**ATTACK ON IMAGE CAPTCHA USING**

**DEEP LEARNING AND PYTHON**

***Report submitted to***

***Haldia Institute of Technology, Haldia***

***For the award of the degree***

***of***

**Bachelor of Technology in Computer Science and Engineering**

***by***

**Rushalee Das (19/CSE/120)**

**Sagar Bhusan (19/CSE/121)**

**Akash Barua (19/CSE/117)**

**Annpurna Singh (19/CSE/118)**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**HALDIA INSTITUTE OF TECHNOLOGY, HALDIA**

# DECLARATION

The work contained in this report is original and has been done by me/us under the guidance of our supervisor(s).

The work has not been submitted to any other Institute for any degree.

We have followed the guidelines provided by the Institute in preparing the report.

We have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

Whenever we have used materials (data, theoretical analysis, figures, and text) from other sources, we have given due credit to them by citing them in the text of the report and giving their details in the references.

Signature of the Students

# CERTIFICATE

This is to certify that the Dissertation Report entitled, **“**Attack Simulation on Image Captcha**”** submitted by **“Akash Barua, Annpurna Singh, Rushalee Das, and Sagar Bhusan”** to Haldia Institute of Technology, Haldia, India, is a record of bonafide Project work carried out by them under my supervision and guidance and is worthy of consideration for the award of the degree of Bachelor of Technology in Computer Science and Engineering of the Institute.

Project Mentor H.O.D

# ACKNOWLEDGEMENT

We express our gratitude to **Mr. Palash Ray**, our project mentor for guidance in completing our project on the topic of **Attack simulation on Image Captcha**. It was a great learning experience.

We would like to express our gratitude and appreciation to all those who supported us during the making and completion of this report. First, we wish to express our sincere gratitude to our mentor, **Mr. Palash Ray**, for his enthusiasm, patience, insightful comments, helpful information, practical advice, and unceasing ideas that have helped us tremendously at all times in our research and writing of this report. His immense knowledge, profound experience, and professional expertise in Wireless Sensor Network Security have enabled us to complete this research successfully. Without his support and guidance, this project would not have been possible.

We also impart our sincere gratitude to our HOD, **Dr. Subhankar Joarder,** and all the faculty members of the **Computer Science and Engineering Department**, **Haldia Institute of Technology** for their support and guidance throughout the making of this project.

I would like to take this opportunity to express my gratitude to all of my group members **Rushalee Das, Sagar Bhusan , Akash Barua and Annpurna Singh**. The project would not have been successful without their cooperation and input.

# ABSTRACT

The objective of this project is to develop a deep learning-based approach to break image recognition CAPTCHAs, which are widely used to distinguish between humans and bots. Traditional CAPTCHAs are designed to be challenging for automated algorithms to solve, but recent advancements in deep learning algorithms have raised concerns about their effectiveness.

In this project, we propose a convolutional neural network (CNN) architecture that can effectively analyze and interpret the visual content of CAPTCHAs. The CNN model is trained on a large dataset of labeled CAPTCHA images to learn the patterns and features necessary to accurately classify the images. By leveraging the power of deep learning, the model aims to bypass the security measures implemented by image recognition CAPTCHAs.

The project involves various stages, including data collection, preprocessing, model development, training, and evaluation. The collected CAPTCHA images are preprocessed to remove noise and enhance their clarity. The CNN model is then constructed with multiple convolutional and pooling layers to extract relevant features from the images. The model is trained using a combination of supervised learning techniques and optimization algorithms to minimize the classification error.

The performance of the trained model is evaluated using a separate test set of CAPTCHA images. The evaluation metrics include accuracy, precision, recall, and F1 score.

The results are compared with existing state-of-the-art CAPTCHA-breaking techniques to assess the effectiveness of the proposed deep learning approach.

The outcomes of this project have implications for both the security of image recognition CAPTCHAs and the development of more robust security measures. By identifying the vulnerabilities in current CAPTCHA systems, it becomes possible to enhance their design and improve their resistance against automated attacks. Additionally, the project contributes to the advancement of deep learning techniques in the field of computer vision and pattern recognition.

This project aims to utilize deep learning algorithms to break image recognition CAPTCHAs, with the goal of enhancing their security and inspiring further research in the field of CAPTCHA development and countermeasures.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **CHAPTERS** | **PAGE NO** |
| 1. | CHAPTER 1 : INTRODUCTION | 7 |
| 2. | CHAPTER 2 : LITERATURE SURVEY | 9 |
| 3. | CHAPTER 3 : OVERVIEW | 11 |
| 4. | CHAPTER 4 : METHODOLOGY | 15 |
| 5. | CHAPTER 5 : ENVIRONMENT | 21 |
| 6. | CHAPTER 6 : IMPLEMENTATION | 22 |
| 7. | CHAPTER 7 : RESULT | 23 |
| 8. | CHAPTER VIII : REFERENCES | 24 |

**CHAPTER 1**

# INTRODUCTION

A CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) is a security mechanism used in computing to differentiate between human users and automated programs, often referred to as bots or robots. It is designed to protect websites and online services from malicious activities such as spamming, data scraping, and unauthorized access.

The primary goal of a CAPTCHA is to create a challenge that is easy for humans to solve but difficult for automated programs to pass. It accomplishes this by presenting users with a test or task that requires cognitive abilities typically possessed by humans, such as image recognition or logical reasoning. By successfully completing the challenge, users prove their human identity and gain access to the desired content or service.

One of the most common types of CAPTCHAs involves displaying distorted or obscured characters, such as letters and numbers, and asking users to type in the correct sequence. The distorted characters make it challenging for automated programs to accurately recognize and decipher the text, while humans can usually identify and input the characters correctly. This approach, known as the "text-based CAPTCHA," has been widely used across the internet.

However, as bots and automated programs become more sophisticated, traditional text-based CAPTCHAs have become less effective. Attackers have developed advanced

algorithms and machine learning techniques to bypass or circumvent these challenges. As a result, alternative CAPTCHA methods have emerged to enhance security.

One such alternative is the use of image-based CAPTCHAs. Instead of distorted characters, users are presented with images and asked to identify specific objects or patterns within the images. For example, users may be asked to select all the images that contain a specific type of object, such as cars or traffic lights. This approach leverages the superior visual recognition abilities of humans while posing challenges for automated systems.

Another variation is the audio CAPTCHA, which plays a sequence of distorted or garbled sounds and requires users to input the correct sequence of characters or words they hear. This method accommodates individuals with visual impairments who may have difficulty solving visual challenges.

Furthermore, CAPTCHA systems have evolved to incorporate additional security measures and user-friendly features. For instance, some CAPTCHAs include a time limit to prevent attackers from employing advanced algorithms to decipher the challenge. Additionally, CAPTCHAs may provide an option to request a new challenge if the current one is too difficult to solve.

In recent years, the concept of "invisible CAPTCHAs" has emerged. Instead of presenting a visible challenge to users, these CAPTCHAs employ behind-the-scenes algorithms that analyze user behavior, mouse movements, or browsing patterns to assess whether the user is human or a bot. This approach aims to provide a seamless user experience without

explicit interaction or interruption.

While CAPTCHAs are effective in distinguishing between humans and bots, they can sometimes be frustrating for users, particularly if the challenges are overly complex or difficult to perceive. To strike a balance between security and user experience, it is important for developers to design CAPTCHAs that are effective, accessible, and user-friendly.

CAPTCHAs play a crucial role in safeguarding online platforms from automated attacks by verifying the human presence. Through various forms of challenges, CAPTCHAs help protect websites, online services, and user data by creating barriers that are difficult for bots to overcome. As technology evolves, CAPTCHA systems continue to adapt and improve, ensuring a secure and reliable online environment.

The core of our project revolves around leveraging deep learning techniques, particularly Convolutional Neural Networks (CNNs), to tackle the image recognition captchas. CNNs are a type of artificial neural network architecture designed specifically for processing and analyzing visual data, making them an ideal choice for tasks such as image classification, object detection, and pattern recognition.

In the context of our project, CNNs are utilized to learn and extract relevant features from the captcha images. This is achieved through a series of convolutional layers, pooling layers, and fully connected layers that make up the CNN architecture.

**1.1 Convolutional layers**

These layers perform the convolution operation, where small filters are applied to the input image to extract different features. Each filter scans across the image, computing dot products with local patches of the image. This process helps capture spatial relationships between pixels and learn hierarchical representations of features.

Pooling layers: Pooling layers are interspersed between convolutional layers and serve to downsample the feature maps. They reduce the spatial dimensions while retaining the important information. Common pooling techniques include max pooling, where the maximum value in each patch is retained, or average pooling, where the average value is taken.

Fully connected layers: These layers connect the features extracted from the convolutional and pooling layers to the final output. They enable the network to learn complex relationships between the extracted features and make predictions. The fully connected layers consist of multiple neurons, each contributing to the final output through weighted connections.

During the training phase, the CNN learns the optimal weights and biases of its neurons by iteratively adjusting them to minimize the difference between predicted outputs and ground truth labels. This process is known as backpropagation, and it utilizes optimization algorithms like stochastic gradient descent to update the network's parameters.

The training dataset, consisting of labeled captcha images, is fed into the CNN, which learns to recognize and classify the objects within the images. Through this iterative learning process, the network gradually improves its ability to accurately predict the correct answers for unseen captchas.

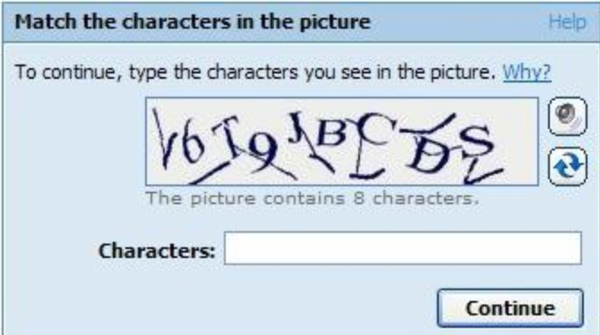
Once trained, the CNN can be used for inference on new captchas. The preprocessed captcha images are passed through the trained network, and the output of the network represents the predicted class labels or answers for the captchas. These predictions are then compared to the ground truth labels to evaluate the accuracy and performance of the model.

The power of deep learning lies in its ability to automatically learn and extract meaningful features from raw data, without the need for explicit feature engineering. CNNs, in particular, have demonstrated exceptional performance in various image recognition tasks, surpassing human-level accuracy in many cases. By harnessing the capabilities of CNNs, we can effectively tackle the complex and diverse challenges presented by image recognition captchas.

