Deeper Network for Image Classification

Bhanu Pratap Singh (210760006)

Ec21279@qmul.ac.uk

***Abstract*- This paper shows the performing and evaluating image classification task with the deeper networks. This paper analyses ResNet, VGG16, GoogleNet and their performance in the classification task of MNIST and CIFAR-10 dataset. In this report the models are critically discussed. Next the models are trained on MNIST and the CIFAR-10. It is shown that the MNIST keep stability where as CIFAR-10 often suffers from the vanishing gradient problem. After that modifications are made to the models where we see the benefits of batch normalisation with CIFAR-10 om instability.**

Keywords*- *Deep Learning, Computer Vision, Image Classification, VGG, GoogleNet, Resnet, MNIST, CIFAR-10

I.INTRODUCTION

Deep learning is the field of machine learning which is used in speech recognition, image classification and various classification problems. Image classification is used to identify weather the object is present in the picture or to determine the main feature of the image. Deep learning uses the set network structure to learn the hierarchical structural feature of the image completely from the training data and can extract abstract feature that are closer to the higher-level semantic of the image, so the performance in image recognition far exceeds that of traditional method.

An in-depth discussion of both VGG and GoogLeNet will be presented in this report as well as an analysis of the strengths and shortcomings of each in relation to competing models and an introduction of the methodology used in both models. Next, their performance will be tested on two popular datasets, MNIST and CIFAR-10. The models will be recreated so that they can have these datasets inputted in them and so that they can be classified. Following this, certain modifications will be put in place in order to seek improvement in the model's performances. This is done applying techniques, such as batch normalization as discussed in [51, kernel regularization and adjusting the learning rate amongst others, along with changes which were not necessarily implemented in the original architectures, such as reducing stride and batch sizes.

II.CRITICAL ANALYSIS /RELATED WORK

Image classification is an important task used in image processing. In the field of traditional machine learning, the standard process to identify and classify images one by one is feature extraction, feature screening and finally the feature vector is input into an appropriate classifier to complete feature classification. With the help of deep learning algorithms, the three modules of image feature extraction, screening and classification were integrated into one, and the depth of 5 layers of convolutional layers and 3 layers of fully connected layers was designed. Convolutional neural network structure, layer by layer extraction of image information in different directions, for example, shallow convolution usually obtains general features such as image edges, and deep convolution generally obtains specific distribution characteristics of a specific data set.

AlexNet won the 2012 ILSVRC annual championship with a record low error rate of 15.4%. It is worth mentioning that the error rate of the runner-up winner that year was 26.2%. In 2014, GoogleNet took another approach to improve the recognition effect from the perspective of designing the network structure [3]. Its main contribution is to design the Inception module structure to capture features of different scales and reduce dimensionality through 1×1 convolution. Another work in 2014 was VGG, which further proved the importance of the depth of the network in improving the effect of the model [5]. In 2015, He K et al. In order to solve the problem of first saturation and then decrease in accuracy in training, the concept of residual learning was introduced into the field of deep learning. The core idea is to use all the following layers when the neural network reaches saturation in a certain layer. Another reason for the success of the VGG model was the utilization of smaller receptive fields of size 3x3 used throughout all the VGG configurations, with there even being three 1×1 receptive fields applied to configuration C. This is significantly smaller than the 11×11 receptive field which the rival CNN model of

Krizhevsky et al. [5] has implemented. The use of smaller receptive fields results in the reduction of computational complexity in the VGG model [6], perhaps counteracting the complexity gained from the networks depth. This, nonetheless, begs the question of whether using a larger receptive field could lead to further improvement in the performance. albeit at the sacrifice of computational are simplicity. But in ResNet, a part of the convolutional layer is short circuited. When the training is saturated, the goal of all the next layers becomes a function of mapping f(x)=0. The emergence of residual learning ensures the stability of network training on the premise of deepening the network depth and improving the performance of the model. In 2015, ResNet also won the 2015 ImageNet challenge with an ultra-low error rate of 3.6%.

Szegedy et al. used auxiliary classifiers to first counter the vanishing gradient issue by bettering the models training convergence [6]. Unfortunately, they did not improve convergence during the early training period meaning that they would have to be trained for a longer time for these to finally have an effect on the accuracy. To avoid this issue, they introduced batch normalization as described by loffeer al. [7] into the auxiliary classifiers to act as a regularizer. The performance of the main classifier does increase with the introduction of batch normalization, but they admit that this is only weak evidence for it acting as a regularizer.

