Deceiving computers in Reverse Turing Test through Deep Learning

Jimut Bahan Pal [1]

Under the Guidance Of

Dripta Maharaj [2]

[1]Department of Computer Science
[2]Department of Mathematical Science
Ramakrishna Mission Vivekananda Educational and Research Institute
Howrah - 711202

June 22, 2020

Table of contents I

Table of contents II

Table of contents III

Table of contents IV

- What is a Turing test?
  - Alan Turing in 1950, devised a method to tell humans and computers apart.
  - Used Imitation game.
  - Imitation game - Played with 3 people: a man (A), a woman (B) and an interrogator (C)
  - Job of interrogator is to determine which one is a man and woman, via question answer through text.
  - In Turing test, the interrogator needs to differentiate between a Human and a Computer.
  - Think of it as a chatbot.

- What is a reverse Turing test?
  - The roles of Humans and Computer are interchanged.
  - Computers (Servers) determines whether a user is a bot or human.
  - Specially used for stopping bots which waste resources by creating spamming accounts.
  - Also used to stop scraping bots.

Introduction

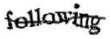
Motivation for breaking CAPTCHA

- A common method to perform Reverse Turing Test is to use CAPTCHA.
  - CAPTCHA is an acronym for **C**ompletely **A**utomated **P**ublic **T**uring test to tell **C**omputers and **H**umans **A**part.
- Automated scripts should not be more successful than 1 in 10,000 (0.01%) for any type of attack, while the humans success rate should approach 90%.
- We have worked on cracking a common type of CAPTCHA - Text CAPTCHA (most of them with over 99% accuracy on test set).
- The Optical Character Recognition (OCR) task was the backbone for building up CNN (during Yann LeCun's time).
- Text based CAPTCHAs are becoming hard for humans day by day, and hackers are enhancing the Deep Learning Technologies to solve more difficult CAPTCHAs.
- We wanted to show how easy it was to crack text based CAPTCHAs, and the measures one can take to improve the Reverse Turing Test system.

- There are a various types of CAPTCHAs.
- Text based Simple CAPTCHAs (From Wikipedia and JCAPTCHA)

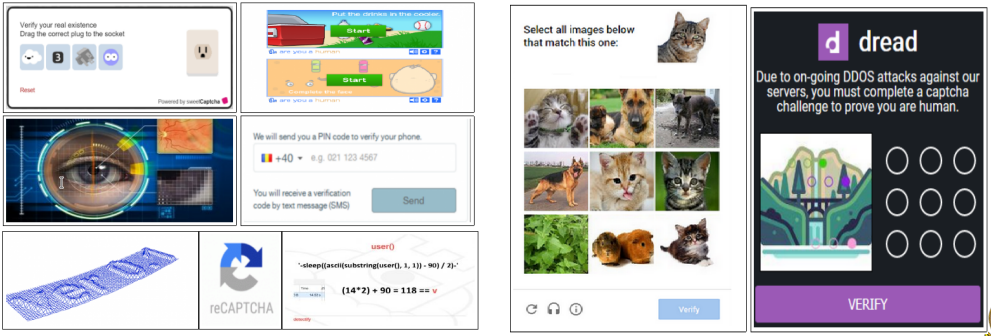
- Text based CAPTCHAs (From JCAPTCHA and Pixprofit)

- Modern Text CAPTCHAs (From Wikipedia and PayPal)

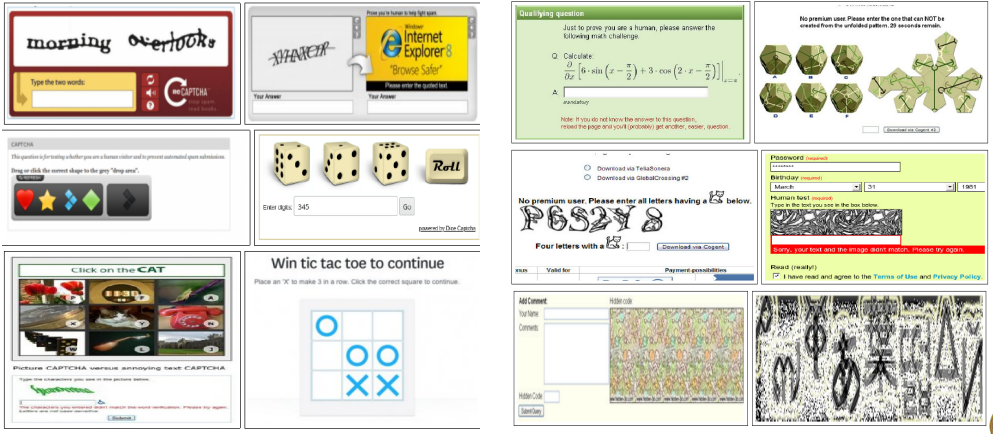


- More modern types of Text CAPTCHAs

- There are other types of CAPTCHA as well (Picture Courtesey: IIHglobaland Wikipedia)
- Sweet CAPTCHA – That lets you match easy images, instead of Twisted Texts, Playthru CAPTCHA replaces plain text via microgames, Text Verification CAPTCHA helps to get genuine users without pain, Google's reCAPTCHA uses cookies, picture selection CAPTCHAs etc



- Mini Games are usually a better choice for CAPTCHAs and are trending now-a-days
- Sometimes there can be very hard CAPTCHAs as well as shown in right



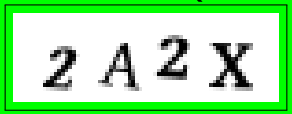
Coming up: Our Work

Our Work

Overview

- We have collected various types of CAPTCHAs and have broken them with human level accuracy.
- We have used various methods such as k-NN, CNN and mainly Deep Learning Techniques to break the CAPTCHAs.
- When we have used open source data generators, we have the freedom to increase the amount of data, and with huge data our results were state of the art!
- We have also found certain conclusions on how we can improve the text based CAPTCHAs.
- Our main intension was to prove that text based CAPTCHAs are not as difficult as before to crack, and with Deep Learning techniques, we can easily crack text based CAPTCHAs.
- We have uploaded the data generated in this project to figshare, hence other could use them, and used Google's free Colab GPUs for training.

CAPTCHA (4 letter)



CAPTCHA (4 letter)

About the data

- Original dataset was found on Kaggle (`https://www.kaggle.com/genesis16/captcha-4-letter`) and the cleaned version can be found here (`https://jimut123.github.io/blogs/CAPTCHA/data/captcha_4_letter.tar.gz`) (Hosted on my website).
- Contains about 9955 PNG files.
- Total size is about 10.4 MB.
- The images have 3 channels (RGB) and of dimension 72x24.
- From Figure 1 we can see that 0,1,I,O are missing, since that can be confusing for humans sometimes.

CAPTCHA (4 letter)

- Distribution of letters in the CAPTCHA (4 letter) dataset:

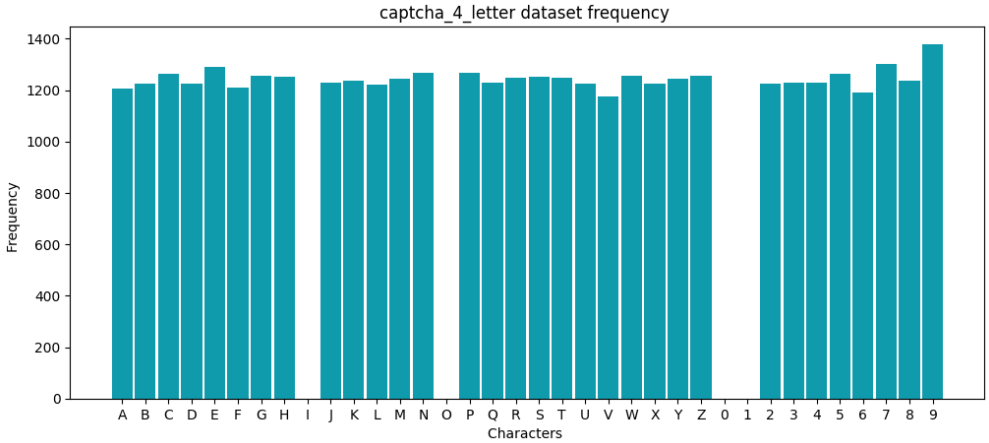


Figure: The initial distribution of the letters for the Captcha (4 letter) dataset.

CAPTCHA (4 letter)

A Naive Model

- We have first done some preprocessing on the data to feed into our model.
- Performed adaptive thresholding of binary inverse.

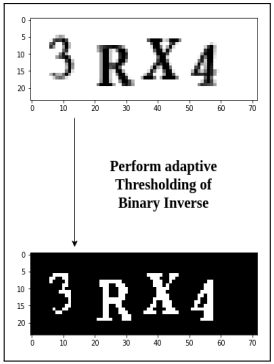


Figure: Performing adaptive thresholding of Binary Inverse before passing it to the model.

- The structure of the naive model.

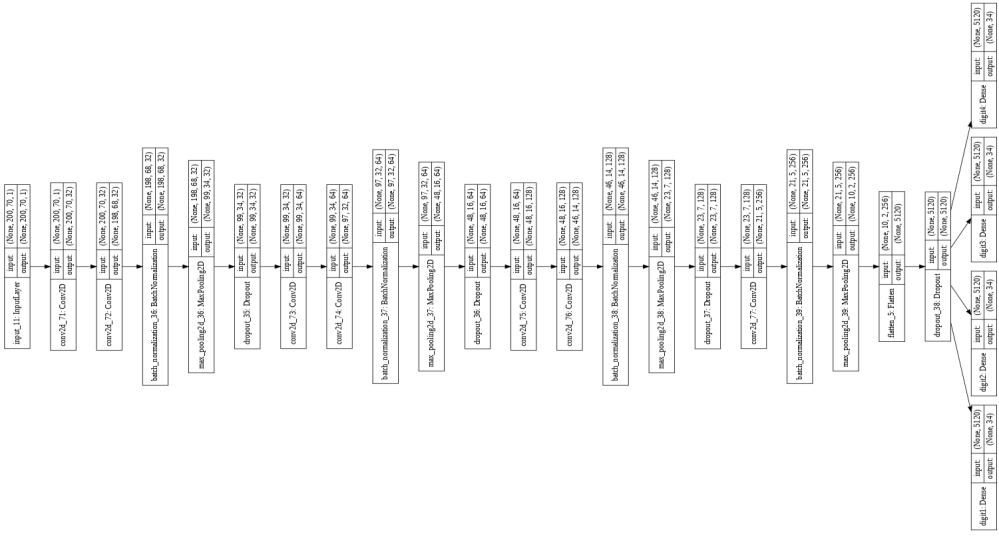
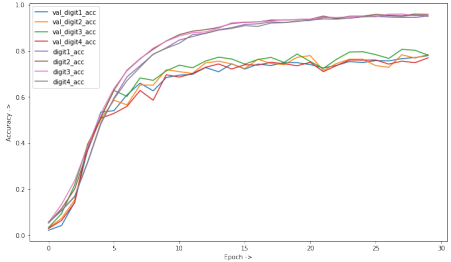
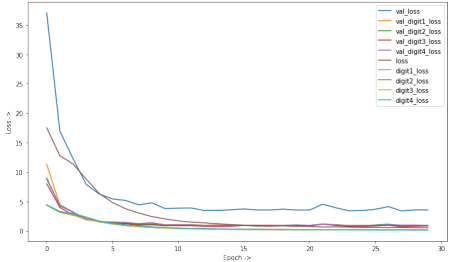


Figure: Naive model for predicting the labels of the 4 letter CAPTCHA dataset.

- The graph of accuracy and loss obtained from each layer of the model.



Accuracy



Loss

- Trained for 30 epochs (9 second per epochs), 4.5 minutes to train the whole dataset.
- 77 % accuracy on test dataset.

- Some visualisations of the layers.

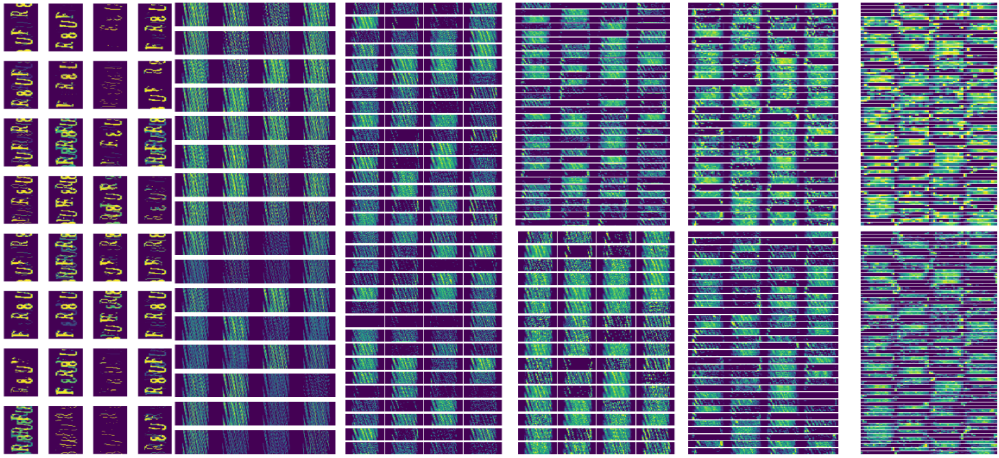


Figure: The visualization obtained when a CAPTCHA of **ufr8** is passed through the model.

- The prediction on unseen data.

Figure: Prediction on some of the sample of the test set.

- 3x3 Filters learned by the first layer of this model.

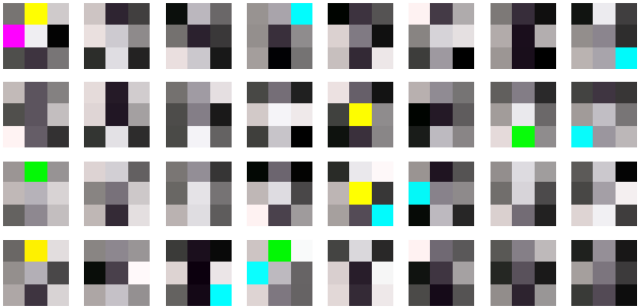


Figure: The 3x3 kernels (filter) learned by the model for layer 1.

- Weights learned by the layers of this model.

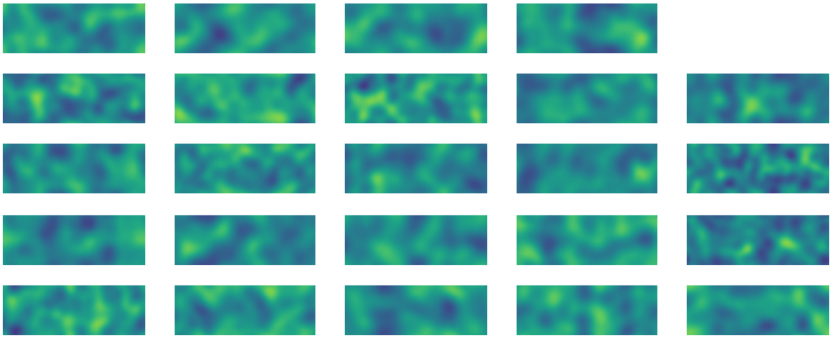
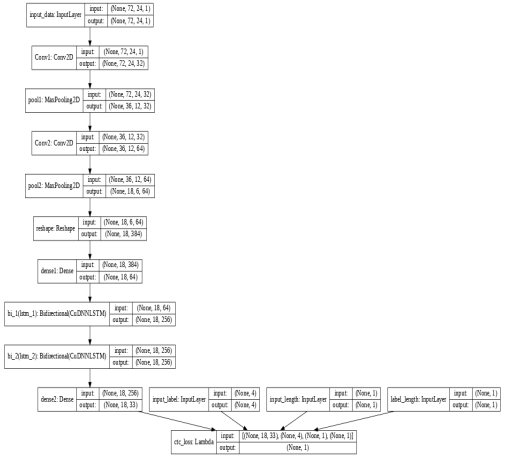


Figure: The weights learned by the model for layers 1 to 24.