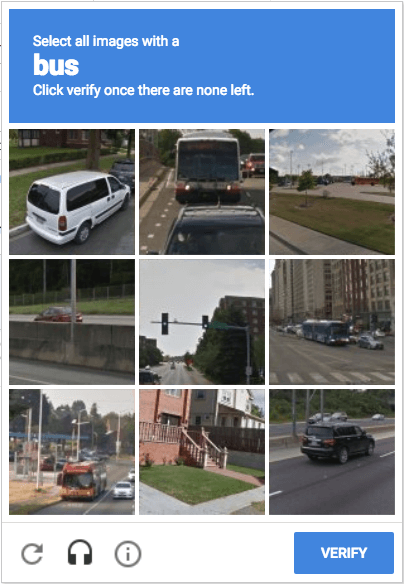
Through this deep learning approach, our project aims to develop a robust and accurate solution for solving image recognition captchas based on vehicles. By training a CNN model on a large dataset of labeled captcha images and leveraging the network's ability to learn intricate visual patterns, we strive to achieve a high success rate in automatically solving these captchas, highlighting the vulnerabilities and limitations of current captcha systems.

**1.2 Types of CAPTCHA**

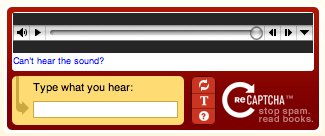
1. Text-based CAPTCHA: This is the most common type of CAPTCHA that presents users with distorted or obscured text characters. Users are required to type the characters correctly to pass the verification process.



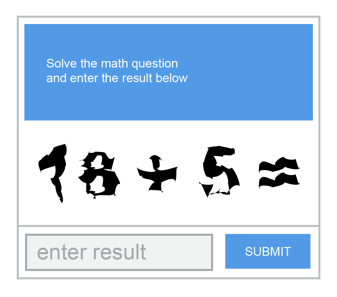
2. Image-based CAPTCHA: Instead of text, this type of CAPTCHA presents users with images and asks them to select specific objects, recognize patterns, or perform certain actions to prove their human identity.



3. Audio CAPTCHA: In situations where visual challenges are not suitable, audio CAPTCHA provides an alternative. Users are required to listen to an audio clip and accurately transcribe the spoken characters or words.



4. Math CAPTCHA: This type of CAPTCHA presents users with mathematical problems or equations that they must solve to verify their human identity. These can include simple arithmetic operations or more complex mathematical challenges.

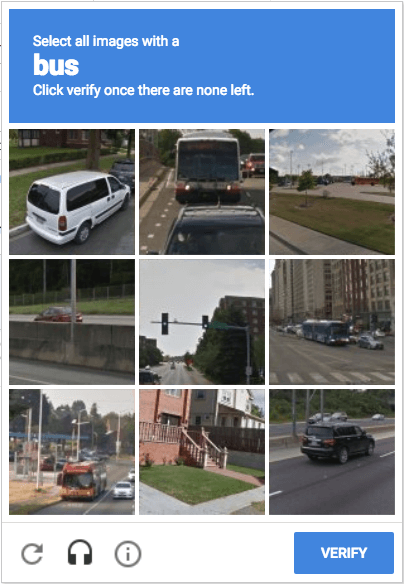


**1.3 Image CAPTCHAs**

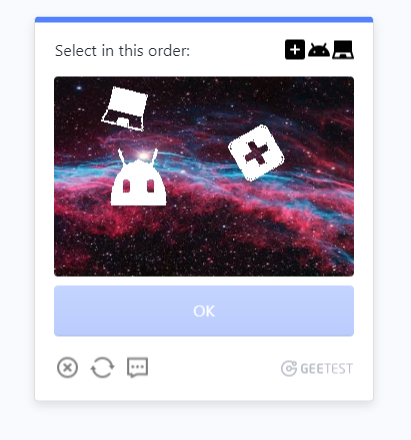
There are several types of image-based CAPTCHAs used to differentiate between humans and bots. These CAPTCHAs typically present users with images and require them to

perform specific tasks or provide certain information. Here are some common types of image CAPTCHAs:

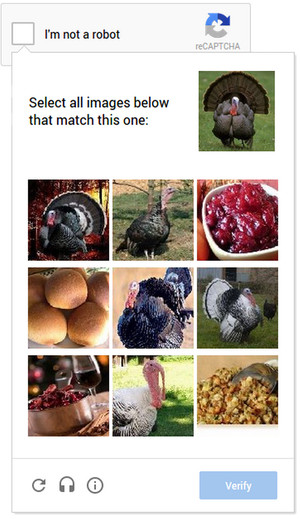
1. Image Recognition CAPTCHA: In this type, users are presented with a set of images and asked to identify specific objects or patterns within those images. For example, users may be required to select images that contain a particular object (e.g., cars, traffic lights, or animals) or images that match a given description (e.g., images with mountains or rivers).



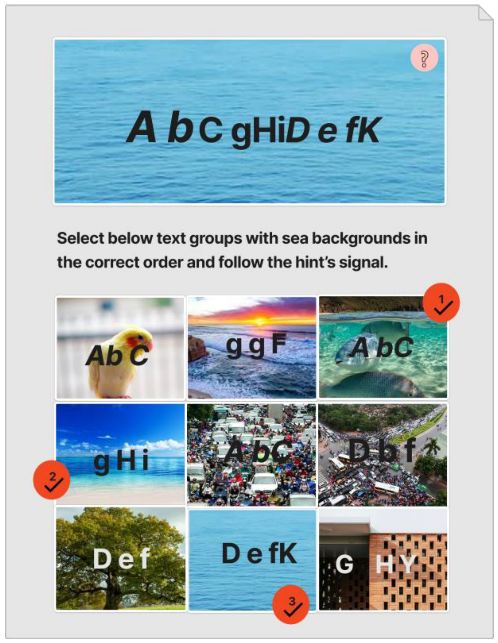
2. Image Ordering CAPTCHA: Here, users are shown a sequence of images that are randomly shuffled, and they are asked to arrange the images in a specific order based on certain criteria. This can involve sorting images by size, color, object type, or any other distinguishing feature.



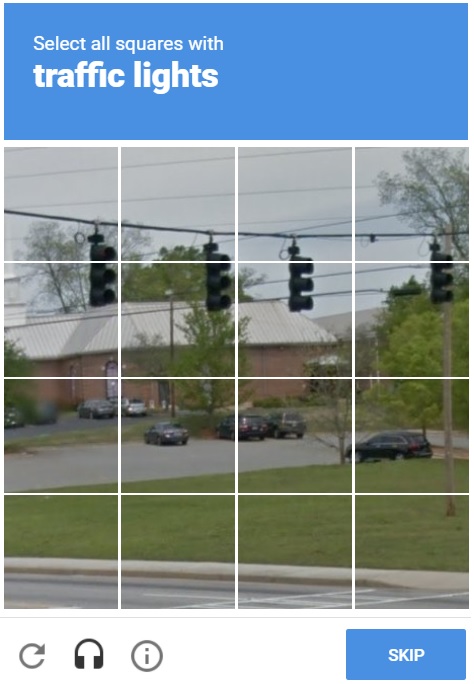
3. Image Matching CAPTCHA: This type requires users to find matching pairs of images from a larger set of images. Users need to identify images that are identical or display similar characteristics. This can involve matching images with similar patterns, colors, shapes, or objects.



4. Image Obfuscation CAPTCHA: In this variant, users are presented with an image that is distorted or partially obscured. They are then asked to identify specific elements or information hidden within the distorted image. This can include deciphering text, recognizing obscured objects, or solving visual puzzles.



5. Image Grid CAPTCHA: Users are presented with a grid or a matrix of images, and they need to select specific images based on given criteria. For example, users may be asked to identify images that fit a particular category, contain a specific object, or have certain visual characteristics.



6. Image Annotation CAPTCHA: This type involves users providing annotations or labels for specific objects or regions within an image. Users may be asked to draw bounding boxes around objects, label different parts of an image, or mark specific features within the image.

**1.4 Image recognition CAPTCHA**

Image recognition CAPTCHA is an effective security measure that utilizes visual content to differentiate between human users and automated bots. By presenting users with images and requiring them to perform specific tasks related to the visual content, this approach leverages human visual understanding while posing challenges to automated algorithms.

The primary objective of image recognition CAPTCHA is to enhance the security of online applications by preventing automated attacks. Bots often struggle with image analysis and object recognition, making it difficult for them to pass the CAPTCHA test. This helps protect against brute-force login attempts, spam registrations, and content scraping.

Moreover, image recognition CAPTCHA offers improved accessibility for users with visual impairments. Traditional text-based CAPTCHAs can be problematic for individuals who rely on screen readers or have difficulty reading distorted text. Image recognition CAPTCHA provides an inclusive alternative that relies on visual comprehension rather than textual interpretation.

Implementing image recognition CAPTCHA involves advanced computer vision techniques and machine learning algorithms. These technologies enable the system to analyze and interpret visual content, identify objects or patterns, and evaluate user responses.

In summary, image recognition CAPTCHA serves as a robust security measure by challenging automated bots while ensuring accessibility for all users. It strengthens the protection of online platforms, mitigates the risk of unauthorized access, and fosters a secure and user-friendly online environment.

In our Project we have created an image recognition CAPTCHA within a simple website which allows its user to click and verify whether they have given the correct input or not according to the questions asked.

**1.5 Advantages of CAPTCHA**

1. Security: CAPTCHA helps protect websites and online services from automated attacks, such as brute-force login attempts, account creation by bots, and spam comments or messages.

2. User Verification: By requiring users to solve CAPTCHA challenges, websites can ensure that interactions are performed by real human users and not automated scripts or bots.

3. Data Protection: CAPTCHA helps safeguard sensitive information by preventing automated bots from accessing and extracting data from websites and online forms.

4. User Experience: While CAPTCHA can sometimes be seen as an additional step in user interactions, it provides a necessary layer of security without significantly compromising the user experience.

**1.6 Why to Use CAPTCHA**

1. User Registration: CAPTCHA can be used during the user registration process to prevent automated bots from creating fake accounts or spamming the system.

2. Login Pages: Implementing CAPTCHA on login pages helps protect user accounts from brute-force attacks and unauthorized access attempts.

3. Comment Sections and Contact Forms: CAPTCHA can be used to filter out spam comments and messages by ensuring that only genuine users can submit content or contact the website owner.

4. Online Transactions: CAPTCHA can enhance the security of online transactions, such as purchasing products or completing financial transactions, by verifying the human identity of users.

CAPTCHA is a crucial security measure used to verify the authenticity of human users and protect websites and online services from automated attacks and spam. By employing various challenges that are easy for humans but difficult for machines, CAPTCHA helps ensure the integrity and security of online interactions.

