCHA [21]: count the number of the distorted geometric annulus (1-8) in a complex background.	89	8.89	3.1×10^{-5}
2015	CAPTCHaStar [14]: recognize space by moving the cursor inside a drawable area containing stars. It contains (black and white) small white squares (stars), placed in a squared black space to form a shape.	90.2	<27	9.0×10^{-4}
2014	FaceDCAPTCHA [25]: detects distorted human faces (2-4) from fake faces from a grid.	98.5	-	6.0×10^{-5}

[29] and Inception-v3 [62] as standard backbone CNNs to verify the security strengths of SMC. The lower accuracy of this attack scheme implies the effectiveness of our proposal. The main contributions of this paper are:

- A novel Style Matching CAPTCHA is devised leveraging neural style transfer. The proposed method matches the styles or patterns to thwart available object detectors, vision-based AI tools, and related *bot* attacks.
- Human performance in solving the SMC challenges through web and Smartphone applications implies improved usability which achieves state-of-the-art performance.
- Deep learning-based attack schemes underlying on the standard CNNs fails to achieve satisfactory performance to break an SMC challenge.

- Several image pre-processing methods (e.g., denoising) are applied to break the strengths of SMC. A comparative study with text-CAPTCHAs is presented. In-depth security analysis implies the benefits of our SMC.
- Comprehensive evaluation implies the robustness of our scheme, reduces the probability of attack, and widens the applicability of proposed SMC.

The remainder of this paper is organized as follows: Section 2 presents a study on CAPTCHAs and NST. Section 3 describes the proposed method, Section 4 analyses security aspects, Section 5 discusses experimental results. Section 6 states limitations and future work, followed by a conclusion in Section 7.

2 Related Works

Several CAPTCHA techniques have been developed since its inception in 2003 [37, 66]. Existing schemes can be extensively categorized into the following classes: (a) text, (b) image, (c) audio, (d) video, (e) cognitive, and (f) miscellaneous. Among these, text-CAPTCHAs are broadly studied [45].

Recently, a text-based CAPTCHA technique based on the Hindi language that concurrently uses printed and handwritten Hindi characters is presented [39]. In this work, k -nearest neighbours, support vector machines, and random forest classifiers are used to break ten distinct coloured CAPTCHAs. Also, recognition of hollow Hindi characters in text-CAPTCHAs is described and achieved good performance to recognize distorted and multi-scaled hollow characters [36]. However, modern optical character recognition (OCR) technology can breach text-CAPTCHAs and compromise security [63]. Remarkable success has been attained to break monochrome Devanagari CAPTCHA schemes using several classifiers, such as Random Forest in [38]. Several new CAPTCHA design strategies are developed using deep learning in recent times. On the contrary, image-CAPTCHAs are used to distinguish human and malicious bots using various vision-based schemes, such as object detection, target recognition, scene understanding, etc. These schemes are deployable on various touch devices, and smartphones with better convenience and usability [80]. Thus, we have studied image-CAPTCHAs and a concise study is presented in Table 2.

2.1 Generic Image-based CAPTCHAs

The naming CAPTCHA, distinguishing CAPTCHA, and anomaly CAPTCHA [77] are the initial image-CAPTCHAs. The users require to find the similarities or dissimilarities in a set of images. However, these schemes suffer from misspelling, mislabelling from users, synonym words, and polysemy words. To solve Implicit CAPTCHA [3], users need to tap on the ideal position or a particular word on an image. Collage CAPTCHA [58] is developed by combining various objects into a single image, and the users are asked to choose specific objects. It necessitates a database of tagged object images, which are dispersed and rotated arbitrarily in the background. Collage CAPTCHA is often attacked by using object segmentation and recognition-based methods. ARTIFACIAL [53] introduces a complex 2D facial model that embraces human face recognition capability in the challenge. However, ARTIFACIAL can also be cracked easily, as described in [81]. ASSIRA [19] uses a grid to represent images of cats and dogs, and users must recognize the cats. Multiple-choice questions are used by ASSIRA to expand the solution space and thus improve security. Golle *et al.* [24] use image recognition to distinguish cats and dogs in ASSIRA by integrating colour and textural features to undermine the scheme. IMAGINATION [17] exploits people’s imagination capability by allowing them to interpret images from a distorted and cluttered background. It comprises two subsequent steps: click and annotation. Still, it is vulnerable to attack, as specified in [81]. A significant breakthrough is Google’s reCAPTCHA [67]. It is ideated with a virtual checkbox, and interestingly it does not require any text, image, audio, or video data to pass the challenge. The latest version of Google’s No CAPTCHA reCAPTCHA [56] increases usability. However, deep learning models can solve Google’s No CAPTCHA reCAPTCHA [61]. Polakis *et al.* [48] present a scheme using an image selection and modification mechanism based on Facebook’s social authentication to boost security. Human face photos that are indistinct, obstructed, or from the rear side are used as features in this technique to protect against malicious bot efforts. Users can recognize their friends from these avatars based on prior information. Several CAPTCHAs based on human faces, such as FR-CAPTCHA [26] and FaceD-CAPTCHA [25], have also been tested. In both schemes, the face images are blended over a complex background followed by various geometric transfor-

mations (*e.g.*, rotation) and noise. FP-CAPTCHA [50] challenges the users to click on human facial points such as the eye, nose, and mouth, which are laid over a cluttered background and additional noises. HandCAPTCHA is implemented using a randomized combination of two real and five to seven fake hand-images [6]. In addition to hand biometric verification, liveness detection is added to the verification pipeline to improve security [7]. Vessel CAPTCHA [16] targets 3D brain vessel segmentation. It divides the image into 2D patches and users identify the patches containing a vessel or its part. Most of these techniques are based on object detection or localization which are vulnerable to attacks. Jia *et al.* [31] proposed a novel image-text-based model for creating CAPTCHA that is based on cognitive processes and semantic reasoning. In order to create a multi-conditional CAPTCHA that can resist the attack of CNN’s classification, this technique combines three features: sentence, object, and location.

2.2 Mobile device-based CAPTCHAs

There has been a lot of research done on mobile-based CAPTCHA schemes [9, 64, 74], and a few image-based examples are included here.

Noise CAPTCHA applies two noisy images of varying sizes and a concealed object or message at a precise location in the image [46]. To pass a CAPTCHA test, participants require to drag the noisy image over a large image until the hidden item is visible, followed by a submit button. An orientation sensor-based CAPTCHA scheme named SenCAPTCHA is described in [20]. It asks users to tilt their phones to direct a coloured ball toward the center of an animal’s eye after displaying an image of that animal on the screen. TapCAPTCHA [2] is based on audio and gesture interaction with smartphone devices, and assessed its usage by visually challenged people. Also, TapCAPTCHA is compared with audio CAPTCHA based on efficiency, accuracy, user satisfaction, and workload. In augmented Reality CAPTCHA [32], a user is asked to prompt with a specific marker in a 3D physical environment. As the mobile device rotates, the CAPTCHA’s position changes. Once the CAPTCHA shape is detected, a user needs to spin the mobile device for angular alternation. Annuli-CAPTCHA [78] uses overlapping of annuli which consist of circles and ovals as geometrical shapes. Users are asked to enter the correct number of circles and ovals in the query for the solution.

2.3 Neural Style Transfer (NST) based CAPTCHAs

Deep learning-based DeepCAPTCHA [47] applies adversarial noise, and strengthens security by defending against common image processing attacks. Recently, NST has been adapted in various image-CAPTCHAs. SACAPTCHA [63] generates a synthetic image, by transferring different shapes of various styles. Users are instructed to click on foreground style-transferred regions based on a brief description to solve the challenge. On the contrary, an attack-model using mask R-CNN to determine various shapes which are originally applied to improve the resilience of SACAPTCHA is described in [49]. The experimental result (*i.e.*, maximum 96% F1 score) is enough to recognize an object/shape provided in the challenge. The results imply SACAPTCHA is also vulnerable to the mask R-CNN-based attacking scheme. In another direction, Generative Adversarial Networks (GAN)-based end-to-end text-CAPTCHA cracking technique is proposed in [41]. It follows cycle-GAN-based synthesizers to create a large number of synthetic CAPTCHA examples for training in addition to active transfer learning. It achieves 97.6% highest success to break real-world CAPTCHAs from various websites.

In Grid-CAPTCHA [13], users should choose one out of nine stylized images according to a brief scene description. In addition, the scheme employs the same style to convert all of the images to stylized versions. To baffle recent

CNNs, StyleCAPTCHA [10] has been proposed. NST blends human face images with reference styles to produce stylized face images. The challenge involves classifying ten stylized images into either human faces or animal faces. Inspired by these works, we present a new image-CAPTCHA algorithm to strengthen security with maintaining human performance.

3 Proposed Methodology

The Style Matching CAPTCHA (SMC) generates a random challenge Q underlying on NST, which requires three main components: a content-image (I_c), a style-image (I_s), and a stylized-image (I_t), as defined earlier (Table 1). Table 1 contains all the symbols that are used in this article. All the symbols are sorted according to their names. Additionally, the symbols are described whenever they are used in this article.

3.1 Style Matching CAPTCHA (SMC): Overview

A vital target of a CAPTCHA algorithm H is to maximize the recognition gap (δ) between the success rates of α fraction of the human population (β), and an intelligent agent (η), *i.e.*, $|\beta - \eta| = \delta \geq \epsilon$; with $\epsilon > 0$, and ideally $\delta \approx 1$. To maximize the margin of δ , serial repetition of the randomized challenge Q should be solved for ω times within a stipulated time-interval τ for each answer. According to [66], these parameters $U = \{\alpha, \beta, \eta, \tau\}$ are essential to define H as a hard AI problem. The verification of a user by solving Q for ω times (τ time for each answer) in a sequential repetitive manner, is denoted as $Q_\omega = \prod_{i=1}^{\omega} Q_i$. As the answers are Boolean (yes/no), a user is permitted, if $Q_\omega = 1$. Considering all the prerequisite parameters, we define a generic $Q_\omega = H(I_t, U)$. As U represents the cognitive constraints, thus, from the algorithmic design perspective, it can be simplified as $Q_\omega \approx H(I_t)$.

$$I_t = N(I_c, I_s); Q_\omega = H(I_t, U) \approx H(I_t) \quad (1)$$

where, I_c , I_s , and I_t represent the content image, style image, and stylized images respectively. N is the deep network, H represents the algorithm for creating Q_{SMC} , and U is the cognitive constraint. Q represents a random challenge, and Q_ω is the user verification by solving Q for ω numbers of times. Now, we define SMC specifically as Q_{SMC} from a generic Q_ω . Our approach chooses I_s from a set of random samples provided in a grid structure (Fig. 4). By observing I_c and output I_t , a user has to select the appropriate style/pattern I_s . From user's perspective, the challenge is posed as

$$Q_{SMC} = H(I_s | I_c, I_t) \quad (2)$$

where, Q_{SMC} represents the challenge of SMC, H represents the algorithm for creating Q_{SMC} , and I_c , I_s , and I_t represent the content image, style image, and stylized image, respectively.

Considering the key components, from the designer's view:

$$Q_{SMC} = H(I_c, I_s, I_t, \phi, M) \quad (3)$$

where, ϕ is a random function that selects images at random from the database (D) and M is a mapping function that places the styled or stylized images in the image grid.

A random function ϕ selects the I_c and I_s (from database D) with which a style transferred image I_t is generated using a deep network N to implement NST. Database D contains a set of content-images (I_c), denoted as $C = \{I_c\}$; and a set of style-images (I_s), denoted as $S = \{I_s\}$. It can be noted that any standard deep network can be defined as $N = f(\sum_{n=i} W_i X_i + b)$, where f is

the activation function, W_i is weight, X_i is input, and b is the bias in the i^{th} layer. Here, we denote N for simplicity. In addition, a few more random styles which are not used in NST but are selected for filling the remaining empty places in the style-grid (SG), denoted as fake styles \tilde{I}_s . A challenging Q_{SMC} is generated with all of these images by positioning the actual (I_s) and wrong (\tilde{I}_s) styles in the grid SG arbitrarily. A function M maintains the correct pair-wise mapping between (I_s, I_t) with respect to the reference content I_c .

Now, the user needs to match the correct one-to-one correspondence between (I_s, I_t). For a given I_c , a series of valid matching of ω number of I_t with the respective I_s is considered as a correct solution of a given query. It is defined as $Q_{SMC} = \prod_{i=1}^{\omega} (I_s^i, I_t^i)$, where each I_t^i is produced using an ordered pair of $\{(I_c, I_s^i)\}_{i=1}^{\omega}$ inputs to the N , respectively. It can also be stated as

$$Q_{SMC} = H(I_s^i | I_c, I_t^i) = H(I_s^i | I_c, N(I_c, I_s^i)) \quad (4)$$

where $i \in [1, \omega]$, ω is the number of times a user has been verified by solving Q_{SMC} , I_c , I_s , and I_t represent the content image, style image, and stylized image, and N is the deep neural network. Conversely, deep networks use non-linear activation between the layers, random hyper-parameters, and random weight distributions in the learning process by optimizing the loss function. Thus, it is hard for a system to replicate the same model-output by guessing the model-parameters. As a result, it adds more security to solving a challenge by an automated program within a given time limit.

Algorithm 1 SMC Algorithm (Q_{SMC})

Require: Input image-sets: Content C , and Style S , row $m = 3$, and column $n = 3$

Ensure: CAPTCHA Challenge Q_{SMC}

Define: Style-grid $SG_{m,n} = Null$, and stylized-grid $TG_m = Null$. A random function ϕ , and a mapping function M . A random number, $p[1, n] \in \mathbb{N}$; and a set P of $(n - 1)$ numbers, $\{P[1, n] \in \mathbb{N} | P \setminus p\}$

```

1:  $I_c \leftarrow \phi(C)$  ▷  $I_c$  for content images
2: for  $i = 1 : m$  do
3:   generate  $p$  and  $P$ 
4:    $I_{si} \leftarrow \phi(S)$  ▷  $I_s$  for real styles
5:    $SG_{i,p} \leftarrow M(I_{si})$ 
6:    $I_t \leftarrow NST(I_c, I_{si})$  ▷ using Eq.1,  $I_t$  for stylized images
7:    $TG_i \leftarrow M(I_t)$ 
8:   for  $j = 1 : n - 1$  do
9:      $\tilde{I}_{sj} \leftarrow \phi(S)$  ▷  $\tilde{I}_s$  for fake styles
10:     $SG_{i,p[j]} \leftarrow M(\tilde{I}_{sj})$ 
11:   end for
12: end for
13:  $Q_{SMC} \leftarrow H(I_c, SG, TG)$  ▷ using Eq.3-4

```

SMC Algorithm: Our Algorithm-1 produces a random challenge Q_{SMC} , shown in Fig.1 and Fig. 4. Initially, style-grid SG and stylized-grid TG are considered as two empty image-grids. A function ϕ is used to select a random content (I_c) chosen from C (Content images from Database D). Next, three random styles (I_s) are selected from S which are placed on the $m \times n$ style-grid SG , one in each row using a function M to maintain the correspondence with the actual/real style and its respective stylized rendering I_t in TG . To adhere randomness in the positioning of an actual style, a random natural number p is generated as an index which is used for placing a real I_s in a row. The indexes of remaining free-places in the same row are stored in a set P . Each row's style placement task is altered according to the indexes stored in p and P .

