INFERENCE PROJECT

ISRAILOV SARDOR

Abstract—In order to sort the big and small bins and to know the time for the robot to come, at the postal sorting machines, various techniques are used. The most common is to use the “ultrasonic sensor” with the mechanical system that detects big to small bins. However, this system doesn’t allow to differentiate between the small bin and the absence of the bin. With the neural network model trained, one can easily differentiate among three of them and add different kind of bins with different sizes depending on the postal client. In this work, I would train the Google-Net to differentiate among 3 states: small bin, large bin and the absence of the bin.

Index Terms—Robot, Udacity, CNN , deep learning.

1 Introduction

ONE of the most common problem problems in robotics is perception of the environment. There have been numerous breakthroughs in this field since the image recognition challenge of 2012. Deep neural networks are able to classify images better than humans, nowadays. However, one of the common application of image classification (by classes) is sorting of objects. One could sort any objects on the conveyor given the powerful embedded system, given the powerful network trained on the well-collected data. During many years, postal sorting machines used different sensors to identify different bins(big bins with journals or small bins with letters). In the era of AI, one of the most flexible ways is to sort the bins using image classification. In this paper, one can find three steps to image classification: gathering data, making appropriate classes as well as choosing model, and finally training model. Moreover, testing of the model on the new data is performed, and one of the ways to put it on the JETSON TX2.

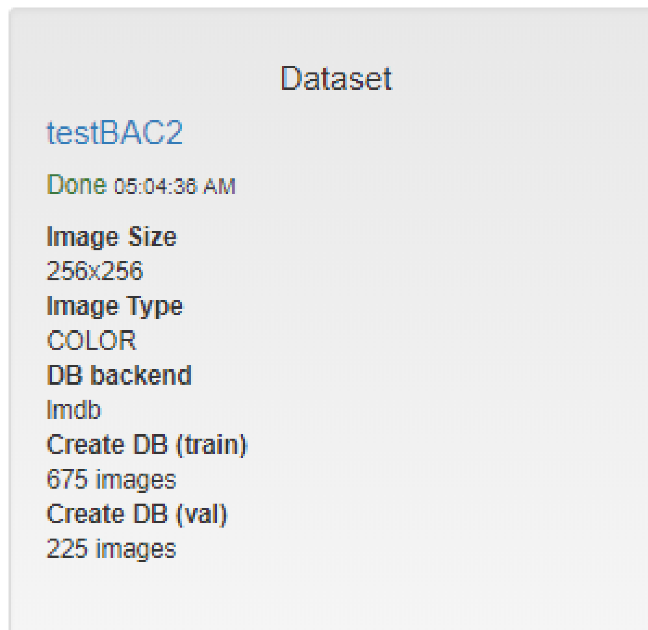


Figure 1. Data-set parameters.

2 Background / Formulation

The Nvidia Digits workspace contains three networks: LeNet, AlexNet and GoogleNet. The chosen networks for this research are AlexNet and GoogleNet, because they take 256*256 RGB images as their input and historically they demonstrated some huge success classifying images. Due to the combination of INCEPTION modules and limited amount of data, GoogleNet was more adapted to correctly classify the images and identify the small/large bins and their absence.

The SGD (stochastic gradient descent) was used as an optimiser with 0.01 as the basic learning rate. Every-time the model was ”overfitted” i.e the validation accuracy increased during model’s ”learning by heart” the training dataset, the learning rate was decreased by 10. This process permitted increased accuracy and small time consumed due to the relatively large learning rate at the beginning. The main results were achieved with GooglNet with the batch size of 30, though 15 was enough. [?]

In the future sections, the following points will be discussed

- the way of data acquisition and its conversion to the dataset
- results for the GoogleNet and AlexNet and their interpretation.
- Discussion and the potention of the project

3 Data Acquisition

The RGB images for all three classes have been collected using Samsung S9 phone with the predefined accuracy (7MP) 1x1. The dataset consists of 900 images of a big bin, 700 images of a small bin and 300 images of the conveyor without a bin. The 25 percent of images were used for the validation. After the model have been trained and validated, it has been tested on some new images. After the images have been collected some prepossessing was done with the help of matlab using functions as ”imread”, ”imwrite” and ”resize” to save the images on the 256x256 pixel scale. The full code can be found on the git link in the annexe.

