I swear I'm not a robot!

Type the characters above:

I might be a robot

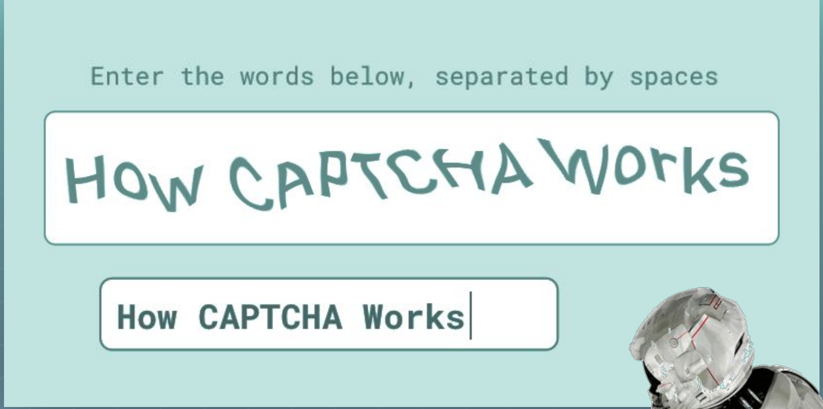
Breaking CAPTCHA

By,
Ray Zhao

Background

What is CAPTCHA?

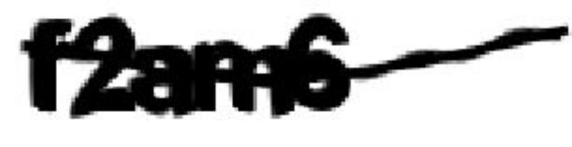
- CAPTCHA is an acronym for Completely Automated Public Turing test to tell Computers and Humans Apart.

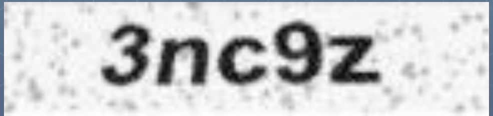


Enter the words below, separated by spaces

How CAPTCHA Works

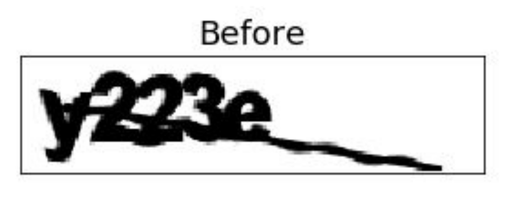
How CAPTCHA Works

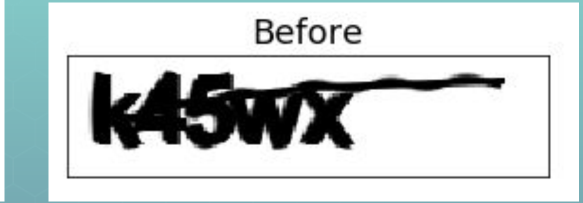




- Can we teach a machine to read and predict the text-based CAPTCHA?

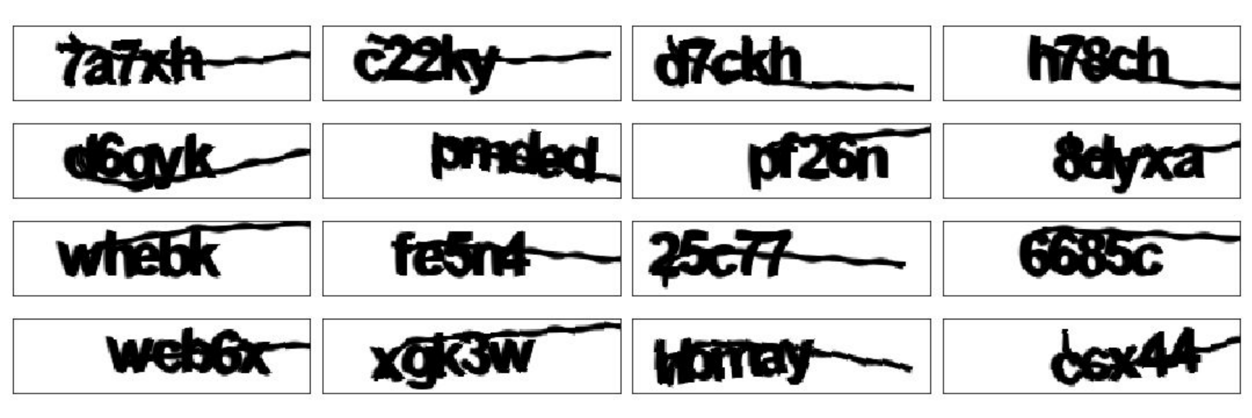
Captcha Data





- This version of CAPTCHA consists on 5 characters of either numbers or lowercase alphabetical letters.
- Placement of characters are the same throughout the data

