CBOD: A Resource-Efficient and Robust Image-Based CAPTCHA Scheme

[1,2]Md. Shamim Imtiaz*, [1]Kamrul Hasan, [1]Amina Khatun, [1,3]Mohammad Shorif Uddin
[1]Department of Computer Science and Engineering, Jahangirnagar University, Dhaka 1342, Bangladesh
[2]Department of Computer Science and Engineering, Pirojpur Science and Technology University, Pirojpur 8500, Bangladesh
[3]Green University of Bangladesh, Narayanganj 1461, Bangladesh

*Corresponding Author: **Md. Shamim Imtiaz**

Abstract: Traditional text-based and conventional image-based CAPTCHA schemes are increasingly vulnerable to automated attacks leveraging modern deep learning and computer vision architectures. To address this security risk without compromising user experience, this paper introduces Color Boundary Object Detection (CBOD), a robust and user-friendly image-based CAPTCHA scheme designed to resist automated object detection pipelines. The CBOD framework constructs challenges through a multi-step pipeline: strategic spatial preprocessing, composite quad-image merging, the algorithmic insertion of randomly shaped and colored polygonal boundaries around target objects, and the application of a variable layer of mixed random noise. Security evaluations demonstrate that the proposed scheme achieves exceptional combinatorial robustness, generating a conservatively estimated 1.17×10^{12} unique challenges from a minimal baseline asset library of 150 images distributed across 10 semantic categories. This structural resilience stems from defensive mechanisms such as image discontinuity, foreground occlusion, and localized noise artifacts, which intentionally exploit the structural vulnerabilities of deep learning-based object de-tection frameworks. Concurrently, an extensive usability study involving 285 participants establishes high human performance, characterized by a 96% first-attempt success rate and a low average completion time of 6.8 seconds. While the current implementation utilizes a predefined asset taxonomy and lacks specialized assistive features for visually impaired individuals, CBOD provides a highly secure, lightweight verification mechanism that effectively balances low backend resource overhead with a seamless user experience.

Keywords: CAPTCHA; Image-based CAPTCHA; Text-based CAPTCHA; Color Boundary Object Detection (CBOD)

1. Introduction

Maintaining integrity against automated exploitation is a persistent concern for modern web services. Auto-mated computer scripts can rapidly compromise standard form fields, such as login, registration, and email inputs, leading to traffic congestion, degraded server performance, database pollution, and systemic service failures (Hasan, 2016). Furthermore, distributed denial-of-service (DDoS) attacks pose severe operational risks by disrupting client-server communication channels (Bera et al., 2014; Ghavimi & Chen, 2014). Mitigating these automated threats requires a reliable gatekeeping mechanism capable of accurately differentiating between human operators and au-tomated programs (bots). This function is primarily fulfilled by CAPTCHA mechanisms, a concept originating from the foundational Turing test principles established by Alan Turing in 1950 (Copeland & Proudfoot, 2009). Over the past few decades, numerous CAPTCHA designs have emerged, broadly categorized into text, image, audio, video, and puzzle-based verification systems (Singh & Pal, 2014). Among these, text- and image-based methods have achieved the highest rate of commercial adoption. However, rapid progress in computer vision and artificial intelligence has significantly eroded the security baselines of these traditional approaches (Osadchy et al., 2017). Text-based CAPTCHAs, which historically served as the standard security paradigm, have become largely obsolete due to the efficacy of advanced segmentation and optical character recognition techniques (Bursztein et al., 2014). Image-based verification schemes initially offered a more resilient alternative, yet modern deep learning frame-works have introduced critical vulnerabilities. Advanced neural networks excel at automated object classification and localization, narrowing the operational performance gap between humans and machines on conventional visual challenges (Bursztein et al., 2010; Goodfellow et al., 2013).

The core motivation of this research is to resolve the trade-off between security and usability in visual verifica-tion. Existing image-based schemes often rely on massive image databases to maintain unpredictability incurring high storage costs or introduce excessive visual distortions that frustrate human users. This paper addresses these limitations by introducing a novel image-based scheme named Color Boundary Object Detection (CBOD) CAPTCHA. Instead of simple object identification, the CBOD mechanism requires the user to recognize a specific target object and correctly identify the colors of the randomly generated polygonal boundaries

surrounding it.

The primary contributions and technical highlights of this study are detailed below:

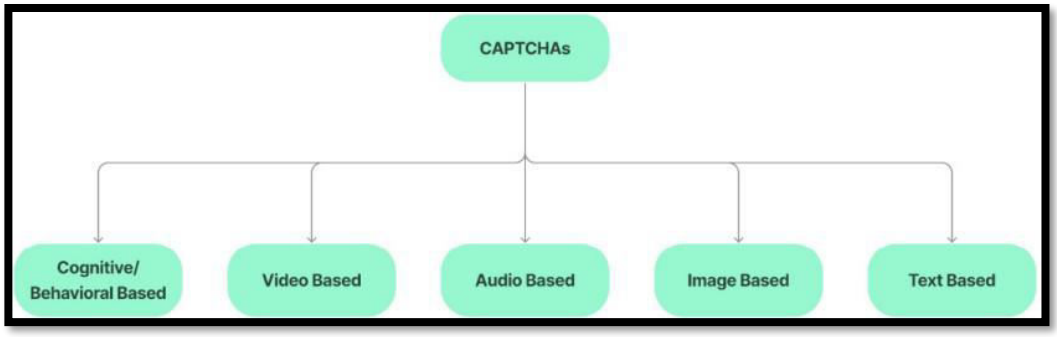
- The system introduces a resource-efficient generation method that produces approximately **1.17×10^{1}Z** unique CAPTCHA challenges from a minimal database footprint of only 150 source images across 10 distinct categories, validated through a formal mathematical proof of combinatorial complexity to eliminate heavy storage dependencies.
- The model establishes a layered defense framework utilizing localized cropping, downsampling, canvas merg-ing, boundary randomization, and mixed noise injection that forces context decoupling, artificial occlusion, and edge discontinuity to degrade the feature extraction capabilities of automated object detection solvers.
- The architecture provides an optimized, human-centric design validated via a large-scale usability study with 285 participants demonstrating a 96% first-attempt human success rate and a low average completion time of 6.8 seconds, proving it is intuitive and friction-free compared to conventional security methods.

The remainder of this paper is organized as follows: Section 2 reviews relevant literature and isolates current research gaps. Section 3 formalizes the technical methodology and algorithmic pipeline of the CBOD scheme. Section 4 presents experimental evaluations, security assessments, and user experience analysis. Finally, Section 5 concludes the paper and outlines prospective research directions.

2. Literature Review

In 1950, Alan Turing proposed a method to distinguishes human and machine behavior by testing a machine's ability to manifest intelligent behavior like the human or not. He named it as "Turing Test" (Copeland & Proudfoot, 2009). In the early year of 1997, Andrei Broder described a method that distinguishes between human and machine by showing a distorted word to the users, and the users were asked to type the word for confirming human (Baird & Luk, 2003). In 2003, the Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA) was first proposed by Von Ahn et al. (2003). The main purpose of CAPTCHA generation was to prevent machines from automatically registering on the websites, much more effective for finding spam mail, slowing Danial of Service (DoS) attacks and finding brute force attacks on passwords (Aldwairi et al., 2017).

In the beginning, CAPTCHAs were created using text and is displayed by using distorted texts (Shirali-Shahreza & Shirali-Shahreza, 2011). A classification model for the CAPTCHA was made by Yan & El Ahmad (2008), where they classify it with text-based, image-based, and sound-based schemes. Recently in 2020, Dionysiou & Athana-sopoulos (2020) classify the CAPTCHA model into ten important categories: (a) text-based CAPTCHA (Bursztein et al., 2011), (b) Google reCAPTCHA (Gaggi, 2022), (c) Google No CAPTCHA (Sivakorn et al., 2016), (d) math CAPTCHA (Shirali-Shahreza & Shirali-Shahreza, 2007b), (e) honeypot CAPTCHA (Abdullahi et al., 2019), (f) sign-in CAPTCHA (Althamary & El-Alfy, 2017), (g) time-based CAPTCHA, (h) biometrics CAPTCHA, (i) confident CAPTCHA, (j) audio CAPTCHA (Choudhary et al., 2013). Currently, there are numerous CAPTCHA versions that rely on input requests Kumar et al. (2022) (e.g., Puzzle-based CAPTCHA Aljarbou (2019), Video-based CAPTCHA (Zhang et al., 2019), Game-based CAPTCHA(Xu et al., 2020), Flash-based gaming CAPTCHA Aldwairi et al. (2020), Pro CAPTCHA (Nanglae & Bhattarakosol, 2024), vr CAPTCHA (Li et al., 2021), Illusion CAPTCHA (Ding et al., 2025), Adaptive CAPTCHA (Wan et al., 2024)).