The initial data-set contained 300 for each class and some dozens of images for the purpose of the test; it was sufficient for the accurate classification of the same bins over a period of time, but the resulting model wasn't accurate enough in recognising new bins with some marks (small paint etc.) Several things need to be taken into account during photos:

- the position of the bin on the conveyor can be different, so several position need to be tried
- different bins(with paper/without, with small paint etc.)
- several position of the camera

Overall, 4 big bins and two small bins participated in the process.

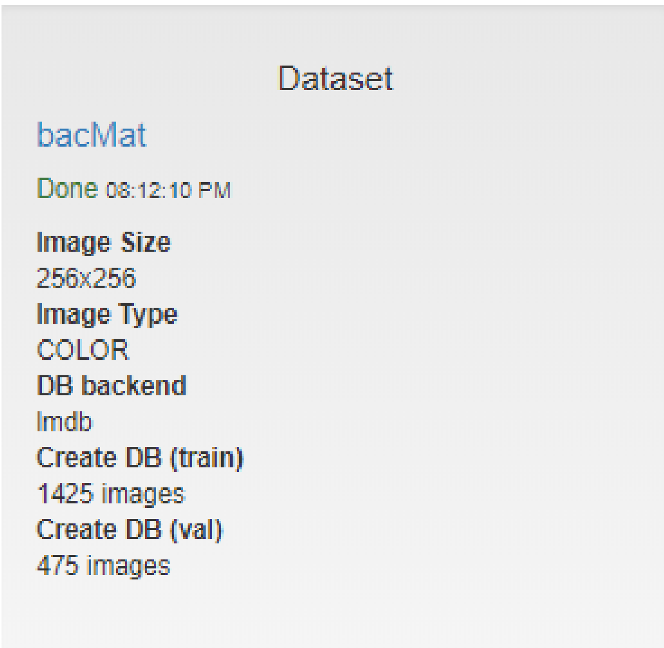


Figure 2. Data-set parameters.

4 Results

4.1 GoogLeNet

Excellent accuracy results have been obtained using GoogLeNet. One can clearly see that significant results have been achieved after the 10th batch training.

4.2 AlexNet

Much worse results have been obtained using AlexNet with the same initial parameters.

5 Discussion

After several tries modifying the data-set, the satisfactory results were obtained. The Google-Net was able to predict with 100 percent accuracy the bins and its absence on the given environment.

Figure 3. Evolution of deep learning.

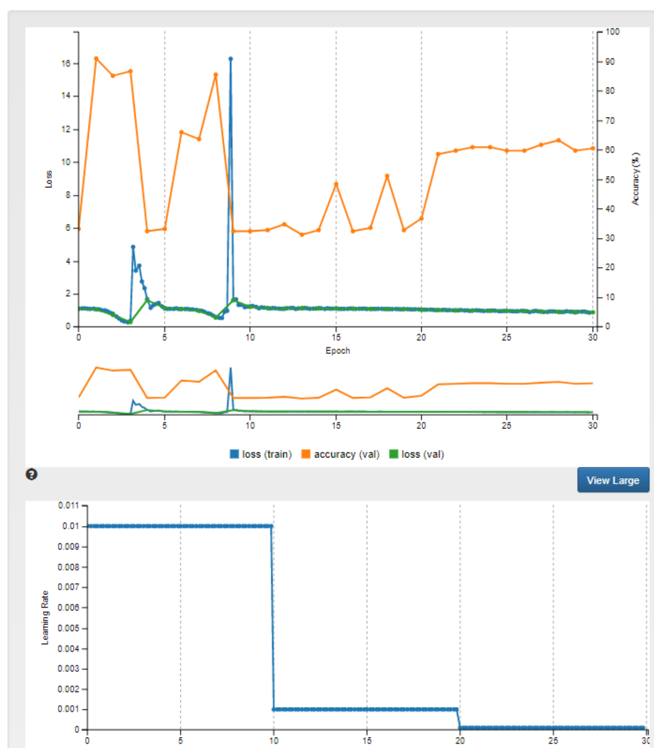


Figure 4. Evolution of deep learning using AlexNet.