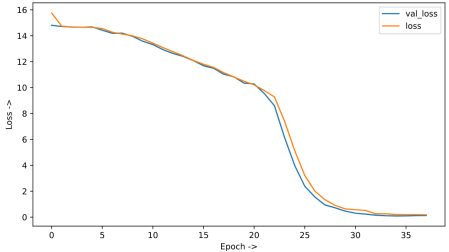
CAPTCHA (4 letter)

A better model

- The structure of the better model.



- The graph of loss obtained from each layer of the model.



Loss

- A total of 38 epochs, with a total time of 4.43 minutes to finish the whole training.
- 99.87 % accuracy on test dataset.

- The prediction on unseen data.

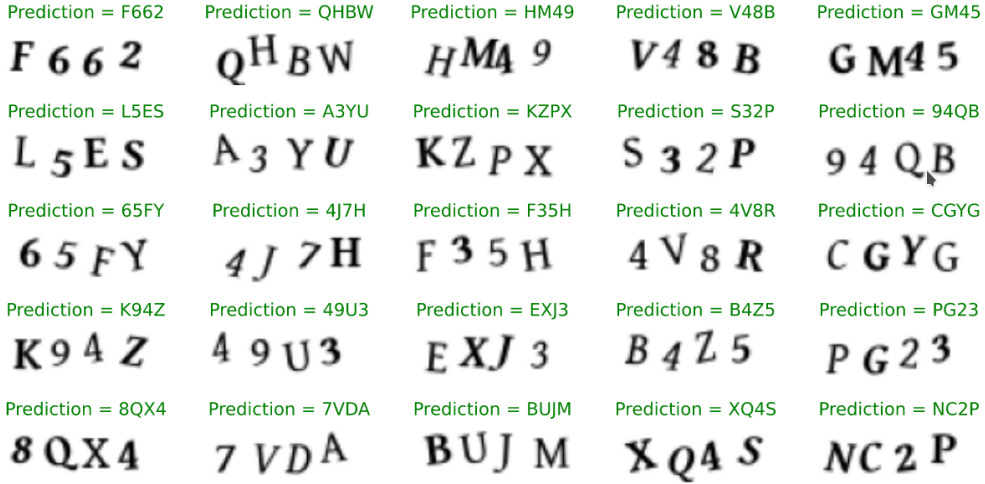
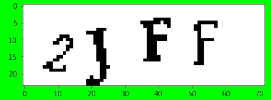


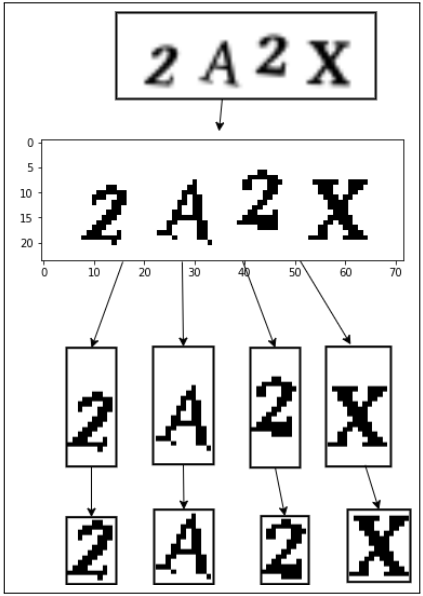
Figure: Prediction on some of the sample of the test set.

c4l_16x16_550 dataset



- The separable characters from **CAPTCHA (4 letter)** were extracted to get this dataset, which helped in better classification of the **CAPTCHA (4 letter)** dataset.
- The dataset contains 32 characters, dimension 16x16, 550 images from each character.
- Total size is about 1.4 MB.
- The images have are mainly black and white images (having intensity of 0 and 255 only).
- From Figure 9 we can see that 0,1,I,O are missing, since that can be confusing for humans sometimes. This is a proper subset of **CAPTCHA (4 letter)** dataset.

- The basic procedure to extract the images



- Distribution of letters in the CAPTCHA (4 letter) dataset:

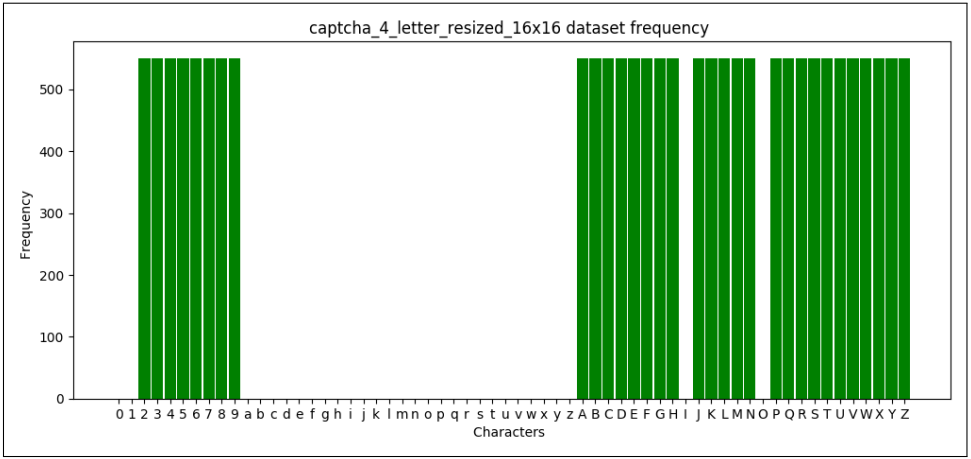
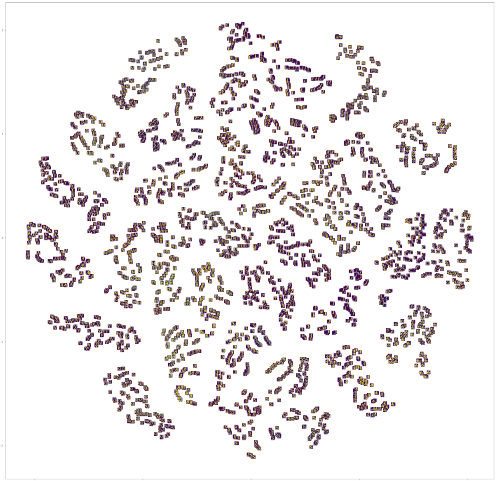
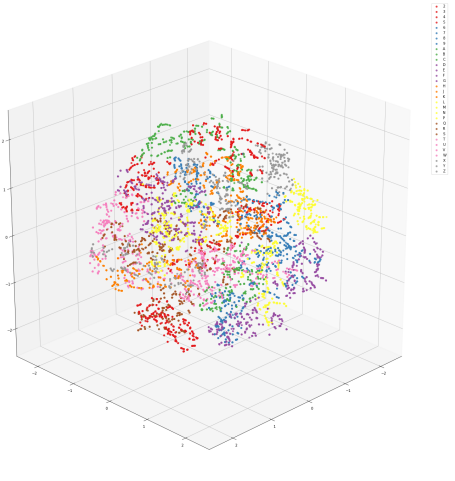


Figure: Frequency of characters obtained from c4l-16x16_550 dataset.

- The t-SNE plots of the letters of the c4l_16x16_550 data



In 2D



In 3D

- The structure of the CIFAR-10 like model for predicting the individual characters of the c4l_16x16_550 dataset.

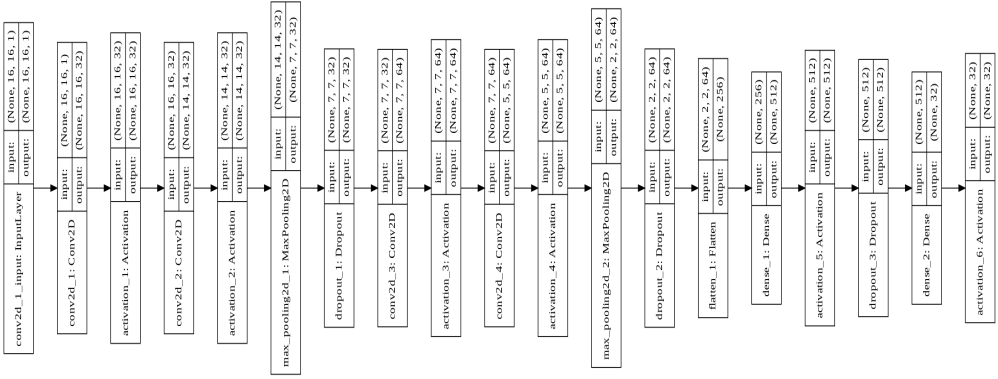
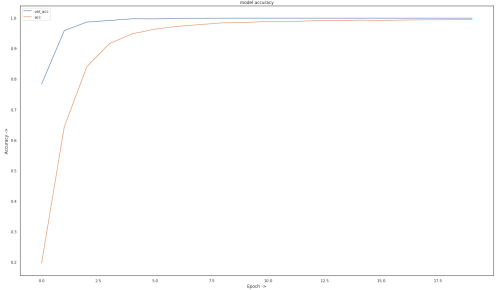
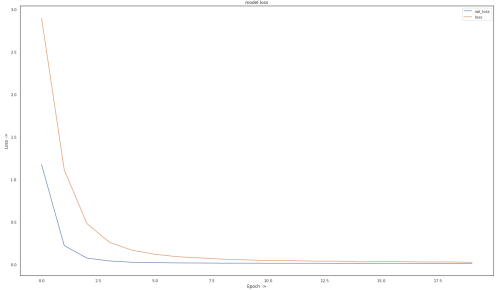


Figure: CIFAR-10 like model for predicting the classes of c4l_16x16_550 dataset.

- The graph of accuracy and loss obtained from each layer of the model.



Accuracy



Loss

- Trained for 20 epochs, 0.6 minutes to train the whole dataset.
- 99.91 % accuracy on test dataset.
- Some visualisations of the layers are shown in the following slides.

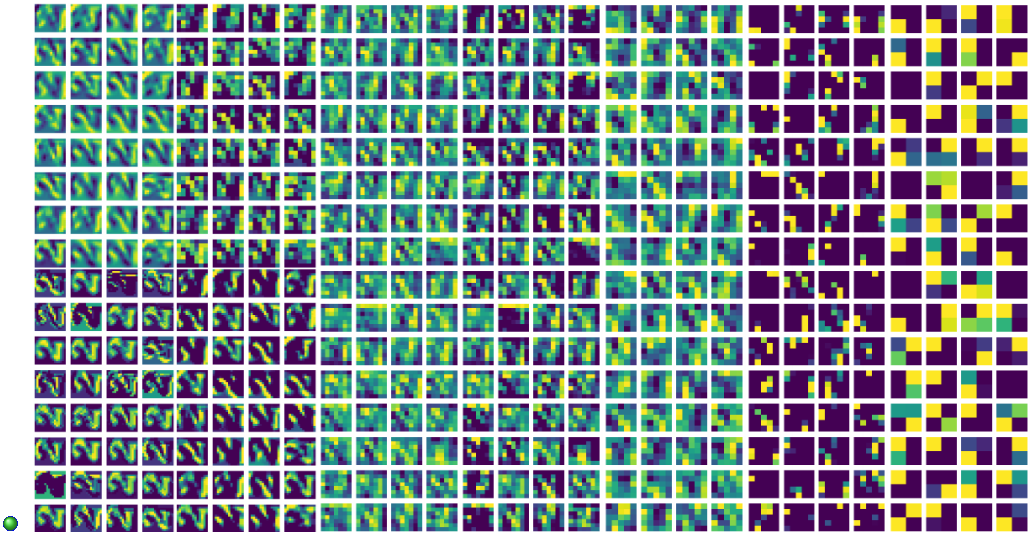


Figure: The visualization obtained when an image of **2** is passed through the model.

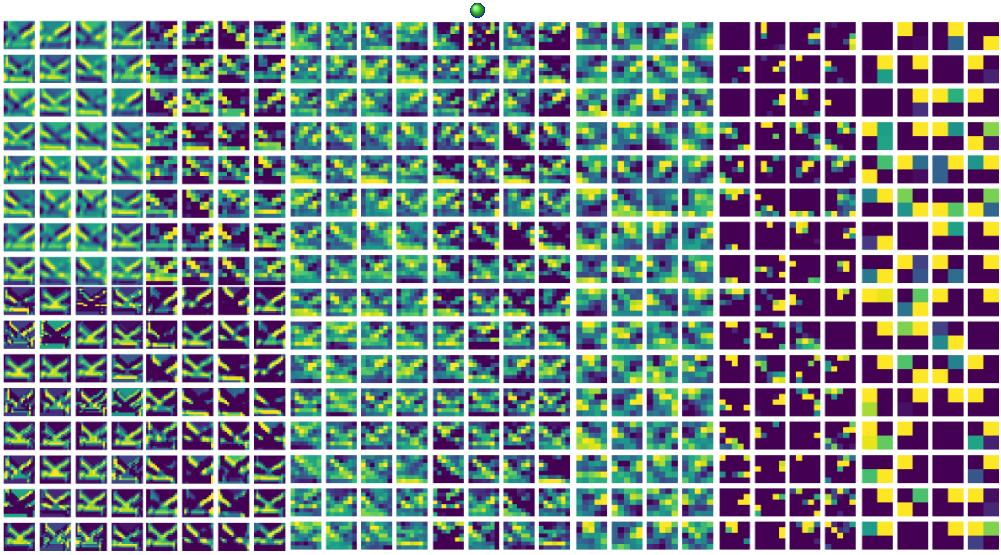


Figure: The visualization obtained when an image of **k** is passed through the model.

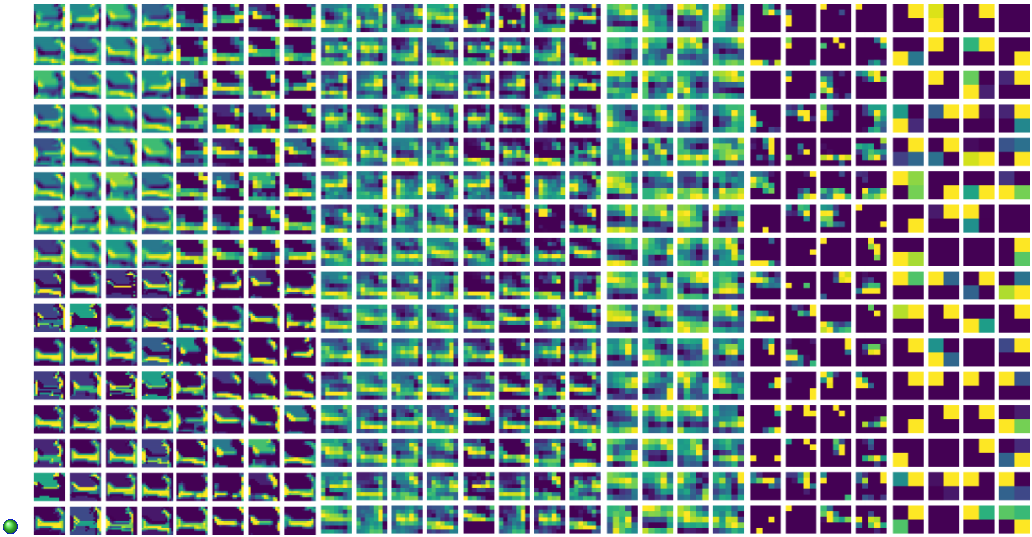


Figure: The visualization obtained when an image of **L** is passed through the model.