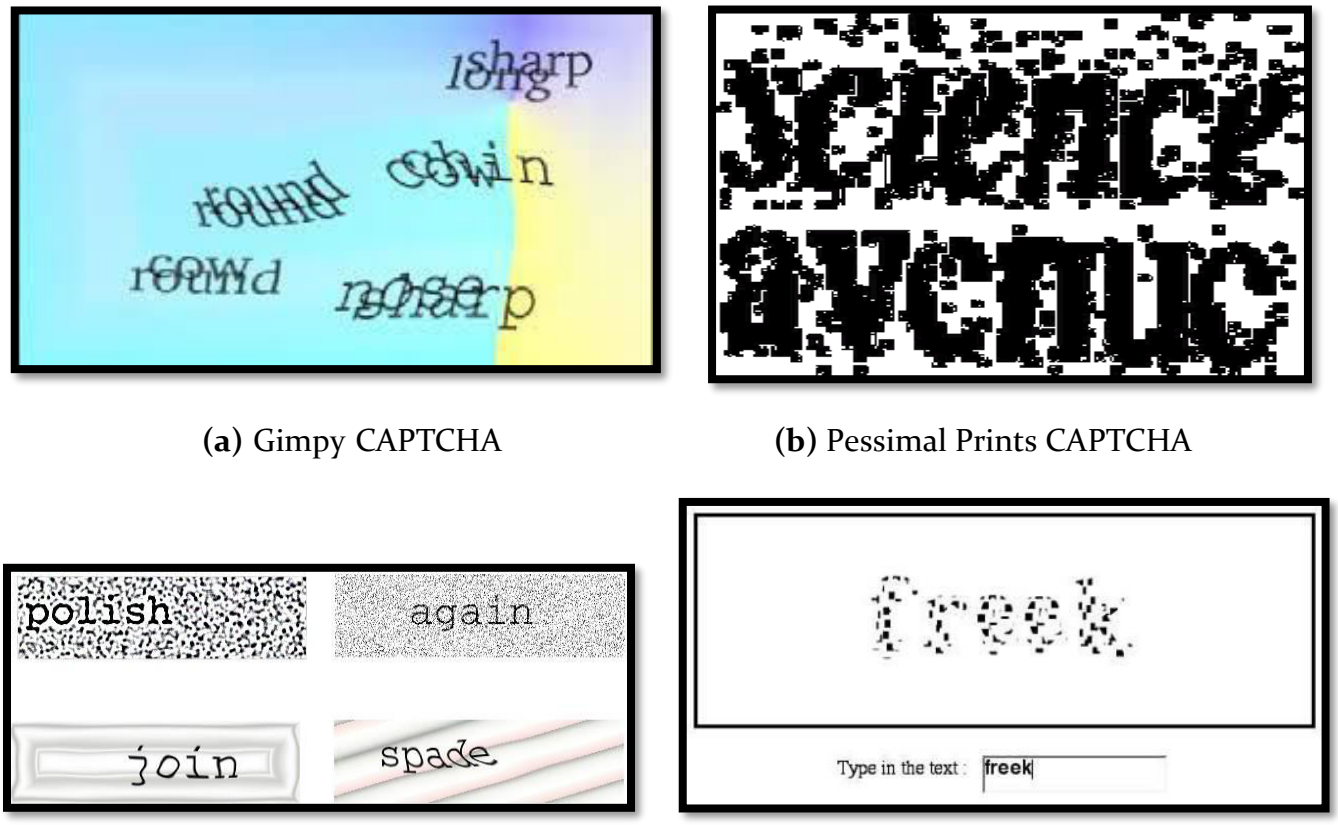
As illustrated in Figure 1, CAPTCHA codes are currently categorized as cognitive/behavioral-based, video-based, audio-based, image-based, text-based, and others.

Figure 1: Main schemes of CAPTCHA (Dinh & Hoang, 2023)

2.1. ***Text-based CAPTCHA***

Text-based captcha is easy to apply where a series of digits and letters are displayed to the users by adding some modifications into the characters e.g. sound, scatter, rotate, create 3D letters. These changes are made in such a way that any robot program is unable to read those characters (Bursztein et al., 2011). Text-based CAPTCHAs have been used by most websites such as Microsoft, Google, Yahoo, Ticketmaster, Pay-Pal, eBay, etc. Figure 2 represents

different types of text-based CAPTCHAs (e.g., Figure 2a shows the Gimpy CAPTCHA (Von Ahn et al., 2004), Figure 2b shows the Pessimal Prints CAPTCHA (Baird et al., 2003), Figure 2c shows the EZ-Gimpy CAPTCHA (Hajjdiab et al., 2017), and Figure 2d shows the Scatter Type CAPTCHA (Baird et al., 2005)). The evolution of CAPTCHA technology is a continuous arms race between security designers and bot developers. While early systems focused on text distortion, the rise of powerful machine learning models has shifted focus towards challenges that leverage higher-order human cognitive abilities, such as object recognition and semantic understanding (Tariq et al., 2023).



(a) Gimpy CAPTCHA **(b)** Pessimal Prints CAPTCHA

(c) EZ-Gimpy CAPTCHA **(d)** Scatter Type CAPTCHA

Figure 2: Example of different types of text-based CAPTC

2.2. ***Image-based CAPTCHA***

The first wave of image-based CAPTCHAs moved beyond character recognition to object identification. Founda-tional schemes like ESP-PIX asked users to identify a common concept among a set of images (Aldwairi et al., 2020). Collage CAPTCHA required users to click on all images belonging to a specific category (Shirali-Shahreza & Shirali-Shahreza, 2007a). Perhaps

the most well-known example, Microsoft's Asirra, challenged users to distinguish cats from dogs Elson et al. (2007). While innovative for their time, these systems shared a common vulnerability: they relied on clean, well-defined images, making them susceptible to early object recognition algorithms as computer vision technology improved.

2.3. *Advanced 3D Model-based CAPTCHA*

To counter improving recognition algorithms, researchers developed more complex CAPTCHAs. DeepCAPTCHA introduced the use of 3D rendered objects, asking users to sort them by size Nejati et al. (2014). A later enhance-ment introduced immutable adversarial noise that was specifically designed to confuse and mislead neural networks while remaining largely interpretable to the human eye Osadchy et al. (2017).

Other approaches have focused on user interaction, such as click-based CAPTCHAs that require identifying ob-jects in cluttered scenes Chow et al. (2008) or drag-based CAPTCHAs that analyze mouse dynamics (Gossweiler et al., 2009). Commercially, this approach is best exemplified by Google's reCAPTCHA, which has evolved from ex- plicit challenges (v2) to a more passive (Plesner et al., 2024), risk-analysis engine (v3) that scores user interactions in the background (Akrout et al., 2019). Recent analysis suggests that the robustness of systems like reCAPTCHAv2 may rely heavily on contextual signals like cookies and browser history, making it a form of advanced, context-aware verification rather than a pure visual test. Different types of image-based CAPTCHA are shown in Figure 3.

2.4. *Rise of Deep Learning Based Solvers*

The last decade has been defined by the dominance of deep learning, particularly Convolutional Neural Networks (CNNs), in computer vision (Derea et al., 2023). This has had a profound impact on CAPTCHA security. State-of-the-art object detection models (e.g., YOLO, Faster R-CNN) and segmentation models can now easily solve many image-based CAPTCHAs that rely on simple object identification (IJIRCST, 2024). Furthermore, research has shown that even CAPTCHAs with significant clutter and distortion can be broken with enough training data. Generative Adversarial Networks (GANs) can be used to generate training data or to clean noisy CAPTCHA images before feeding them to a solver (Moradi et al., 2024). This reality necessitates that new CAPTCHA designs must actively defend against the specific mechanisms of DL models, rather than just presenting a generic AI problem.

