Robotic Inference Project Report

Ahmed Khalid

Abstract—Standard state-of-the-art architectures are sued in this paper to solve 2 classification problems. Hyperparameter choices were laid out, then the method used for data acquisition is explained. Moreover, AlexNet and GoogleLeNet are tested to solve a real-time classification problem and a hand posture classification problem. Finally, results of the 2 problems are discussed to reach the conclusion and lay out the future work needed to improve the classification networks.

Index Terms—Robot, IEEEtran, Udacity, LaTeX, deep learning.

1 Introduction

THE motive for writing this paper is to build a network that can classify the 3 hand postures from the game paper-rock-scissors so that it could be deployed on an embedded system and used in a commercial game. The general procedure and data collecting methods used in this paper would also be a building block in making a network that could interpret sign language. An attempt on this problem was done before and explained in a paper authored by Rambabu Gupta [1]. Gupta experimented with different custom architectures to find a simple one that was suitable for the task while the method used in this paper depends on state-of-the-art standard architectures because of time limitations. Also, a simple real-time classification problem was addressed in this paper as part of the robotic inference project.

2 Background / Formulation

Due to the short amount of time on our hands the experiments on the 2 problems were conducted using well-known standard architectures, specifically AlexNet and GoogleLeNet. This enabled us to reach good results in a short amount of time.

2.1 Real-time Object Classification

Initially AlexNet was the preferred choice but then GoogleLeNet emerged as a better choice. Because AlexNet often stuck in local minima of the error function that did not satisfy the accuracy and inference time requirements of the project. Stochastic Gradient Descent (SGD) solver was used with Base Learning rate set to .01 as a standard initial value with step decay to make changes to the network weights more subtle as the training goes on.

2.2 Hand Posture Classification

AlexNet was chosen for this problem due to the fact that it is relatively easy (3 classes only) so a less deep network is less prone to overfitting, much faster to train and also the inference time is lower. Stochastic Gradient Descent (SGD) solver was used with Base Learning rate set to .01 as a standard initial value and exponential learning rate decay to make changes to the weights more subtle as the training goes on.

3 Data Acquisition

The dataset used for the project consists of around 14100 images, divided into 3 classes with each one having around 4700 images. The images were converted to grayscale and resized to 256x256. The data was collected using a laptop's integrated camera by using a simple python script. The following method was used to capture images for each class. 130 images of one posture in different angles and positions in image x 2 different backgrounds x 2 different lighting conditions which results in 520 images for each class. Then data augmentation [2] was used to generate more data and add more variety to the dataset. Each image from the original dataset was used to generate 8 distinct images. Resulting in a total of around 4200 augmented images + 520 original images for each class, For a grand total of nearly 14100 images. The following operations were performed randomly each time to produce a distinct images:

- Horizontal flipping
- Vertical Flipping
- Random Cropping
- Blurring
- Adding noise
- Tweaking normalization
- Tweaking brightness
- Scaling
- Rotation
- Applying shear

For the given dataset, a distinct test set was required for the evaluate command to run and evaluate network results, so the data was divided into 60% training, 20% validation and 20% testing as a recommended standard for small datasets [3].

But as for the hand posture classification problem, the data was divided into 80% training and 20% validation/test set. The validation set could be used as the test set as it was not used to modify the network or fine-tune so it was fine to use it for testing,

Fig. 1. Augmented rock image.

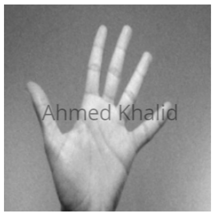


Fig. 2. Augmented paper image.

Fig. 3. Augmented scissors image.

4 RESULTS

We discuss the results of each project independently.