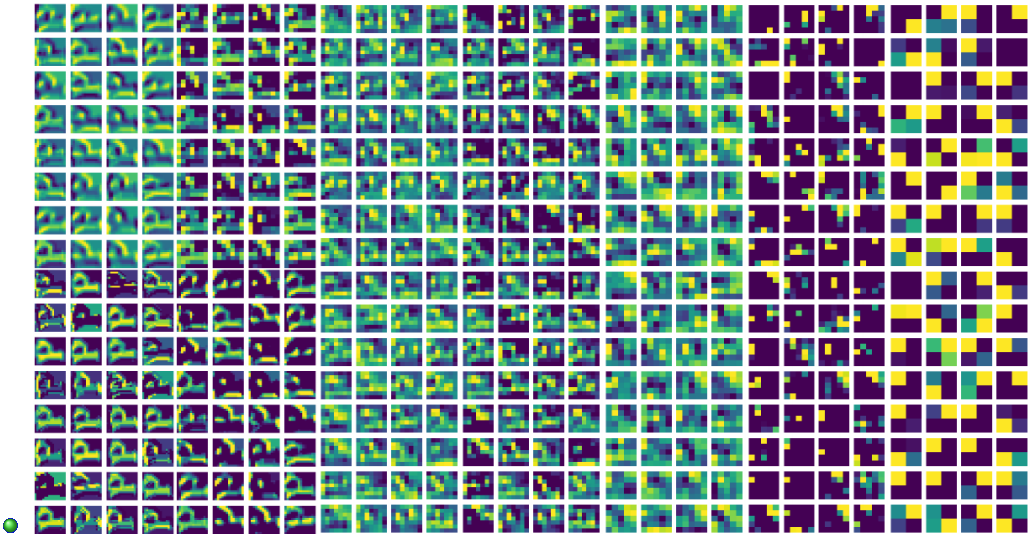
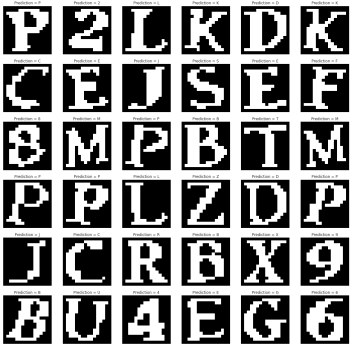


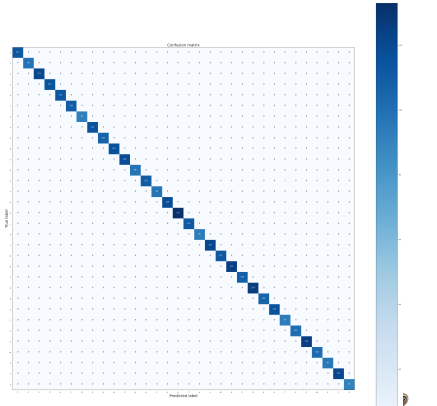
Figure: The visualization obtained when an image of **p** is passed through the model.

c4l_16x16_550

- Performance of the model on unseen data.



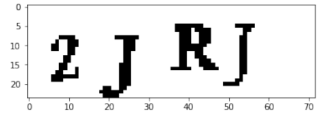
Prediction on unseen test data by the model.



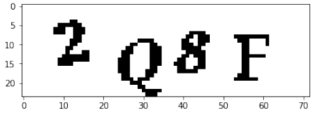
Confusion Matrix Obtained

- Some of the wrong predictions of the model when used to predict some subset of **CAPTCHA (4 letter)** dataset.

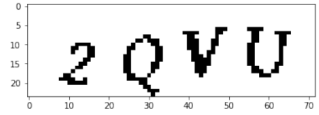
2JFF classified as 2AFF



2JRJ classified as 2KRJ



2Q8F classified as 2A8F

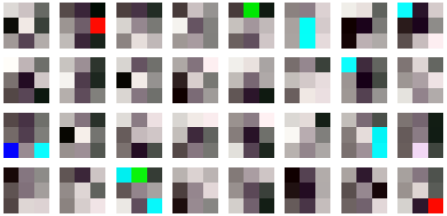


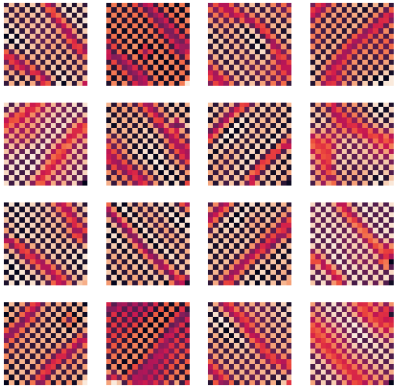
2QVU classified as 2JVU

- The cause of this may be overfitting.
- The weights and the kernel learned by the model is shown in the next slide.

c4l_16x16_550

- The 3x3 kernels learned for layer 1 and the weights learned by the model from layers 1 to 16.





JAM CAPTCHA dataset

JAM CAPTCHA

About the data

- We found this CAPTCHA from the JAM's website.
- We scraped about 10,000 files, but used only 308 files for making the dataset. We made a custom tool to easily label the dataset.
- Initial dimension of the data is about 100x40x3, with three channels, which was unnecessary.
- Our main aim was to remove the line and then segment the characters and use some simple model to classify those images.
- We applied a Gaussian filter with $\sigma = 1$ and $\mu = 0$ to remove the strikethrough. We thresholded the value and then applied k-NN algorithm after segmentation.

JAM CAPTCHA

- Distribution of letters in the JAM CAPTCHA dataset:

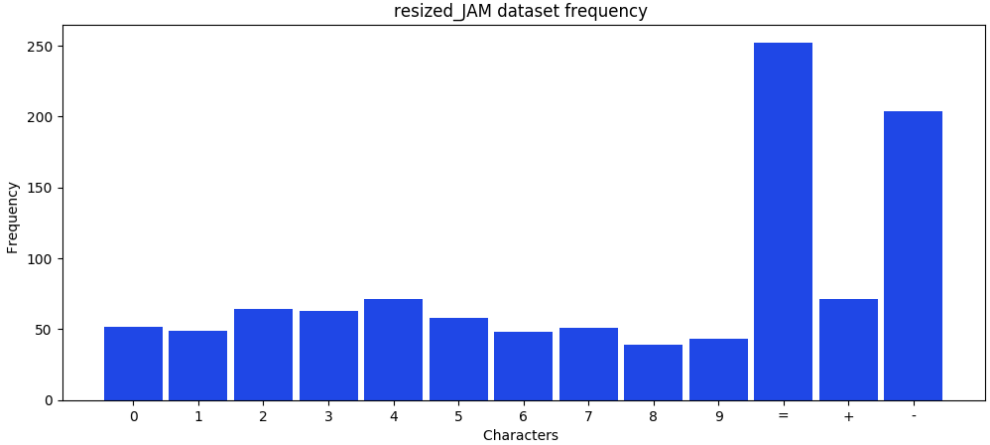
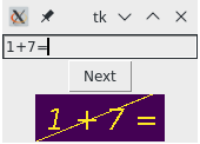


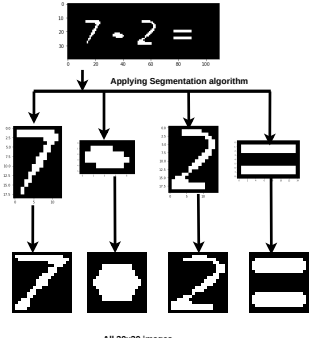
Figure: The initial distribution of the letters for the JAM CAPTCHA dataset.

JAM CAPTCHA

- Our custom made labeller to label JAM dataset:

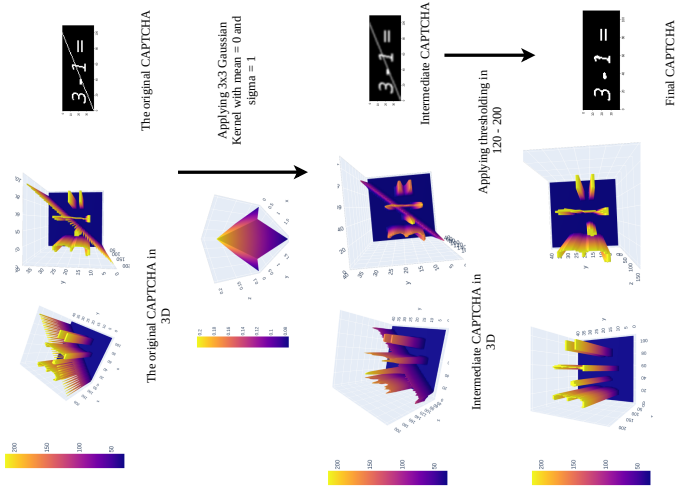


- The segmentation applied to JAM CAPTCHA after removing strikethrough.

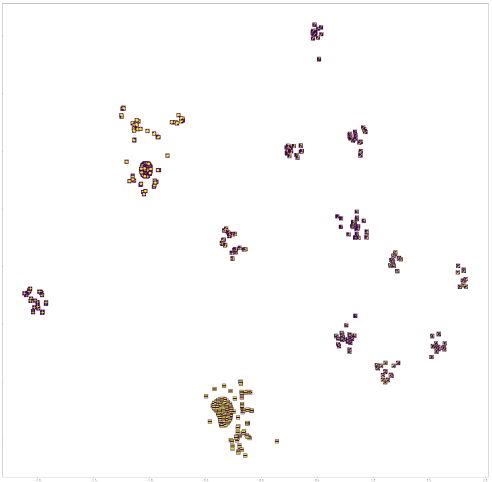


JAM CAPTCHA

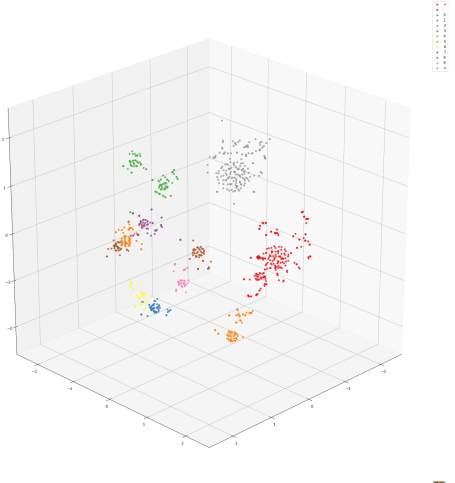
- Process of segmentation of the JAM CAPTCHA.



- The t-SNE plots of the letters of the segmented 20x20 JAM CAPTCHA data.

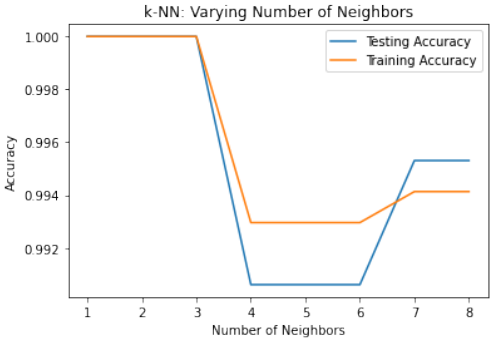


In 2D

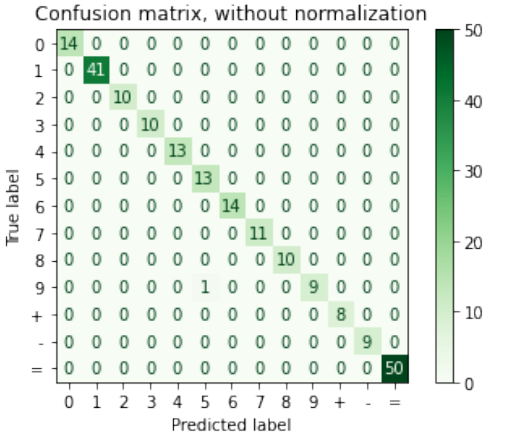


In 3D

- With a k of 7 we got about 99.53% accuracy.



Varying the value of K



Confusion Matrix obtained

- JAM website needs a better CAPTCHA !

CAPTCHA-version-2 dataset

CAPTCHA-version-2

About the data

- We found this CAPTCHA from Kaggle (`https://www.kaggle.com/fournierp/captcha-version-2-images`). The cleaned version can be found here (`https://jimut123.github.io/blogs/CAPTCHA/data/captcha_v2.tar.gz`)
- Contains about 1070 images, with a total size of 21.5 MB.
- The dimension is 200x50 with 3 channels, and have characters a-z and digits 0-9.
- The only problem was it contained a strike almost same as the thickness of individual characters, along with some noise.
- The distribution of the data is shown in the next slide.

- We can see there are a few missing letter, with a sudden spike in the frequency of the letter **n**.

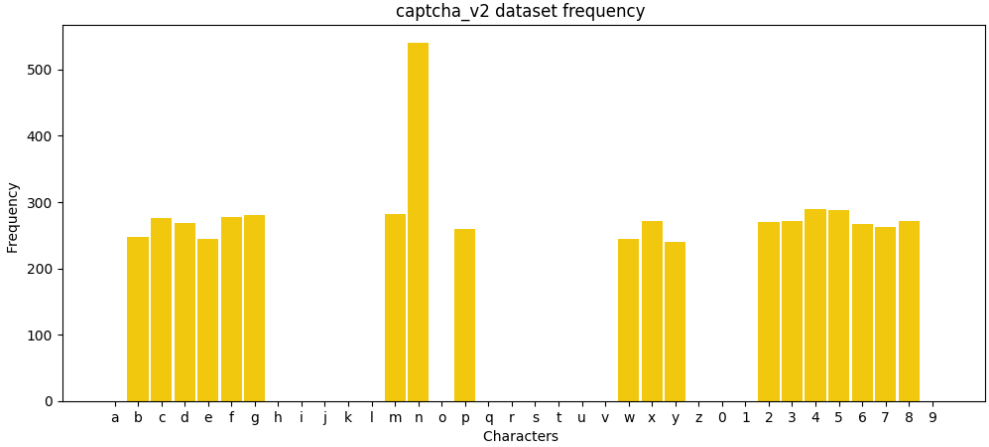


Figure: The initial distribution of the letters for the CAPTCHA-version-2 dataset.

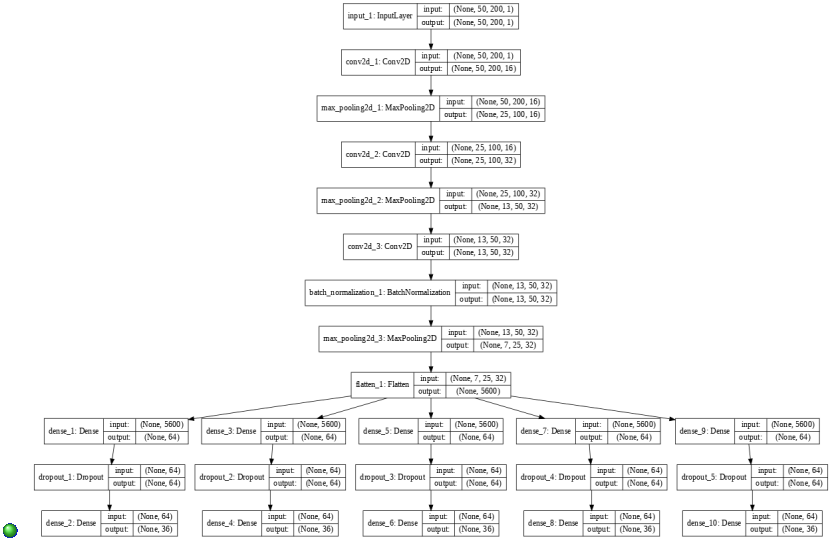


Figure: The model for the **CAPTCHA-version-2** dataset in Keras.

