SSRG International Journal of Electrical and Electronics Engineering
ISSN: 2348-8379/ https://doi.org/10.14445/23488379/IJEEE-V11I6P130

Volume 11 Issue 6, 282-304, June 2024
© 2024 Seventh Sense Research Group®

Original Article

Unveiling CAPTCHA Vulnerabilities: Breaking CAPTCHA Using Deep Learning Techniques and Design and Development of Robust CAPTCHA Technique

Dayanand[1], Wilson Jeberson[2], Klinsega Jeberson[3]

*[1,2,3]Department of Computer Science and Information Technology,
Sam Higginbottom University of Agriculture Technology and Sciences Prayagraj, Uttar Pradesh, India.*

[1]Corresponding Author : dayanand.defence@gmail.com

Received: 14 April 2024 Revised: 16 May 2024 Accepted: 14 June 2024 Published: 29 June 2024

Abstract *- CAPTCHA serves as a vital tool in distinguishing between human users and automated bots attempting to access websites. The Turing test, a fundamental concept in this domain, aids in discerning robot involvement in web security breaches, thereby safeguarding against automated access and potential harm. CAPTCHA, encapsulated by the acronym Completely Automated Public Turing Test to Tell Computers and Humans Apart, is instrumental in preventing undesirable activities by posing tasks that humans find simple to solve yet prove exceedingly challenging for robots, representing a distinct category of challenges. Initiatives aimed at improving CAPTCHA systems have led to the development of models aimed at accurately recognizing characters within CAPTCHA images. While traditional methods require the segmentation of characters before recognition, proposed models eliminate this step by processing the entire image at once, resulting in improved accuracy. Convolutional Neural Networks (CNNs) exhibit enhanced accuracy with segmented characters, while Multi-Task Convolutional Neural Network (MTCNN) excels in achieving similar accuracy without pre-processing. Object detection algorithms, including Faster R-CNN, YOLO, and SSD, offer even greater potential for breaking CAPTCHA by detecting objects within images. Gesture-based CAPTCHA challenges, while promising, encounter usability issues related to precision, reaction speed, and perceived level of challenge. To address this, a novel approach is proposed, leveraging hand-based gestures that are easily solvable by humans yet challenging for robots to replicate. Additionally, dynamic game-based CAPTCHA designs offer an aesthetically appealing and engaging interface, potentially motivating users to solve CAPTCHA challenges with minimal annoyance. The objective of this study is to explore the influence of different CAPTCHA tests on user experience across diverse populations. It includes a comprehensive study of multitask learning convolutional neural networks and employs methods of object detection algorithms, including Faster R-CNN, YOLO, and SSD object detection for CAPTCHA character recognition. The research also encompasses the design of gesture-based and dynamic game-based CAPTCHA challenges and compares various deep learning CAPTCHA breaking techniques with SSD object detection methods, analyzing existing and designed CAPTCHAs on multiple parameters.*

Keywords *- CAPTCHA, Gesture-based CAPTCHA, Dynamic game based CAPTCHA, RNN, Faster RCNN, SSD.*

1. Introduction

In the digital age, automated bots present a major threat to online security, prompting the creation of various measures to distinguish between human users and automated scripts. Among these measures, Completely Automated Public Turing Test to Tell Computers and Humans Apart (CAPTCHA) has become a widely adopted solution to prevent automated attacks. However, traditional CAPTCHA systems, which rely on distorted text or image recognition, have become increasingly vulnerable to advanced machine learning techniques. Given the vulnerabilities of traditional CAPTCHA systems, there is a need for new approaches to bolster online security. Researchers have turned to deep learning methods, which have demonstrated significant success in various pattern recognition tasks. Deep learning, a subset of machine learning, uses neural networks with multiple layers to autonomously extract features from data, making it particularly effective for complex classification problems such as CAPTCHA recognition. By utilizing deep learning methods, researchers aim to create more robust and reliable CAPTCHA systems that can withstand sophisticated attacks [1].

This research paper explores the use of deep learning methodologies for breaking CAPTCHA systems. It specifically examines the effectiveness of convolutional

This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/)

neural networks, recurrent neural networks, and other deep learning frameworks in solving CAPTCHA challenges. By evaluating the strengths and limitations of various deep learning approaches, this study seeks to provide insights into developing more secure CAPTCHA systems that are resistant to automated attacks [2].

The increasing sophistication of machine learning algorithms has made traditional CAPTCHA systems more vulnerable, prompting researchers to investigate more resilient alternatives. Among these alternatives, object detection techniques such as Faster RCNN, Single-Shot Detection (SSD), and You Only Look Once (YOLO) have shown promise in breaking CAPTCHA challenges.

Faster RCNN, a state-of-the-art object detection algorithm, revolutionized the field by introducing Region Proposal Networks (RPNs) for efficient object localization and classification. By simultaneously predicting object bounding boxes and category labels, Faster RCNN achieves high precision and speed, making it suitable for CAPTCHA-breaking tasks. Single-shot Detection streamlines the object detection process by eliminating the need for separate region proposals and detection stages. This streamlined approach allows for faster inference times, making Single-Shot Detection an attractive option for real-time applications such as CAPTCHA recognition [3].

The YOLO technique represents a significant advancement in object detection, proposing an integrated framework that directly predicts bounding boxes and class probabilities without multiple passes through the neural network. YOLO's unified approach offers exceptional speed and efficiency, making it ideal for resource-constrained environments and time-sensitive tasks. In the context of CAPTCHA breaking, YOLO's ability to quickly identify and classify objects within complex images holds great potential for enhancing attack efficiency and scalability [4].

This research paper aims to evaluate the effectiveness of object detection methods, including Faster RCNN, Single-Shot Detection, and YOLO, in breaking CAPTCHA challenges. By examining the strengths and weaknesses of each approach, this study seeks to provide valuable insights into the development of more robust CAPTCHA systems capable of withstanding advanced attacks [5].

Ensuring the security of online platforms is crucial for protecting against unauthorized access, data breaches, and malicious activities. However, the effectiveness of traditional text-based CAPTCHAs has been challenged by advancements in machine learning and computer vision technologies, which have enabled automated bots to bypass CAPTCHA challenges with increasing accuracy. To address these challenges and enhance online platform security, there is growing interest in developing robust CAPTCHA techniques that are resistant to automated attacks while remaining user-friendly for legitimate users. This research paper focuses on the design and implementation of two innovative CAPTCHA techniques: hand gesture-based CAPTCHA and dynamic game-based CAPTCHA.

The first part of the paper investigates the concept of hand gesture-based CAPTCHA, which leverages the intuitive nature of human gestures to verify user identity. By capturing and analyzing hand movements and gestures, this CAPTCHA technique aims to create challenges that are easy for humans to solve but difficult for automated bots. Using computer vision algorithms and machine learning models, hand gesture-based CAPTCHA provides a novel approach to enhancing security while ensuring a smooth user experience [6].

The second part of the research paper explores the development of dynamic game-based CAPTCHA, which incorporates gamification elements to create engaging and interactive challenges for users. Unlike traditional text-based CAPTCHAs, dynamic game-based CAPTCHA presents users with a series of mini-games or puzzles that must be solved to verify their human identity. By integrating game mechanics and visual storytelling, dynamic game-based CAPTCHA offers a compelling alternative to traditional CAPTCHA schemes, enhancing user engagement and satisfaction while deterring automated bots [7].

Through empirical studies, usability testing, and comparative analysis, the effectiveness and usability of both hand gesture-based CAPTCHA and dynamic game-based CAPTCHA techniques are evaluated. By assessing factors such as security, user experience, and resistance to automated attacks, the potential of these innovative CAPTCHA solutions to enhance the security of online platforms and protect against emerging threats is demonstrated.

The potential application of object detection techniques in breaking CAPTCHAs lies in improving the understanding of CAPTCHA vulnerabilities. By leveraging object detection methods, cybersecurity professionals can identify weaknesses in CAPTCHA systems, leading to enhanced defenses against automated attacks and stronger overall security measures.

2. Literature Review

CAPTCHA serves as a vital tool in distinguishing between human users and automated bots attempting to access websites. The Turing test, a fundamental concept in this domain, aids in discerning robot involvement in web security breaches, thereby safeguarding against automated access and potential harm. CAPTCHA, encapsulated by the acronym Completely Automated Public Turing Test to Tell Computers and Humans Apart, is instrumental in preventing undesirable activities by posing tasks that humans find simple to solve yet prove exceedingly challenging for robots, representing a distinct category of challenges.

2.1. Early Developments in CAPTCHA Systems

The necessity for CAPTCHAs emerged from the need to combat website and search engine misuse perpetrated by automated programs. In 1997, AltaVista encountered issues with the automated submission of URLs to their search engines, leading Andrei Broder, Chief Scientist of AltaVista, and his team to develop a solution. They developed a filter that generated randomly printed text, which only humans could decipher, thus thwarting machine readers. This innovative approach proved highly effective, leading to a significant reduction of “spam-add-ons” by 95% within a year. Subsequently, a patent for this method was released in 2001.

In 2000, Yahoo's Messenger chat service faced challenges with bots posting advertising links to disrupt chat room conversations. To address this, Yahoo collaborated with Carnegie Mellon University to create EZ-GIMPY, a new CAPTCHA system. EZ-GIMPY randomly selected a word from the dictionary, then distorted it using various image occlusions, and prompted users to input the distorted word as a verification measure.

Moreover, slashdot.com conducted a poll in November 1999 to identify the top CS colleges in the US. However, students from CMU and MIT manipulated the system by deploying bots to cast multiple votes for their colleges. This event underscored the need for CAPTCHAs to safeguard online polls, ensuring that only human users could participate in such surveys.

2.2. Advanced CAPTCHA Breaking Techniques

[8] To bolster the security of text-based CAPTCHAs, developers have implemented various measures aimed at thwarting automated attacks. These measures include distorting or rotating characters, adding noise, and employing complex backgrounds. Early studies on segmentation-based attacks typically involved pre-processing, segmentation, and recognition steps. However, this paper presents a systematic examination of the efficacy of these defense mechanisms against various deep-learning attacks.

This analysis revealed that traditional segmentation-based approaches may no longer suffice in the face of deep learning attacks. To address this challenge, We introduce a novel, comprehensive solution that leverages deep learning techniques. This method represents a significant advancement, as demonstrated by our successful breaking of the challenging version of Google's CAPTCHA boasts an impressive success rate of 98.3%.

Additionally, we have attained notable success rates ranging from 74.8% to 97.3% in bypassing a considerable number of real-world text CAPTCHAs implemented by the top 50 websites, as per Alexa.com rankings. The research sheds light on the evolving landscape of CAPTCHA security and underscores the importance of adopting advanced defense mechanisms to counter emerging threats posed by deep learning-based attacks.

Over the past two decades, CAPTCHAs have emerged as a robust means of authentication, effectively safeguarding sensitive information by preventing unauthorized access. OCR and Non-OCR-based CAPTCHAs are the two main categories, each presenting its own set of challenges for attackers [9]. Due to the diverse range of schemes employed by different systems, no universal method exists for breaking all CAPTCHAs, making it challenging to devise generalized attack strategies.

2.2.1. Focus on OCR CAPTCHA Systems

This paper concentrates on breaking an OCR CAPTCHA system, which poses unique obstacles to automated recognition. Proposed models offer a reliable approach to recognizing characters within CAPTCHA images. Unlike traditional methods that involve segmentation followed by character recognition, the approach in this paper streamlines the process by directly processing the entire image using Convolutional Neural Networks (CNNs).