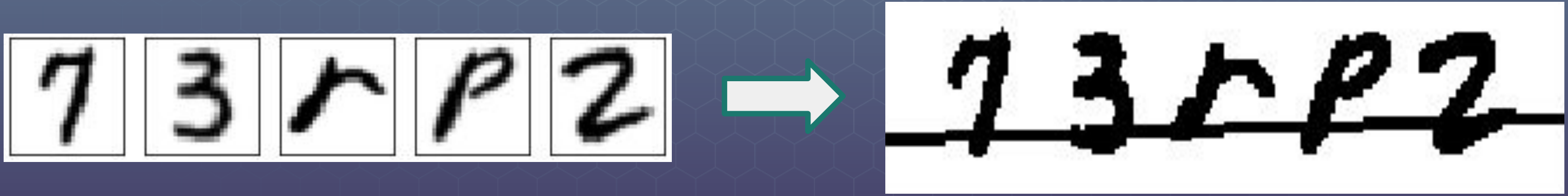
- Rotate and shift CAPTCHAs
- Prevents model from memorizing character location



Hand Drawn CAPTCHA

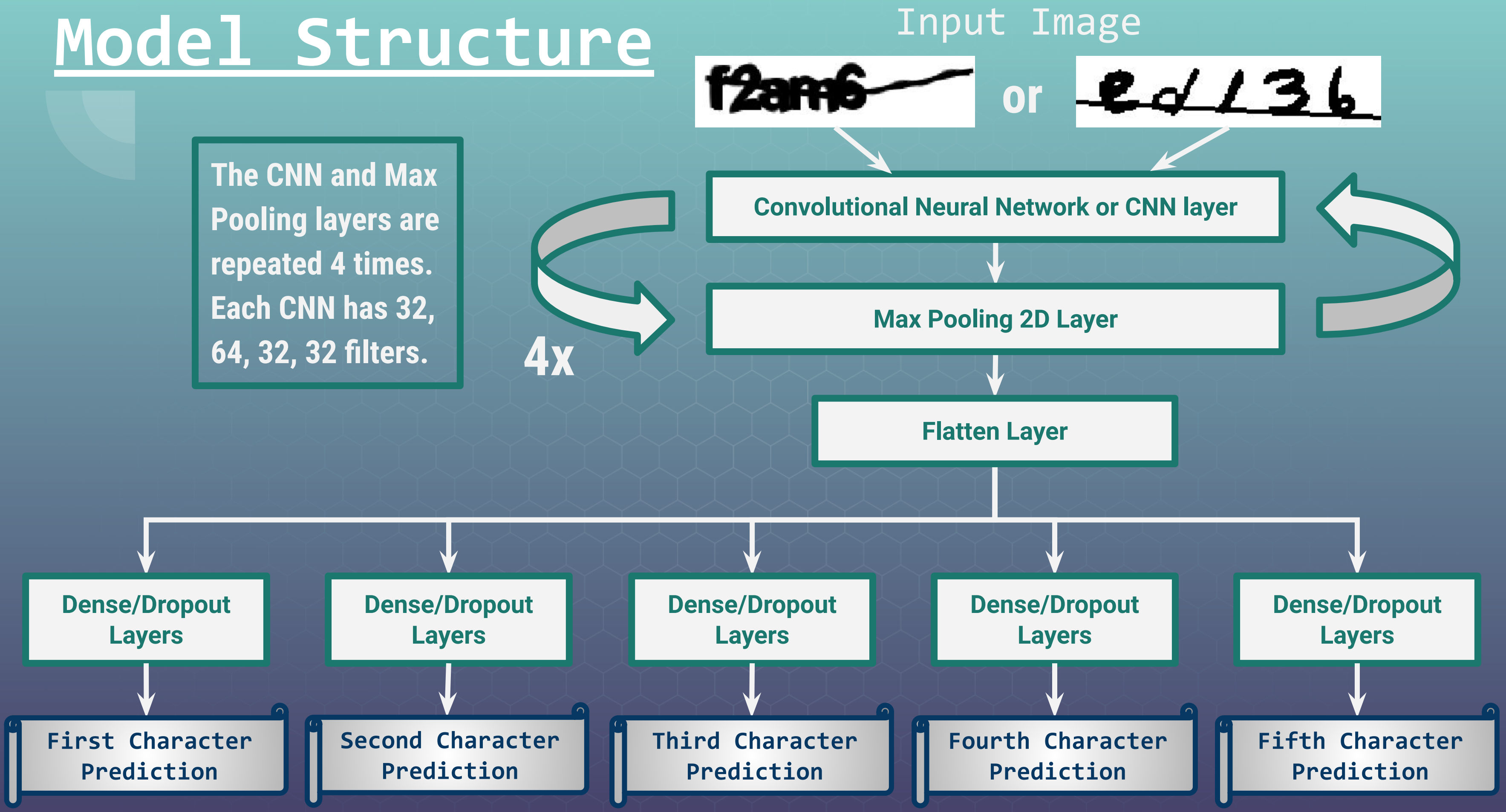
- The EMNIST dataset is an extension of the MNIST digits dataset that has alphabetical letters.

