# SteinsCNN: Bypassing image based captcha using CNN

**P17: Akhil Namboodiri, Karan Pradeep Gala, Sahil Santosh Sawant**
Department of Computer Science
North Carolina State University
Raleigh 27606
anamboo@ncsu.edu, kgala2@ncsu.edu, ssawant2@ncsu.edu

## 1 Background

**CAPTCHA** stands for "Completely Automated Public Turing Test to Tell Computers and Humans Apart." CAPTCHA is used to protect websites from bots. This is accomplished by developing an assessment comprised of various CAPTCHAs that are easily passed by humans but are inaccessible to current computer programs. As a result, after grading these tests, we can prevent automated bot attacks. CAPTCHA is used to prevent network traffic generated by bots from causing a denial of service. CAPTCHAs are broadly classified into five types: text, image, audio, video, and puzzle.

Image-based captchas are a widely used technique for user authentication. It is important to know if these images can be recognized and bypassed, causing the increase in image-based captcha detection bots. They use visual content rather than textual representations of frequently occurring elements such as bridges, cars, crosswalks, and so on. Effectively, the user is expected to judge the theme and select images that match the provided keyword or those that do not. The experiment is to determine how closely we can achieve a model that can detect the images that are observed in these image-based captchas.

## 2 Proposed method

### 2.1 Intuition

The base model that we created was the starting point of our experiment. Once we achieved the baseline, we devised different approaches in order to analyze the different effects of it towards the accuracy of the model. Digital Image Processing was utilized in order to understand whether the use of image processing method could help the model to learn better, as we sharpened the edges and thus helped distinguish the objects in the image much clearer. Principal Component Analysis (PCA) was used in order to understand the training time, as PCA compressed the image, it would definitely make the training time for the model significantly less, due to the less dimensionality. K fold Cross Validation was used to make the model robust and less susceptible to overfitting by using all of the input images in both training and validation. If we were measuring accuracy, having k different accuracy results where all of the data was used in the test phase would always be better and more reliable than having one accuracy result produced by a train-test split where all of the data was not used in the test phase.

# 3 Plan and experiment

## 3.1 Dataset

The proposed model makes use of **deathlyface's reCaptcha dataset**(3). The dataset is divided into **12 classes** and contains **11,774 images**. One of these classes is labeled "others," which does not contribute to categorizing the specific images that we seek. As a result, we decide to disregard the class and work with a total of **11 classes: Car, Crosswalk, Bus, Hydrant, Palm, Traffic Light, Bicycle, Bridge, Motorcycle, Chimney, Mountain.** We also notice that the dataset contains an inconsistent number of data points for each class, which varies widely.

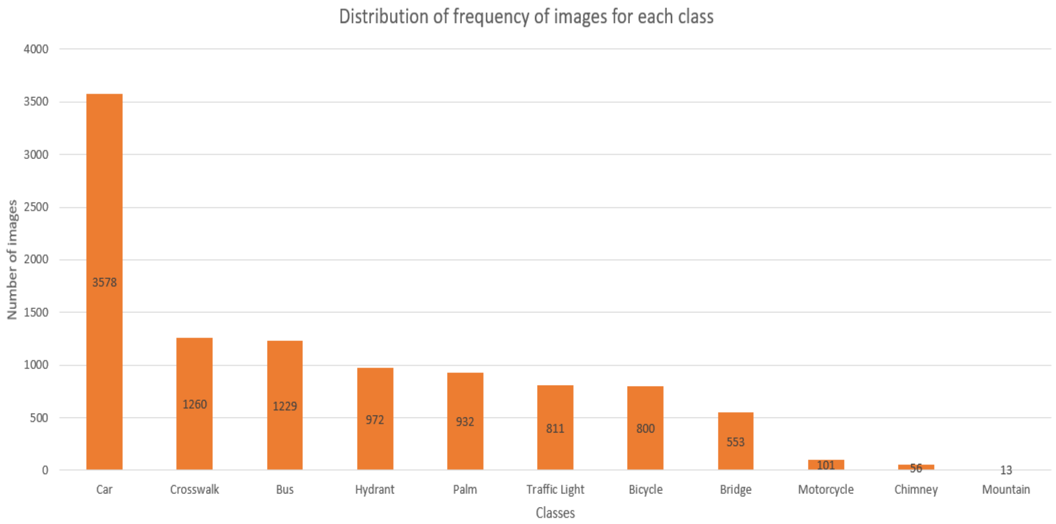

Figure 1: Distribution of frequency of images for each class

## 3.2 Pre-processing of the dataset:

The images are first converted into an array using the asarray() function. All the images in the dataset are then processed at **96x96 pixel** resolution. The image labels are then applied to the images.

