ECE5545/CS5775 A4

Naman Makkar

TOTAL POINTS

16 / 17

QUESTION 1

1 Functionality 10 / 10

✓ **+ 5 pts** *Conv Test all passed*

+ 4 pts Conv Test passed > 60/84

+ 3 pts Conv Test passed > 40/84

+ 2 pts Conv Test passed > 20 / 84

+ 1 pts Conv test failed

✓ **+ 5 pts** *Matmul test all passed*

+ 1 pts Matmul test failed

QUESTION 2

2 Report 5 / 6

+ 2 pts Q3 (quantization) Complete

✓ **+ 1 pts** *Q3 (quantization) quality*

✓ **+ 2 pts** *Q4 (SVD) complete*

+ 1 pts Q4 (SVD) quality

✓ **+ 2 pts** *Q5 (SVD + NN) complete*

+ 1 pts Q5 (SVD + NN) Quality

+ 0 pts Not complete

QUESTION 3

3 Logmul Extra Credits 1 / 1

✓ **+ 1 pts** *Attempt logmult extra credits.*

+ 0 pts Click here to replace this description.

ML Hardware Assignment 4

Naman Makkar (nbm49)

May 2023

1 Experiments on Numerical Precision

The plots for Numerical Precision vs Reconstruction Error were plotted for Winograd Convolution, Fast Fourier Transform and Log Matrix Multiplication. The reconstruction error utilized here was the L2 loss function. The values of numerical precision used were - **Float 16, Float 32, Float 64**. The maximum reconstruction error was observed in **Float 16 precision** while the lowest reconstruction error was observed for Float 64. It was also observed that Log Matrix Multiplication had a marginally higher reconstruction error than the other 2 methods. Additionally, the reconstruction error reduced with increasing precision. The plots can be observed in **Figures 1-4**.

2 Experiments on SVD Reconstruction Error and Speedup

For the first set of experiments with SVD, the reconstruction error of matrices was plotted with the ranks of the low rank approximations of the matrix. The results of these experiments can be observed in **Figures 5-9** The second set of experiments, the runtime speedup and the FLOPs speedup of SVD was calculated on a matrix multiplication operation between the weights of a linear layer and the activation of the previous layer. The plots for the second set of experiments can be observed in **Figures 10-12** where the rank vs runtime speedup and rank vs FLOPs speedup have been plotted for the matrix multiplication. It was observed that the lowest rank values achieved the greatest FLOPs speedup whereas the greatest runtime speedup was observed for ranks between $\mathbf{2^4}$ and $\mathbf{2^5}$.

3 MNIST Training

The given fully connected neural network was trained on the MNIST dataset and low rank approximation was carried out on the model with the help of SVD. SVD was utilised to carry out low rank approximation on the first and the second fully connected layers of the model. For the first fully connected layer ranks were