Next, a deep neural style transfer NST method is utilized to produce stylized I_t and place them in the respective column of TG . The remaining empty positions in SG are filled with counterfeit/fake styles (\tilde{I}_s) according to P using M . This process is serially repeated for each row (here, $\omega = m = 3$), and it is scalable to a higher row/column value. These classes of images with their corresponding grid representations (I_c, SG, TG) are juxtaposed in a single frame as algorithmic output $H(I_c, SG, TG)$. Finally, a user-friendly interface

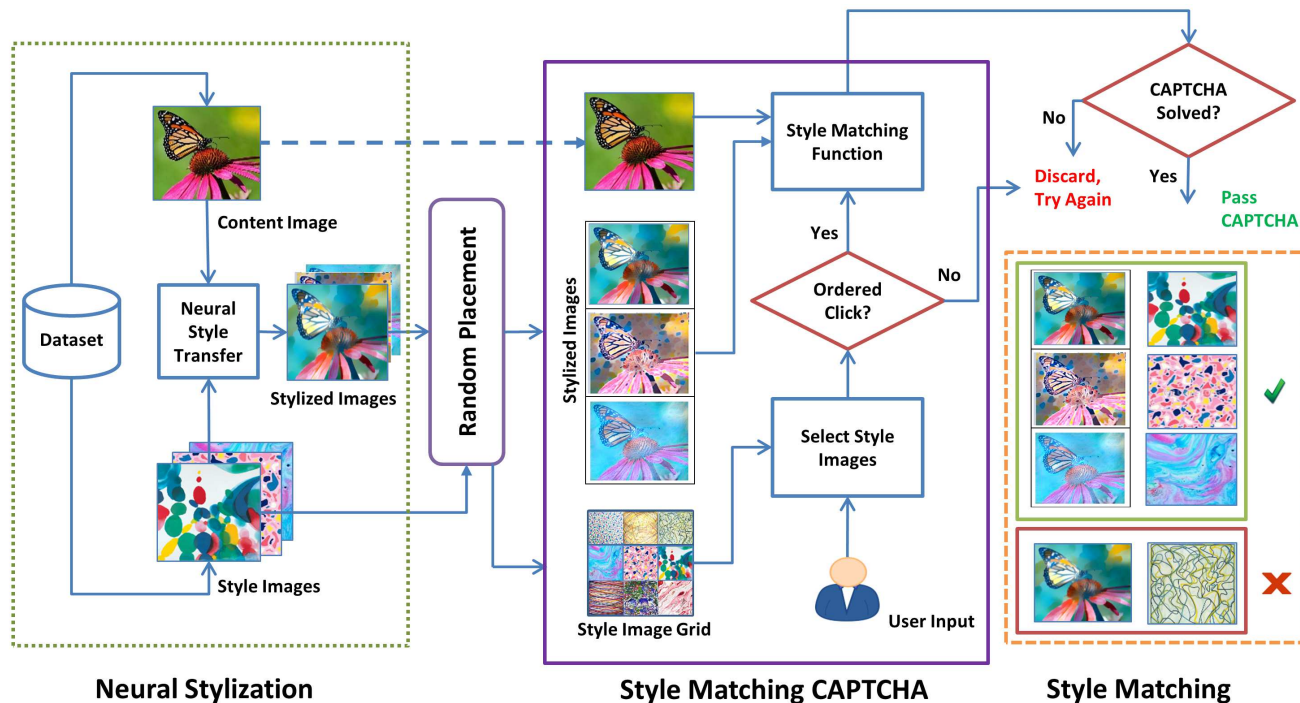
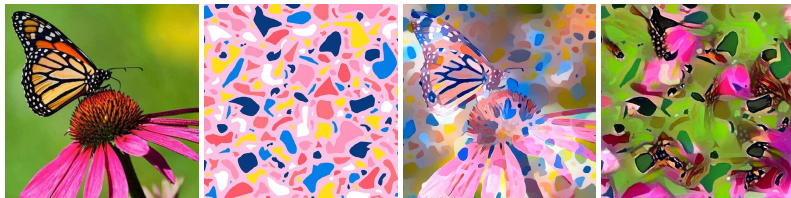


Fig. 4: Artistic Style Matching CAPTCHA (SMC). It leverages deep neural style transfer (NST) using the content and style images, shown on the left side. A set of randomly selected style images are placed in an image-based grid which offers security against malicious attacks. The user is asked to select (click on the SMC web page or finger-touch on SMC-App) the input style images for correct matching based on the rendered stylized and content images as the cue, according to a specific order, shown in the middle. The correct and wrong matching scheme is illustrated on the right side.



(a) Left to Right: I_c , I_s , $I_{t_1} = I_{c,s}$ and $I_{t_2} = I_{s,c}$. $SSIM(I_c, I_{t_1}) = 0.555$, $SSIM(I_s, I_{t_1}) = 0.338$, and $SSIM(I_c, I_{t_2}) = 0.343$



(b) Left to Right: I_c , I_s , $I_{t_1} = I_{c,s}$ and $I_{t_2} = I_{s,c}$. $SSIM(I_c, I_{t_1}) = 0.897$, $SSIM(I_s, I_{t_1}) = 0.548$, and $SSIM(I_c, I_{t_2}) = 0.509$

Fig. 5: Changing the roles of content to style inputs in NST, and vice versa to measure the similarity score using the Structural Similarity Index (SSIM). A higher SSIM [0, 1] value denotes more similarity between the image-pair.

(i.e., SMC-App or web-application) presents this random challenge Q_{SMC} to the user for solving.

Particularly, image pre-processing techniques are applied for basic structural representation of SMC. Firstly, a noisy background is created, over which a 3×3 grid for style images and a 1×3 grid for stylized images are generated. Next, 3 stylized images are chosen and placed randomly on the stylized-grid (TG). Next, 3 corresponding style images are selected and placed randomly on style-grid (SG) using a mapping function (M). Likewise, function M is used to place fake style images on the remaining 6 positions in the SG grid. Also, random rotation and scaling are applied to the style images at the processing

stage. To remove additional black pixels appearing at image-boundary regions due to rotation, an alpha channel is added to convert those pixels transparent. Finally, the resized style images are placed over the SG grid.

The Gram matrix (6) in NST (Sec. 3.2) learns the feature map distribution within a layer. The style-loss L_{style} (8) improves the matching rate between the feature map distributions of I_s and I_t in a layer. It is obvious that N optimizes two different loss functions $L_{content}$ (5) and L_{style} (8), for two different inputs I_c and I_s , respectively. So, it does not produce the same output I_t , if we interchange the roles of input images and vice versa, i.e., $I_{t_1} = I_{c,s}$ and $I_{t_2} = I_{s,c}$. The results $I_{c,s}$ and $I_{s,c}$ are significantly different for two cases as $N(I_c, I_s) \neq N(I_s, I_c)$, i.e., $I_{t_1} \neq I_{t_2}$. Clearly, our objective is to generate Q_{SMC} as a CAPTCHA challenge based on a blended image derived by NST. To add more insights, we have tested the significance of interchanging the roles of content and style inputs. However, this input alternation strategy leads to a more challenging situation for the users in solving a random Q_{SMC} easily.

Here, the outcomes both possibilities i.e., I_{t_1} and I_{t_2} are shown in Fig. 5. However, we prefer I_{t_1} according to the structural similarity index (SSIM), as I_{t_1} offers a balance between usability and security, evident from Fig. 5.

During verification, a human participant requires careful observation of the content and stylized output images to pass the challenge. The user is requested to select the corresponding styles from the style-grid, one per row. Humans can easily detect the patterns where automatic programs or bots can't perform the task with rigorous attempts. If a user selects three styles/patterns correctly, it is considered a successful solution of Q_{SMC} , otherwise not (Algorithm-1).

3.2 Neural Style Transfer (NST)

We have revisited NST [22, 34] to design the proposed SMC. A deep network (N) applies a non-linear filter bank at various layers to generate feature maps. N computes feature maps F^l at layer l from an I_c with a dimension of $H_l \times W_l \times D_l$, height is H_l , width is W_l , and the number of channels is D_l at the

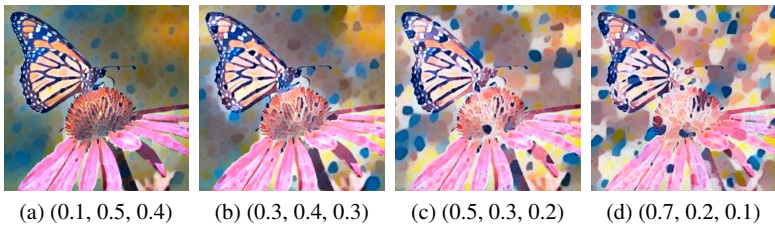


Fig. 6: Variation of hyperparameters ($\gamma_1, \gamma_2, \gamma_3$) in Eq. (10) given as a triplet

l^{th} layer. The feature space is represented as $F_{i,j}^l \in \mathbb{R}^{H_l \times W_l \times D_l}$, and $F_{i,j}^l$ is the activation of i^{th} filter at j^{th} location in layer l of N . Consider F^l be the actual feature map of I_c , and \hat{F}^l is the feature map of rendered stylized I_t at layer l . The loss function (squared Euclidean norm) between these feature maps is

$$L_{content}(I_c, I_t, l) = \frac{1}{2} \left\| F_{i,j}^l - \hat{F}_{i,j}^l \right\|_2^2 \quad (5)$$

The higher layers of N infer high-level content information and the object's appearance in the feature maps. These higher layers are apposite for representing content summary over the style/texture information. To learn the textural pattern in the layers, feature correlation is an effective measure that is computed using the inner product of the feature maps (*i.e.*, i^{th} filter at the j^{th} location) at layer l . This correlation is represented in the gram matrix, defined as

$$G_{ij}^l = \sum_k F_{ik}^l F_{jk}^l \quad (6)$$

where, F_{ik}^l and F_{jk}^l represent feature spaces at l^{th} layer. The stylization process at layer l can be optimized using the gram matrix G_{ij}^l of corresponding source style (I_s), and the gram matrix \hat{G}_{ij}^l of rendered styled (I_t), respectively.

$$E_l(I_s, I_t) = \frac{1}{(2H_l W_l D_l)^2} \left\| G_{ij}^l - \hat{G}_{ij}^l \right\|_2^2 \quad (7)$$

where, $E_l(I_s, I_t)$ is MSE loss between the gram matrix of the style image and the stylized image, I_s and I_t represent the style image and stylized image, H_l is the height, W_l is the width and D_l is the number of channels at the l^{th} layer. The total stylization loss, including all the layers, is given as

$$L_{style}(I_s, I_t, l) = \sum_{l \in L} w_l E_l(I_s, I_t) \quad (8)$$

where, w_l are the weights at layer l and L is the number of layers in N , I_s and I_t represent the style image and stylized image. Finally, these two loss functions are linearly combined and jointly optimized to minimize the error. Also, to enrich spatial smoothness, total variation loss L_{tv} [34] is used. It is the sum of the absolute differences between the neighborhood pixels, denoted as $px_{i,j}$ and $px_{i+1,j+1}$.

$$L_{tv}(I_t) = \sum_{i,j} |I_t(px_{i,j}) - I_t(px_{i+1,j+1})| \quad (9)$$

Combining these three losses, the joint-loss function is

$$L_{total} = \gamma_1 L_{content} + \gamma_2 L_{style} + \gamma_3 L_{tv} \quad (10)$$

where, $\{\gamma_i\}_{i=1}^3$ are hyperparameters that estimate a trade-off between the content and style in the rendering process. $L_{content}$ is total content loss from Eq. (5), L_{style} is total style loss from Eq. (8), L_{tv} is total variation loss, Eq. (9).

Here, we consider a random variation in their values such that $\sum_{i=1}^3 \gamma_i = 1$. It offers additional randomness in the perturbation (Fig. 6) to hinder malicious bots. The VGG-19 [54, 59] pre-trained on ImageNet, is adapted for implementing SMC. The stylization process with the loss values at four intermediate iterations within 100-2000 is shown in Fig. 8. All the experiments are conducted in Tensorflow 2.x using Python 3.7 scripts, and Google Colab GPU environment is used for deep learning experiments simulation.

3.3 Hyperparameters

The total loss is obtained from Eq. (10), where γ_1, γ_2 , and γ_3 are the hyperparameters that estimate trade-off between the content and style in the rendering process. With this variation, we can control the amount of style that would be present in the stylized output image. Fig. 6 illustrates this process with 4 types of variations with the hyperparameters. It is quite natural that if we reduce the style component, the attack accuracy will also degrade as well, as it would be a tough job for the users also. Therefore, we have maintained a fair balance between style and content images to produce the stylized output. However, we have generated a lot of stylized output with reduced style content and performed a Type-III attack simulation. In this evaluation, only 35% accuracy is achieved that shows the chances of an attack are minimal.

3.4 Dataset Description

Element Based Textures Dataset (EiBa) [23] consists of procedurally generated realistic images with variations in shapes, colours, etc. It includes 30k texture images with various levels of local symmetry, stationarity, and density of (3M) localized texels. Element-based textures are a type of texture made up of texels, which are named elements that are dispersed according to statistical distributions. The textile, fashion, and interior design industries are the most common users of this dataset. Texel-Att is frequently utilised since current texture descriptors fail to correctly define element-based texture. Texel-Att is a framework for representing and classifying element-based textures that is fine-grained and attribute-based. In our experiment, the EiBa dataset is used as style images. In addition, 2000 style and pattern images are collected from other resources such as Kaggle's Abstract Art Gallery¹.

The content images are collected mainly from the Kaggle repository². Our dataset consists of high-quality images of more than 100 object categories such as human faces, animals, birds, flowers, etc. Around 2000 fine-grained content images are collected and stored in Database. Finally, with style image and content image, we successfully created around 3000 stylized images. All the images are resized to 512×512 pixels and stored in our database. Samples of these image categories are shown in Fig. 7.

4 Security Analysis

An important attribute of a CAPTCHA is its resiliency to various malicious attacks. Here, the security benefits of SMC over various attacks are described.

4.1 General Deep Learning Attack

Image-CAPTCHAs are vulnerable to deep learning attacks. It is possible for CNNs to extract and recognize the content information from a stylized image. However, to find style or pattern information from the stylized image, it is tough to identify the textured pixels using modern AI tools or CNNs. To explore further in this direction, we have simulated several attack schemes by considering various datasets, assuming that an attacker has collected some random samples of stylized and style images which are used in SMC challenges at various sessions. Next, a thorough systematic cropping technique is applied to these samples to generate more sub-samples for training a CNN for recognition. From each stylized image, 150 sub-samples are generated, and an 80:20 ratio is followed for training (80%) and testing (20%) using the ResNet-50 [29] and Inception-v3 [62] as standard CNN backbones. Our objective is to

¹ <https://www.kaggle.com/bryanb/abstract-art-gallery>

² <https://www.kaggle.com/c/cvdl2020finegrained>

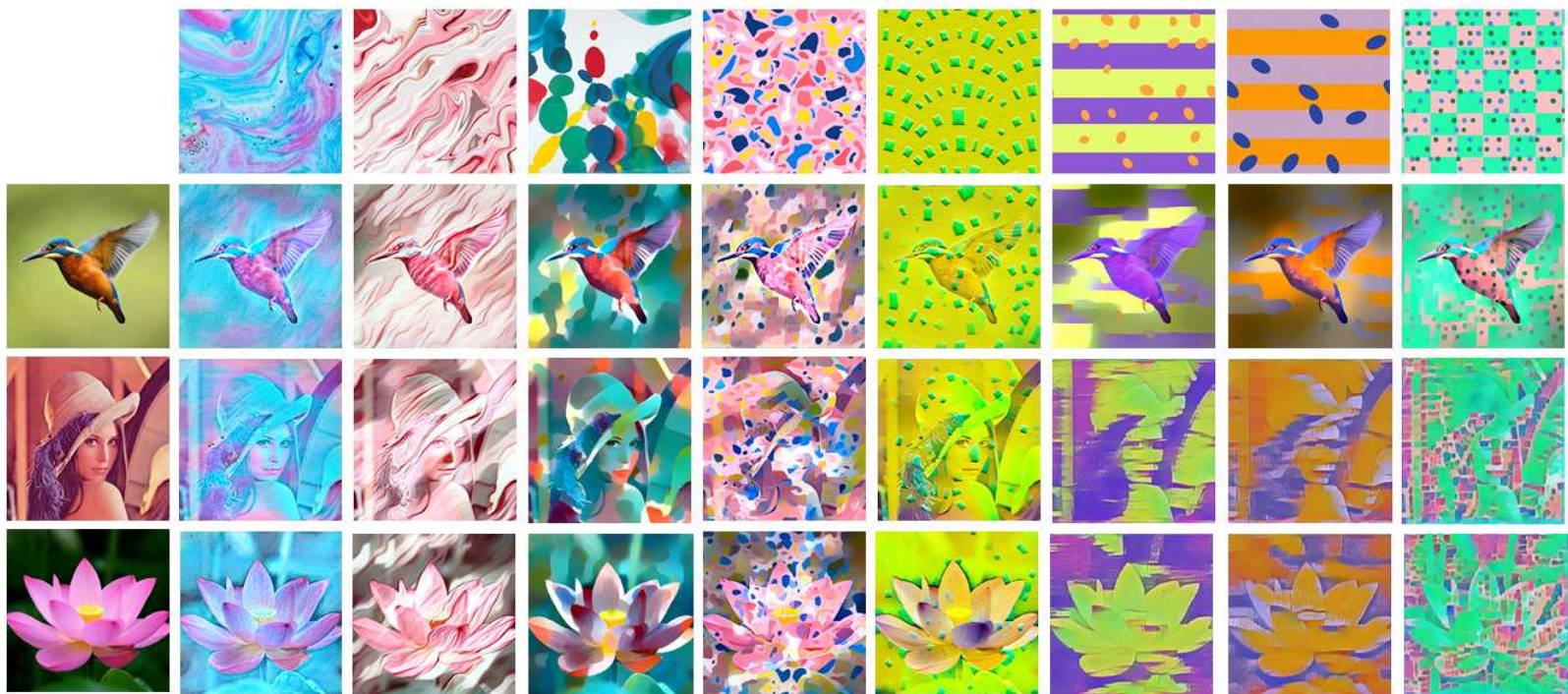


Fig. 7: Examples of content (row), styles (col), and their NST outputs.

evaluate the accuracy of a deep model to classify the required styles using a few samples with less manual supervision. This simple classification indicates the underlying strengths of SMC to thwart deep learning attacks. We have created 4 types of threat models and related datasets in each case.