While creating models for training the data, additional image processing techniques have been employed as a part of the pre-processing stage which includes **image sharpening** and **contrast enhancement**. Image sharpening is a technique which specifically targets to fine tune the details of the provided image while highlighting its edges. Contrast enhancement as the name suggests aims at improving the overall visibility of the image by adjusting the corresponding darkness and brightness tone.

**Data augmentation** carried out at the pre-processing stage also helped increase the training data necessary for improving the robustness of our model. The different preprocessing techniques that were used include **image Rotation, image zoom, image shifting, image flipping**. These image processing techniques have helped to improve the prediction in a model as it becomes more accurate in recognizing samples the model has never seen before. There is enough data for the model to understand and train all the available parameters.

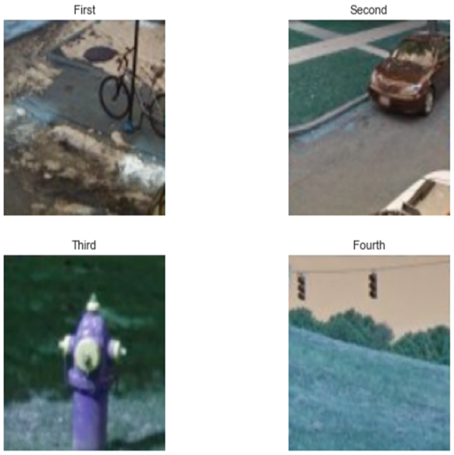

Figure 2: Before preprocessing

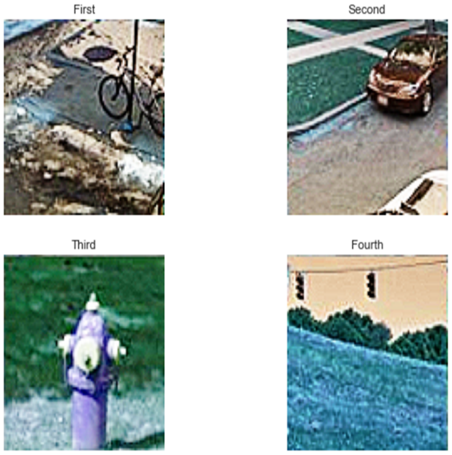

Figure 3: After preprocessing

### 3.3 Details of experiment

#### 3.3.1 Base Model

We developed a model that takes an image of 96x96 pixels as input and then employs a convolutional neural network with five convolutional and MaxPooling layers. This is followed by a couple of flatten layers and a couple of dense layers. This model will undergo further refinements in the following steps during the refinement process. For the activation, we have used ReLU. The model that was developed is as follows:

#### 3.3.2 Model 2: Base Model with enhancement using Digital Image Processing

Image enhancement was performed on the input images as part of the preprocessing techniques. The two main techniques used were contrast enhancement and image sharpening.

**Contrast enhancement:** Performed histogram equalization using the input image's intensity distribution as a basis and enhanced contrast to distinguish between darker and lighter image portions.

**Sharpening:** Used a sharpening filter kernel [[0,-1,0], [-1,5,-1], [0,-1,0]] over the input image to sharpen, to make it look crisp and clear by enhancing the edges of objects in the image.

