COLLEGE OF ENGINEERING

UNIVERSITY OF NOTRE DAME

2021 CSE 60625 Final Project Report

Prepared by

Subhadyuti Sahoo

University of Notre Dame,

Notre Dame, IN, 46556

1 OBJECTIVES AND EXPECTED SIGNIFICANCE

The objective of my project is to evaluate as to how far a GPU like NVIDIA GeForce RTX 3070 can be trusted to run deep learning algorithms on it. Image classification has gained significance over the years as this forms the basis for object detection systems in autonomous vehicles, drones and underwater robots. More often, the images need to be trained right from scratch as well-known datasets like ImageNet, Pascal VOC and COCO – to name a few – may not contain all the categories of images of objects which are either found in nature or manufactured. Training such images on readily-available computer GPUs is thus prudent as direct deployment of such image classification models on systems like NVIDIA Jetson boards become easier.

Another objective is to verify if ImageAI can be trusted to deliver its own promise of accurate image classification algorithms, using fewer lines of codes compared to the ones provided by Keras. If such a thing is indeed possible, then machine learning scientists and enthusiasts will not have to rely only on Keras which demand significant computational resources for running deep learning algorithms.

These objectives will help evaluate as to how far NVIDIA has been successful in fulfilling its promise of bringing cuDNN powered and GPU accelerated deep learning to the machine learning scientists, especially to those who cannot afford to spend a fortune behind setting up multiple GPUs with the limited budget they generally have.

2 RELATED BACKGROUND

The dataset used for this project is the one provided to the students for detecting objects in drone-captured images. The dataset can be found here: https://byu.app.box.com/s/hdgztcu12j7fij397jmd68h4og6ln1jw

Codes have been used from ImageAI website for training the images, recording the training and validation accuracy scores and evaluating the prediction probabilities of the images from each category. In order to segregate the given dataset into training and validation datasets and put them into different folders for making them work with ImageAI, inspiration was taken from a question answered in Stack Overflow. Furthermore, codes have also been used from Keras and from Chapter 5 of Deep Learning with Python as I made an attempt to re-train the given images on pre-trained ResNet50 architecture in order to check how the training and validation accuracy scores compare with the ones trained right from scratch.

3 APPROACH

The steps followed and the approaches taken are described underneath in a chronological order:

1. The images, downloaded from the aforementioned Dropbox link, were categorized into 12 main categories:
    a. Boat
    b. Building
    c. Cars
    d. Drones
    e. Group
    f. Horseride
    g. Paraglider
    h. Person
    i. Riding
    j. Trucks
    k. Wakeboard
    l. Whales
2. After categorization of the images, one image from each category were randomly selected in order to evaluate the prediction probability of that image belonging to one of the 12 categories.
3. The rest of the images from each category were randomly segregated into training (70%) and validation (30%) datasets.
4. The training dataset was trained on image classification architectures like DenseNet121, InceptionV3, MobileNetV2 and ResNet50 from ImageAI one after another (as parallel training of images was out of bounds for NVIDIA GeForce RTX 3070). Each one of the architectures took nearly 3 hours to be trained with the training images.
5. Training time for each of them was recorded, accuracy and loss scores of training and validation datasets were recorded and plotted and the best models from each one of them, after training the training dataset for 20 epochs, was saved for further evaluation purposes.
6. The images were re-trained on pre-trained ResNet50 architecture from Keras (transfer learning). However, such re-training did not improve the training and validation accuracy scores over 20 epochs. Furthermore, re-training the training images took nearly 12 hours, way more than training those images from scratch using ImageAI.
7. Attempt was also made to re-train the training dataset on the other pre-trained architectures like DenseNet121, InceptionV3 and MobileNetV2, but after calculating the number of hours each one of them would take to complete training and after observing that the training and validation accuracy scores were not improving even after 3 epochs for any of them, the plan of transfer learning with these three architectures had to be discarded.

8. The images, selected for evaluating prediction probabilities in Step **2**, were tested on the best models from each of the four image classification architectures from ImageAI – DenseNet121, InceptionV3, MobileNetV2, ResNet50 – and their prediction probabilities were recorded and plotted.

