Captcha Recognition

Motivation:

Recognizing digits in a real world can help improve navigation services and building accurate maps. It is central to a variety of emerging machine learning and computer vision applications.

A sample application of interest is the problem of identifying house numbers posted on the fronts of buildings. Recognizing characters is much more difficult than MNIST in natural scenes. We get image obtained from house numbers in Google Street View images and predict the written number.

Datasets used: 74k char, CIFAR-10, SVHN

Project Formulation:
This project consists of three stages. Image segmentation and two machine learning(ML) problems: Identifying if the image has a digit and Identifying the digit in the image.

.

Approach:

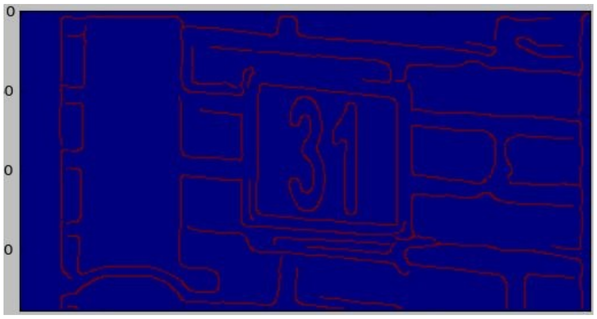
Stage 1 : Image segmentation

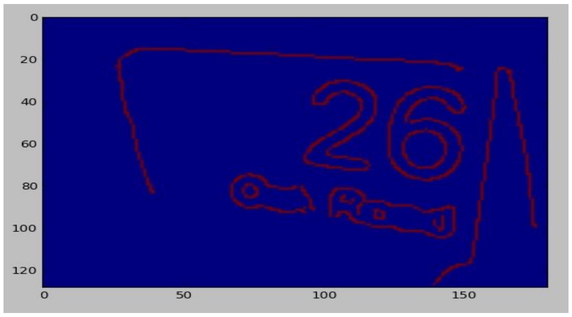
Here we show the various stages of preprocessing for two sample images

1) The images we chose are:

2) Image segmentation phase:

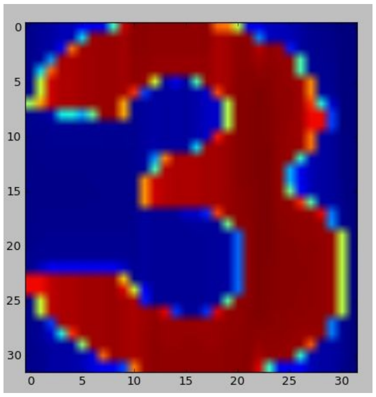
In this phase, we clean and create gray scale of the images. Then we apply otsu’s method to improve image content. We further use canny’s method to find boundaries along edges. After going through this process, the images will look as follows:

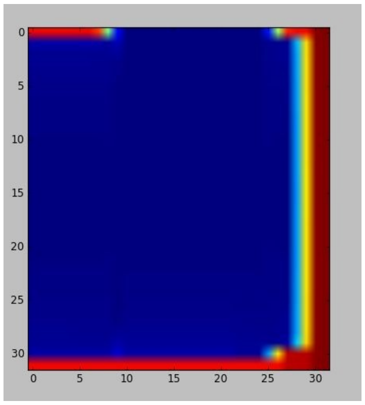


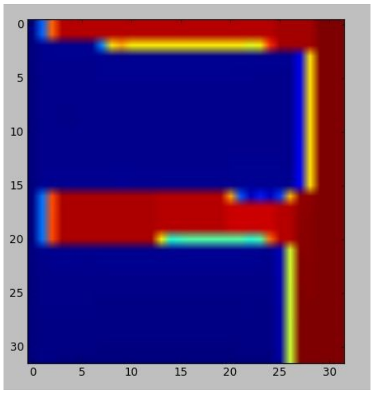


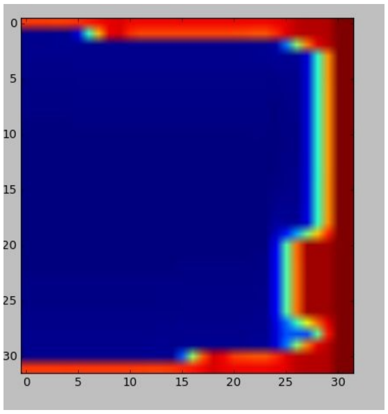
3) Possible digit segments:

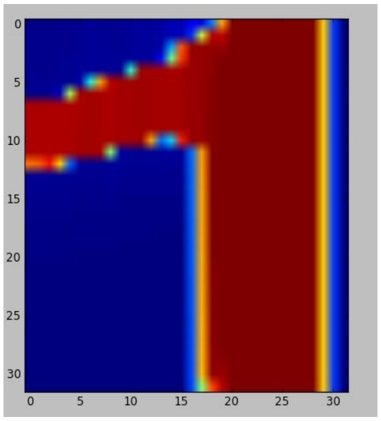
Here we try to segment objects in this image using skimage library’s built-in functionalities, label and border along with logical xor. We reject few segments based on certain criteria like area<10, etc. By putting the above image 1 throgh this process, we identify these six as possible digit segments