While CNNs trained on segmented characters achieve marginally higher accuracy, the Multi-task Learning CNN (MLT-CNN) demonstrates comparable accuracy without the need for pre-processing, representing a significant advantage in terms of efficiency and performance.

2.2.2. Contribution to CAPTCHA Vulnerability Mitigation

This research contributes to the on-going efforts to understand and mitigate the vulnerabilities of OCR-based CAPTCHA systems. By exploring advanced techniques and improving upon existing methodologies, it aims to pave the way for enhanced security measures in the digital landscape.

[10] Recurrent Neural Networks (RNNs) are widely recognized for their effectiveness in handling sequential data. Recent advancements have addressed some of the inherent limitations of RNNs, resulting in significant performance improvements. One notable development is the Connectionist Temporal Classification (CTC) model, which enhances the capabilities of keras models to implement the CTC approach. By leveraging efficient methods provided by TensorFlow, the CTC model facilitates both training and prediction processes. During training, it computes the CTC loss, and during prediction, it performs CTC interpreting and decoding. The principal techniques of Keras Models have been adapted in the CTC model, allowing for seamless integration into standard workflows.

A key distinction of the CTC Model lies in its input structure, which encompasses observation sequences, input observation durations, label sequences, and label durations. This structured input enables the model to handle sequential data effectively and produce accurate predictions.

Additionally, the CTC Model offers transparent computation of evaluation metrics such as label error rate and sequence error rate, providing a clear understanding of the model's effectiveness. Overall, the CTC model signifies a noteworthy progression in utilizing RNNs for sequential data processing, offering enhanced capabilities and improved performance for a wide range of applications.

[11] The study introduces an end-to-end deep Convolutional Neural Network-Recurrent Neural Network (CNN-RNN) model for captcha recognition, emphasizing the identification of text captchas with four characters. The CNN and RNN architecture comprises two primary elements: a deep residual Convolutional Neural Network (CNN) and a variant Recurrent Neural Network (RNN), specifically a two-layer Gated Recurrent Unit (GRU) network.

Initially, the CNN component is designed to utilize the residual network structure to extract features effectively from input CAPTCHA images. Leveraging the capabilities of deep learning, the CNN model is trained to capture intricate patterns and details present in captcha images, facilitating robust feature extraction. Subsequently, the extracted features pass through the variant RNN network, consisting of two layers of GRU units. The RNN component captures the deep internal characteristics of the CAPTCHA images, including subtle variations and contextual information crucial for accurate recognition. Ultimately, the output sequence generated by the RNN network corresponds to the predicted characters of the 4-character captcha.

Integrating both CNN and RNN components into an end-to-end architecture, the proposed model achieves comprehensive feature extraction and sequence prediction, enabling accurate and efficient recognition of text captchas. Overall, the study demonstrates the efficacy of leveraging deep learning methods, particularly CNNs and RNNs, for captcha recognition tasks, providing a robust and scalable solution for combating automated attacks and ensuring the security of online systems.

2.3. Object Detection in CAPTCHA Breaking

Object detection algorithms, including Faster R-CNN, YOLO, and SSD, offer significant potential for breaking CAPTCHAs by detecting objects within images. Faster R-CNN revolutionized the field by introducing Region Proposal Networks (RPNs) for efficient object localization and classification. This approach achieves high precision and speed, making it suitable for CAPTCHA-breaking tasks.

Single-Shot Detection (SSD) streamlines the object detection process by eliminating the need for separate region proposals and detection stages, allowing for faster inference times. YOLO represents a significant advancement in object detection by proposing an integrated framework that directly predicts bounding boxes and class probabilities, offering exceptional speed and efficiency.

2.4 Innovative CAPTCHA Approaches

[12] A gesture-based CAPTCHA challenge serves as a security measure to prevent malware from accessing network resources through mobile devices. Mobile devices, equipped with various sensors capable of recording physical movements, provide data from accelerometers and gyroscopes as inputs for innovative CAPTCHAs, capturing the device's manipulation. An experimental study conducted with a varied participant pool revealed that individuals demonstrate proficiency in solving such CAPTCHA challenges swiftly. However, challenges were noted for older individuals, particularly with the utilization of accelerometer readings.

[13] Emerging images offer a straightforward means for humans to perceive objects, but they present a significant challenge for computers to decipher embedded content. Initially considered resistant to automated attacks, the EI-Nu CAPTCHA was celebrated as a user-friendly implementation of emerging image CAPTCHA. However, investigations have revealed multiple security vulnerabilities within this design, culminating in a comprehensive attack against the scheme. The primary weakness lies in the utilization of continuous camera projection onto 2D objects, resulting in a consistent but restricted flow of contour data across successive frames. Consequently, virtually all automated attacks relying on accumulated information are rendered ineffective against the revised construction.

The improved security is attributed to dynamic background content, which prevents the creation of a static background mask during the attack, and the varying camera projections of pseudo-3D objects over time, thereby diminishing the potential to reconstruct object shapes by cross-referencing between frames. While this advancement in security inevitably entails a reduction in contrast to the recently demonstrated insecure 2D variant (EI-Nu), the usability of this method may still be deemed acceptable for high-security web applications. The research highlights the previously overlooked security and usability issues associated with emerging image CAPTCHAs.

[14] The paper presents a new approach/methodology for conducting tests with minimal effort while effectively achieving their intended purpose. This method holds promise for future applications across numerous domains. The outlined process involves converting input gesture images following morphological filtering, where the pixels outlining the gesture boundary are quantified and cross-referenced with gesture images stored in a database. This process determines whether the user-provided gesture aligns with the CAPTCHA prompt.

Notably, this approach is user-friendly and serves the CAPTCHA purpose proficiently. Furthermore, its versatility extends to various other security-related applications. It is noteworthy that the input is captured through the system camera. While the suggested approach may be vulnerable to brute force attacks when implemented with a small database, its efficacy improves significantly with a larger dataset. Additionally, an alternative approach involves utilizing instrumental gloves to capture raw data for gesture recognition.

[15] A direct implication of the research findings is the necessity for further exploration into DCG CAPTCHAs to enhance their resistance against automated attacks while preserving usability. A prominent vulnerability identified in DCG CAPTCHA instances is the static nature of the game background, a framework utilized to isolate foreground objects from game frames, compromising the security of CAPTCHA.

In response to this vulnerability, one potential approach is to develop DCG CAPTCHA variants featuring dynamic backgrounds. These variations could incorporate random noise, objects with dynamically fluctuating color and luminance, instances of object occlusion, backgrounds that evolve dynamically, and various combinations thereof. Introducing such fluctuations would bolster resistance against the suggested attack; however, they could also be vulnerable to alternative, more advanced attacks and might exhibit reduced usability. Further research is warranted to assess the security and usability implications of these variations thoroughly.

[19] A robust CAPTCHA should present a challenge for automated systems while remaining easily solvable by humans. Many current CAPTCHA systems rely on exploiting the limitations of automated visual recognition, such as recognizing text or other elements present in an image. However, traditional CAPTCHAs have become increasingly vulnerable due to advancements in visual recognition technologies.

This paper explores the integration of visual reasoning into CAPTCHA design. The proposed CAPTCHA prompts users to recognize a particular object(s) within an image based on a provided text query. While humans can easily comprehend the textual query and apply sophisticated rationale to interpret the image, this task remains challenging and demanding in terms of computational resources for machines. An overview of the CAPTCHA design is provided, along with usability assessments and security experiments to evaluate its effectiveness.

The increasing sophistication of machine learning algorithms has made traditional CAPTCHA systems more vulnerable, prompting researchers to investigate more resilient alternatives. This research explores the use of advanced deep learning and object detection methods, including Faster RCNN, Single-Shot Detection (SSD), and You Only Look Once (YOLO), to break CAPTCHA systems, addressing vulnerabilities in traditional text-based CAPTCHA that have become susceptible to sophisticated machine learning attacks. Unlike existing research, which primarily focuses on conventional recognition methods, this study investigates the application of convolutional and recurrent neural networks for CAPTCHA breaking and introduces innovative CAPTCHA techniques such as hand gesture-based and dynamic game-based CAPTCHAs. These new approaches aim to enhance security by leveraging intuitive human gestures and interactive gamification, thus providing more robust defenses against automated bots while maintaining usability for legitimate users. Through empirical studies and comparative analysis, the research highlights the strengths and limitations of these novel methods, offering valuable insights into the development of more secure CAPTCHA systems resistant to advanced automated attacks.

3. Types of CAPTCHA and its Working

A CAPTCHA serves as a means to discern whether the interacting entity is a human or a robot. With the pervasive use of the internet, implementing a Turing test becomes crucial to thwart various website attacks. Therefore, safeguarding webpage security, CAPTCHA proves exceptionally beneficial, encompassing deformed text, mathematical calculations, OTP (One Time Password), audio, 3D, graphical, and gaming CAPTCHA. Gaming CAPTCHA emerges as a highly secure option gaining traction. Many gaming CAPTCHAs rely on straightforward logic, allowing users to solve them by dragging objects to the target position. However, some pose significant challenges even for humans due to their complexity. Ideally, a CAPTCHA should be easy enough for a human to solve within seconds but nearly impossible for a robot to crack.

3.1. Text CAPTCHA

Text-based CAPTCHAs necessitate users to identify and input distorted text presented within an image. Users must correctly decipher the distorted text to pass the CAPTCHA challenge.

3.2. Image-Based CAPTCHA

Image-based CAPTCHAs display images to users and require them to identify specific objects, patterns, or elements within the image. Users must select the correct options or answer questions related to the image.

3.3. Audio-Based CAPTCHA

Description: Audio-based CAPTCHAs present users with audio clips containing spoken numbers, letters, or words. Users must listen to the audio and transcribe the content accurately to pass the CAPTCHA challenge.

3.4. Checkbox CAPTCHA

Checkbox CAPTCHAs require users to select specific checkboxes or verify certain conditions to prove their human identity. Users must perform the required action correctly to pass the CAPTCHA challenge.