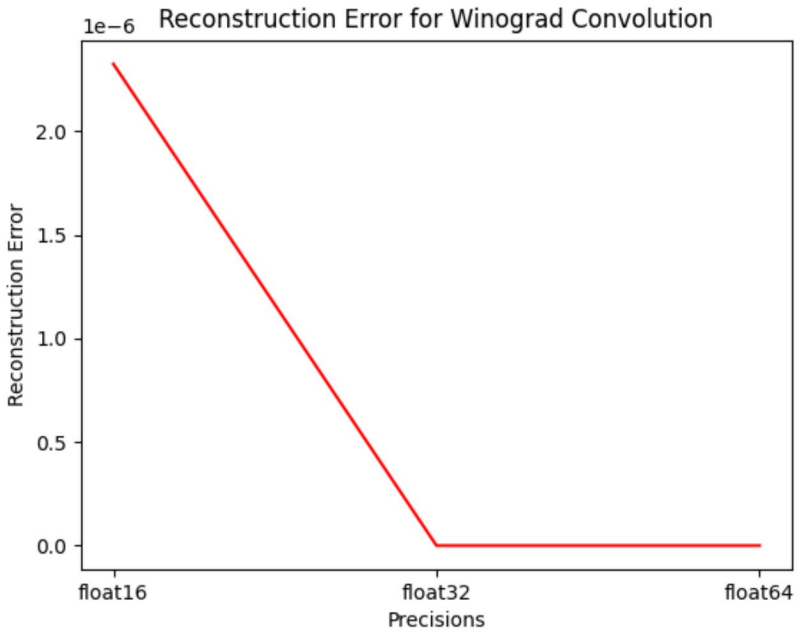

Figure 1: Precision vs Reconstruction Error for Winograd Convolution

Figure 2: Precision vs Reconstruction Error for Fast Fourier Transform

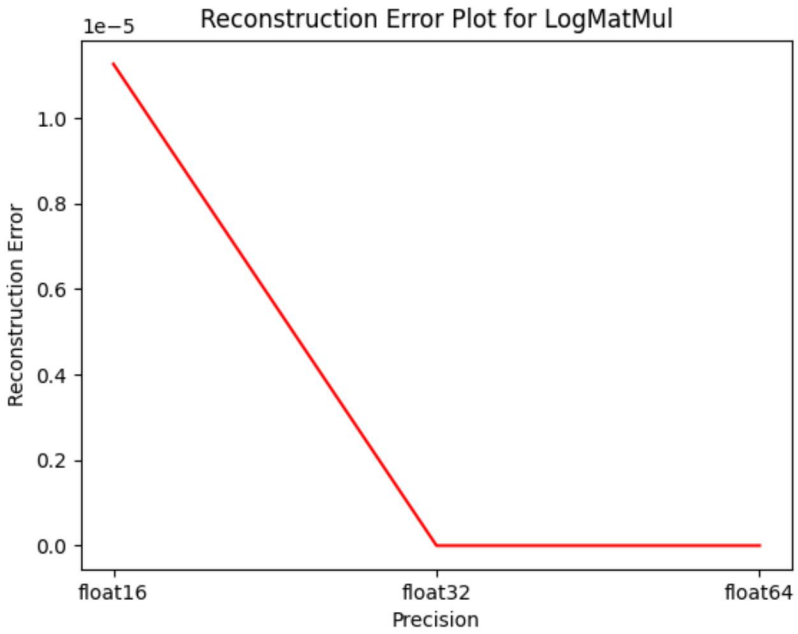

Figure 3: Precision vs Reconstruction Error for Log Matrix Multiplication

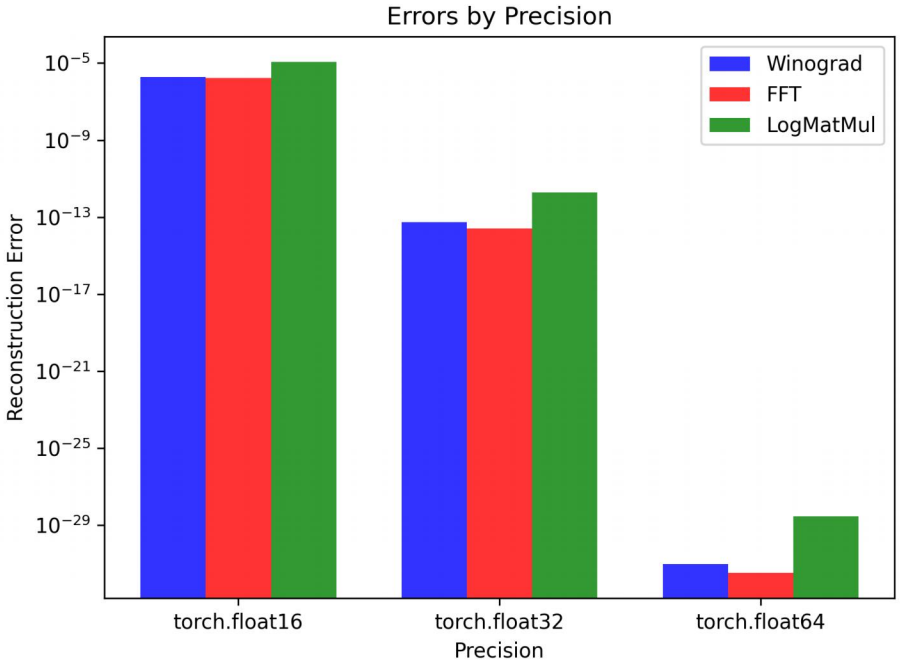