2.5. ***Research Gap***

The current research landscape highlights a significant gap in the design of image-based CAPTCHAs. Many existing image-based CAPTCHAs are either: 1) vulnerable to modern DL-based solvers, 2) heavy dependence on massive image databases for security-through-obscurity, or 3) excessive complexity that degrades user experience and increases computational cost. To address these challenges, there is a clear demand for a CAPTCHA mechanism that offers strong resistance to DL-based attacks, remains resource-efficient, and ensures a smooth user experience. To bridge this gap, the proposed CBOD approach combines several lightweight defensive layers, such as prepro-cessing, occlusion, and noise, into a unified framework that is intuitive for humans but challenging for machines to understand.

3. Methodology

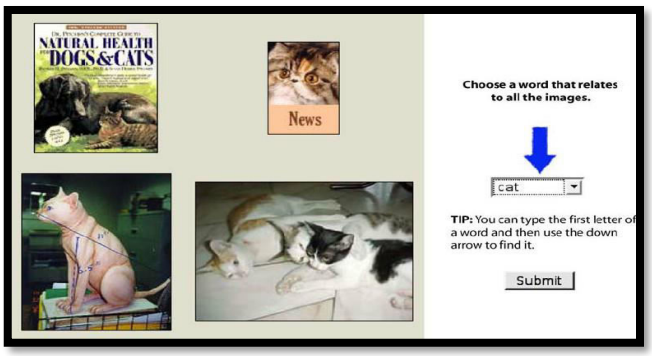
The CBOD generation framework is designed to be a multi-layered defense system. Each stage of the process introduces ambiguity for automated solvers while preserving clarity for human users. This section provides a detailed, refined explanation of the algorithms and principles underpinning the CBOD scheme.

3.1. ***Overview***

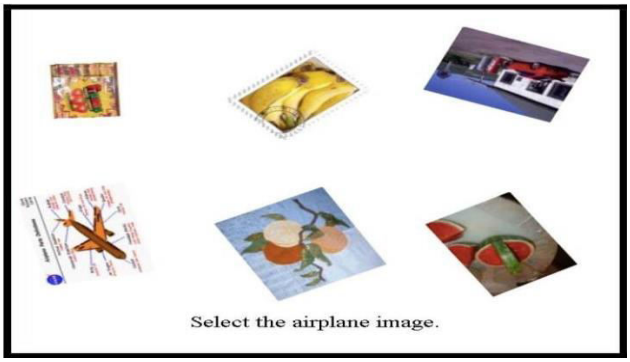
The experiment utilized a dataset of **150** images, carefully selected for clarity and recognizability. The images were organized into **10** distinct categories: table, parrot, cat, dog, flower, car, house, airplane, fruit, and crow. Images were collected from publicly accessible datasets such as ImageNet, COCO, and other websites. Simple objects with transparent backgrounds were also filtered.

We measured the computational cost of generating a single CBOD-CAPTCHA. On a standard server (Intel Core **i7**, **16**GB RAM), the average generation time was approximately **55** milliseconds. This low overhead confirms the scheme's efficiency, as it relies on standard image manipulation libraries rather than computationally expensive deep learning models.

The CBOD generation pipeline is initiated upon an agent's request for a CAPTCHA. The process starts with some

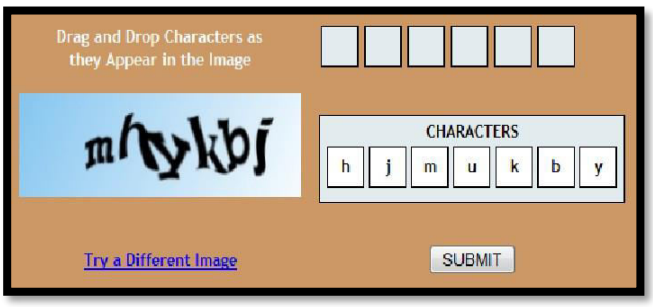


(a) ESP-PIX CAPTCHA

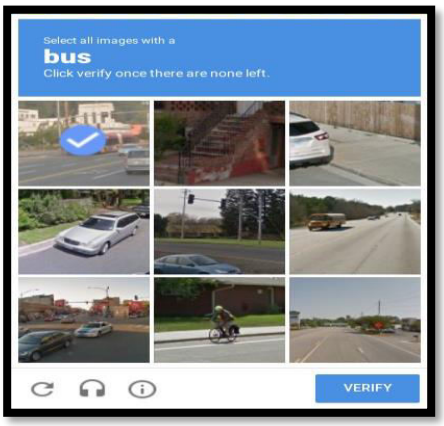


(b) Collage CAPTCHA

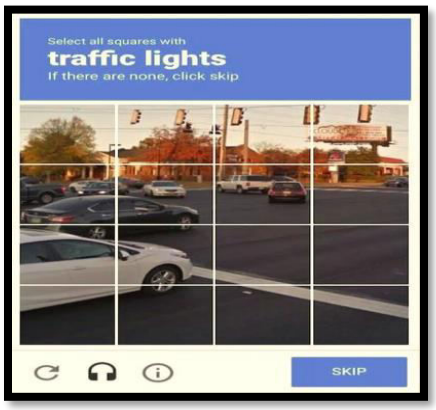
(c) ASIRRA CAPTCHA



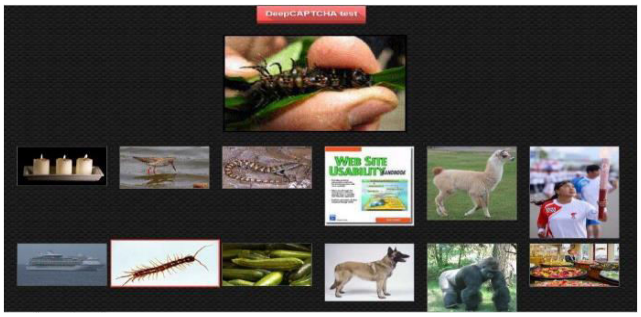
(d) Drag-based CAPTCHA



(e) Click-based CAPTCHA



(f) reCAPTCHA



(g) Deep CAPTCHA

Figure 3: Example of different types of image-based CAPTCHA

preprocessing steps described in Subsection 3.2. Four images are randomly selected from the base image dataset and merged within a single image to generate a CBOD-CAPTCHA, described in Algorithm 2. A random boundary polygon is drawn around each target object in the merged image, described in Algorithm 3. Subsection 3.3 discusses the technique of drawing the random boundaries in detail. After that, to deceive object-detection methods, mixed random noises are applied, described in Algorithm 4. The tolerance of impulsive noise was kept under a certain level; since it could make the image incomprehensible for humans. As for the CAPTCHA test, an agent asked to select the colors, which surround the target objects of a randomly selected category among the categories of four randomly chosen images. If the agent can figure out the correct colors, then the agent is considered human. Algorithm 1 formally describes the complete process of the CBOD-CAPTCHA model, and Figure 4 shows the flowchart of the process.

