Robotic Inference

Vladislav Artamonov

Abstract—Perception using Neural Networks is widely used in modern robotic applications. This report presents the results obtained from using Deep Neural Networks to perform image classification tasks. The results are presented considering the constraints imposed by embedded systems such as limited energy supply and maximum inference time. The conclusion is that a simpler model is often the best choice if it can achieve the desired accuracy. The outcome of this investigation will be used for a robot that can autonomously locate Lego bricks in the home environment with further applications such as sorting.

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

1 Introduction

Autonomous robots can be very useful for a broad range of applications including find and fetch objects. This use can be very beneficial especially in super boring tasks such as organizing Lego bricks scattered all over the floor. For instance, a robot could autonomously locate and bring each brick as per the rules defined by children or adults.

Approaches to accomplish this mission could be using RFID technology [1] but it requires that each brick is previously identified with a some sort of physical tag. A more flexible option is to use modern Machine Learning techniques combining camera sensors and Deep Neural Networks to perform object recognition.

The drawback is that such applications demand large hardware resources. If latency and connectivity is not a concern, image recognition can be implemented in the cloud. Unfortunately, it is usually not possible as many mission-critical applications demand a very fast response time and have unreliable connectivity. It is also common to run on batteries and require efficient use of the available power.

Recent studies show that energy constraint is an upper bound on maximum achievable accuracy and model complexity [2]. In other words, to preserve battery on a robotic application, one should find the model with lower complexity that can achieve the desired accuracy given a maximum inference time.

With the intention of exploring the performance of machine learning in robotic applications with limited hardware resources, this report presents a comparison of two prototype inference models using two widely known architectures: AlexNet [3] and GoogLeNet [4]. The desire is to establish the foundation that will be used later in a project to autonomously find and organize Lego's inside the house.

2 Background / Formulation

This project is part of the Robotics Software Engineer Nanodegree from Udacity. Two datasets were used to train two types of deep neural network models used for inferencing. The first dataset was provided in the course material and was used for learning purposes. The second dataset was collected from random Lego bricks found on the floor at a house.

	Supplied Dataset		**Collected Dataset**	
	AlexNet	GoogLeNet	AlexNet	GoogLeNet
Number of epochs	6	6	20	20
Initial learning rate	0.01	0.01	0.01	0.01
Decrease Fator	0.1	0.1	0.1	0.1
Number of Steps	3	3	10	7

TABLE 1
Hyperparameters

The models were trained in a virtual environment using the NVIDIA Deep Learning GPU Training System (DIGITS). The same hyperparameters were used for easy comparison. Their values are shown in the Table 1.

Finally, the inference time of each model trained with the supplied dataset was measured using a script provided. Considering the fairly simple datasets, it was expected that AlexNet, a less complex network would perform inference faster when compared with GoggLeNet.

3 Data Acquisition

3.1 Supplied Dataset

The first dataset is supplied by Udacity as part of the learning materials. It contains 10094 images. 75% is used for training and 25% for validation. The test images were not directly provided to the students. The pictures were taken from objects passing on a conveyor belt and separated in three different labels: ”Bottle”, ”Candy Box”, and ”Nothing”. The PNG images have a resolution of 256x256 pixels and are provided inside the Udacity workspace. See examples of images in Figure 1.

3.2 Collected Dataset

The second dataset was collected using the built-in camera on a MacBook. To facilitate the capture process, a python script using the OpenCV package was used. The PNG images were stored using a 1280 x 720 pixel resolution following the folder structure used in DIGITS as the labels. The images were separated into four different categories ”empty”, ”green brick”, ”orange brick” and ”yellow brick”

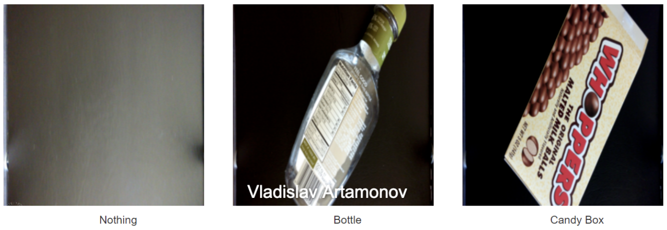


Fig. 1. Supplied dataset

with number of images per class is almost equal except for the images without any bricks. The ”empty” class contained 25% less of examples. Before training, all images were resized to 256x256 using ”Squash” transformation so that pre-trained Neural Networks could be used.