III. METHOD/MODEL DESCRIPTION

A. *VGG16*

VGGNet was born out of the need to reduce the number of parameters in convolution layers and to make the training more efficient. It is an example of CNNs that highlight spatial exploitation. The VGG uses about 13 million parameters which can be considered as one of the main drawbacks of VGG. VGG uses a fixed filter of size 3x3 in the hidden layers to help reduce the number of variables. It is a 19 layered network that successfully demonstrated that concurrent placement of small sized filters like (3x3) could induce the same effect as a large sized filter such as (5x5) or (7x7). A max-pooling layer, placed after the convolutional layer helps tune the network.

VGG is a 19 layer deep network that uses fixed size kernels for convolutions. The original architecture of VGG takes a 224x224 GB image as input which goes through the network of hidden layers. The number of hidden lavers depends on the variant off VGG being used. I'm making use of VGG16 for this coursework which has 16 hidden layers. The network has a fixed 3x3 filter with stride 1, five MaxPool layers, each after a batch of convolution layers using a 2x2 kernel with stride 2. These help with the fine tuning of the network. ReLU activation has been applied to these layers, which are followed by the Dense layers having 4096neurons and the last output with softmax activation.

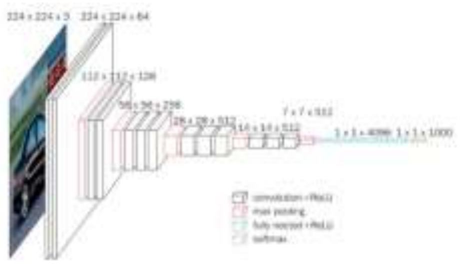
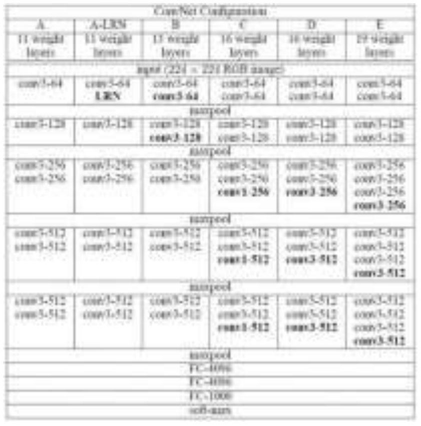


Figure 1: VGG16 Architecture



ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure2: VGG Configuration

B. *GoogleNet*

GoogLeNet was the first one to introduce the block concept for CNNs. It incorporates multi-scale convolutional transformations using split, transform and merge ideas called Inception blocks. These blocks use kernels of different sizes to effectively capture spatial information at scales. GoogLeNet replaces the conventional convolution layers with blocks, each block being a network in itself. It uses Inception blocks and thus follows a similar architecture as inception. It has two base architectures viz, the naive version and The dimensionality reduction version. The naïve one uses 3 different sized kernels 1xl, 3x3, 5x5 to perform convolutions combined with one max pooling layer. Outputs from these layers are concatenated to be passed on the module ahead. Adding an extra xl convolution layer before the 3x3 and 5x5 convolution layer reduces the input channels for the next module, thus regulating the computation complexity of the model.

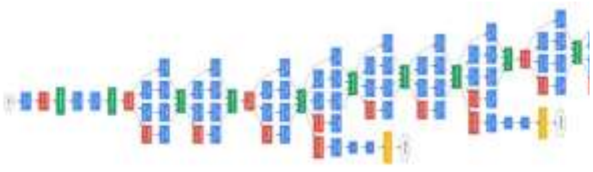
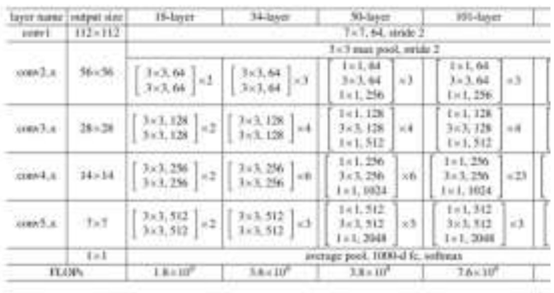


Figure3: GoogleNet Architecture

C. *ResNet*

ResNet architecture uses the CNN blocks multiple times, so let us create a class for CNN block, which takes input channels and output channels. There is a batchnorm2d after each conv layer. Then create a ResNet class that takes the input of a number of blocks, layers, image channels, and the number of classes. The Figure 4 shows the different layer configuration of ResNet.