Figure 4: Precision vs Reconstruction Error comparison between Winograd Convolution, Fast Fourier Transform and Log Matrix Multiplication

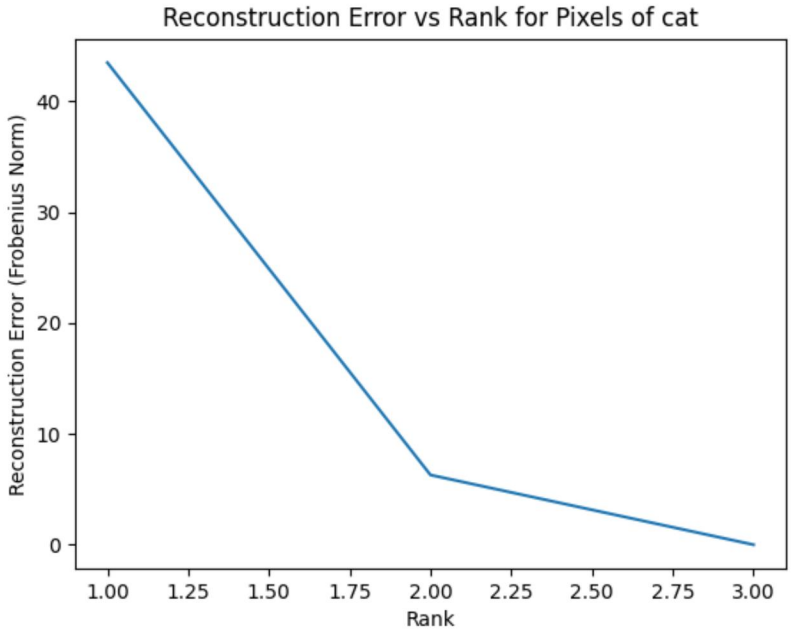

Figure 5: Reconstruction Error vs Rank for low rank approximation of the pixels of a cat

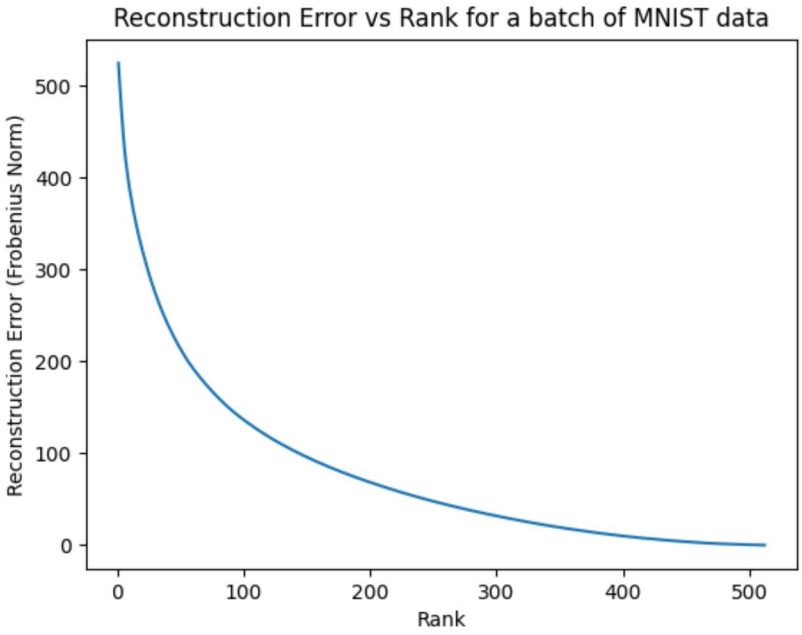

Figure 6: Reconstruction Error vs Rank for low rank approximation of a batch of MNIST digits