- Hand drawn CAPTCHAs were created by concatenating random individual handwritten numbers and letters.
- Then a line was added at a random angle



Model Structure

The CNN and Max Pooling layers are repeated 4 times. Each CNN has 32, 64, 32, 32 filters.



Input Image

or

Convolutional Neural Network or CNN layer

Max Pooling 2D Layer

4x

Flatten Layer

Dense/Dropout Layers

Dense/Dropout Layers

Dense/Dropout Layers

Dense/Dropout Layers

Dense/Dropout Layers

First Character Prediction

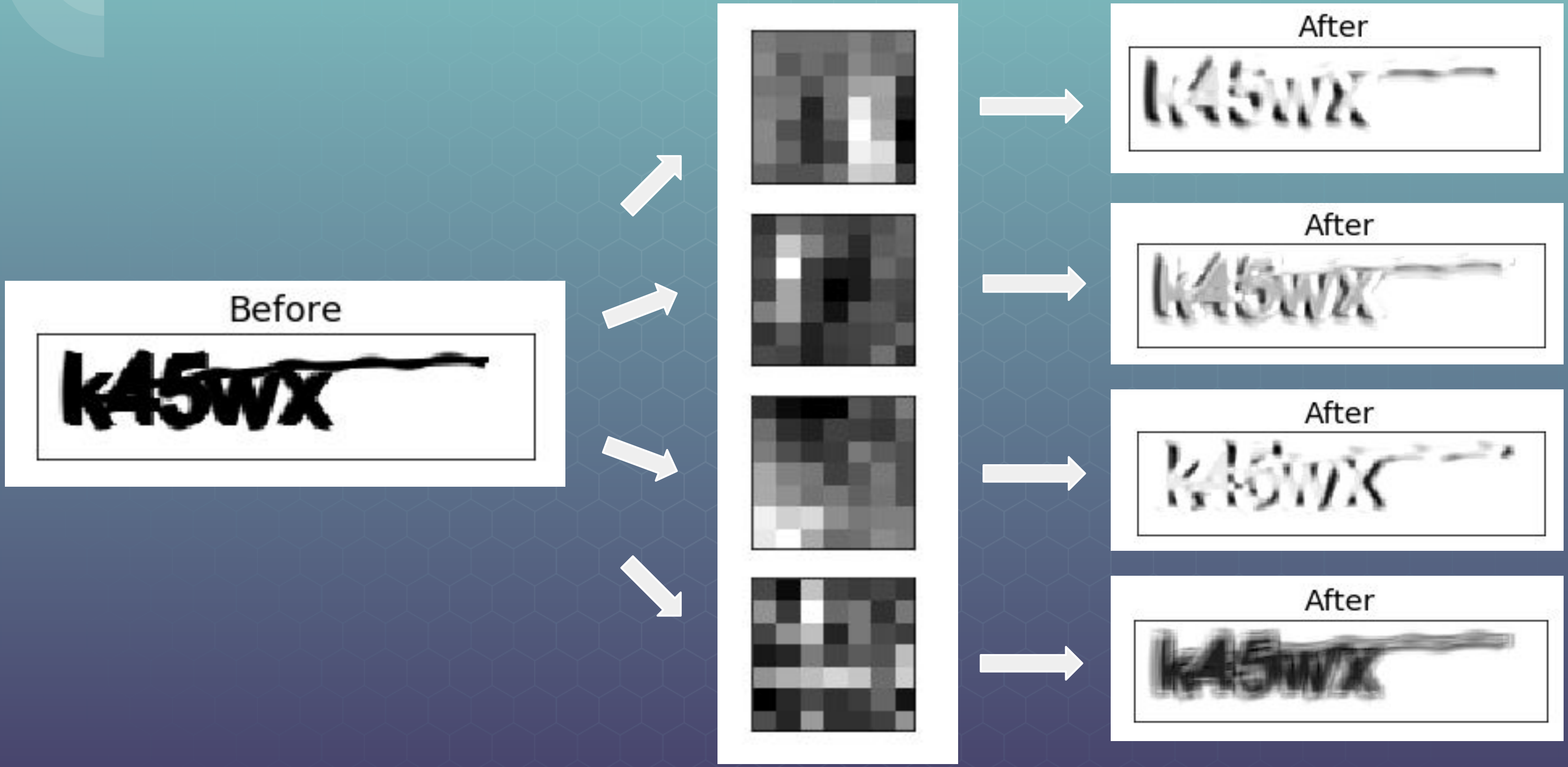
Second Character Prediction

Third Character Prediction

Fourth Character Prediction

Fifth Character Prediction

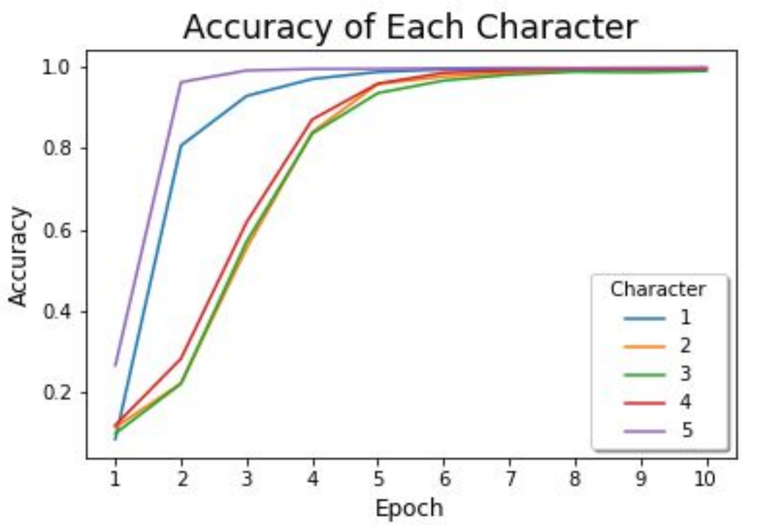
What’s happening in the CNN layers?