4.1 Real-time Object Classification

The first project required at least 75% accuracy and an inference time below 10 ms. While both AlexNet and GoogLeNet achieved an inference time below the required limit, AlexNet struggled with getting out of local minima which resulted in low accuracy. This is demonstrated in the following figure: We clearly see that the network is stuck

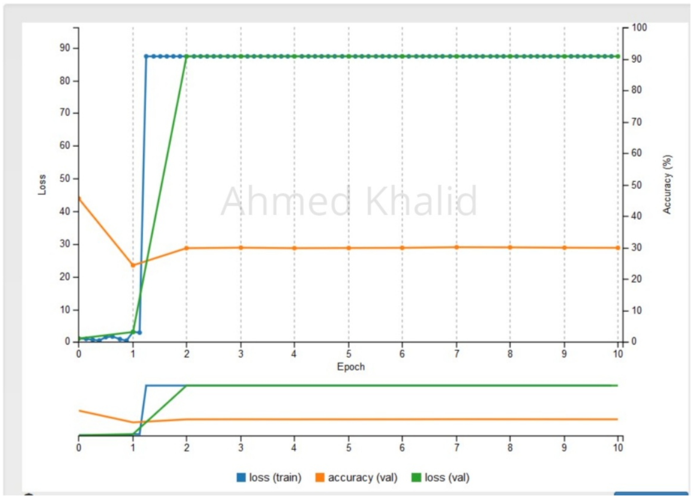


Fig. 4. AlexNet training results

in a local minimum and is unable to continue learning. But with GoogLeNet the network was able to easily achieve a validation accuracy of 100% after 4 epochs only.

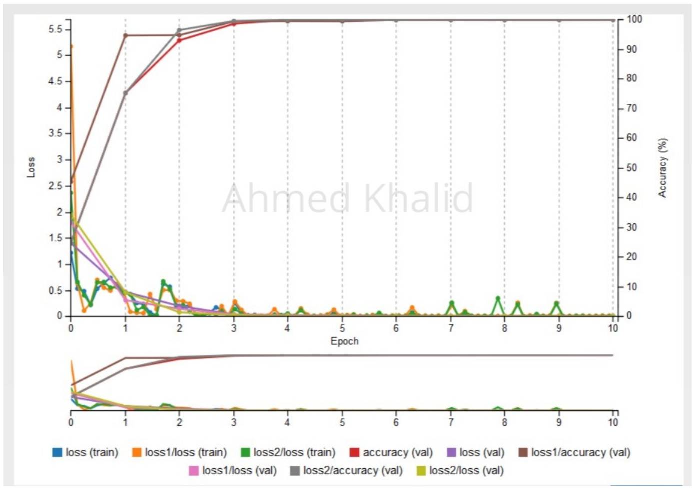
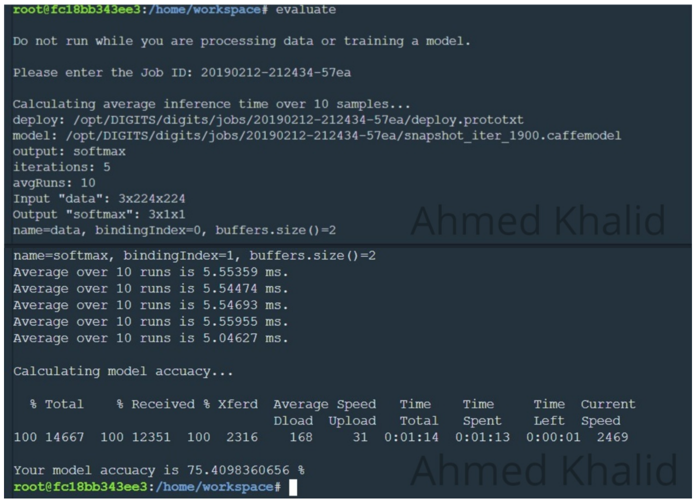