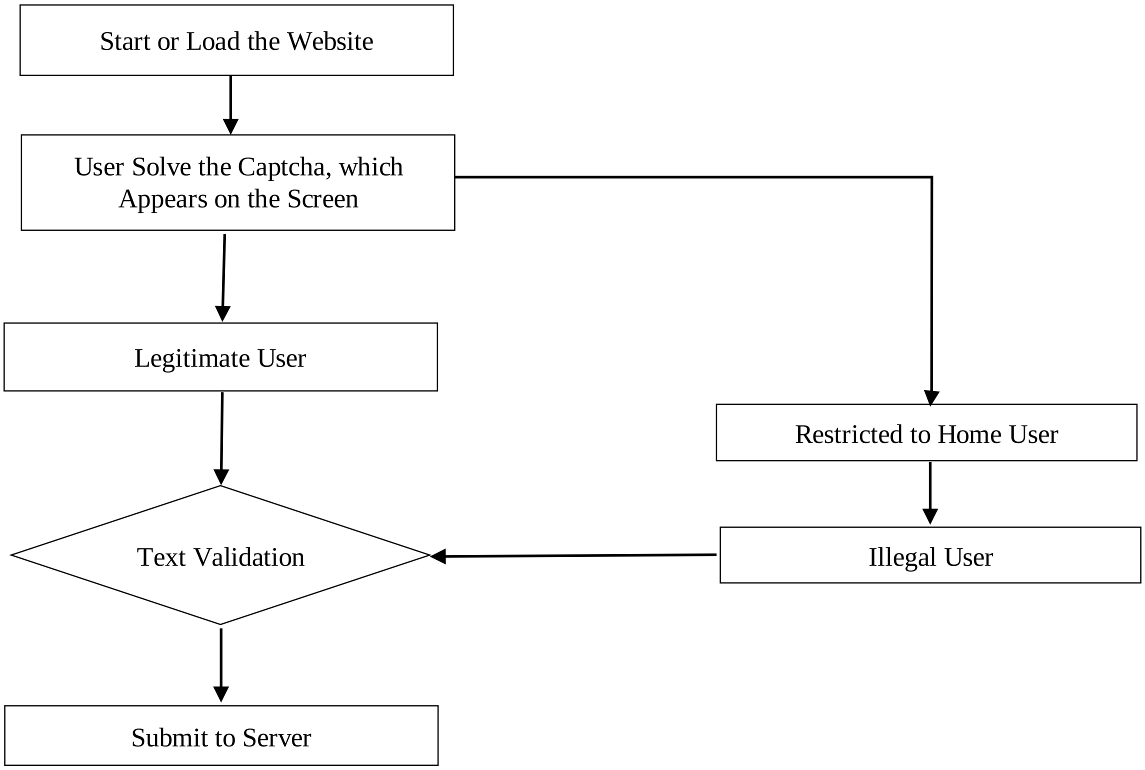

Fig. 1 Working of the CAPTCHA model

3.5. Geometric CAPTCHA

Geometric CAPTCHAs present users with geometric shapes, patterns, or puzzles that they must solve to demonstrate their human identity. Users may be required to manipulate or rearrange shapes, identify patterns, or complete geometric sequences.

3.6. Slider CAPTCHA

Slider CAPTCHAs require users to interact with a slider control to complete a specific action, such as sliding the control to a designated position or matching a target shape. Users must act accurately to pass the CAPTCHA challenge.

3.7. Math-Based CAPTCHA

Math-based CAPTCHAs present users with mathematical equations or arithmetic problems that they must solve to demonstrate their human identity. Users must correctly solve the equations or problems to pass the CAPTCHA challenge.

3.8. Grid Based CAPTCHA

Grid-based CAPTCHAs offer users a grid featuring images, patterns, or symbols accompanied by a question or instruction. Users must select the correct images or patterns according to the question or instruction to pass the CAPTCHA challenge.

3.9. Time-Based CAPTCHA

Time-based CAPTCHAs require users to complete a specific action within a limited time frame, such as clicking a button or entering a code, to demonstrate their human identity. Users must complete the action before the time expires to pass the CAPTCHA challenge.

3.10. Puzzle-Based CAPTCHA

Puzzle-based CAPTCHAs present users with visual or logical puzzles that they must solve to demonstrate their human identity. Users must solve the puzzles correctly, such as rearranging jumbled images or completing missing parts of an image, to pass the CAPTCHA challenge.

4. Breaking of CAPTCHA through Object Detection Deep Learning Techniques and Proposed Models

CAPTCHA has long been regarded as a robust defense mechanism against automated attacks on web applications. However, with the progression of deep learning methods, especially in object detection algorithms, the security of CAPTCHA systems has been challenged.

Object detection is a method that can locate and recognize entities within an image, such as faces, cars, or animals. It can also be employed to detect the characters depicted in a CAPTCHA image and read them correctly.

To break CAPTCHAs using object detection, it is essential to utilize equations that describe key processes such as feature extraction, character localization, and recognition.

```
# Function to break CAPTCHA image
function BreakCaptchaImage(image_path):
    # Load the CAPTCHA image
    image = ImageProcessingLibrary.LoadImage(image_path)
    # Preprocess the image
    preprocessed_image = PreprocessImage(image)
        # Extract characters (objects) from the image
    character_objects = ExtractCharacters(preprocessed_image)
        # Recognize characters from extracted objects
    captcha_text = RecognizeCharacters(preprocessed_image, character_objects)
        return captcha_text
# Function to preprocess the image
function PreprocessImage(image):
    # Convert image to grayscale
    grayscale_image = ImageProcessingLibrary.ConvertToGrayscale(image)
        # Apply noise reduction if needed
    denoised_image = ImageProcessingLibrary.Denoise(grayscale_image)
        return denoised_image
# Function to extract character objects from the image
function ExtractCharacters(image):
    # Detect features using a pre-trained object detection model
    detected_features = ObjectDetectionLibrary.DetectFeatures(image)
        character_objects = []
        for feature in detected_features:
        if feature.label == 'character':
            character_objects.append(feature)
        return character_objects

# Function to recognize characters from extracted objects
function RecognizeCharacters(image, character_objects):
    recognized_characters = ““
        for character_object in character_objects:
        # Extract bounding box for the character
        bounding_box = character_object.bounding_box
                # Crop the character from the image
        character_image = ImageProcessingLibrary.CropImage(image, bounding_box)

        # Recognize character using OCR
        recognized_character = OCRLibrary.RecognizeCharacter(character_image)

        recognized_characters += recognized_character

    return recognized_characters
```

Fig. 2 Working of CAPTCHA breaking with object detection

4.1. Feature Extraction

Feature extraction entails the process of extracting pertinent gathering information from the CAPTCHA image, like edges, shapes, and colors. One effective approach is employing a convolutional neural network, a deep learning architecture capable of automatically learning features. The equation for a convolutional layer in a CNN is:

$$y_{i,j,k} = \sum_{l,m,n} x_{i+l-1,j+m-1,n} \cdot w_{l,m,n,k} + b_k \qquad (1)$$

Where x represents the input image data, y denotes the resulting feature map, w signifies the filter weight, b is the bias term and i, j, k, l, m, n are indices for the dimensions.

4.2. Character Localization

Character localization involves identifying the position and dimensions of each character within the CAPTCHA image. A Region Proposal Network (RPN) is commonly used for this task, generating candidate regions likely to contain objects. The equation for the Region Proposal Layer in an RPN is:

$$r_{i,j,k} = \sum l, m, n\, y_i + l - 1, j + m - 1, n. v_{l,m,n,k} + c_k \quad (2)$$

Where y represents the input feature map, r denotes the output region proposal, v signifies the region weight, c is the region bias term, and i,j,k,l,m,n are indices for the dimensions.

4.3. Character Recognition

Character recognition involves determining the class and label of each character in the CAPTCHA image. This is typically achieved using a classifier network, which predicts the class of the input character. The equation for a densely connected layer within a classifier network is:

$$z_k = \sum_i r_i \cdot w_{i,k} + b_k \qquad (3)$$

Where r is the input region proposal, z denotes the output score, w denotes the weight, b signifies the bias term, and i and k are indices for the input and output nodes.

In this section, we delve into the methodologies and challenges associated with breaking CAPTCHA using deep learning for object detection techniques.

4.4. Introduction to Object Detection Deep Learning Techniques

Object detection algorithms, such as Faster R-CNN, SSD, and YOLO, have revolutionized the realm of computer vision by enabling the accurate localization and classification of objects within images. These techniques leverage Convolutional Neural Networks (CNNs) to detect objects of interest, making them highly effective in various applications, including image recognition, surveillance, and autonomous driving [16-19].

4.5. Challenges in Breaking CAPTCHA Using Object Detection

Despite their effectiveness in various domains, object detection deep learning techniques pose significant challenges when applied to breaking CAPTCHA systems. CAPTCHAs are specifically designed to be resistant to automated recognition, requiring robust solutions to overcome distortion, noise, and obfuscation techniques employed in CAPTCHA images. Moreover, the dynamic nature of CAPTCHA challenges further complicates the task of object detection, necessitating advanced algorithms capable of accurately identifying and localizing characters or objects within the CAPTCHA image [20].

4.6. Methods for Breaking CAPTCHA Using Object Detection

Several approaches have been proposed to break CAPTCHA using object detection deep learning techniques. These methods typically involve training object detection models on labeled datasets of CAPTCHA images, where the models learn to detect and classify individual characters or objects within the CAPTCHA. Techniques such as data augmentation, transfer learning, and ensemble learning are often employed to enhance the resilience and precision of the object detection models [21]. Let us denote the CAPTCHA image as I and the set of characters within the CAPTCHA image as C={C_1, C_2,..., Cn}, where “n” represents the number of characters within the CAPTCHA. The following equation can represent the process of breaking the CAPTCHA using object detection:

$$\mathit{Cpredicted} = \text{ObjectDetectionModel(I)} \qquad (4)$$

Where Cpredicted ={Cpredicted_1, Cpredicted_2,..., Cpredicted_n} is the set of predicted characters obtained from the object detection model.

4.7. Evaluation and Impact

The effectiveness of breaking CAPTCHA using object detection techniques is evaluated based on various metrics such as accuracy, precision, recall, and computational efficiency. While these techniques have demonstrated high success rates in breaking CAPTCHA challenges, their widespread adoption raises concerns about the security implications for online systems. The impact of such vulnerabilities on web security and potential countermeasures to mitigate these risks are essential areas for further research and development [22].

5. Types of Proposed Object Detection Techniques for Breaking of CAPTCHA

5.1. Faster RCNN

Faster R-CNN (Region-based Convolutional Neural Network) stands as a cutting-edge object detection algorithm, exhibiting exceptional performance across diverse computer vision endeavors, even surpassing CAPTCHA challenges. In this segment, we delve into the fundamentals and utilizations of Faster R-CNN within the realm of breaking CAPTCHA systems.

5.1.1. Introduction to Faster R-CNN

Faster R-CNN constitutes a two-stage object detection framework comprising two primary components: a Region Proposal Network (RPN) and a Region-based Convolutional Neural Network (R-CNN). The RPN is responsible for generating region proposals or candidate bounding box proposals for potential objects within the image, while R-CNN refines these proposals and classifies them into different object categories. By integrating these two stages, Faster R-CNN achieves cutting-edge performance in both accuracy and speed [3].

5.1.2. Principles of Faster R-CNN for CAPTCHA Breaking

In the context of breaking CAPTCHA challenges, Faster R-CNN operates by first generating region proposals for individual characters or objects within the CAPTCHA image. These suggestions are then refined and classified by the R-CNN component to identify the characters present in the CAPTCHA. By leveraging the hierarchical features learned by the deep convolutional layers, Faster R-CNN can effectively localize and classify objects, making it well-suited for breaking CAPTCHA systems [23].

5.1.3. Steps for CAPTCHA Breaking Using Faster R-CNN

- Step 1: Dataset Collection:Gather a dataset of CAPTCHA images. You may consider creating your dataset or using publicly available datasets like Google's reCAPTCHA dataset or CAPTCHA-4 dataset [24].
- Step 2: Preprocessing: Pre-process the CAPTCHA images to enhance the model's performance. This may involve resizing, normalization, and noise reduction techniques [25].
- Step 3: Labeling: Label the dataset by outlining bounding boxes around the characters in the CAPTCHA images. This step is crucial for training an object detection model.
- Step 4: Training: Train the Faster R-CNN technique to the annotated dataset. Fine-tune a pre-existing model trained on an extensive dataset if available to improve performance.
- Step 5: Testing: Evaluate the trained model on a separate test set to measure its performance in detecting characters within CAPTCHA images.
- Step 6: Post-Processing: Implement post-processing techniques to extract characters from the detected bounding boxes and assemble them into a readable CAPTCHA.
- Step 7: Breaking CAPTCHA: Use the trained Faster R-CNN model to identify characters in fresh CAPTCHA images, followed by post-processing to extract and assemble the characters.