Model Results

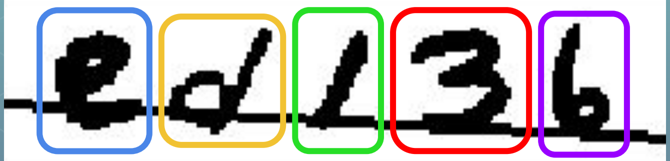
CAPTCHA Data

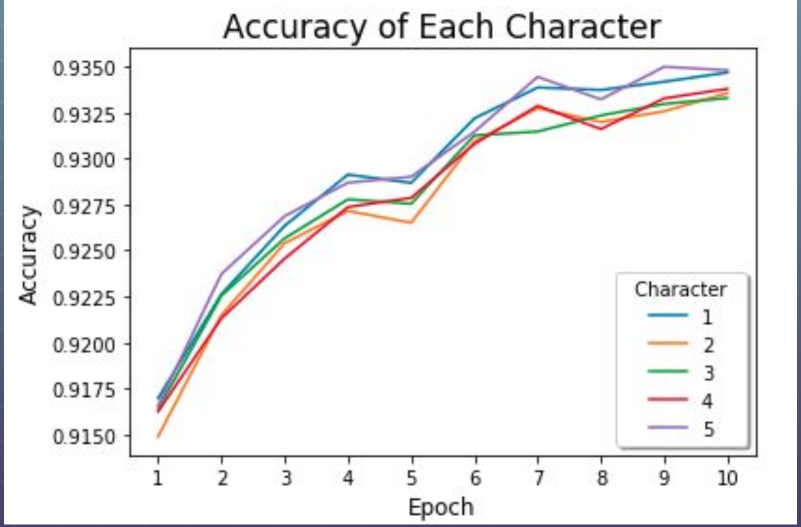
99.8%, 99.5%, 98.9%, 99.6%, 99.8%



Hand Drawn CAPTCHA Data

93.5%, 93.4%, 93.3%, 93.4%, 93.5%

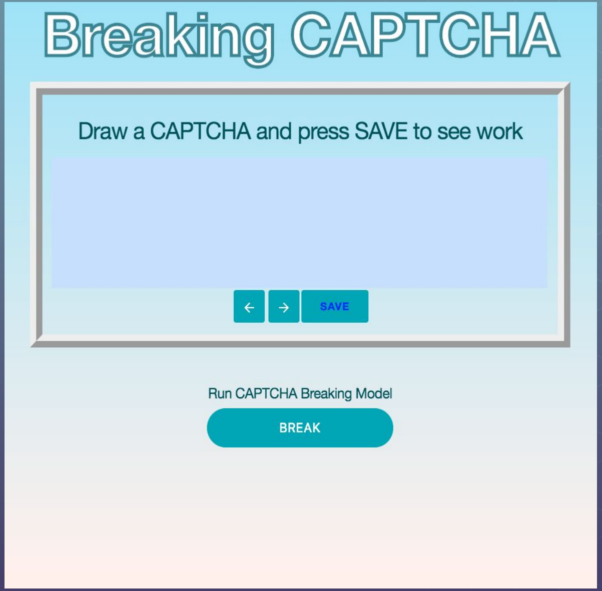




Web App

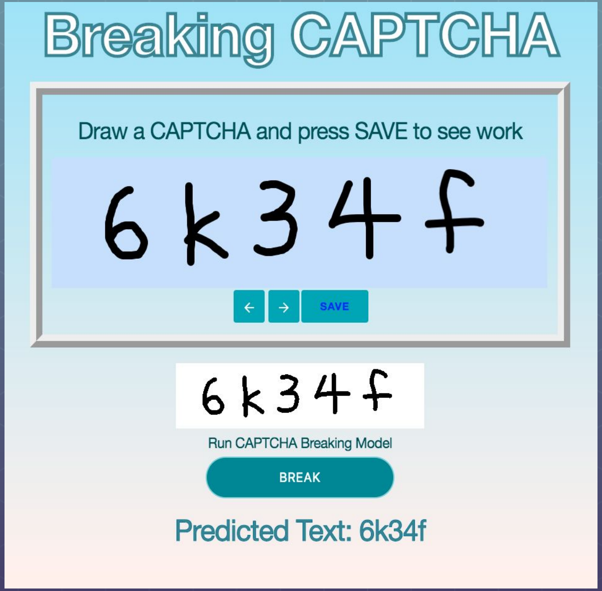
- Hand draw a CAPTCHA for the model to predict
- Press Save to see what you’ve drawn so far.

- Once you’re satisfied, push BREAK to have the model predict.


Breaking CAPTCHA

Draw a CAPTCHA and press SAVE to see work

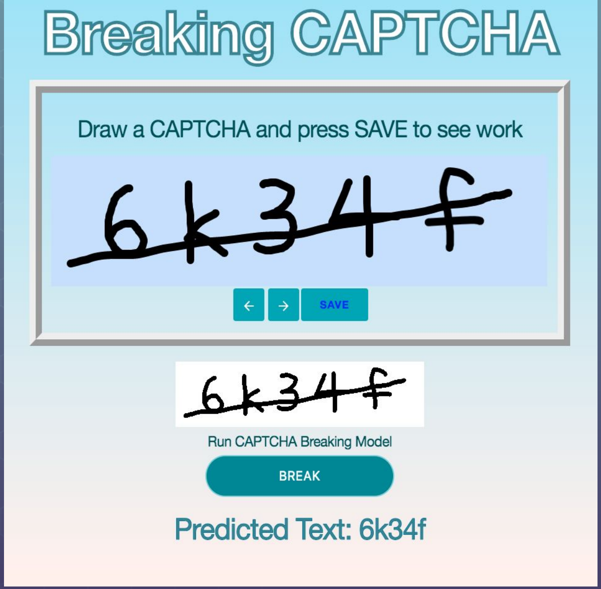
Run CAPTCHA Breaking Model


Breaking CAPTCHA

Draw a CAPTCHA and press SAVE to see work

Run CAPTCHA Breaking Model

Predicted Text: 6k34f


Breaking CAPTCHA

Draw a CAPTCHA and press SAVE to see work

