High-Speed Logo Scan Algorithm using Neural Networks

Unni Krishnan Ambady

Singapore

Building and Construction Authority

Abstract

This project is an experiment to design, develop and test a set of Neural Networks to detect logos on print media. Two different types of networks are designed and tested, and the results are compared. To deploy them on a real-time basis, an improved design is proposed that can optimise the GPU usage time. The networks are concurrently activated on different GPUs and each network will have different depths in terms of hidden layers and hence differ in execution periods. The time taken to process an image, will be determined by the shortest time taken by whichever network that successfully finishes the task first. A mechanism using a "strobe" signal is proposed to communicate between the concurrently running multiple GPUs and deemed as the advanced feature in this project. This feature will allow some of the GPUs to stop early and thus reduce the total computational cost.

1. Introduction

Professional printed advertising designers avoid using QR codes as far as possible because it is not pleasant to put the black & white patches on an expensive advertisement on print media. Instead the advertisers resolve to alternative solutions such as displaying the full url of their website, Twitter, Instagram or Facebook as a means for the customers to follow-up. Scanning company logos to get directed to their webpage will be the most ideal solution in this case. But no such commercial implementation is available as of today, which can reach the performance level of QR Codes. This project has developed a solution using deep Convolutional Neural Network (CNN). We have designed a CNN which can perform in a fast and efficient manner to achieve better performance in terms of speed as compared to any existing [1] commercial solutions in the open market.

2. Related Works

The design of the Convolutional Neural Network (CNN) is based on the YOLO [2] and DeepFace [3] articles are sources of inspiration in designing the CNN for this project. WebLogo-2M: Scalable Logo Detection by Deep Learning [4] is an inspiration on the methodology for getting the data from internet for training the networks.

3. Dataset

The data required for this project is downloaded from the web using a web-crawler application (google-image-downloader). There are 6 sets of different company logos as shown in table-1 that are used to train the network.

Apple	BMW	Mc Donald's	Nike	Stanford	Wendy's

Table 1: The set of logos used for training

The data is downloaded in jpeg format using google-image-downloader application. A standalone python program is developed to rename the data files in the format <id>_LOGO_<company>.jpg; where <id> is a number 0 to 5 and <company> is the name of the logo as listed in table-1. While writing, the image size is converted to 64 x 64 pixels, maintaining the 3-colours. In each category there are 200 images, totalling 1200 images for this experiment. Further to this, the data is sorted to "train", "dev" and "test" sets (864, 216 and 120 respectively). The data is compressed into hdf5 file format for ease of uploading to Amazon Web Services (AWS) where the testing is performed.

It is assumed that a given image contains only one logo of a company and occupies at least 40% of the total image area. Wherever the logos downloaded are detected as duplicate, thus reducing the total count, data augmentation is done by zooming, shift centre and rotate clockwise/anti-clockwise up to 15^o; avoiding flipping at all times.

Since the data and code size is relatively small, the Jupiter notebook is activated on the AWS and its interface accessed from the PC (or Mac Pro).

From the edge device which takes photos of images the data types can further be classified into 3 types (Barcodes, QR codes and Logos) and all three in the following example represents 'Stanford'. These can be concurrently processed by three different applications and the first of the three that finishes the decoding will decide the total execution period.

Bar code for "www.stanford.edu"	QR code for "www.stanford.edu"	Logo
www.stanford.edu		
Barcode reader	QR code reader	CNN for Logos

Table:2 The edge device may have multiple scenarios.

Note: This project focuses only on the Neural Network development for Logo Identification

4. Methods

Figure-1 shows the overall system architecture. Even though the 3 networks demonstrated below can be merged into one, the execution time will be equal for all types of inputs. Here, the idea of this architecture is to mix networks of heterogeneous types with varying execution periods.

