Contents

Chapter 1

Introduction

A CAPTCHA or 'Completely Automated Public Turing test to tell Computers and Humans Apart' is a test which is used, as the name suggests, to determine whether or not a user is human. The main application of a CAPTCHA is to ensure that malicious bots are not being used in order to access a website for any given reason, such as spam. A commonly used form of CAPTCHAs are text-based CAPTCHAs, where an image is provided to the user, containing letters, numbers, or a combination of the two. These characters are then distorted and overlapped, making them difficult to read, in order to ensure that bots cannot access the page that has been secured using the text-based CAPTCHA.

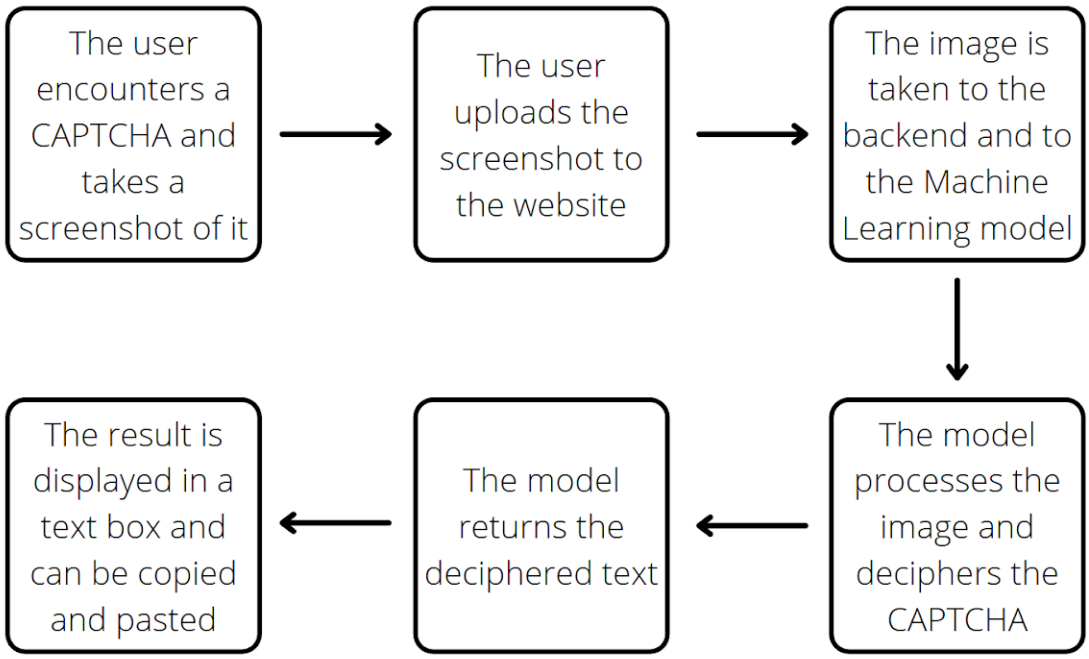
1.1 Problem Statement

The problem statement for this project is to create an application that can solve text-based CAPTCHAs using Machine Learning techniques. The reason text-based CAPTCHAs have been chosen for this project is because, despite being widely used in sectors such as banking or e-ticket booking, they are often inaccessible for people with visual impairments, reading disabilities, or people who do not speak English[1]. Hence, there is a need to create an application that can be used in order to help these people access sites that use CAPTCHAs for security.

1.2 Approach

1.2.1 Application

In order to use the application to solve a given text-based CAPTCHA, the user needs to take a screenshot of the web page that has presented them with the CAPTCHA, and upload this screenshot to the web application. This image is taken to the server side, where it is pre-processed and fed to the Machine Learning model. The model processes this image, breaks the CAPTCHA, and returns the decoded text to the user via a text box, from where they can easily copy the text, and paste it where required. The following box diagram is a graphical representation of the aforementioned procedure.