Run CAPTCHA Breaking Model

Predicted Text: 6k34f

Conclusion

- Building models to predict CAPTCHA is dependent on the data it is trained on.
- This leads to websites creating new versions of CAPTCHAs to combat people training models to predict CAPTCHAs.
- Link to Web App

Tech Stack

amazon web services

Dash by plotly

pillow

The friendly PIL fork

CSS

HTML

TensorFlow

OpenCV

Keras

jupyter

Thank you for listening

Contact Information:

Phone:

(650) 804-8986

Email:

rzhao97@gmail.com

Github:

https://github.com/rzhao97/

Linkedin:

https://www.linkedin.com/in/rzhao97/

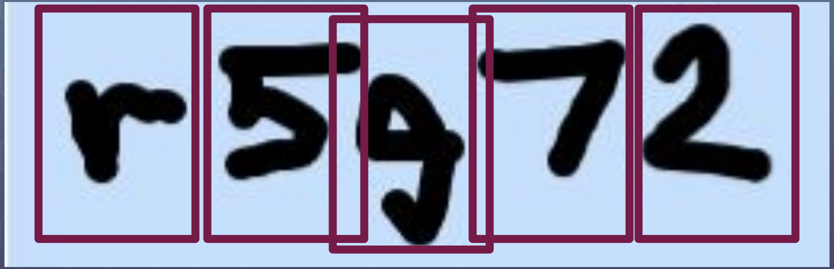
Transition to Handwritten CAPTCHA

- CAPTCHA breaking models are very dependent on the data it is trained on, if enough CAPTCHA data exists, it can be broken