4 PROJECT OUTCOME

The accuracy and the loss scores for both training and validation datasets trained with image classification architectures from ImageAI were almost ideal: the accuracy scores kept on increasing with each epoch till they reached a saturation point and the loss scores kept on decreasing till they reached a saturation point. **Figures 1** through **4** show the accuracy and loss scores for both training and validation datasets for all the four image classification architectures used from ImageAI – DenseNet121, InceptionV3, MobileNetV2 and ResNet50. The objective related to the potency of ImageAI to train images with just a few lines of codes got tested here. ImageAI has pulled this one off with flying colors.

The training dataset was then trained on the pre-trained ResNet50 architecture from Keras. As already mentioned above, the training and validation accuracy scores for this one did not improve even after 20 epochs. Furthermore, the training took nearly 12 hours, which was way more than the training times of any of the architectures with ImageAI. **Figure 5** shows the accuracy scores of training and validation accuracy scores of the model re-trained with pre-trained ResNet50 architecture from Keras. The most notable observation from this plot is that the accuracy scores do not change with the number of epochs. **Figure 6** shows the time taken in seconds by each architecture to train the training dataset.

Nonetheless, the best models from each of the four architectures from ImageAI were then used for evaluating the prediction probabilities of the test images. **Figure 7** shows the prediction probabilities of each category of image when tested with the best models from each architecture from ImageAI. It is worth observing that most of the categories did not even get classified even though the training and validation accuracy scores were so high. It is also true that, almost all the images of “truck” category got classified accurately with DenseNet121 and InceptionV3 and all the images of “paraglider” category got classified accurately with ResNet50 architecture. With MobileNetV2 architecture, the categories “cars”, “horseride” and “person” got classified accurately. Apart from these, the results did not meet the expectations.

The sheer imbalance in the number of samples for each of the categories could only be a rational justification for such an abnormality in the classification results. The number of samples provided for categories like “person” and “car” easily outnumber the number of samples provided for the other categories. Hence, it is highly possible that, during training of the images, most of the categories other than “person” and “car” got incorrectly classified as “person” and “car” and that is why, the actual categories did not get detected when even the best models were tested upon randomly selected images from each category.

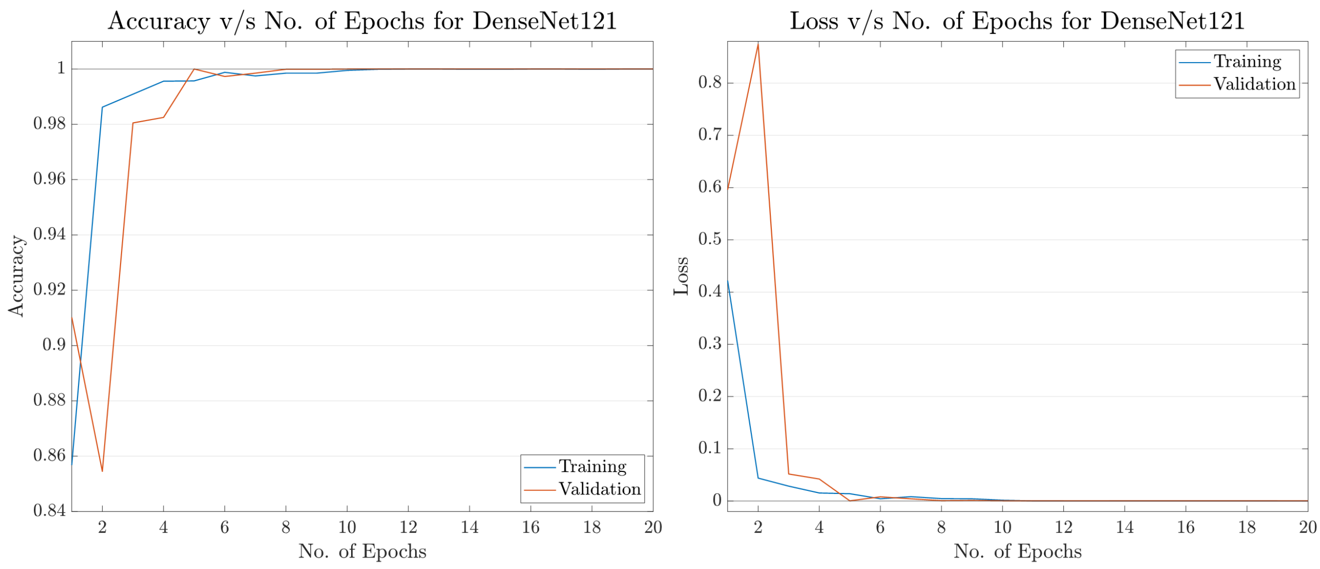

Figure 1 – Accuracy and Loss Scores for Training and Validation Datasets for DenseNet121