1.2.2 Machine Learning Model

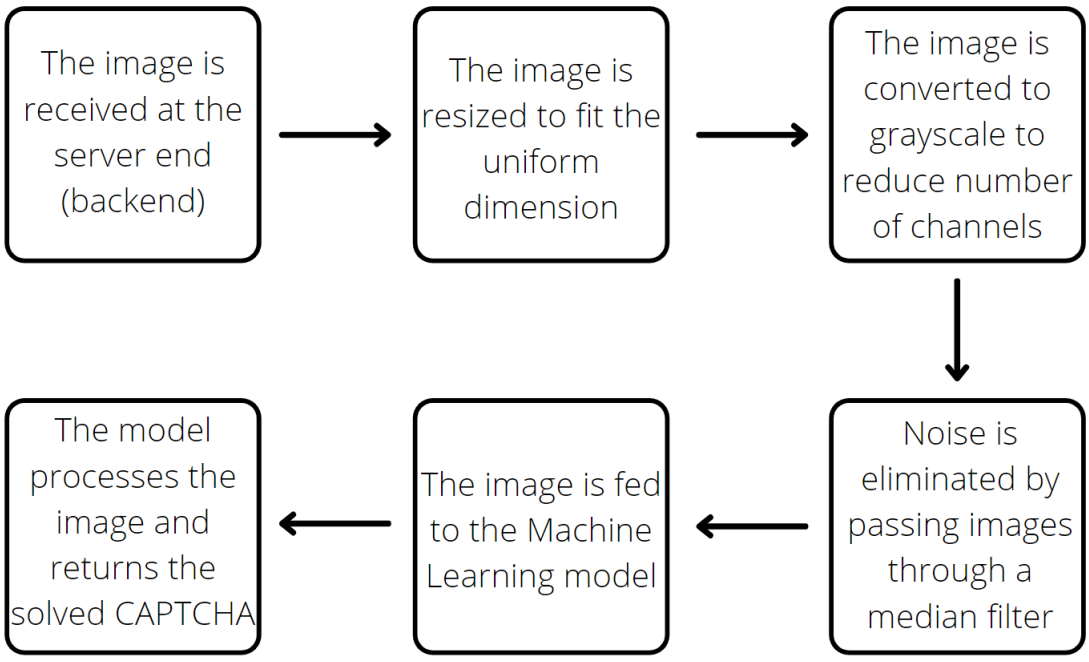
For the pre-processing of the images, they must first be scaled to uniform dimensions (height and width), and then converted to grayscale format in

order to reduce the number of channels from three to one. In order to remove noise in the image, a noise reduction algorithm or a median filter[2] may be used.

The basic structure of the model is as follows:

1. Convolutional-Maxpool Layer
2. VGG-19 Layer
3. Dense Layer
4. Recurrent Neural Network Layer
5. Softmax Layer

This may be subject to change, depending on how heavy the final model is.



Chapter 2

Use Cases

On average, a person takes approximately 10 seconds to solve a text-based CAPTCHA. However, many people are unable to do this for various reasons, and are thus dependent on others in order to perform sensitive tasks such as online banking. This includes people with visual and learning disabilities[3], as well as non-English speakers. These situations have been explored below.

2.1 Visually Impaired People

Text-based CAPTCHAs require the person to be able to see the distorted characters on their screen, interpret and decipher these characters, and return the characters within a given amount of time in order to verify that they are human. However, a person with visual impairments cannot perform this task exclusively on their own. While they may be capable of reading plain text, it can be greatly difficult for them to read distorted text. In this case, an application that solves the text-based CAPTCHA for them would assist them while also ensuring their independence.

2.2 People with Reading Disabilities

People who suffer from dyslexia, for example, find it difficult to read text as it is. For them to read distorted characters on their screen is a virtually impossible task. However, if they had an application that returned the correct

text to them, they could simply copy that text, paste it where they need to enter it, and be able to access the desired website.

2.3 Non-English Speakers

Most of the time, the characters in text-based CAPTCHAs tend to be letters of the English alphabet. A non-English speaker would not be able to differentiate these characters easily. In this case, an application to solve the CAPTCHA for them would be of great use.

Chapter 3

Novelty

The use of Machine Learning in order to solve CAPTCHAs is a heavily explored domain. Despite this, there are certain areas that previous projects have been unable to cover, which this project aims to tackle.

3.1 Competition

As mentioned previously, there have been several attempts in the past to solve CAPTCHAs using Machine Learning. There have also been products created specifically to assist the visually impaired access websites. Some of the prominent competitors have been discussed below.

3.1.1 CAPTCHA Be Gone

CAPTCHA Be Gone is a series of web extensions available as a service in order to solve different types of CAPTCHAs, including text-based CAPTCHAs. According to their website, they detect CAPTCHAs on web pages, solve them, and copy the result to the user's clipboard with the press of a "single keystroke". However, CAPTCHA Be Gone appears to have been inactive since approximately 2017. It is also worth noting that their code is not open source, so they do not elaborate on how the user's privacy is maintained, despite clearly mentioning that on their website. The service is also currently restricted exclusively to the Firefox, Chrome, and Internet Explorer browsers[4].

