COMS6998 - High Performance Machine Learning
Submission by - Saher Iqbal si2443

C2: Time Measurement of code in C1

```
Using device :  cuda
Files already downloaded and verified
********************************************************************************************

Running Part C2
100%|                                                      | 391/391 [00:20<00:00, 18.74it/s]
Epoch 1 Complete:
        Data-Loading Time: 12.474472 seconds
        Training Time: 7.977726 seconds
        Total Epoch Time: 20.865773 seconds

100%|                                                      | 391/391 [00:17<00:00, 22.57it/s]
Epoch 2 Complete:
        Data-Loading Time: 12.299684 seconds
        Training Time: 4.547659 seconds
        Total Epoch Time: 17.328946 seconds

100%|                                                      | 391/391 [00:17<00:00, 22.52it/s]
Epoch 3 Complete:
        Data-Loading Time: 12.310602 seconds
        Training Time: 4.588929 seconds
        Total Epoch Time: 17.367394 seconds

100%|                                                      | 391/391 [00:17<00:00, 22.43it/s]
Epoch 4 Complete:
        Data-Loading Time: 12.712015 seconds
        Training Time: 4.258908 seconds
        Total Epoch Time: 17.431723 seconds

100%|                                                      | 391/391 [00:17<00:00, 22.22it/s]
Epoch 5 Complete:
        Data-Loading Time: 12.723368 seconds
        Training Time: 4.406731 seconds
        Total Epoch Time: 17.595104 seconds


********************************************************************************************
```
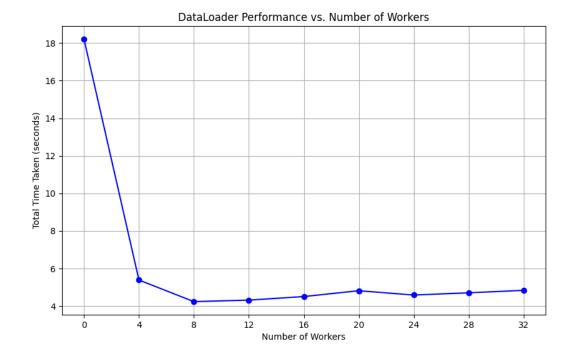
C3: I/O optimization for code in C2

(C3.1)

The time spent on data loading operation corresponding to the number of workers was obtained as the following:

worker_counts = [0, 4, 8, 12, 16, 20, 24, 28, 32]
Time spent on dataloading = [18.21, 5.39, 4.24, 4.32, 4.51, 4.82, 4.59, 4.71, 4.84]

DataLoader Performance vs. Number of Workers

(C3.2)

8 workers are needed for best runtime performance.

C4: Profiling starting from code in C3

The data-loading and the compute time taken when the number of workers are 8 is less than when the number of workers are 1, this is because of the following reasons:

1. Parallelism Reduces I/O Bottlenecks: Multiple workers load data in parallel, minimizing read times and efficiently using I/O resources.
2. Efficient GPU/CPU Utilization: Overlapping data loading with model training ensures constant data supply, maximizing computational resource usage.
3. Maximized Hardware Usage: Leveraging multi-core CPUs enhances processing speed, reducing overall training time.
4. Minimized Idle Times: Concurrent data processing decreases GPU/CPU wait times, speeding up epochs.

For Workers = 1:

```
Running Part C4
100%|                                                              | 391/391 [00:19<00:00, 19.63it/s]
Epoch 1 Complete:
        Data-Loading Time: 15.306376 seconds
        Training Time: 4.190655 seconds
        Total Epoch Time: 19.923492 seconds

100%|                                                              | 391/391 [00:20<00:00, 19.49it/s]
Epoch 2 Complete:
        Data-Loading Time: 15.496626 seconds
        Training Time: 4.135516 seconds
        Total Epoch Time: 20.058533 seconds

100%|                                                              | 391/391 [00:20<00:00, 19.43it/s]
Epoch 3 Complete:
        Data-Loading Time: 15.442163 seconds
        Training Time: 4.257587 seconds
        Total Epoch Time: 20.126625 seconds

100%|                                                              | 391/391 [00:20<00:00, 19.49it/s]
Epoch 4 Complete:
        Data-Loading Time: 15.537230 seconds
        Training Time: 4.104396 seconds
        Total Epoch Time: 20.066886 seconds

100%|                                                              | 391/391 [00:20<00:00, 19.49it/s]
Epoch 5 Complete:
        Data-Loading Time: 15.540941 seconds
        Training Time: 4.087969 seconds
        Total Epoch Time: 20.061603 seconds
```

For Workers = 8:

```
100%|                                                              | 391/391 [00:17<00:00, 21.76it/s]
Epoch 1 Complete:
        Data-Loading Time: 12.706402 seconds
        Training Time: 4.484733 seconds
        Total Epoch Time: 17.968760 seconds

100%|                                                              | 391/391 [00:18<00:00, 21.69it/s]
Epoch 2 Complete:
        Data-Loading Time: 12.728301 seconds
        Training Time: 4.503965 seconds
        Total Epoch Time: 18.029632 seconds

100%|                                                              | 391/391 [00:17<00:00, 21.74it/s]
Epoch 3 Complete:
        Data-Loading Time: 12.584895 seconds
        Training Time: 4.622347 seconds
        Total Epoch Time: 17.986342 seconds

100%|                                                              | 391/391 [00:17<00:00, 21.76it/s]
Epoch 4 Complete:
        Data-Loading Time: 12.594632 seconds
        Training Time: 4.563164 seconds
        Total Epoch Time: 17.970563 seconds

100%|                                                              | 391/391 [00:17<00:00, 21.79it/s]
Epoch 5 Complete:
        Data-Loading Time: 12.477023 seconds
        Training Time: 4.681191 seconds
        Total Epoch Time: 17.948173 seconds
```