A total of 756 images was collected and separated into three groups: 80% for training, 10% for validation, and 10% for testing. For simplicity, the objects were placed on the floor with a neutral background. Sample images can be seen on the Figure 2.

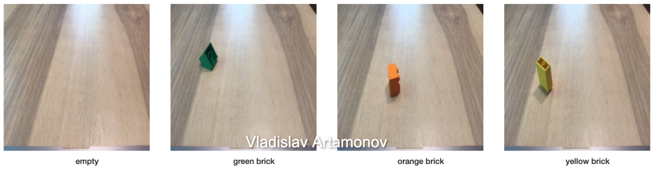


Fig. 2. Collected Dataset Sample

4 Results

4.1 Supplied Dataset

Figure 3 shows the learning curves obtained for the supplied dataset. It is possible to see that both models achieved scores close to 100% for the validation set.

Using the ’evaluation’ command, provided by Udacity, it was observed that both models achieved the same accuracy of 75.41% using the test images. Training time for AlexNet was under 4 minutes while the training time for GoogLeNet three times longer. Inference time for the AlexNet was always bellow 5ms, while inference time for GoogLeNet was slightly above 5ms most of the time.

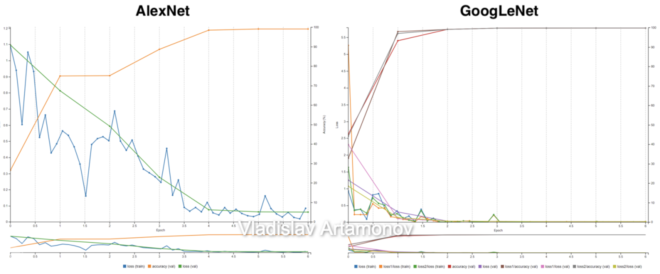


Fig. 3. Learning curves on the Supplied dataset

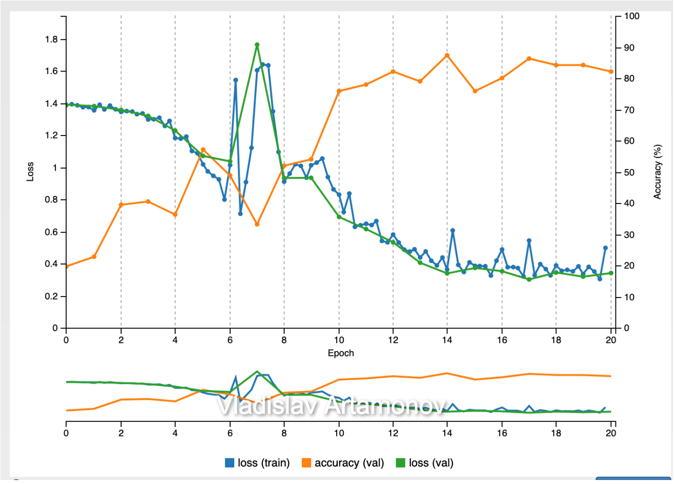
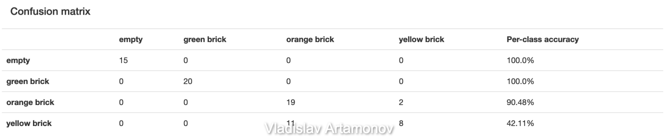


Fig. 4. AlexNet learning curves on the collected dataset

4.2 Collected Dataset

AlexNet was able to reach about 82% of accuracy on the validation set after 20 epochs (Figure 4).

It is possible to conclude from the confusion matrix for the validation dataset of 75 images that the network had problems classifying yellow bricks (Figure 5).



Confusion matrix

	empty	green brick	orange brick	yellow brick	Per-class accuracy
empty	15	0	0	0	100.0%
green brick	0	20	0	0	100.0%
orange brick	0	0	19	2	90.48%
yellow brick	0	0	11	8	42.11%

Fig. 5. Confusion matrix for AlexNet run on the test part of the collected dataset

On the other hand, GoogLeNet was able to archive 100% accuracy on the validation dataset with the same number of epochs (Figure 6). It was able to properly classify all images from the test dataset except one (99% of accuracy) where it confused orange brick with yellow one. However, GoogLeNet was approximetely 20% slower than AlexNet.