3.1.2 CAPTCHAsolutions

CAPTCHAsolutions is a Chrome Browser Extension used in order to solve reCAPTCHA v2. However, it was last updated in 2018, and their website appears to have security issues, as the Mozilla Firefox Browser throws a 'Potential Security Risk Ahead' warning while attempting to access the website[5].

3.1.3 Others

While other services to solve CAPTCHAs do exist, these are either limited to a specific platform, such as Chrome or Firefox, or they do not deal with text-based CAPTCHAs.

3.2 Project Novelty

The following sections explain why this project is unique.

3.2.1 Not Platform Specific

As the implementation of the project is in the form of either a desktop or mobile application, there is no need for the user to ensure that they are using a specific browser to access the desired website. They can make use of any platform they like and simply feed the CAPTCHA data into the application, externally.

3.2.2 Variable Text Lengths

So far, text-based CAPTCHA solvers have restricted themselves to a specified number of characters when solving the CAPTCHA. With this project, the aim is to be able to incorporate anywhere between four and six characters when attempting to find a solution for a given text-based CAPTCHA.

Chapter 4

Literature Survey and Related Work

4.1 Deep-CAPTCHA

The authors of the paper entitled 'Deep-CAPTCHA: a deep learning based CAPTCHA solver for vulnerability assessment'[2] developed a Convolutional Neural Network model in order to investigate the weaknesses and vulnerabilities of 5-letter text-based CAPTCHA. The model was trained on a dataset of 500,000 CAPTCHAs and the final test accuracy attained for the alphanumeric dataset was 98.31%. To achieve this, the authors began by normalising the images by first reducing the dimensions of each image to half the original size, and then converting the images to grayscale format and passing them through a median filter to remove noise.

The model consisted of three Convolutional Neural Network blocks, where each block had 1 convolutional layer, 1 ReLU activation function, 5x5 kernels and a 2x2 Max-Pooling layer. A flatten function was used, followed by a dense layer, and finally a Softmax layer. The model employed Binary Cross-Entropy Loss as the loss function, and Adam as the optimiser. Analysis of the model showed that common errors included "1" and "7", "w" and "m", and "g" and "8". However, the paper does not work with variable character recognition and has classified that under potential future work.

4.2 Neural Network CAPTCHA Cracker

The authors of the paper entitled 'Neural Network CAPTCHA Cracker'[6] have used Convolutional Neural Networks and Recurrent Neural Networks, along with the ReLU activation function, with the aim of creating a model that is capable of breaking image-based CAPTCHAs. In order to create a synthetic dataset, a Java module was used to generate a string of 4 to 7 characters, and add noise, for each CAPTCHA. The final size of the dataset used by the authors was around 13 Million images, all with the same dimensions. For variable length CAPTCHAs, this model achieved an accuracy of 80%. The Convolutional Neural Network was used in order to learn the features of the image, while the Recurrent Neural Network was used to output the character sequence.

Prior to feeding the images to the model, they were converted to grayscale using the Python Imaging Library. This paper involves the use of two models, one making use of Long Short-Term Memory (LSTM) cells, and the other using multiple Softmax layers. The main blocks of the the model are the Convolutional Neural Network, which consists of 2 Convolutional-Maxpool layers, followed by a dense layer, and a dropout layer, and Recurrent Neural Network, which has an LSTM of size 256, whose training was sped up via Gradient Clipping. The approach in this paper was not impacted by the fragility inherent in attacks while manually cleaning or segmenting an image, and lead them to the finding that neural networks can learn to perform complicated tasks such as the simultaneous localization and segmentation of ordered sequences of objects.

Chapter 5

Future Work

This project deals exclusively with text-based CAPTCHAs, and is targeted towards those with visual impairments, reading disabilities, and non-English speakers. For future implementation, these functionalities could be extended to those with hearing disabilities, in order to help them use audio CAPTCHAs. This will continue to be helpful for people who do not speak English as non-English audio CAPTCHAs are extremely rare.

Bibliography

[1] S. Hollier, J. Sajka, J. White, and M. Cooper, "Inaccessibility of captcha," 2019.

[2] Z. Noury and M. Rezaei, "Deep-captcha: a deep learning based captcha solver for vulnerability assessment," 2020.

[3] Moreno, Lourdes, González-García, María, Martinez, and Paloma, "Captcha and accessibility. is this the best we can do?," 2014.

[4] "Captcha be gone," 2016.

[5] "Captchasolutions," 2018.

[6] G. Garg, "Neural network captcha cracker," 2015.