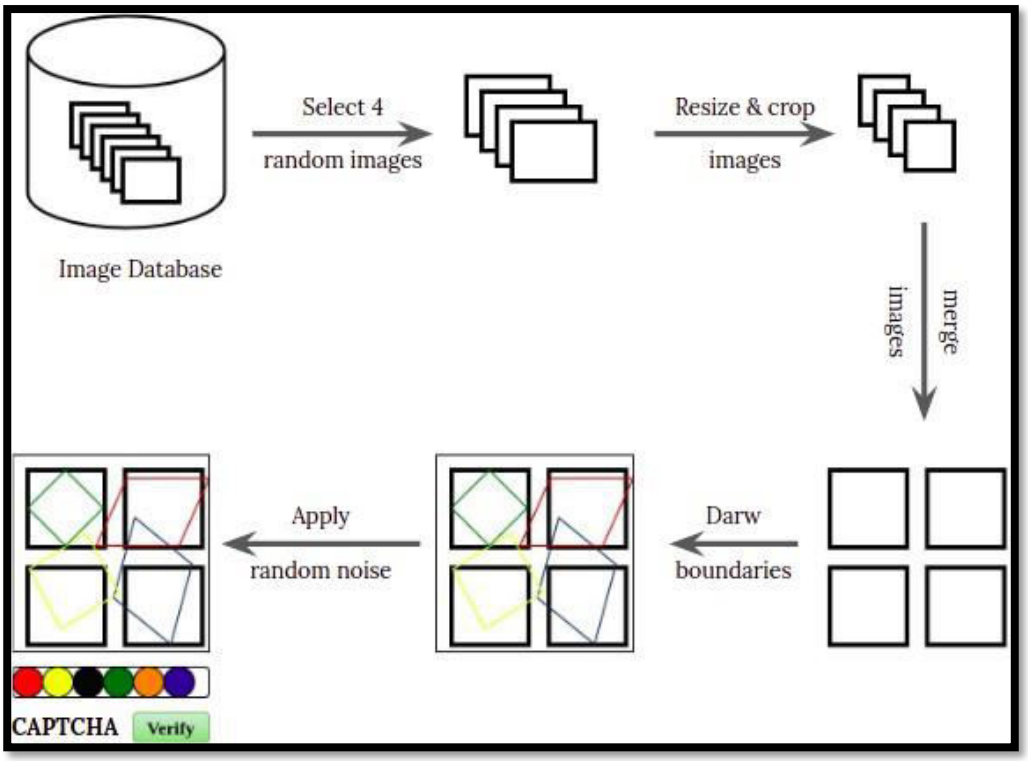


Figure 4: Flowchart of CBOD-CAPTCHA generation model

3.2. *Image Preprocessing*

To defend against the attacks efficiently, especially against the deep learning-based attacks, requires preventive measures. In this CAPTCHA mechanism, images are first cropped and resized into low-resolution images.

3.2.1. *Cropping*

This is not a random crop. It is a targeted process designed to remove contextual information. The reason behind cropping images is to remove some of the details of the target object. As a result, classifying the target object using existing object detection-based algorithms becomes extremely challenging. But the human brain can identify those objects, see Figure 5.

Algorithm 1: CBOD-CAPTCHA

Input: Agent request

Output: Returns whether the agent is human or not

$Lim \leftarrow$ list of 4 randomly selected images from image database

$CBOD \leftarrow$ **Generate CAPTCHA** ()

Ask the agent to select the color that bounds the objects in *CBOD* of the randomly selected category

if *agent selects correct colors* **then** **return** the agent is human **else** **return** the agent is not human

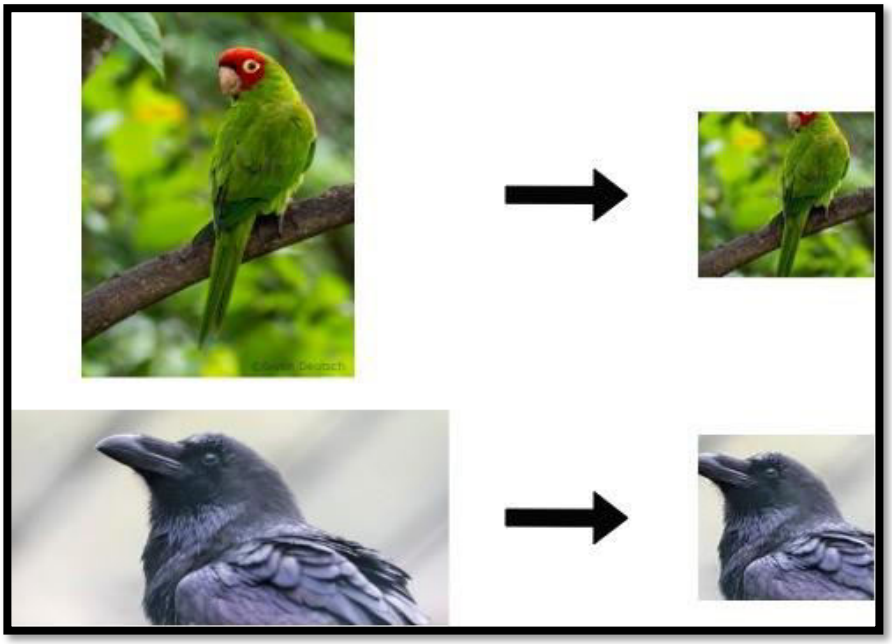


Figure 5: Image preprocessing for generating CAPTCHA

3.2.2. *Resizing*

All cropped images are resized to a low resolution, for instance, **100** × **100** pixels. This low resolution signifi-cantly hinders the performance of deep CNNs, which are typically trained on higher-resolution images and rely on fine-grained texture and edge features that are lost during down sampling. Another reason for reducing resolution is to make the image inconsistent for denoising after applying mixed random noise.

3.3. *Draw Random Color Boundaries*

This feature is the key novelty point of this study. Drawing a random color border around the target object and asking to find its color involves recognizing the target object first and then find the boundary line. The boundary

is designed to be easily visible to humans but difficult for segmentation algorithms to trace precisely. Additionally, some mixed random noises are applied on the CAPTCHA image that makes the detection difficult by the object detection programs.

3.3.1. *Target Object Localization*

For each quarter Q_i or each target object, a random boundary line is drawn by first determining a random center pixel (x_r, y_r) of the target object. This center is calculated based on a pre-annotated centroid of the original image.

3.3.2. *Polygon Vertex Selection*

A set of pixels P surrounding the target object Q_i is generated algorithmically to form the polygon's vertices. To select N vertices (a random integer from **4** to **6**), points are sampled at random angles (0-360°) and radii from the object's centroid. Each vertex's radius ranges from **1.2x** the bounding box radius to **1.6x** the radius distance.

3.3.3. *Color and Thickness*

The boundary color is chosen randomly from a predefined set C = {*red, green, blue, yellow, black, brown, orange*}. The boundary thickness is also randomized within a range of **10** to **15** pixels. This range is thick enough to be unambiguous for human vision but variable enough to complicate analysis by automated tools.

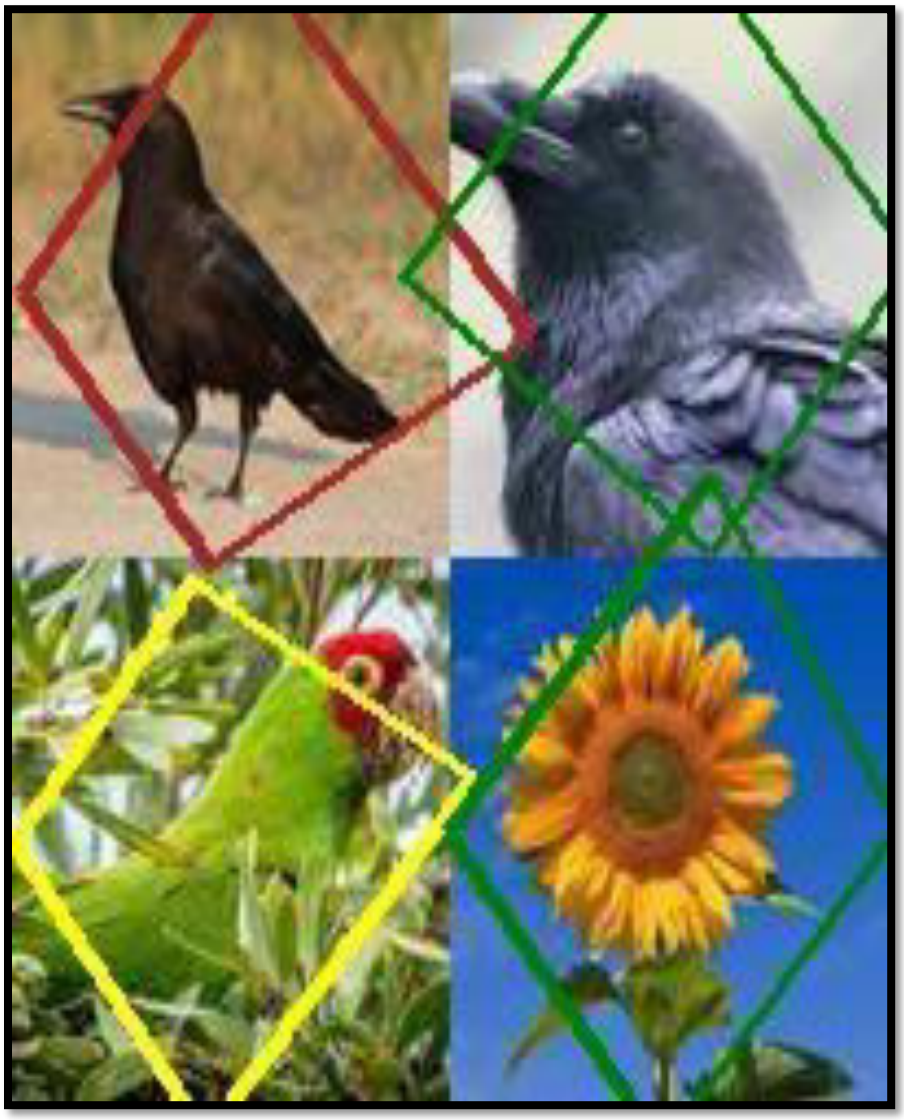


Figure 6: Merged image after drawing random boundaries

A polygon B is drawn on Q_i with border-color *clr* using the pixels of P. Algorithm 3 describes the procedure and Figure 6 shows an example of drawing random color boundaries.