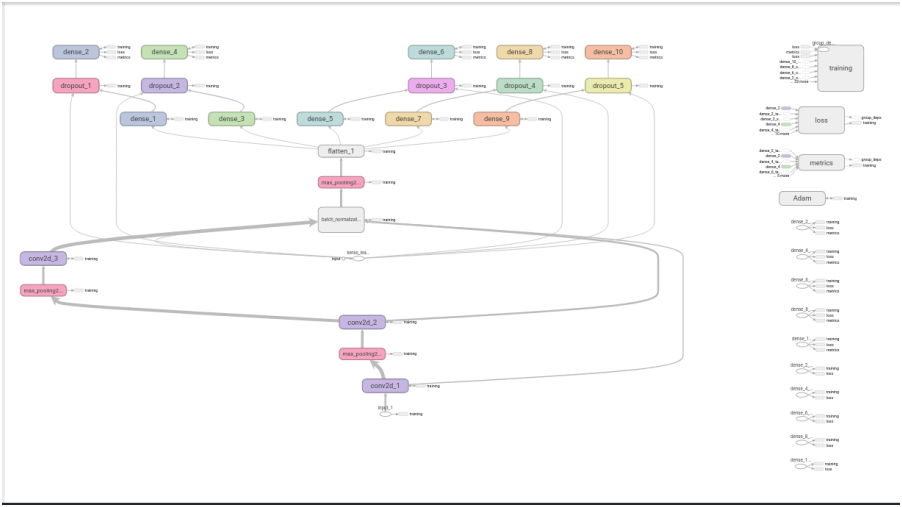
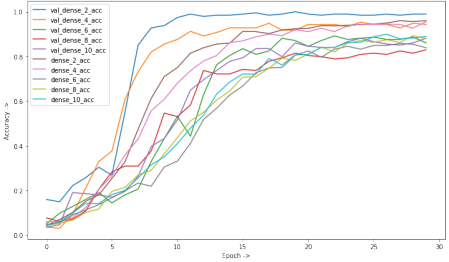
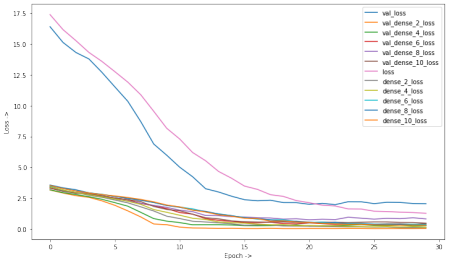


Figure: The model for the **CAPTCHA-version-2** dataset in tensorboard.

- The graph of accuracy and loss obtained from each layer of the model.



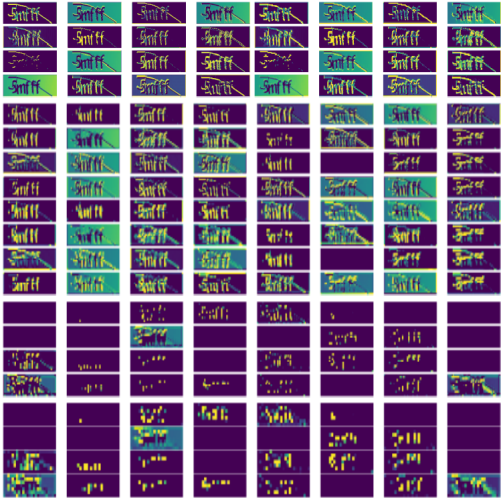
Accuracy



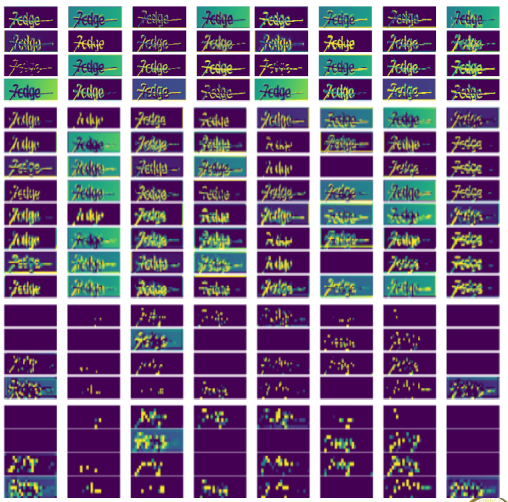
Loss

- Trained for 30 epochs, 38 seconds to train the whole dataset.
- 90.102 % accuracy on test dataset.

- The visualizations of the layers when image is passed through it.



5mfff



7cdge

1L-labelled-CAPTCHA dataset

1L-labelled-CAPTCHA

About the data

- We found this CAPTCHA from Kaggle `https://www.kaggle.com/digdeepbro/100000-labeled-captchas`. The cleaned version can be found here on figshare (`https://dx.doi.org/10.6084/m9.figshare.12046881.v1`)
- Contains 109,053 files of 200x50, with a total size of 215.1 MiB.
- This is same as the previous CAPTCHA but had a strikethrough of same thickness as the original letters.
- The only problem was it contained a strike almost same as the thickness of individual characters, along with some noise.

- We can see there are a few missing letters, with almost same distribution.

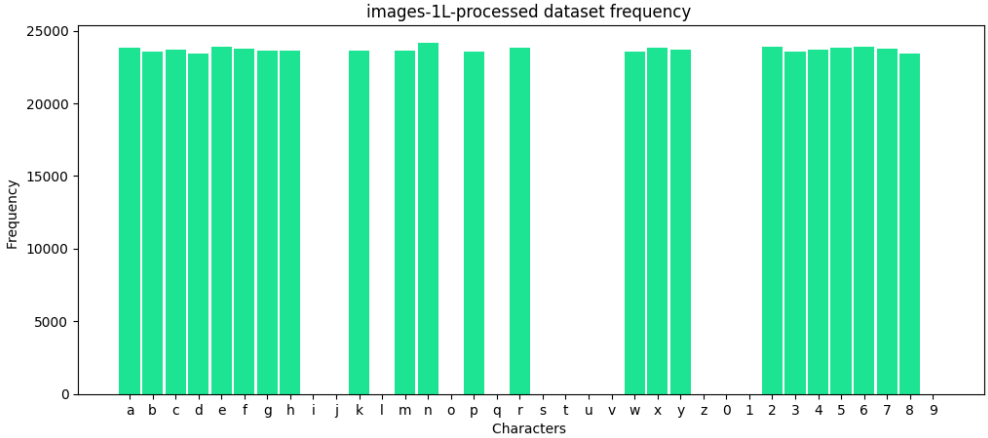
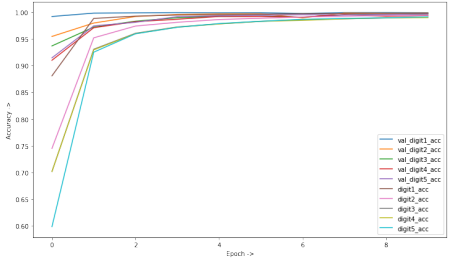


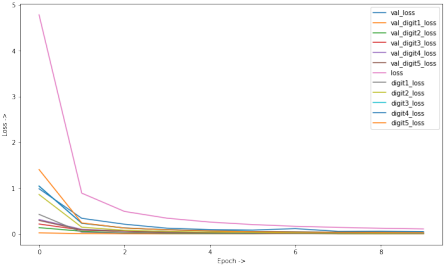
Figure: The initial distribution of the letters for the 1L-labelled-CAPTCHA dataset.

Figure: The model for the **captcha-1L** dataset in Keras.

- The graph of accuracy and loss obtained from each layer of the model.



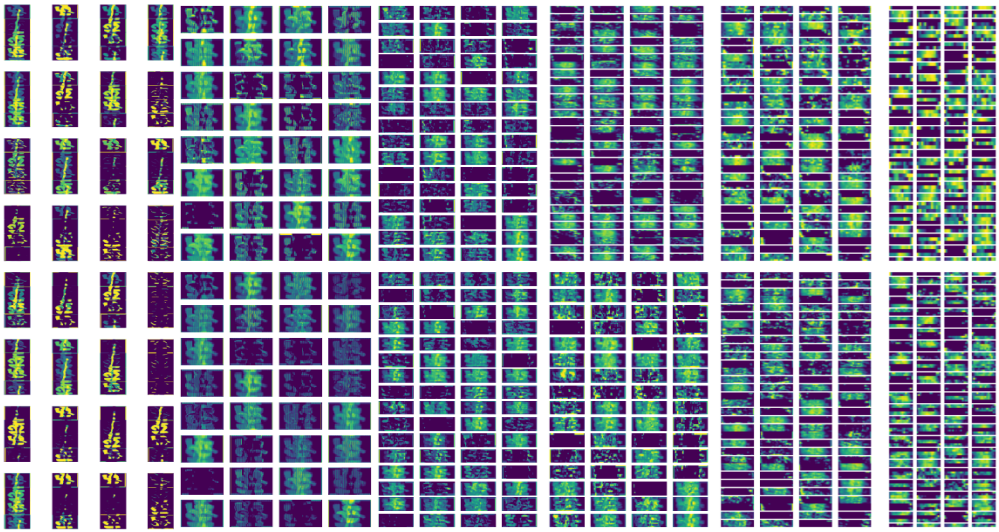
Accuracy



Loss

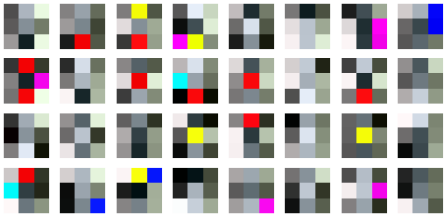
- Trained for 30 epochs, 35.5 minutes to train the whole dataset.
- 99.67 % accuracy on test dataset.

- The visualizations of the layers when CAPTCHA having text **r543n** is passed through it.



1L-labelled-CAPTCHA

- The 3x3 kernels learned for layer 1 and the weights learned by the model from layers 1 to 25.

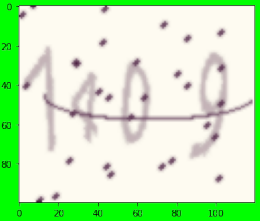


- The model was tested with a subset of images, and all of them were predicted correctly as shown in the next slide.

- The prediction on unseen data.

Figure: Prediction on some of the sample of the test set.

Faded CAPTCHA dataset



Faded CAPTCHA

About the data

- We found this CAPTCHA from Github `https://github.com/JackonYang/captcha-tensorflow`.
- The generator was written in Python3.
- The faded CAPTCHA dataset comprises of about 604800 files of dimension 100x120x3 for the three channels.
- We uploaded the dataset on figshare. It contains about 3.7 GB of data.
- The data samples mainly contains numbers with a lot of noise and a variety of colours.

- We can see there are a only numbers with same distribution.

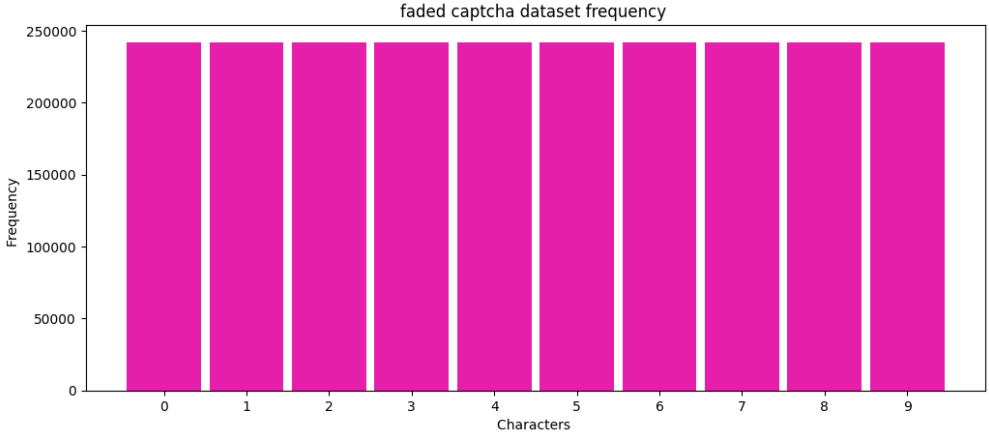


Figure: The initial distribution of the letters for the Faded CAPTCHA dataset.

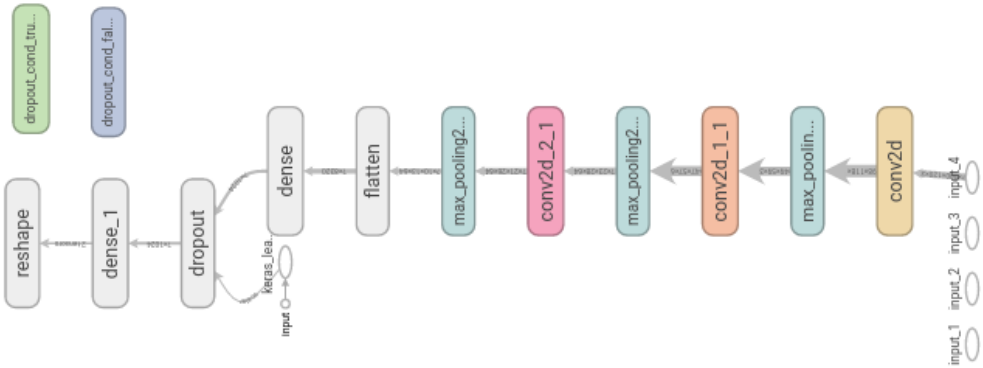
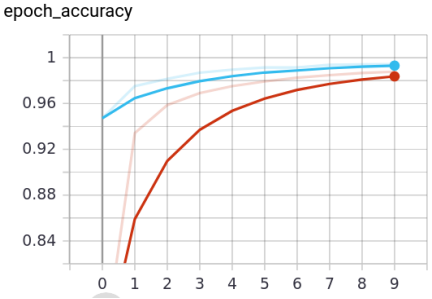
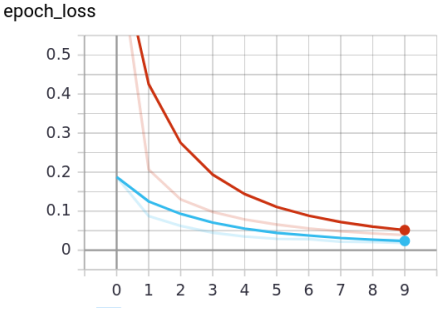


Figure: The model for the **Faded CAPTCHA** dataset in Tensorboard.

- The graph of accuracy and loss obtained from each layer of the model.



Accuracy



Loss

- Trained for 10 epochs, 2.56 hours to train the whole dataset.
- 98.78% on the validation set and an accuracy of 99.44% on the test set.

- The prediction on unseen data.

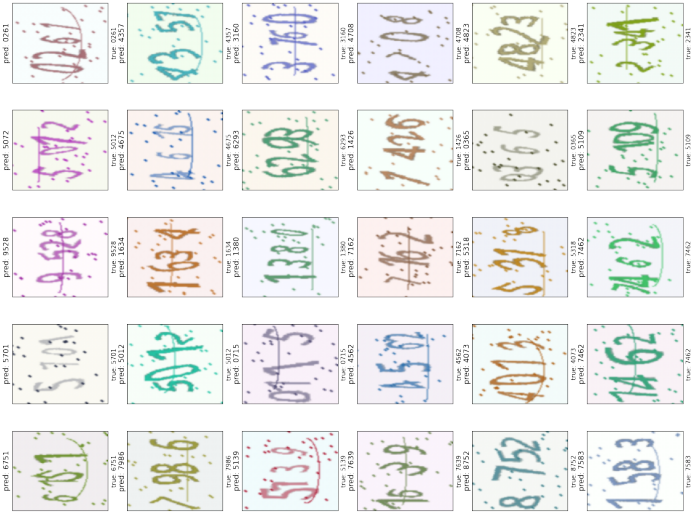


Figure: Prediction on some of the sample of the test set.

Circle CAPTCHA dataset

Circle CAPTCHA

About the data

- We found this CAPTCHA from Github `https://github.com/py-radicz/captcha_gen`.
- The generator was written in Python3.
- The Circle CAPTCHA dataset comprises of about 222K files of dimension 35x120x3 for the three channels.
- We uploaded the dataset on figshare. It contains about 437.58 MB of data.
- The data samples mainly contains letters a-z and numbers 0-9 with different patches of circle in them. The colour intensity of the letters also vary from sample to sample.

- We can see there are numbers and letters with equal distributions.