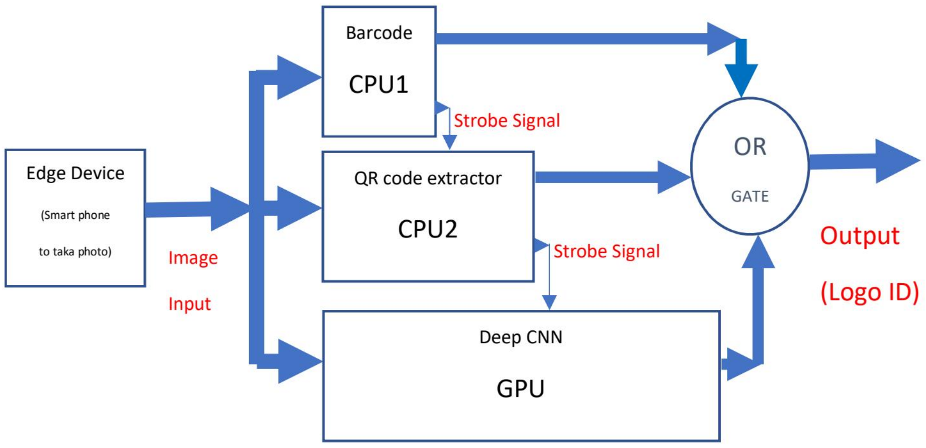


Figure 1: System Architecture

4.1 Network Architecture

In this project two different models are developed and tested, and the results are compared. The first model is a fully connected neural network with 3-hidden layers.

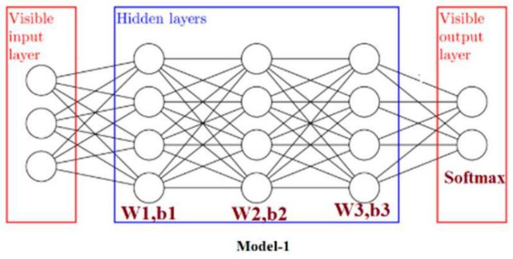


Figure 2: Model 1, using Fully Connected Neural Network

The second network tested is a Convolutional Neural Network (CNN). In order to improve the performance, in terms of execution period, the following minimum layers are set. This design reduces the execution period, giving better test- accuracy for CNN as compared to the fully connected neural network. Another advantage of this design is that it is possible to increase the size of the image to 256 x 256 (or even higher) without deteriorating its performance in terms of speed and achive better accuracy.

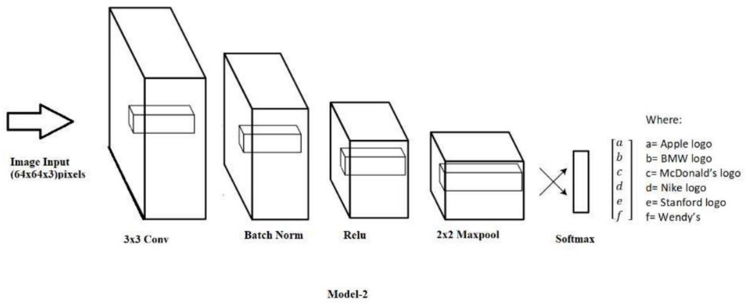
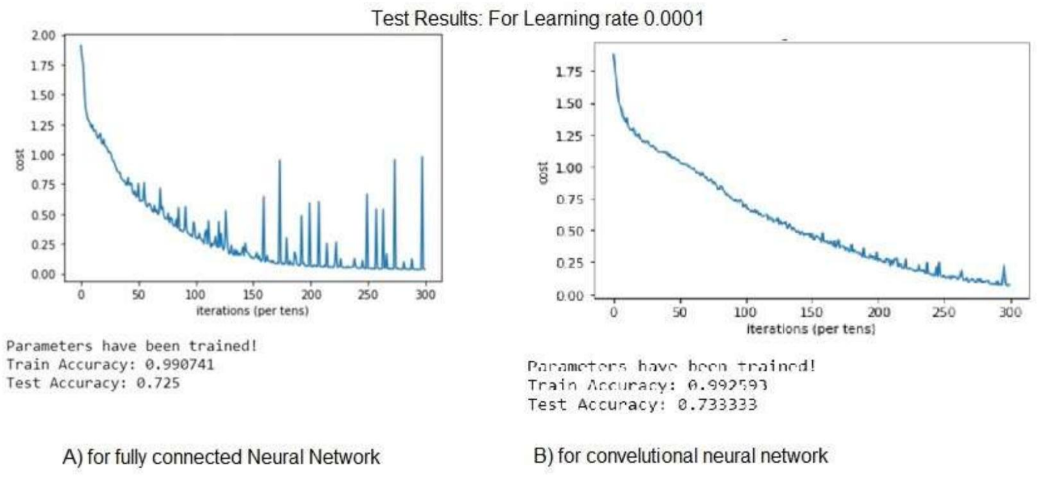