- Type-I: Train and test both with stylized image samples.
- Type-II: Train and test both with style image samples.
- Type-III: Train with stylized and test with style samples.
- Type-IV: Train with style and test with stylized samples.

Here, Type-I and Type-II perform like a general supervised classification task with similar categorical variables. Type III and IV are more challenging threat models as the training set provides simple pattern information but testing data is conjugated with other (content) information through NST and vice-versa, like targeted adversarial examples.

SMC is implemented for both PCs and smart devices. Therefore, we have chosen the size of images as 256×256 pixels for PC's and 128×128 pixels, for Smartphones and touch devices. In Section-3.4, we have described the datasets, which are used in SMC. First, all the content images, style images, and stylized images are collected and resized to 256×256 pixels and 128×128 pixels, for respective devices. For attack simulation, three distinct scenarios have been experimented with.

- Case-1: the size of style and stylized images is 256×256 pixels.
- Case-2: the size of style and stylized images are reduced to 128×128 pixels.
- Case-3: size remains same as 128×128 pixels with added noise on both image categories.

We have accumulated different style images and corresponding stylized images from our database to create training and testing data. A thorough systematic cropping, row-by-row, and column-by-column divide the style and stylized images into smaller segments like 32×32 and 64×64 pixels. Also, image augmentation is applied to increase the sample size. Finally, these smaller image segments are divided with a ratio of 80:20 for training and testing, respectively. Twelve distinct datasets containing both styled and stylized images at different

scenarios are generated for attack simulation. The outcomes of our attack simulation are given in Table 3. A dataset has been created that consists of 1560 stylized sub-samples for training and 390 stylized sub-samples for testing in Type-I simulation. Similarly, another dataset is created for Type-II simulation, consisting of only style images. Our primary focus is on Type-III-IV, where style images are used for training and stylized images are used for testing, or stylized images are used for training and style images are used for testing, respectively. For Type III-IV, the dataset comprises a total of 1950 training sub-samples and 390 sub-samples for testing.

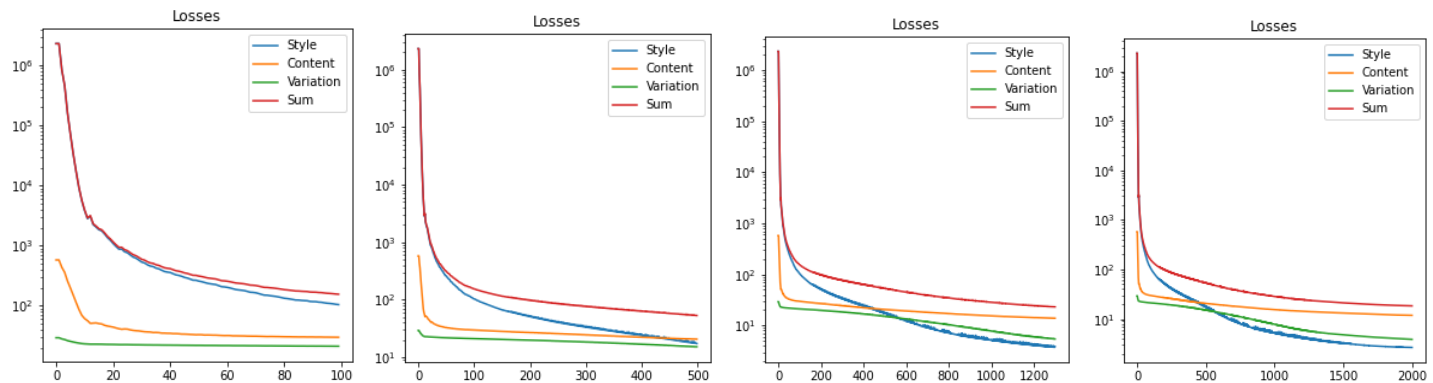
ResNet-50 and Inception-v3 are used as CNN backbones and are trained for 50 epochs with a batch size of 8. The Stochastic gradient descent (SGD) optimizer with 0.01 as a learning rate is used for training. The results are given in Table 3. The objective is to verify whether available CNN can be used for classifying the stylized outputs which are considered Type-I deep learning attack methods on SMC. The classification accuracy on these styled samples using Inception-v3 is 96.47%, and ResNet-50 is 85.01% (Table 3) with 256×256 resolution. Similarly, corresponding style images which produce the stylized images are classified using the same CNNs for Type-II attacks. The accuracy using Inception-v3 is 97.54%, and ResNet-50 is 85.68%. The CNNs can classify the stylized and style images with high accuracy and precision. For the entire process of neural-style transfer operations and various attack analyses, we have used the Google Colaboratory.

4.2 Attacks at Latent Layers

We have delved into style-content blending mechanism at mid-level convolutional layers of VGG-19 in NST architecture by exploring the gram-matrix formulation and loss functions. The mid-level layer summarizes a latent representation of compact feature space. It is widely used in encoder-decoder architecture, GAN-based model representations, etc. The mid-level blocks learn latent-features that are not easily interpretable into object/style categories from such bottleneck layer(s). Hence, feature learning is very difficult when the model- and hyper-parameters are intrinsically highly random initially, and are further optimized during training. However, current state-of-the-art CNNs



(a) Intermediate output at 100, 500, 1300 and 2000 iterations



(b) Intermediate loss-functions at 100, 500, 1300 and 2000 iterations

Fig. 8: Stylization process with loss values are optimized progressively at various steps: 100, 500, 1300, and 2000 iterations (left to right).

Table 3: Performance of deep learning attacks on SMC. Style image-set: S and Stylized image-set: T

Type	Model	Training Image	Testing Image	Case-1: Image Size 256×256		Case-2: Image Size 128×128		Case-3: Image Size 128×128 after denoising	
				Accuracy	Precision	Accuracy	Precision	Accuracy	Precision
Type-I	Inception-v3 ResNet-50	1560(T)	390(T)	96.47	96.46	95.47	96.57	76.48	76.74
				85.01	85.34	94.65	94.94	77.31	77.47
Type-II	Inception-v3 ResNet-50	1560(S)	390(S)	97.54	97.88	94.54	94.88	77.39	77.71
				85.68	86.76	95.68	95.76	78.06	78.31
Type-III	Inception-v3 ResNet-50	1950(T)	390(S)	36.23	37.96	34.23	34.96	30.21	30.22
				33.22	33.68	36.90	36.28	31.68	31.75
Type-IV	Inception-v3 ResNet-50	1950(S)	390(T)	35.86	39.43	26.86	27.43	23.59	23.62
				37.24	38.94	27.78	28.15	23.81	23.95

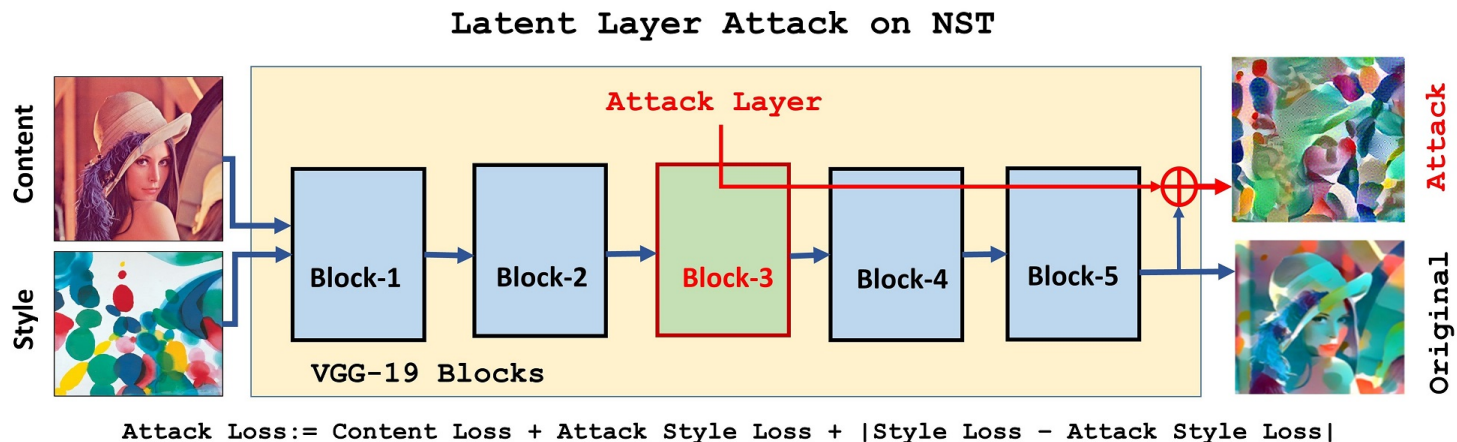


Fig. 9: Attack at intermediate layer of pre-trained VGG-19 (Block 3, Conv layer-2) of NST architecture.



Fig. 10: a) Left: MSE of stylized images extracted from intermediate latent layers. Content output from conv-1 layer; Style output from conv-2 layer. MSE of stylized images extracted from intermediate latent layers. b) Right: Content output from conv-4 layer; style output from conv-7 layer. The corresponding similar common style images are shown in middle.






Content With Style Image	Block-2 conv1	Block-2 conv1	Block-3 conv1	Block-3 conv1
	Sample-1	Sample-2	Sample-3	Sample-4
	0.403	0.835	0.331	0.844
	0.119	0.548	0.098	0.596
	0.415	0.859	0.417	0.953
	0.236	0.913	0.234	1.000
	0.000	0.314	0.001	0.404

Fig. 11: Intermediate layer output from block-2 conv-1 and block-3 conv-1.

are very powerful to crack the underlying strengths of NST easily. In our experimental study, the highest performance of latent-layer based attack simulation is about 45% which is reasonable and much lower than other existing works such as SACAPTCHA [63]: 82%, CAPTCHAStar [14]: 96%, and others. Hence, our proposed SMC can improve the strengths to a significant extent (more than double) than other works. A comparative analysis is presented in Table 13.

Our assumption might be relevant for an attack at some latent/intermediate convolutional layer/block of CNNs, shown in Fig. 9. We have assumed that an

intruder can access the mid-level latent-style feature representation of block-3 convolutional layers of VGG-19. Particularly, ‘block3_conv1’ for style-image and ‘block3_conv2’ for content-image are accessible to render a stylized output. Whereas, the actual style is learned and transferred through all CNN blocks in the main model. Accordingly, we have defined an attack-loss function based on the latent-layer as

$$L_{attack} = L_{style}^{attack}(I_s, I_t, l_{attack}) = w_l E_l(I_s, I_t) \quad (11)$$

where, l_{attack} implies the latent-layer under attack, w_l are the weights at layer l , $E_l(I_s, I_t)$ is the MSE loss between the gram matrix of style image, I_s and I_t represent the style image and stylized image respectively. The total attack-loss is defined as

$$L_{total}^{attack} = L_{content} + L_{attack} + \lambda \|L_{style} - L_{attack}\| \quad (12)$$

where, λ denotes the hyperparameters in model simulation at the attacker’s side. The difference between actual style-loss and attack style-loss is added as a penalty-term in error estimation. Generally, it is used as $L1$ norm for regularization to generalize learning tasks in CNNs. Now, simplifying the original joint-loss function (Eq. 10) by ignoring total variation loss (which is actually used as a variational regularizer for spatial smoothness in [34]) to relate with style-attack loss function

$$L_{NST}^{original} = L_{content} + L_{style} \quad (13)$$

For a more realistic attack-simulation, the parameters of NST at the designer’s side as well as at the attacker’s end should be almost identical, *i.e.*, $\lambda \approx \gamma \approx 1$. Because, we have simplified and approximated Eq. (10) and Eq. (13), as $\gamma \approx 1$. Hence, the total attack loss can be simplified from Eq. (12) as

$$\begin{aligned} L_{total}^{attack} &= L_{content} + \lambda L_{style} + (1 - \lambda) L_{attack} \\ &= L_{content} + L_{style} + (1 - \lambda) L_{attack} \\ &= L_{content} + L_{style} + (\gamma - \lambda) L_{attack} \\ &= L_{NST}^{original} + \Delta L_{attack} \end{aligned} \quad (14)$$

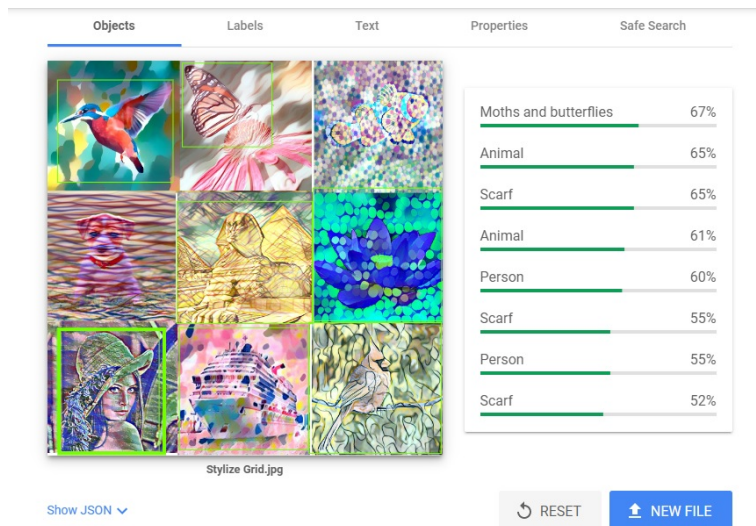


Fig. 12: Object detection by Google Vision AI API

where, λ denotes the hyperparameters in model simulation at the attacker's side and γ is the hyperparameters that estimate a trade-off between the content and style in the rendering process at the designer side. Δ denotes a marginal difference ($\gamma - \lambda$) between hyperparameters.

The objective for a latent-layer-based attack should incorporate an efficient optimization of all types of model parameters and regularization such that $\Delta = 0$, yielding $L_{total}^{attack} = L_{NST}^{original}$.

However, γ (set of actual hyperparameters in NST) and λ (set of adaptive attack-model's hyperparameters) are different parameters, very sensitive, and random by nature. Hence, the attack model should be very sophisticated and efficient for achieving excellent attack success which is obtained in text-CAPTCHAs and related other NST-methods easily. Mean square error (MSE) estimates error in actual NST outcome with adapted latent-layer output, and the results are given in Fig. 10 and Fig. 11.

The Visual Geometry Group (VGG)-19 networks receive both style and content inputs, and each image's feature representation is separately derived from a distinct layer. We can obtain distinct stylized outputs from different intermediate layers for a specific content and a style input pair. In Fig. 10, we have calculated MSE value of stylized images, which are obtained from various combinations of style and content images. We have chosen a few similar examples of styles. On the left side, conv-1 layer output is used for content, and conv-2 layer output is used for style. On the right side, conv-4 layer output is used for content and conv-7 layer is used for styled output.

After obtaining all this intermediate stylized output, we have tested the general DL attack simulation of Type-III whether it is possible to extract style information from these stylized images obtained from latent layers. The accuracy is 14% which is permissible compared to other works. It implies style recognition from intermediate layers of stylized images is a difficult task.