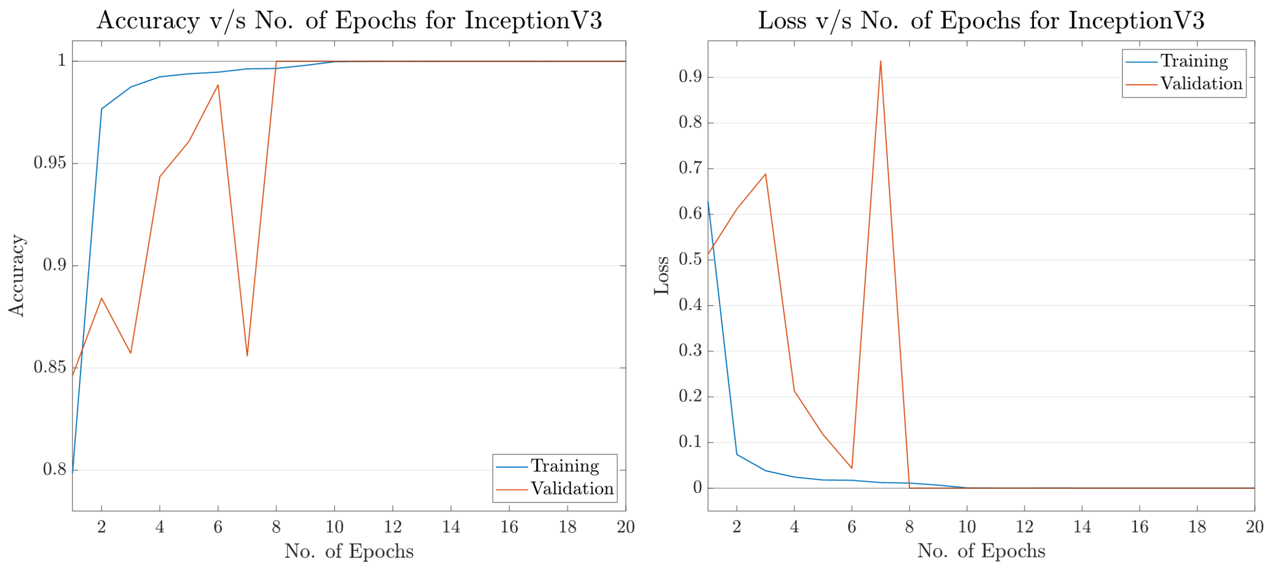

Figure 2 – Accuracy and Loss Scores for Training and Validation Datasets for InceptionV3