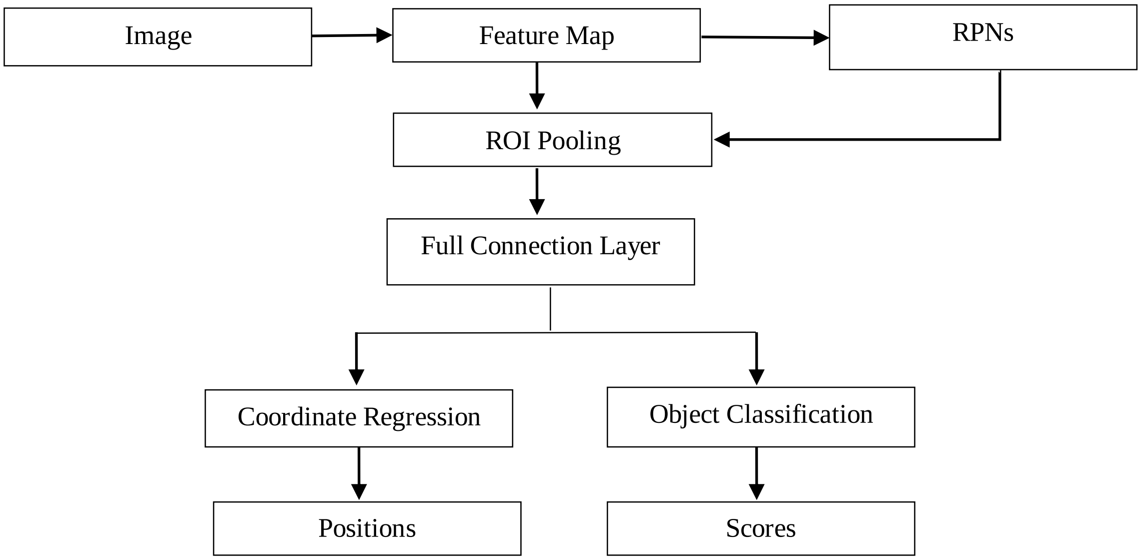

Fig. 3 Working of Faster RCNN algorithm

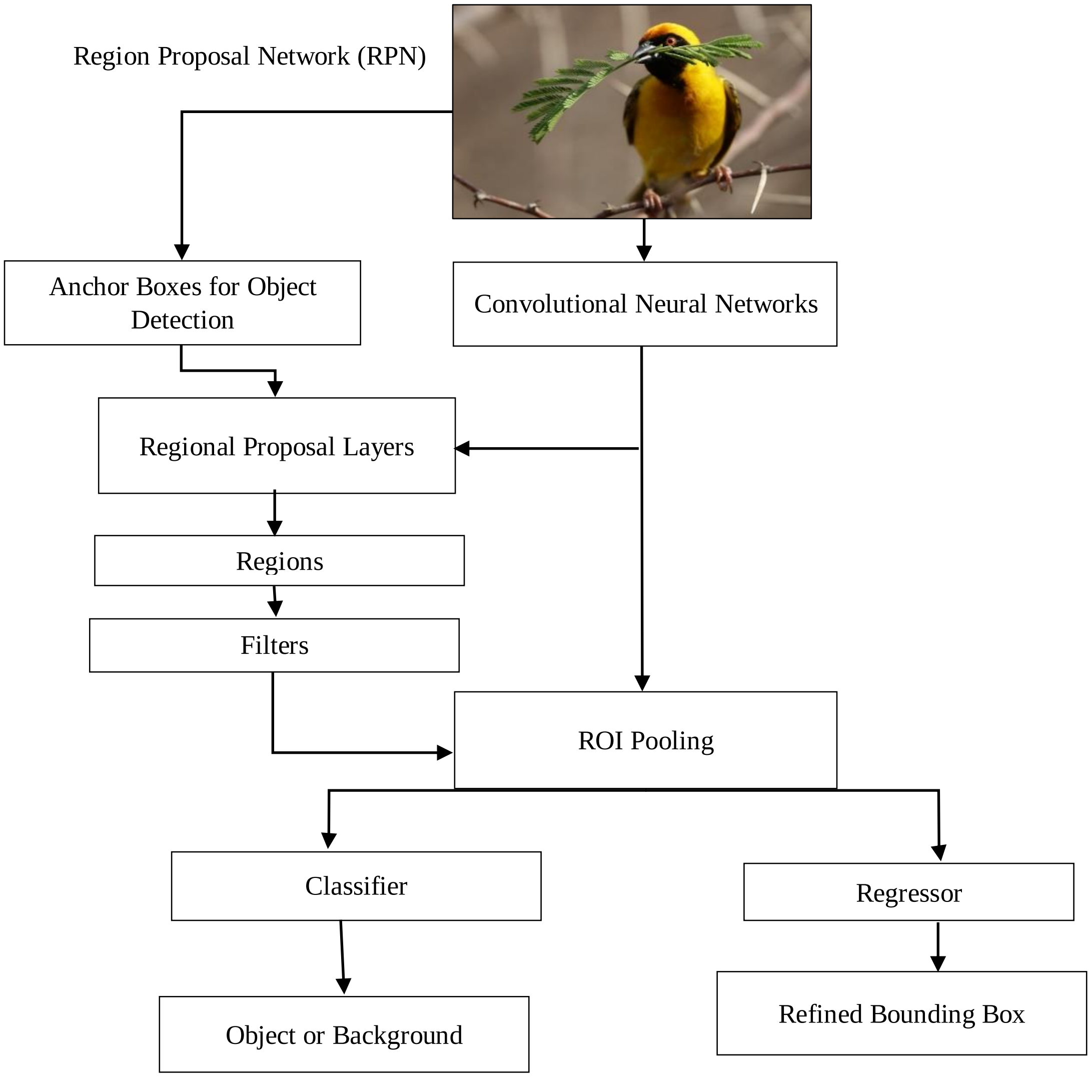

Fig. 4 Breaking of CAPTCHA with Faster RCNN algorithm

5.1.4. Working

- Step 1: Region Proposal Network (RPN): The input image undergoes feature extraction, producing feature maps. This process is facilitated by the Region Proposal Network (RPN), a sub-CNN network typically utilizing architectures like VGG or Alex Net. In this instance, VGG is employed to generate the base feature map. Through RPN, the image is partitioned into background and foreground regions.

The methodology of the Region Proposal Network (RPN) in Faster R-CNN can be expressed by the following equations:

Anchor Boxes Generation

$$A = \{ (w_i, h_j) \mid i = 1,...,Ni,\ j = 1,,...,Nj\} \quad (5)$$

Where A represents the set of anchor boxes generated over the input image, wi and hj denote the width and height of every anchor box, respectively, and Ni and Nj represent the number of anchor box widths and heights.

Convolutional Layer

$$F = \mathrm{Conv}(I) \quad (6)$$

Here, "I" represents the input image, while "F" denotes the feature map obtained after applying a convolutional layer applied to the input image.

Sliding Window

$$P = \mathrm{Slide}\ (F) \quad (7)$$

P represents the set of anchor box positions obtained by moving a small window across the feature map F.

Anchor Box Prediction

$$B = \mathrm{RPN}\ (P) \quad (8)$$

B represents the set of predicted bounding boxes generated by the Region Proposal Network (RPN) based on the anchor box positions P.

Classification and Regression

$$C, R=\mathrm{RPN\_Classification\_Regression}\ (B) \quad (9)$$

C represents the predicted class probabilities for each bounding box, while "R" signifies the predicted bounding box regression offsets. The functions Conv, Slide, RPN, and RPN_Classification_Regression represent the operations performed by the convolutional layer, sliding window mechanism, and Region Proposal Network (RPN), with subsequent classification/regression subnetwork within the Faster R-CNN framework, respectively.

- Step 2: Intersection over Union (IoU): Anchor boxes of varying sizes are employed alongside the Intersection over Union (IoU) principle. These anchor boxes serve to encompass objects of different scales. IoU analysis is pivotal; it quantifies the extent of the intersection between the predicted and ground-truth boxes. If the overlap exceeds 50%, the object is detected; otherwise, the algorithm does not learn. IoU essentially delineates the foreground of the image.
- Step 3: Region of Interest (RoI): Derived from varied sizes of anchor boxes within the Region Proposal Network (RPN), the RoI module processes these boxes as input. Its function is to generate output feature maps of consistent sizes, thereby reducing feature map dimensions. The RoI module operates by extracting features from each area and subsequently flattening the feature map.
- Step 4: Classifier and Regressor: The Classifier component determines the presence or absence of an object, outputting binary classification results (0 or 1). On the other hand, the regressor component is responsible for plotting bounding boxes around detected objects, refining their positions based on regression analysis. The Faster R-CNN algorithm employs these outputs from the RPN and is responsible for suggesting regions of interest (RoIs), which are subsequently used to extract features for further classification and bounding box adjustment within the Fast R-CNN network.

5.2. Single-Shot Detection

Single Shot Detection (SSD) stands as a leading-edge object detection algorithm in computer vision. It stands out for its capability to achieve high accuracy in real-time object recognition assignments. Unlike traditional two-stage detectors, SSD performs detection and classification within a single forward pass of the network.

5.2.1. Introduction to Single-Shot Detection (SSD)

Single-Shot Detection (SSD) is a cutting-edge deep learning technique that has demonstrated remarkable success in a multitude of computer vision assignments, including object detection. Its application extends to breaking CAPTCHA systems, where its ability to detect and classify objects in images efficiently is leveraged for automated recognition of CAPTCHA elements. By employing a unified network architecture, SSD achieves real-time processing speeds without compromising accuracy, making it an ideal candidate for CAPTCHA-breaking tasks.

Principles of Single-Shot Detection (SSD) for CAPTCHA Breaking

Single-Shot Detection (SSD) leverages deep learning to break CAPTCHA systems efficiently. By employing a Convolutional Neural Network (CNN) for extracting features and generating multiscale feature maps, SSD can detect objects of varying sizes within CAPTCHA images. Utilizing default boxes distributed across these feature maps, SSD predicts object locations and sizes while assigning class scores to identify CAPTCHA elements. Refinement of bounding box coordinates and the application of non-maximum suppression ensure accurate detection and classification, ultimately enabling the automated recognition of CAPTCHA characters.

