Chandini, Joselyne, Joel

CS 360 Final Project

05.11.2024

# The Impact of Noise in Captcha Through Deep Learning

## Motivation, Question, and Hypothesis

The existence of captcha is one of the pillars upholding modern cyber security. With its role in preventing mal-use of AI and other softwares to  crash websites, flood emails, and cause a lot of other potential cyber mayhem. With the further development of AI and machine learning technologies becoming more accessible, there are concerns on how well what security we do have can be upheld against new potential cyber threats.

Our group wanted to investigate how the amount of noise in a captcha image could impact the accuracy derived from running them through neural networks. Looking at how well neural networks perform, we can get insight into how imminent a threat AI is to the security captcha provides. We hypothesized that the accuracy will decrease for the dataset with more noise. The RGB data is a little more inconsistent pixel to pixel, and there are also case differences between letters in the same captcha string, so it's likely the machine will find it harder to be more accurate when run on the dataset with more noise.

## Data and MachineTraining

We used two datasets for this project; a classic captcha image dataset with numbers and letters and a captcha where each image has more noise (different colors, letter cases, and randomly placed dots).

We used deep learning neural networks, specifically a Convolutional Neural Network (CNN) which is a neural network that applies a set of learnable filters to the input. Each filter in a layer captures specific features from the input image. We first encoded the images from the dataset, using an algorithm that sections off the pixels where the meat of the captcha is on each image and encodes the RGB values from there. After that, we trained it on the CNN model, setting the encoded data into corresponding keys from a dictionary of all available/corresponding characters. After training, we ran on the test set and created a confusion matrix that showed us how well the model performed.

**Results and Interpretations**

After running the model, we got the following results:

