License Plate Recognition

Xi You ucspouch@bu.edu
Ziyu Shen shen99@bu.edu
Guangxing Ren rengx@bu.edu

Boston University Machine Learning Course Final Project

Github:https://github.com/renguangxing/CS542-finalproject

Abstract

In this paper, a license plate recognition algorithm based on the convolutional Neural Network is being proposed. A large number of vehicle license plate characters are used to test the performance of the algorithm. Result showed that the convolutional Neural Network is highly useable and efficient in image recognization.

Keywords: Convolutional Neural Network, Keras, Image Computing, OpenCV

1 Introduction

Image recognizations applications are gaining huge attention in recent years. For example, in the commercial area. Apple starts to apply Face ID to all its IOS device. Which allow users to more conveniently unlock their iPhone/IPad and more securely finish financial transactions. In the security Area, the Chinese government has a project called "The Web of The Sky" which place cameras in every corner of the street to monitor every move and every individual. The video recorded will get processed by a face identification system and outlaws will easily be located. In this project, we try to redevelop very popular Image recognizations called License plate recognization. This system allows parking lot and police efficiently identify the car plate of the individual vehicle. We preprocessed and sliced the license plate to individual character and design and trained a Convolutional Neural Network to detect the characters.

2 Research and Possible Solution

We conduct research for the existing solution and there are three possible tracks

1. Convolutional Neural Network: CNN is widely used for image recognition for a long period of time. There are various well-developed network structure and framework/platform to utilize. This is the main reason why we choose CNN as a solution. Convolutional neuron network mainly consists of a convolutional layer, pooling layer, and fully connected layer. Convolutional layer basically takes filters on a single image and each filter picks different signal(area of the picture) in horizontal, vertical and diagonal directions. We choose CNN as our final method.
2. Edge Detection: Edge Detection is a popular image computing methods. Here is one of the team members CS585(Image and Video computing) report that illustrates the application of Edge Detection to detect license plate: http://cs-people.bu.edu/rengx/finalreport.html
3. Hidden Markov Model Hidden Markov Model is a statistical Markov model in which the system being modeled is assumed to be a Markov process with unobservable states. The hidden Markov model can be represented as the simplest dynamic Bayesian network.

Submitted to 33rd Conference on Neural Information Processing Systems (NeurIPS 2019). Do not distribute.

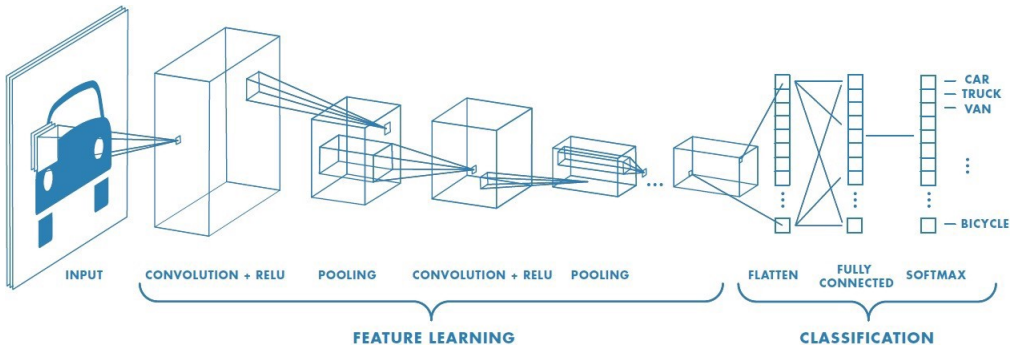

Figure 1: Convolutional Neural Network Layers

3 Convolution:

The aim of those filters is creating a map of each slice in the image that feature occurs. So convolutional networks perform a sort of search. During the search, a match is found, it will be mapped into a feature space where the location of the match will be saved. By repeating the above steps, the convolutional layer can record features of the single image in different directions. After convolutional layer, it might be passed into a nonlinear transform such as reLu or tanh, which compress input into a range between 1 and -1.