Faster R-CNN is a deep convolutional network that is presented to the user as an integrated, single, and complete network for object detection. It is capable of accurately and rapidly predicting the positions of various objects. It is a fast and efficient object detector [51]. It detects and recognizes an object within the content image. However, the confidence score degrades if the same content image is blended with a stylized effect via NST. Table 4 shows the result of content/object detection from stylized images using Faster RCNN. The contents are detected but recognized incorrectly. Faster RCNN can not recognise the objects fish and dog in Table 4 and Fig. 3. However, it recognizes other object classes such as flower, human face, and ship with certain confidence scores. In Fig. 13, Faster RCNN is applied to an image-grid, and five out of nine images are detected correctly while the remaining

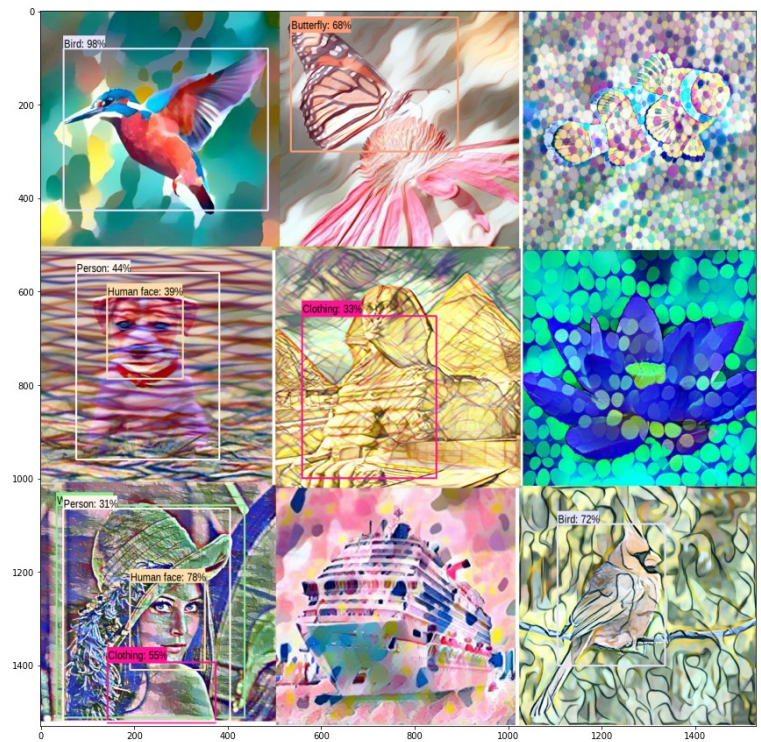


Fig. 13: Object detection in image-grid by Faster RCNN

Table 4: Object Detection performance of Faster RCNN using SMC challenges

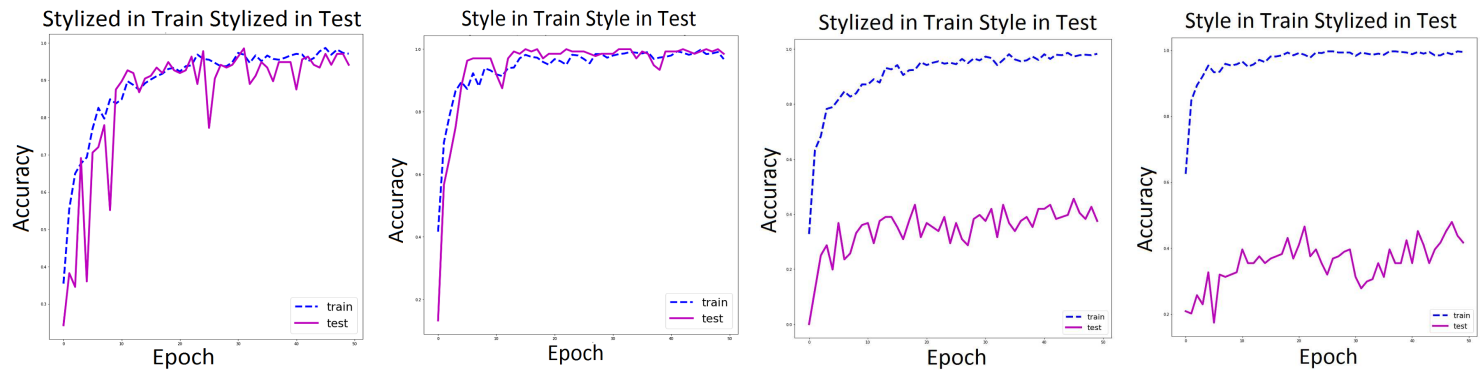
Object	Correct Recognition	Wrong Recognition
Fish	0%	100%
Dog	0%	100%
Flower	87%	13%
Human Face	97%	3%
Ship	14%	86%

images are not detected. Faster RCNN uses region proposal networks (RPN) to select the regions for pooling. In Faster R-CNN, RPN is trained such that all anchors in a mini-batch of size 256, are extracted from a single image. For a single image, the features are correlated and easier for convergence while it is difficult for a blended or stylized image. As a result, though the correct region can be found, but the correct classification is not easily possible. Particularly, it is observed that the degree of recognition declines when the style images are dark and bold in nature. Sometimes, the dark lines within the style image can be recognized as content. Thus, carefully chosen style images (during dataset creation) can have a significant impact on contents during the stylization process in NST.

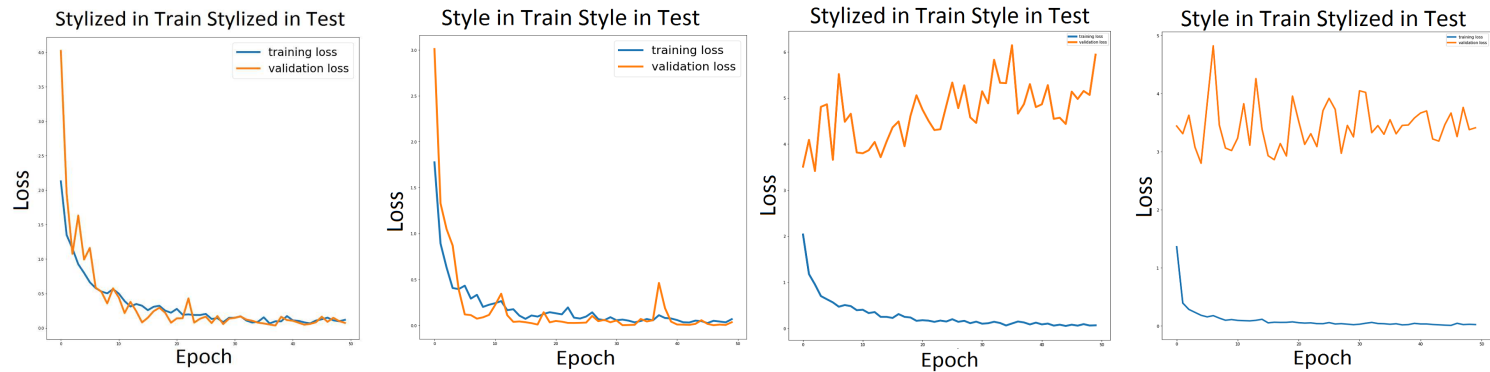
The Google Cloud Vision AI API makes it effortless for developers to incorporate vision recognition capabilities into their applications, such as distinguishing between images, locating faces and landmarks, recognizing text with OCR, and marking explicit content. Google vision AI also fails to detect the images in the same image-grid, shown in Fig. 12. It justifies our objective to match the styles rather than objects.

4.3 Randomness in Stylization

To investigate the randomness in the stylization, we have computed the Structural Similarity Index (SSIM) and cosine similarity scores between the input styles and rendered stylized outputs. For this test, ten randomly selected classes of style images and stylized output images using ten different contents with the selected 10 styles.



(a) Determine the accuracy of four different circumstances using both style and stylized images for training and testing. Number of epochs: 50 and learning rate: 0.02. The results of the first two are obvious. Our prime interest is in the last two. The results are in Table 3.



(b) Attack loss when stylized images and style images are used for training and testing

Fig. 14: Attack Simulation: Accuracy and loss by Inception-v3 model for stylized and style images of size 256×256

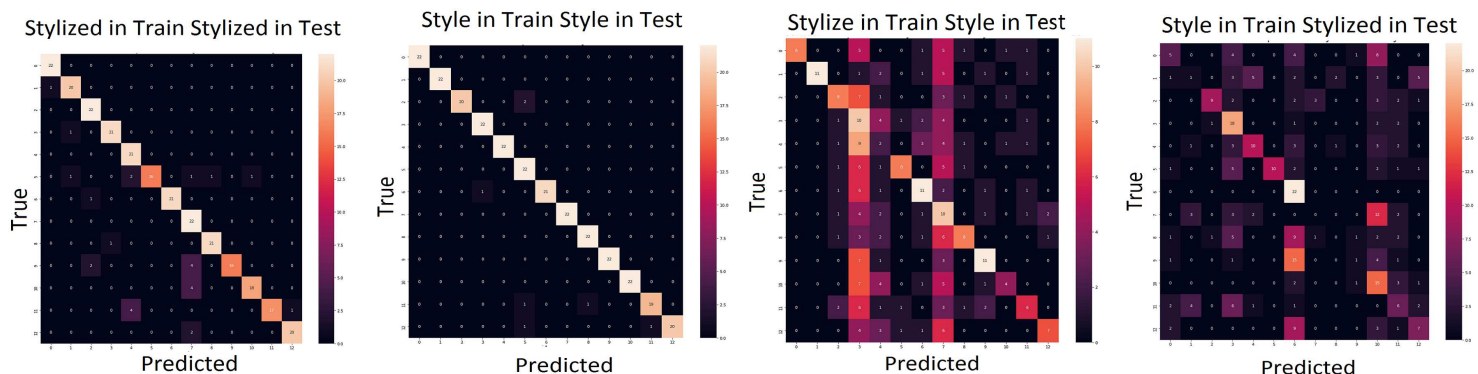


Fig. 15: Confusion matrix of various threat model simulation with image-size 256×256 .

In Fig. 16, we have plotted the SSIM and cosine similarities between these two input categories in the NST using heatmaps. It is evident from the heatmap that intra-style (same style, different content) and inter-style (different style, same content) rendering are highly random to reproduce similar outputs. For a particular style (intra) and 10 different contents, the SSIM value varies between 0.218 and 0.524. Likewise, inter-style variations lie within 0.337 to 0.369. Similar variations are also observed using cosine similarities. It is hard to maintain a trade-off between security and usability in CAPTCHA design. Any particular scheme cannot be resilient to the most possible attacks. Thus, it is considered as an open problem for robust security analysis.

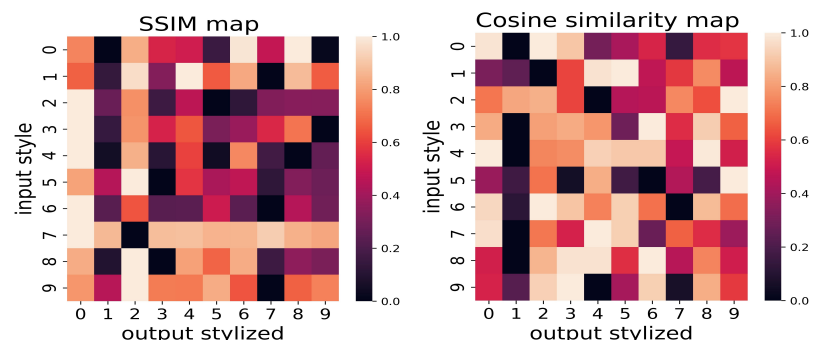


Fig. 16: SSIM and Cosine similarity between output stylized and input styles.

4.4 Style Extraction from SMC

Style extraction from stylized images is a major concern for attack analysis on SMC. We have carried out style transfer as stated in Section 3.2. Here, simply,

a white image is considered as a content image, and the stylized image is used as a style image. The output of our experiment is depicted in Fig. 17. We have

observed that the original style is not accurately regenerated. However, the colours and textures are likely to be reproduced to a certain extent akin to the original one. Inspired by this direction, we plan to conduct an in-depth study in near future, such as the suitability of auto-encoder and decoder architecture.

4.5 Resilience of SMC over Denoising

Few recent schemes has applied denoising for breaking the text-CAPTCHAs [39]. Likewise, we have tested several types of noise added to the SMC to offer an extra layer of security, shown in Fig. 18. Particularly, we have included the Gaussian filter, median filter, and DnCNN [78] to remove the noise from images. Next, we have simulated a deep-learning attack on these denoised CAPTCHAs to evaluate the robustness and security of our proposed system. On the stylized images, a few conventionally noisy patches, such as circles, arcs, shapes, and lines, as well as periodic and style noises are included.

4.5.1 Effects of additional noises on SMC

A wide variety of noises with various sizes and colours, including random lines, random shapes, periodic noise, and blended noise are included over the SMC challenge for further study, depicted in Fig. 18. These types of noises deliver certain additional strengths, such as resilience against object segmentation attacks, and other schemes based on fundamental image pre-processing methods. However, in our case, the original style/pattern image can still be recognised by a human user, even after incorporating complex and random patterns and noise. In Fig. 18(b).i, there are several different coloured inclined lines, as well as thicker and thinner horizontal and vertical lines as noise. Random-shaped objects, including circles, squares, and triangles of different colours and sizes, are superimposed in Fig. 18(c).i. Similarly, Fig. 18(d).i includes periodic noise that can be from different directions. In order for users to recognise the style of a stylized image properly, the lines of periodic noise should be narrow. Fig. 18(e).i depicts blended-style noise on stylized images. Random styles are blended over the targeted-styled image with a low opacity. This increases the security of the SMC algorithm by making it more difficult to recover the actual stylized image.

4.5.2 Denoise Techniques

We have applied several denoising methods for simulating deep learning attacks on noisy stylized images. In our case, cleaning the image is not an easy task due to the large variation in the noise applied to the images. We have attempted to use some denoising methods implemented in Matlab (Version R2020a), which are as follows.

- Gaussian filter (a low-pass filter) is used to blur certain areas of an image and reduce noise (high-frequency components). To attain the desired result, the filter is implemented as an odd-sized symmetric kernel which is passed over each pixel of the region of interest. The effect of applying the Gaussian filter to the noisy images is depicted in Fig. 18.(b),(c),(d),(e). A Gaussian filter can effectively remove the salt-and-pepper noise from the images. However, it cannot easily remove the noises that have been used in the proposed SMC algorithm. This type of filter has futile effects in denoising our noisy stylized images.
- Median filter is frequently used to eliminate noise from an image. It may sometimes preserve edges while reducing noise. However, the median filter is unable to eliminate the noise that we have applied, as shown in Fig. 18.(b),(c),(d),(e). However, in a wider perspective of the image examples, it outperforms the Gaussian filter. In some instances, the colour or texture of the shapes has been altered slightly. As a result, the median filter is ineffective for denoising our approach.

Table 5: The SSIM index and MSE value are used to compare stylized and denoised images.

Noise types	Gaussian Filter		Median Filter		DnCNN	
	SSIM	MSE	SSIM	MSE	SSIM	MSE
Random line	0.742	0.492	0.686	0.811	0.659	0.526
Random Shape	0.807	0.381	0.545	0.601	0.829	0.397
Periodic noise	0.534	0.706	0.346	0.908	0.311	0.998
Blended style	0.597	0.559	0.536	0.636	0.595	0.578

- A pre-trained, simplest, and quickest denoising convolutional neural network is DnCNN [78].

It uses single-channel images as its input. To eliminate the noise, we have divided the noisy RGB image into three distinct colour channels and employed a DnCNN. The denoised RGB image is created by recombining the three denoised colour channels. However, the noise we have applied cannot be eliminated by the DnCNN network either, as shown in Fig. 18.(b),(c),(d),(e).

Following the denoising operation, we have compared the denoised images with the original stylized images (from Fig. 18). The SSIM index and mean square error (MSE) are computed for comparison, and the results are shown in Table 5. We have observed that the SSIM value is on the lower side, while the MSE values are on the higher side. It indicates that noise reduction techniques are not efficient to remove the noises employed on stylized images of SMC.

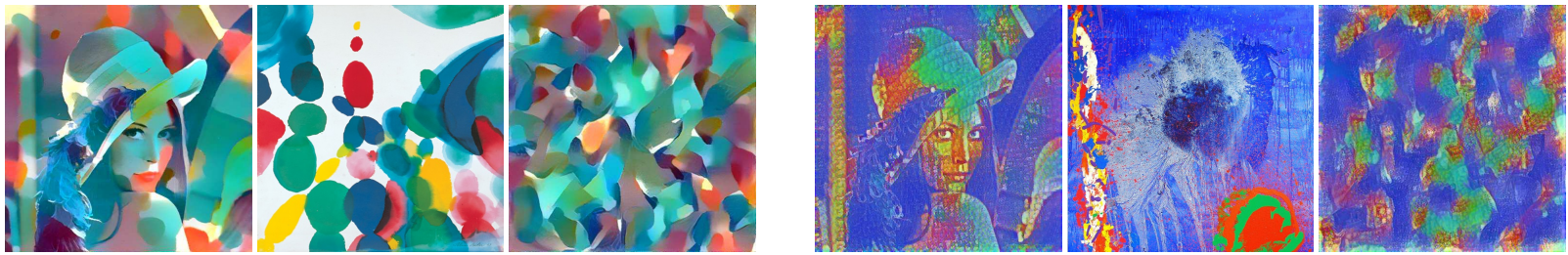
4.5.3 Post-Denoising Attack analysis

To adhere insights on the denoising schemes, we have performed another attack analysis on the denoised stylized images and style images as described in Section 4.1. As shown in Table 3 (Type-III and Type-IV), we have selected 1950 denoised style images and 1950 denoised stylized images for the attack analysis after the denoising procedure. The training and test images are split in an 80:20 ratio, and each image is resized to 128×128 pixels. ResNet-50 and Inception-v3 are used as CNN backbones and are trained for 50 epochs with a mini-batch size of 8. The stochastic gradient descent (SGD) optimizer with a 0.01 learning rate is used for training. The result clearly shows that it is very difficult to attack successfully, if we apply user-defined noises in the images. Although, the noisy style and stylized images can be easily recognized by human users as described earlier.