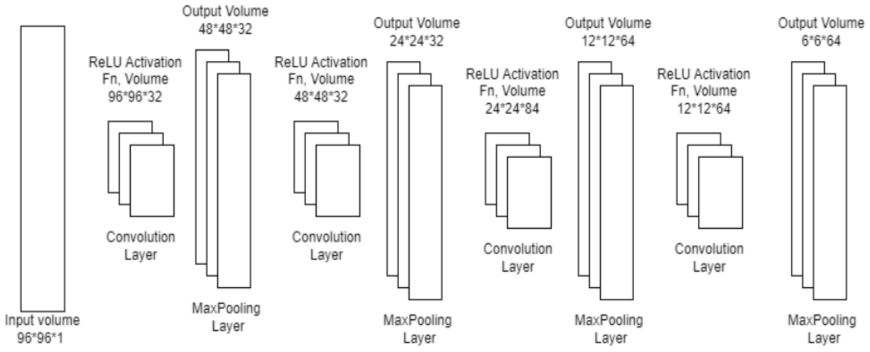

Figure 4: Base Model architecture

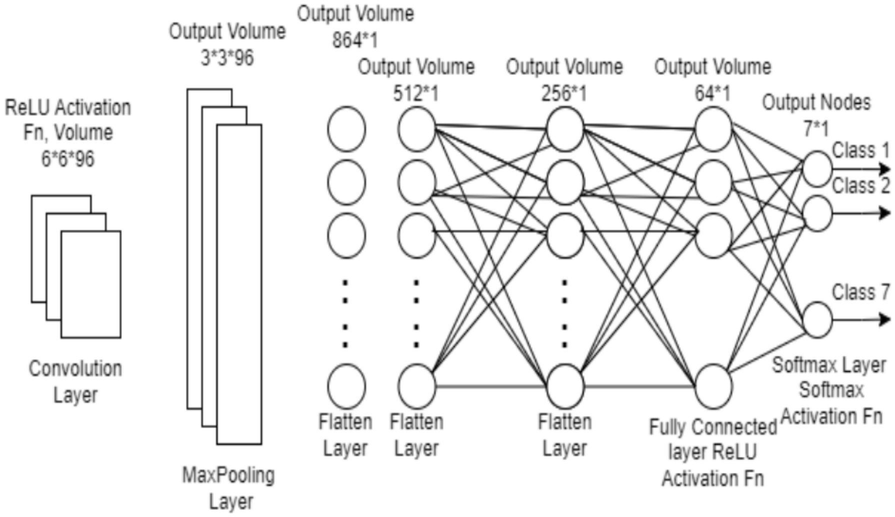

Figure 5: Base Model architecture

#### 3.3.3 Model 3: Enhancement using Principal Component Analysis

Principal Component Analysis (PCA) was the third technique which we used as a preprocessing technique. It takes in the image and treats the columns as attributes. As the PCA is applied and the columns are reduced, we get a compressed form of image, while retaining a great amount of information in it. Thus, we achieve a smaller dimension of image from the start while retaining most information, hence the training time is improved cause of processing of fewer dimensions.

#### 3.3.4 Model 4: Enhancement using K-fold cross validation

For our fourth model we have applied 3 folds cross validation on the same base CNN model, so that we can compare accuracies of different techniques of model evaluation. The k-fold CV ensures that all the input images are used in both training and validation, thereby making the model robust and less susceptible to overfitting.

#### 3.3.5 Selection of Optimizer:

We tested the model with several optimizers, including Stochastic Gradient Descent, Adam, and Adagrad. We found **Adam Optimizer** to produce the most stable results, so we decided to use it for the model.

For loss function, we compared KL Divergence along with Categorical Crossentropy and found that there was more consistently decreasing loss calculated in **Categorical Crossentropy** and decided to use it for the base model.

We have split the dataset into train and test as **80-20** and employed **Stratified Sampling** to ensure that the representative sample from the population is divided into relatively similar sub-populations.

## 4 Results

We have understood that, to get the best result while considering as many classes as we can, we take classes while getting a training accuracy of and a testing accuracy of. For now, we have understood it to be the best case, but we will be carrying out further design analysis and refining for better performance.