layer name	output size	18-layer	34-layer	50-layer	101-layer
conv1	112×112	7×7, 64, stride 2			
conv2_x	56×56	3×3 max pool, stride 2			
		$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$
conv4_x	14×14	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times23$
conv5_x	7×7	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax			
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9

Figure4: ResNet Architecuture

IV.EXPERIMENTS

A. Datasets

This paper uses MNIST and CIFAR-10 datasets for performance analysis and evaluation.

a) MNIST

MNIST data mainly composed of some pictures of handwritten digits and corresponding labels. MNIST dataset has 10 types of pictures, corresponding to 0-9. MNIST dataset is a dataset of 60000 28x28 gray-scale images of 10 digits, along with a set of 10000 images. The digits have been size-normalised and centred in a fixed-size image.

Figure5: MNIST dataset examples

b) CIFAR-10

CIFAR-10 dataset consists of 60000 32x32 RGB images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. The dataset is divided into five training batches and one training batch contains 10000 images. The test batch exactly contains1000 random images. The training batch contain 5000 images.

Figure6 : CIFAR-10 dataset examples

B. Training Models

a) MNIST

1.VGG16:This paper uses the origin model of “Very Deep Convolutional Networks for Large scale image Recognition”[1]
2. GoogLeNet: This paper uses the origin
model of "Going Deeper with Convolutions" [2]. There are some changes with the batch normalisation.
3. ResNet18: Based on "Deep residual learning
for image recognition" [1]
4. ResNet34: Based on "Deep residual learning
for image recognition" [1]

b)CIFAR-10

1.VGG16: This paper uses the origin model of “Very Deep Convolutional Networks for Large scale image Recognition”[1]
2. GoogLeNet: This paper uses the origin
model of "Going Deeper with Convolutions" [2]. There are some changes with the batch normalisation.
3. ResNet18: Based on "Deep residual learning
for image recognition" [1]
4. ResNet34: Based on "Deep residual learning

for image recognition" [1]

Table1: Training Accuracy for both datasets

MODEL	MNIST	CIFAR-10
ResNet18	99.3	93.1
ResNet34	99.4	89.8
GoogleNet(without batch normalisation)	99.0	62.8
GoogleNet (with batch normalisation)	99.1	87.6
VGG16	99.8	74.1

From the analysis of table 1 we can conclude that the *ResNet34*, *VGG*16 structure gives the best training accuracy for the *MNIST* dataset while for the *CIFAR-10* dataset *ResNet18* model gives the highest accuracy.

C. Results

a) Validation Accuracy

Table2: Validation Accuracy for both datasets

MODEL	MNIST	CIFAR-10
ResNet18	99.0	78.0
ResNet34	99.5	79.5
GoogleNet(without batch normalisation)	98.1	61.2
GoogleNet (with batch normalisation)	99.08	79.4
VGG16	98.9	70.8

From the analysis of table 2 we can conclude that the ResNet34 structure gives the best validation accuracy for the *MNIST* dataset while for the *CIFAR-10* dataset ResNet18 and *GoogleNet* with batch normalisation models gives the highest accuracy.

b) Testing

While testing the models on both the datasets the result is given in below table

Table3: Testing Accuracy for both datasets

MODEL	MNIST	CIFAR-10
ResNet18	99.5	76.38
ResNet34	98.8	79.1
GoogleNet(without batch normalisation)	98.6	60.5
GoogleNet (with batch normalisation)	99.01	79.5
VGG16	98.4	71.4

From the analysis of table 3 we can conclude that the *ResNet18* structure gives the best validation accuracy for the *MNIST* dataset while for the *CIFAR-10* dataset ResNet34 and *GoogleNet* with batch normalisation models gives the highest accuracy.