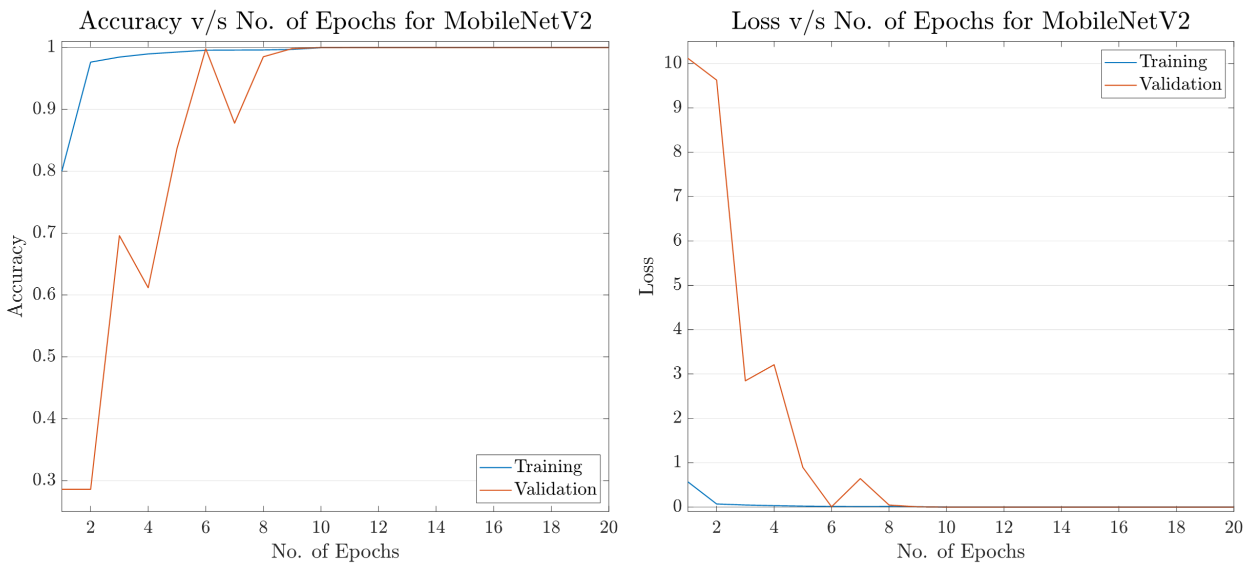

Figure 3 – Accuracy and Loss Scores for Training and Validation Datasets for MobileNetV2

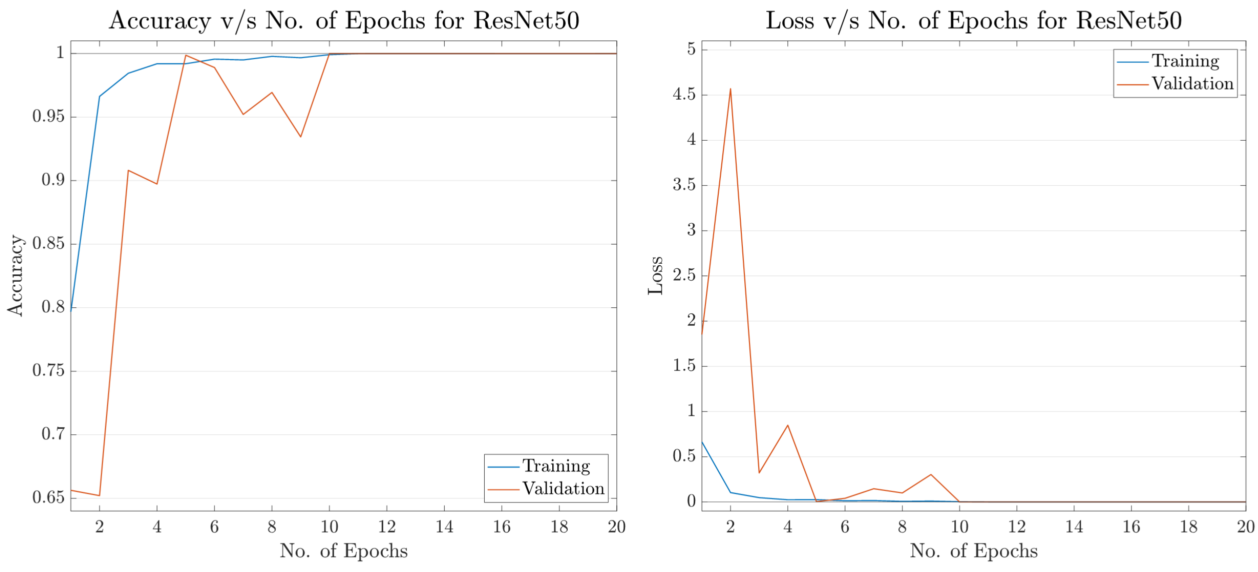

Figure 4 – Accuracy and Loss Scores for Training and Validation Datasets for ResNet50