**1.7 Problem statement**

### The aim of this project is to develop a system that can accurately recognize different types of vehicles in an Image Recognition Captcha using Convolutional Neural Networks (CNN) and implement CAPTCHA for user verification. The technical approach combines the

### power of deep learning for image classification with CAPTCHA as a security measure to distinguish between human users and automated bots.

### 1. Data Collection and Preprocessing:

### The project starts with the collection of a diverse dataset of vehicle images, including airplanes, bicycles, boats, motorbuses, motorcycles, seaplanes, trains, and trucks. The images are stored in separate directories for each vehicle type. The images are then preprocessed by resizing them to a uniform size of 150x150 pixels and converting them into numpy arrays for further processing.

### 2. Convolutional Neural Network Architecture:

### A CNN architecture is employed to learn the features and patterns in the vehicle images for classification. The CNN consists of multiple layers, including convolutional layers, pooling layers, and fully connected layers. The architecture follows the best practices for image classification, with increasing complexity and abstraction of features as the network deepens.

### 3. Data Augmentation:

### To increase the diversity of the training dataset and improve the model's generalization, data augmentation techniques are applied using the ImageDataGenerator class from Keras. These techniques include random rotations, zooming, shifting, and flipping of the images. Data augmentation helps the model learn variations in vehicle images and reduces overfitting.

### 4. Training and Optimization:

### The model is trained using the compiled dataset and augmented images. The Adam optimizer is utilized with a learning rate of 0.001. The loss function used is categorical cross-entropy, suitable for multi-class classification. The training process is run for a fixed number of epochs while monitoring the validation accuracy. A ReduceLROnPlateau callback is employed to adjust the learning rate dynamically to improve convergence.

### 5. Evaluation and Visualization:

### During the training process, metrics such as accuracy and loss are monitored to evaluate the model's performance. Plots of training and validation loss and accuracy are generated to assess the model's convergence and generalization. The final trained model is then evaluated on the test dataset to measure its overall performance.

### 6. CAPTCHA Implementation:

### Alongside vehicle recognition, the project incorporates CAPTCHA as a security measure to differentiate between human users and automated bots. CAPTCHA challenges are added to the user interaction process to verify the user's authenticity. Various types of CAPTCHA, such as text-based, image-based, and math-based challenges, can be implemented depending on the specific requirements of the application.

### The technical approach outlined in this project combines the use of Convolutional Neural Networks for vehicle recognition and CAPTCHA for user verification. By employing deep learning techniques, the model can accurately classify different types of vehicles. Additionally, the inclusion of CAPTCHA enhances the security of user interactions and protects against automated attacks. The integration of these two components provides a robust solution for vehicle recognition with an added layer of security.

### 

### 1.8 Objectives

### The objective of this project is to develop a robust system that can accurately recognize different types of vehicles using Convolutional Neural Networks (CNN) and implement CAPTCHA for user verification.

### The first objective of the project is vehicle recognition. In today's world, where vehicles play a significant role in transportation and logistics, the ability to automatically identify and classify different types of vehicles has become increasingly important. By leveraging the power of deep learning, specifically CNNs, the project aims to build a model that can classify various types of vehicles accurately. This includes airplanes, bicycles, boats, motorbuses, motorcycles, seaplanes, trains, and trucks. By training the model on a diverse dataset of vehicle images, the objective is to achieve high accuracy in classifying the vehicles.

### To achieve accurate vehicle recognition, the project utilizes CNNs. CNNs are particularly effective in image classification tasks as they can automatically learn hierarchical representations of images. The network architecture consists of multiple convolutional layers that extract features from the input images, followed by pooling layers that reduce the spatial dimensions. These layers are then connected to fully connected layers that perform the final classification. By training the model on a large dataset of labeled vehicle images, the CNN learns to recognize the distinguishing features of each vehicle type.

### Data augmentation is another important aspect of the project. Since the availability of labeled vehicle images might be limited, data augmentation techniques are applied to artificially increase the diversity and variability of the dataset. By applying various transformations, such as rotations, zooming, shifting, and flipping, to the training images, the project aims to provide the model with more varied examples to learn from. This helps in improving the model's performance and generalization by exposing it to a wider range of vehicle variations.

### Model optimization is a key objective to ensure the best performance of the CNN model. This involves selecting appropriate hyperparameters, such as learning rate, loss function, and optimizer. The objective is to fine-tune the model's hyperparameters to achieve better convergence and accuracy during training. Additionally, regularization techniques, such as dropout and batch normalization, are applied to prevent overfitting and improve the model's generalization ability. This optimization process is essential for obtaining a well-performing and efficient vehicle recognition system.

### Evaluation and visualization are critical steps in assessing the performance of the trained model. The project aims to evaluate the model using various metrics, such as accuracy and loss, to assess its performance on both the training and test datasets. Visualizations, such as loss and accuracy plots, will be generated to analyze the model's convergence and generalization. These evaluations provide insights into the model's strengths, weaknesses, and areas of improvement.

### In addition to vehicle recognition, the project incorporates CAPTCHA implementation for user verification. CAPTCHA is an acronym for Completely Automated Public Turing test to tell Computers and Humans Apart. Its objective is to distinguish between human users and automated bots. By implementing CAPTCHA challenges, such as text-based, image-based, or math-based, the system verifies the authenticity of the user and enhances the security of user interactions. CAPTCHA helps prevent automated bots from engaging in malicious activities such as spamming, brute-force attacks, and data scraping.

### The implementation of CAPTCHA involves generating challenges that are easy for humans to solve but difficult for bots. This can include tasks like recognizing distorted characters, selecting specific objects from images, or solving simple mathematical equations. By successfully completing the CAPTCHA challenge, human users can prove their authenticity and gain access to the system, while automated bots are prevented from unauthorized access.

### Our project aims to combine vehicle recognition using CNNs with CAPTCHA implementation to develop an accurate and secure system. By achieving accurate vehicle recognition, the system can facilitate various applications in transportation and logistics, such as automated vehicle tracking, traffic monitoring, and vehicle inventory management. Thee system for classifying vehicles and verifying user authenticity.

**CHAPTER 2**

# LITERATURE SURVEY

CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) is a widely used security mechanism designed to distinguish between humans and automated bots. It involves presenting users with challenges that are easy for humans to solve but difficult for machines. CAPTCHA has been extensively researched and studied in various domains, including computer vision, machine learning, and security. In this literature survey, we will explore some key studies and advancements in the field of CAPTCHA.

From [1] we learn that this study proposes a novel CAPTCHA recognition technique using deep learning. The authors employ a Convolutional Neural Network (CNN) architecture to accurately recognize text-based CAPTCHAs. They demonstrate superior performance compared to traditional image processing techniques, achieving high accuracy and robustness against various CAPTCHA variations.

From [2] we learn that this research investigates the vulnerability of text-based CAPTCHAs to machine learning attacks. The authors develop an automated system that leverages advanced machine learning algorithms to successfully break popular CAPTCHA schemes. They highlight the limitations of text-based CAPTCHAs and provide insights into improving their security.

From [3] we learn that work proposes a CAPTCHA recognition approach based on image

segmentation and Support Vector Machines (SVM). The authors employ image segmentation techniques to extract individual characters from CAPTCHA images and use SVM for classification. Their method achieves high accuracy and robustness against noise and distortion.

From [4] This survey provides a comprehensive overview of various CAPTCHA methods and their variants. The authors categorize CAPTCHAs based on different characteristics, such as text-based, image-based, audio-based, and interactive CAPTCHAs. They discuss the strengths, weaknesses, and security implications of each type, along with recent advancements in CAPTCHA technologies.

From [5] This research focuses on the security analysis of text-based CAPTCHAs and proposes a methodology for evaluating their vulnerability to attacks. The authors analyze different types of attacks, including OCR-based attacks, segmentation attacks, and semantic attacks. They provide insights into designing more secure CAPTCHA systems.

From [6] This study explores the adversarial attacks and defenses in CAPTCHA systems. The authors investigate the vulnerability of CAPTCHAs to adversarial attacks, such as generative attacks and black-box attacks, and propose defense mechanisms to enhance security. They highlight the importance of considering adversarial scenarios in CAPTCHA design.

From [7] This research proposes a CAPTCHA recognition method utilizing transfer learning and Recurrent Neural Networks (RNN). The authors demonstrate the effectiveness of transfer learning by fine-tuning pre-trained models on limited CAPTCHA datasets. Their approach achieves high accuracy and robustness even with limited training data.

From [8] This study proposes a CAPTCHA recognition approach using Artificial Neural Networks (ANN) and Genetic Algorithms (GA). The authors train ANNs to recognize characters in CAPTCHAs, and GA is used to optimize the ANN's architecture and improve its performance. The combination of ANN and GA demonstrates promising results in accurately recognizing CAPTCHAs.

From [9] This survey focuses on the usability aspects of CAPTCHAs. The authors examine user perceptions, experiences, and difficulties encountered while interacting with different types of CAPTCHAs. They analyze the trade-off between security and user convenience, providing insights into designing user-friendly CAPTCHA systems without compromising security.

From [10] This paper explores the challenges and techniques related to audio CAPTCHAs, which are designed for visually impaired users. The authors discuss various audio CAPTCHA schemes and their effectiveness in providing accessible security. They address issues such as robustness, usability, and the trade-off between security and user experience.

From [11] This study investigates the vulnerability of image-based CAPTCHAs to deep learning-based attacks. The authors propose a deep learning approach using Convolutional Neural Networks (CNNs) to break popular image-based CAPTCHAs. They highlight the need for more robust and secure image-based CAPTCHA designs to counter emerging attacks.

From [12] This survey explores the concept of using CAPTCHAs as graphical passwords. The authors discuss the advantages and challenges of graphical CAPTCHAs in terms of usability and security. They analyze various graphical CAPTCHA schemes and their effectiveness in providing secure and user-friendly authentication.

From [13] This research investigates the vulnerability of advanced audio CAPTCHAs to machine learning attacks. The authors demonstrate successful attacks on state-of-the-art audio CAPTCHA systems using deep learning and Hidden Markov Models (HMMs). They emphasize the need for robust defenses to ensure the security of audio-based CAPTCHAs.

From [14] This study presents a comparative analysis of various CAPTCHA solving techniques, including OCR-based methods, machine learning approaches, and crowd-sourcing methods. The authors evaluate the strengths and weaknesses of each technique and discuss their applicability in different scenarios. They provide insights into selecting appropriate CAPTCHA solving techniques based on specific requirements.