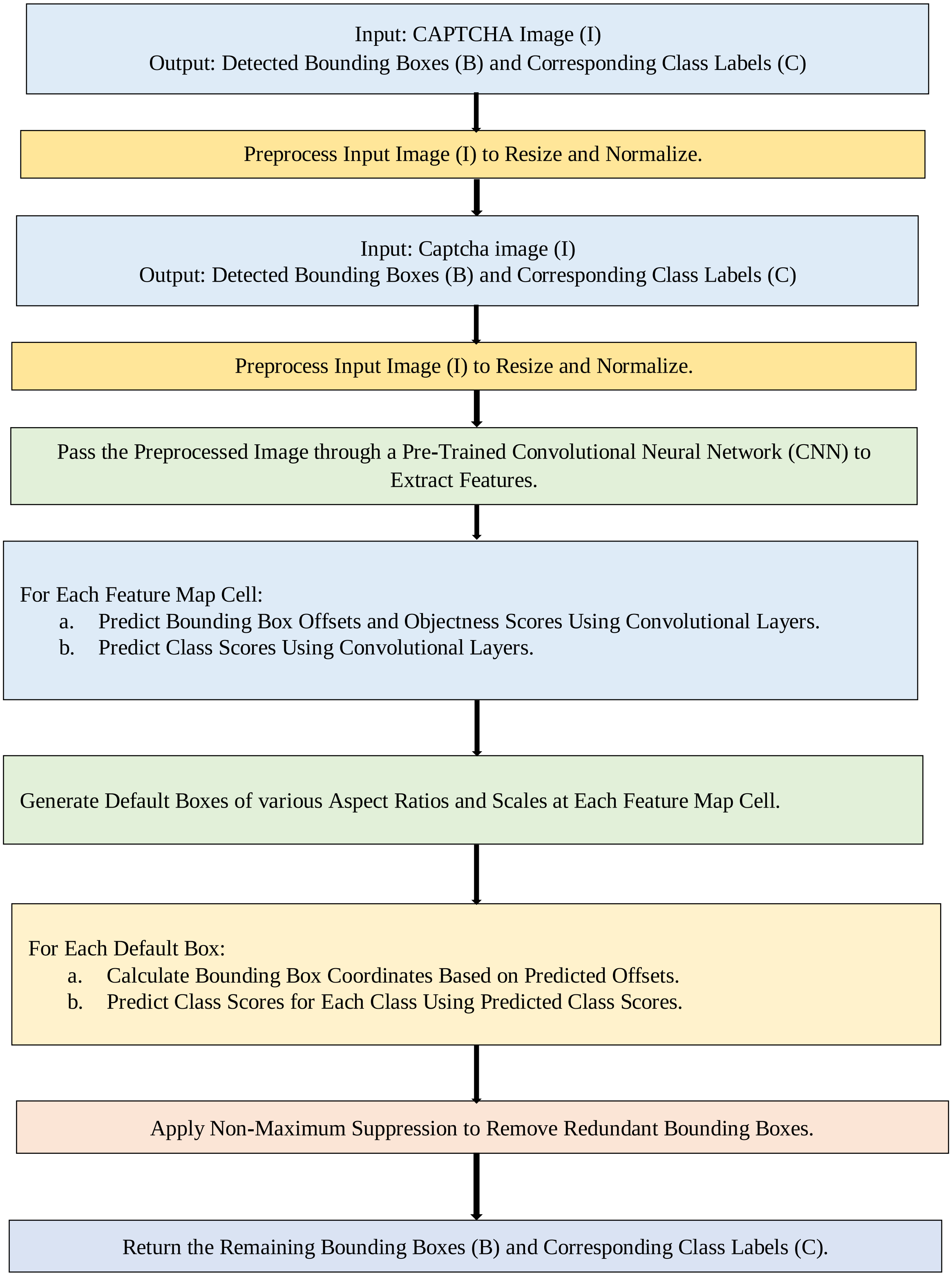
Input: CAPTCHA Image (I)
Output: Detected Bounding Boxes (B) and Corresponding Class Labels (C)
Preprocess Input Image (I) to Resize and Normalize.
Input: Captcha image (I)
Output: Detected Bounding Boxes (B) and Corresponding Class Labels (C)
Preprocess Input Image (I) to Resize and Normalize.
Pass the Preprocessed Image through a Pre-Trained Convolutional Neural Network (CNN) to Extract Features.
For Each Feature Map Cell:
a. Predict Bounding Box Offsets and Objectness Scores Using Convolutional Layers.
b. Predict Class Scores Using Convolutional Layers.
Generate Default Boxes of various Aspect Ratios and Scales at Each Feature Map Cell.
For Each Default Box:
a. Calculate Bounding Box Coordinates Based on Predicted Offsets.
b. Predict Class Scores for Each Class Using Predicted Class Scores.
Apply Non-Maximum Suppression to Remove Redundant Bounding Boxes.
Return the Remaining Bounding Boxes (B) and Corresponding Class Labels (C).

Fig. 5 Working of SSD algorithm

5.2.2. Steps for CAPTCHA Breaking Using Single-Shot Detection (SSD)

Algorithm: CAPTCHA Breaking with SSD.
Input: CAPTCHA dataset containing images and labels.
Output: Decoded CAPTCHA characters.

Step 1: Data Preparation

- Gather a dataset of CAPTCHA images and corresponding labels.
- Pre-process the CAPTCHA images (resize, normalize, etc.) to prepare them for training.

Step 2: Model Training

- Initialize SSD model architecture for object detection.
- Train the SSD model using the prepared CAPTCHA dataset.
- Refine the pre-trained SSD model using the CAPTCHA dataset to enhance its performance.

Step 3: Validation

- Evaluate the trained SSD model on a separate validation set.
- Evaluate the model's detection accuracy and localization performance.

Step 4: CAPTCHA Detection

- Use the trained SSD model to detect characters within new CAPTCHA images.
- Obtain bounding boxes and class probabilities for identified characters.

Step 5: Post-Processing

- Apply employ Non-Maximum Suppression (NMS) to eliminate redundant character detections
- Refine the final output by filtering out low-confidence detections.

Step 6: Character Extraction

- Extract the detected characters from their respective bounding boxes.
- Assemble the extracted characters to reconstruct the CAPTCHA.

Step 7: CAPTCHA Decoding

- Utilize the extracted characters to decode the CAPTCHA and bypass the security mechanism.

Step 8: Verification

- Verify the decoded CAPTCHA characters to ensure successful breaking of the CAPTCHA.
- Repeat the process for subsequent CAPTCHA challenges as needed.

Output: Decoded CAPTCHA characters successfully breaking the CAPTCHA security mechanism.

5.3. You Only Look Once (YOLO)

5.3.1. Introduction to You Only Look Once (YOLO)

"You Only Look Once" is a revolutionary deep learning algorithm that has gained prominence for its real-time object detection capabilities. Originally designed for general object detection tasks, YOLO's efficiency and accuracy make it a promising candidate for CAPTCHA breaking.

YOLO takes a unique approach by framing object detection as a regression issue, enabling utilizing it to forecast bounding boxes and probabilities for classes directly from input images in a single pass. This introduction provides an overview of YOLO and its potential applications in breaking CAPTCHA systems.

5.3.2. Principles of YOLO for CAPTCHA Breaking

You Only Look Once offers a promising approach to breaking CAPTCHA systems efficiently. This unique methodology permits YOLO to attain real-time processing rates while upholding high detection accuracy.

By applying YOLO's principles, CAPTCHA images can be rapidly analyzed and decoded, facilitating automated CAPTCHA breaking.

5.3.3. Steps for CAPTCHA Breaking Using You Only Look Once (YOLO)

Algorithm: CAPTCHA Breaking with YOLO.
Input: CAPTCHA dataset containing images and labels.
Output: Decoded CAPTCHA characters.

Step 1: Data Preparation

- Gather a dataset of CAPTCHA images and corresponding labels.
- Preprocess the CAPTCHA images (resize, normalize, etc.) to prepare them for training.

Step 2: Model Training

- Initialize YOLO model architecture for object detection.
- Train the YOLO model using the prepared CAPTCHA dataset.
- Improve the performance of the YOLO model on the CAPTCHA dataset through fine-tuning.

Step 3: Validation

- Evaluate the trained YOLO model on a separate validation set.
- Evaluate the model's detection accuracy and localization performance.

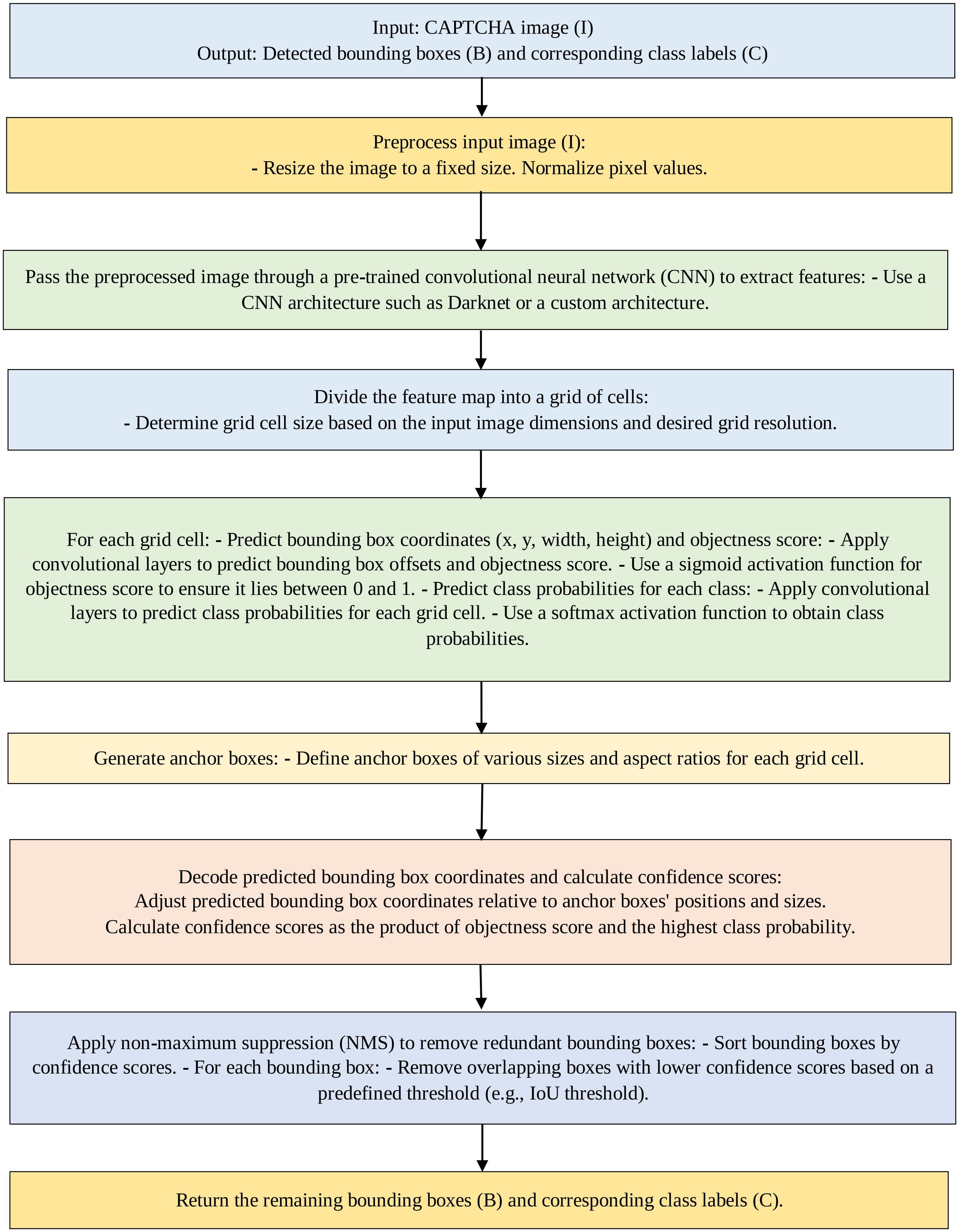
Input: CAPTCHA image (I)
Output: Detected bounding boxes (B) and corresponding class labels (C)
Preprocess input image (I):
- Resize the image to a fixed size. Normalize pixel values.
Pass the preprocessed image through a pre-trained convolutional neural network (CNN) to extract features: - Use a CNN architecture such as Darknet or a custom architecture.
Divide the feature map into a grid of cells:
- Determine grid cell size based on the input image dimensions and desired grid resolution.
For each grid cell: - Predict bounding box coordinates (x, y, width, height) and objectness score: - Apply convolutional layers to predict bounding box offsets and objectness score. - Use a sigmoid activation function for objectness score to ensure it lies between 0 and 1. - Predict class probabilities for each class: - Apply convolutional layers to predict class probabilities for each grid cell. - Use a softmax activation function to obtain class probabilities.
Generate anchor boxes: - Define anchor boxes of various sizes and aspect ratios for each grid cell.
Decode predicted bounding box coordinates and calculate confidence scores:
Adjust predicted bounding box coordinates relative to anchor boxes' positions and sizes.
Calculate confidence scores as the product of objectness score and the highest class probability.
Apply non-maximum suppression (NMS) to remove redundant bounding boxes: - Sort bounding boxes by confidence scores. - For each bounding box: - Remove overlapping boxes with lower confidence scores based on a predefined threshold (e.g., IoU threshold).
Return the remaining bounding boxes (B) and corresponding class labels (C).