Algorithm 2: Generate CAPTCHA

Input: Lim: list of 4 images

Output: I_C: CAPTCHA Image

$Im \leftarrow$ merge images from Lim

$I_b \leftarrow$

DrawRandomBoundar

y(I_m) $I_c \leftarrow$

ApplyMixedRandomNois

e(I_b) **return** I_C

Algorithm 3: Draw Random Boundary

Input: I_m: four merged images

Output: I_b: random boundary drawn image

$I_b \leftarrow I_m$

for *each quarter image* $Q_i \in I_b$ **do**

- $(x_r, y_r) \leftarrow$ a random pixel near the center pixel of the target object of Q_i
- P $\leftarrow$ set of random pixels around the target object of Q_i
- B $\leftarrow$ a polygon by connecting the pixels in P
- *clr* $\leftarrow$ a random color from C
- Darw B on Q_i with border color *clr* and thickness between 10 to 15 pixels

return Ib

3.4. *Apply Mixed Random Noise*

The final defensive layer is the application of mixed random noise. The unpredictability of the type of noise is key to defend against attacks based on deep learning.

3.4.1. *Noise Types and Parameters*

The algorithm randomly chooses between 'Salt and Pepper' noise (with a specified density, e.g., 0.05, meaning 5% of pixels are affected), and 'Speckle' noise (multiplicative noise with

a specified variance, e.g., 0.04).

3.4.2. ***Rationale***

Using a mix of noise types prevents a bot from applying a single, optimized denoising filter. A median filter effective against Salt and Pepper noise may perform poorly on Speckle noise, and vice-versa. Because the underlying images are already low-resolution, any aggressive denoising attempt is likely to corrupt the object features as much as it removes the noise, further confusing a classifier.

Algorithm 4 outlines the process where n_c specifies the type of impulse noise to be applied. Figure 7 illustrates a sample CBOD-CAPTCHA image after the application of mixed random noise.

Figure 8 shows two examples of CBOD-CAPTCHA. For testing the CAPTCHA of Figure 8a, if an agent selects the colors {*red, green*} that surround the target objects (crow), then the agent will be considered as human.

Figure 7: CAPTCHA images with random noise

Algorithm 4: Apply Mixed Random Noise

Input: I_b: random boundary drawn image

Output: I_n: noisy image as CAPTCHA

$n_c \leftarrow$ a random integer between 1 – 4

if n_c *is* 1 **then**

  $I_n \leftarrow$ apply Salt Noise on I_b

else if n_c *is* 2 **then**

  $I_n \leftarrow$ apply Pepper Noise on I_b

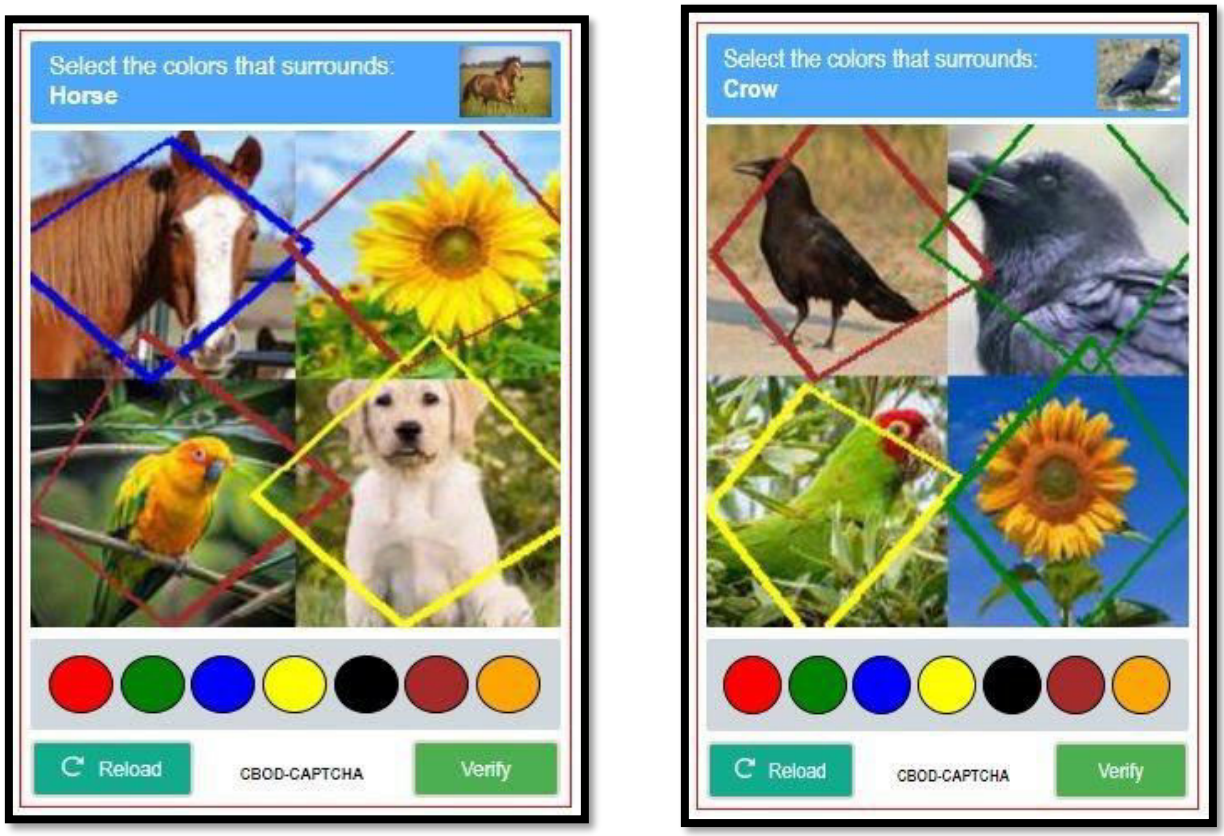
else if n_c *is* 3 **then**

  $I_n \leftarrow$ apply Salt and Pepper Noise on I_b

else

  $I_n \leftarrow$ apply Speckle Noise on I_b

return I_n



(a) **(b)**

Figure 8: Sample of CBOD-CAPTCHA

4. Result Analysis

4.1. ***Result***

To implement the CBOD-CAPTCHA algorithm, we used only **150** images. Remarkably, despite this limited dataset, the algorithm can generate approximately **1.17** × **10**[12] unique CAPTCHAs. Each CAPTCHA includes ran-domly selected images, dynamically generated boundaries with random widths, and varied color assignments. The probability of generating the same CAPTCHA twice here is extremely low. This makes the system highly robust and unpredictable.

4.2. ***Mathematical Justification***

The immense robustness of CBOD stems from a combinatorial explosion of possibilities. The total number of unique CAPTCHAs is a product of several independent random factors. Let's break down each term based on our dataset of **150** images in **10** categories (**15** images per category) and **7** boundary colors:

1. **Ways to choose 4 distinct images from 150:** This is a combination calculation, $C(150, 4)$.

$$C(150,4) = \frac{150!}{4!\,(150-4)!} = \frac{150\ x\ 149\ x\ 148\ x\ 147}{4\ x\ 3\ x\ 2\ x\ 1} = 20{,}260{,}275$$

2. **Ways to arrange these 4 images in the 4 quadrants:** This is a permutation of 4 items, 4!.

$$4! = 4 \times 3 \times 2 \times 1 = 24$$

3. **Ways to assign 7 colors to the 4 boundaries:** For each of the 4 boundaries, there are 7 color choices. This is 7^4.

$$7^4 = 7 \times 7 \times 7 \times 7 = 2{,}401$$

4. **Total Theoretical Combinations:**

$$\text{Total} = C(150, 4) \times 4! \times 7^4 = 20{,}260{,}275 \times 24 \times 2{,}401 \approx 1.17 \times 10^{12}$$

This calculation is a conservative floor. It does not account for the virtually infinite number of unique polygonal boundary shapes that can be generated for each image, nor the different noise patterns. This demonstrates that even with a small asset base, the probability of an attacker encountering the same CAPTCHA twice is negligible, rendering pre-computation or replay attacks infeasible.