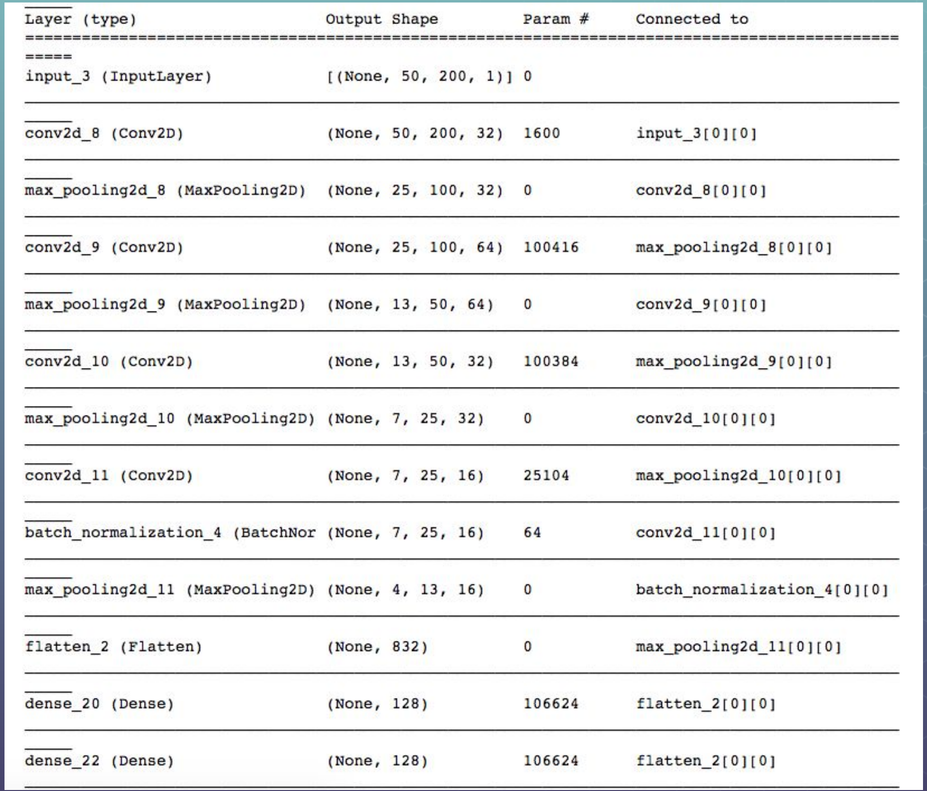
- Using handwritten numbers and letters data as an example

- I made a web app that can break handwritten CAPTCHAs in the similar fashion

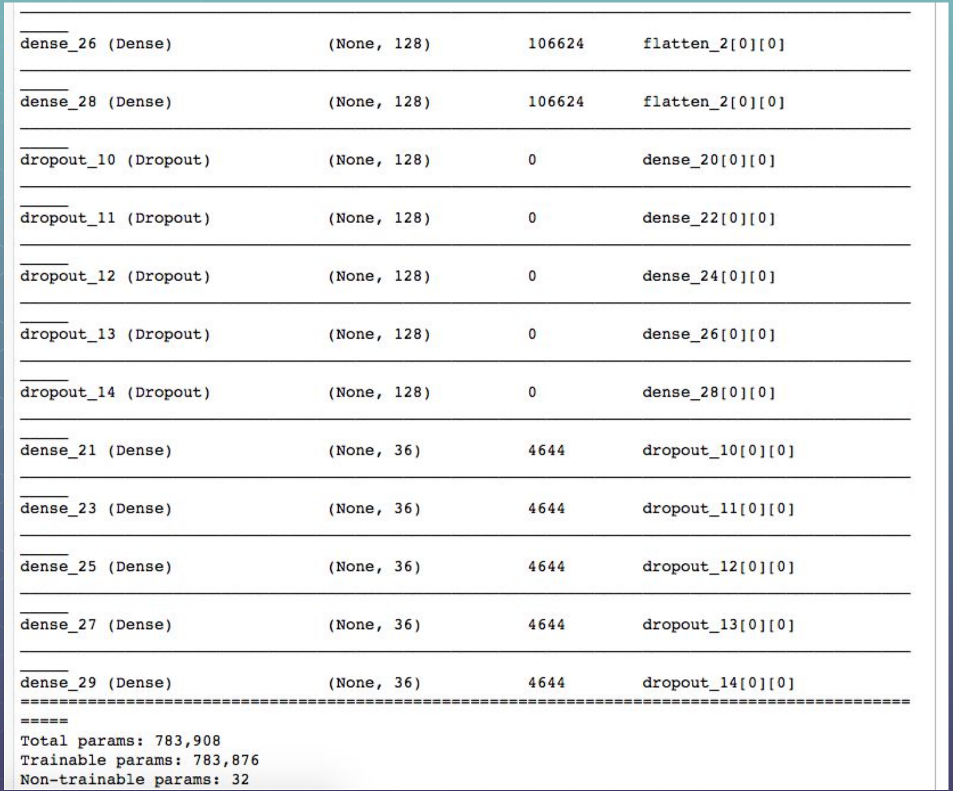
- The CAPTCHA will be split into individual characters then each character will be identified