Figure 3: Model-2 using Convolutional Neural Network (CNN)

Results & Discussions

The system is tested by tuning the hyperparameter learning rate and for the 2- different networks the results are compared as in figure -4

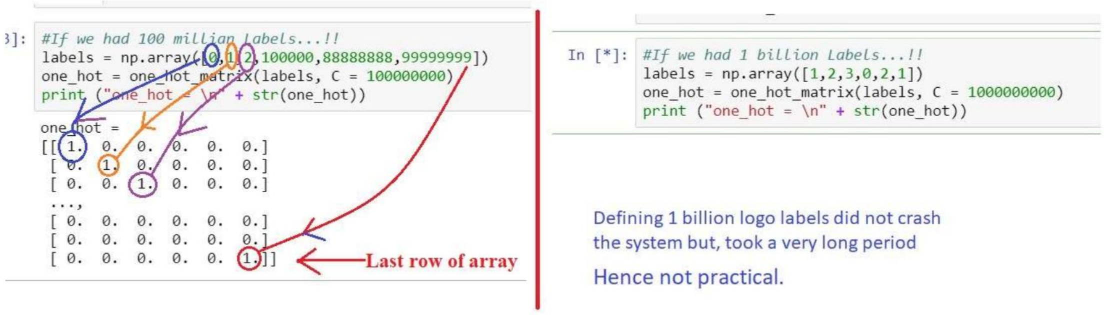


```
Parameters have been trained!
Train Accuracy: 0.990741
Test Accuracy: 0.725
```

```
Parameters have been trained!
Train Accuracy: 0.992593
Test Accuracy: 0.733333
```

Figure 4: Test Results and comparison of results

The performance of the Model -2 (CNN) is better than Model-1 in terms of test accuracy. Further, the CNN model is tested by increasing the resolution of the images to 255 x 255 and 3 colours. The time taken for execution was of the same order even with increase in data input size. Therefore, it may be concluded that CNN network with a SoftMax at the last stage is a better choice.



```
]: #If we had 100 million Labels...!!
   labels = np.array([0,1,2,100000,88888888,99999999])
   one_hot = one_hot_matrix(labels, C = 100000000)
   print ("one_hot = \n" + str(one_hot))

   one_hot =
   [[ 1.  0.  0.  0.  0.  0.]
    [ 0.  1.  0.  0.  0.  0.]
    [ 0.  0.  1.  0.  0.  0.]
    ...,
    [ 0.  0.  0.  0.  0.  0.]
    [ 0.  0.  0.  0.  0.  0.]
    [ 0.  0.  0.  0.  0.  1.]]
```

Last row of array

```
In [*]: #If we had 1 billion Labels...!!
        labels = np.array([1,2,3,0,2,1])
        one_hot = one_hot_matrix(labels, C = 1000000000)
        print ("one_hot = \n" + str(one_hot))
```

Defining 1 billion logo labels did not crash the system but, took a very long period

Hence not practical.

Figure 5: Capacity stress-test using 100 million output vectors.

Defining 100 million logo labels were successfully tested and hence it is concluded that this program can be practically extended to 100 million logos. Defining 1 billion logo labels did not crash the system but, took a very long period just to define the output vector and hence concluded as not practical in the given test setup.

5. Conclusion and Future Work

With successful implementation and testing of a CNN, the next step will be:

- Port the application as an iOT using NVIDIA Jetson Nano
- Bundle it as an API on AWS so that the smart phone application can access it conveniently
- Implement deeper CNN to improve the accuracy
- Progressively train the network with larger number of Logos
- Deploy it as a smart phone application for the public to enjoy it free

6. Acknowledgements

I wish to thank Mr Steven Chen for all the guidance and support without which this project could not have been taken to successful completion.

7. References

[1] Smart phone application “CamFind”.

[2] Lior Wolf, Tal Hassner, Yaniv Taigman (2009) The One-Shot Similarity Kernel. IEEE 12th International Conference on Computer Vision.

[3] Taigman Y.,et.al. (2014) DeepFace: Closing the Gap to Human-Level Performance in Face Verification. IEEE Conference on Computer Vision and Pattern Recognition.

[4] Hang S ; Shaogang G; Xiatian Z et.al. (2017) WebLogo-2M: Scalable Logo Detection by Deep Learning from the Web IEEE International Conference on Computer Vision Workshops (ICCVW)