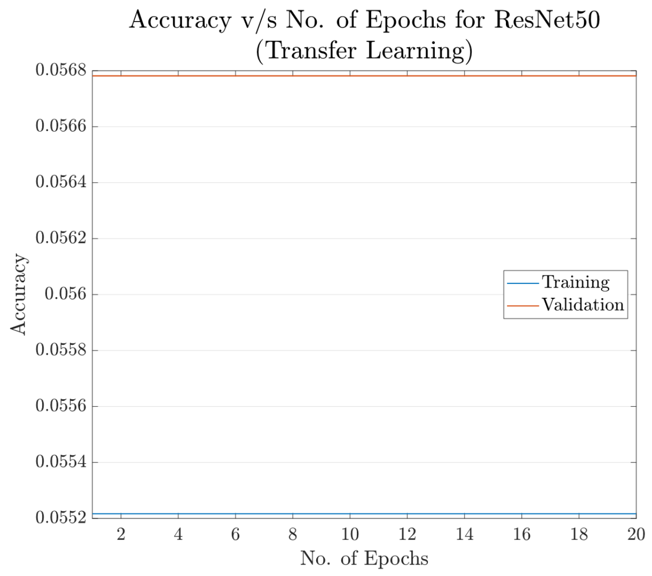

Figure 5 – Accuracy Scores for Training and Validation Datasets for Transfer Learning with ResNet50