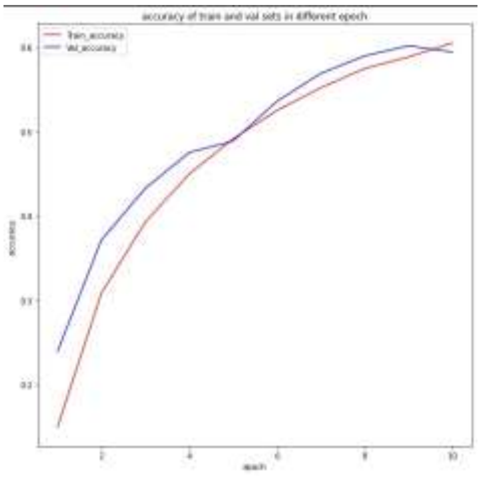


Figure7: GoogleNet without Batch Normalisation

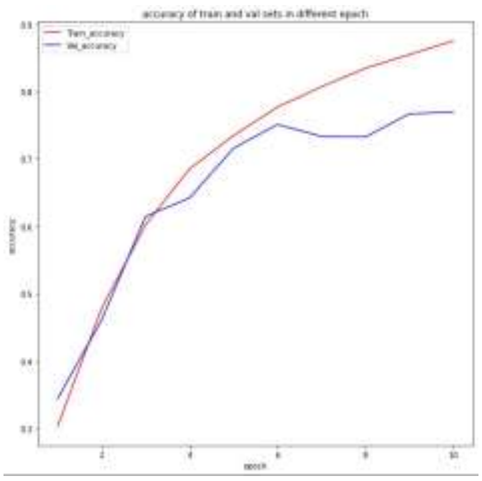


Figure8: GoogleNet with Batch Normalisation

Figure 7 and figure 8 shows the testing accuracy of GoogleNet model without batch normalisation and with batch normalisation respectively on both dataset MNIST and CIFAR-10. We can analysed that the with batch normalisation it giving the better accuracy on both dataset as compared to without batch normalisation. Also, batch normalisation is helping the model to train faster.

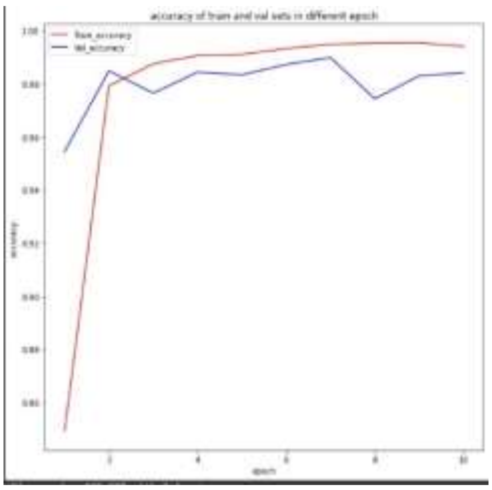


Figure9: ResNet18 accuracy on MNIST

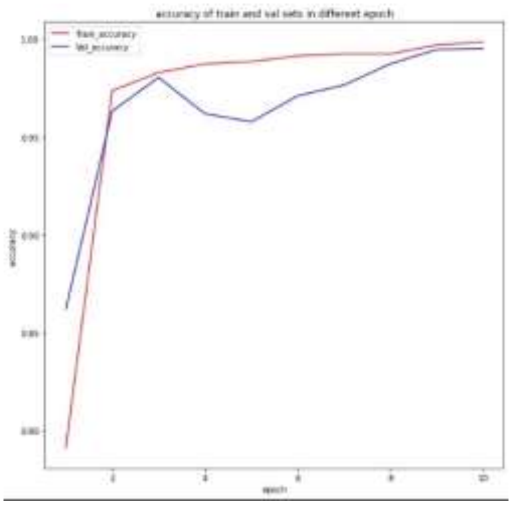


Figure10: ResNet34 accuracy on MNIST

Figure 9 and figure 10 shows the ResNet18 and ResNet34 accuracy on MNIST dataset respectively. As we can analysed from the figure that ResNet18 is giving the better accuracy for MNIST dataset. The accuracy is low in ResNet34 because it is deeper as compared to the ResNet18. The CIFAR-10 accuracy curve on both ResNet18 and ResNet34 is almost same as the MNIST dataset.

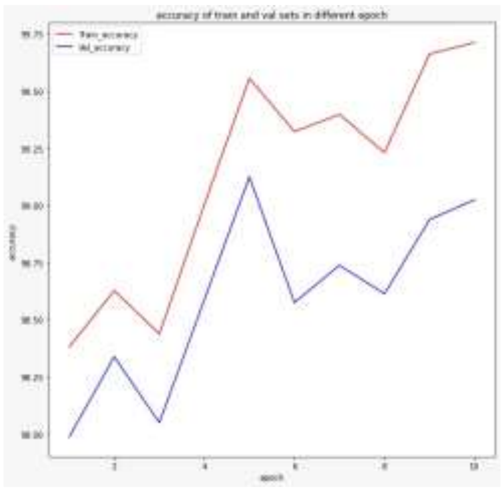


Figure11: VGG16 accuracy on MNIST

Figure11 shows the accuracy of VGG16 on the MNIST dataset. From the figure we can analyse that that the model is learning best from its training data. The curve is almost same for the CIFAR-10 dataset.