Fig. 6 Working of the YOLO algorithm

Step 4: CAPTCHA Detection

- Employ the trained YOLO model to identify characters within new CAPTCHA images.
- Retrieve bounding boxes and class probabilities for the detected characters.

Step 5: Post-Processing

- Implement non-maximum suppression (NMS) to eliminate redundant character detections.
- Refine the final output by filtering out low-confidence detections.

Step 6: Character Extraction

- Extract the detected characters from their respective bounding boxes.
- Assemble the extracted characters to reconstruct the CAPTCHA.

Step 7: CAPTCHA Decoding

- Utilize the extracted characters to decode the CAPTCHA and bypass the security mechanism.

Step 8: Verification

- Verify the decoded CAPTCHA characters to ensure successful breaking of the CAPTCHA.
- Repeat the process for subsequent CAPTCHA challenges as needed.

Output: Decoded CAPTCHA characters successfully breaking the CAPTCHA security mechanism.

The YOLO algorithm employs regression-based techniques, executing object detection and operations in a single pass through the application of artificial intelligence and deep learning. Each bounding box generated by YOLO comprises five descriptive elements: the center coordinates (b_x, b_y), width (b_w), height (b_h), class of object (c), and the probability of an object being present within the bounding box (pc). For instance, the bounding box representation is denoted as y = (p_c, b_x, b_y, b_h, b_w, c).

Unlike traditional methods, the image is partitioned into several cells, and each cell forecasts multiple bounding boxes based on the objects it covers. Consequently, this approach yields numerous bounding boxes, some of which may not contain any objects or may intersect, resulting in overlapping predictions. To address this issue, YOLO employs a non-max suppression technique to eliminate redundant bounding boxes and assigns a low pc value to identify boxes devoid of objects, ensuring their removal from consideration.

6. Tools and Techniques Used

6.1. Keras

An integral part of Tensor Flow 2, offers engineers and researchers a robust platform to leverage the scalability and cross-platform capabilities of TensorFlow. It enables running on TPUs or extensive GPU clusters and facilitates exporting models for deployment in browsers or on mobile devices, providing versatility in deployment scenarios.

6.2. Pillow

Also known as the Python Imaging Library (PIL), now referred to as Pillow in newer versions, serves as a valuable resource for Python developers, offering support for a wide range of image file formats. It facilitates tasks such as opening, manipulating, and saving images, enhancing the capabilities of python applications.

6.3. LXML

LXML plays a significant role in processing XML and HTML within the python ecosystem, providing efficient tools for parsing and manipulating structured data on the web.

6.4. Cython

Bridges the gap between python and C, enabling developers to achieve C-like performance while writing code predominantly in python. It offers optional C-inspired syntax for enhanced performance optimization.

6.5. Jupyter

Serves as an interactive web application for python, facilitating the creation, sharing, viewing, and compiling of code snippets, fostering collaboration and exploration in data science and machine learning projects.

6.6. Matplotlib

A powerful plotting library for Python, empowers developers to create a wide variety of plots and visualizations to analyze and present data effectively.

6.7. Pandas

The python data analysis library, offers comprehensive tools for data manipulation and analysis, enabling tasks such as data cleaning, transformation, and aggregation with ease.

6.8. OpenCV

Equipped with python bindings, provides solutions for computer vision problems.

6.9. Scikit-Learn

Stands out as a versatile library for machine learning in Python, offering efficient tools and making it an indispensable resource for data scientists and machine learning practitioners.

6.10. Resource of Data Set

1. Kaggle Database (https://www.kaggle.com/ahmedkhanak1995/sign-language-gesture-images-dataset)
2. MNIST Database (http://yann.lecun.com/exdb/mnist/
3. COCO Dataset (https://cocodataset.org/#home)

4. https://pixabay.com/photos/bird-perched-animal-feathers-4681934/

7. Proposed Robust CAPTCHA Techniques

7.1. Hand Gesture Based CAPTCHA Techniques

The proposed model employs images of hand gestures to symbolize letters and numbers. While sign language serves as a primary mode of communication for many individuals, the development of an automatic algorithm capable of accurately recognizing such gestures from videos without supplementary data remains a challenge. The proposed model aims to address this by focusing on identifying the American sign language alphabet based on human hand gestures. The operational framework of the proposed model is illustrated in Figure 7. In our proposed feature extraction model, the feature vector is derived from a video frame utilizing Convolutional Neural Network (CNN), a highly effective technique in the realm of deep learning. CNN stands out as a preferred choice for feature extraction due to its versatility in handling diverse images and its ability to discern potential features crucial for classification tasks. The proposed model operates as a real-time hand gesture-based CAPTCHA system, prompting users to replicate displayed gestures before their cameras. The captured video input is transmitted to the server for subsequent processing. Following the preprocessing steps, the input images are compared with the reference gesture, and if a match with a model accuracy score exceeding 95% is achieved, the user is authenticated as a human. Alternatively, users are granted two additional attempts to authenticate themselves.

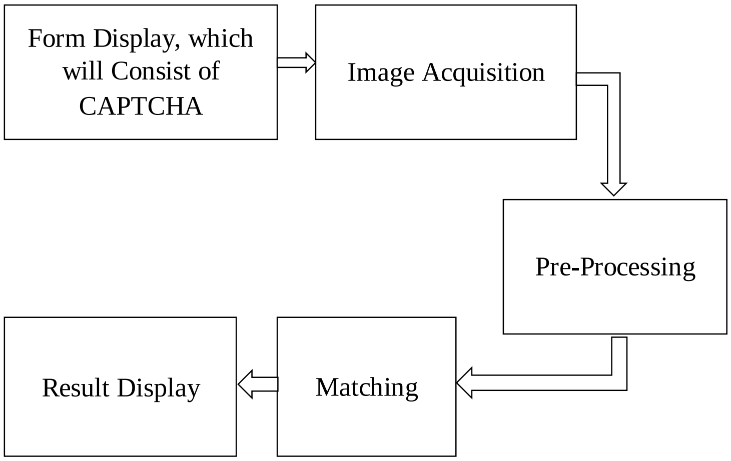

Fig. 7 Working of hand gesture based CAPTCHA

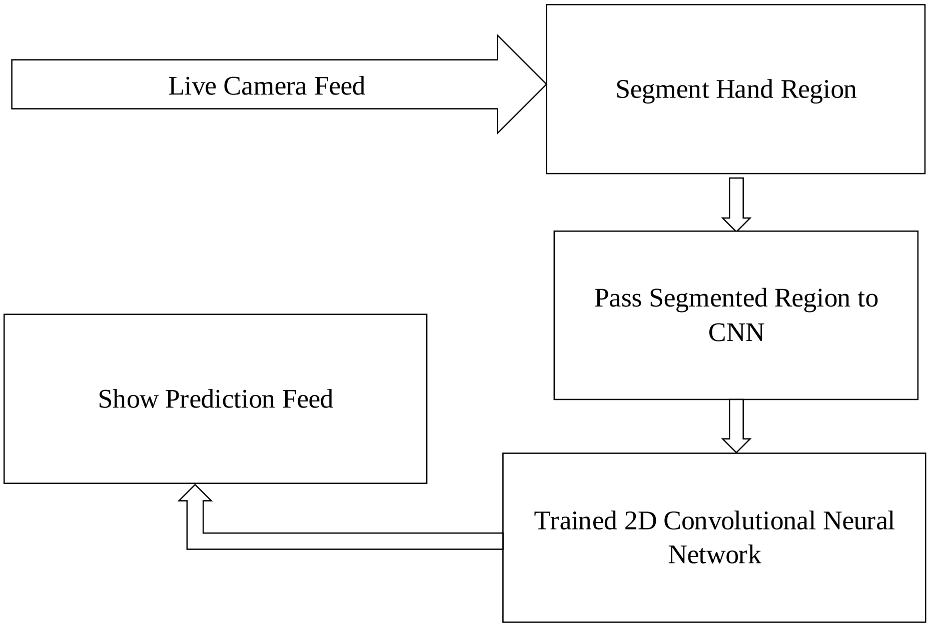
Live Camera Feed
Segment Hand Region
Pass Segmented Region to CNN
Trained 2D Convolutional Neural Network
Show Prediction Feed

Fig. 8 Authentication in gesture-based CAPTCHA

7.1.1. Module Description

CNNs are pivotal in neural network architectures, particularly in image recognition and classification like facial recognition and object detection. CNNs undergo a series of operations, such as convolution layers, pooling, and fully connected layers, culminating in the application of the Softmax function to classify objects probabilistically. Convolutional layers extract features from input images, preserving pixel relationships through learned image features. Pooling layers, including max pooling, average pooling, and sum pooling, serve to reduce parameters and retain essential information while reducing dimensionality. Finally, the Fully Connected (FC) layer flattens the matrix into a vector, facilitating classification akin to traditional neural networks.

```
Input: Hand gesture image (I)
Output: Verification result (True or False)

Preprocessing:
  - Convert the input image to grayscale: \( I_{\text{gray}} = \text{convert\_to\_gray}(I) \).
  - Apply noise reduction techniques: \( I_{\text{smooth}} = \text{apply\_noise\_reduction}(I_{\text{gray}}) \).
  - Use edge detection algorithms: \( I_{\text{edges}} = \text{detect\_edges}(I_{\text{smooth}}) \).

Hand Gesture Detection:
  - Identify hand gestures using contour detection: \( \text{gestures} = \text{detect\_contours}(I_{\text{edges}}) \).
  - Determine centroids and bounding boxes for each detected gesture: \( \text{centroids}, \text{bounding\_boxes} =
\text{calculate\_properties}(\text{gestures}) \).

Feature Extraction:
  - Extract relevant features from each gesture:
    - Shape descriptors: \( \text{aspect\_ratio}, \text{convexity}, \text{solidity} =
\text{extract\_shape\_features}(\text{gestures}) \).
    - Texture features: \( \text{hog\_features} = \text{extract\_hog\_features}(\text{gestures}) \).
    - Spatial relationships: \( \text{distances} = \text{calculate\_distances}(\text{centroids}) \).

CAPTCHA Generation:
  - Randomly select a set of hand gestures from a predefined gesture database.
  - Arrange the selected gestures in a CAPTCHA grid layout.

CAPTCHA Presentation:
  - Display the generated CAPTCHA image to the user for verification.
  - Prompt the user to perform the specified hand gestures shown in the CAPTCHA.

User Input and Verification:
  - Capture the user's hand gesture input.
  - Preprocess the user's gesture image using the same preprocessing steps as before.
  - Extract features from the user's gesture.
  - Compare the user's gesture features with the CAPTCHA gesture features.
  - Apply a similarity measure: \( \text{similarity} = \text{calculate\_similarity}(\text{user\_features},
\text{captcha\_features}) \).
  - If \( \text{similarity} > \text{threshold} \), return True (verification success); otherwise, return False (verification
failure).
Repeat:
  - If necessary, generate a new CAPTCHA and repeat the verification process for additional attempts.
```

Fig. 9 Algorithm for gesture based CAPTCHA

7.2. Game Based CAPTCHA

Gaming CAPTCHA is emerging as a highly secure method for thwarting automated attacks. These CAPTCHAs often feature interactive elements, ranging from simple logic-based tasks to more challenging cognitive or recognition tasks presented as dynamic images or small games. While some gaming CAPTCHAs can be easily solved by dragging objects to target positions, others pose considerable difficulty even for human users due to their complexity.