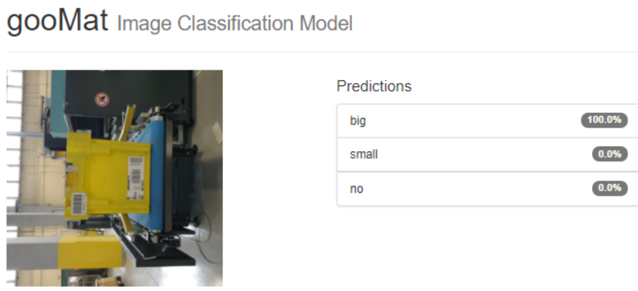


Figure 5. small bin

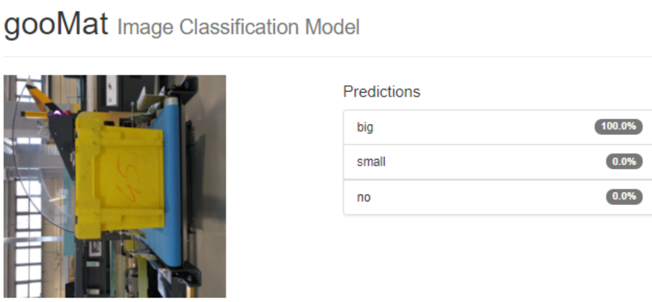


Figure 6. different big bin

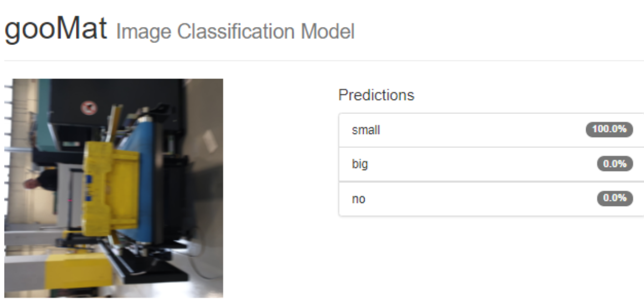


Figure 7. small bin

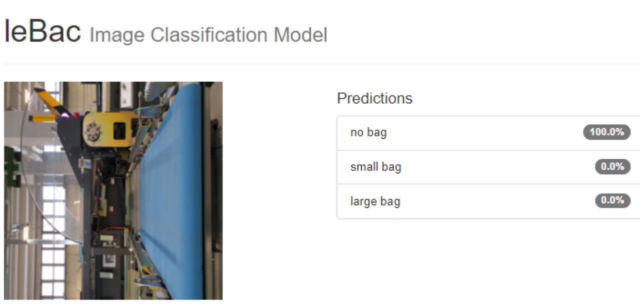


Figure 8. no bin on the conveyor.

5.1 Interpreting results of GoogleNet

Google-Net is a very deep neural network model compared to AlexNet. Due to its inception modules and convolutional (1x1, 3x3, 5x5) layers as well as 3x3 max-pooling, the network has been able to generalise the image, and differentiate between small and large bins as well as their absence.

5.2 Interpreting results of AlexNet

One clearly see that the validation accuracy doesn't converge, and after the model, certain tests have been conducted that revealed that the network differentiates between the absence and presence of the bin, but can't differentiate the bins depending on their sizes.

This is closely related on the network architecture. Initially, AlexNet have been developed to differentiate between many classes that don't have the close similarity among them like there is between large and small bins.

6 Conclusion / Future work

The accuracy of the network for the supplied data meet the requirement. However, the accuracy of the classification may vary with the bins (many scratches and some stuff attached to it) etc. The inference time need to be taken into the account and if necessary, one may want to change the model to have a good compromise between the inference time and the classification accuracy.

The appropriate tests need to be taken before implementing the model with the predefined hyperparamters on the embedded system within the robot. One of the many tests would be to see if the robot would be able to classify "fast" enough for the given application. Another of the many tests may be to attach the paper on small/big bins and to classify it with the provided model.

The provided github code permits to deploy the model on the Jetson TX2.

Once the embedded system is installed on the robot, it can now if the bin is small or big. Moreover, the robot can know whether or not it need to approach the bin to take it to another place.

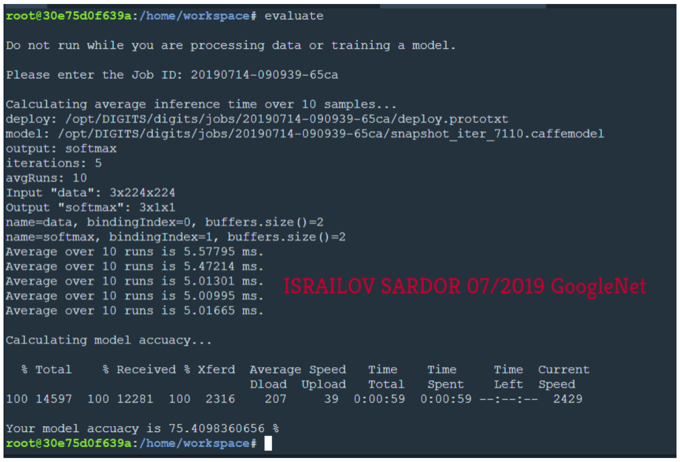
The simple training of GoogleNet on the dataset (p1_data) revealed that it can achieve the accuracy more than 75 percent and the inference time less than 5.6 seconds. In the chosen application, the inference time is not that crucial as the accuracy. Inference time will effect the speed of bin sorting theoretically, though inference time is greater than the mechanical constant time of the robot (how much time it takes to make the robot move), which will determine mainly the speed of sorting.