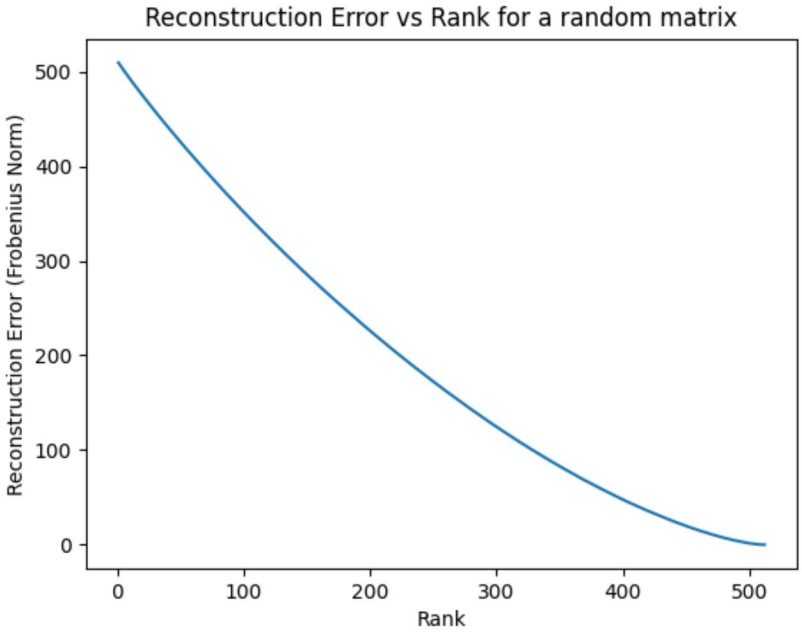

Figure 7: Reconstruction Error vs Rank for low rank approximation of a random matrix

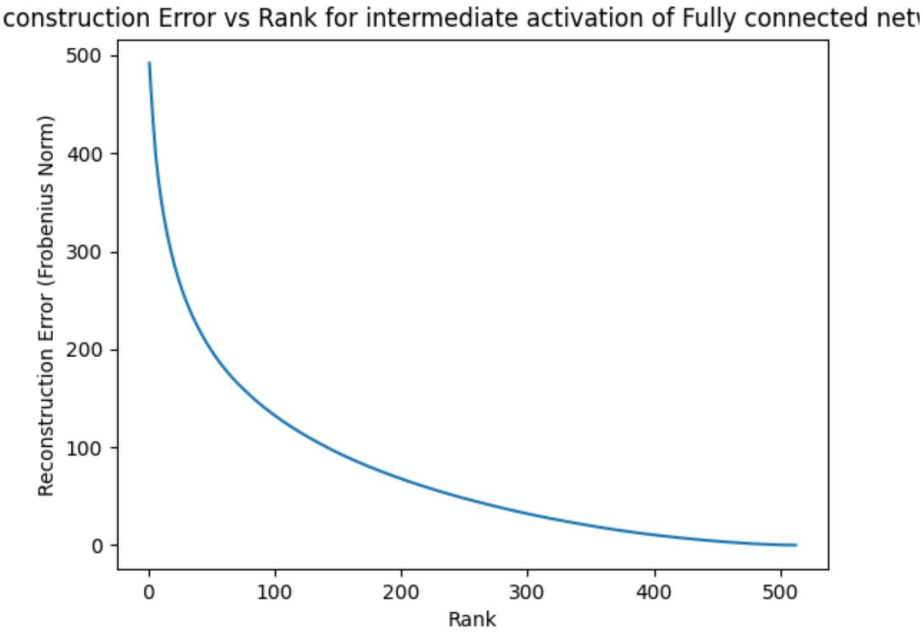

Figure 8: Reconstruction Error vs Rank for low rank approximation of an intermediate activation of a fully connected network