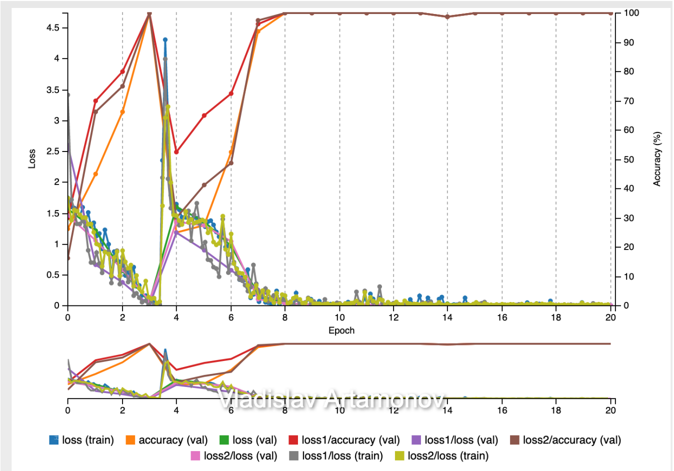


Fig. 6. GoogLeNet learning curves on the collected dataset

5 Discussion

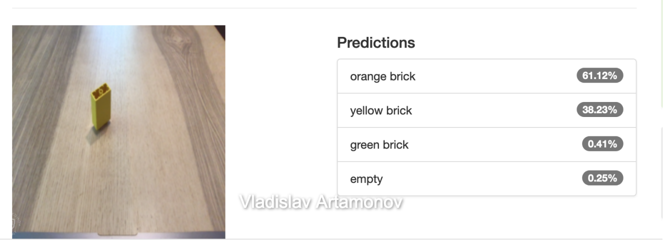
5.1 Supplied Dataset

The provided dataset was very helpful to quickly start learning how to use the DIGITS workspace to train Deep Neural Networks for image recognition. The results obtained using GoogLeNet and AlexNet were very similar. It is worth noting that the accuracy achieved in the test set (75%) is significantly lower than the accuracy achieved during training for the validation set (higher than 90%). It indicates that the trained model is failing to generalize or the model is overfitted.

Considering the lower training time and the slightly faster inference time, one could choose to use AlexNet. Nevertheless, in the provided application, battery consumption is not a concern. Therefore, GoogLeNet can be selected and after additional training with a more complete dataset, better results could potentially be achieved.

5.2 Collected Dataset

For the collected dataset, while the both networks performed very well on the validation datasets, AlextNet failed to properly classify yellow bricks from the validation data. When inspecting the examples where the network failed, it looks like it had mostly problems with shadow covering the most of the brick, like in the Figure 7.



Predictions

orange brick	61.12%
yellow brick	38.23%
green brick	0.41%
empty	0.25%

Fig. 7. Convolution Layer Activation

What could probably help with dealing with situations like that is making pictures from different angles or using data augmentation for training images.

When training the GoogLeNet network, high accuracy on the validation set was achieved very early. Selecting the model achieved on the epoch number 3 or 7 would significantly decrease the learning time while also preventing the network from overfitting.

Given that all images provided in this dataset were captured with a very similar background and from the same angle, it is plausible to assume that the accuracy is expected to decrease significantly when the objects are placed in different environments.

For the collected dataset the inference time was not exactly measured since the evaluation was performed in the DIGITS workspace. Instead, approximate times provided by the GUI were used to compare the running time.

6 Conclusion

Despite achieving a high accuracy for the test dataset, GoogLeNet may not perform well for the target application of locating bricks anywhere in the house autonomously. It is far from a commercially viable product. Nevertheless, it achieved the main learning goals and has laid the foundation for future work. The project was great to understand the core challenges of training Deep Neural Networks, especially the notion that accuracy is not the only important parameter to be considered. Inference time and model complexity are equally important, especially for robotic applications. It was also important to get a sense of how much the quality of the dataset has a strong influence on the results.

7 Future work

The first activity to be done in the future is to deploy both models in hardware with lower performance and compare the inference performance. Later, the models can be further improved by collecting more data and enhancing the training dataset with several augmentation techniques. Finally, the goal is to include not only the classification step but also improve the perception pipeline to execute a complete segmentation task.

References

[1] T. Deyle, M. S. Reynolds, and C. C. Kemp, "Finding and navigating to household objects with UHF RFID tags by optimizing RF signal strength," *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2579–2586, September 2014.

[2] A. Canziani, A. Paszke, and E. Culurciello, "An analysis of deep neural network models for practical applications," *CoRR*, vol. abs/1605.07678, 2016.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, pp. 84–90, May 2017.

[4] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *CoRR*, vol. abs/1409.4842, 2014.