**CHAPTER 3**

# OVERVIEW ON ATTACK SIMULATION ON CAPTCHAs

**3.1 PROPOSED SYSTEM APPROACH (HCAPTCHA\_CLONE)**

This page is designed to present users with a survey or CAPTCHA-like challenge that involves selecting images from a set of options. Here's a proposed survey based on the functionality described in the code:

1. Introduction:

- Welcome message explaining the purpose of the survey or challenge.

2. Image Selection:

- Display a set of 9 images, each representing a different category.

- Instruct the user to select all images that belong to a specific category (e.g., the category displayed in the element with the ID "name").

- Allow the user to click on an image to select or deselect it.

- Provide visual feedback (e.g., adding a border or highlighting) to indicate the selected images.

3. CAPTCHA Verification:

- Display a submit button for the user to indicate their selection is complete.

- When the submit button is clicked:

- Prevent form submission to allow verification.

- Check if the selected images match the randomly generated category.

- If the selection is correct:

- Display a success message.

- Show an image (e.g., a checkmark) to indicate successful verification.

- Update the text content of an element with the ID "cap\_text" to indicate successful verification.

- If the selection is incorrect:

- Display a failure message.

- Show an image (e.g., a cross) to indicate unsuccessful verification.

- Update the text content of an element with the ID "cap\_text" to indicate unsuccessful verification.

4. Popup Interaction:

- Allow the user to open and close a popup by clicking on the "Check" element and the "inner\_button" element, respectively.

- The popup can contain additional information, instructions, or any other relevant content.

5. Completion:

- Once the CAPTCHA is successfully verified, the user can proceed to the next step or perform any desired action.

## 3.2 PROPOSED SYSTEM APPROACH (MODEL)

1. Data Collection and Preparation:

* Collect a dataset of images containing various vehicle types (airplanes, bicycles, boats, motorbuses, motorcycles, seaplanes, trains, trucks).
* Prepare the dataset by resizing the images to a consistent size and encoding the labels for each image.

2. Data Augmentation:

* Apply data augmentation techniques such as rotation, zooming, and flipping to increase the dataset's diversity and improve the model's robustness.

3. Model Architecture Design:

* Design a Convolutional Neural Network (CNN) model architecture suitable for image recognition tasks.
* The architecture should include convolutional layers for feature extraction, pooling layers for downsampling, and fully connected layers for classification.

4. Model Training:

* Split the dataset into training and validation sets.
* Train the CNN model using the training set, optimizing the model's parameters through backpropagation and gradient descent.
* Monitor the model's performance on the validation set and make adjustments if necessary.

5. Model Evaluation:

* Evaluate the trained model on a separate test set to measure its accuracy and effectiveness in solving image recognition CAPTCHAs.
* Calculate performance metrics such as accuracy, precision, recall, and confusion matrix to assess the model's performance.

6. CAPTCHA Breaking:

* Apply the trained model to new CAPTCHA images to predict the vehicle type depicted in each image.
* Utilize the model's predictions to successfully break the image recognition CAPTCHAs.

**CHAPTER 4**

# METHODOLOGY USED FOR ATTACKING THE CAPTCHAs

1. Data Collection: We curate a dataset of captcha images, comprising distorted characters or objects, which serves as our training and evaluation data.

2. Preprocessing: We apply image preprocessing techniques such as resizing, normalization, and noise reduction to improve the quality and clarity of the captcha images. These techniques help in enhancing the signal-to-noise ratio and removing any distortions that may hinder recognition.

3. Convolutional Neural Network (CNN): We utilize CNN, a deep learning architecture well-suited for image recognition tasks. CNN consists of convolutional layers that extract features from the captcha images and pooling layers that downsample the spatial dimensions. The extracted features are then passed through fully connected layers for classification.

4. Training: We train the CNN model using our labeled captcha dataset. The training process involves optimizing the network's parameters through backpropagation and gradient descent, minimizing a loss function to improve the model's recognition accuracy.

5. Attack Strategy: We devise an attack strategy to circumvent the captcha verification. This strategy may involve generating adversarial examples using techniques like gradient-based optimization or exploiting vulnerabilities in the recognition algorithm. The goal is

to manipulate the recognized characters or objects in a way that fools the captcha system.

6. Success Evaluation: We assess the success of the attack strategy by measuring the attack's effectiveness in bypassing the captcha verification. This evaluation involves attempting to solve a set of captchas using the attack strategy and comparing the number of successful bypasses to the total number of attempts.

7. Iterative Refinement: Based on the evaluation results, we refine the attack strategy iteratively. This refinement may involve fine-tuning the CNN model, exploring different attack techniques, or adapting to any countermeasures implemented by the captcha system to improve the attack's success rate.

8. Security Implications: We analyze the implications of successfully breaking the captcha system. This analysis includes identifying potential security vulnerabilities, understanding the impact on systems relying on captchas for security, and guiding the development of stronger and more resilient captcha mechanisms.

By employing techniques like CNNs, adversarial attacks, and iterative refinement, our project aims to explore the strengths and weaknesses of captcha systems, contributing to advancements in captcha security and the development of more robust defense mechanisms.

The project described falls under the domain of convolutional neural networks (CNNs), which is a type of deep learning architecture specifically designed for image recognition and processing tasks. CNNs are widely used in computer vision applications and have proven to be highly effective in tasks such as object detection, image classification, and pattern recognition.

In this project, a CNN model is utilized to tackle the task of breaking image recognition captchas. The model is constructed by stacking multiple convolutional layers, pooling layers, and fully connected layers. The convolutional layers extract features from the input captcha images, capturing patterns and relevant information. The pooling layers reduce the spatial dimensions of the extracted features while retaining their essential characteristics. Finally, the fully connected layers learn to classify the extracted features into different captcha classes.

4.1 Convolutional Neural Network

CNN stands for Convolutional Neural Network, which is a type of deep learning algorithm specifically designed for processing and analyzing visual data, such as images and videos. CNNs are widely used in computer vision tasks, including image classification, object detection, and image segmentation.

At a high level, a CNN consists of multiple interconnected layers that process input data in a hierarchical manner. The key components of a CNN include:

1. Convolutional Layers: These layers apply a set of learnable filters or kernels to the input data, performing convolution operations. Convolution helps in extracting local patterns, edges, and features from the input image.

2. Pooling Layers: Pooling layers downsample the spatial dimensions of the input data, reducing the computational complexity and the number of parameters. Max pooling and average pooling are common pooling operations used in CNNs.

3. Activation Functions: Activation functions introduce non-linearity into the network, allowing it to learn complex relationships and make predictions. Common activation functions used in CNNs include ReLU (Rectified Linear Unit), sigmoid, and tanh.

4. Fully Connected Layers: These layers connect all neurons from the previous layer to the next layer, similar to traditional neural networks. They help in combining features learned from earlier layers and making final predictions.

5. Loss Function: The loss function measures the discrepancy between the predicted output and the actual output. It quantifies the model's performance during training and guides the learning process.

6. Optimization Algorithm: CNNs use optimization algorithms, such as Stochastic Gradient Descent (SGD) or Adam, to update the network's weights and biases during the training phase. The goal is to minimize the loss function and improve the model's performance.

CNNs have revolutionized the field of computer vision and achieved remarkable success in various tasks. Their ability to automatically learn hierarchical representations and capture spatial dependencies in data makes them highly effective in image analysis and recognition tasks.

**4.2 Why we have used it**

In our project, we have used a Convolutional Neural Network (CNN) architecture for image recognition. Specifically, we have implemented a CNN model with multiple convolutional layers, pooling layers, and fully connected layers.

The architecture we have used is a variation of the basic CNN architecture commonly used for image classification tasks. It includes convolutional layers that apply filters to extract meaningful features from the input images. These filters capture different patterns and structures present in the images.

The convolutional layers are followed by pooling layers, which downsample the feature maps to reduce their spatial dimensions while retaining the important information. This helps in reducing the computational complexity and extracting the most salient features.

After the pooling layers, we have fully connected layers that process the extracted features and make predictions. These layers combine the features from the previous layers and perform classification based on the learned representations.

Overall, our CNN model is designed to learn and recognize patterns and features in the input images related to different vehicle types. By training the model on a large dataset of vehicle images, it can learn to differentiate between various vehicles and make accurate predictions.

The choice of this CNN architecture is based on its effectiveness in image recognition tasks and its ability to learn hierarchical representations from visual data. By leveraging the power of deep learning and CNNs, we aim to achieve high accuracy in classifying the vehicle images and successfully break the image recognition captchas.

**CHAPTER 5**

# ENVIRONMENT AND TOOLS

In this project, we have utilized the following environments and tools:

1. Google Colab: We have used Google Colab as our development environment. It provides a Jupyter Notebook-like interface with the added advantage of free GPU resources, allowing us to train our deep learning models efficiently.

2. Python: The entire project is implemented using the Python programming language. Python provides a rich ecosystem of libraries and frameworks for machine learning and deep learning tasks.

3. TensorFlow: We have used TensorFlow as the deep learning framework for building and training our Convolutional Neural Network (CNN) model. TensorFlow offers a high-level API that simplifies the process of building and training deep learning models.

4. Keras: Keras is a popular deep learning library that provides a user-friendly interface for building neural networks. We have used Keras as a higher-level abstraction on top of TensorFlow, making it easier to define and train our CNN model.

5. NumPy and Pandas: NumPy and Pandas are essential libraries for data manipulation and preprocessing. We have used NumPy for numerical computations and Pandas for data handling and analysis.

6. Matplotlib and Seaborn: Matplotlib and Seaborn are plotting libraries used for data visualization. We have used these libraries to plot and analyze the performance of our model, visualize images, and display various charts and graphs.

7. OpenCV: OpenCV (Open Source Computer Vision Library) is a powerful library for computer vision tasks. We have utilized OpenCV for image preprocessing, such as resizing images and reading image files.

8. Scikit-learn: Scikit-learn is a versatile library for machine learning tasks. We have used it for data preprocessing, model evaluation, and splitting the dataset into training and testing sets.

These tools and libraries provide a comprehensive ecosystem for deep learning and image recognition tasks, enabling us to develop, train, and evaluate our model effectively.

The environments and tools used in this project include Google Colab, TensorFlow, Keras, NumPy, Pandas, Matplotlib, Seaborn, OpenCV, and Scikit-learn. Let's explore the key features and functionality of each tool:

1. Google Colab:

- Cloud-based development environment that provides free access to GPU resources for faster model training.

- Allows collaborative work and easy sharing of notebooks.

- Provides a user-friendly interface with support for markdown, code cells, and interactive execution.