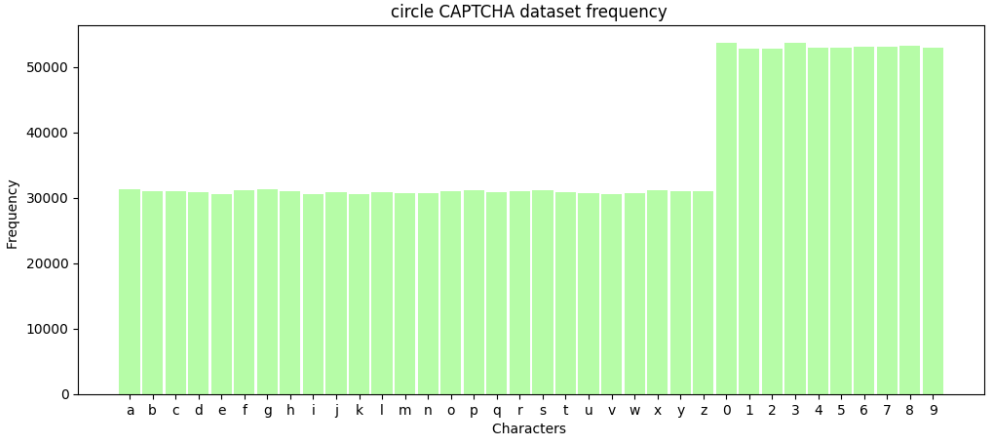


Figure: The initial distribution of the letters for the Circle CAPTCHA dataset.

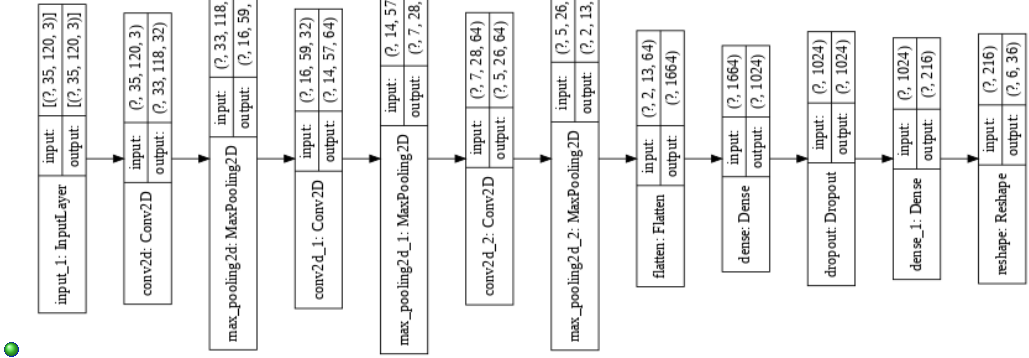
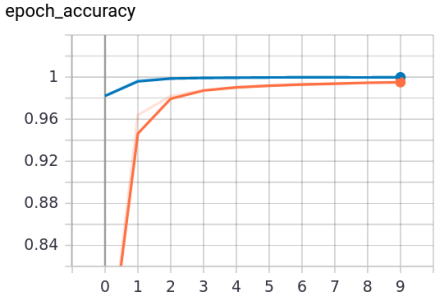


Figure: The model for the **Circle CAPTCHA** dataset in Keras.

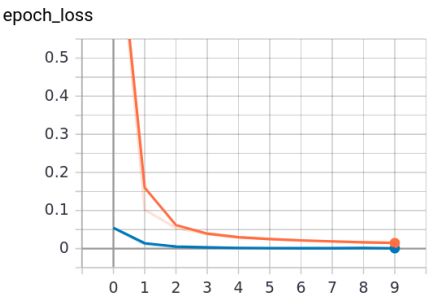
Circle CAPTCHA

A model for the **Circle CAPTCHA** dataset

- The graph of accuracy and loss obtained from the model during training.



Accuracy



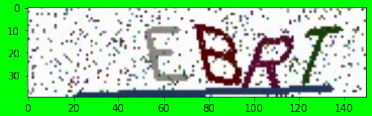
Loss

- Trained for 10 epochs, 1.80 hours to train the whole dataset.
- 99.98% on the validation set and an accuracy of 99.99% on the test set.

- The prediction on unseen data.

Figure: Prediction on some of the sample of the test set.

Sphinx CAPTCHA dataset



Sphinx CAPTCHA

About the data

- We found this CAPTCHA from Github `https://github.com/davidpalves/sphinx-captcha`.
- The generator was written in Python3.
- The Sphinx CAPTCHA dataset comprises of about 990K files of dimension 368x123x3 for the three channels.
- We uploaded the dataset on figshare. It contains about 12.2 GB of data.
- The data samples mainly contains letters A-Z with different color of strikethrough in them. The colour intensity of the letters also vary from sample to sample, with a lot of noise.

- We can see there are letters with almost equal distributions.

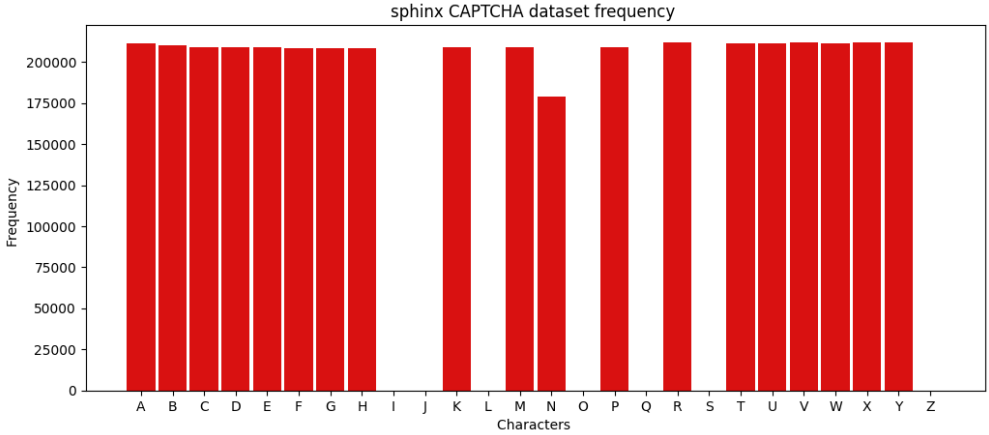


Figure: The initial distribution of the letters for the Sphinx CAPTCHA dataset.

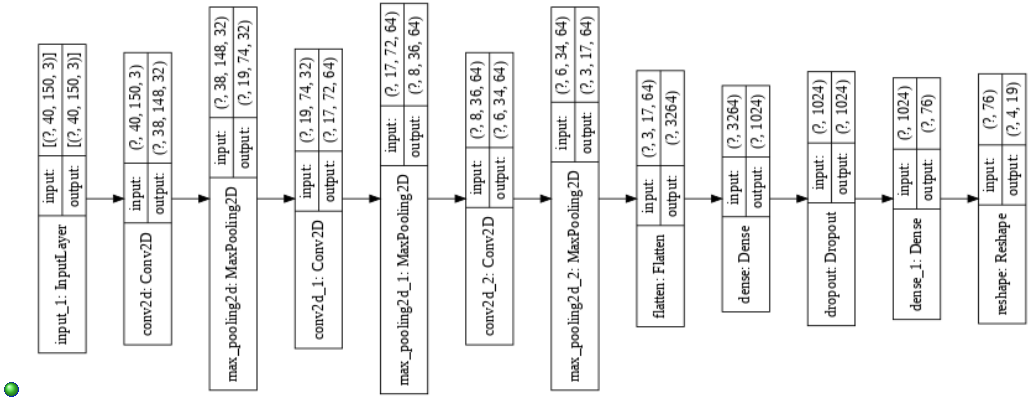
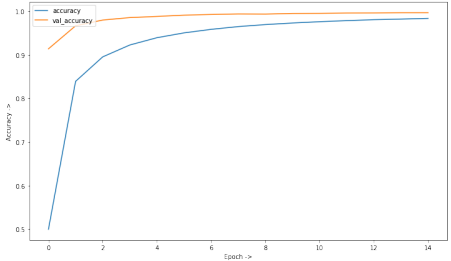
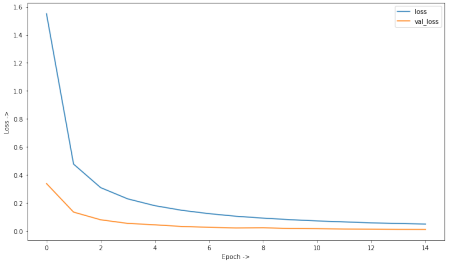


Figure: The model for the **Sphinx CAPTCHA** dataset in Keras.

- The graph of accuracy and loss obtained from the model during training.



Accuracy



Loss

- Trained for 15 epochs, 5.3875 hours to train the whole dataset.
- accuracy of 99.62% on the test set.

- The prediction on unseen data.

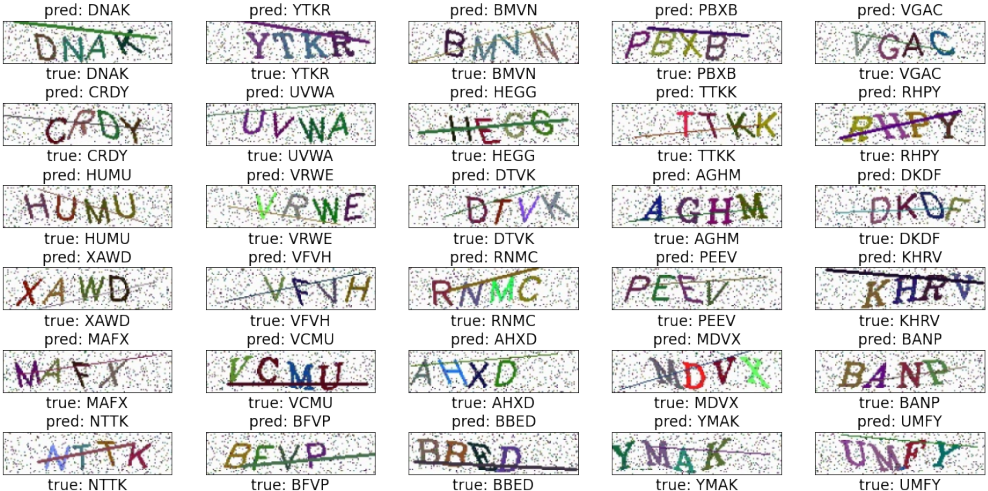
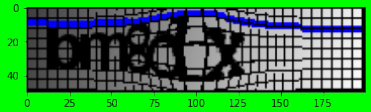


Figure: Prediction on some of the sample of the test set.

Fish Eye CAPTCHA dataset



Fish Eye CAPTCHA

About the data

- We found this CAPTCHA from Java CAPTCHA website and created our own dataset.
- The generator was written in Java.
- The Fish Eye CAPTCHA dataset comprises of about 2 Million files of dimension 50x200x3 for the three channels, and was a five letter CAPTCHA.
- We uploaded the dataset on figshare `https://figshare.com/articles/Fish_Eye/12235946`. It contains about 9.84 GB of data.
- The data samples mainly contains letters a-z, and numbers. It have a fish eye type noise on the middle and grids to confuse the Deep Learning Model.

- We can see there are letters with almost equal distributions.

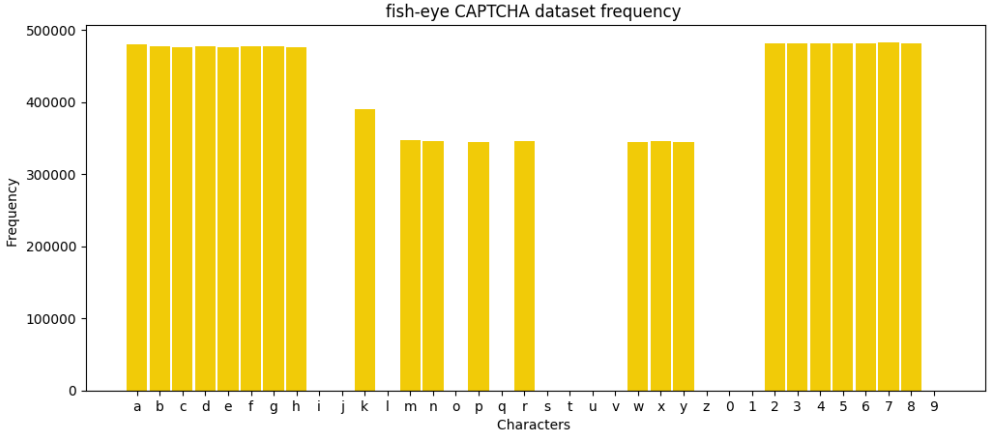


Figure: The initial distribution of the letters for the Fish Eye CAPTCHA dataset.

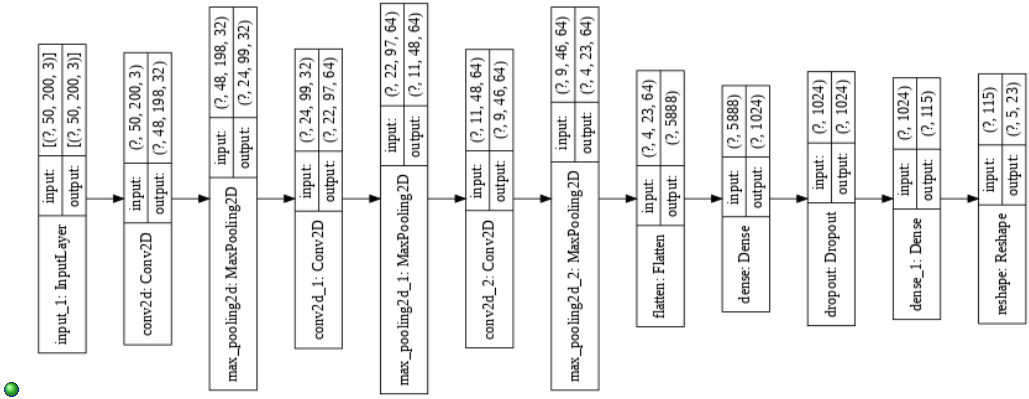
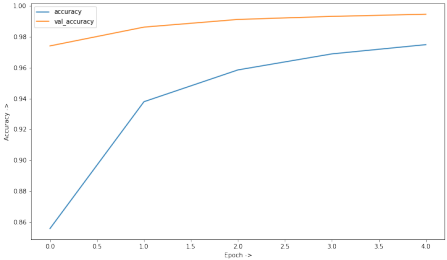
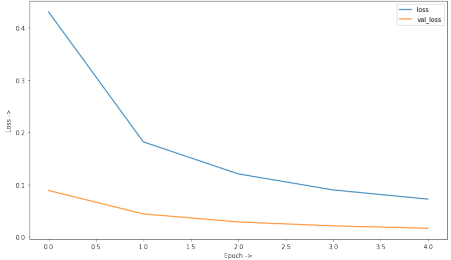


Figure: The model for the **Fish Eye CAPTCHA** dataset in Keras.

- The graph of accuracy and loss obtained from the model during training.



Accuracy



Loss

- Trained for 5 epochs, 10.83 hours to train the whole dataset.
- Accuracy obtained was 97.49 % on validation set and 99.46 % for test set.

- The prediction on unseen data.

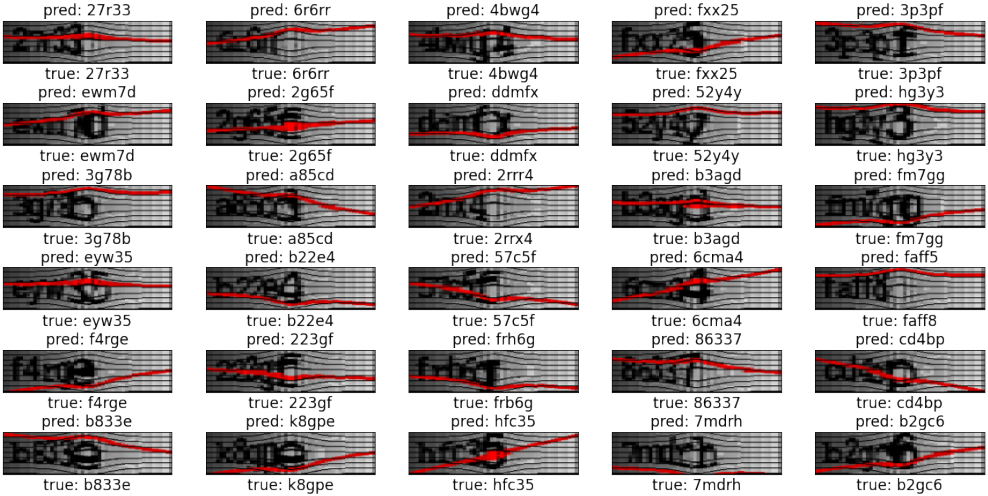
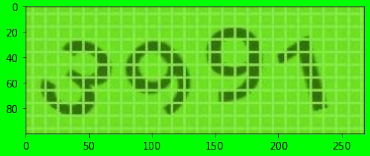


Figure: Prediction on some of the sample of the test set.