C5: Training in GPUs vs CPUs

On GPU:

```
*********************************************************************************

Running C5 on CUDA
100%|                                                        | 391/391 [00:17<00:00, 21.78it/s]
100%|                                                        | 391/391 [00:17<00:00, 21.77it/s]
100%|                                                        | 391/391 [00:17<00:00, 21.75it/s]
100%|                                                        | 391/391 [00:17<00:00, 21.78it/s]
100%|                                                        | 391/391 [00:17<00:00, 21.81it/s]
        Average Time over 5 Epochs: 17.956238 seconds


*********************************************************************************
```

On CPU:

```
*********************************************************************************

Running C5 on CPU
device: cpu
100%|                                                        | 391/391 [09:14<00:00,  1.42s/it]
100%|                                                        | 391/391 [09:21<00:00,  1.44s/it]
100%|                                                        | 391/391 [09:19<00:00,  1.43s/it]
100%|                                                        | 391/391 [09:17<00:00,  1.43s/it]
100%|                                                        | 391/391 [09:18<00:00,  1.43s/it]
        Average Time over 5 Epochs: 558.418599 seconds

*********************************************************************************
```

C6: Experimenting with different optimizers

```
Running C6

Using SGD

100%|                                                        | 391/391 [00:17<00:00, 21.91it/s]
100%|                                                        | 391/391 [00:17<00:00, 21.76it/s]
100%|                                                        | 391/391 [00:18<00:00, 21.51it/s]
100%|                                                        | 391/391 [00:18<00:00, 21.54it/s]
100%|                                                        | 391/391 [00:18<00:00, 21.68it/s]
        Average Training Time over 5 Epochs: 4.597331 seconds

        Training Loss: 0.885713

        Training Accuracy:75.00%

Using SGD with nesterov

100%|                                                        | 391/391 [00:18<00:00, 21.43it/s]
100%|                                                        | 391/391 [00:18<00:00, 21.41it/s]
100%|                                                        | 391/391 [00:18<00:00, 21.38it/s]
100%|                                                        | 391/391 [00:18<00:00, 21.37it/s]
100%|                                                        | 391/391 [00:18<00:00, 21.39it/s]
        Average Training Time over 5 Epochs: 4.590066 seconds

        Training Loss: 0.882991

        Training Accuracy:67.50%

Using Adgrad

100%|                                                        | 391/391 [00:18<00:00, 21.27it/s]
100%|                                                        | 391/391 [00:18<00:00, 21.21it/s]
100%|                                                        | 391/391 [00:18<00:00, 21.30it/s]
100%|                                                        | 391/391 [00:18<00:00, 21.38it/s]
100%|                                                        | 391/391 [00:18<00:00, 21.39it/s]
        Average Training Time over 5 Epochs: 4.811080 seconds

        Training Loss: 1.180395

        Training Accuracy:56.25%
```

Using Adadelta
100%| | 391/391 [00:19<00:00, 20.16it/s]
100%| | 391/391 [00:19<00:00, 20.19it/s]
100%| | 391/391 [00:19<00:00, 20.20it/s]
100%| | 391/391 [00:19<00:00, 20.17it/s]
100%| | 391/391 [00:19<00:00, 20.21it/s]
        Average Training Time over 5 Epochs: 4.891982 seconds

        Training Loss: 0.470770

        Training Accuracy:83.75%

Using Adam
100%| | 391/391 [00:18<00:00, 21.12it/s]
100%| | 391/391 [00:18<00:00, 21.35it/s]
100%| | 391/391 [00:18<00:00, 21.48it/s]
100%| | 391/391 [00:18<00:00, 21.45it/s]
100%| | 391/391 [00:18<00:00, 21.60it/s]
        Average Training Time over 5 Epochs: 5.011298 seconds

        Training Loss: 1.828660

        Training Accuracy:21.25%

C7: Experimenting without Batch Norm

100%| | 391/391 [00:15<00:00, 25.24it/s]
100%| | 391/391 [00:15<00:00, 25.36it/s]
100%| | 391/391 [00:15<00:00, 25.47it/s]
100%| | 391/391 [00:15<00:00, 25.39it/s]
100%| | 391/391 [00:15<00:00, 25.48it/s]
        Average Training Time over 5 Epochs: 3.192359 seconds

        Training Loss: 0.769391

        Training Accuracy:75.00%

Q.1. How many convolutional layers are in the ResNet-18 model?

ResNet-18 model consists of 17 convolutional layers. The first layer is a single convolutional layer, followed by 4 main blocks, each of which contains 2 sets of 2 convolutional layers within the basic blocks. So, 1 + (4*2*2) = 17 convolutional layers.

Q.2. What is the input dimension of the last linear layer?

The input dimension of the last linear layer of the ResNet-18 model is **512.**

Q.3. How many trainable parameters and how many gradients in the ResNet-18 model that you build (please show both the answer and the code that you use to count them), when using SGD optimizer?

```
Running Q3 using SGD optimizer
Number of trainable parameters: 11173962
Number of gradients: 11173962
```

Q.4. Same question as Q3, except now using Adam (only the answer is required, not the code).

```
Using Adam optimizer for Q4

Number of trainable parameters: 11173962
Number of gradients: 11173962
```