John Haag,

Zach Kovalenko,

Thomas Shear

Artificial Intelligence Project proposal

Google’s CAPTCHA has become a staple part of life on the internet. Nearly every webpage utilizes it when asking for users to sign up for services. This tool is designed to prevent robots (bots) from signing up for fake accounts within legitimate services. A large example of Captcha in use, is in the online game RuneScape. This game has an economy that has grown into real world money. There is an advantage to having multiple accounts because someone could sell the things they get in-game to real world players for real world money. The play of the game itself is highly repeatable and consists mostly of mouse interaction on a top down interface. Many tasks in the game are automatable, but against terms of service, if a bot farm was able to create accounts and play the game then the owner of the farm could stand to make good money in the real world. Google’s CAPTCHA is part of the effort to restrict bot’s access to the game. Google’s CAPTCHA is used in many other fields as well. Another important example of this is online shopping websites especially ones that deal with highly sought-after collectible items. Certain clothing brands such as Supreme, Adidas, Nike, Bape and others release items with a very limited supply. Many people use bots in order to make their checkout of the product a lot faster than any person on just a normal computer possibly could. Which then leads to people with bots taking all of these items and reselling them online at a large markup. And, the actual collectors are usually not able to get these items unless they have bots. Another prominent example of Google CAPTCHA shining is in review websites. These websites look to prevent bots from creating illegitimate product reviews designed to artificially inflate a products review score and make the product look better than it is in reality. In recent years, CAPTCHA has grown in difficulty as machine learning algorithms have advanced to the point that they are better at image recognition then most humans. Google has changed the CAPTCHA algorithm to utilize less image and sound detection and more activity, history, and interaction detection. These include monitoring the mouse for regular movement or jumping from place to place. Along with seeing if the browser has activity on other websites to show that it is detecting human interaction and not just a server farm running bots. We are looking to create a bot/algorithm that is capable of defeating Google’s CAPTCHA.

To do this we must develop an application capable of detecting the location of a Google CAPTCHA user verification tool on any website. Then we must be able to trick the verification tool into believing that we are a real person instead of automated software that is trying to get past the verification tool. To do this, we must create a program capable of recognizing the box that defines that classic “I’m not a robot” checkbox. That can be accomplished through utilizing image recognition algorithms to find what the box looks like on the screen. After careful research and discussion with the group members of this project, we have decided to do this using TensorFlow. Which is an Artificial Intelligence framework which uses the Python programming language to execute programs. Our research shows that Google tests the movement of the mouse towards the checkbox and around the page. We must mimic actual human movements with the mouse and not have precise movements or jump the mouse from place to place. We also learned that Google checks the browser’s cookie database and other attributes to assist with determining if the user is human. We must utilize browsers that have been used by actual humans or if being used on a new browser we must first simulate legitimate traffic to other websites to build a human like history. The newer CAPTCHAs monitor user activity before and after the checkbox is clicked. We must maintain our human like emulations throughout our visit to the target page. A success will be defined as having moved the mouse over the CAPTCHA and clicking the checkbox. If the checkbox turns to a green checkmark then we have succeeded. If the CAPTCHA asks for any additional information, we consider this to be a failure. These are all things that our group might look further into upon successful completion of beating Captcha version 1 which only intakes a mouse click from the user.

