Udacity RoboND Project: Robotic Inference
Image Classification Using GoogLeNet

Trijeet Kr. Modak

Abstract—This paper explores a deep neural network architecture, GoogLeNet, to classify objects on a conveyor belt or lying on any flat surface. Training, validation and testing of the model is performed over Nvidia DIGITS platform which simplifies the data and model management process. Data acquisition, training and results are also discussed in this paper. The final accuracy of the models and the inference time met the project requirements.

Index Terms—Robot, GoogLeNet, Udacity, neural network, deep learning, classification.

1 Introduction

It is important for robots to understand its surroundings correctly to carry out the operation(s) it has been assigned with. One of the simplest yet elegant way to achieve this is through vision, i.e. using cameras attached to a robot and processing the video feeds from them. Such processing includes techniques such as image classification, object detection in images, motion detection, etc. This paper focuses on image classification using which the a robot can identify and classify known objects in its surrounding.

Pattern recognition and machine learning [1] algorithms such as Deep Neural Networks [2] can simplify the task of object classification by learning the features of the objects. A neural network can be trained with existing labeled images of various objects and later can be used to identify the objects from a previously unseen image. Hence, if a robot is equipped with such a trained model, it can quickly infer the objects in its surrounding from its camera feeds. GoogLeNet [3] is one such architecture that won the ImageNet [4] 2014 challenge. Hence GoogLeNet is used for classification of the images in this project as it is faster and has a high accuracy rate.

2 Background / Formulation

DIGITS [5] is a deep learning platform by NVIDIA that simplifies the entire model training process. It provides a few pre-trained models such as LeNet, AlexLet and GoogLeNet. It also allows custom models to be imported and used. It also handles creation and management of datasets for the model training. As a result, DIGITS was chosen to train the models.

There are two parts to the project where two models are to be trained, each to classify and detect items as listed below:

- Classes for Object Detection on Conveyor Belt (Model 1 - Udacity Dataset):
  1) Bottle,
  2) Candy Box, and
  3) Nothing
- Classes for Object Detection on self collected data (Model 2 - Collected Dataset):
  1) Battery,
  2) Breadboard,
  3) DC Motor, and
  4) Nothing

3 Data Acquisition

3.1 Model 1 (Udacity Dataset)

Dataset for object detection on a conveyor belt is provided by Udacity. The following table describes the details of the dataset. The train-eval-test ratio used is 14:5:1

Images	*Bottle Images*	*Candy Box Images*	*Nothing Images*	***Total***
Training	3198	1747	2122	7067
Validation	1142	624	758	2524
Testing	228	124	151	503
Total	4568	2495	3031	10094

TABLE 1
Dataset provided by Udacity

The following table shows a few sample images from the dataset:

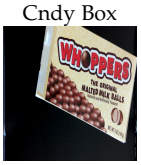
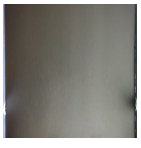
Bottle	Cndy Box	Nothing
		
		

TABLE 2
Samples from the udacity dataset

3.1.1 Model 2 (Collected Dataset)

Images for battery, breadboard, dc motor and 'nothing' is collected placing them flat on a dark surface using Intel Realsense D435 cameras at a resolution of 640x480. The details are presented below:

Images	*Battery*	*Bread Board*	*DC Motor*	*Nothing*	***Total***
Training	71	71	71	71	7067
Validation	26	25	25	26	2524
Testing	5	5	5	5	503
Total	102	101	101	102	406

TABLE 3
Collected Dataset

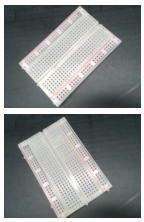
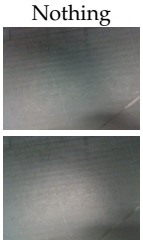
Battery	Bread Board	DC Motor	Nothing
			

TABLE 4
Samples from the collected dataset

4 Results

4.1 Model 1 (Udacity Dataset)

The model is trained for 30 epochs on the Nvidia DIGITS platform running on Tesla K80 GPU in the workspace provided by Udacity. It took slightly less than an hour to complete the training process. The training loss and accuracy curves are presented below:

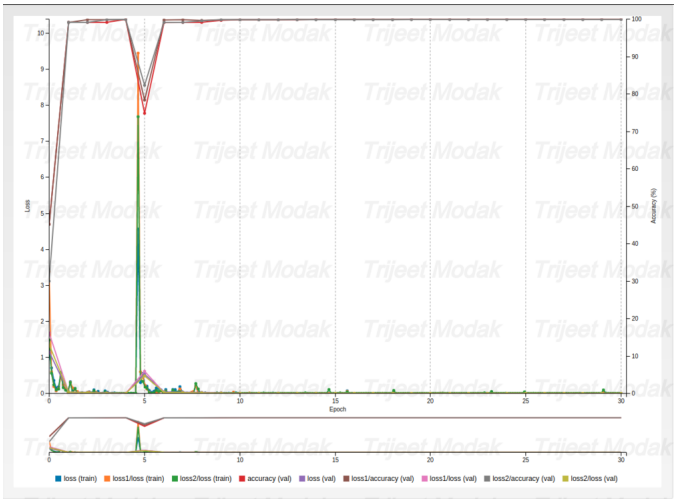


Fig. 1. Model 1 loss/accuracy vs epoch curve

The result obtained on running the *evaluate* command on the first model resulted in the output presented in the following screenshot. The model reached an accuracy of 75.40% with an inference time of roughly 5ms which is in acceptable range of the project requirements.

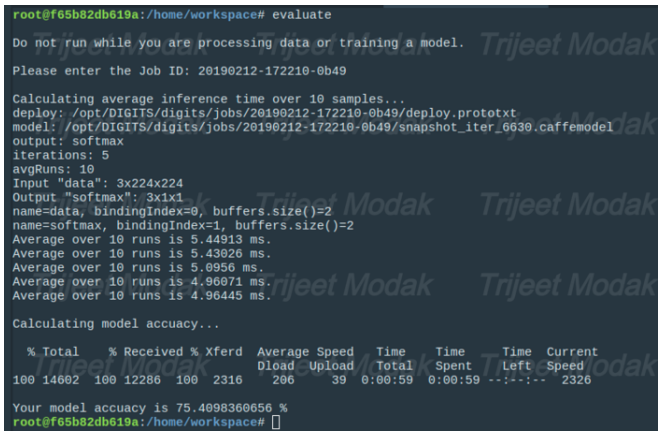


Fig. 2. Model 1 loss/accuracy vs epoch curve

4.2 Model 2 (Collected Dataset)

The second model is also trained over Nvidia DIGITS for 100 epochs which took about 10 minutes. At the end of the training it attained a validation accuracy and validation loss of 99.10% and 0.021 respectively. The training and validation curves are presented in the following figure:

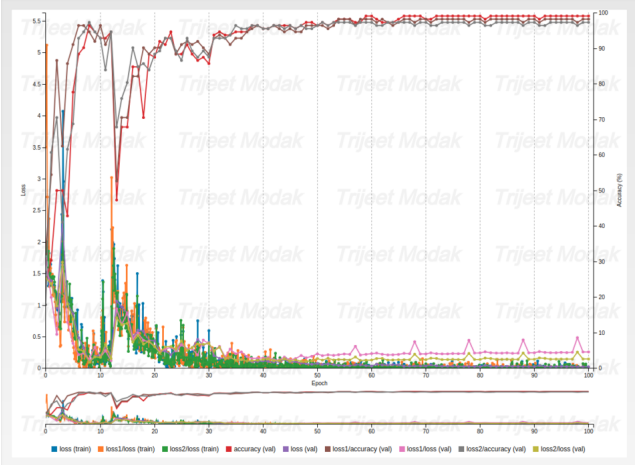


Fig. 3. Model 2 loss/accuracy vs epoch curve

Once the model got trained, it is tested against a few images corresponding to the trained classes but which were not used during the training or validation process. The inference results of this model are presented in Fig. 4.

5 Discussion

It is quite astonishing that GoogLeNet was able to train itself using only about 100 images per class for the second model with over 95% accuracy. One reason for this could be because there were only 4 classes to predict from and there were no subclasses in the dataset.

Although a high accuracy is always desirable, it is important that it doesn't take too long to make an inference especially in a realtime scenario. GoogLeNet seems to be fast enough to be used in real-time object recognition operations for robotic inference projects.

Fig. 4. Model 2 loss/accuracy vs epoch curve

6 Conclusion / Future work

GoogLeNet was used to model two different scenarios of object classification/recognition, one with a set of supplied data from a conveyor belt and another set that was manually collected while doing the project. In both cases GoogLeNet was able to predict the correct class of the object almost always with a very high confidence value. The results were promising and met the project requirements.

References

[1] N. M. Nasrabadi, "Pattern recognition and machine learning," *Journal of electronic imaging*, vol. 16, no. 4, p. 049901, 2007.
[2] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.
[3] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.
[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 248–255, Ieee, 2009.
[5] L. Yeager, J. Bernauer, A. Gray, and M. Houston, "Digits: the deep learning gpu training system," in *ICML 2015 AutoML Workshop*, 2015.