4.3. *Security Against Automated Attacks*

We generated a large dataset of over 10,000 unique CBOD CAPTCHAs and run the pre-trained detector on them and report the object detection success rate. The hypothesis was that the combination of cropping, resizing, merging, and noise will severely degrade the detector's performance. The bot, which leveraged a powerful YOLOv5 object detector, achieved a success rate of only 8.7% across 10,000 unique CBOD instances, shown in Figure 9.

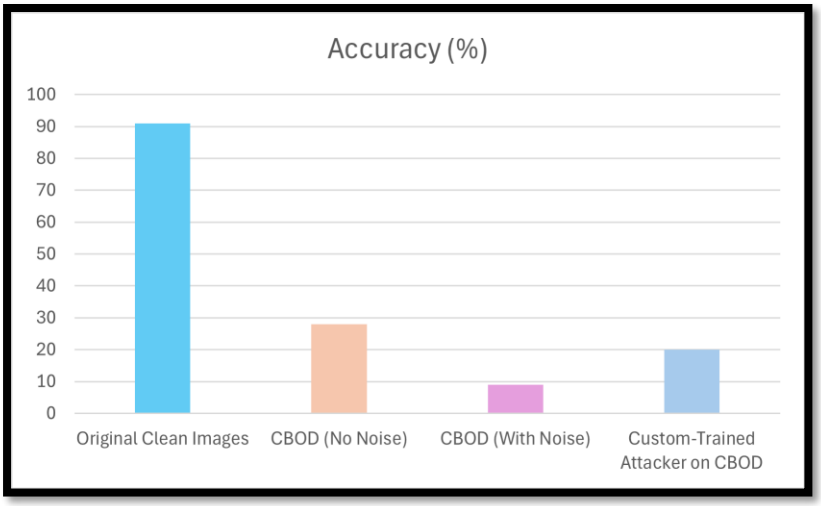


Figure 9: Accuracy of an automated object detector (e.g., YOLOv5) under noise and attack

4.3.1. *Defense against Deep Learning (Object Detection/Classification)*

CBOD's design directly counters the operational principles of CNN-based object detectors. The combination of targeted cropping and aggressive resizing creates a significant 'domain gap'. Models like YOLO or Faster R-CNN, pre-trained on large, high-quality datasets like ImageNet or COCO, perform poorly on these degraded images without extensive, specific retraining.

The merging of four distinct, unrelated, and pre-processed images creates artificial edges and occlusions at the quadrant boundaries. These artifacts are not present in natural images and can confuse object detectors, leading to false positives or failed detections. The random boundary drawn around the object further occludes parts of the object and its immediate surroundings.

The applied noise corrupts the high-frequency features (edges, textures) that CNNs heavily rely on for classification. As mentioned, the low resolution of the base images makes denoising a destructive process, making it difficult for a bot to recover a 'clean' image for analysis.

4.3.2. *Defense against Adversarial Attacks*

While not explicitly designed with adversarial defense as the primary goal, the inherent randomness of the CBOD generation process provides a strong implicit defense. In the context of CBOD, an attacker cannot craft a single, universal adversarial perturbation. Every CAPTCHA instance is unique, with a different underlying image composition, boundary shape, boundary color, and noise pattern. The attack surface is a constantly moving target. Crafting a successful attack would require a new perturbation to be generated in real time for every single CAPTCHA, which is computationally prohibitive and defeats the purpose of an efficient automated attack.

4.4. *Role of Noise and Boundaries as an Adversarial Defense*

The random boundaries and mixed noise in CBOD are not merely visual distractions for humans, they act as query-agnostic adversarial perturbations. These elements intentionally introduce high-frequency disruptions into the image, targeting the feature extraction layers of CNNs to degrade their performance. To evaluate this effect, an experiment can be designed where noise density is systematically varied, and its impact is measured on both automated solver accuracy and human success rates. The objective is to identify an optimal 'sweet spot', a balance point where machine performance significantly deteriorates while human usability remains largely unaffected, as shown in Figure 10.

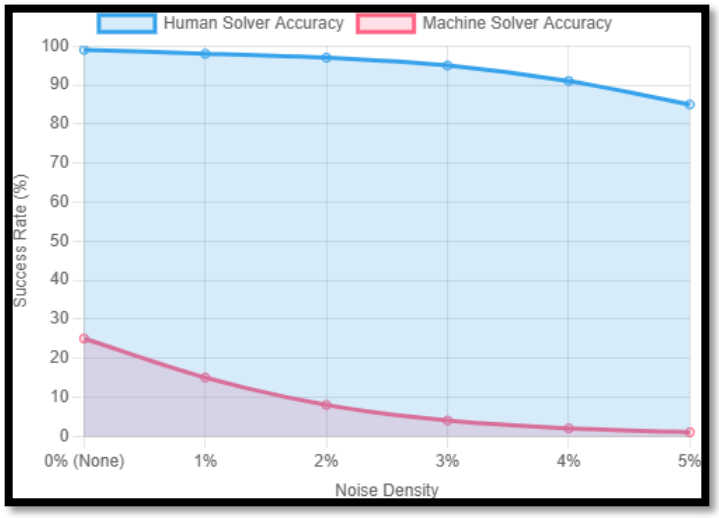


Figure 10: Conceptual trade-off analysis of CAPTCHA on the impact of increasing noise density

4.5. *Comparative Analysis*

In Table 1, we present a comprehensive summary that provides a clear, multidimensional comparison between CBOD and existing CAPTCHA schemes. The table incorporates our newly obtained technical security metrics to highlight the robustness and effectiveness of the proposed approach.

Table 1: Comparison of CBOD with other standard CAPTCHA systems based on various metrics

Metric	CBOD (Proposed)	ASIRRA	Text-based (e.g., Gimpy)	Google reCAPTCHA v3	Click-based (e.g., h Captcha)
Bot Success Rate	Low (Projected < 10%)	Medium (Vulnerable to CNNs)	High (Largely broken)	Very Low	Medium (20 – 40%)
Human Solve Time	~ 6.8*s*	~ 10*s*	~ 9*s*	~ 1*s* (Passive)	~ 8*s*
Dataset Dependency	Low (150 images)	High (Large DB of cats/dogs)	N/A (Generated)	N/A (Behavioral data)	High (Curated image sets)
Privacy Impact	Low (Stateless)	Low	Low	High (User tracking)	Medium to High (Tracking via behavioral cues)

4.6. *User Experience Evaluation*

To robustly validate our usability claims and assess the user experience of the proposed CBOD-CAPTCHA, we developed a dedicated website for testing its implementation and gathering user feedback. An online survey was conducted using Google Forms, through which we collected a total of 285 responses. Participants were selected to represent a diverse

demographic distribution, including age (from 10 to 48 years), educational background (from primary education to postgraduate degrees, as shown in Figure 11), and varying levels of internet usage frequency (daily, weekly, and monthly). We have used a get Time() function, which calculates the time between displaying a CAPTCHA & submitting it by the user and noticed that users took an average of 6.8 seconds to solve the CAPTCHA. This diversity was intended to enhance the generalizability and reliability of the findings.

*4.6.1. **Evaluation Metrics***

We collected data on a richer set of metrics to provide a holistic view of usability:

1. Success Rate: The percentage of participants who correctly solved the CAPTCHA on their first attempt.
2. Average Completion Time: The time elapsed from the moment the CAPTCHA was displayed until a successful submission.
3. Satisfaction: Measured using the industry-standard System Usability Scale (SUS), a 10-item questionnaire that yields a composite score from 0 to 100.
4. Error Rate: The average number of failed attempts before a successful submission.