In conclusion, the given paper may help to sort specific bins in the postal sorting sectors, however the model can be trained on more variety of bins and be employed in the industry.

7 Annexe

For further references go to the next url: https://github.com/ss555/Inference_bin

Some more images of the unsuccessful try that developed model that was able to differentiate correctly in some



```
root@30e75d0f639a:/home/workspace# evaluate

Do not run while you are processing data or training a model.

Please enter the Job ID: 20190714-090939-65ca

Calculating average inference time over 10 samples...
deploy: /opt/DIGITS/digits/jobs/20190714-090939-65ca/deploy.prototxt
model: /opt/DIGITS/digits/jobs/20190714-090939-65ca/snapshot_iter_7110.caffemodel
output: softmax
iterations: 5
avgRuns: 10
Input "data": 3x224x224
Output "softmax": 3x1x1
name=data, bindingIndex=0, buffers.size()=2
name=softmax, bindingIndex=1, buffers.size()=2
Average over 10 runs is 5.57795 ms.
Average over 10 runs is 5.47214 ms.
Average over 10 runs is 5.01301 ms.
Average over 10 runs is 5.00995 ms.
Average over 10 runs is 5.01665 ms.

Calculating model accuacy...

  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 14597  100 12281  100  2316    207     39  0:00:59  0:00:59 --:--:--  2429

Your model accuacy is 75.4098360656 %
root@30e75d0f639a:/home/workspace#
```


Figure 9. inference time and accuracy of GoogleNet.

Figure 10. dataset - p1data.

limited cases(wrong data collection): the model trained using the batch of 30. There are mismatches between the model evolution during different trials due to the randomness of the picked images out of the dataset.

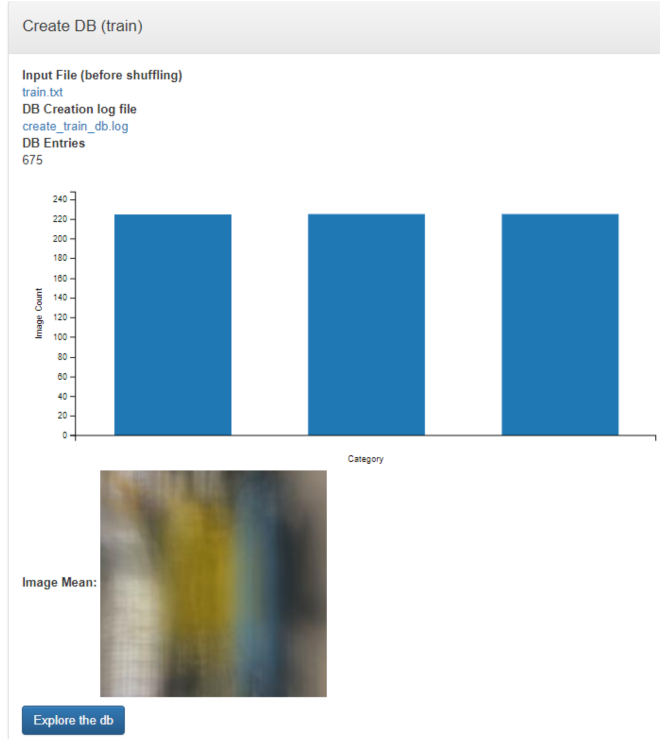


Figure 11. Dataset parameters.