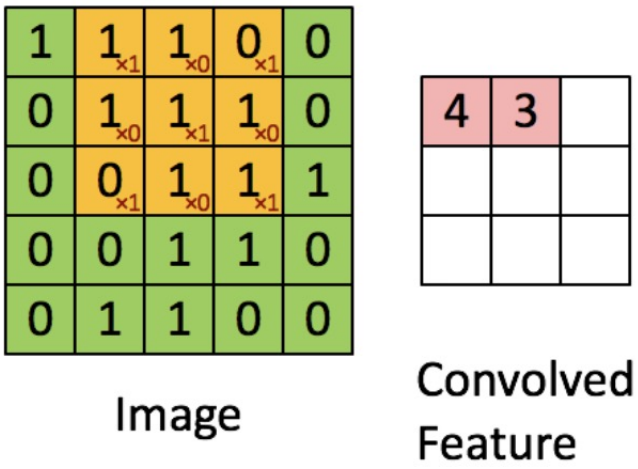

Figure 2: Convolution Filters

We have two 5*5 two 3*3 convolution layer with relu as activation

4 Pooling layer:

Pooling layer has various kinds, such as maxPooling, average and downsample. In pooling layer, maps will be applied a slice/patch once a time. For example, maxPooling will take the largest value in the slice/patch and discard other information in maps. It is kind of compressing maps into smaller dimensions and save key feature at the same time. The fully connected layer will classify output on each node based on weights.

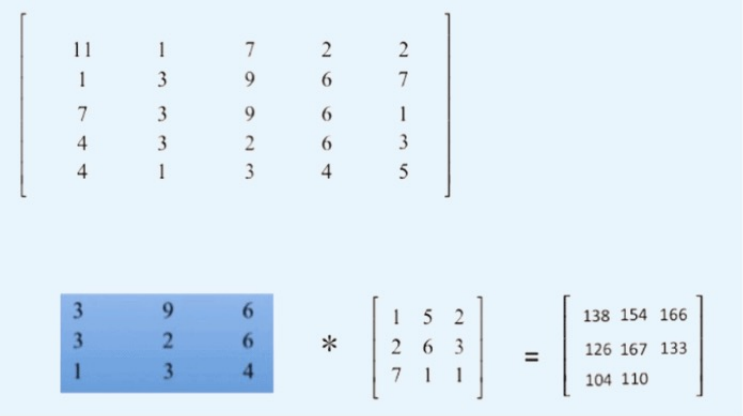

Figure 3: Max Pooling

We have two 3*3 max-pooling layers

5 Our Library:

5.1 Keras:

Keras is chosen as a developing framework for this project. keras is well developed and user-friendly for starters. It provides various APIs. As a result, we can more focus on how to improve our network in high level rather than in debugging our network structure. Keras is indeed more readable and concise, allowing us to build first end-to-end deep learning models faster while skipping the implementation details. Kears is built on top of Tensorflow, which is widely used for deep learning.

5.2 OpenCV:

OpenCV is a library of programming functions mainly aimed at real-time computer vision. We apply methods like Grey Scale and threshold in OpenCV to preprocess our image. Grey Scale and threshold change image from 3 channel(RGB) to 1 channel.

6 Workflow Diagram:

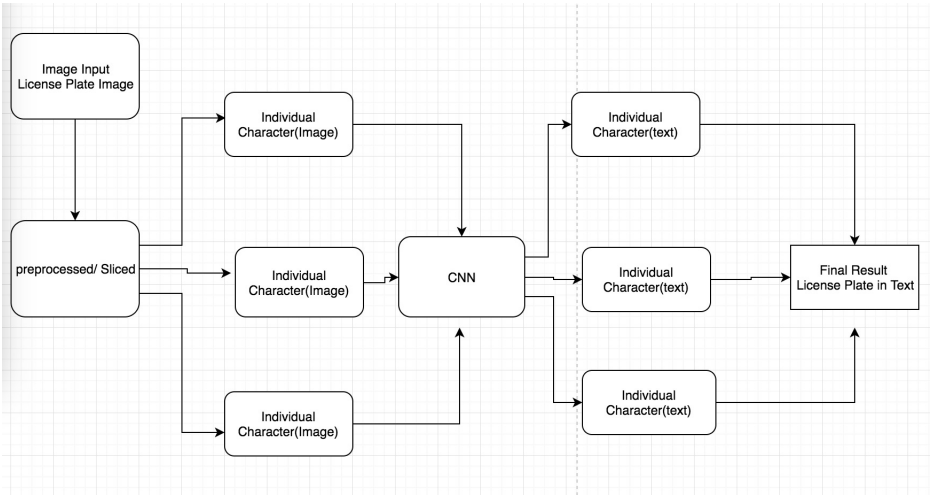

Figure 4: Max Pooling

7 Input Data:

We have over 2000 license plate image as our data. The images are being processed. The name of the image file is the license plate in text. The data are being separate as Training, Testing, and Validation. This is the sample of the input:

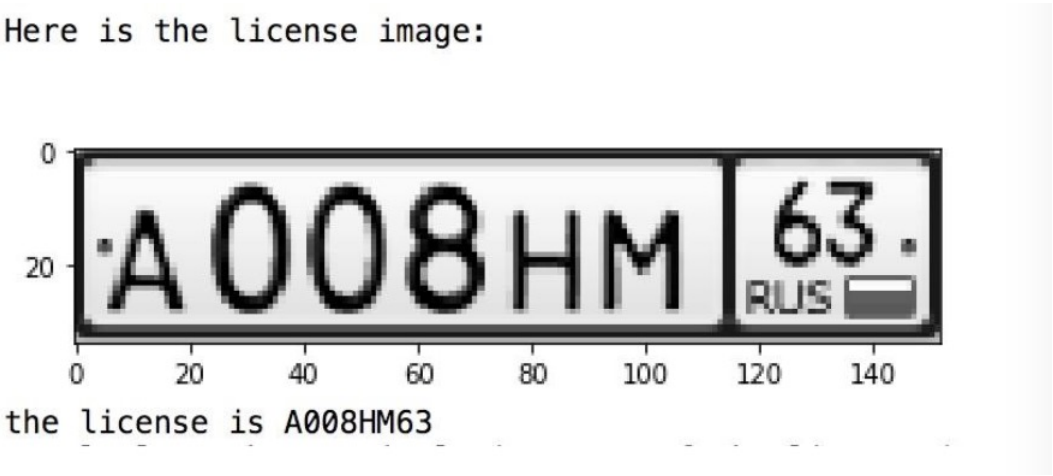

Figure 5: Input Data

One Challenge of sliced the plate image to Character is on the right side of the plate. On top of the right side is 63 and the bottom is RUS. We only need 63 in our result

8 Pre Processing:

8.1 Step-one:

We calculate the vertical histogram of the licensed image. then by the $get_loc_x()$ function, we can get each font's x-axis location so we can dig out the font patches

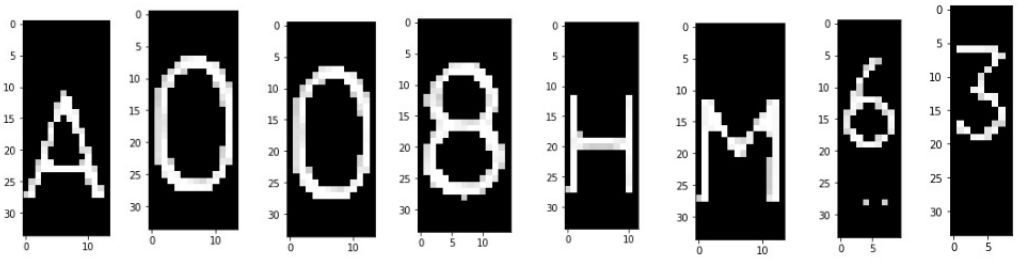

Figure 6: Step One of Pre Processing

8.2 Step-two:

However, We still need to get the y-axis location of each font Then, we calculate the horizontal of the licensed image the location of each font have already done, and resize each font patch into size(20,20)

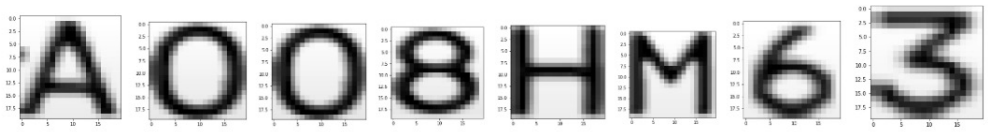

Figure 7: Step two of Pre Processing

Build The Convolutional Neural Network Using Keras

```
keras_model = Sequential()
keras_model.add(Conv2D(32, (5, 5), input_shape=(20, 20, 1),padding = 'SAME'))
keras_model.add(Activation('relu'))
keras_model.add(Conv2D(32, (5, 5)))
keras_model.add(Activation('relu'))
keras_model.add(MaxPooling2D(pool_size=(2, 2)))
keras_model.add(Conv2D(64, (3, 3)))
keras_model.add(Activation('relu'))
keras_model.add(Conv2D(64, (3, 3)))
keras_model.add(Activation('relu'))
keras_model.add(MaxPooling2D(pool_size=(2, 2)))
keras_model.add(Flatten())
keras_model.add(Dense(200))
keras_model.add(Activation('relu'))
keras_model.add(Dropout(0.4))
keras_model.add(Dense(200))
keras_model.add(Activation('relu'))
keras_model.add(Dense(36))
```