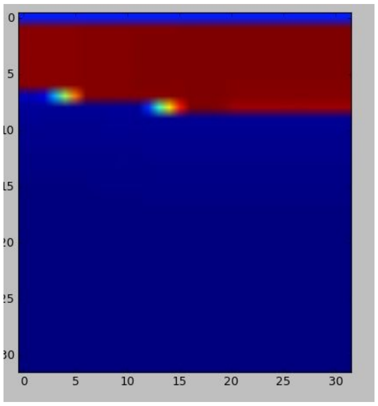




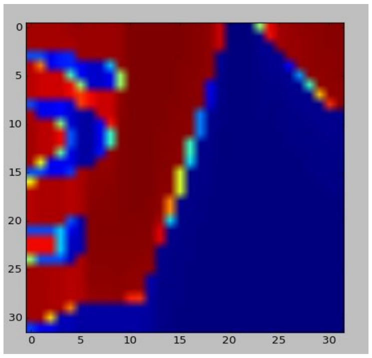


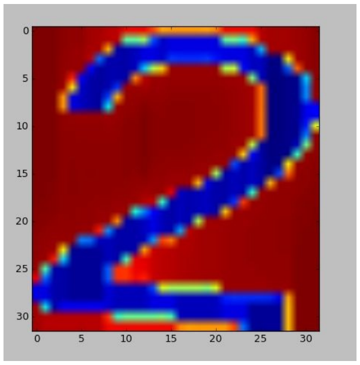


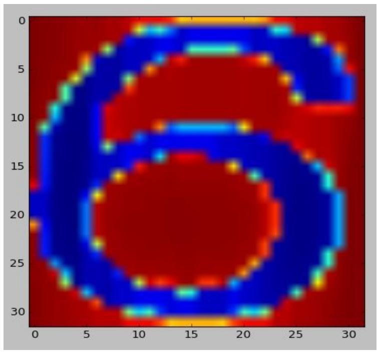


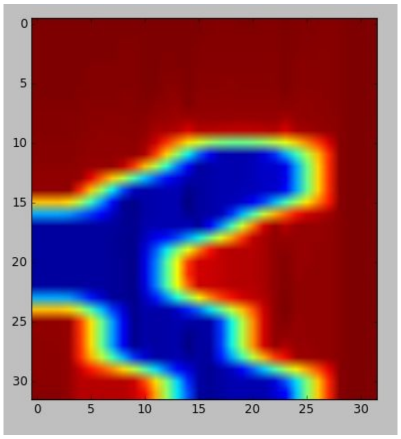


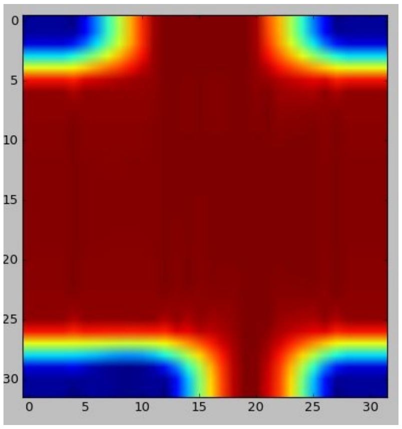
For image 2, the digit segments are:











We call them stage 2 and stage 3 respectively from here on.

Stage 2 : Identifying whether an image segment is a digit or not

To get training dataset for stage 2, we make use of Cifar-10 and 74k char datasets.The Cifar and 74k char dataset is available in pickle format. Which we were not able to load it in our computer due to memory error. So we converted it into a .mat file which we observed to be relatively faster. We require only digits for our model, so we use the images with digit labels only from the 74k char dataset which happens to be of size ~12k images. We classify these as 1’s in our training set. We pick 12k images randomly from 74k images of cifar dataset and classiy them as 0’s in our training set (to ensure balance between 0’s and 1’s in the train data). We shuffle the data for future use (to avoid making monotone batches in gradient descent).

We tried three different models for stage1 : SVM with grayscale, CNN with grayscale and CNN with rgb. Some images perform better with SVM, some with CNN and some with both.

Model 1 : SVM with grayscale :

We used sklearn’s svm implementation. For the first example, SVM performed better than CNN.and gave output as 1 for first and fifth digit segments.

Model 2: CNN with grayscale :

We implemented CNN with grayscale using minibatch stochastic gradient descent algorithm using tensorflow. The input is 32x32x1 image. It is convoluted using [4,4,1,2] kernel. This convoluted image, 512 nodes is inputed into layer with 4 nodes and then 2 nodes. We used softmax probabilities for 0 and 1 in the last stage. For the second example, CNN performed better than SVM and gave output as 1 for second and third digits.

Model 3: CNN with rgb:

The model is highly unstable with one convolution layer. It tends to go to one side(all 0's or all 1's based on the data in the batch). So we did not proceed further.

Stage 3 : Give an image with a number in it we will predict the number using CNN.

In this the 32x32x3 images are inputted into a neural network model with 3 convolutions of kernel dimensions of [5,5,3,10], [5,5,10,15], [5,5,15,80] with stride of [1,2,2,1].
CNN is implemented using minibatch stochastic gradient descent algorithm using tensorflow. Each batch is of size 128 images.

Conclusion :

2. CNN is able to classify data better than SVM.
3. obtained ~80% test accuracy and ~85% train accuracy for this stage.

Literature Survey (Research Papers used for our project):

1. Reading Digits in Natural Images with Unsupervised Feature Learning By Netzer et. al
2. Multi-digit Number Recognition from Street View Imagery using Deep Convolutional Neural Networks By Ian J. Goodfellow et. al

Citations :

1. http://www.cs.toronto.edu/~kriz/cifar.html
2. http://ufldl.stanford.edu/housenumbers/
3. http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/