Fig. 5. GoogleLeNet training results

Also, the network delivered satisfactory results when testing it with the **evaluate** command achieving over 75% accuracy and nearly 5.5 ms inference time.



```
root@fc18bb343ee3:/home/workspace# evaluate

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

Please enter the Job ID: 20190212-212434-57ea

Calculating average inference time over 10 samples...
deploy: /opt/DIGITS/digits/jobs/20190212-212434-57ea/deploy.prototxt
model: /opt/DIGITS/digits/jobs/20190212-212434-57ea/snapshot_iter_1900.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 5.55359 ms.
Average over 10 runs is 5.54474 ms.
Average over 10 runs is 5.54693 ms.
Average over 10 runs is 5.55955 ms.
Average over 10 runs is 5.04627 ms.

Calculating model accuacy...

  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 14667  100 12351  100  2316    168     31  0:01:14  0:01:13  0:00:01  2469

Your model accuacy is 75.4098360656 %
root@fc18bb343ee3:/home/workspace#
```

Fig. 6. Evaluate command output

4.2 Hand Posture Classification

While GoogLeNet achieved satisfactory results, AlexNet was faster to train, achieving nearly 100% validation accuracy after 10 epochs. When tested with images outside of the dataset the network was able to correctly classify them as long as the hand was dominant in the picture i.e. taking most of the area of the image. But it had difficulty when the hand started getting away from the camera, especially with rock and paper images. Also shadows and large bright light spots hindered the network ability to correctly classify the images.

This figure shows the training progress of the network:

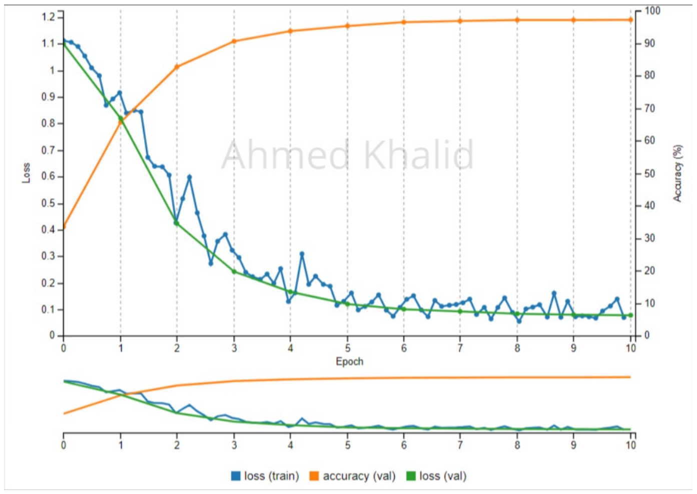
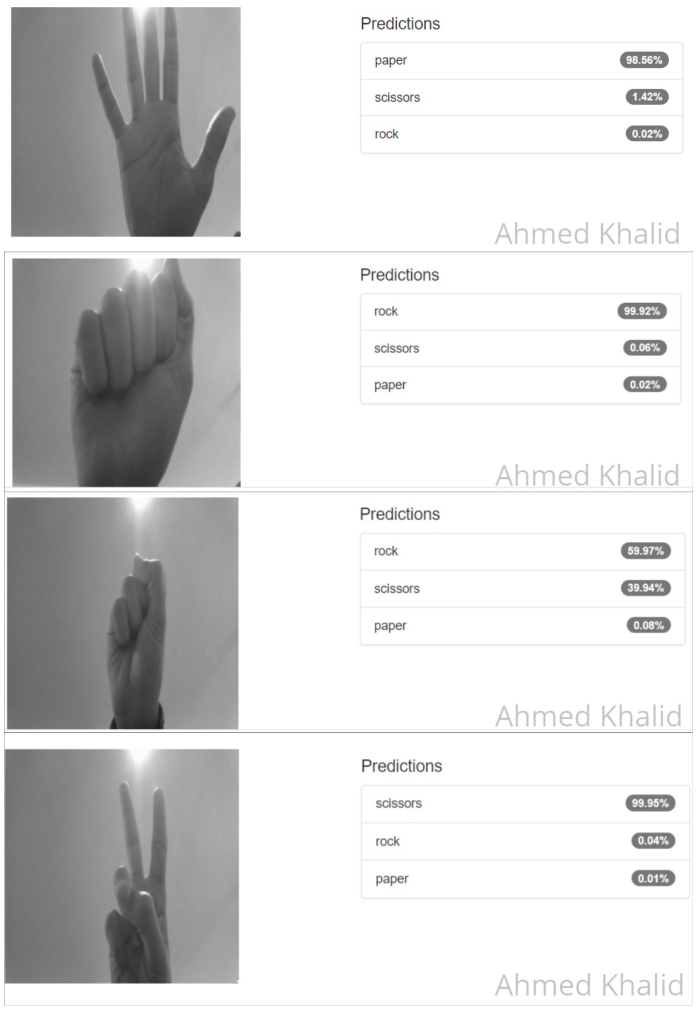