Mini CAPTCHA dataset



Mini CAPTCHA

About the data

- We found this CAPTCHA from Github `https://github.com/imanhpr/miniCaptcha`, and created our own dataset.
- The generator was written in Python3.
- The Mini CAPTCHA dataset comprises of about 1 Million files of dimension 368x159x3 for the three channels, and was a four letter CAPTCHA.
- We uploaded the dataset on figshare `https://figshare.com/articles/Mini_Captcha/12286697`. It contains about 5.58 GB of data.
- The data samples mainly contains letters [a-z][A-Z], and numbers [0-9]. It have a variety of colours and different orientations of the letters, with grid like structures, which is challenging for DL models.

- We can see there are letters with almost equal distributions.

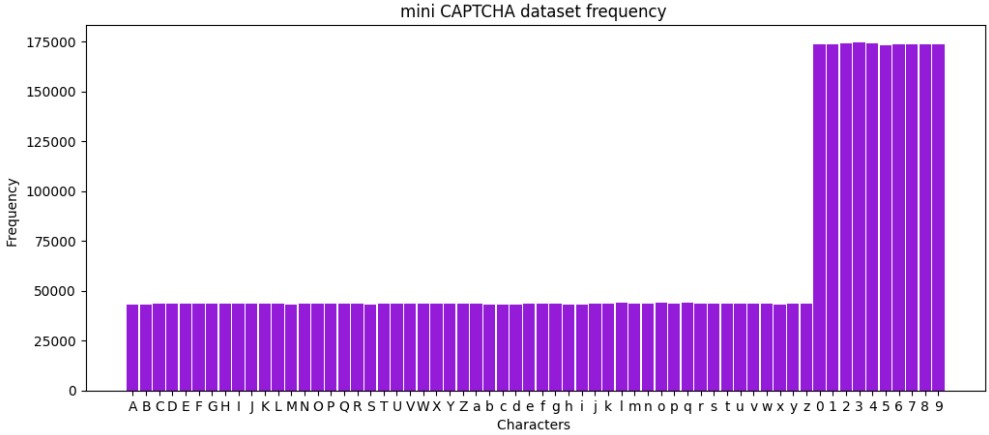


Figure: The initial distribution of the letters for the Mini CAPTCHA dataset.

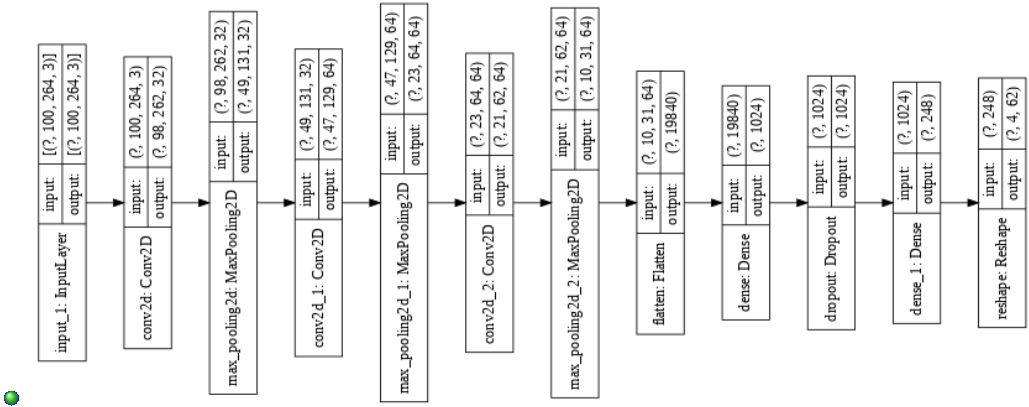
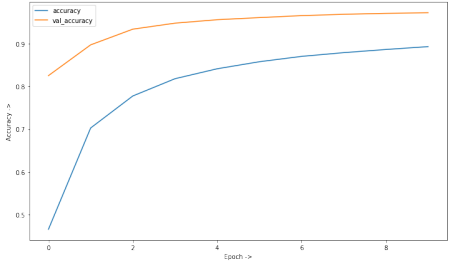


Figure: The model for the **Mini CAPTCHA** dataset in Keras.

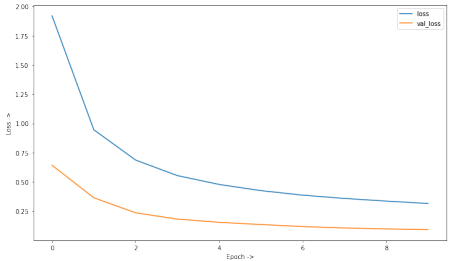
Fish Eye CAPTCHA

A model for the **Mini CAPTCHA** dataset

- The graph of accuracy and loss obtained from the model during training.



Accuracy



Loss

- Trained for 10 epochs, 5.75 hours to train the whole dataset.
- Accuracy obtained was about 97.25% on validation set and 97.09% on the test set.

- The prediction on unseen data.

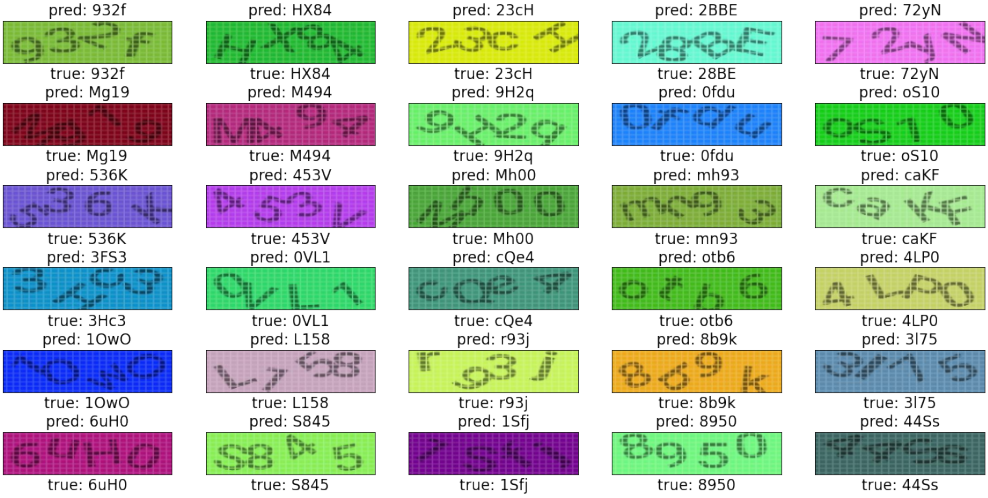
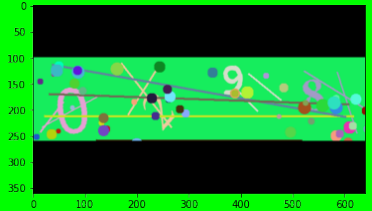


Figure: Prediction on some of the sample of the test set.

Multicolor CAPTCHA dataset



Multicolor CAPTCHA

About the data

- We found this CAPTCHA from Github `https://github.com/J-Rios/multicolor_captcha_generator`, and created our own dataset.
- The generator was written in Python3.
- The Multicolor CAPTCHA dataset comprises of about 1 Million files of dimension 375x223x3 for the three channels, and was a four letter CAPTCHA of difficulty level 4.
- We uploaded the dataset on figshare. It contains about 12.7 GB of data. We have used about 50% of the original dataset to train our model.
- The data samples mainly contains numbers [0-9]. It have a variety of colours and different orientations of the letters, with a lot of noise, which is even difficult for humans sometimes to decipher.

- We can see there are labels as numbers with almost equal distributions.

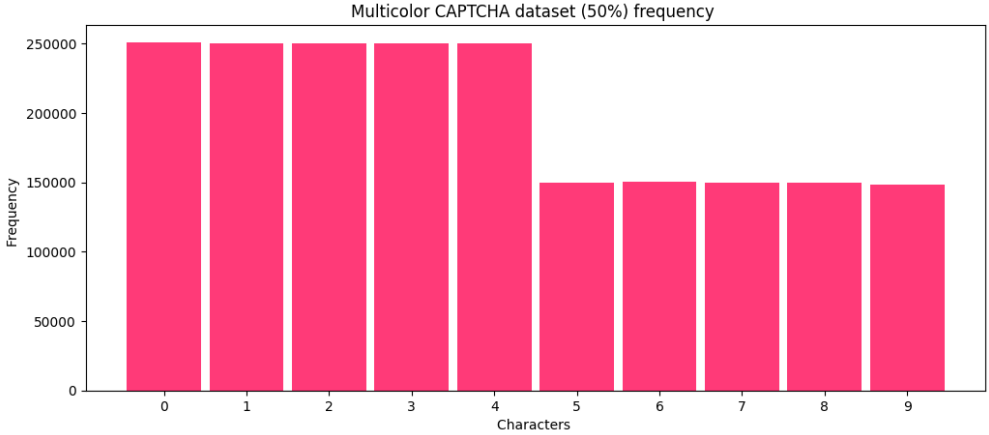


Figure: The initial distribution of the letters for the Multicolor CAPTCHA dataset.

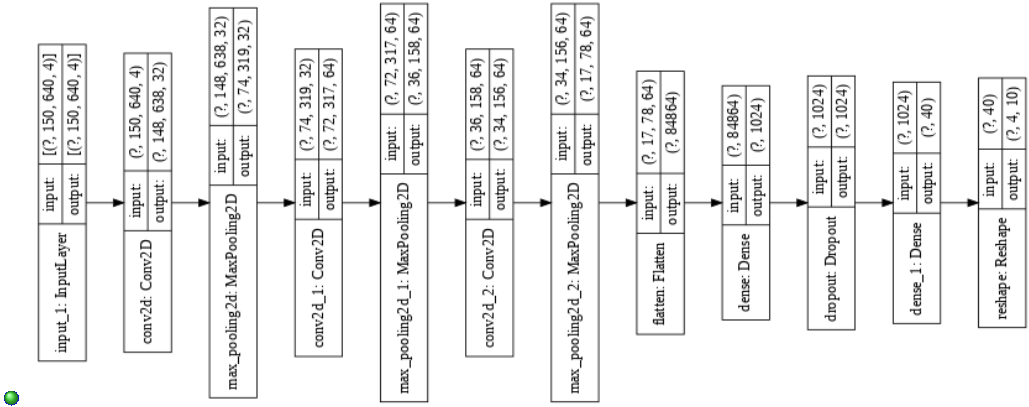
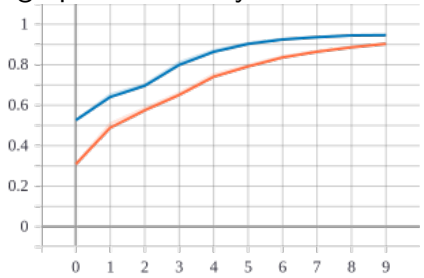
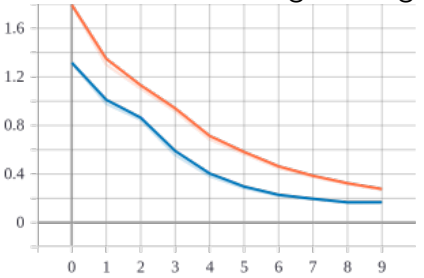


Figure: The model for the **Multicolor CAPTCHA** dataset in Keras.

- The graph of accuracy and loss obtained from the model during training.



Accuracy



Loss

- Trained for 9 epochs, 8.705 hours to train the whole dataset.
- Accuracy obtained was about 92.45% on validation set and 95.69% on the test set.

- The prediction on unseen data.

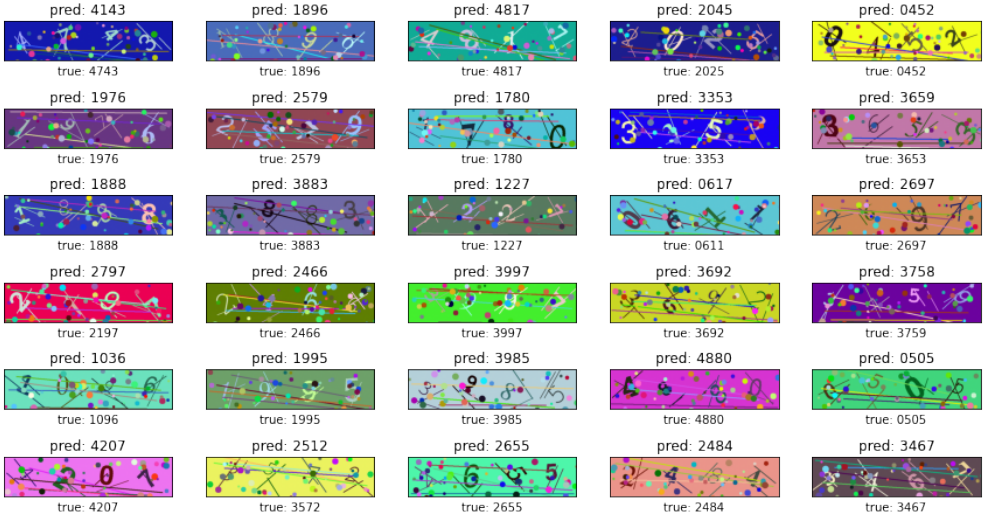
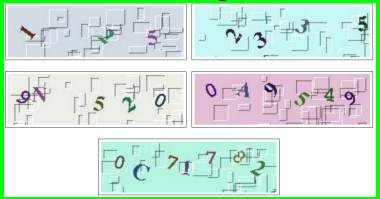


Figure: Prediction on some of the sample of the test set.

Determining complexity of CAPTCHAs through Railway-CAPTCHA



Determining complexity of CAPTCHAs through Railway-CAPTCHA

About the data

- We found this CAPTCHA from Github `https://github.com/JasonLiTW/simple-railway-captcha-solver`, and created our own dataset comprising of 3, 4, 5, 6, and 7 letter CAPTCHAs.
- The generator was written in Python3.
- We created 100K files of each of the CAPTCHAs to compare the accuracies obtained by using similar type of model.
- We uploaded the dataset on figshare. The images are of constant size, i.e., 60x200 for each of the CAPTCHAs.
- The data samples mainly contains numbers [0-9] with high probability and letters [A-Z]. It have a variety of colours and different orientations of the letters, with a lot of noise in the form of grids.

- We will see each of the distributions of the Railway CAPTCHA.