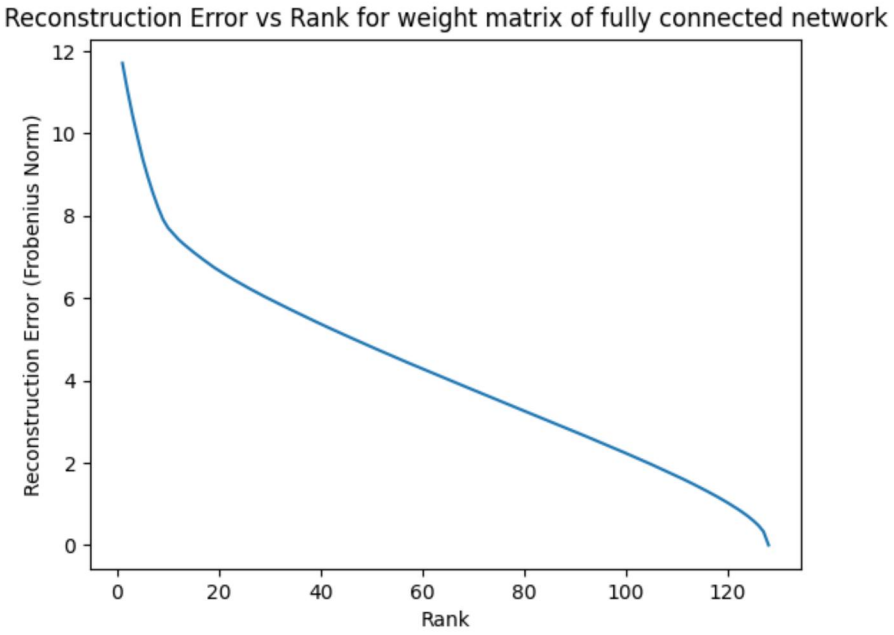

Figure 9: Reconstruction Error vs Rank for low rank approximation of the weight matrix of a linear layer of a fully connected network