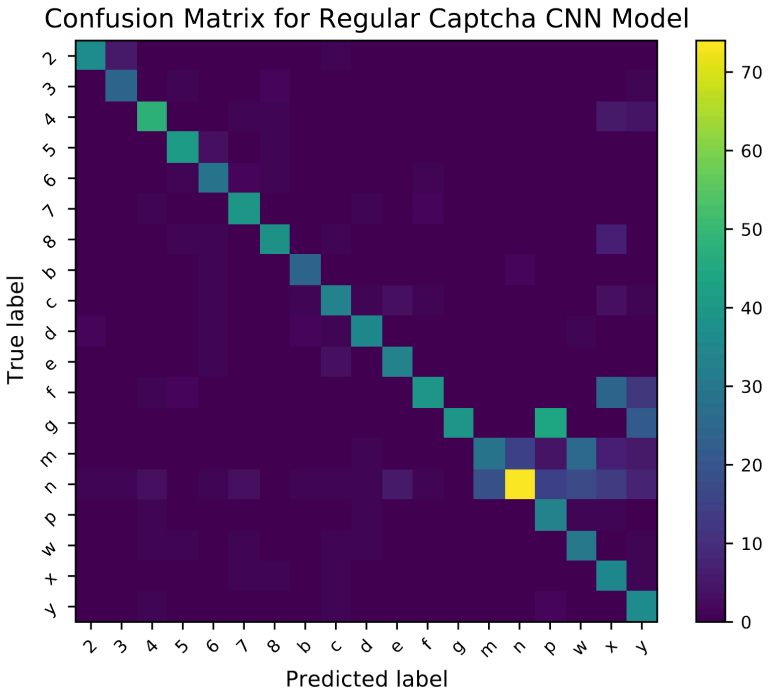


**Figure 1** Confusion matrix on no-noise captcha images

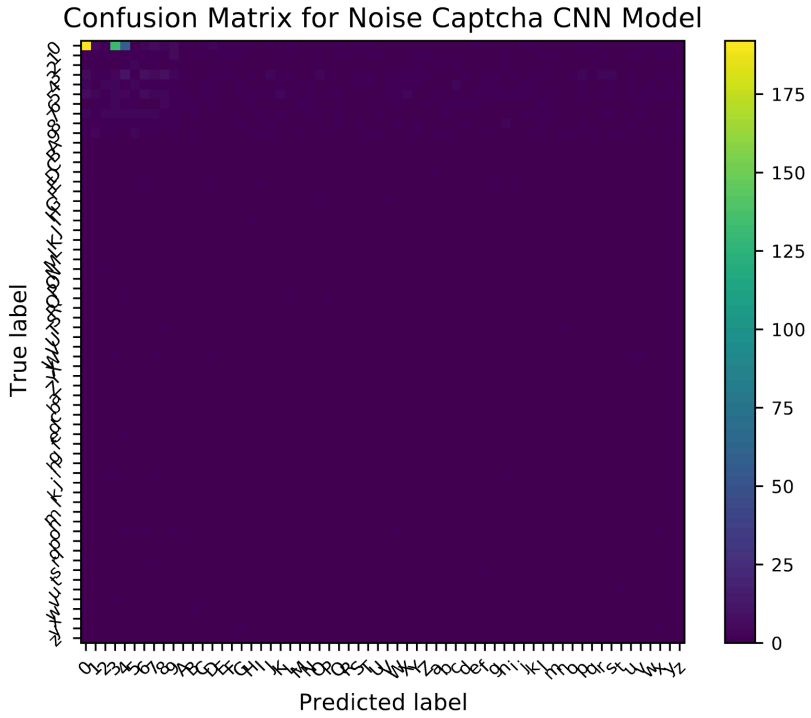


**Figure 2** Confusion matrix on captcha images with noise

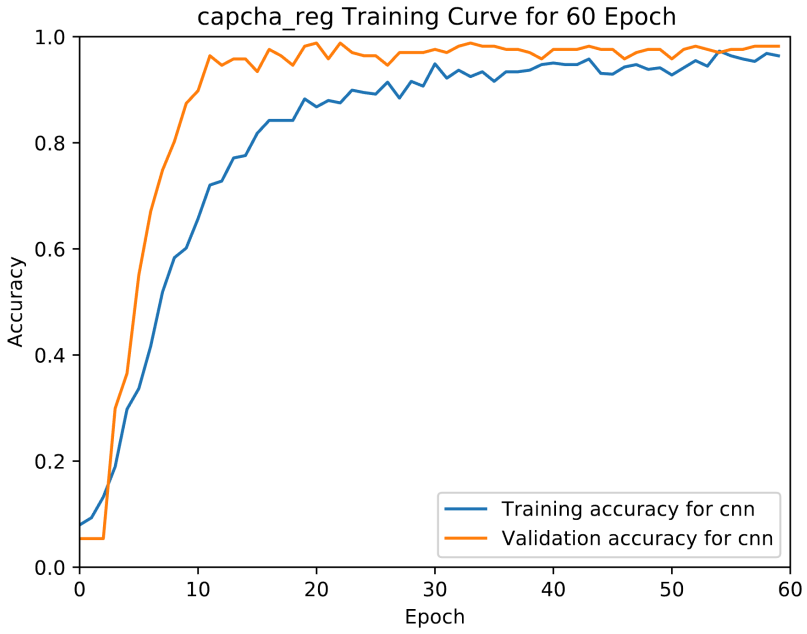


**Figure 3** Training curve on no-noise captcha images\

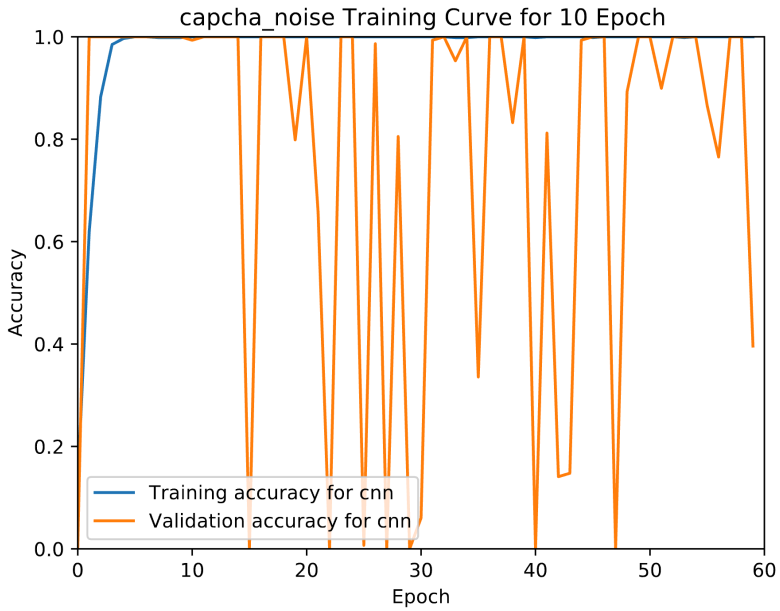


**Figure 4** Training curve on captcha images with noise

As you can see from the confusion matrices and training curves above, the model predicts the noiseless dataset with much more accuracy than the one with noise. It could be the case that our model struggled more with the varying letter cases, but the variety in RGB values, both because of the noise and the different colors of each image, also very well could have contributed heavily to the low accuracy we see. The high noise made correlations harder to make, which could explain the high fluctuation of the training rate in Figure 4 as well.

## Conclusions and Future Work

In future work we might generate entire captcha solutions at once instead of sequentially (likely with RNN). Another area for improvement would be to train an algorithm to section off each character's pixels for the data preprocessing rather than taking in every pixel for each

example. It would be nice to also categorize the data somehow to see how different types of noise affect performance, but it will be hard to split up the data when all data is rgb.