Model Summary from Keras

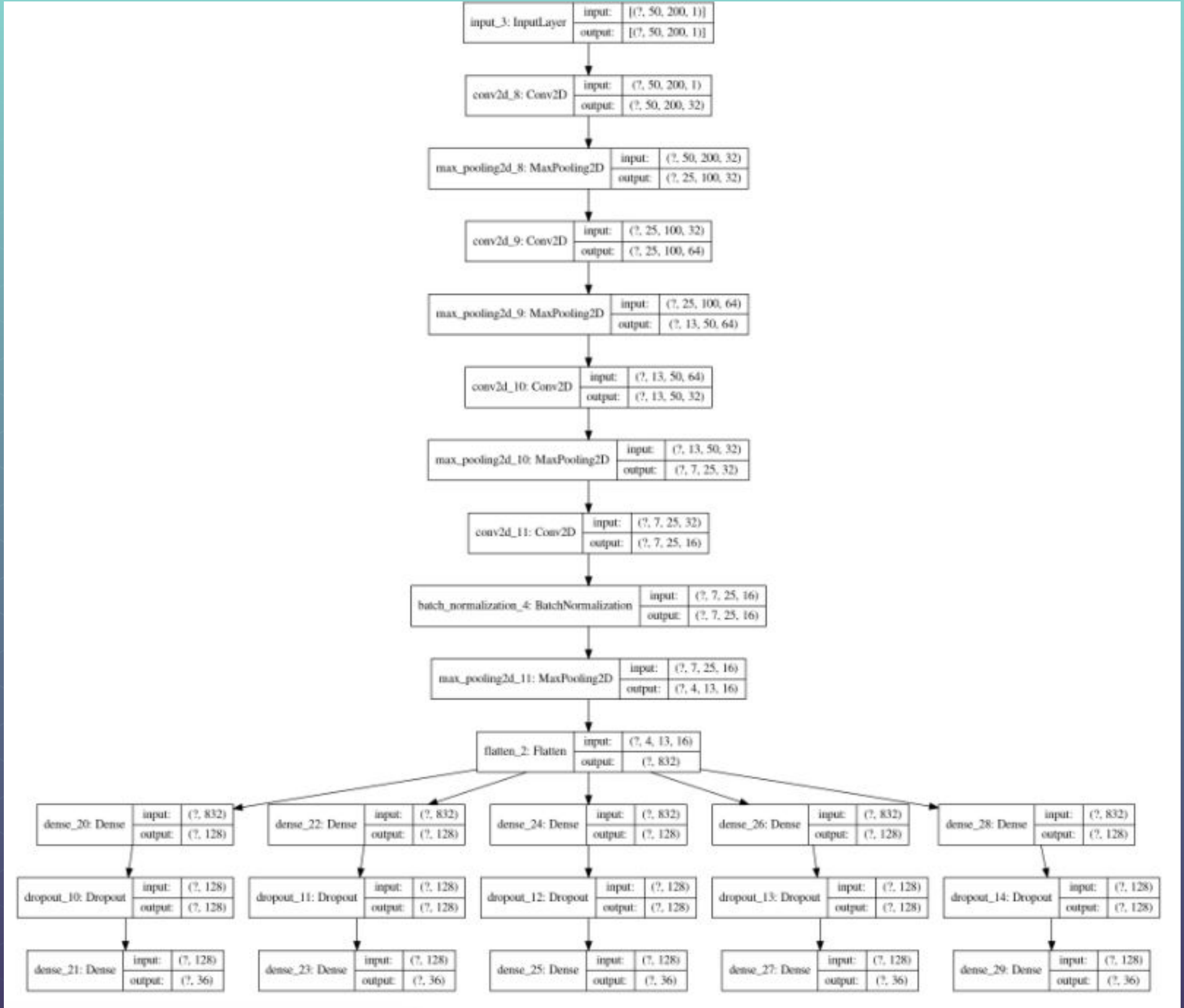


```
_____
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
=====
input_3 (InputLayer)            [(None, 50, 200, 1)] 0
__________________________________________________________________________________________________
_____
conv2d_8 (Conv2D)               (None, 50, 200, 32)  1600        input_3[0][0]
__________________________________________________________________________________________________
_____
max_pooling2d_8 (MaxPooling2D)  (None, 25, 100, 32)  0           conv2d_8[0][0]
__________________________________________________________________________________________________
_____
conv2d_9 (Conv2D)               (None, 25, 100, 64)  100416      max_pooling2d_8[0][0]
__________________________________________________________________________________________________
_____
max_pooling2d_9 (MaxPooling2D)  (None, 13, 50, 64)   0           conv2d_9[0][0]
__________________________________________________________________________________________________
_____
conv2d_10 (Conv2D)              (None, 13, 50, 32)   100384      max_pooling2d_9[0][0]
__________________________________________________________________________________________________
_____
max_pooling2d_10 (MaxPooling2D) (None, 7, 25, 32)    0           conv2d_10[0][0]
__________________________________________________________________________________________________
_____
conv2d_11 (Conv2D)              (None, 7, 25, 16)    25104       max_pooling2d_10[0][0]
__________________________________________________________________________________________________
_____
batch_normalization_4 (BatchNor (None, 7, 25, 16)    64          conv2d_11[0][0]
__________________________________________________________________________________________________
_____
max_pooling2d_11 (MaxPooling2D) (None, 4, 13, 16)    0           batch_normalization_4[0][0]
__________________________________________________________________________________________________
_____
flatten_2 (Flatten)             (None, 832)          0           max_pooling2d_11[0][0]
__________________________________________________________________________________________________
_____
dense_20 (Dense)                (None, 128)          106624      flatten_2[0][0]
__________________________________________________________________________________________________
_____
dense_22 (Dense)                (None, 128)          106624      flatten_2[0][0]
__________________________________________________________________________________________________
```



```
__________________________________________________________________________________________________
_____
dense_26 (Dense)                (None, 128)          106624      flatten_2[0][0]
__________________________________________________________________________________________________
_____
dense_28 (Dense)                (None, 128)          106624      flatten_2[0][0]
__________________________________________________________________________________________________
_____
dropout_10 (Dropout)            (None, 128)          0           dense_20[0][0]
__________________________________________________________________________________________________
_____
dropout_11 (Dropout)            (None, 128)          0           dense_22[0][0]
__________________________________________________________________________________________________
_____
dropout_12 (Dropout)            (None, 128)          0           dense_24[0][0]
__________________________________________________________________________________________________
_____
dropout_13 (Dropout)            (None, 128)          0           dense_26[0][0]
__________________________________________________________________________________________________
_____
dropout_14 (Dropout)            (None, 128)          0           dense_28[0][0]
__________________________________________________________________________________________________
_____
dense_21 (Dense)                (None, 36)           4644        dropout_10[0][0]
__________________________________________________________________________________________________
_____
dense_23 (Dense)                (None, 36)           4644        dropout_11[0][0]
__________________________________________________________________________________________________
_____
dense_25 (Dense)                (None, 36)           4644        dropout_12[0][0]
__________________________________________________________________________________________________
_____
dense_27 (Dense)                (None, 36)           4644        dropout_13[0][0]
__________________________________________________________________________________________________
_____
dense_29 (Dense)                (None, 36)           4644        dropout_14[0][0]
==================================================================================================
=====
Total params: 783,908
Trainable params: 783,876
Non-trainable params: 32
```