After applying the noises, overall test accuracy decreases significantly for Type-III and IV schemes by 23% to 34% (Table 3). Indeed, this is a substantially lower success rate of an attack, whereas a higher success rate has been attained to break other schemes (like Captchastar[14] or Deepcaptcha[47]). It demonstrates that with the mild use of standard noises in our SMC scheme, it is very hard to achieve a higher success rate on our threat model.

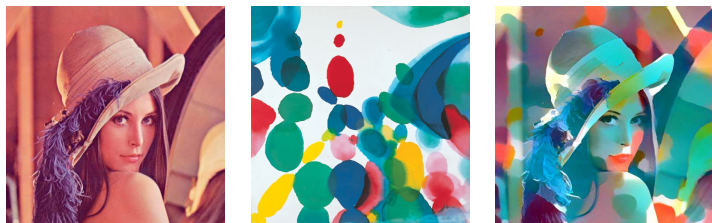
4.5.4 Denoising: A Comparison with Text-CAPTCHA

In SMC, stylized images are our main concern. In Fig. 18, we have applied various user-defined noises to the stylized image and tried to denoise it with some of the available denoising filters. However, we couldn't eliminate those noises. It means denoising operations might not be applicable to SMC. In text-CAPTCHA, the main objective is to recognize the distorted text/object. Whereas, in SMC, we need to recognize or extract the style for deep learning attacks. We have created some text-CAPTCHAs samples and applied random lines as noises, as shown in Fig. 19. After several image processing operations, denoising becomes marginally effective to recognize the embedded-texts. Likewise, we have applied the same noises and denoising procedures for our stylized image. The resulting binary images could not represent the input style/pattern information. It evinces that effective denoising operations for



(a) Stylized image, Original style image, Recovered style image are in pair

Fig. 17: Style recovery from stylized image



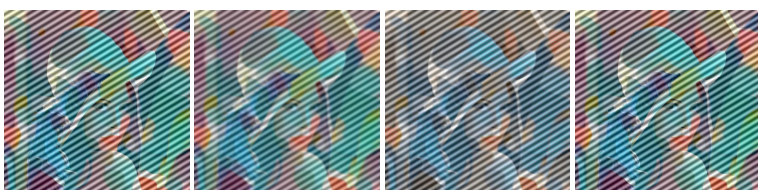
(a) Left to right i. Content image, ii. Style image, iii. Stylized image



(b) Left to right: i. Stylized image with random Line Noise, ii. After applying Gaussian Filter, iii. After applying Median Filter, iv. After applying DnCNN



(c) Left to right i. Stylized image with random Shape Noise, ii. After applying Gaussian Filter, iii. After applying Median Filter, iv. After applying DnCNN



(d) Left to right i. Stylized image with periodic Noise, ii. After applying Gaussian Filter, iii. After applying Median Filter, iv. After applying DnCNN



(e) Left to right i. Stylized image blended with another style, ii. After applying Gaussian Filter, iii. After applying Median Filter, iv. After applying DnCNN

Fig. 18: Row-wise (b-e): stylized images after applying noises are shown at the left-most side, and the other images are obtained after denoising effects.

text-CAPTCHA are not always apposite for solving SMC. This observation is our rationale to develop SMC.

4.6 Object Segmentation and Detection Attack

SMC does not enquire to identify, localize or detect foreground object(s), which is mainly followed by other existing methods. Instead, we pose our challenge to match the pattern from a highly blended styled image. Hence, our method can not be solved by object segmentation and detection tools. Also, SMC is resilient to well-known image processing tasks such as boundary/edge detection, noise removal, pixel-level segmentation, etc. It is a major benefit of our proposal.

4.7 Random Guess Attack

The style-grid is a 3×3 image matrix, and three answers are essential to solve an SMC challenge, one per row for each session. So, there are 3 possible cases of random guessing attacks.

- In the case of the global selection of three styles from all 9 in the grid, the probability of a random guessing attack is $(9 \times 8 \times 7)^{-1} = 0.00198 = 1.98 \times 10^{-3}$. Alternatively, the probability can also be computed using a general combination rule ${}^9C_3 = \binom{9}{3} = 0.0119 = 1.19 \times 10^{-2}$. For the row-wise selection of one style (per row), the probability of a random guessing attack is $3^{-3} = 0.0370 = 3.70 \times 10^{-2}$.
- Now, considering the size of the style-grid is 500×500 pixels, and each style pattern varies randomly within 100×100 to 160×160 pixels, with an average of 130×130 pixels. Using these spatial dimensions, simply applying a global selection strategy, the probability of random guessing is

$$\frac{130^2}{500^2} \times \frac{130^2}{500^2 - 130^2} \times \frac{130^2}{500^2 - 2 \times 130^2} = 3.83 \times 10^{-4}$$

- Also, the row-wise selection of a style from the grid is considered for probability estimation. Considering the same dimension of style-grid and the average height of each row 160 pixels and width of 500 pixels, the probability is

$$\left(\frac{130 \times 130}{500 \times 160} \right)^3 = 9.42 \times 10^{-3}$$

Thus, it is not easy to pass an SMC by random guessing.

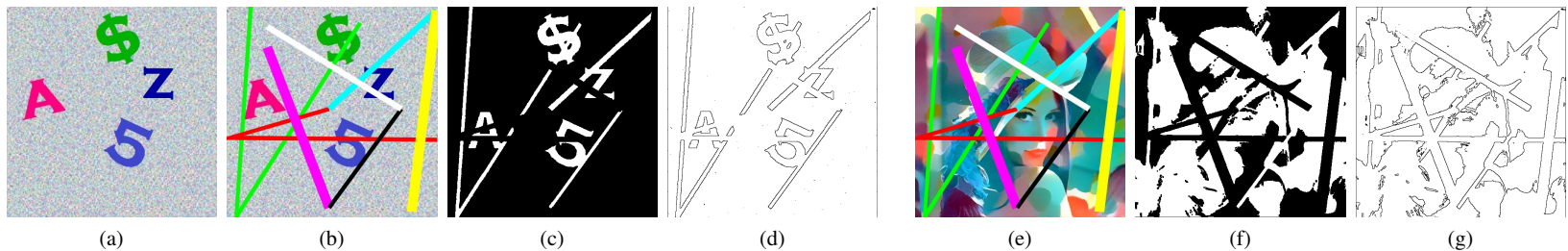


Fig. 19: Noise and denoise operations are applied to a text-CAPTCHA and a stylized image. (a) Sample Text-Captcha, (b) Random lines added as noise, (c) Denoised Text Captcha image in binary, (d) Edge Detection of image-(c), (e) Stylized image with noise, (f) Denoised stylized image in binary, (g) Edge detection of image-(f)

4.8 False Accept Rate (FAR)

It represents the probability of a bot's success to solve SMC. The false reject rate denotes the probability that a human is unsuccessful in solving it. An acceptable 1.5% FAR is bounded in [47]. FAR depends on the number of possible answers n_s (here, n_s is the total number of query styles in SMC) and the number of successful solutions of q challenges by a bot. It is defined as $FAR = (n_s)^{-q}$. In [47], $n_s=8$ and $q=2$ have been considered, resulting in 1.5625% FAR. In SMC, $n_s=9$ and $q=3$, which imply $FAR=0.137\%$ which is much lesser than the 1.5% limit. Also, considering $q=2$, the FAR is 1.234%. Thus, SMC attains better FAR over others.

5 Usability Study and Result Analysis

The human accuracy (%) and solving time (seconds) are computed for the performance evaluation of the users on SMC.

Accuracy: it is determined by the ratio of correct answers to solve SMC and the total number of responses.

$$Accuracy(\%) = \frac{Correct\ Responses}{Total\ Number\ of\ Responses} \times 100 \quad (15)$$

Solving Time: time (in seconds) taken to solve an SMC.

5.1 Experimental Setup

Our SMC algorithm is implemented using PHP and MySQL and has been exhaustively tested on web browsers. To further analyze the effectiveness of the algorithm, we have conducted a usability study. For this purpose, we contacted our departmental students, faculty, and staff members and asked them to volunteer for the usability study. In order to facilitate the study, we have installed XAMP (Version 8.0.25) and SMC on 30 PCs. The volunteers are also present during the test sessions to monitor the proceedings. The student volunteers, faculties, and staff members of various departments from different institutions have participated in response and feedback collection by solving a set of random SMC queries. There are no businesses or sponsors involved in this research. The participants generously contributed their time and feedback to us. We have guided them through an illustrative session about SMC challenge solving and providing their feedback.

The participants acknowledged that they understood our research objectives and gave their consent to participate before the response collection process began. It should be noted that there is no financial business involved in this research. To ensure the user's privacy, their details are collected anonymously while maintaining ethical considerations. The participants agreed to

Table 6: Group Division based on User's Age and Gender

Age / Gender	Group	No. Users	Users in %
8 - 16 yrs	A	7	4.61
17 - 30 yrs	B	95	62.50
31 - 50 yrs	C	40	26.32
Above 50 yrs	D	10	6.58
Total Female	F	67	44.08
Total Male	M	85	55.92

share their overall experience during the response collection, which is invaluable to our research. We take great care to ensure that all privacy and ethical issues are addressed in our research efforts. Altogether 152 persons (male: 85 and female: 67 users) with various age groups between 8-65 years have participated in the evaluation task. Table 6 provides information about their age-group and gender. Fig. 20 depicts the user completing the SMC challenge on a PC and Smartphone/mobile devices. In both cases, the correct styles are selected and highlighted with a coloured rectangle. After correctly submitting the responses, the SMC is verified successfully. Additionally, Fig. 20 also describes how the users are submitting their responses in the departmental laboratory. The user responses are collected from a local server in a structured datasheet for analysis.

We have observed that many participants are already accustomed to using PCs and/or mobiles to solve CAPTCHAs on various websites. Additionally, a few young children and elderly people lack any prior knowledge of how to solve a CAPTCHA. A brief description of the solution procedure of an SMC challenge is demonstrated to the participants by a group of 15 student volunteers who are involved during response collection. Next, each participant is requested to solve three SMC at three different sessions, *i.e.*, 9 answers from each user. A total of 1368 (9×152 users) responses are recorded accordingly. The human performance in solving SMC is given in Table 7 and Table 8. Lastly, the participants have provided their remarks through a feedback form. Following the collection of all responses and feedback, we generate a final datasheet for our analysis.

Additionally, SMC can easily be deployed on mobile devices (SMC-App) for a brief usability study. The responses are provided by 30 participants in our department laboratory. We have conducted a similar user-friendliness and satisfaction survey to collect their feedback. We have observed that the participants easily understood SMC-App and solved it within a reasonable time. We conducted all of the studies while maintaining ethical concerns and without any financial objective. The details are described in section 5.4.

5.2 Solving Accuracy

The average human accuracy (%) in solving SMC (Eq.15) is 95.6% (Table 7). It improves in successive sessions when the users are familiar with the

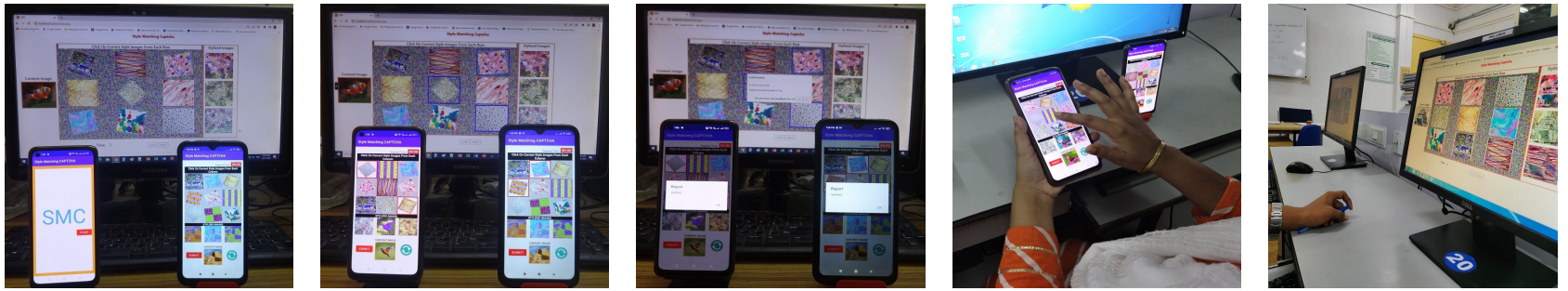


Fig. 20: Usability testing of SMC at the departmental laboratory using PCs and mobiles

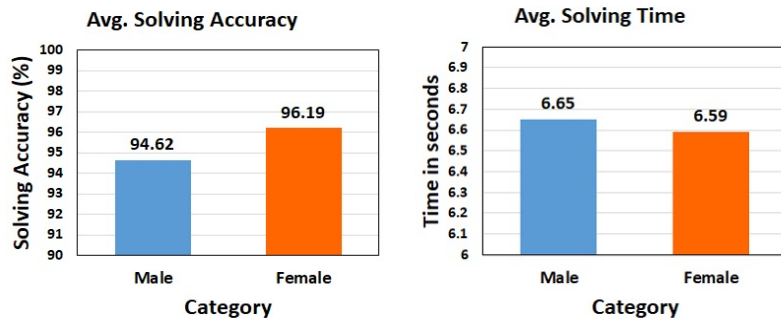


Fig. 21: Average solving accuracy and average solving time of SMC from an equal number of males and females.

solving technique. However, there is a slight variation in accuracy between the sessions. It is interesting that humans take much less time to recognize an image even though it is distorted than typing the text (6-8 characters) to solve a text-CAPTCHA or a cognitive question. Fig-22(a) displays the accuracy of the solving SMC of all 152 participants in three different sessions category-wise. We observed that the accuracy is lower for category D, which is the group of aged people. The rest of the groups performed much better at solving SMC. The complete details of human accuracy to solve SMC are described in Table 7. In Table 9, we have compared the solving accuracy for SMC with an equal number of male and female participants. We have observed that solving accuracy by the female (96.19%) is better than the male (94.62%). We have also compared the performance of the individuals who are accustomed to solving CAPTCHA with those who are not.

The result of these evaluations are given in Table 10. Those who are familiar with CAPTCHA have an average solving accuracy of 96.53%, while those who are unfamiliar have an accuracy of 95.90% only. It is evident that those who are familiar with CAPTCHAs perform a little bit more accurately. However, people who are unfamiliar with CAPTCHAs are also performing quite well considering their lack of experience. This indicates that CAPTCHAs are not overly difficult to decipher, even for those with no prior knowledge. The results of the evaluation suggest that familiarity with CAPTCHAs does offer a slight advantage in terms of solving accuracy. However, it is important to note that the difference between the two groups is not particularly large. This implies that a basic understanding of CAPTCHAs is sufficient for most users to be able to decode them. Furthermore, it is clear that the majority of users are able to employ the knowledge they possess to accurately solve the CAPTCHAs presented to them.

5.3 Solving Time

The participant's average solving time in seconds (s) to answer each SMC query at three different sessions is 6.59s. Additionally, Table 8 provides de-

tailed information on the timely results of different categories of participants in order to solve SMC. Fig. 22(b) shows the chart, where the average solving times of all 152 participants are plotted category-wise. It is observed that participant's skills have improved as they are solving more SMC at various experimental sessions. It is evident from Table 8 that the timely test result of category-A participants is excellent for solving SMC within 4.91 seconds during experiment-3 on session-1, which is minimal.

In Table 9, we have compared the solving time for SMC with an equal number of male and female participants. We have observed that the male's solving time (6.65s) is better than the female's (6.59s). We have also compared the average completion time of the individuals who are accustomed to solving CAPTCHA with those who are not. Table 10 shows the results of this comparison. Those who are used to solving CAPTCHAs had an average completion time of 6.65 seconds, while those who were not were slightly slower at 7.65 seconds. This indicates that those with more experience in dealing with CAPTCHAs are able to solve them more efficiently. However, the fact that those who are not familiar with CAPTCHAs were still able to complete the task implies that CAPTCHAs are not overly complex and can be solved by anyone with a reasonable level of understanding. In addition to this, the results also suggest that it is possible to improve the accuracy of CAPTCHAs by allowing people to become more familiar with them. By providing tutorials and other resources to help new users become accustomed to solving CAPTCHAs, it is possible to reduce the completion time while also increasing accuracy.

Thus, the solving time could effectively prevent the bots by imposing a time limit for responding to an SMC query. More visual explanation (*e.g.*, histogram analysis and standard deviation on answering time taken by the participants) is given in Fig. 23.