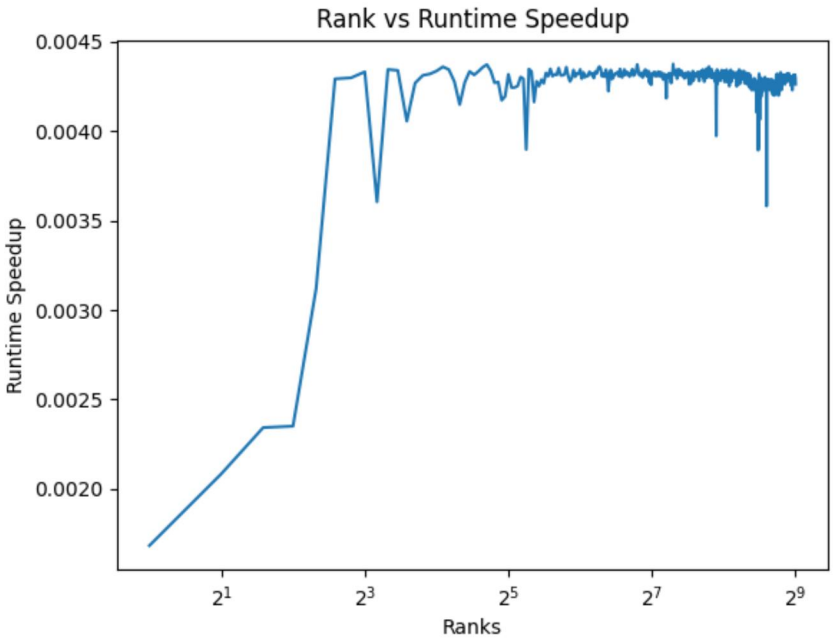

Figure 10: Rank vs Runtime Speedup for matrix multiplication

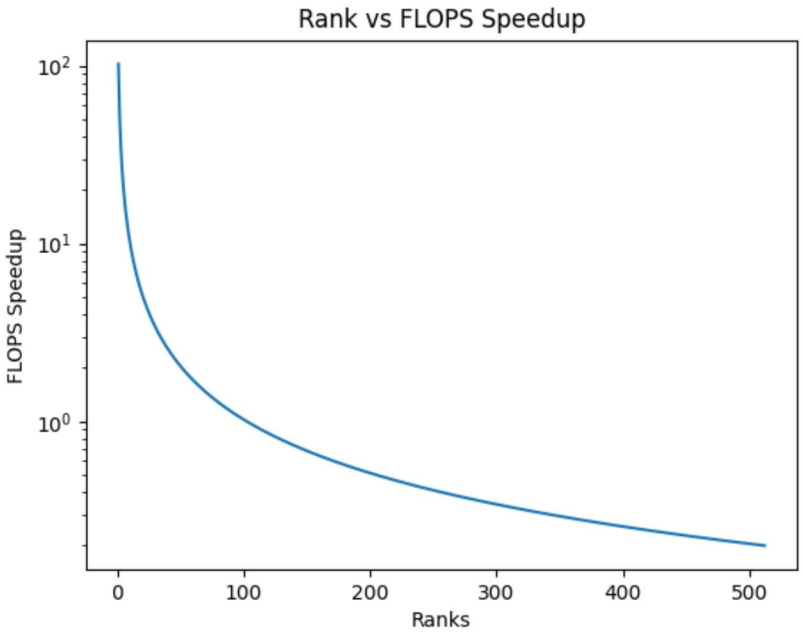

Figure 11: Rank vs FLOPs Speedup for matrix multiplication

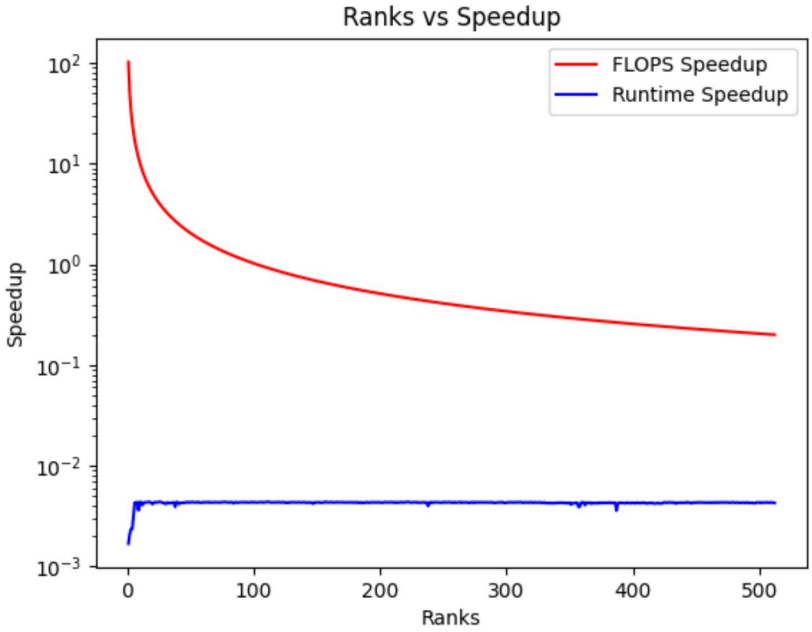

Figure 12: Rank vs Runtime and FLOPs Speedup for matrix multiplication

chosen from **8-128** with a separation of **32** between the different ranks, while for the second fully connected layer, ranks were chosen from **1-33** with a separation of **15** between the different ranks. After the model had finished training, the low rank approximation was carried out and the compression ratios, accuracies and runtimes for each low rank approximated model were collected. The plots for Compression Ratio vs Accuracy and Compression Ratio vs Runtime can be observed in **Figures 13-15**. It was observed that the accuracy uniformly reduces and reaches below 90% as the compression ratios increase and reach values greater than 7. However, no consistent trend can be observed in the runtime of the models, only a sharp increase and decrease can be observed for various compression ratios. The lowest runtime of **2.39s** was observed for a compression ratio of **2.86**.