Fig. 7. Evaluate command output


Predictions

paper	98.56%
scissors	1.42%
rock	0.02%

Predictions

rock	99.92%
scissors	0.06%
paper	0.02%

Predictions

rock	59.97%
scissors	39.94%
paper	0.08%

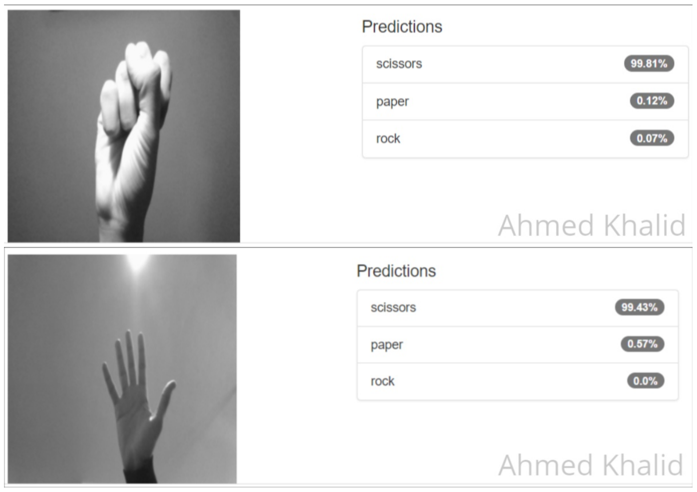
Predictions

scissors	99.95%
rock	0.04%
paper	0.01%

Fig. 8. Correctly classified images

5 Discussion

The results of the network used for the hand posture recognition is good for the given dataset and shows the power of data augmentation which helped to make a lot of training


Predictions

scissors	99.81%
paper	0.12%
rock	0.07%

Predictions

scissors	99.43%
paper	0.57%
rock	0.0%

Fig. 9. Misclassified images

data. But when tested with different real-world images the network started to misclassify images under new conditions like new lighting conditions and when the hand is far from the camera. This shows that the dataset needs more variety when collecting it to be able to adapt to new situations.

6 Conclusion / Future work

The network used for the real-time object classification problem achieved the requirements for the task but for it to be deployed in the real world it needs to achieve better accuracy, possibly by collecting more data and using data augmentation to generate even more. The network used for hand posture recognition needs a dataset that has more variety to be more deployed in the real world. Some of the enhancements to be made to the dataset are:

- Taking images of more people’s hands, not just one
- Taking images at different distances from the camera
- Take images in the presence of more lighting conditions
- Take more images of paper and rock. Because learning them is harder on the network than scissors

References

[1] G. Rambabu, “Applying Deep Learning for Classifying Images of Hand Postures in the Rock-Paper-Scissors Game,” June 2017.
[2] B. Raj, “Data Augmentation | How to use Deep Learning when you have Limited Data Part 2,” Apr. 2018.
[3] N. Andrew, “Train / Dev / Test sets - Practical aspects of Deep Learning.”