5.4 Usability and Feedback Analysis on SMC-App

In addition to web-based usability testing via PC on SMC, we have conducted a rapid usability test with 30 users to assess the performance of our SMC-App. We have organized a comprehensive setup with 10 Android devices at our departmental laboratory. The SMC-App is installed on every mobile device and the volunteers have demonstrated the SMC-solving procedure through the SMC-App, *i.e.*, how it works, how to solve it, etc. to the participants. After a brief discussion, the participants submitted their responses three times at the allotted sessions. Each session is thoroughly monitored by our team of volunteers in order to ensure the accuracy of the results. Additionally, the feedback from the participants is documented and analyzed to identify areas of improvement in the SMC-App. This enabled us to make the necessary changes and further optimize the usability, performance, and user experience of the SMC-App.

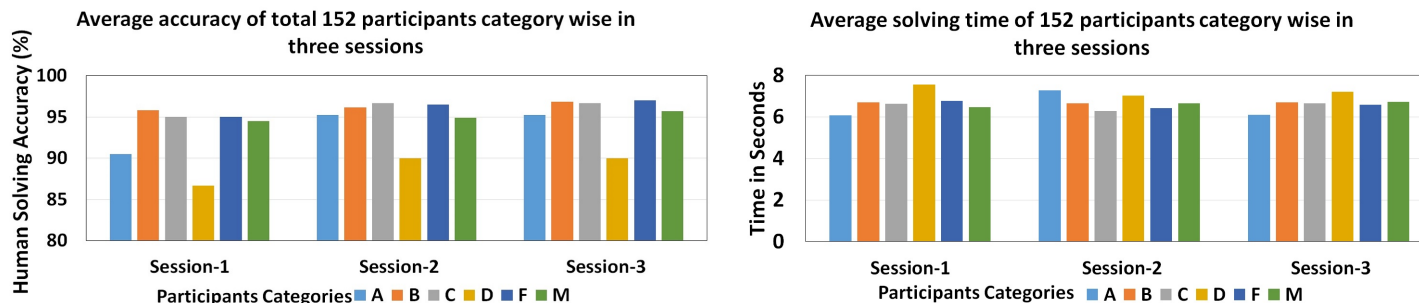
Our volunteers have conducted three sessions, during which they have collected a total of nine responses from each participant and received feedback

Table 7: SMC solving accuracy (%) by the participants at various sessions

Category	Session-1			Session-2			Session-3			Average
	EXP-1	EXP-2	EXP-3	EXP-1	EXP-2	EXP-3	EXP-1	EXP-2	EXP-3	
A (8-16 yrs)	85.71	100.00	85.71	100.00	100.00	85.71	100.00	85.71	100.00	93.65
B (17-30 yrs)	94.74	96.84	95.79	93.68	96.84	97.89	94.74	97.89	97.89	96.26
C (31-50 yrs)	90.00	97.50	97.50	100.00	95.00	95.00	97.50	95.00	97.50	96.11
D (>50 yrs)	90.00	80.00	90.00	100.00	90.00	80.00	90.00	90.00	90.00	88.89
F (Female)	91.04	97.01	97.01	95.52	95.52	98.51	94.03	97.01	100.00	96.19
M (Male)	94.12	95.29	94.12	95.29	96.47	92.94	96.47	95.29	95.29	95.03

Table 8: Time (s) taken to solve SMC by all participants at various sessions

Category	Session-1			Session-2			Session-3			Average
	EXP-1	EXP-2	EXP-3	EXP-1	EXP-2	EXP-3	EXP-1	EXP-2	EXP-3	
A (8-16 yrs)	7.10	6.22	4.91	6.34	8.09	7.39	7.42	5.88	5.01	6.48
B (17-30 yrs)	8.01	6.72	5.37	6.84	6.69	6.43	7.11	6.74	6.27	6.69
C (31-50 yrs)	7.36	6.59	5.95	6.16	6.41	6.26	6.85	6.91	6.20	6.52
D (>50 yrs)	8.66	7.46	6.57	6.71	7.11	7.25	8.12	7.16	6.36	7.27
F (Female)	8.32	6.58	5.41	6.42	6.59	6.25	6.82	6.84	6.11	6.59
M (Male)	7.53	6.65	5.25	6.55	6.76	6.63	7.37	6.72	6.06	6.61



(a) Average Accuracy(%) Category-wise of 152 participants

(b) Average Solving time of total 152 participants category wise

Fig. 22: Participant’s average accuracy(%) and average solving time(s).

Table 9: Performance comparison with an equal number of male and female participants.

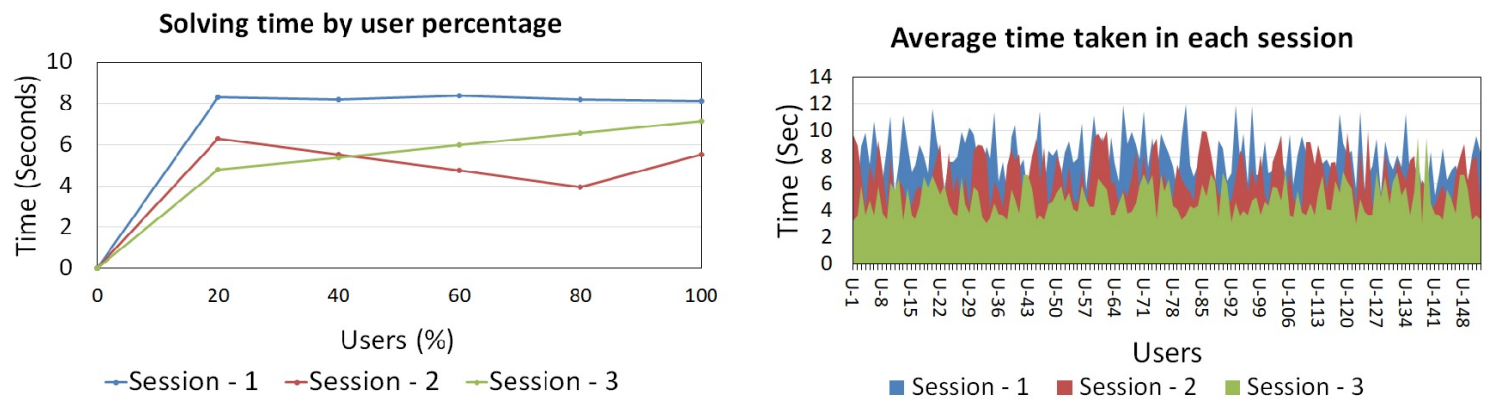
Category		Session-1			Session-2			Session-3			Average
		EXP-1	EXP-2	EXP-3	EXP-1	EXP-2	EXP-3	EXP-1	EXP-2	EXP-3	
Male (67)	Solving Accuracy	93.98	94.42	93.81	95.11	95.86	92.54	95.97	95.04	94.88	94.62%
Female (67)	Solving Accuracy	91.04	97.01	97.01	95.52	95.52	98.51	94.03	97.01	100.00	96.19%
Male (67)	Solving Time	7.43	6.65	5.31	6.58	6.81	6.71	7.51	6.66	6.24	6.65s
Female (67)	Solving Time	8.32	6.58	5.41	6.42	6.59	6.25	6.82	6.84	6.11	6.59s

Table 10: People who are familiar with solving CAPTCHA vs. those who are not familiar with solving CAPTCHA.

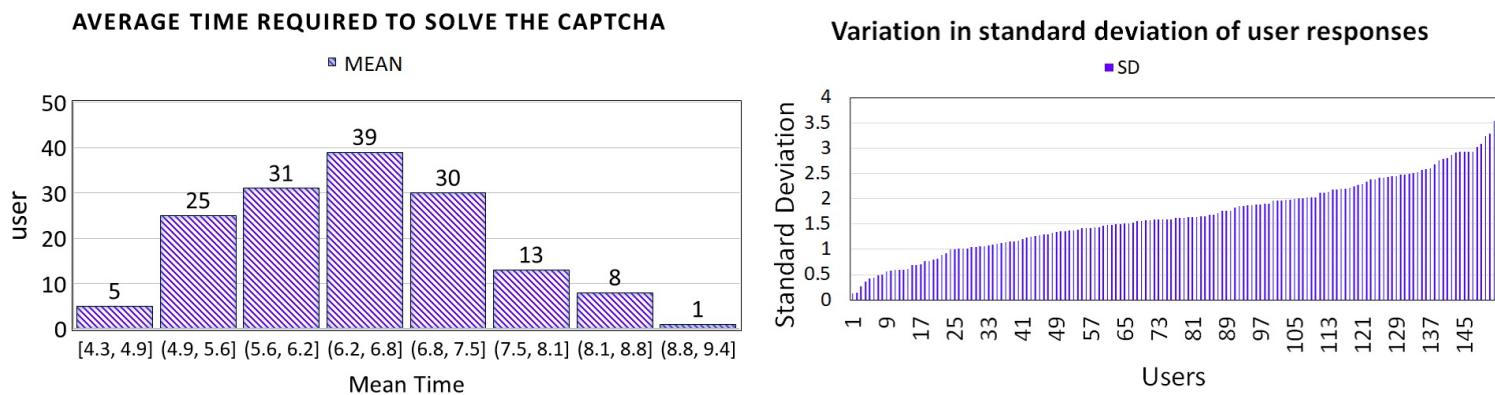
Category		Session-1			Session-2			Session-3			Average
		EXP-1	EXP-2	EXP-3	EXP-1	EXP-2	EXP-3	EXP-1	EXP-2	EXP-3	
Familiar (117)	Solving Accuracy	95.86	96.02	96.22	97.25	96.54	96.73	96.35	96.83	97.05	96.53%
Not-familiar (35)	Solving Accuracy	95.02	95.52	95.82	96.01	95.87	96.11	96.32	96.03	96.47	95.90%
Familiar (117)	Solving Time	7.51	7.11	6.69	6.71	6.54	6.22	5.86	5.91	5.78	6.48s
Not-familiar (35)	Solving Time	8.25	8.12	7.95	7.91	7.64	7.51	7.12	7.23	7.15	7.65s

Table 11: Usability study on SMC-App regarding solving time and solving accuracy

No. Exp	Session-1		Session-2		Session-3		Average Solving Time (s)	Average Accuracy (%)
	Solving Time	Accuracy	Solving Time	Accuracy	Solving Time	Accuracy		
EXP-1	7.41	94.58	5.19	96.75	4.58	96.26	5.72	95.86
EXP-2	5.54	94.96	4.86	96.41	4.26	97.87	4.88	96.41
EXP-3	5.35	95.43	4.88	97.06	4.15	97.72	4.79	96.73



(a) (i) Time taken to solve random challenges by the participants in three experiments; (ii) Overall time taken by each user implying how the response time improves at different test sessions.



(b) (i) Histogram representation of time taken in solving SMC by the number of participants; and (ii) Standard deviation of response time for the users in sorted order.

Fig. 23: Performance evaluation on SMC.

Table 12: Post-test questionnaire on SMC responses by PC and mobile devices (SMC-App)

SI No.	Questionnaires	Feedback on PC's		Feedback on Mobile Devices	
		Mean	Standard Deviation	Mean	Standard Deviation
1	SMC provides a good user-friendly and interactive interface	9.30	0.61	9.43	0.31
2	SMC is resilient against different types of attacks	8.51	1.03	8.63	1.17
3	Organization of information on the SMC screen is clear	8.97	1.01	9.01	0.65
4	Solving time is satisfactory	8.84	1.12	9.11	0.83
5	SMC is more reliable than other image-CAPTCHAs	9.21	0.98	8.76	1.21
6	SMC relies on natural inherent cognition of human	8.80	0.98	8.96	1.05
7	SMC is easily understandable and recognizable	8.73	0.62	9.27	0.48
8	SMC is simple and easy to utilize	9.08	0.67	9.18	0.46
9	Comfortable while solving SMC	9.21	0.54	9.23	0.41
10	Successfully completed SMC verification	9.12	0.51	9.25	0.68

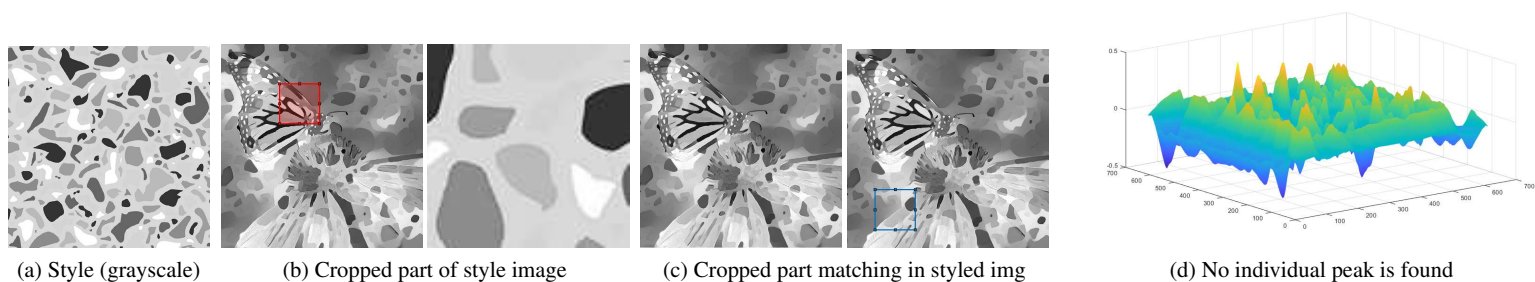


Fig. 24: Cross-correlation between the style and stylized images.

Table 13: Performance comparison with various image-CAPTCHAs

Reference	Accuracy (%)	Response time (s)	Probability	Dataset Used	Scheme Attacked By
ARTIFACIAL [53]	99.7	14.00	2.00×10^{-1}	3D wire model of a generic head and a 512×512 cylindrical texture map of an arbitrary person.	Zhu et al.[81] with a success rate of 18%
ASSIRA [19]	83.0	15.00	5.00×10^{-1}	total, 23,208 cat and dog images and 332 test subjects are used.	Zhu et al. [81] with a success rate of 82.7%
CONSCHEME [79]	85.1	15.12	7.69×10^{-2}	310 number of images, each size is 450×350 pixels, and the maximum radius of deformed regions is 50 pixels.	NA
DeRection [79]	86.4	12.77	4.60×10^{-3}	NA	NA
HandCAPTCHA [6]	98.6	9.67	4.68×10^{-1}	600 background images, hand images of 500 persons, and 400 fake hand images.	NA
SACAPTCHA † [63]	92.1	10.12	4.80×10^{-3}	6000 SACAPTCHAs are produced. 4000 are used for the training set, 1000 used for validation and the remaining 1000 are used for testing.	Rathor et al. [49] with the success rate of 96% and 82% for two separate datasets,
Grid CAPTCHA † [13]	75.0	11.83	7.50×10^{-5}	60,000 icons in 188 categories. The icon size is 112×112 and the grid size is 344×334 .	NA
DeepCAPTCHA [47]	86.7	7.66	7.00×10^{-1}	1000 adversarial images for the MNIST and ILSVRC-2012 datasets.	NA
Annulus [21]	89.0	8.89	3.10×10^{-3}	NA	NA
CAPTCHaStar [14]	90.2	<27.00	9.00×10^{-2}	NA	Gougeon et al.[27] with a success rate of 96%.
Proposed SMC (Web-App)	96.6	6.52	3.83×10^{-4}	Elba Datasets and other 2k style images from Kaggle’s Abstract Art Gallery. 2K Content images and 3K stylized images.	NA
Proposed SMC (Smart-App)	96.3	5.13	1.49×10^{-3}	Same as above	NA

on the SMC-App. After collecting all of the information, our volunteers have accumulated it for further processing and analysis. Upon examination of the data and responses from the mobile devices, we have determined that the users’ performance has been highly satisfactory, which is an incredibly positive outcome.

Interestingly, better performance has been attained through SMC-App than SMC (web version) in PCs. Table 11 shows the performance of the users in solving SMC on their smartphones. The average solving time of 30 users is 5.13 seconds. It shows that the performance via App is faster than PC-based testing. Also, the average accuracy is calculated 96.33% which implies an improvement too. After response collection, the participants provided their experiential feedback. The users are asked to submit a score on 10-scale for each of the ten questions. Table 12 shows the mean score and standard deviation for each question of the survey on SMC-App.

5.5 Feedback Analysis

In order to ensure the greatest ease of use and convenience, we have kindly requested each participant to provide their feedback. Following the conclusion of the answering sessions, all participants have supplied their feedback in accordance with eight questions, with each question being rated out of ten. A higher score denotes a higher value and quality of response.

