Nvidia Computer Vision Assignment

Abhishek Bhagwat, NUID: 001444721

Abstract: The purpose of this report is to learn the fundamentals of computer vison in deep learning and complete and evaluate the tasks that are part of the Nvidia Computer Vision Certification. The Nvidia course provides an interactive platform called as Digits where in we can build new deep learning models or use pre-existing state of the art models in CV like Alexnet, Googlelen etc. The Nvidia course consists of 6 tasks. The first task is to predict if the image contains Louie(dog) or it does not. The second task is to improve the performance of the model created in the previous task. Task 3 is to deploy the model created in the previous task so that it can be used to make decisions in real time. Task 4 is about using an existing model so that we don't have to spend time building a new model from scratch. Task 5 involves changing the network architecture to solve specific problems that we might have. Task 6 is the assessment to be completed to obtain the Nvidia certification.

Task 1:

The first GPU task is to predict where Louie (Beagle dog) is present in the image or not. For this we choose the dataset Images of Beagles. Selected a model, specified the number of epochs and then trained the model. The model was then fed images of dogs and was asked to predict whether the image was of Louie or not. For this task I ran 4 experiments by varying the parameters and the models available. The results and the setting of the experiments are documents as follows.

Experiment 1:

Epochs:10

Model: Alexnet

Learning Rate: 0.01(variable)

Accuracy: 77%

Loss: 0.58

Experiment 2:

Epochs:100

Model: Alexnet

Learning Rate: 0.01(fixed)

Accuracy: 99%

Loss: 0.39

Experiment 3:

Epochs:2

Model: Googlelen

Learning Rate: 0.01

Accuracy: 51%

Loss: 0.73

Experiment 4:

Epochs:100

Model: Googlelen

Learning Rate: 0.01

Accuracy: 100%

Loss: 0.0029

Task 1 Inference: It is observed that the accuracy of the model seems to improve more when we train them for a greater number of epochs and keep the learning rate fixed instead of variable. Greater number of epochs gives the network opportunity to learn better features about our data. Thus, improving its performance. The architecture Googlelen significantly outperformed Alexnet for the same problem with similar parameters.

TASK 2

In Task 1 the data used was limited and thus this impacted the performance of the model. In Task 2 we train the model with a larger dataset. The dataset used is called Cats and Dogs. The model had to predict whether the images is of a cat or a dog. In this task, I have simulated 4 experiments by varying the parameters and models and evaluated the results. The results and the settings of the experiments are as follows.

Experiment 1:

Epochs:5

Model: Alexnet

Learning Rate: 0.01

Accuracy: 79%

Loss: 0.43

Experiment 2:

Epochs:50

Model: Alexnet

Learning Rate: 0.01 (Variable)

Accuracy: 94%

Loss: 0.19

Experiment 3:

Epochs:50 (25 completed)

Model: Googlelen

Learning Rate: 0.01(Variable)

Accuracy: 96%

Loss: 0.10

Experiment 4:

Epochs: 30(25 completed)

Model: Googlelen

Learning Rate: 0.01(variable)

Accuracy: 95%

Loss: 0.11

Task 2 Inference: It is observed that more data we have the better our model performs. This is because using more data our model can learn more features and thus increase overall performance. This is one of the primary reasons why data augmentation is used to improve model performance. From the experiments, it observed that Googlenet outperforms Alexnet by a considerable margin. Googlenet gives better performance using half as many epochs as Alexnet. However, the training time for Googlelen is a lot more. Thus, it might be beneficial to use Alexnet for this problem to preserve resources. This shows its important to choose the model based on the resources available and the requirements of the problem too.

TASK 3

This task focuses on deployment. The models that we train to solve different types of problems have to be used to solve problems in real time. This can be done by developing applications that use the knowledge that our model has acquired to perform different types of tasks. In this task we deploy the model that we had had created in the previous task. Deployment requires that we write code in such a way that the input is passed to the network in the format the network expects. The output then again has to be converted in a way that can be interpreted by the end user.

We create a Doggie door that lets Louie (dog) enter that house and stops the cat(nala) fr0m entering the house. We use the weights that the model has learning after training and the architecture of the model. We then write a code in python that uses the model to predict if its Louie or Nala at the front door.

TASK 4

Its not necessary to build a model from scratch all the time. It is always possible that someone has worked on a similar problem and dataset and has a model ready which can be used a base model to solve related problems. In this task we use the model that we had created in the previous task to identify cats and dogs and further train it to improve our accuracy. In this task, I simulated 4 experiments by varying the parameters and architecture and their output is as follows.

Experiment 1:

Epochs:5

Model: Dogs and Cats

Learning Rate: 0.01(variable)

Accuracy: 80%

Loss: 0.44

Optimizer: SGD

Experiment 2:

Epochs:50

Model: Dogs and Cats

Learning Rate: 0.001(fixed)

Accuracy: 89%

Loss: 0.25

Optimizer: SGD

Experiment 3:

Epochs: 50

Model: Dogs and Cats

Learning Rate: 0.001 (fixed)

Accuracy: 85%

Loss: 0.32

Optimizer: SGD

Experiment 4:

Epochs:50

Model: Dogs and Cats

Learning Rate: 0.001

Accuracy: 93%

Loss: 0.23

Optimizer: Adam

TASK 5

This task focuses on object detection. The purpose is to detect objects and localize them in an image. For this purpose, I used the same dataset and the model to make use of their trained weights and architecture. The we take images and use our model to predict whether it's a dog or a cat. The network then provides us with the output. If the network provides an output which says that the animal in the image is a dog. This is the sliding window approach. sliding window approach involves deploying an image classifier trained to classify 256X256 portions of an image as either "dog" or "cat" to an application that moves across an image one 256X256 section at a time. Then the next problem we have is to locate in which part of the image the dog is present. The output layer consists of the softmax function. The function take k number of inputs and normalizes it into probability distribution consisting of k probabilities.

Further, we iterate over the image and classify each grid_square to create a heatmap. The output is represented by blue where there is no dog and red where there is a dog.

We can also tweak the architecture of the model to suit our requirement. We visualize the Alexnet network using the visualize feature in the Digits platform. We replace a layer named fc6 in the network with a new layer conv6. However, after visualization we can still see that fc6 is present in the architecture. This is because other layers fc7, drop6 and relu6 reference f6. We then make changes so that the layers fc7, drop6 and relu6 now connect to our new layers instead of fc6.

We removed a fully connected layer fc6 and replaced it with a convolutional layer conv6. By converting AlexNet to a "Fully Convolutional Network" we'll be able to feed images of various sizes to our network without first splitting them into grid squares. We then replace fc7 through fc8 will convolutional layers. We then save a run the model. We can observe that the Fully Convolutional Neural network detects the dogs with greater accuracy compared to sliding window approach. Its also less time consuming compared to sliding window approach.

To further improve the performance, we will use an end to end object detection network used to detect and localize dogs. We use the DetectNet trained using the COCO dataset. We can thus use specialized network to solve the problems that we want.

Task 6:

Completed the assessment required to obtain the Nvidia certification. The assessment involved training a model on the cats and dogs dataset and to obtain accuracy of greater than 80 and then using the models architecture and weights to deploy it.