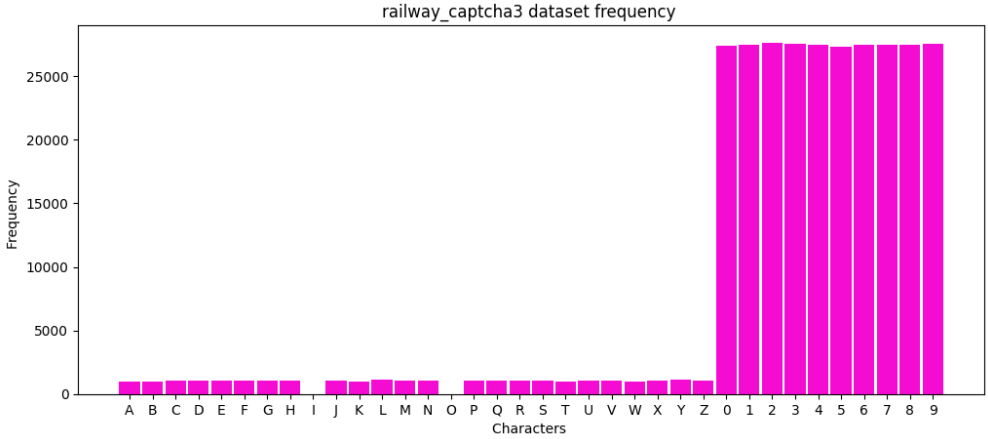
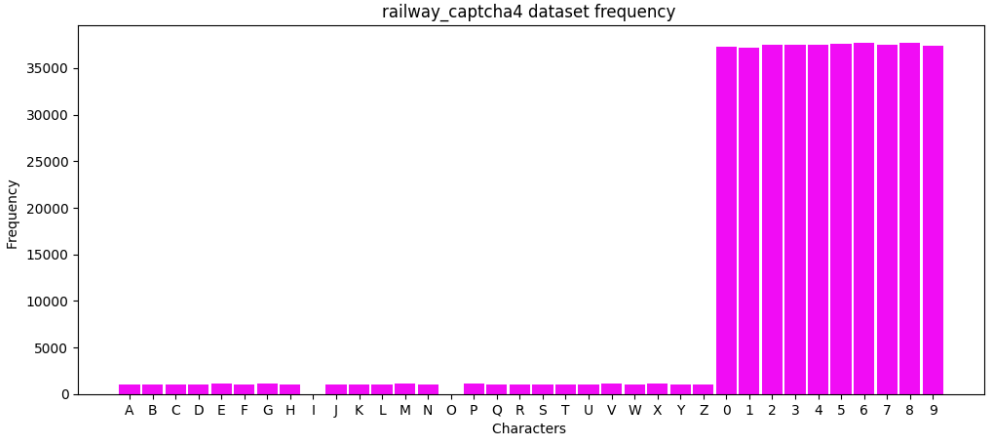
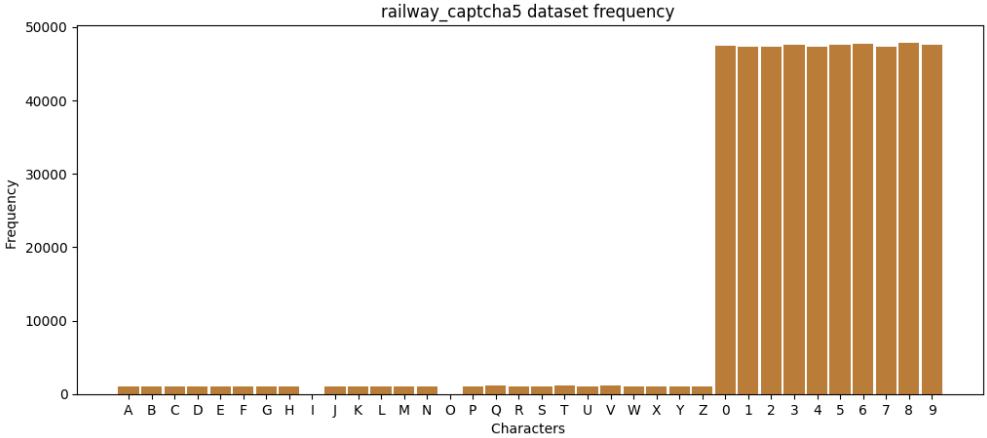


Figure: The initial distribution of the letters for the Railway CAPTCHA (3 letters) dataset.

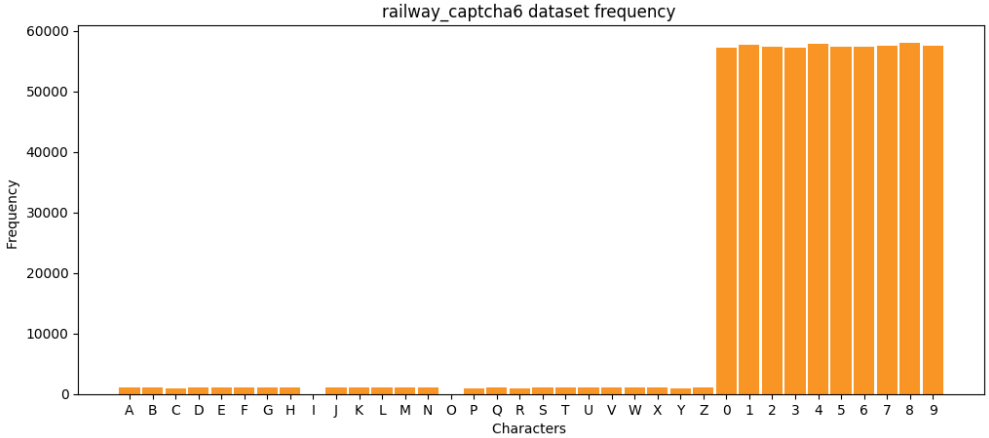
- The initial distribution of the letters for the Railway CAPTCHA (4 letters) dataset.



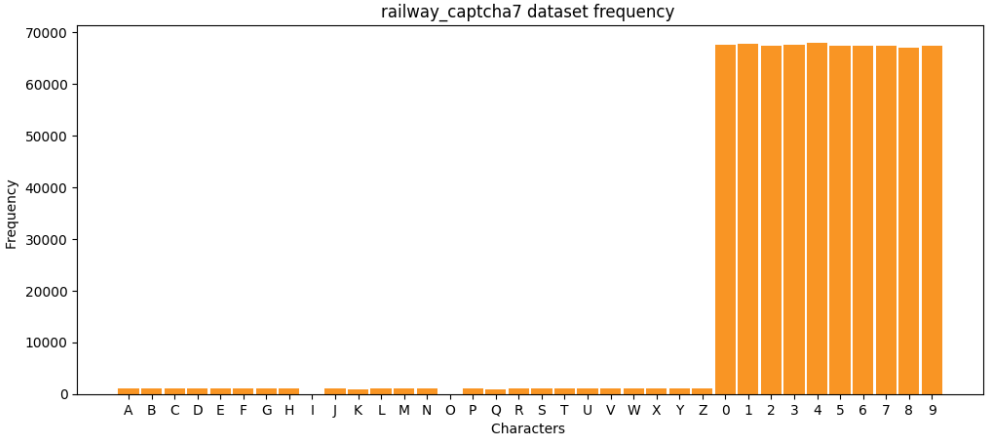
- The initial distribution of the letters for the Railway CAPTCHA (5 letters) dataset.



- The initial distribution of the letters for the Railway CAPTCHA (6 letters) dataset.



- The initial distribution of the letters for the Railway CAPTCHA (7 letters) dataset.



- We have seen that the distribution of the letters of the Railway CAPTCHA remains constant.
- This may be due to the fact the way the CAPTCHA generator has been made, which syncs to the clock speed to create such uniform distribution.
- In the coming slides we will show how the models differ in the output layers for each of the Railway CAPTCHAs.
- The models accept 1 channel black and white images, so it needs to be preprocessed first.
- Increasing the number of data samples may increase the accuracy, but we have used a small amount of data for this purpose.

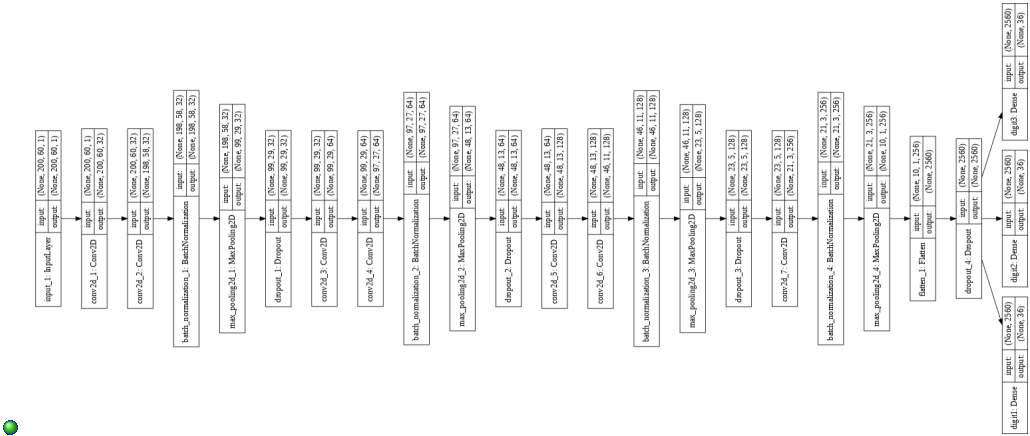


Figure: The model for the **Railway CAPTCHA (3 letters)** dataset in Keras.

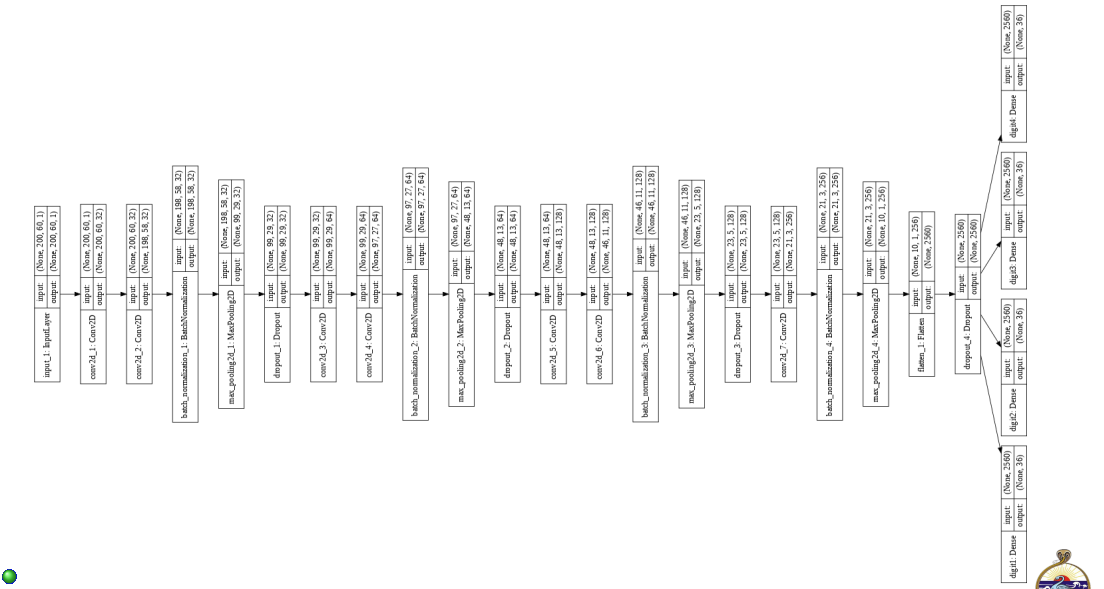


Figure: The model for the **Railway CAPTCHA (4 letters)** dataset in Keras.

Figure: The model for the **Railway CAPTCHA (5 letters)** dataset in Keras.

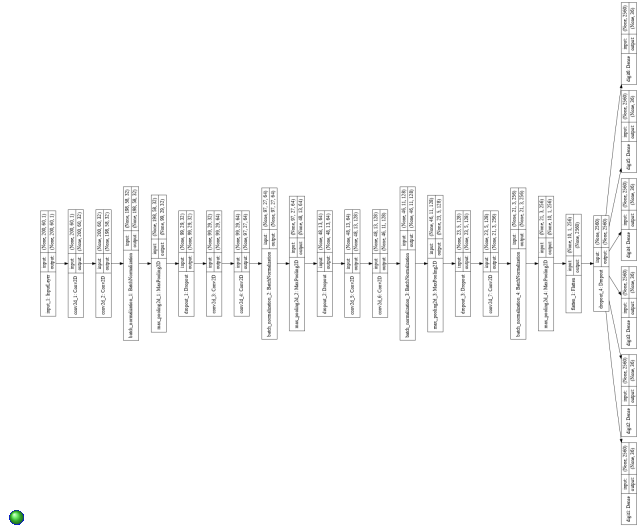


Figure: The model for the **Railway CAPTCHA (6 letters)** dataset in Keras.

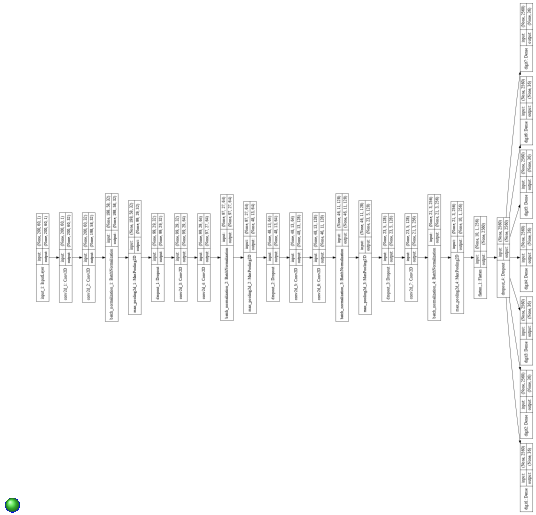
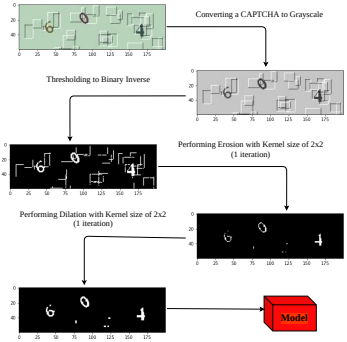


Figure: The model for the **Railway CAPTCHA (7 letters)** dataset in Keras.

Railway CAPTCHA

Accuracies obtained from the **Railway CAPTCHA** datasets.

- We now show the preprocessing that is done before passing through the railway CAPTCHA model.

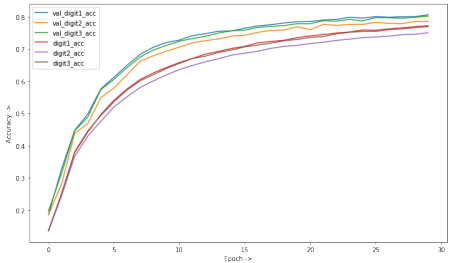


- In the following slides, we will show the performance of the model.

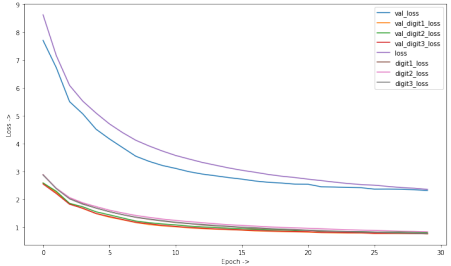
Railway CAPTCHA

Performance of the model (3 letter)

- The graph of accuracy and loss obtained from the model during training.



Accuracy



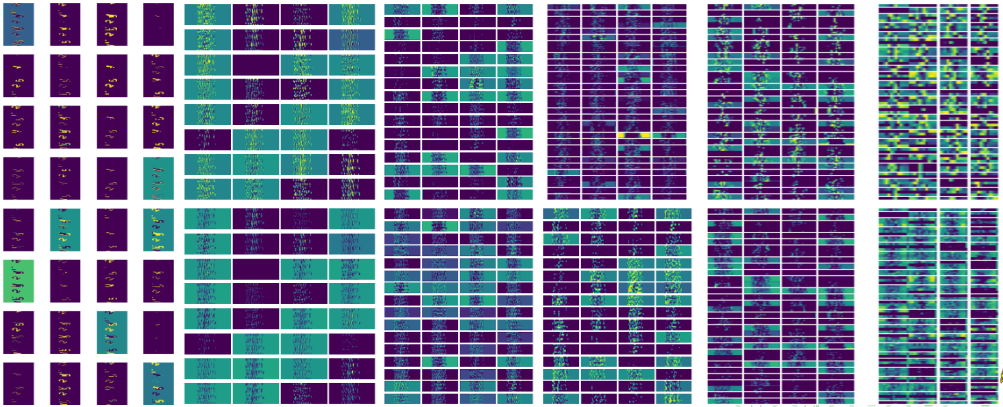
Loss

- Trained for 30 epochs, 40.5 minutes to train the whole dataset.
- Accuracy obtained was about 80% on the test set, 9 images were mislabelled.