D. Further Modification

Modification 1 by reducing the learning rate of the model we observe that the accuracy of model increases by reducing the learning rate from 0.1 to 0.01 we noticed that model is learning better.

Table4: Accuracy for different learning rate

Models	Learning Rate 0.1	Learning Rate 0.01
ResNet18	97.3	99.0
ResNet34	96.5	99.5
GoogleNet(without batch normalisation)	97.5	98.1
GoogleNet(with batch normalisation)	97.9	99.08
VGG16	97.5	98.9

From the table 4 we compare the learning rates of different model on MNIST dataset. We analysed that the decreasing the learning rate gives us the better accuracy.

Modification 2 by introducing the batch normalisation in the GoogleNet we observe that the accuracy increased as compared to the GoogleNet without batch normalisation because by introducing batch norm layer in the output it increases its training time.

V.CONCLUSION

Many modifications of the original architecture of GoogleNet, VGG and ResNet have been discussed and experimented with to determine how to best approach the classification task on both CIFAR-10 and MNIST dataset.

The experimental results yield solid evidence that compact structure and batch normalization help VGG, GoogLeNet improve performance in terms of training time and error rate, meanwhile verify that ResNet applying PReLU does ease overfitting. In addition, data augmentation can enhance the training of the deep networks, but there is a risk of missing useful information.

On the other hand, stability was a major issue for training on CIFAR-10 with the initial architectures of VGG and GoogLeNet, with the vanishing gradient issue often occurring. It was found that batch normalization was a significant aid lo combating this problem, with it acting as a regularize [5]. In the GoogleNet model, performance was enhanced through reducing the stride size and the filter size to 1 and 5×5 respectively into the first convolution layer. probably due to the fact that information was previously lost using larger Alters and stride sizes, which may have been necessary for *Szegedy et al*. when training on a much larger ILSVRC dataset. However, a limitation of this reduction was more intensive computation which would require more powerful GPUs for shorter training limes. Also, reducing the batch size fed into the GoogleNet model for training helped improve performance, but again, training time would have increased. Therefore, if time permitted, more epochs would have been used testing with smaller filter sizes and smaller batches to search for better performance in GoogLeNet, Also, *Szegedy et al*. used an auxiliary classifier in their model which may have helped with the convergence of the GoogLeNet model in this paper.

For this coursework, I perform image classification tasks with VGG, GoogLeNet and Reset and evaluate them on MNIST and CIFAR-10. In the meantime, do some modification to the original network architectures to enhance their performance. More experiments are needed in the future to improve the research of this article. Perhaps the use of data augmentation to generate image data mimicking the much smaller MNIST and CIFAR-10 datasets could have been cleverly incorporated into the training processes of the GoogLeNet and VGG-D models discussed to help boost their performances.

VI.REFRENCES

[1]Very Deep Convolutional Networks For Large-scale Image Recognition - Karen Simonyan, Andrew Zisserman, Visual Geometry Group, Department of Engineering Science, University of Oxford, ICLR (2015)

[2]Going Deeper with Convolutions – Christian Szegedy, Wei Li, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, Google Inc., University of North Carolina, Chapel Hill, University of Michigan, Ann Arbor, Magic Leap Inc., ILSVRC(2014), CVPR (2015)

[3]The Mist Database Of Handwritten Digits Yann LeCun, Courant Institute NYU, Corinna Cortes, Google Labs New York, Christopher J.C. Burges, Microsoft Research Redmond.

[4] Krishna ST, Kalluri HK. Deep learning and transfer learning approaches for image classification. International Journal of Recent Technology and Engineering (IJRTE). 2019 Feb;7(5S4):427-32.