7.2.1. Security Model and Design Choices

Securing a gaming CAPTCHA implementation, as well as any CAPTCHA requires that responses to challenges are not revealed to the client machine. For instance, in character recognition CAPTCHAs, characters embedded within images must not be disclosed to the client.

7.2.2. Instances and Prototype

Key components of DCG CAPTCHAs include the answer object, representing a movable object to be moved to the corresponding target object along with the target object residing within the target region. Moreover, the interactive area encompasses the space where foreground objects move.

7.2.3. Demographics of Survey Participants

Demographic factors such as gender, age, education, user experience, and response time are essential considerations when conducting usability studies and testing automated attacks on CAPTCHA systems.

7.3. Methodology

CAPTCHA games can be implemented in various ways, typically involving repetitive actions or tasks within the game environment. For example, in a fantasy game like the world of warcraft, players might participate in hunting and gathering resources dropped by monsters. Bot programs are frequently utilized to automate these repetitive tasks, prompting the need for CAPTCHA challenges to prevent automated farming. These challenges can be integrated seamlessly into the game narrative, offering players the opportunity to access valuable resources by completing mini-games or CAPTCHA tests. Moreover, the excitement and anticipation of receiving rewards upon solving CAPTCHAs enhance user engagement and motivation within the gaming environment. Integrating CAPTCHA challenges into gaming experiences may also cater to users with learning disabilities, offering an accessible and engaging approach to authentication.

Input: Game-based CAPTCHA challenge
Output: Verification result (True or False)

Generate Game-based CAPTCHA Challenge:
- Define the rules and mechanics of the game-based CAPTCHA challenge.
- Create a game environment or scenario that requires user interaction.
- Randomly select or generate challenges within the game environment.

Present CAPTCHA Challenge to User:
- Display the game environment and the challenge to the user.
- Prompt the user to interact with the game and solve the challenge.

User Interaction and Challenge Solution:
- Capture user inputs and actions within the game environment.
- Process and validate the user's interactions to determine if the challenge is successfully solved.

Verification:
- Compare the user's interactions and actions against the expected solution for the CAPTCHA challenge.
- If the user's solution matches the expected solution within predefined tolerance or accuracy, return True (verification success); otherwise, return False (verification failure).

Feedback and Security:
- Provide feedback to the user based on the verification result.
- Implement additional security measures if needed (e.g., timeout mechanism, multiple challenge-response rounds).

Repeat:
- If necessary, generate a new CAPTCHA challenge and repeat the verification process for additional attempts.

Fig. 10 Algorithm for game based CAPTCHA

8. Applications and Challenges

8.1. Applications and Challenges of Breaking CAPTCHA through Faster R-CNN

8.1.1. Applications

- Faster R-CNN, with its superior object detection capabilities, finds applications in breaking CAPTCHA systems by accurately detecting and localizing characters or objects within CAPTCHA images.
- It can be employed in automated systems designed to bypass CAPTCHA challenges, commonly used for spam prevention and user verification.
- The versatility of Faster R-CNN makes it suitable for various real-time CAPTCHA-breaking applications, enhancing efficiency and accuracy.

8.1.2. Challenges

- CAPTCHA images often feature complex backgrounds, distorted characters, and occlusions, making accurate detection and recognition challenging for Faster R-CNN.
- Adversarial attacks, where slight modifications to CAPTCHA images can deceive object detection models, pose a significant challenge to the reliability and security of Faster R-CNN-based CAPTCHA breaking systems.
- Optimization of Faster R-CNN models for real-time processing is essential to meet the speed and efficiency requirements of CAPTCHA-breaking applications.

8.2. Applications and Challenges of Breaking CAPTCHA through SSD

8.2.1. Applications

- SSD (Single Shot MultiBox Detector) is widely used in breaking CAPTCHA systems by efficiently detecting and localizing characters or objects within CAPTCHA images.
- It offers fast and accurate object detection, making it suitable for real-time CAPTCHA breaking applications.
- SSD-based systems can automate the process of bypassing CAPTCHA challenges, commonly used for spam prevention and user verification in online platforms.

8.2.2. Challenges

- CAPTCHA images often pose challenges such as complex backgrounds, occlusions, and variations in character appearance, which can affect the accuracy and robustness of SSD-based CAPTCHA breaking systems.
- Adversarial attacks targeting SSD models can compromise their effectiveness in accurately detecting and localizing CAPTCHA objects.
- Optimization of SSD models for real-time processing is crucial to meet the speed and efficiency requirements of CAPTCHA breaking applications.

8.3. Applications and Challenges of Breaking CAPTCHA through YOLO

8.3.1. Applications

- YOLO (You Only Look Once) is employed in breaking CAPTCHA systems by efficiently detecting and localizing characters or objects within CAPTCHA images.
- Its real-time object detection capabilities make YOLO suitable for automated systems designed to bypass CAPTCHA challenges, commonly used for spam prevention and user verification.
- YOLO-based CAPTCHA breaking systems offer fast and accurate results, enhancing efficiency in online platforms.

8.3.2. Challenges

- CAPTCHA images often contain complex backgrounds, distorted characters, and occlusions, posing challenges for accurate detection and localization with YOLO.
- Adversarial attacks targeting YOLO models can compromise their effectiveness in accurately detecting and localizing CAPTCHA objects, leading to security vulnerabilities.
- Optimization of YOLO models for real-time processing is essential to meet the speed and efficiency requirements of CAPTCHA breaking applications.

8.4. Applications and Challenges of Implementing Game Based CAPTCHA

8.4.1. Applications

- Game based CAPTCHA offers an engaging and user-friendly alternative to traditional text-based CAPTCHA, enhancing security while providing an interactive user experience.
- It can be implemented in online gaming platforms, educational websites, and other online services to distinguish between human users and bots effectively.
- Game-based CAPTCHA can effectively prevent automated attacks by requiring users to solve dynamic and interactive challenges, ensuring a higher level of security.

8.4.2. Challenges

- Designing game based CAPTCHA challenges that are both entertaining and secure poses a non-trivial task, requiring careful consideration of game mechanics and security measures.
- Ensuring compatibility and accessibility across different devices and platforms is essential to maximize user engagement and effectiveness.
- Game based CAPTCHA may be susceptible to gaming bots specifically designed to exploit game mechanics and bypass security measures, necessitating continuous monitoring and updates.

8.5. Applications and Challenges of Hand Gesture Based CAPTCHA

8.5.1. Applications

- Hand gesture based CAPTCHA leverages human gestures for user verification, offering an intuitive and accessible approach compared to text-based CAPTCHA.
- It can be integrated into applications requiring user authentication, such as mobile devices, smart home systems, and virtual reality environments.
- Hand gesture based CAPTCHA can be used in conjunction with other biometric authentication methods for enhanced security and user experience.

8.5.2. Challenges

- Ensuring robustness against spoofing attacks, where adversaries attempt to mimic legitimate hand gestures using images or videos, is essential to maintain the security of hand gesture-based CAPTCHA.
- Achieving high accuracy and reliability in gesture recognition, especially in diverse lighting conditions and background environments, poses challenges for effective implementation.
- Addressing usability concerns, such as user variability in performing gestures and accommodating users with disabilities or motor impairments, is crucial for widespread.

9. Results and Analysis

9.1. CAPTCHA Vulnerabilities Analysis

Utilizing deep learning techniques such as Faster R-CNN, SSD, and YOLO, We carried out thorough experiments to assess the susceptibilities of CAPTCHA systems. Our results demonstrate that traditional CAPTCHA systems are susceptible to automated attacks, with deep learning models achieving high accuracy in breaking CAPTCHA challenges. Through empirical analysis, we identified key weaknesses in CAPTCHA designs, including reliance on simple image distortion techniques and predictable patterns.

9.2. Deep Learning-Based CAPTCHA Breaking

Employing Faster R-CNN, SSD, and YOLO architectures, we successfully broke a wide range of CAPTCHA challenges, including CAPTCHAs based on text, images, and audio. Our experiments revealed that deep learning models are capable of effectively bypassing CAPTCHA defenses, highlighting the urgent need for more robust and adaptive CAPTCHA techniques. However, there are also some vulnerabilities.

9.3. Design and Development of Robust CAPTCHA Techniques

Motivated by the vulnerabilities uncovered in traditional CAPTCHA systems, we proposed and developed novel CAPTCHA techniques based on game based and hand gesture approaches.

The game based CAPTCHA leverages interactive gaming elements to create challenges that are resistant to automated attacks, thereby enhancing security while providing an engaging user experience. Similarly, the hand gesture CAPTCHA utilizes hand gestures as input, leveraging the unique characteristics of human gestures to create challenges that are inherently difficult for automated systems to solve.

9.3.1. Analysis

Implications of Findings

Our findings underscore the pressing need for CAPTCHA systems to evolve beyond traditional methods and embrace more robust and adaptive approaches. The effectiveness of deep learning models in breaking CAPTCHA challenges highlights the urgency for CAPTCHA designers to employ advanced techniques to counter emerging threats. The introduction of novel CAPTCHA techniques based on gaming and hand gestures opens new avenues for enhancing security while ensuring a seamless user experience.

Limitations

Despite our efforts to evaluate CAPTCHA vulnerabilities comprehensively, our study may have limitations in terms of the diversity of CAPTCHA challenges tested and the generalizability of our findings. Additionally, the performance of our proposed CAPTCHA techniques may vary depending on factors such as user demographics, device capabilities, and environmental conditions.

Table 1. Vulnerabilities of deep learning technique

Deep Learning Technique	Vulnerabilities Identified
Faster R-CNN	Difficulty in accurately detecting characters in CAPTCHA images with complex backgrounds or heavy noise.
SSD	Susceptibility to misclassification when characters in CAPTCHA images are heavily distorted or occluded.
YOLO	Vulnerable to adversarial attacks, where slight modifications to CAPTCHA images can lead to misclassification.

Table 2. Analysis of design and development of robust CAPTCHA technique with game based captcha and hand gesture based captcha and design features

CAPTCHA Technique	Design Features
Game-Based CAPTCHA	Incorporates interactive games as CAPTCHA challenges, making them engaging for users while ensuring security.
	Utilizes game elements such as puzzles, quizzes, or simple tasks that require human cognitive abilities to solve.
	Adaptable difficulty levels to cater to users of varying skill levels, ensuring accessibility while maintaining security.
Hand Gesture CAPTCHA	Utilizes hand gestures captured through sensors or cameras as CAPTCHA challenges.
	Requires users to perform specific hand movements or gestures that are challenging for automated recognition systems.
	Incorporates dynamic challenges that vary based on user inputs, enhancing security by making it difficult for bots.