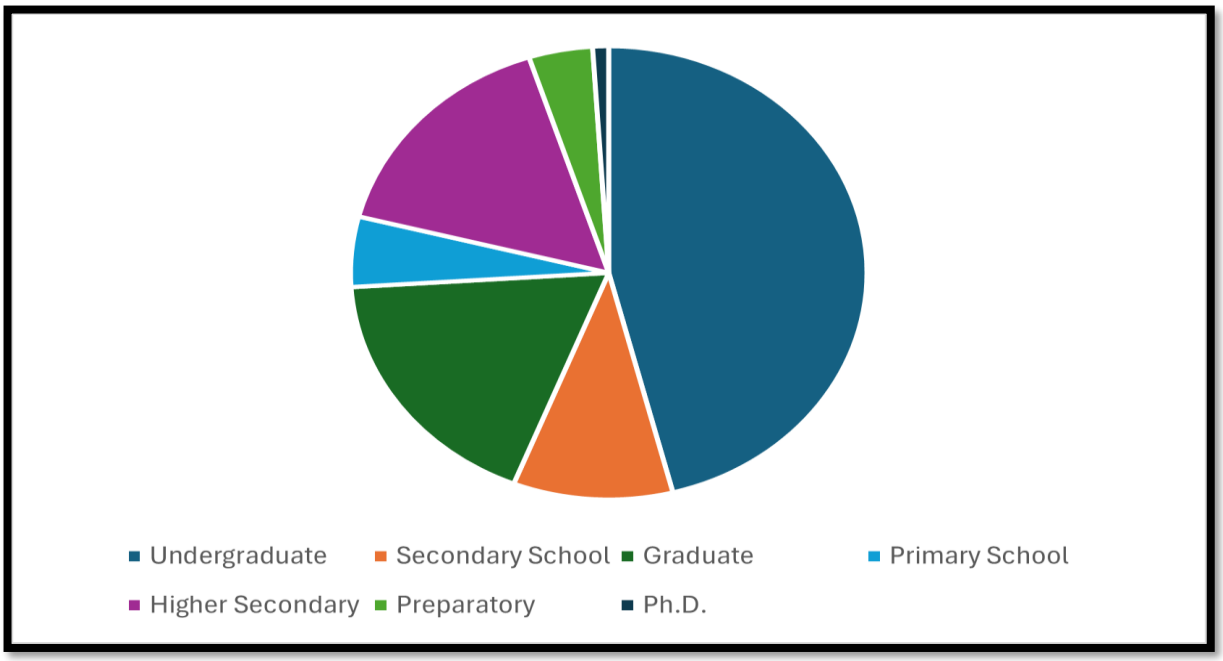


Figure 11: Educational level of participants for the evaluation of user experience

4.7. Users' preference

The human performance data demonstrated excellent usability. The first-attempt success rate for CBOD was 96%, with an average completion time of 6.8 seconds. The error rate was minimal, with users making, on average, only 4% incorrect attempts before success. Subjective feedback was overwhelmingly positive, with the CBOD scheme achieving an average System Usability Scale (SUS) score of 85.2. According to established benchmarks, a SUS score above 80.3 is considered 'excellent' and falls in the top 10% of systems tested. Figure 12 summarizes the users' usability and performance compared to a few standard CAPTCHA schemes.

5. Conclusion

This paper has introduced, formalized, and empirically validated the Color Boundary Object Detection (CBOD) CAPTCHA scheme. By presenting a structured algorithmic framework, a formal mathematical proof of combina-torial complexity, and rigorous empirical evaluations against deep learning architectures, this study demonstrates that CBOD effectively addresses the historical trade-off between system security and user execution friction. The primary operational advantage of the proposed scheme lies in its resource efficiency. Through combinatorial permu-tations, the pipeline yields an immense challenge space from a very small footprint of source images, presenting a lightweight alternative to conventional security systems that depend on massive, curated asset databases. Further-more, the multi layered defense architecture combining spatial down sampling, artificial occlusion, and multi modal noise is specifically calibrated to disrupt the internal feature representation layers of automated solvers, ensuring robust protection against modern computer vision threats.

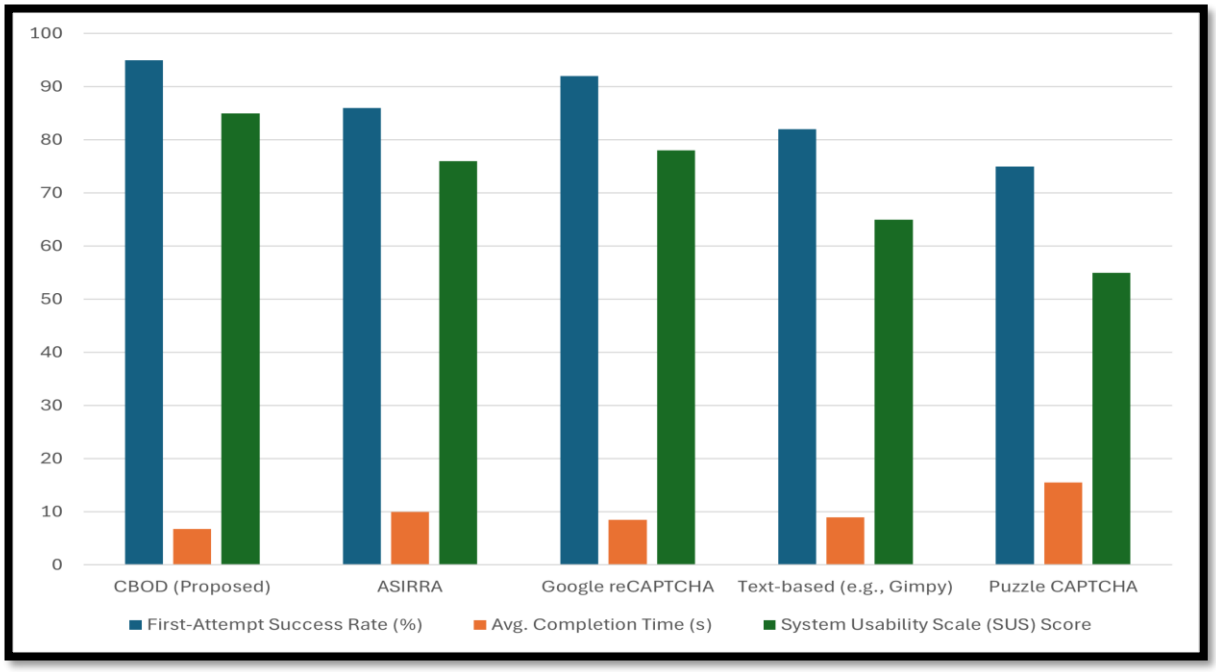


Figure 12: Usability of CBOD compare to other standard CAPTCHA schemes

Future work will focus on expanding the scope and accessibility of the CBOD architecture. We aim to integrate a formal adversarial noise model directly into the generation pipeline to provide analytical security guarantees against adaptive machine learning attacks. Additionally, we plan to design and evaluate an audio-based variant of the CBOD protocol to extend accessibility to visually impaired individuals. Ultimately, CBOD provides a practical, robust, and human-centric paradigm for securing contemporary web architecture against automated exploitation.

Availability of Data and Material

Available upon reasonable request.

Funding

The authors did not receive any funding from any organization for this research.

Acknowledgements

Not applicable.

References:

1. Abdullahi, M., Aliyu, S., & Junaidu, S. (2019). An enhanced intrusion detection system using honeypot and captcha techniques. Fudma Journal of Sciences, 3, 202–209.
2. Akrout, I., Feriani, A., & Akrout, M. (2019). Hacking google recaptcha v3 using reinforcement learning. arXiv preprint arXiv:1903.01003, .
3. Aldwairi, M., Abu-Dalo, A. M., & Jarrah, M. (2017). Pattern matching of signature-based ids using myers algorithm under mapreduce framework. EURASIP Journal on Information Security, 2017, 1–11.
4. Aldwairi, M., Mohammed, S., & Padmanabhan, M. L. (2020). Efficient and secure flash-based gaming captcha.
5. Journal of Parallel and Distributed Computing, 142, 27–35.
6. Aljarbou, Y. S. (2019). Improving of current captcha systems. In 2019 2nd International Conference on Computer Applications & Information Security (ICCAIS) (pp. 1–6). IEEE.
7. Althamary, I. A., & El-Alfy, E.-S. M. (2017). A more secure scheme for captcha-based authentication in cloud environment. In 2017 8th international conference on information technology (ICIT) (pp. 405–411). IEEE.
8. Baird, H. S., Coates, A. L., & Fateman, R. J. (2003). Pessimalprint: a reverse turing test. International Journal on Document Analysis and Recognition, 5, 158–163.
9. Baird, H. S., & Luk, M. (2003). Protecting websites with reading-based captchas. In Second International Web Document Analysis Workshop. Citeseer.
10. Baird, H. S., Moll, M. A., & Wang, S.-Y. (2005). Scattertype: A legible but hard-to-

segment captcha. In Eighth International Conference on Document Analysis and Recognition (ICDAR'05) (pp. 935–939). IEEE.