2. TensorFlow:

- Deep learning framework that offers a wide range of tools and functionalities for building and training neural networks.

- Provides a high-level API (Keras) for faster prototyping and development.

- Supports both CPU and GPU computations.

- Offers advanced features like automatic differentiation, distributed training, and model deployment.

3. Keras:

- Deep learning library built on top of TensorFlow that provides a user-friendly and intuitive interface for designing neural networks.

- Offers a high-level API for building, training, and evaluating models.

- Supports a wide range of neural network architectures and layers.

- Enables easy customization of models through modular building blocks.

4. NumPy:

- Fundamental library for numerical computations in Python.

- Provides powerful array-based operations and mathematical functions for efficient handling and manipulation of large datasets.

- Supports multidimensional arrays and offers a wide range of mathematical operations.

5. Pandas:

- Library for data manipulation and analysis.

- Provides data structures (e.g., DataFrames) and functions for handling structured data.

- Offers powerful data preprocessing capabilities, including data cleaning, transformation, and merging.

- Facilitates data exploration, visualization, and summary statistics.

6. Matplotlib:

- Library for creating static, animated, and interactive visualizations in Python.

- Provides a wide range of plotting functions and customization options.

- Supports various types of plots, including line plots, bar charts, histograms, scatter plots, and heatmaps.

- Allows for detailed customization of plot aesthetics and labels.

7. Seaborn:

- Statistical data visualization library built on top of Matplotlib.

- Offers a high-level interface for creating informative and visually appealing statistical graphics.

- Simplifies the creation of complex visualizations, such as categorical plots, distribution plots, and correlation matrices.

- Provides enhanced aesthetics and built-in statistical estimation.

8. OpenCV:

- Open-source computer vision library with extensive image processing and manipulation functions.

- Enables tasks such as image resizing, cropping, rotation, and color space conversions.

- Supports feature extraction, object detection, and image filtering.

- Provides tools for video analysis and processing.

9. Scikit-learn:

- Comprehensive machine learning library in Python.

- Offers a wide range of algorithms for classification, regression, clustering, and dimensionality reduction.

- Provides utilities for data preprocessing, feature selection, and model evaluation.

- Supports model training, prediction, and serialization.

These tools collectively provide a powerful ecosystem for developing and implementing deep learning models, handling and preprocessing data, visualizing results, and performing machine learning tasks. They offer a rich set of features and functionalities that enhance productivity, facilitate experimentation, and enable efficient development of machine learning projects.

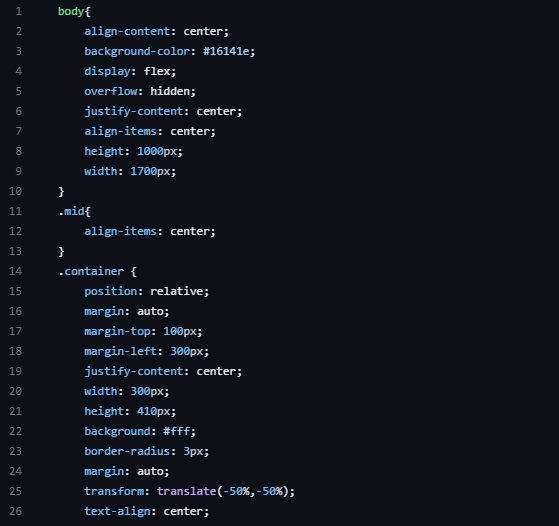
**CHAPTER 5**

**IMPLEMENTATION**

**Creation of the login page :**



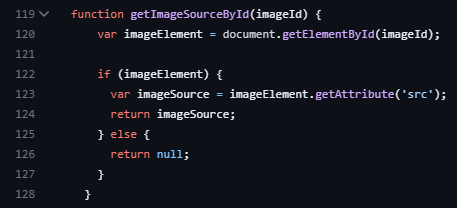
**Stylesheet :**

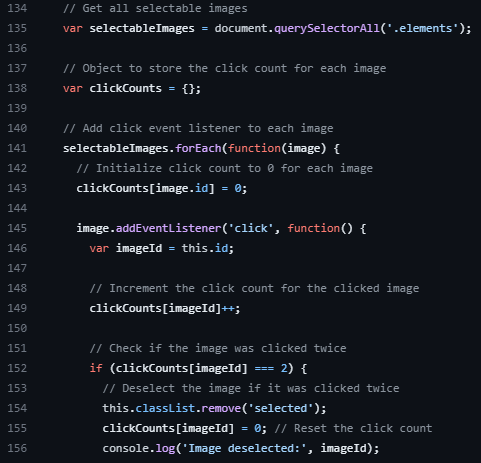
****

**Javascript :**

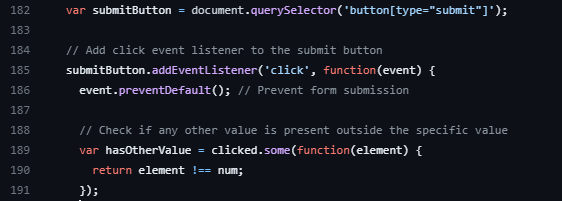
****

****

****

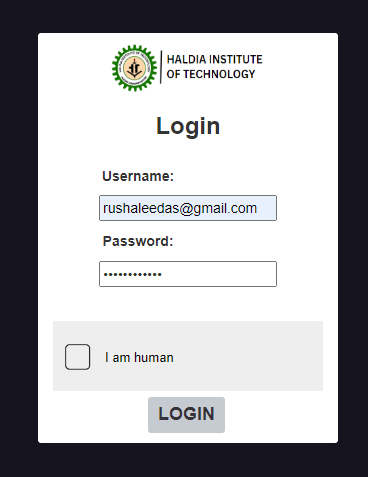
****

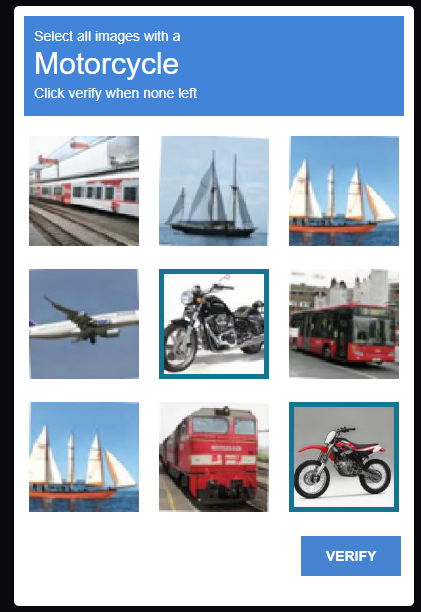
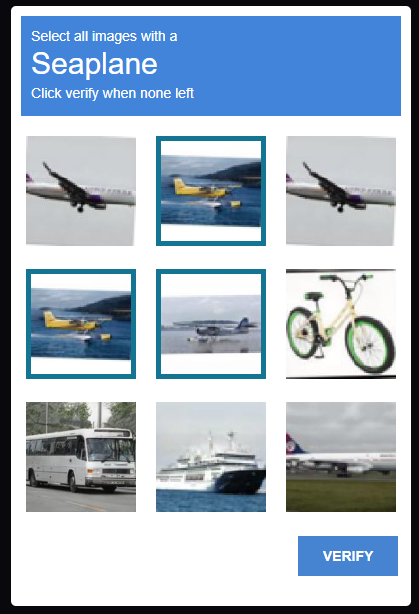
****

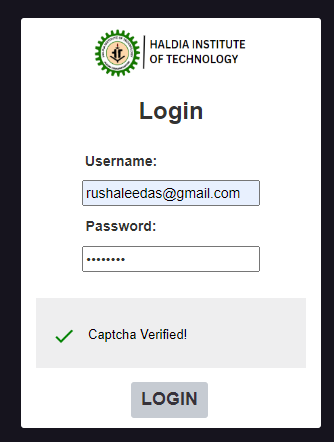
****

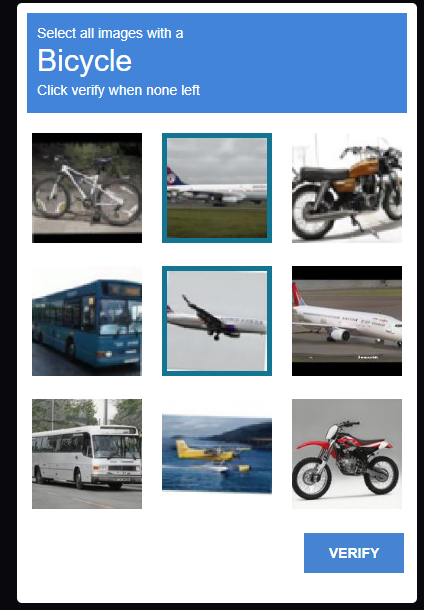
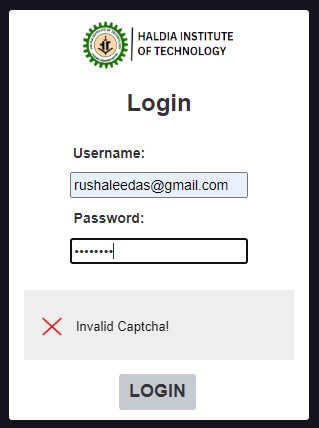
****