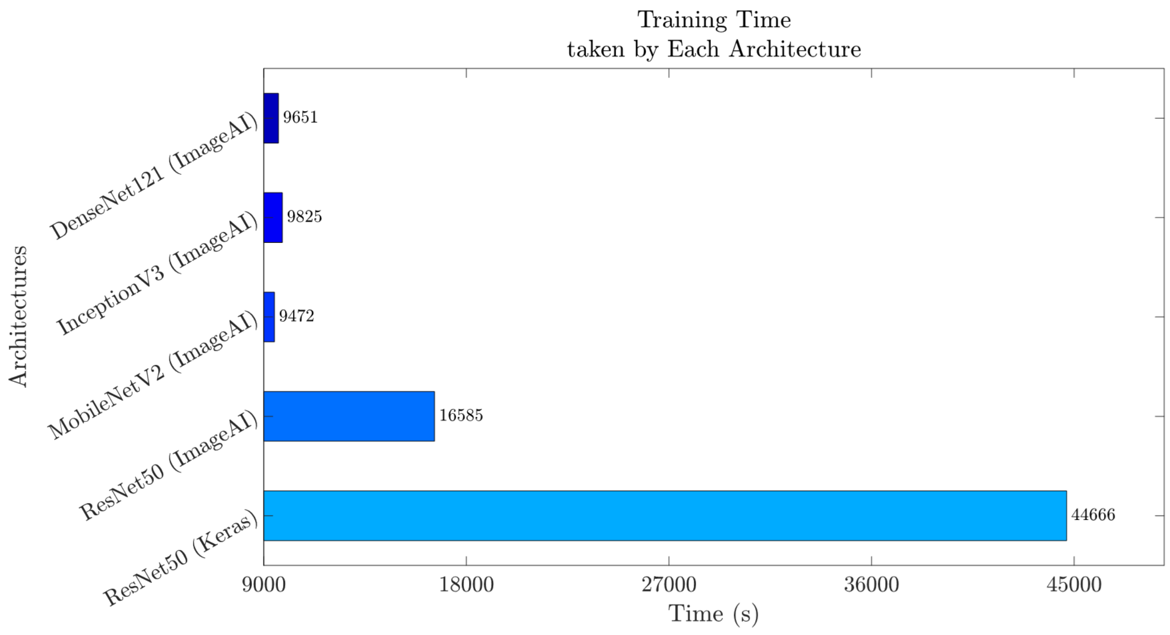

Figure 6 – Time Taken for Training by Each Architecture

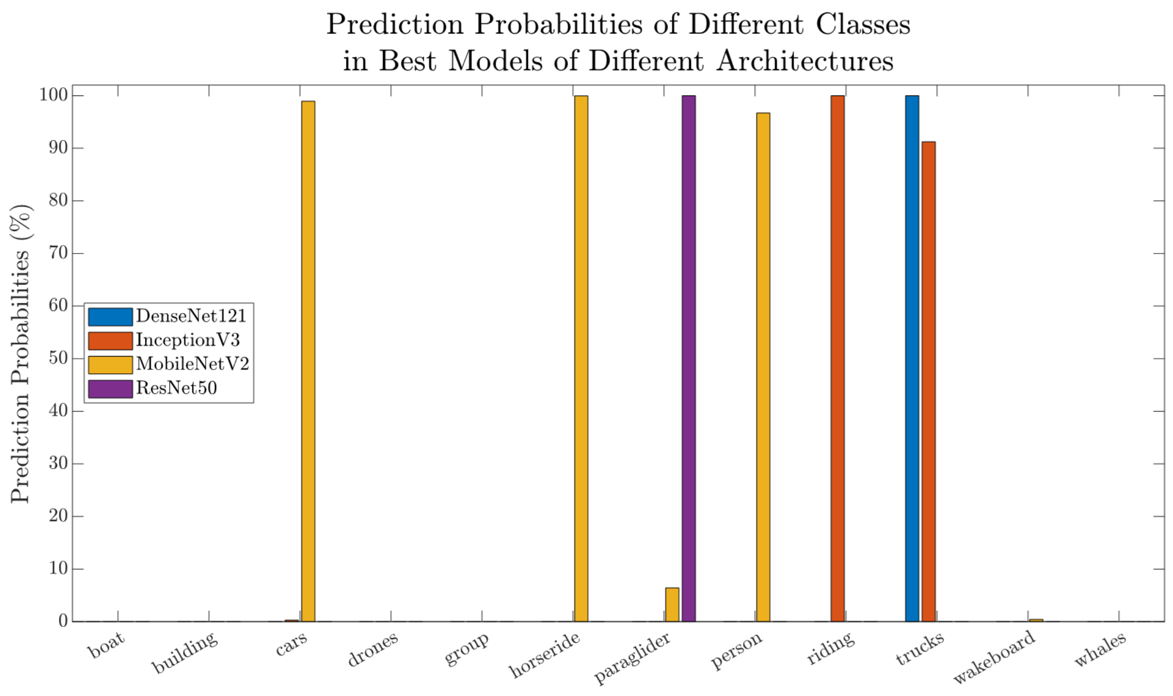

Figure 7 – Prediction Probabilities of Different Classes in Best Models of Different Architectures

5 UNEXPECTED EVENTS

I had originally planned to test the given dataset on object detection algorithms like YOLOv3, YOLOv4 and YOLOv5. However, after coming to realize the limitations of the computation power of NVIDIA GeForce RTX 3070 regarding training the images on object detection algorithms, I had to discard the plan and take up the plan for image classification instead.

6 LESSONS LEARNED

6.1 LESSONS LEARNT DURING THIS PROJECT

1. Mobile GPUs from NVIDIA still have to go a long way in delivering hardware-accelerated and cuDNN-powered deep learning capabilities to machine learning scientists and enthusiasts. Till then, discrete, parallel, multiple GPUs are the only way forward.

2. ImageAI has succeeded in delivering image classification algorithms with fewer lines of codes compared to Keras.

3. Architectures from Keras need so much of computational power that running them on mobile NVIDIA GPUs is still not possible.

4. Near-perfect accuracy and loss scores for training and validation datasets do not necessarily mean equally perfect classification scores for the test dataset.

6.2 RECOMMENDATION

In future, it would be far better to have, if possible, nearly equal number of images for each category for classification purposes. The bias towards one category could be reduced and sensible results could be obtained for test datasets, especially after obtaining near-perfect accuracy and loss scores for training and validation datasets.

7 PROJECT PERFORMANCE

7.1 ON TIME

The milestones were pretty much achieved within the deadlines I set for each one of them. In fact, since I had to discard the plan of transfer learning with architectures like DenseNet121, InceptionV3 and MobileNetV2 from Keras for reasons already stated above, I ended up being ahead of the deadlines on most of the occasions.

7.2 MEETING ORIGINAL EXPECTATIONS

The original expectation with the project was to move ahead with object detection algorithms. That expectation could not be met because of constraints in the computational power of NVIDIA GeForce RTX 3070, which is primarily a mobile GPU.

The other expectation was to conduct transfer learning with architectures such as DenseNet121, InceptionV3 and MobileNetV2 from Keras. This expectation was also not met as re-training the training dataset on the pre-trained architectures was taking nearly 12 hours for each one of them. Such a re-training time would have caused severe, irreversible damage to the mobile GPU and hence, pursuing it further, given the fact that accuracy and loss scores for training and validation datasets were also not improving with each epoch, would have been utterly insensible.