11. Bera, S., Misra, S., & Rodrigues, J. J. (2014). Cloud computing applications for smart grid: A survey. IEEE Transac-tions on Parallel and Distributed Systems, 26, 1477–1494.
12. Bursztein, E., Aigrain, J., Moscicki, A., & Mitchell, J. C. (2014). The end is nigh: Generic solving of text-based captchas. In 8th {USENIX} Workshop on Offensive Technologies ({WOOT} 14).
13. Bursztein, E., Bethard, S., Fabry, C., Mitchell, J. C., & Jurafsky, D. (2010). How good are humans at solving captchas? a large scale evaluation. In 2010 IEEE symposium on security and privacy (pp. 399–413). IEEE.
14. Bursztein, E., Martin, M., & Mitchell, J. (2011). Text-based captcha strengths and weaknesses. In Proceedings of the 18th ACM conference on Computer and communications security (pp. 125–138).
15. Choudhary, S., Saroha, R., Dahiya, Y., & Choudhary, S. (2013). Understanding captcha: text and audio based captcha with its applications. International Journal of Advanced Research in Computer Science and Software Engi-neering, 3.
16. Chow, R., Golle, P., Jakobsson, M., Wang, L., & Wang, X. (2008). Making captchas clickable. In Proceedings of the 9th workshop on Mobile computing systems and applications (pp. 91–94).
17. Copeland, J., & Proudfoot, D. (2009). Turing's test. In Parsing the Turing Test (pp. 119–138). Springer.
18. Derea, Z., Zou, B., Al-Shargabi, A. A., Thobhani, A., & Abdussalam, A. (2023). Deep learning based captcha recognition network with grouping strategy. Sensors, 23, 9487.
19. Ding, Z., Deng, G., Liu, Y., Ding, J., Chen, J., Sui, Y., & Li, Y. (2025). Illusioncaptcha: A captcha based on visual illusion. In Proceedings of the ACM on Web Conference 2025 (pp. 3683–3691).
20. Dinh, N. T., & Hoang, V. T. (2023). Recent advances of captcha security analysis: a short literature review. Procedia Computer Science, 218, 2550–2562.
21. Dionysiou, A., & Athanasopoulos, E. (2020). Sok: Machine vs. machine–a systematic classification of automated machine learning-based captcha solvers. Computers & Security, (p. 101947).
22. Elson, J., Douceur, J. R., Howell, J., & Saul, J. (2007). Asirra: a captcha that exploits interest-aligned manual image categorization. In ACM Conference on Computer and Communications Security (pp. 366–374). volume 7.
23. Gaggi, O. (2022). A study on accessibility of google recaptcha systems. In Proceedings of the 2022 Workshop on Open Challenges in Online Social Networks (pp. 25–30).
24. Ghavimi, F., & Chen, H.-H. (2014). M2m communications in 3gpp lte/lte-a networks: Architectures, service re-quirements, challenges, and applications. IEEE Communications Surveys & Tutorials, 17, 525–549.

25. Goodfellow, I. J., Bulatov, Y., Ibarz, J., Arnoud, S., & Shet, V. (2013). Multi-digit number recognition from street view imagery using deep convolutional neural networks. arXiv preprint arXiv:1312.6082, .
26. Gossweiler, R., Kamvar, M., & Baluja, S. (2009). What's up captcha? a captcha based on image orientation. In Proceedings of the 18th international conference on World wide web (pp. 841–850).
27. Hajjdiab, H., Ghazal, M., & Khalil, A. (2017). Random image matching captcha system. ELCVIA: Electronic Letters on Computer Vision and Image Analysis, 16, 0001–13.
28. Hasan, W. K. A. (2016). A survey of current research on captcha. Int. J. Comput. Sci. Eng. Surv.(IJCSES), 7, 141–157.
29. IJIRCST, I. (2024). A comparative analysis of cnn, rcnn & faster rcnn object detection algorithm for captcha break-ing. International Journal of Innovative Research in Computer Science and Technology (IJIRCST),
30. Kumar, M., Jindal, M., & Kumar, M. (2022). A systematic survey on captcha recognition: types, creation and breaking techniques. Archives of Computational Methods in Engineering, 29, 1107–1136.
31. Li, X., Chen, Y., Patibanda, R., & Mueller, F. (2021). vrcaptcha: exploring captcha designs in virtual reality. In Extended abstracts of the 2021 CHI conference on human factors in computing systems (pp. 1–4).
32. Moradi, M., Moradi, M., Palazzo, S., Rundo, F., & Spampinato, C. (2024). Image captchas: When deep learning breaks the mold. IEEE Access, .
33. Nanglae, N., & Bhattarakosol, P. (2024). Procaptcha: A profile-based captcha for personal password authentication. Plos One, 19, e0311197.
34. Nejati, H., Cheung, N.-M., Sosa, R., & Koh, D. C. (2014). Deepcaptcha: an image captcha based on depth perception. In Proceedings of the 5th ACM multimedia systems conference (pp. 81–90).
35. Osadchy, M., Hernandez-Castro, J., Gibson, S., Dunkelman, O., & Pérez-Cabo, D. (2017). No bot expects the deep-captcha! introducing immutable adversarial examples, with applications to captcha generation. IEEE Transactions on Information Forensics and Security, 12, 2640–2653.
36. Plesner, A., Vontobel, T., & Wattenhofer, R. (2024). Breaking recaptchav2. In 2024 IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC) (pp. 1047–1056). IEEE.
37. Shirali-Shahreza, M., & Shirali-Shahreza, S. (2007a). Collage captcha. In 2007 9th International Symposium on Signal Processing and Its Applications (pp. 1–4). IEEE.
38. Shirali-Shahreza, M., & Shirali-Shahreza, S. (2007b). Question-based captcha. In International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007) (pp. 54–58). IEEE volume 4.
39. Shirali-Shahreza, S., & Shirali-Shahreza, M. (2011). Multilingual highlighting captcha. In 2011 Eighth International Conference on Information Technology: New Generations (pp. 447–452). IEEE.

40. Singh, V. P., & Pal, P. (2014). Survey of different types of captcha. International Journal of Computer Science and Information Technologies, 5, 2242–2245.
41. Sivakorn, S., Polakis, J., & Keromytis, A. D. (2016). I'm not a human: Breaking the google recaptcha. Black Hat, 14, 1–12.
42. Tariq, N., Khan, F. A., Moqurrab, S. A., & Srivastava, G. (2023). Captcha types and breaking techniques: Design issues, challenges, and future research directions. arXiv preprint arXiv:2307.10239, .
43. Von Ahn, L., Blum, M., Hopper, N. J., & Langford, J. (2003). Captcha: Using hard ai problems for security. In International conference on the theory and applications of cryptographic techniques (pp. 294–311). Springer.
44. Von Ahn, L., Blum, M., & Langford, J. (2004). Telling humans and computers apart automatically. Communications of the ACM, 47, 56–60.
45. Wan, X., Johari, J., & Ruslan, F. A. (2024). Adaptive captcha: a crnn-based text captcha solver with adaptive fusion filter networks. Applied Sciences, 14, 5016.
46. Xu, X., Liu, L., & Li, B. (2020). A survey of captcha technologies to distinguish between human and computer. Neurocomputing, 408, 292–307.
47. Yan, J., & El Ahmad, A. S. (2008). Usability of captchas or usability issues in captcha design. In Proceedings of the 4th symposium on Usable privacy and security (pp. 44–52).
48. Zhang, Y., Gao, H., Pei, G., Luo, S., Chang, G., & Cheng, N. (2019). A survey of research on captcha designing and breaking techniques. In 2019 18th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/13th IEEE International Conference on Big Data Science and Engineering (Trust Com/Big Data SE) (pp. 75–84). IEEE.