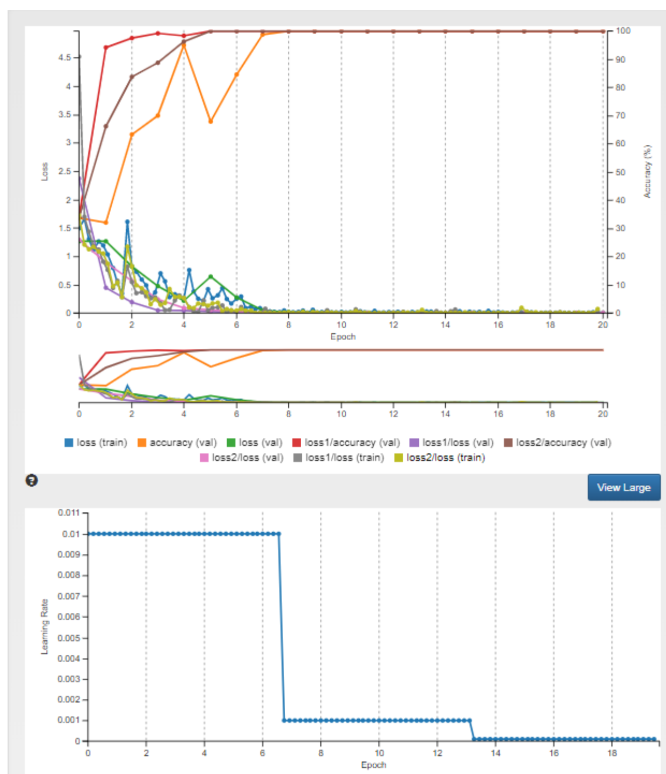


Figure 12. Evolution of deep learning.

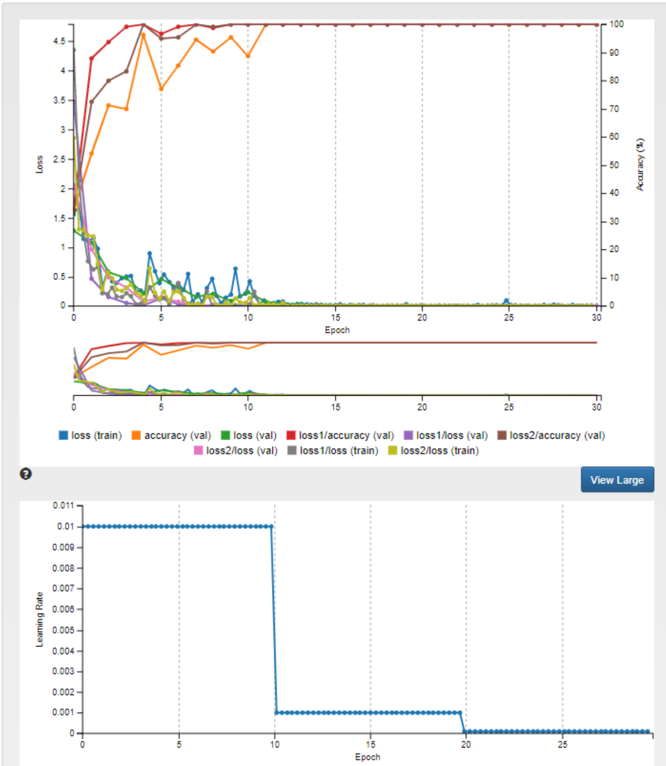


Figure 13. Evolution of deep learning.

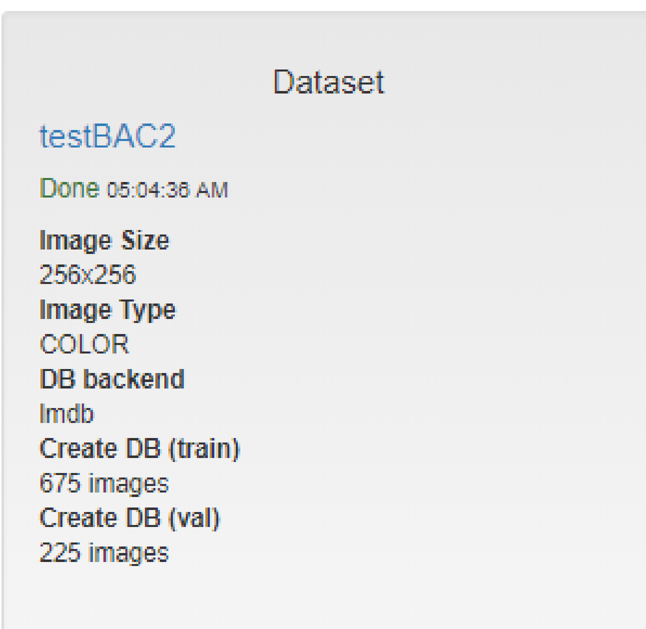


Figure 14. Initial Dataset parameters.