Table 12 shows a summary of a general questionnaire and user responses regarding the SMC verification system. The questionnaire was composed of questions about the system’s interface, easiness, reliability, robustness, and solving time. Each question was marked out of ten, with ten being the best score. The responses from the participants indicated that the SMC verification system was satisfactory. A few of them are discussed here. The interface was rated at 9.30 out of 10, with a standard deviation of 0.61. The easiness of the system was rated at 8.73 out of 10, with a standard deviation of 0.62. The reliability of the system was rated at 9.21 out of 10, with a standard deviation of 0.98. Finally, the solving time of the system was rated at 8.84 out of 10, with a standard deviation of 1.12.

These results showed that the SMC verification system was found to be generally satisfactory. The participants found the interface, easiness, language independence, and solving time of the system to be of a high standard. Moreover, their responses also suggested that SMC is able to rely on natural human behavior and easily distinguish between humans and bots. In conclusion, the results from the questionnaire and user responses have been analyzed, and it has been determined that the SMC verification system is highly usable, reliable, and secure. This indicates that SMC could be a viable CAPTCHA system for websites and applications. As the system relies on natural human behavior and is easily distinguishable from bots, it is expected to provide an effective security measure for online services. Additionally, due to its ease of use, reliability, and fast solving time, the system could provide an excellent user experience. Therefore, we are optimistic that the SMC verification system could be a great CAPTCHA system for many different applications.

5.6 Performance Comparison with State-of-the-arts

A comparison with state-of-the-art image-CAPTCHAs is presented in Table 13. Our method offers a well-balanced performance than those methods. Though the accuracy of ARTIFACIAL (99.7%) and HandCAPTCHA (98.6%) is higher than our SMC (96.6%), however, the response time and probability of attacks of SMC are significantly less than these two methods. The SACAPTCHA and Grid-CAPTCHA use NST, like SMC. However, their performances are lower than SMC. ARTIFACIAL and ASSIRA were attacked with a success rate of 18% and 82.7%, respectively in [81]. Whereas CAPTCHaStar was attacked with a success rate of 96% in [27]. Similar to our SMC, SACAPTCHA scheme employs NST and is attacked with a success rate of 96% in [49]. In comparison with these schemes, we have simulated attacks on SMC with a success rate of 34.72%, which indicates SMC is resilient to bots. It is clear that overall performance, including the accuracy (96.6%), the response time (6.52s), and probability (3.83×10^{-4}), our proposed SMC offers a significant improvement over all approaches mentioned in Table 13.

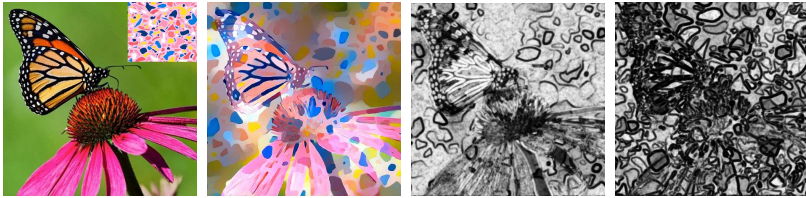


Fig. 25: SSIM comparison between the pairs: stylized vs content and stylized vs style image. Left to right: a) Content with inset style, b) stylized, c) Content vs stylized: SSIM (0.555); MSE (4755.258); Peak-SNR (11.359); and SNR (7.386). d) Style vs stylized: SSIM (0.338); MSE (6562.253); Peak-SNR (9.960); and SNR (5.987).

5.7 Structural Similarity Index (SSIM)

An intuitive image quality metric named Structural Similarity Index, SSIM uses three attributes to quantify visual impact: brightness, contrast, and structure. These three attributes are multiplied to compute the overall index [73]. SSIM compares the content and style images with stylized images in SMC.

$$SSIM(x, y) = [l(x, y)]^\alpha \cdot [c(x, y)]^\beta \cdot [s(x, y)]^\gamma \quad (16)$$

where, $\alpha > 0$, $\beta > 0$ and $\gamma > 0$ are parameters used to adjust the relative importance of the three components.

$$l(x, y) = \frac{2\mu_x\mu_y + e_1}{\mu_x^2 + \mu_y^2 + e_1} \quad (17)$$

$$c(x, y) = \frac{2\sigma_x\sigma_y + e_2}{\sigma_x^2 + \sigma_y^2 + e_2} \quad (18)$$

$$s(x, y) = \frac{\sigma_{xy} + e_3}{\sigma_x\sigma_y + e_3} \quad (19)$$

The μ_x , μ_y , σ_x , σ_y , and σ_{xy} are the local means, standard deviations, and cross-covariance for images x and y . If $\alpha = \beta = \gamma = 1$, and $e_3 = e_2/2$, then the index simplifies to:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + e_1)(2\sigma_{xy} + e_2)}{(\mu_x^2 + \mu_y^2 + e_1)(\sigma_x^2 + \sigma_y^2 + e_2)} \quad (20)$$

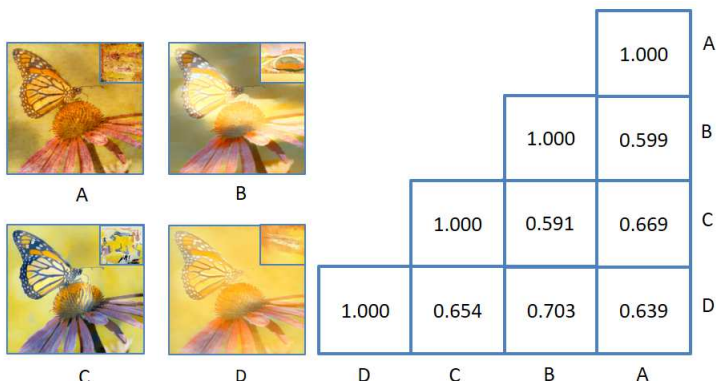


Fig. 26: Comparison of similar type of style images with SSIM values

The SSIM ranges [0,1], where 1 means a perfect match between the reconstructed image with the original one, and 0 implies no ideal match. Fig.

25 describes the differences between the input content and reference stylized images and their differences using various metrics.

The SSIM map is a numeric array of non-negative integers with the same size as the input image that contains local values of the SSIM index. In the local SSIM map, small values represent dark pixels and those values indicate the locations where the input image differs from the reference image significantly. Large values of local SSIM appear as bright pixels. Regions with large local SSIM values correspond to uniform regions of reference images where NST has a small impact. In Fig.25, the SSIM map is displayed for each comparison with style and content image versus stylized image.

Likewise, we have determined the Mean Square Error (MSE) and Peak Signal to Noise Ratio (PSNR) between these images. Our objective is to show how much the content and style images are blended through NST. The SSIM value is near about 0.5 which shows a clear difference between the images. It implies if the attacker has access to a stylized image database, then it is very hard to find the content and style images. In [12], it is evident that we can retrieve the original content from a stylized image if we have style information. However, it's tough to recover input style from the stylized output, which is a wide exploration region to investigate.

5.8 2D Normalized Cross Correlation (NCC)

Normalized cross-correlation (NCC) is a template-matching algorithm in computer vision. It's a common method for determining the degree of resemblance (or dissimilarity) between two images. The key benefit of NCC over traditional cross-correlation is that it's less susceptible to linear variations in light amplitude between two images. It's a simple method for matching two image patches, which may be used for feature identification or as part of more advanced algorithms [40].

For 2D images, template matching uses a reference image which can be a sample of an original image or a synthesized prototype of the pattern for some other applications. The aim is to find if there is an occurrence and where, or at least a similar enough occurrence of the template in the target image. Correlation coefficients are returned as a numeric matrix with values within the [-1, 1] range and are defined as

$$\psi(u, v) = \frac{\sum_{x,y} [I(x, y) - \bar{I}_{u,v}] [\Upsilon(\tilde{x}, \tilde{y}) - \bar{\Upsilon}]}{\{\sum_{x,y} [I(x, y) - \bar{I}_{u,v}]^2 \sum_{x,y} [\Upsilon(\tilde{x}, \tilde{y}) - \bar{\Upsilon}]^2\}^{0.5}} \quad (21)$$

where \tilde{x} represents $(x - u)$ and \tilde{y} represents $(y - v)$, I is the image, $\bar{\Upsilon}$ is the mean of template Υ , $\bar{I}_{u,v}$ is the mean of $I(x, y)$ in the region under the template.

We have compared a region of style (template) with the stylized image (target), in Fig. 24. First, a style is converted to a grayscale image, and a random square-sized region is cropped to check any similarity with the target stylized image. It is observed that NCC cannot find a proper region in the stylized image. From Fig. 24. d, it is observed that there is no optimal peak value. It evinces that there is no significant similarity exists between the template and the target image. The NCC value for the template and target image is a matrix, ranging between ± 1 . From this test, we have computed the maximum value as 0.3921 and the minimum value as -0.43 from the NCC matrix. These values are neither close to +1 nor -1. It signifies no similarity is found between the template and target images. Thus, it is tough to reconstruct the style image from the stylized image after style transfer, as no similarity is detected between the style and stylized images.

Style image selection for style-grid in SMC is a significant challenge. We have studied that few style images are outwardly comparative and the SSIM index is likewise high for them. Considering this case, it is also hard to determine the style images from the stylized images. Fig. 26 compares the

SSIM index of similar types of styles and their stylized representations. Some SSIM values are near to 0.7, which is significantly high, and depict that style images are almost similar. It may result in a wrong answer to an SMC query.

6 Limitations and Future Work

In this work, SMC presents a more reliable, user-friendly, and secure image-CAPTCHA algorithm which addresses many challenges and implies an improvement over existing methods. However, currently, it bears a few limitations in various aspects, summarized below.

Architectural Design: SMC is adopted from elementary NST. In some recent works, several other issues have been handled to improve the robustness, computation time, model architecture, restoration, and other aspects of NST. Among these, an important issue is content leak [43] which is a major concern for security. However, as our target is to develop a novel image-CAPTCHA, thus, we have not focused on these important and more sophisticated design goals in the proposed SMC.

Database: In a few cases, the output stylized images might be rendered with the input styles with a degree of visual similarities. It might confuse the users when selecting the most appropriate styles, causing a Style conflict. As a result, it might be difficult to select the correct styles when solving SMC. A similar style and related stylized images are illustrated in Fig. 27. It is clear that if we choose style images that contain similar colours or similar types of textures/patterns, it will be difficult for the users to select the correct style images by observing the content and stylized images. As a result, it may decrease the solving accuracy, or take more solving time. The chances of such conflicting cases will be mitigated with a larger style dataset size.

Currently, we have tested our scheme with a smaller dataset, containing only 1950 images, which is not sufficient for a rigorous deep-learning attack simulation and usability study (Table 3). Intuitively, the inclusion of more styles and stylized images in the dataset could reduce the success rate of attacks.

Also, Storing images in the database may cause serious security risks due to the potential of attackers gaining access to the database which is an obvious challenge to most of the image-CAPTCHAs.

Usability At present, our SMC is useful for persons with reasonable vision capability. SMC is not suitable for differently abled users with low vision. Moreover, the usability tests could be conducted with diverse variations with a large number of users for future study.

Security: SMC can reduce attack rates considerably, as described earlier. It thwarts several diverse types of traditional as well as deep learning attacks. Currently, these aforesaid limitations of SMC will be tackled by developing a more secure and robust CAPTCHA in the near future. For example, to guard against phishing and spoofing attacks, adding an extra security layer may be beneficial. A detailed usability study will also be performed with a large number of participants and their demographic information. Furthermore, detailed attack analysis will be performed with a large database of style and stylized images. In addition, extraction of the original input style from the stylized images would be another direction of our study.

7 Conclusion

This paper proposes a Style Matching CAPTCHA, namely SMC, by adapting neural style transfer underlying deep neural networks. SMC generates a random challenge and inquires the users to match with the most appropriate style or pattern used in the stylization process along with a content image. Unlike other NST-based existing schemes, which ask the user to select a salient object (content) or area of interest, we have proposed SMC to find the style image used in the stylization task to solve a challenge by observing the semantic

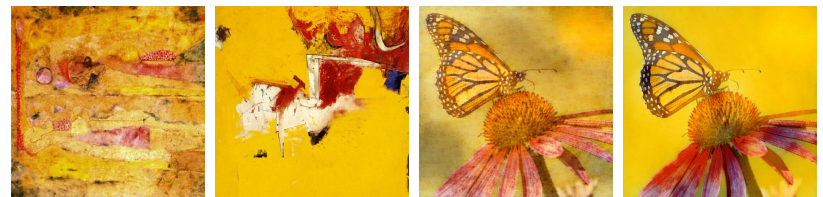


Fig. 27: The first two are similar style images. The next two are corresponding stylized images. Observing these two stylized images, it is difficult for the user to select the correct style.

correlation between the content and styled output image. Our in-depth analysis demonstrates that SMC offers a well-balance between usability and security to design an image-CAPTCHA. SMC randomly generates a challenge that is easily recognizable by humans, maintaining the difficulty for automated intelligent programs. Comprehensive security analysis implies that SMC can effectively thwart bot attacks with intelligent tools and deep learning models. Moreover, traditional image denoising-based attacks which are generally effective for text-CAPTCHAs are explored to analyze the strengths of proposed SMC. To improve the security and robustness, we will explore the suitability of composite styles (*i.e.*, blending of two different random styles) in our SMC algorithm. Also, two-stage NST can be an alternative solution to baffle deep learning attacks from various latent layers. Overall, it is a positive approach to enhance the security of image-CAPTCHA design in a new direction.

Acknowledgement

The authors would like to thank the Editors and anonymous Reviewers for their valuable comments. The authors are also thankful to the volunteers, experts, and participants who have provided their responses and suggestions for this research.

References

1. Acien, A., Morales, A., Fierrez, J., Vera-Rodriguez, R., Delgado-Mohatar, O.: Becaptcha: Behavioral bot detection using touchscreen and mobile sensors benchmarked on humidd. *Engineering Applications of Artificial Intelligence* **98**, 104058 (2021). DOI <https://doi.org/10.1016/j.engappai.2020.104058>
2. Alnfai, M., Alassery, F.: Tapcaptcha: non-visual captcha on touchscreens for visually impaired people. *Journal on Multimodal User Interfaces* pp. 1–14 (2022). DOI <https://doi.org/10.1007/s12193-022-00394-2>
3. Baird, H.S., Bentley, J.L.: Implicit captchas. In: *Document Recognition and Retrieval XII*, vol. 5676, pp. 191–196. International Society for Optics and Photonics (2005). DOI <https://doi.org/10.1117/12.590944>
4. Bera, A., Bhattacharjee, D.: Human identification using selected features from finger geometric profiles. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **50**(3), 747–761 (2017)
5. Bera, A., Bhattacharjee, D., Nasipuri, M.: Person recognition using alternative hand geometry. *International Journal of Biometrics* **6**(3), 231–247 (2014). DOI <https://doi.org/10.1504/IJBM.2014.064403>
6. Bera, A., Bhattacharjee, D., Nasipuri, M.: Hand biometric verification with hand image-based captcha. In: *Advanced Computing and Systems for Security*, pp. 3–18. Springer (2018). DOI https://doi.org/10.1007/978-981-10-8180-4_1
7. Bera, A., Bhattacharjee, D., Shum, H.P.: Two-stage human verification using handcaptcha and anti-spoofed finger biometrics with feature se-