The purpose of this Artificial intelligence project is to design and implement Artificial intelligence practices that can be recognizable and repeatable to beat the Google Captcha. Our experiment begins with downloading TensorFlow framework onto the base computer. TensorFlow has a multitude of different uses. In the case of our project we will use TensorFlow to create an object detection model. A successful Object detection model requires a dataset from which TensorFlow can detect objects. An object can be anything the user decides they want to build an AI project with, it may be a deck of cards, or a table in a room. The main object that TensorFlow will be detecting in our project will be a picture or computer screenshots of the Google Captcha Touring test. The Experiment will contain at minimum 100 pictures or screenshots of different types of Google Captcha’s with different backgrounds that will make object detection more difficult for the computer. It is important to make the object detection difficult for the computer to detect, in order for the model to be robust enough to operate in any circumstance. In order to create the best model possible our experiment will have a training set, a test set and potential for using cross-validation. The training set will have pictures of the Google Captcha, a Google Chrome icon, and an exit button at the top of the browser. These training images will be roughly 60% of the total number of images we have for each object we are detecting. Once our model has been trained to detect the Captcha and the group is confident it works, we must next determine whether we have more than one algorithm or model. In the case that we only have one algorithm, our group will proceed to the testing phase of the project. However, if we have more than one algorithm, we will use cross-validation to create the best fit model and algorithm. This cross-validation set will compare the performances of the training models. The labeling of each object will determine whether the program is executed as it was intended. This particular part of the experiment may be one of the most important, as without the correct labeling the program will fail to execute the way it was intended to. The testing phase we will use the model that we believe to be the best fit, we have trained the model and now it is time to release it into the real world. This part of the experiment we as people will find webpages that have a Captcha and run our Object detection model on that page to see whether we have correctly trained our model. Once the Object detection model has been thoroughly tested and continues to work, we will have to write a virus using the Python Programming Language. This virus will use the object detection model, and simulated mouse or cursor events in order to click the checkbox of the Captcha, and the exit button respectively.

In order to ensure this project is repeatable to prove time and again the accuracy and effectiveness of our method some transparency is required. The user of our program must be using the same operating system, browser, and potentially the same or similar hardware IE Computer or GPU (Graphical Processing Unit). The operating system our program will be executed within is Windows 10, and the browser only fitting for this project is Google Chrome. Google Chrome was chosen for this project due to the compatibility with Captcha as both are created by Google. Another important note, is that other browsers trying to run the same program may not work due to a lack of downloaded drivers on the user’s computer. The hardware although still important may be the one constant that is semi-flexible, due to the volume of pictures trained in our dataset an Nvidia GPU is encouraged but not required. The GPU will take significantly less time to compute than a regular CPU (Central Processing Unit). Of the constants listed above only a few things will be changing. These variables are, the ability to change the size of the Captcha box, as well as the background colors and design of the browser. This will ensure that our program can recognize the Google CAPTCHA anywhere on the screen, no matter the size of the screen or size of the Captcha.

We will obtain our Dataset by capturing screenshots of the Google CAPTCHA in multiple different settings. We will also get screen shots off of the internet and any possible other sources that we may find while working on our project.

The timeline for our project will be as follows. The project will be split up into five main tasks. Our first task will be to complete the project proposal. This will be done during the first two weeks of the project. Starting September 28th, 2020 and finishing it up by October 12th, 2020. Our second task will be to setup a server so that we can have the Google CAPTCHA to practice on in the future. Task two should not be a difficult one so we would like to begin working on this task on October 12th, 2020 and have this complete by October 19th, 2020. Our third task will be to gather data for our TensorFlow object detection. So that we can start training it to recognize the Google CAPTCHA box. By data we mean screen shots of the Google CAPTCHA in as many possible scenarios that we can think of. As a part of this task we will also be training our object detection model. We will begin working on task three by October 19th, 2020 and we are looking to have it complete by November 9th, 2020. We are planning on working on this part for three weeks to ensure that our object detection is working as flawlessly as possible. Our fourth task will be to create a virus to click the Google CAPTCHA checkmark box in order to tell the computer that a human is clicking the box. We will begin working on task number four by November 9th, 2020 and we are going to have it complete by November 23rd, 2020. We are leaving two weeks to work on this task to ensure that our virus is working properly and, in the event, that any problems arise while working with Python we will make sure they are solved. As we are all new to the Python programming language. Our fifth task will be ensuring that everything is working correctly and creating the presentation. The fifth task will begin on November 23rd, 2020 and be complete by November 30th, 2020. We aim to complete the project before December 1st, 2020. As we are not sure on the exact day that we will need to have the presentation complete. If there ends up being more time to work on the project, we will use the week of November 30th, 2020 to December 7th, 2020 to ensure that everything is working correctly and to make our presentation as accurate as possible. The presentation that we will be working on will include each step of the project including the proposal, data gathering, object detection training model, the virus creation process, and a demonstration of the final product in different settings.

Below are examples of the Captcha’s we will use for our dataset.