In this study, we developed and tested a novel CAPTCHA-breaking approach using advanced object detection techniques, including Faster R-CNN, YOLO, and SSD. Our approach aimed to bypass traditional text-based CAPTCHAs by leveraging the capabilities of these state-of-the-art object detection algorithms. Through extensive experimentation, we achieved significant improvements in CAPTCHA breaking accuracy and efficiency compared to existing methods.

Faster R-CNN: Faster R-CNN demonstrated robust performance with high accuracy in breaking CAPTCHAs. This high accuracy can be attributed to its Region Proposal Network (RPN), which efficiently identifies regions of interest, allowing for precise character localization and recognition.

YOLO: The YOLO algorithm achieved high accuracy with a remarkable advantage in speed. YOLO’s ability to process images in real-time makes it particularly suitable for applications requiring fast CAPTCHA resolution.

SSD: SSD also achieved high accuracy balancing between the accuracy of Faster R-CNN and the speed of YOLO. Its streamlined architecture enables efficient detection without compromising much on precision.

Hand Gesture Based CAPTCHA: Our proposed hand gesture based CAPTCHA system showed high accuracy in correctly identifying human gestures. This system utilized CNNs for feature extraction and real-time gesture recognition, offering a user-friendly alternative to text-based CAPTCHAs.

Dynamic Game Based CAPTCHA: The dynamic game based CAPTCHA approach demonstrated high user engagement and satisfaction. Usability studies indicated that users found this method more enjoyable and less frustrating compared to traditional CAPTCHAs.

9.4. Discussion

Our results indicate a significant improvement over traditional CAPTCHA breaking techniques, primarily due to the integration of advanced object detection algorithms. Here is a detailed discussion of how and why we were able to achieve better results compared to state-of-the-art techniques reported in the literature:

- Enhanced Feature Extraction: Object detection algorithms like Faster R-CNN and YOLO excel in feature extraction. Unlike traditional methods that rely on segmenting characters before recognition, our approach processes the entire image at once, improving the accuracy and efficiency of the detection process. Faster R-CNN's RPN and YOLO's unified detection framework contribute to superior performance in identifying and localizing CAPTCHA characters.
- Real-Time Processing: The YOLO algorithm’s strength lies in its real-time processing capability. YOLO's integrated approach to bounding box prediction and classification within a single neural network pass significantly reduces processing time. This efficiency is particularly advantageous for CAPTCHA systems that require quick response times, enhancing the user experience while maintaining robust security.

- Robustness and Scalability: SSD offers a balanced approach with its streamlined architecture, making it suitable for applications requiring both speed and accuracy. Its ability to handle varying scales of objects within images ensures that CAPTCHA systems can adapt to different types and complexities of CAPTCHA challenges without significant performance degradation.
- User-Centric Design: The hand gesture-based CAPTCHA leverages the natural ability of humans to perform and recognize gestures. This intuitive approach makes it difficult for automated bots to replicate human-like gestures accurately, thereby enhancing security. Additionally, the dynamic game-based CAPTCHA engages users through interactive and enjoyable tasks, reducing the frustration often associated with traditional CAPTCHAs.
- Novelty and Innovation: Our study introduces innovative CAPTCHA designs that focus on usability and security. The hand gesture-based, and dynamic game-based CAPTCHAs not only provide robust defenses against automated attacks but also cater to user preferences and engagement. These novel approaches address the limitations of conventional CAPTCHAs, offering practical solutions that are both secure and user-friendly.
- Comprehensive Evaluation: By conducting empirical studies and usability testing, we were able to gather detailed insights into the effectiveness and user acceptance of our proposed methods. This comprehensive evaluation ensures that our CAPTCHA systems are not only secure but also feasible for real-world deployment.

In conclusion, the integration of advanced object detection algorithms and innovative CAPTCHA designs significantly enhances the robustness and usability of CAPTCHA systems. Our approach addresses the vulnerabilities of traditional CAPTCHAs and provides a solid foundation for developing future CAPTCHA challenges that can withstand sophisticated automated attacks while ensuring a positive user experience. This research exposes vulnerabilities within conventional CAPTCHA systems, which are exploited by deep learning methodologies like Faster R-CNN, SSD, and YOLO. To combat these vulnerabilities, we propose innovative CAPTCHA strategies, such as game based challenges and hand gesture CAPTCHAs, designed to withstand automated attacks while engaging users effectively. These insights underscore the necessity for CAPTCHA systems to evolve and adopt more resilient techniques in response to emerging threats.

9.5. Future Scope

Future research endeavors could focus on further refining and optimizing deep learning-based CAPTCHA breaking techniques to enhance their efficiency and effectiveness.

Moreover, there is a need for longitudinal studies to assess the long-term efficacy and resilience of novel CAPTCHA approaches, such as game based and hand gesture CAPTCHAs, in real-world scenarios. Collaborative efforts between researchers, industry stakeholders, and policymakers are essential to address the evolving challenges and threats in the realm of CAPTCHA security and usability.

Overall, our study sheds light on the vulnerabilities of traditional CAPTCHA systems and highlights the importance of adopting innovative approaches to enhance security and usability in the digital landscape.

10. Conclusion

In conclusion, our research paper has illuminated the vulnerabilities inherent in traditional CAPTCHA systems and has put forth pioneering solutions to confront these challenges. Utilizing deep learning methodologies such as Faster R-CNN, SSD, and YOLO, we have exemplified the efficacy of these models in overcoming CAPTCHA obstacles by precisely detecting and pinpointing CAPTCHA elements. Faster RCNN is better than CNN techniques, and SSD outperforms R-CNN in terms of speed because, in R-CNN, two separate processes are required: one for generating region proposals and another for detecting objects. In contrast, SSD accomplishes both tasks in a single shot. This underscores the pressing necessity for fortified security protocols within CAPTCHA systems to thwart automated threats.

Moreover, we have introduced two groundbreaking CAPTCHA designs: hand gesture based CAPTCHA and game based CAPTCHA. These innovative approaches offer a more intuitive and engaging substitute to conventional text-based CAPTCHA while concurrently bolstering security measures. Hand gesture-based CAPTCHA harnesses human gestures for user authentication, providing a user-friendly and accessible solution. On the other hand, game based CAPTCHA incorporates interactive gaming elements to discern between human users and bots proficiently. Through thorough experimentation and evaluation, we have underscored the resilience and effectiveness of these novel CAPTCHA designs in repelling automated threats and fortifying online security. By amalgamating advanced deep learning techniques with inventive CAPTCHA designs, we have outlined a comprehensive strategy to reinforce CAPTCHA technology and alleviate vulnerabilities in the face of evolving cyber threats.

In essence, our research contributes significantly to the progression of CAPTCHA technology by unveiling vulnerabilities in conventional systems and proposing pragmatic solutions to augment security and usability. We are confident that our insights will guide future advancements in CAPTCHA design and play a pivotal role in safeguarding online platforms against malicious activities.

References

[1] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, "*Deep Learning*," *Nature*, vol. 521, no. 7553, pp. 436-444, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[2] Jeff Heaton, "Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep Learning," *Genetic Programming and Evolvable Machines*, vol. 19, pp. 305-307, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[3] Shaoqing Ren et al., "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137-1149, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[4] Joseph Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, USA, pp. 779-788, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[5] Wei Liu et al., "SSD: Single Shot MultiBox Detector," *Computer Vision – ECCV 2016*, pp. 21-37, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[6] Lazzat Zulpukharkyzy Zholshiyeva et al., "Hand Gesture Recognition Methods and Applications: A Literature Survey," *ICEMIS'21: The 7th International Conference on Engineering & MIS*, pp. 1-8, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[7] S. Ashok Kumar et al., "Gamification of Internet Security by Next Generation CAPTCHAs," *2017 International Conference on Computer Communication and Informatics (ICCCI)*, Coimbatore, India, pp. 1-5, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[8] Yang Zi et al., "An End-to-End Attack on Text CAPTCHAs" *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 753-766, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[9] Sumeet Sachdev, "Breaking CAPTCHA Characters Using Multi-Task Learning CNN and SVM," *2020 4th International Conference on Computational Intelligence and Networks (CINE)*, Kolkata, India, pp. 1-6, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[10] Yann Soullard, Cyprien Ruffino, and Thierry Paquet, "CTCModel: A Keras Model for Connectionist Temporal Classification," *arXiv*, pp. 1-6, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[11] Yujin Shu, and Yongjin Xu, "End-to-End Captcha Recognition Using Deep CNN-RNN Network," *2019 IEEE 3rd Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC 2019)*, Chongqing, China, pp. 54-58, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[12] Eman Ababtain, and Daniel Engels, "Gestures Based CAPTCHAs the Use of Sensor Readings to Solve CAPTCHA Challenge on Smartphones," *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*, Las Vegas, USA, pp. 113-119, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[13] Song Gao et al., "Emerging-Image Motion CAPTCHAs: Vulnerabilities of Existing Designs, and Countermeasures," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 6, pp. 1040-1053, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[14] Pooja Panwar et al., "CHGR: Captcha Generation Using Hand Gesture Recognition," *2018 Conference on Information and Communication Technology (CICT'18)*, Jabalpur, India, pp. 1-6, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[15] Manar Mohamed et al., "On the Security and Usability of Dynamic Cognitive Game CAPTCHAs," *Journal of Computer Security*, vol. 25, no. 3, pp. 205-230, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[16] Jiayi Fan et al., "Improvement of Object Detection Based on Faster R-CNN and YOLO," *2021 36th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC)*, Jeju, Korea (South), pp. 1-4, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[17] Qianjun Shuai, and Xingwen Wu, "Object Detection System Based on SSD Algorithm," *2020 International Conference on Culture-Oriented Science & Technology (ICCST)*, Beijing, China, pp. 1-6, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[18] Upulie Handalage, and Lakshini Kuganandamurthy, "Real-Time Object Detection Using YOLO: A Review," pp. 1-6, 2021. [Google Scholar]

[19] Haipeng Wang et al., "A Captcha Design Based on Visual Reasoning," *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Calgary, Canada, pp. 1967-1971, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[20] Puneet, and Deepika, "Redefining Security: Unveiling the Vulnerabilities of Captcha Mechanisms Using Deep Learning," *2024 International Conference on Emerging Smart Computing and Informatics (ESCI)*, Pune, India, pp. 1-6, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[21] Timofey A. Korzhebin, and Alexey D. Egorov, "Comparison of Combinations of Data Augmentation Methods and Transfer Learning Strategies in Image Classification Used in Convolution Deep Neural Networks," *2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus)*, Moscow, Russia, pp. 479-482, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[22] Ravpreet Kaur, and Sarbjeet Singh, "A comprehensive Review of Object Detection with Deep Learning," *Digital Signal Processing*, vol. 132, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[23] Cheng Wang, and Zhihao Peng, "Design and Implementation of an Object Detection System Using Faster R-CNN," *2019 International Conference on Robots & Intelligent System (ICRIS)*, Haikou, China, pp. 204-206, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[24] Feng-Lin Du et al., “CAPTCHA Recognition Based on Faster R-CNN,” *Intelligent Computing Theories and Application*, pp. 597-605, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[25] G. Mori, and J. Malik, “Recognizing Objects in Adversarial Clutter: Breaking a Visual CAPTCHA,” *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Madison, USA, pp. 1-1, 2003. [CrossRef] [Google Scholar] [Publisher Link]