****

****

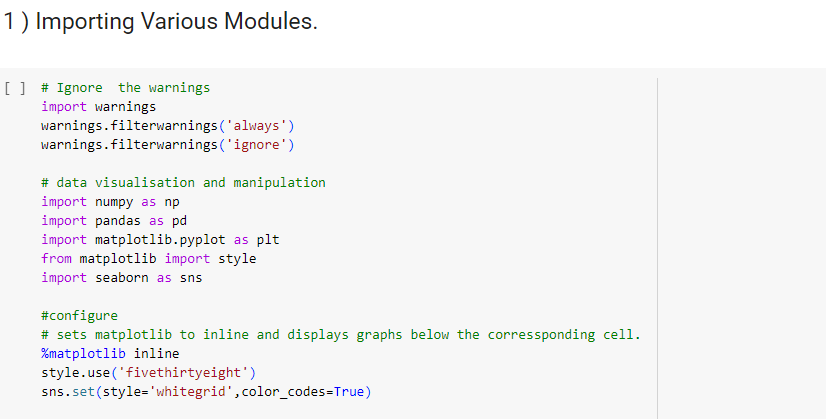
** **

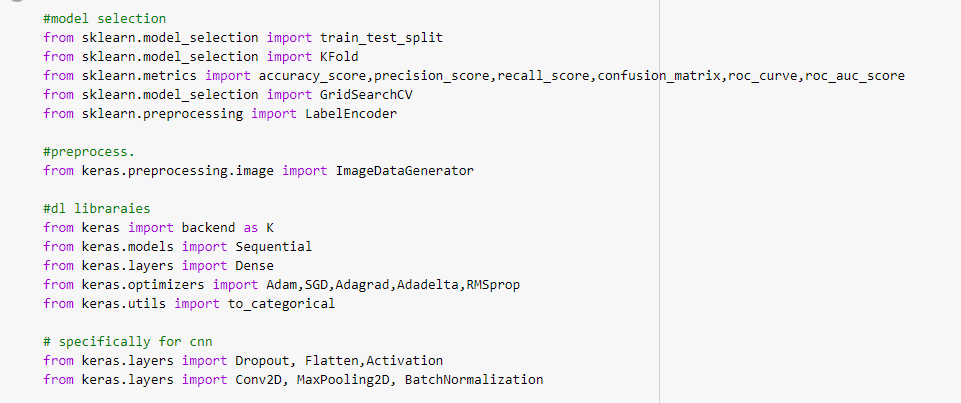
****

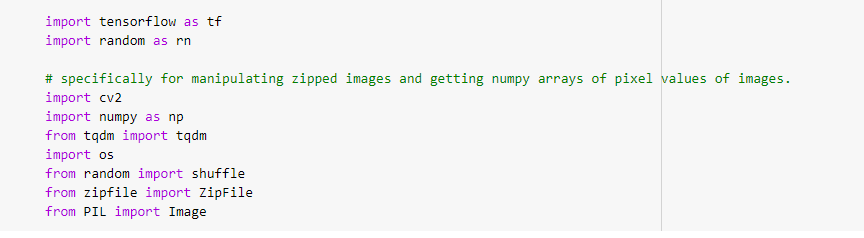
** **

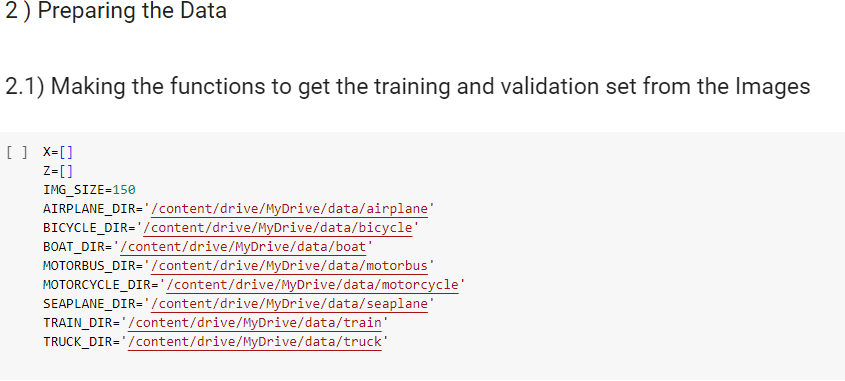
**MODEL CREATION**

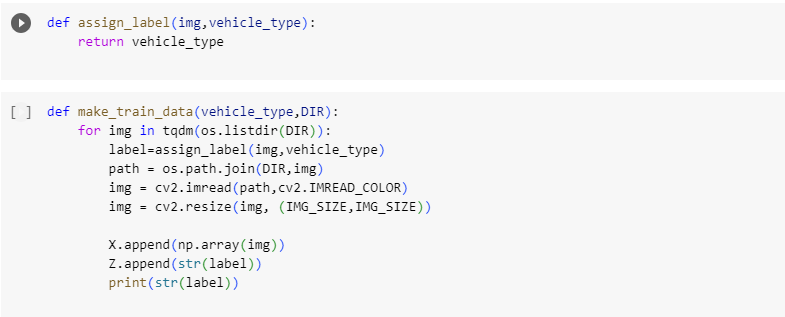
****

****

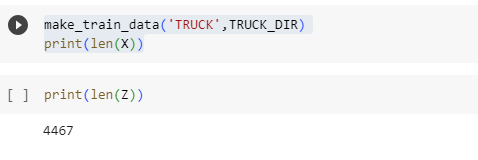
****

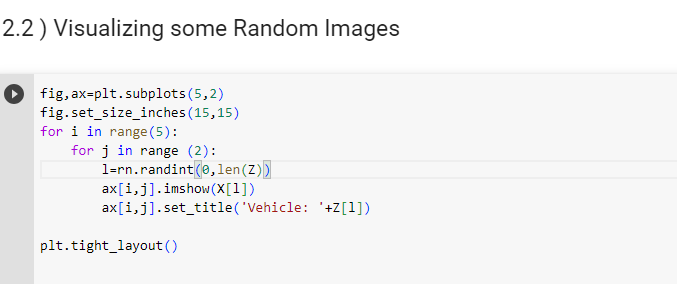
****

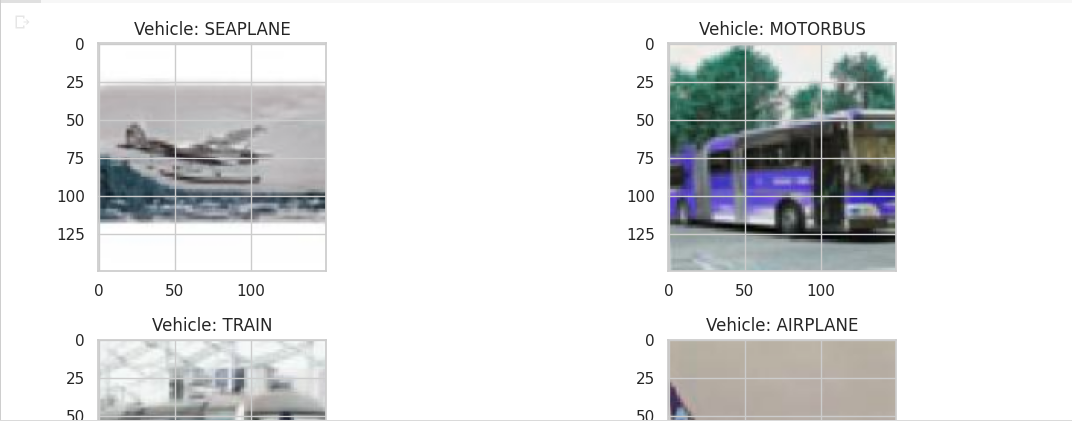
****

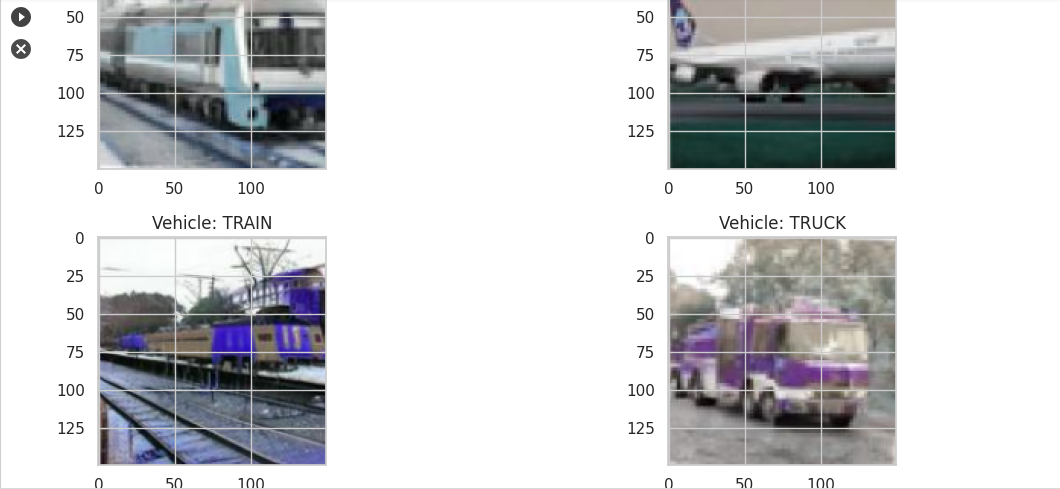
****

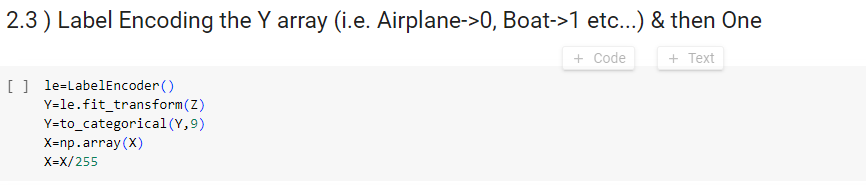
****



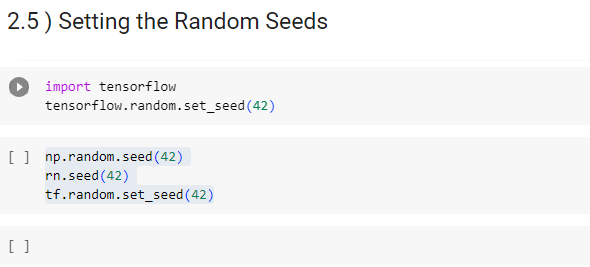
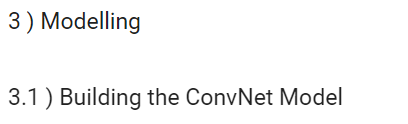