Figure 8: Step two of Pre Processing

8.3 Training Process:

```
 testing accuracy 98.5%
Step 1620:    (2019-04-20 22:02:31.499246)
 training nat accuracy 100.0%
 testing accuracy 99.05%
Step 1630:    (2019-04-20 22:02:33.634676)
 training nat accuracy 99.0%
 testing accuracy 98.64%
Step 1640:    (2019-04-20 22:02:35.912805)
 training nat accuracy 98.0%
 testing accuracy 98.5%
Step 1650:    (2019-04-20 22:02:38.259610)
 training nat accuracy 99.0%
 testing accuracy 98.91%
Step 1660:    (2019-04-20 22:02:40.770506)
 training nat accuracy 98.0%
 testing accuracy 97.96%
Step 1670:    (2019-04-20 22:02:43.254851)
 training nat accuracy 100.0%
 testing accuracy 98.64%
Step 1680:    (2019-04-20 22:02:45.677979)
 training nat accuracy 100.0%
 testing accuracy 98.1%
Step 1690:    (2019-04-20 22:02:48.082490)
 training nat accuracy 100.0%
 testing accuracy 98.23%
Step 1700:    (2019-04-20 22:02:50.535391)
 training nat accuracy 99.0%
 testing accuracy 98.91%
Step 1710:    (2019-04-20 22:02:52.984489)
 training nat accuracy 98.0%
 testing accuracy 97.82%
```

Figure 9: Step two of Pre Processing

8.4 Sample output and accuracy of the evaluation:

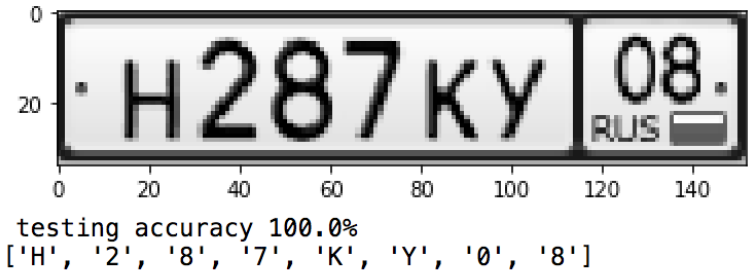

Figure 10: Step two of Pre Processing

9 Conclusion and analysis:

This paper presents a Convolutional Neural Network method to reproduce one of the most useful image recognizations. Out of our surprise, The result is extremely well. We achieve 98 percent accuracy rate. We fully appreciate Keras for letting us easily manipulate our net work. The overall trainng time is around 5 mins. After the traning, we store our model into a H5 file for later convenient. Special thanks go to our TAs Peilun and Xiao Wang for providing the sample Keras Code.

10 Future Work:

For the next step, we want to integrate Edge Detection with convolutional Neural Network. We wish we can input a street image with multiple, locate and identify every car plate.

Citation

[1] Kieran. & Xiao Wang Keras tutorial
https://d1b10bmlvqabco.cloudfront.net/attach/jr1qbl2ujdp76/ijomhtg34rp3n6/judduibdjzc9/keras_in_tf.py. Boston , MA: Boston University.

[2] Dr. Irwin, & M. Cohen, D. Dr. Darryl Plecas Licence Plate Recognition System in British Columbia
https://www.researchgate.net/publication/236888959_Automatic_Number_Plate_Recognition_System_ANPR_A_Survey/
/InternationalJournalofComputerApplications

[3] David J. Roberts. & Meghann Casanova Automated License Plate Recognition (ALPR) Use by Law Enforcement: Policy and Operational Guide(2012)
https://www.ncjrs.gov/pdffiles1/nij/grants/239605.pdf

[4] Prabhu Understanding of Convolutional Neural Network (CNN) — Deep Learning, https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148