Model Structure from Keras



input_3: InputLayer | input: [(?, 50, 200, 1)] | output: [(?, 50, 200, 1)]

conv2d_8: Conv2D | input: (?, 50, 200, 1) | output: (?, 50, 200, 32)

max_pooling2d_8: MaxPooling2D | input: (?, 50, 200, 32) | output: (?, 25, 100, 32)

conv2d_9: Conv2D | input: (?, 25, 100, 32) | output: (?, 25, 100, 64)

max_pooling2d_9: MaxPooling2D | input: (?, 25, 100, 64) | output: (?, 13, 50, 64)

conv2d_10: Conv2D | input: (?, 13, 50, 64) | output: (?, 13, 50, 32)

max_pooling2d_10: MaxPooling2D | input: (?, 13, 50, 32) | output: (?, 7, 25, 32)

conv2d_11: Conv2D | input: (?, 7, 25, 32) | output: (?, 7, 25, 16)

batch_normalization_4: BatchNormalization | input: (?, 7, 25, 16) | output: (?, 7, 25, 16)

max_pooling2d_11: MaxPooling2D | input: (?, 7, 25, 16) | output: (?, 4, 13, 16)

flatten_2: Flatten | input: (?, 4, 13, 16) | output: (?, 832)

dense_20: Dense | input: (?, 832) | output: (?, 128)

dense_22: Dense | input: (?, 832) | output: (?, 128)

dense_24: Dense | input: (?, 832) | output: (?, 128)

dense_26: Dense | input: (?, 832) | output: (?, 128)

dense_28: Dense | input: (?, 832) | output: (?, 128)

dropout_10: Dropout | input: (?, 128) | output: (?, 128)

dropout_11: Dropout | input: (?, 128) | output: (?, 128)

dropout_12: Dropout | input: (?, 128) | output: (?, 128)

dropout_13: Dropout | input: (?, 128) | output: (?, 128)

dropout_14: Dropout | input: (?, 128) | output: (?, 128)

dense_21: Dense | input: (?, 128) | output: (?, 36)

dense_23: Dense | input: (?, 128) | output: (?, 36)

dense_25: Dense | input: (?, 128) | output: (?, 36)

dense_27: Dense | input: (?, 128) | output: (?, 36)

dense_29: Dense | input: (?, 128) | output: (?, 36)