Railway CAPTCHA

Performance of the model (3 letter)

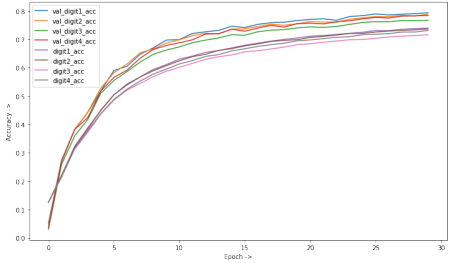
- The visualizations generated when a CAPTCHA of label 254 is passed through the model.



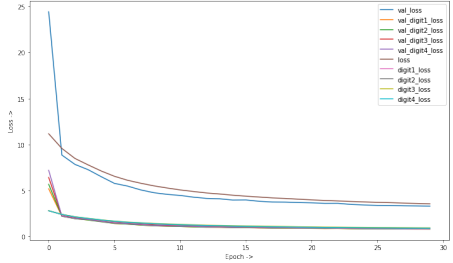
Railway CAPTCHA

Performance of the model (4 letter)

- The graph of accuracy and loss obtained from the model during training.



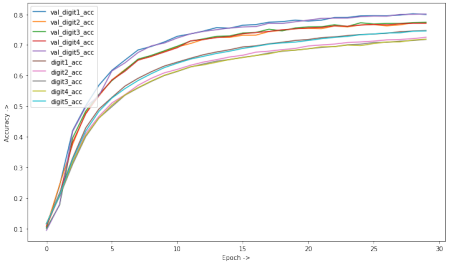
Accuracy



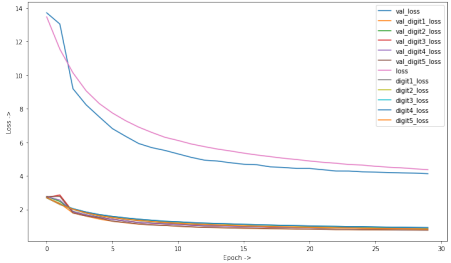
Loss

- Trained for 30 epochs, 43.5 minutes to train the whole dataset.
- Accuracy obtained was about 79% on the test set, 18 images were mislabelled.

- The graph of accuracy and loss obtained from the model during training.



Accuracy



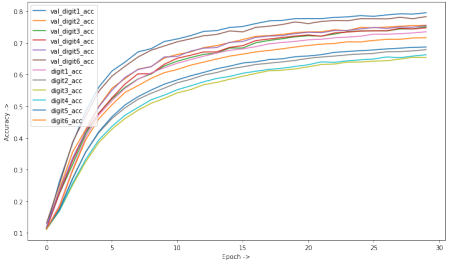
Loss

- Trained for 30 epochs, 41.5 minutes to train the whole dataset.
- Accuracy obtained was about 77% on the test set, and 23 images were mislabelled.

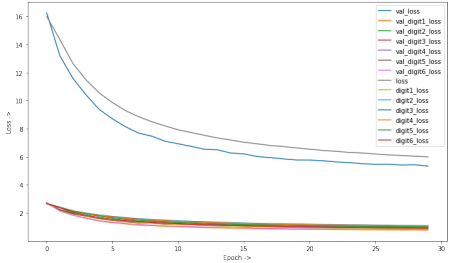
Railway CAPTCHA

Performance of the model (6 letter)

- The graph of accuracy and loss obtained from the model during training.



Accuracy



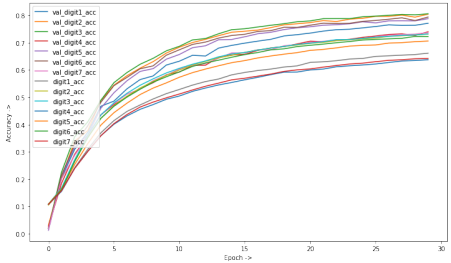
Loss

- Trained for 30 epochs, 41.5 minutes to train the whole dataset.
- Accuracy obtained was about 77% on the test set, and 25 images were mislabelled.

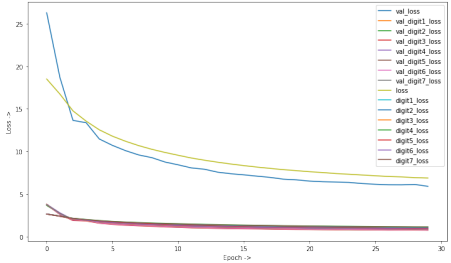
Railway CAPTCHA

Performance of the model (7 letter)

- The graph of accuracy and loss obtained from the model during training.



Accuracy



Loss

- Trained for 30 epochs, 45.5 minutes to train the whole dataset.
- Accuracy obtained was about 75% on the test set, and all the images were mislabelled.

- The prediction on unseen data (3 letter).

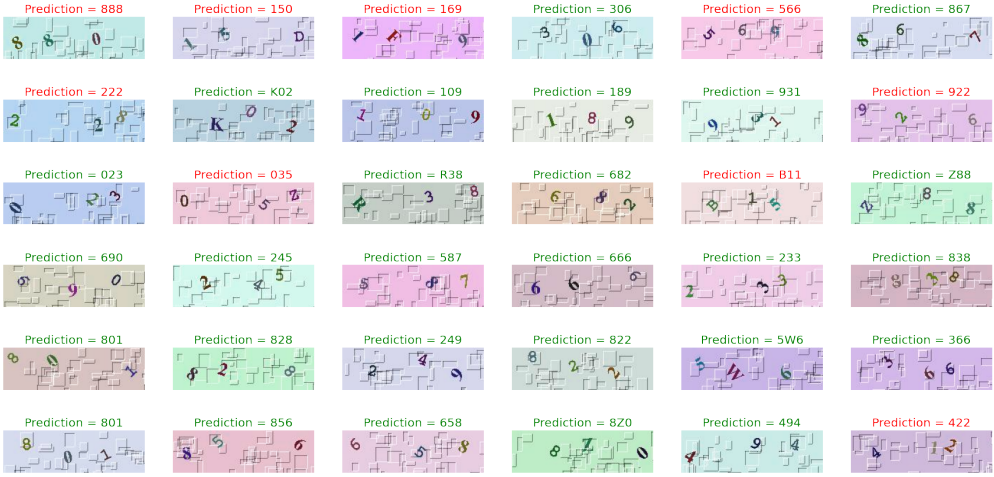


Figure: Prediction on some of the sample of the test set.

- The prediction on unseen data (4 letter).

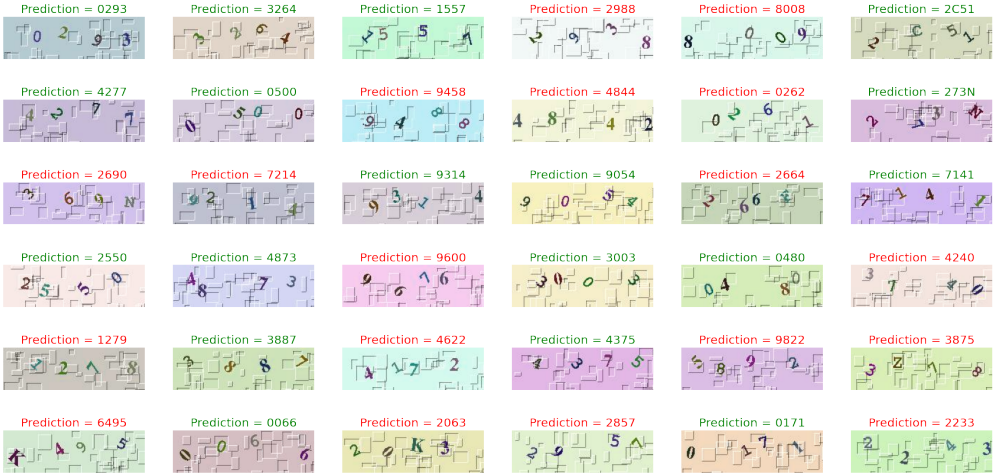


Figure: Prediction on some of the sample of the test set.

- The prediction on unseen data (5 letter).

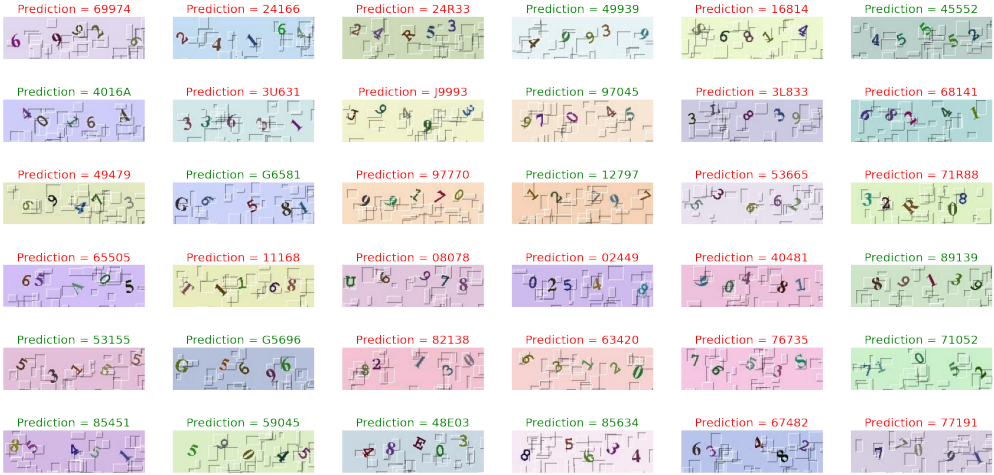


Figure: Prediction on some of the sample of the test set.

- The prediction on unseen data (6 letter).

Figure: Prediction on some of the sample of the test set.

- The prediction on unseen data (7 letter).

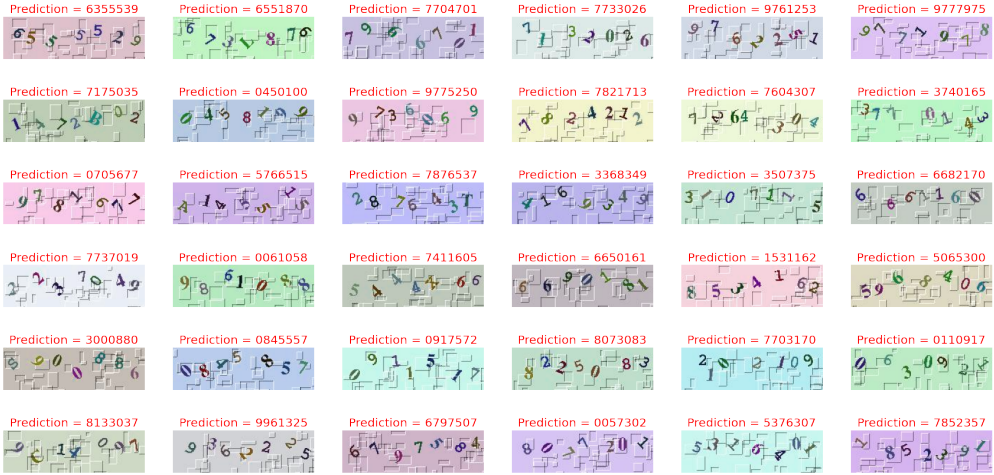


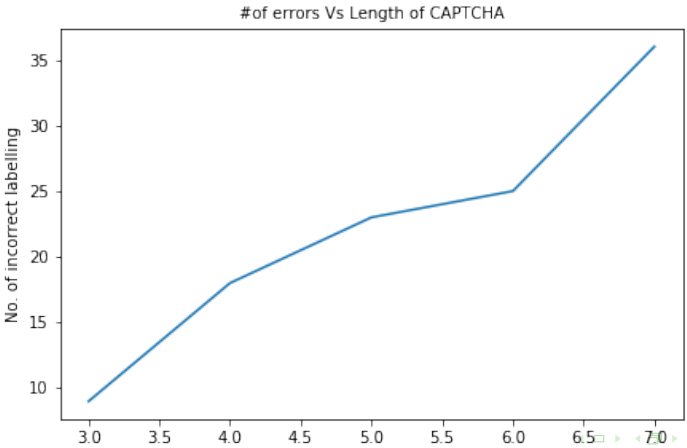
Figure: Prediction on some of the sample of the test set.

Our Claim

- We claim that increasing the number of labels in a CAPTCHA linearly increases the complexity of the CAPTCHA.
- The complexity of CAPTCHAs for humans increases with addition of noise, transformations and other obscurity. The complexity of CAPTCHA increases much more for computers with the same settings with increasing the length for multi-label classification.

- In any case each segmented character recieves an accuracy of classification of about 95 % then if there are about 10 letter then the total accuracy of correct classification becomes $(95)^{10}$ which is 59.87%.
- We conjecture that long length CAPTCHA are better solution to hard games which are not easy to understand at first glance, and takes a reasonable amount of time in determining the moves for genuine users.

- The comparison of size vs complexity of Railway CAPTCHA is shown in the following figure.



CONCLUSIONS

Conclusions

And Future work

- We have seen that Text Based CAPTCHAs are not so much reliable source for performing Reverse Turing Test.
- We have also found that AlexNet with dropouts at the second last layer does a very good job to train ensembles of Deep Learning Model through one model.
- Text based CAPTCHAs are going away from most of the websites recently because it is easy to break them with CNN and DNN techniques.
- Future work may include the application of RNN to predict and digitise contents of handwritten manuscripts which will be a good application for information retrieval.
- Detecting and localizing text in unconstrained environments is one of the most challenging tasks, and it needs state of the art Deep Learning Techniques.

References I

Pal. Jimut Bahan and Maharaj, Dripta
Deceiving Computers in Reverse Turing Test through Deep Learning.
(June 10, 2020). Available at SSRN.

Pal. Jimut Bahan and Maharaj, Tamal
A Deeper Look into Hybrid Images.
arXiv:2001.11302, cs.CV.

Wikipedia contributors.
Reverse Turing test — Wikipedia, The Free Encyclopedia.
https://en.wikipedia.org/wiki/Reverse_Turing_test, last accessed 24-03-2020.

References II

TURING, A. M.
I.—COMPUTING MACHINERY AND INTELLIGENCE,
https://academic.oup.com/mind/article-pdf/LIX/236/433/30123314/lix-236-433.pdf,
https://doi.org/10.1093/mind/LIX.236.433,

Van Der Maaten, Laurens and Hinton, Geoffrey
Visualizing Data using t-SNE,
http://www.jmlr.org/papers/v9/vandermaaten08a.html, Journal of Machine Learning Research,

Acknowledgements I

It is ritual that scholars express their gratitude to their supervisors. This acknowledgement is very special to me to express my deepest sense of gratitude and pay respect to my supervisor, **Dripta Maharaj**, School of Mathematical Sciences, for his constant encouragement, guidance, supervision, and support throughout the completion of my project. His close scrutiny, constructive criticism, and intellectual insight have immensely helped me in every stage of my work. I would like to thank him for patiently answering my often naive questions related to statistical machine learning.

I would like to acknowledge the help recieved from **Anindya Chowdhury** for helping me in data generation and uploading some of the data to public repositories, without which it would have taken much longer time for me to finish this project. I would also like to thank my aunt, **Nupur Kundu**, who had helped in labelling some of the images from the scrapped JAM dataset. I would also like to mention about the Google Collaboratory platform, which has always been a

Acknowledgements II

lab for my experiments, especially this project is completed by the use of their free GPUs.
I would like to thank all the professors of Department of Computer Science for creating an intellectually stimulating environment that enabled me to think more carefully and methodically about my research than I had ever done before. I would like to thank **Swathy Prabhu Maharaj**, **Tamal Maharaj** for the stimulating discussions that he had shared with me during the Computer Vision course.
I'm grateful to my father, Dr. **Jadab Kumar Pal**, Deputy Chief Executive, Indian Statistical Institute, Kolkata for constantly motivating and supporting me to develop this documentation along with the proofreading. I will also mention about my brother **Jisnoo Dev Pal** and my mother **Sumita Pal**, for supporting me.

Acknowledgements III

I would like to thank **Nilayan Paul** for modifying the railway CAPTCHA generator to suit my need. Finally, I acknowledge the help received from all my friends and well-wishers whose constant motivation has promoted the completion of this project.

Thank You

jimutbahanpal@{yahoo,gmail.outlook}.com