For model 1, which was our base model, after running for 50 epochs, we got the following results:

- Architecture: **Convolutional Neural Network**
- Model Parameters: **7,130,156**
- Training Accuracy : **85.41%**
- Test Accuracy: **79.86%**
- Training Time: **1022 seconds**

For our model 2, we obtained the following results after improving the input with preprocessing techniques such as contrast enhancement and image sharpening:

- **Digital Image Processing with Contrast Enhancement:**
  1. Train Accuracy: **86.43%**
  2. Test Accuracy: **80.75%**
  3. Validation Accuracy: **80.75%**

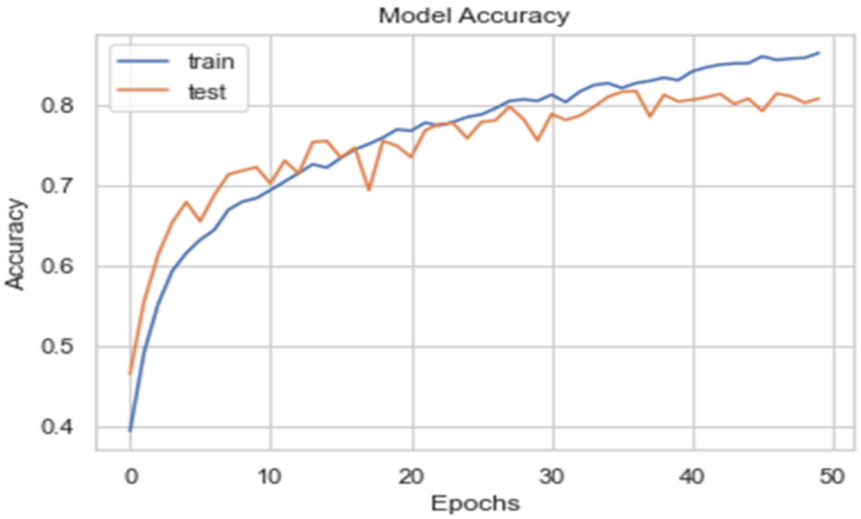

Figure 6: With contrast enhancement only

- **Digital Image processsing with Contrast enhancement and image sharpening:**
  1. Train Accuracy: **86.02%**
  2. Test Accuracy: **76.18%**
  3. Validation Accuracy: **76.11%**

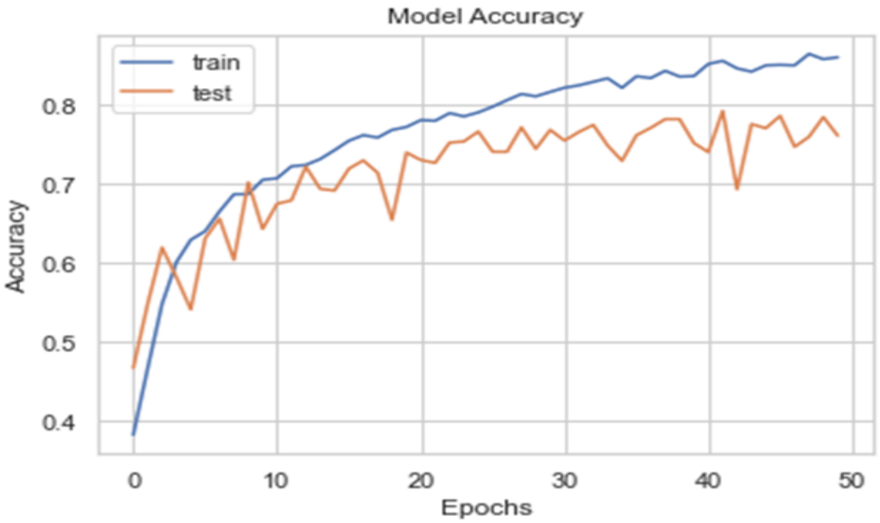

Figure 7: With contrast enhancement and image sharpening

For our third model, we have performed performance enhancement using **Principal Component Analysis (PCA)**. The use of PCA has proven useful in compressing images and analyzing model train time while sacrificing some accuracy.

