Luis Alvarado Rivas
SFSU ID: 920763269

Matrix Multiplication Speed Using Python and Torch

1. Introduction

By taking advantage of the parallel processing capabilities of a GPU matrix multiplications can be done much faster than with plain single threaded python code. To achieve this we can use the torch library and python. In this document I will demonstrate the speed up one can achieve through parallel processing.

2.Method

To properly show the difference we will use the timeit and matplotlib libraries. Timeit will allow us to measure the time it takes to run the plain python code and the torch code and matplotlib will allows us to show the resulting data in a graph. The code will be run using a 3070 NVIDIA GPU by using CUDA. The code will be run 10 times with different values of m as the column count of matrix W and the row count of matrix X.

For the plain multiplication we will be using nested loops. In the code below we can see how each row and column of W and X is iterated over and multiplied to create the resulting matrix.

```
20
21  def plainMatrixMatrixMultiply(X, W, b):
22      counter = 0
23      outputs = []
24      for w in W:
25          outputRow=[]
26          for xCol in zip(*X):
27              output = b
28              for w_j, x_j in zip(w, xCol):
29                  output += w_j * x_j
30              outputRow.append(output)
31          outputs.append(outputRow)
32      return outputs
```

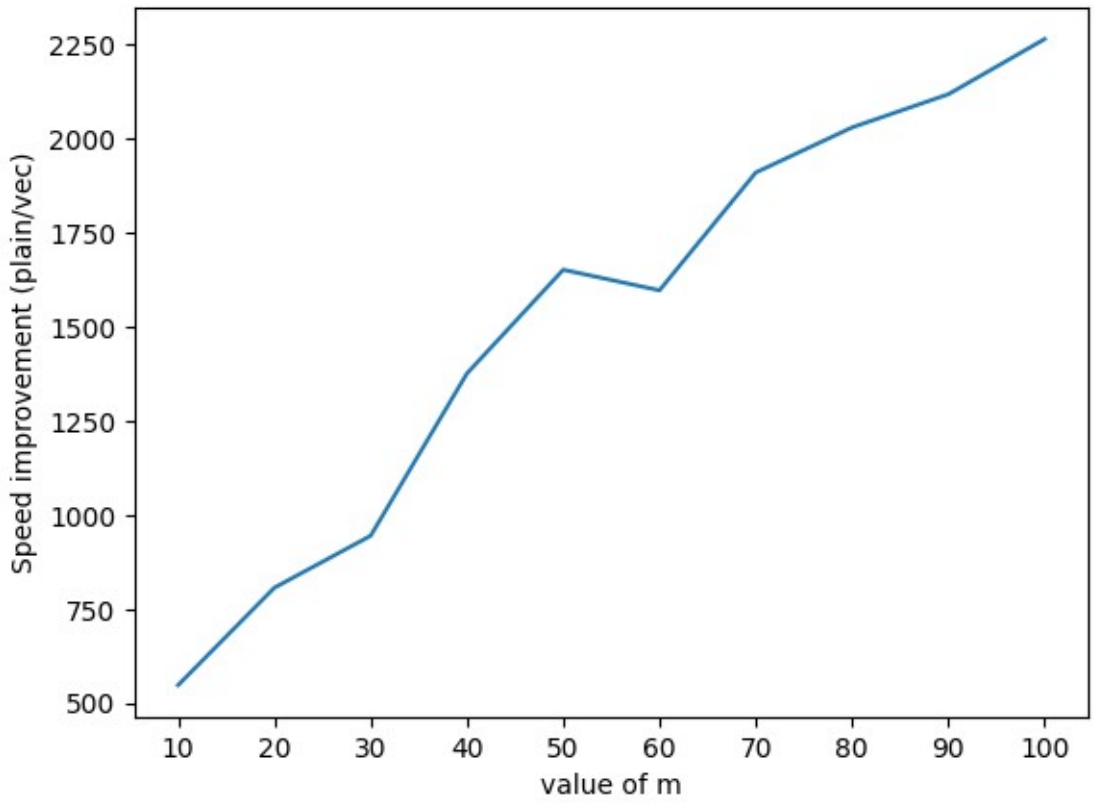
In comparison, the torch code is much simpler in this file. The matrices are converted to torch tensors and the matmul function is called to multiply the matrices.

```
t_b = torch.tensor(b)
X_t = torch.tensor(X)
W_t = torch.tensor(W)
res_pytorch_matrix = timeit.timeit ("W_t.matmul(X_t) + t_b"
```

3.Results

Each function is run 2000 times to get a more accurate measurement of the speed difference. These are the results:

```
C:\Users\OrangeCatBears\Documents\CompSci HW\Deep Learning\Deep-Learning-CSC-project (main -> origin)
λ py .\Homework2.py
cuda available
speedup = 548.9247857226213
speedup = 807.6132454712813
speedup = 945.604534959916
speedup = 1376.599448898609
speedup = 1652.0658683077766
speedup = 1597.4480128452449
speedup = 1910.2651379091294
speedup = 2029.9459446774592
speedup = 2118.105408735125
speedup = 2264.6565651282285
```



4.Analysis

The difference between the two methods is very large. Even in the scenario with the smallest difference we can see that the torch code is 548 times faster than the plain python code. As the matrices become larger that difference increases in an uneven but mostly linear way. The variance may come

from the fact that other programs were being run on the computer at the same time.

5.Conclusion

It is clear from the results that torch excels at multiplying matrices of large sizes due to its use of parallel programming. This advantage is increased when a powerful GPU can be used.