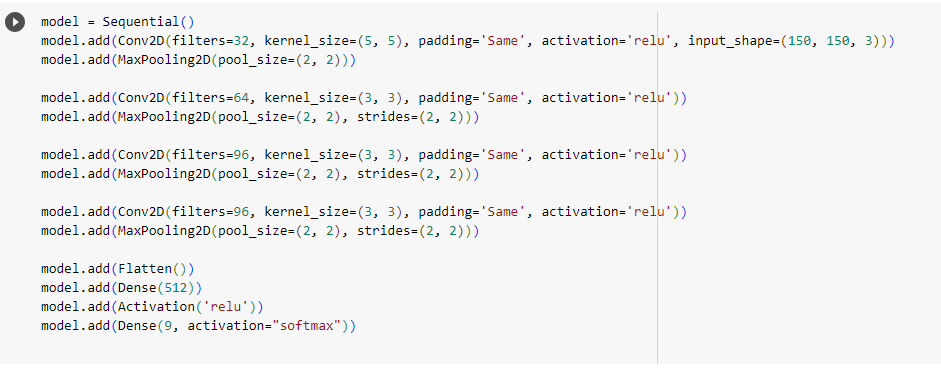




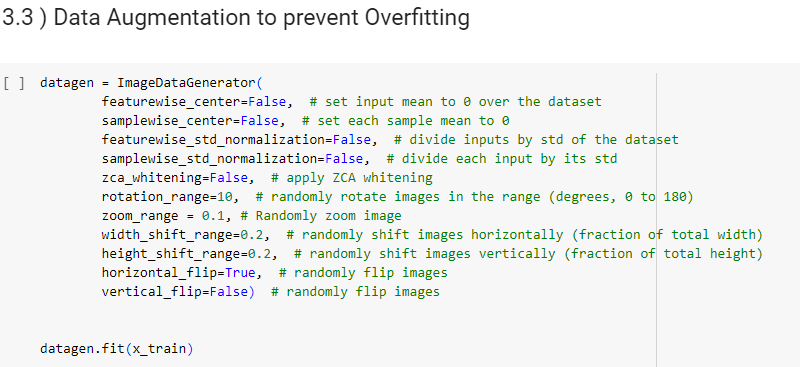




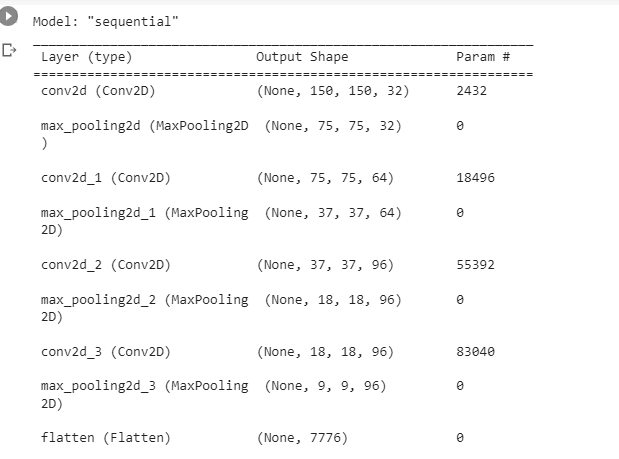


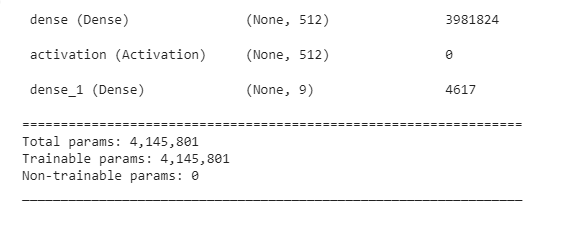


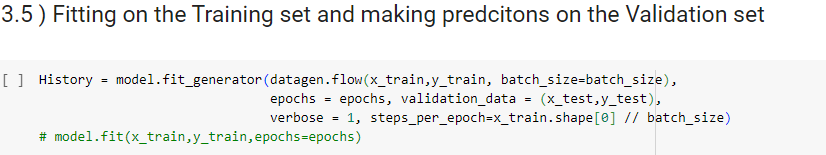


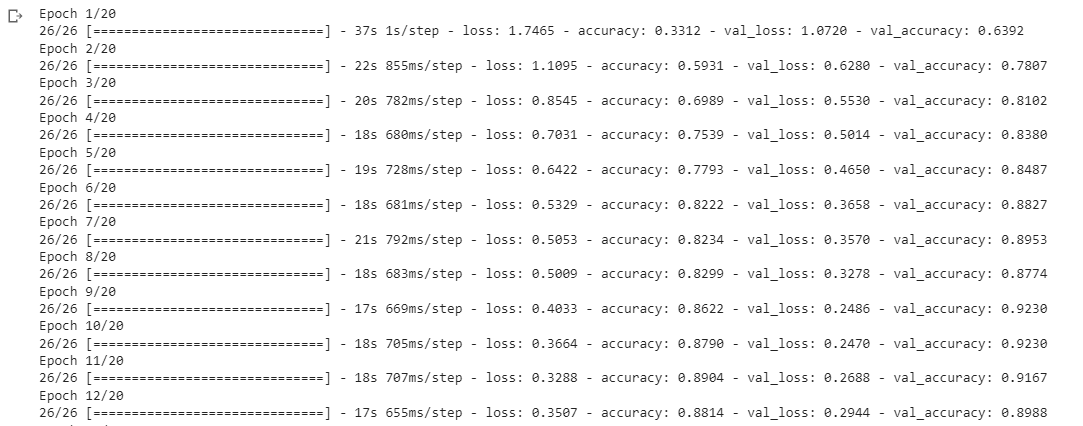


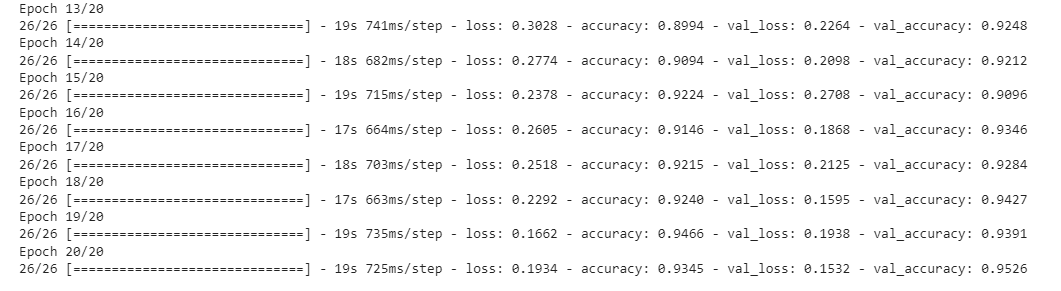


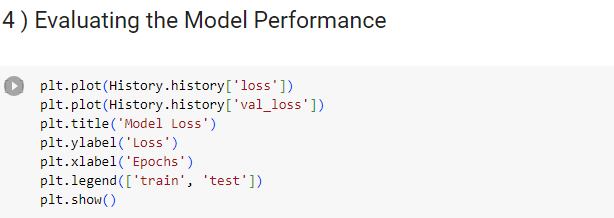


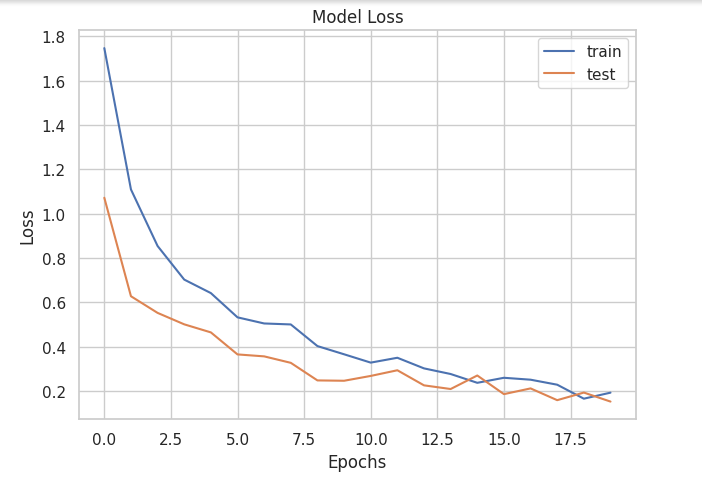


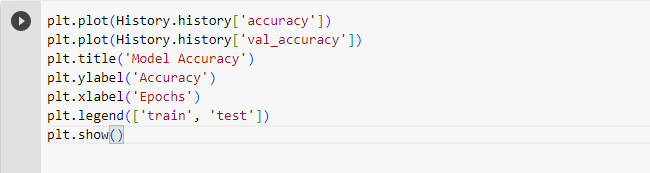


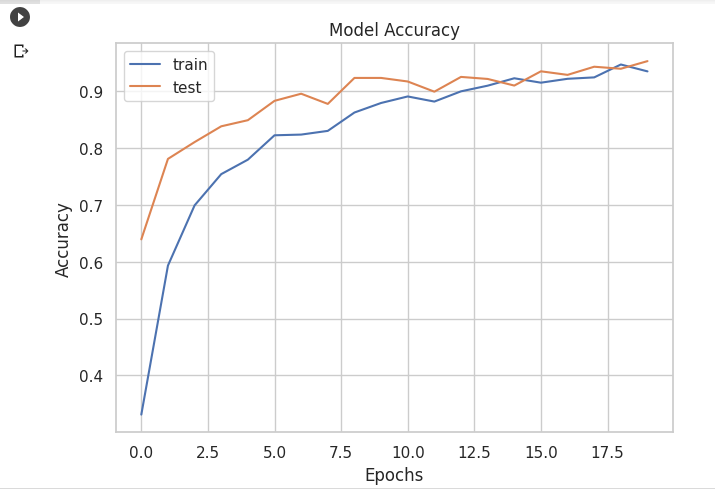


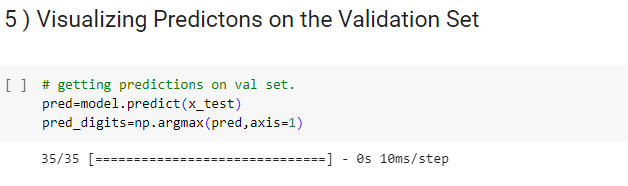










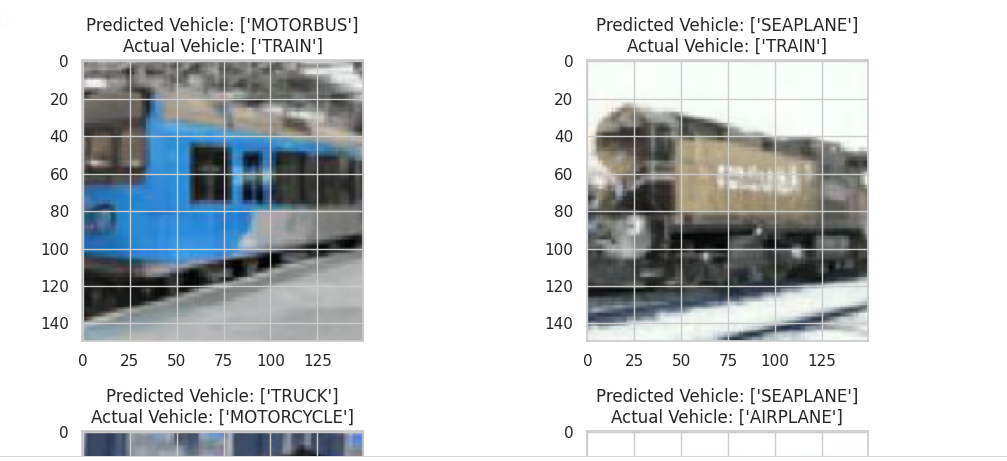












**CHAPTER 7**

# FUTURE WORK

Let's assume that computers will be able to pass the Turing test at some indeterminate time in the future. At that time, a computer will also be able to tell spam apart from quality content, as humans can. This simply obviates the need for CAPTCHAs.