[5] Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems. 2012:25:1097-105

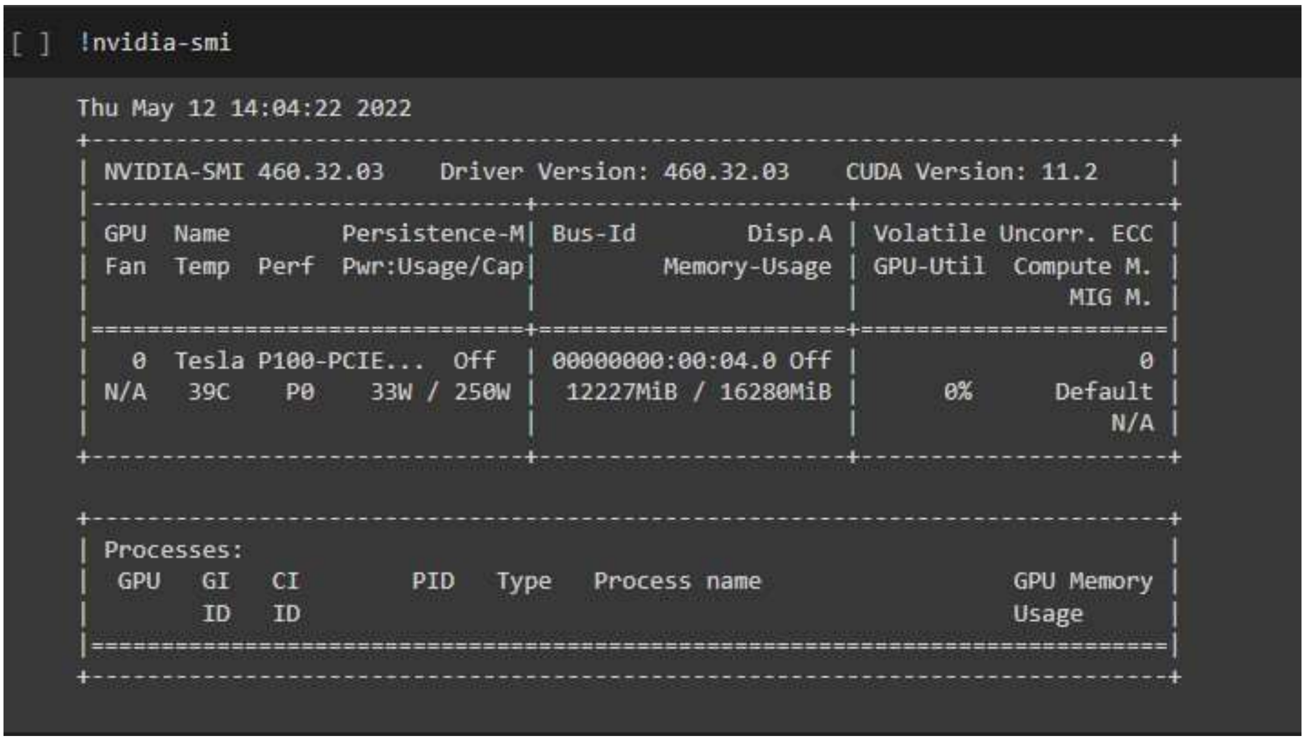
[6] Szegedy C, Vanhoucke V, loffe S, Shlens J, Wojna Z. Rethinking the inception architecture forcomputer vision. InProceedings of the IEEE conference on computer vision and pattern recognition 2016 (pp. 2818-2826).

[7] loffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167. 2015 Feb 11.

[8] Szegedy C, loffe S, Vanhoucke V, Alemi AA. Inception-v4, inception-resnet and the impact of residual connections on learning. In Thirty-first AAAl conference on artificial intelligence 2017 Feb12.

[91 Huang G, Liu Z, Van Der Maaten L, Weinberger KQ. Densely connected convolutional networks. InProceedings of the IEEE conference on computer vision and pattern recognition 2017 (pp.4700-4708).

APPENDIX:



```
[ ]  !nvidia-smi

     Thu May 12 14:04:22 2022
     +-----------------------------------------------------------------------------+
     | NVIDIA-SMI 460.32.03    Driver Version: 460.32.03    CUDA Version: 11.2     |
     |-------------------------------+----------------------+----------------------+
     | GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
     | Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
     |                               |                      |               MIG M. |
     |===============================+======================+======================|
     |   0  Tesla P100-PCIE...  Off  | 00000000:00:04.0 Off |                    0 |
     | N/A   39C    P0    33W / 250W |  12227MiB / 16280MiB |      0%      Default |
     |                               |                      |                  N/A |
     +-------------------------------+----------------------+----------------------+

     +-----------------------------------------------------------------------------+
     | Processes:                                                                  |
     |  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
     |        ID   ID                                                   Usage      |
     |=============================================================================|
     +-----------------------------------------------------------------------------+
```