- Time Taken: **964 seconds**
- Number of components: **12**
- Train Accuracy: **83.68%**
- Test Accuracy: **80.28%**

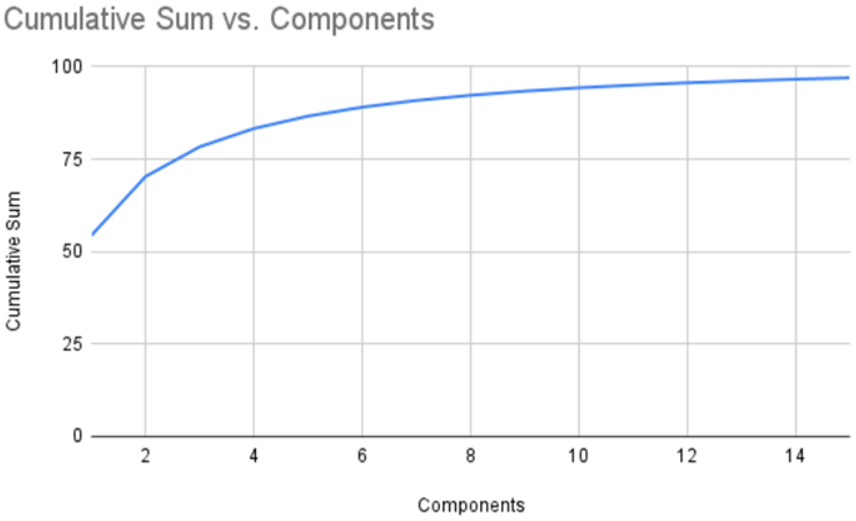

Figure 8: Cumulative sum vs components

For model 4, we have applied 3 folds cross validation on the same base CNN model so that we can compare the accuracy of different techniques of model evaluation. So that all the input images are used in both training and validation, we have used a k-fold CV to make the model robust and less susceptible to overfitting.

- **3 fold CV with Contrast Enhancement:**
  1. Train Accuracy: **91.67%**
  2. Test Accuracy: **91.3%**
  3. Validation Accuracy: **95.3%**
- **3 fold CV with Contrast enhancement and image sharpening:**
  1. Train Accuracy: **90.24%**
  2. Test Accuracy: **89.9%**
  3. Validation Accuracy: **89.87%**

Combining results from all the four proposed models we get,

| Model | Train Accuracy | Testing Accuracy |
|---|---|---|
| Model 1 | 85.41% | 79.86% |
| Model 2 | 86.43% | 80.75% |
| Model 3 | 83.68% | 80.28% |
| Model 4 | **91.67%** | **91.3%** |

Among the other approaches we devised was **Model 4**, which used 3-fold cross validation with contrast enhancement.

## 5 Observations:

- We observe that the train time for the model in case of PCA is reduced by **5.675%**, while sacrificing **2.02%** of accuracy in training and **0.5%** accuracy in testing
- The use of only **contrast enhancement** gives better results as compared to when it is combined with image sharpening.
- When compared to all other approaches tested, item **3-fold cross-validation** yielded the best results.

## 6 Conclusion

- The model showed improvements with use of **3-fold cross validation** along with preprocessing techniques like contrast enhancement and image sharpening.
- PCA does show **improvement in the training time** of model. Hence it can be used in cases with images having too much components, where slight accuracy loss for faster performance is more important.
- We conclude that **K-Fold Cross Validation** in training a CNN model as it helps in training multiple model and averages out with the best weights for the task.
- Using **contrast enhancement** is a better approach than adding image sharpening along with it.

## 7 Github link

The Github link for our project. https://github.ncsu.edu/kgala2/engr-ALDA-Fall2022-P17

## References

[1] Y. Shu and Y. Xu, "End-to-End Captcha Recognition Using Deep CNN-RNN Network," 2019 IEEE 3rd Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), 2019, pp. 54-58, doi: 10.1109/IMCEC46724.2019.8983895.

[2] Saini, Baljit Singh, and Anju Bala. "A review of bot protection using CAPTCHA for web security." IOSR Journal of Computer Engineering 8.6 (2013): 36-42.

[3] https://github.com/brian-the-dev/recaptcha-dataset

[4] D. Wang, M. Moh and T. -S. Moh, "Using Deep Learning to Solve Google reCAPTCHA v2's Image Challenges," 2020 14th International Conference on Ubiquitous Information Management and Communication (IMCOM), 2020, pp. 1-5, doi: 10.1109/IMCOM48794.2020.9001774.

[5] Tam, Jennifer, et al. "Breaking audio captchas." Advances in Neural Information Processing Systems 21 (2008).