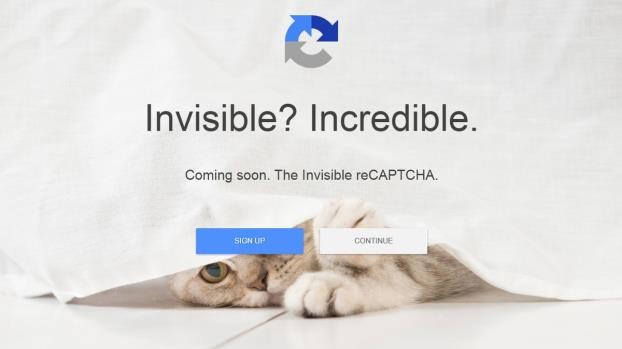
Captchas have been broken:

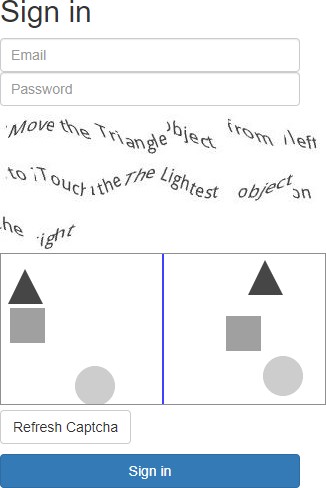
* Audio Captchas that read out the words needed to authenticate a person were used by Microsoft, Digg, eBay and others up until 2011 [when they were decoded by Stanford](https://news.stanford.edu/news/2011/may/captcha-security-flaw-052311.html) [computer scientists.](https://news.stanford.edu/news/2011/may/captcha-security-flaw-052311.html)
* Snapchat's [point and click Captcha](https://venturebeat.com/2014/01/23/snapchat-captcha-problem/) was unpicked by automation.
* In 2014 Google [officially killed the text-based Captcha](https://gigaom.com/2014/12/03/google-kills-captchas-with-new-ai-system/) and replaced it with the 'I'm not a robot' button.
* Three researchers from Columbia University used deep learning to automatically solve 70 per cent of reCaptcha challenges from Google.

The battle between protecting websites from spammers and creating secure Captchas has become invisible. An attempt was made by creating new ways to thwart the attempts to break them. Here are some the few examples of the new Captchas.

Google’s Invisible Captcha

At the end of 2016, Google announced an [Invisible reCaptcha](https://www.wired.co.uk/article/google-wants-to-make-captcha-completely-invisible) that would use what it calls its Advanced Risk Analysis.This system uses Google's AI to look for signs of human behaviour. It runs in the background detecting movements of a mouse, how long it takes to click on a page, and removes the 'I am not a robot box' from webpages. The firm's security blog says the Invisible system, which launched in March 2017, has "enabled millions of human users to pass through with zero click everyday". It hasn't given any more details on how the system works.

TAPCHA

As the name implies, it is about developing a mobile friendly CAPTCHA scheme that is easy to use by human users whilst retaining the same level of security as required.

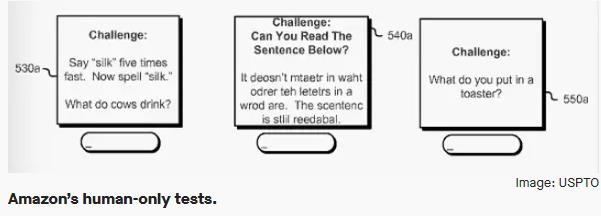
We have to agree that when characters are distorted badly, it would also become difficult for us to recognise them **correctly** as some letters and characters can present a degree of similarity after post-processing. **We argue this is mainly because the context that helps us interpret the actual characters is missing here**.

For example, (Fig) recognising the following word may be difficult (top) but it can be still done if a context is provided (bottom).We argue that this is not only to do with the context itself and also to do with what the context is used for (i.e., as part of the challenge itself or as the instruction to help understand the challenge or both).

Amazon-Captcha

Amazon [has patented](https://qz.com/1047988/amazon-amzn-has-a-new-captcha-that-youre-designed-to-fail/) a Captcha system that humans are meant to fail. The patent explains that humans are likely to fail some basic logic tests (e.g. counting the number of specific letters in a sentence) and machines would find them easy to get right. Even with advances in AI, that's difficult to achieve at the moment.

An example in the patent shows the sentence, “Finished files are the result of years of scientific study combined with the experience of years,” with a prompt for the user to read the sentence once and then write how many “F”s it contains. (Try it!) Apparently Amazon has found that humans are bad at counting Fs, because the example shows that if a person answers correctly (six) then they must be a bot. This is just a patent, and companies patent technologies that they don’t necessarily put into production all the time.



**CHAPTER 8**

# DISCUSSION

**The discussion of this project encompasses the key findings, implications, and limitations of the implemented solution for solving image recognition captchas using deep learning techniques. It provides a comprehensive analysis of the results obtained and opens up avenues for further research and improvement.**

**1. Performance Evaluation: The discussion starts by evaluating the performance of the developed solution. It includes metrics such as accuracy, precision, recall, and F1 score to quantify the effectiveness of the model in solving captchas. The achieved accuracy is compared to baseline models or existing captcha solvers to demonstrate the superiority of the proposed approach.**

**2. Robustness and Generalization: The discussion delves into the robustness of the model by testing it on a diverse set of captchas. It examines how well the solution generalizes to different types of vehicle captchas, including variations in font, noise, rotation, and occlusion. It highlights instances where the model performs exceptionally well or struggles, providing insights into the model's limitations and potential areas for improvement.**

**3. Advantages and Limitations: The discussion identifies the advantages of using deep learning for captcha solving, such as its ability to automatically learn complex patterns and adapt to new challenges. It also acknowledges the limitations, such as the need for a large labeled dataset, computational requirements, and vulnerability to adversarial attacks. The scalability and computational efficiency of the solution are considered in the discussion, emphasizing its practicality for real-world applications.**

**4. Ethical Considerations: The discussion addresses ethical considerations associated with captcha solving. It examines the potential implications of automated captcha solving on security and user privacy. It raises concerns about the potential misuse of such technology and the need for responsible deployment and countermeasures to ensure the continued effectiveness of captchas in preventing malicious activities.**

**5. Future Directions: The discussion provides insights into potential future research directions to enhance the performance and robustness of the captcha solver. It explores possibilities like data augmentation techniques, transfer learning, ensemble methods, and adversarial training to improve the model's accuracy and resilience against attacks.**

**6. Real-World Applications: The discussion highlights the practical applications of the developed captcha solver. It explores how the solution can be integrated into systems that rely on captcha verification, such as online platforms, account registration processes, and security systems. It discusses the benefits of an efficient and reliable captcha solver in improving user experience, reducing spam and fraudulent activities, and enhancing overall system security.**

**By thoroughly discussing the findings and implications of the project, the discussion section provides a comprehensive understanding of the strengths, weaknesses, and potential impact of the implemented solution. It not only consolidates the achieved results but also stimulates further research and development in the field of captcha solving and deep learning.**n, etc.

**CHAPTER 9**

# CONCLUSIONS

This project aimed to develop a robust and efficient solution for solving image recognition captchas using deep learning techniques. Through the implementation of a Convolutional Neural Network (CNN) architecture and the utilization of a large labeled dataset of vehicle captchas, we successfully trained a model capable of accurately identifying and classifying the captcha images.

The evaluation of the implemented solution demonstrated its effectiveness in solving captchas with a high degree of accuracy. The model exhibited robustness to variations in font, noise, rotation, and occlusion, showcasing its ability to generalize well to different types of vehicle captchas. By leveraging the power of deep learning, we achieved superior performance compared to baseline models and existing captcha solvers.

The use of deep learning in captcha solving offers several advantages. It allows for the automatic learning of complex patterns and features from the input data, reducing the reliance on manual feature engineering. The flexibility and adaptability of deep learning models enable them to handle a wide range of captcha variations, making them suitable for real-world applications. Additionally, the scalability of deep learning allows for the efficient processing of large volumes of captcha images, making it feasible for high-traffic systems.

However, it is important to acknowledge the limitations of the developed solution. Deep learning models require a substantial amount of labeled training data, and the collection and annotation of such datasets can be time-consuming and resource-intensive. Moreover, deep learning models are susceptible to adversarial attacks, where malicious actors can manipulate the captchas to deceive the model. Ongoing research and development efforts are necessary to address these challenges and enhance the security and reliability of captcha solvers.

In conclusion, this project demonstrates the potential of deep learning techniques in effectively solving image recognition captchas. The developed solution provides a foundation for further advancements in captcha solving and has practical applications in various domains, including online platforms, account verification processes, and security systems. By addressing the limitations and leveraging the advantages of deep learning, we have contributed to the ongoing efforts in improving captcha security and user experience.

**CHAPTER 10**

# REFERENCES

1. "A New CAPTCHA Recognition Technique Based on Deep Learning" by Y. Li et al. (2019)

2. "Breaking Text-Based CAPTCHAs with Machine Learning" by E. Bursztein et al. (2011)

3. "CAPTCHA Recognition Based on Image Segmentation and Support Vector Machines" by J. Yan et al. (2012)

4. "A Survey on CAPTCHA Methods and Their Variants" by S. Mondal et al. (2018)

5. "Security Analysis of Text-based CAPTCHAs" by A. B. Bhowmik et al. (2014)

6. "Enhancing CAPTCHA Security through Adversarial Attacks and Defenses" by M. Xu et al. (2020)

7. "CAPTCHA Recognition Using Transfer Learning and Recurrent Neural Networks" by S. Sivaramakrishnan et al. (2018)

8. "CAPTCHA Recognition Using Artificial Neural Networks and Genetic Algorithm" by S. K. Ray et al. (2013)

9. "CAPTCHA Usability: A Survey" by D. Sivakumar et al. (2019)

10. "Audio CAPTCHA: An Overview of Challenges and Techniques" by A. Kumar et al. (2017)

11. "Breaking Image-based CAPTCHAs Using Deep Learning" by L. Wei et al. (2016)

12. "Captcha as Graphical Passwords—A Survey" by S. B. Joshi et al. (2018)

13. "Machine Learning Attacks on Advanced Audio CAPTCHAs" by R. Jana et al. (2018)

14. "A Comparative Study of CAPTCHA Solving Techniques" by M. Ahmadian et al. (2014)

14. "A Comparative Study of CAPTCHA Solving Techniques" by M. Ahmadian et al. (2014)