- lection. *Expert Systems with Applications* **171**, 114583 (2021). DOI <https://doi.org/10.1016/j.eswa.2021.114583>
8. Bera, A., Dey, R., Bhattacharjee, D., Nasipuri, M., Shum, H.P.: Spoofing detection on hand images using quality assessment. *Multimedia Tools and Applications* **80**(19), 28603–28626 (2021)
 9. Breiting, F., Tully-Doyle, R., Hassenfeldt, C.: A survey on smartphone user's security choices, awareness and education. *Computers & Security* **88**, 101647 (2020). DOI <https://doi.org/10.1016/j.cose.2019.101647>
 10. Chen, H., Jiang, B., Chen, H.: Stylecaptcha: Captcha based on stylized images to defend against deep networks. In: *Proceedings of the 2020 ACM-IMS on Foundations of Data Science Conference*, pp. 161–170 (2020). DOI <https://doi.org/10.1145/3412815.3416895>
 11. Chen, H., Zhao, L., Wang, Z., Zhang, H., Zuo, Z., Li, A., Xing, W., Lu, D.: Dualast: Dual style-learning networks for artistic style transfer. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 872–881 (2021)
 12. Chen, H.Y., Fang, I., Cheng, C.M., Chiu, W.C., et al.: Self-contained stylization via steganography for reverse and serial style transfer. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2163–2171 (2020)
 13. Cheng, Z., Gao, H., Liu, Z., Wu, H., Zi, Y., Pei, G.: Image-based captchas based on neural style transfer. *IET Information Security* **13**(6), 519–529 (2019). DOI <https://doi.org/10.1049/iet-ifs.2018.5036>
 14. Conti, M., Guarisco, C., Spolaor, R.: Captchastar! a novel captcha based on interactive shape discovery. In: *International Conference on Applied Cryptography and Network Security*, pp. 611–628. Springer (2016). DOI https://doi.org/10.1007/978-3-319-39555-5_33
 15. Conti, M., Pajola, L., Tricomi, P.P.: Captcha attack: Turning captchas against humanity. *arXiv preprint arXiv:2201.04014* (2022). DOI <https://doi.org/10.48550/arXiv.2201.04014>
 16. Dang, V.N., Galati, F., Cortese, R., Di Giacomo, G., Marconetto, V., Mathur, P., Lekadir, K., Lorenzi, M., Prados, F., Zuluaga, M.A.: Vessel-captcha: an efficient learning framework for vessel annotation and segmentation. *Medical Image Analysis* **75**, 102263 (2022). DOI <https://doi.org/10.1016/j.media.2021.102263>
 17. Datta, R., Li, J., Wang, J.Z.: Imagination: a robust image-based captcha generation system. In: *Proceedings of the 13th annual ACM international conference on Multimedia*, pp. 331–334. ACM (2005). DOI <https://doi.org/10.1145/1101149.1101218>
 18. Deng, X.: Enhancing image quality via style transfer for single image super-resolution. *IEEE Signal Processing Letters* **25**(4), 571–575 (2018). DOI [10.1109/LSP.2018.2805809](https://doi.org/10.1109/LSP.2018.2805809)
 19. Elson, J., Douceur, J.R., Howell, J., Saul, J.: Asirra: a captcha that exploits interest-aligned manual image categorization. *CCS* **7**, 366–374 (2007). DOI <https://doi.org/10.1145/1315245.1315291>
 20. Feng, Y., Cao, Q., Qi, H., Ruoti, S.: Sencaptcha: A mobile-first captcha using orientation sensors. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* **4**(2), 1–26 (2020). DOI <https://doi.org/10.1145/3397312>
 21. Gao, H., Cao, F., Zhang, P.: Annulus: A novel image-based captcha scheme. In: *2016 IEEE Region 10 Conference (TENCON)*, pp. 464–467. IEEE (2016). DOI [10.1109/TENCON.2016.7848042](https://doi.org/10.1109/TENCON.2016.7848042)
 22. Gatys, L.A., Ecker, A.S., Bethge, M.: Image style transfer using convolutional neural networks. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2414–2423 (2016)
 23. Godi, M., Joppi, C., Giachetti, A., Pellacini, F., Cristani, M.: Texel-att: Representing and classifying element-based textures by attributes. *arXiv preprint arXiv:1908.11127* (2019). DOI <https://doi.org/10.48550/arXiv.1908.11127>
 24. Golle, P.: Machine learning attacks against the asirra captcha. In: *Proceedings of the 15th ACM conference on Computer and communications security*, pp. 535–542 (2008). DOI <https://doi.org/10.1145/1455770.1455838>
 25. Goswami, G., Powell, B.M., Vatsa, M., Singh, R., Noore, A.: Faced-captcha: Face detection based color image captcha. *Future Generation Computer Systems* **31**, 59–68 (2014). DOI <https://doi.org/10.1016/j.future.2012.08.013>
 26. Goswami, G., Powell, B.M., Vatsa, M., Singh, R., Noore, A.: Fr-captcha: Captcha based on recognizing human faces. *PloS one* **9**(4), e91708 (2014). DOI <https://doi.org/10.1371/journal.pone.0091708>
 27. Gougeon, T., Lacharme, P.: A simple attack on captchastar. In: *International Conference on Information Systems Security and Privacy*, pp. 66–85. Springer (2018). DOI https://doi.org/10.1007/978-3-030-25109-3_4
 28. Gupta, A., Johnson, J., Alahi, A., Fei-Fei, L.: Characterizing and improving stability in neural style transfer. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4067–4076 (2017). DOI <https://doi.org/10.48550/arXiv.1705.02092>
 29. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778 (2016)
 30. Hosam, O., Abousamra, R.: Enhancing deep training of image landmarking with image captcha. In: *2022 8th International Conference on Information Technology Trends (ITT)*, pp. 88–93. IEEE (2022). DOI [10.1109/ITT56123.2022.9863967](https://doi.org/10.1109/ITT56123.2022.9863967)
 31. Jia, X., Xiao, J., Wu, C.: Tics: text-image-based semantic captcha synthesis via multi-condition adversarial learning. *The Visual Computer* pp. 1–13 (2022). DOI <https://doi.org/10.1007/s00371-021-02061-1>
 32. Jin, X., Han, R., Duan, Y., Ning, N., Li, X.: Ar captcha: Recognizing robot by augmented reality. *Concurrency and Computation: Practice and Experience* **33**(15), e5585 (2021). DOI <https://doi.org/10.1002/cpe.5585>
 33. Jing, Y., Yang, Y., Feng, Z., Ye, J., Yu, Y., Song, M.: Neural style transfer: A review. *IEEE transactions on visualization and computer graphics* **26**(11), 3365–3385 (2019). DOI [10.1109/TVCG.2019.2921336](https://doi.org/10.1109/TVCG.2019.2921336)
 34. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: *European conference on computer vision*, pp. 694–711. Springer (2016). DOI https://doi.org/10.1007/978-3-319-46475-6_43
 35. Kong, X., Deng, Y., Tang, F., Dong, W., Ma, C., Chen, Y., He, Z., Xu, C.: Exploring the temporal consistency of arbitrary style transfer: A channel-wise perspective. *IEEE Transactions on Neural Networks and Learning Systems* (2023). DOI [10.1109/TNNLS.2022.3230084](https://doi.org/10.1109/TNNLS.2022.3230084)
 36. Kumar, M., Jindal, M., Kumar, M.: Distortion, rotation and scale invariant recognition of hollow hindi characters. *Sādhanā* **47**(2), 92 (2022). DOI <https://doi.org/10.1007/s12046-022-01847-w>
 37. Kumar, M., Jindal, M., Kumar, M.: A systematic survey on captcha recognition: types, creation and breaking techniques. *Archives of Computational Methods in Engineering* **29**(2), 1107–1136 (2022). DOI <https://doi.org/10.1007/s11831-021-09608-4>
 38. Kumar, M., Jindal, M.K., Kumar, M.: A novel attack on monochrome and greyscale devanagari captchas. *Transactions on Asian and Low-Resource Language Information Processing* **20**(4), 1–30 (2021). DOI <https://doi.org/10.1145/3439798>
 39. Kumar, M., Jindal, M.K., Kumar, M.: Design of innovative captcha for hindi language. *Neural Computing and Applications* pp. 1–36 (2022). DOI <https://doi.org/10.1007/s00521-021-06686-0>
 40. Lewis, J.: Fast normalized cross-correlation, 1995. In: *Vision Interface*, vol. 2010, pp. 120–123 (2010)
 41. Li, C., Chen, X., Wang, H., Wang, P., Zhang, Y., Wang, W.: End-to-end attack on text-based captchas based on cycle-consistent generative

- adversarial network. *Neurocomputing* **433**, 223–236 (2021). DOI <https://doi.org/10.1016/j.neucom.2020.11.057>
42. Liu, X.C., Yang, Y.L., Hall, P.: Learning to warp for style transfer. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3702–3711 (2021)
 43. Ma, Y., Zhao, C., Li, X., Basu, A.: Rast: Restorable arbitrary style transfer via multi-restoration. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 331–340 (2023)
 44. Mallika, Ubhi, J.S., Aggarwal, A.K.: Neural style transfer for image within images and conditional gans for destylization. *Journal of Visual Communication and Image Representation* **85**, 103483 (2022). DOI <https://doi.org/10.1016/j.jvcir.2022.103483>
 45. Mori, G., Malik, J.: Recognizing objects in adversarial clutter: Breaking a visual captcha. In: *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, vol. 1, pp. I–I. IEEE (2003). DOI [10.1109/CVPR.2003.1211347](https://doi.org/10.1109/CVPR.2003.1211347)
 46. Okada, M., Matsuyama, S.: New captcha for smartphones and tablet pc. In: *2012 IEEE Consumer Communications and Networking Conference (CCNC)*, pp. 34–35. IEEE (2012). DOI [10.1109/CCNC.2012.6181038](https://doi.org/10.1109/CCNC.2012.6181038)
 47. Osadchy, M., Hernandez-Castro, J., Gibson, S., Dunkelman, O., Pérez-Cabo, D.: No bot expects the deepcaptcha! introducing immutable adversarial examples with applications to captcha. *Cryptology ePrint Archive* (2016)
 48. Polakis, I., Iliia, P., Maggi, F., Lancini, M., Kontaxis, G., Zanero, S., Ioannidis, S., Keromytis, A.D.: Faces in the distorting mirror: Revisiting photo-based social authentication. In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pp. 501–512 (2014). DOI <https://doi.org/10.1145/2660267.2660317>
 49. Rathor, V.S., Garg, B., Patil, M., Sharma, G.: Security analysis of image captcha using a mask r-cnn-based attack model. *International Journal of Ad Hoc and Ubiquitous Computing* **36**(4), 238–247 (2021)
 50. Ray, P., Giri, D., Kumar, S., Sahoo, P.: Fp-captcha: An improved captcha design scheme based on face points. In: *International Conference on Information Technology and Applied Mathematics*, pp. 218–233. Springer (2019). DOI https://doi.org/10.1007/978-3-030-34152-7_17
 51. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems* **28** (2015)
 52. Ruder, M., Dosovitskiy, A., Brox, T.: Artistic style transfer for videos. In: *German conference on pattern recognition*, pp. 26–36. Springer (2016). DOI https://doi.org/10.1007/978-3-319-45886-1_3
 53. Rui, Y., Liu, Z.: Artificial: Automated reverse turing test using facial features. *Multimedia Systems* **9**(6), 493–502 (2004). DOI <https://doi.org/10.1145/957013.957075>
 54. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. *International journal of computer vision* **115**(3), 211–252 (2015). DOI <https://doi.org/10.1007/s11263-015-0816-y>
 55. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. In: *Proceedings of the IEEE international conference on computer vision*, pp. 618–626 (2017)
 56. Shet, V.: Are you a robot? introducing no captcha recaptcha. *Google Security Blog* **3**, 12 (2014)
 57. Shi, C., Xu, X., Ji, S., Bu, K., Chen, J., Beyah, R., Wang, T.: Adversarial captchas. *IEEE Transactions on Cybernetics* (2021). DOI [10.1109/TCYB.2021.3071395](https://doi.org/10.1109/TCYB.2021.3071395)
 58. Shirali-Shahreza, M., Shirali-Shahreza, S.: Captcha for blind people. In: *2007 IEEE International Symposium on Signal Processing and Information Technology*, pp. 995–998. IEEE (2007). DOI [10.1109/ISSPIT.2007.4458048](https://doi.org/10.1109/ISSPIT.2007.4458048)
 59. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014). DOI <https://doi.org/10.48550/arXiv.1409.1556>
 60. Singh, A., Jaiswal, V., Joshi, G., Sanjeeve, A., Gite, S., Kotecha, K.: Neural style transfer: A critical review. *IEEE Access* (2021). DOI [10.1109/ACCESS.2021.3112996](https://doi.org/10.1109/ACCESS.2021.3112996)
 61. Sivakorn, S., Polakis, I., Keromytis, A.D.: I am robot:(deep) learning to break semantic image captchas. In: *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 388–403. IEEE (2016). DOI [10.1109/EuroSP.2016.37](https://doi.org/10.1109/EuroSP.2016.37)
 62. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826 (2016)
 63. Tang, M., Gao, H., Zhang, Y., Liu, Y., Zhang, P., Wang, P.: Research on deep learning techniques in breaking text-based captchas and designing image-based captcha. *IEEE Transactions on Information Forensics and Security* **13**(10), 2522–2537 (2018). DOI [10.1109/TIFS.2018.2821096](https://doi.org/10.1109/TIFS.2018.2821096)
 64. Thompson, N., McGill, T.J., Wang, X.: “security begins at home”: Determinants of home computer and mobile device security behavior. *computers & security* **70**, 376–391 (2017). DOI <https://doi.org/10.1016/j.cose.2017.07.003>
 65. Uzun, E., Chung, S.P.H., Essa, I., Lee, W.: rtcaptcha: A real-time captcha based liveness detection system. In: *NDSS* (2018)
 66. Von Ahn, L., Blum, M., Hopper, N.J., Langford, J.: Captcha: Using hard ai problems for security. In: *International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 294–311. Springer (2003). DOI <https://doi.org/10.1007/3-540-39200-9>
 67. Von Ahn, L., Maurer, B., McMillen, C., Abraham, D., Blum, M.: recaptcha: Human-based character recognition via web security measures. *Science* **321**(5895), 1465–1468 (2008). DOI [10.1126/science.1160379](https://doi.org/10.1126/science.1160379)
 68. Wang, P., Gao, H., Guo, X., Yuan, Z., Nian, J.: Improving the security of audio captchas with adversarial examples. *IEEE Transactions on Dependable and Secure Computing* (2023). DOI [10.1109/TDSC.2023.3236367](https://doi.org/10.1109/TDSC.2023.3236367)
 69. Wang, P., Gao, H., Shi, Z., Yuan, Z., Hu, J.: Simple and easy: transfer learning-based attacks to text captcha. *IEEE Access* **8**, 59044–59058 (2020). DOI [10.1109/ACCESS.2020.2982945](https://doi.org/10.1109/ACCESS.2020.2982945)
 70. Wang, P., Gao, H., Xiao, C., Guo, X., Gao, Y., Zi, Y.: Extended research on the security of visual reasoning captcha. *IEEE Transactions on Dependable and Secure Computing* (2023). DOI [10.1109/TDSC.2023.3238408](https://doi.org/10.1109/TDSC.2023.3238408)
 71. Wang, P., Li, Y., Vasconcelos, N.: Rethinking and improving the robustness of image style transfer. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 124–133 (2021). DOI <https://doi.org/10.48550/arXiv.2104.05623>
 72. Wang, Y., Wei, Y., Zhang, M., Liu, Y., Wang, B.: Make complex captchas simple: A fast text captcha solver based on a small number of samples. *Information Sciences* **578**, 181–194 (2021). DOI <https://doi.org/10.1016/j.ins.2021.07.040>
 73. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing* **13**(4), 600–612 (2004). DOI [10.1109/TIP.2003.819861](https://doi.org/10.1109/TIP.2003.819861)
 74. Weichbroth, P., Łysik, Ł.: Mobile security: Threats and best practices. *Mobile Information Systems* **2020**, 1–15 (2020). DOI <https://doi.org/10.1155/2020/8828078>

75. Xu, X., Liu, L., Li, B.: A survey of captcha technologies to distinguish between human and computer. *Neurocomputing* **408**, 292–307 (2020). DOI <https://doi.org/10.1016/j.neucom.2019.08.109>
76. Zhang, J., Tsai, M.Y., Kitchat, K., Sun, M.T., Sakai, K., Ku, W.S., Surasak, T., Thaipisutikul, T.: A secure annuli captcha system. *Computers & Security* **125**, 103025 (2023). DOI <https://doi.org/10.1016/j.cose.2022.103025>
77. Zhang, K., Zheng, Y.: Information Security: 7th International Conference, ISC 2004, Palo Alto, CA, USA, September 27–29, 2004, Proceedings, vol. 3225. Springer (2004)
78. Zhang, K., Zuo, W., Chen, Y., Meng, D., Zhang, L.: Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE transactions on image processing* **26**(7), 3142–3155 (2017). DOI 10.1109/TIP.2017.2662206
79. Zhang, P., Gao, H., Cheng, Z., Cao, F.: Two novel image-based captcha schemes based on visual effects. In: CCF Chinese Conference on Computer Vision, pp. 14–25. Springer (2017). DOI https://doi.org/10.1007/978-981-10-7305-2_2
80. Zhu, B., Liu, J., Li, Q., Li, S., Xu, N.: Image-based captcha exploiting context in object recognition (2013). US Patent 8,483,518
81. Zhu, B.B., Yan, J., Li, Q., Yang, C., Liu, J., Xu, N., Yi, M., Cai, K.: Attacks and design of image recognition captchas. In: Proceedings of the 17th ACM conference on Computer and communications security, pp. 187–200. ACM (2010). DOI <https://doi.org/10.1145/1866307.1866329>
82. Zi, Y., Gao, H., Cheng, Z., Liu, Y.: An end-to-end attack on text captchas. *IEEE Transactions on Information Forensics and Security* **15**, 753–766 (2019). DOI 10.1109/TIFS.2019.2928622