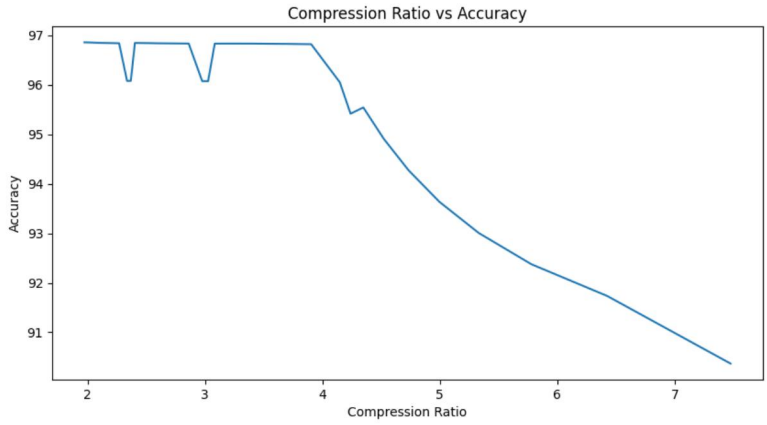

Figure 13: Compression Ratio vs Accuracy for the model trained on MNIST

4 Extra Credit Attempt

For the extra credit a new implementation for Log Matrix Multiplication was tried which aims to approximate the log addition step of the LogMatMul with the help of a look-up table.

$log(a * b) = log(a) + log(b) = log(a * (1 + \frac{b}{a})) = log(b * (1 + \frac{a}{b})) = log(a) + log(1 + \frac{b}{a}) = log(b) + log(1 + \frac{a}{b})$

The lookup table stores the values of $log(1 + \frac{b}{a})$ or $log(1 + \frac{a}{b})$, with the denominator being larger. The values are stored in negative powers of 2, given the size of the lookup table it can store powers of 2 from -1 to -size. Then the absolute difference between the logs of the matrices A and B is calculated and the difference is scaled according to the size of the lookup table in order to

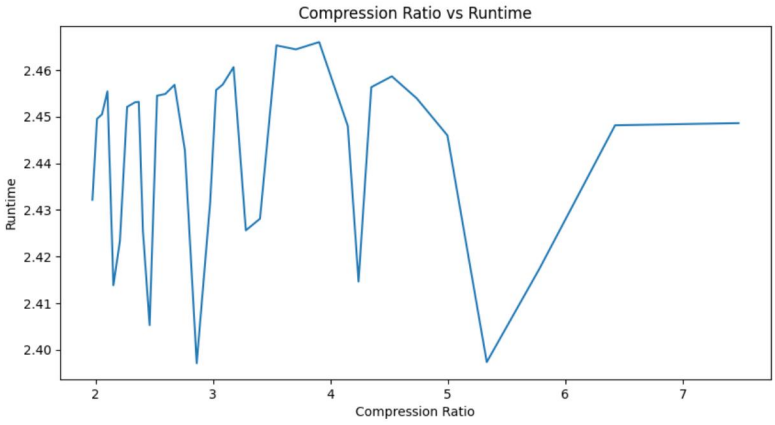

Figure 14: Compression Ratio vs Runtime for the model trained on MNIST

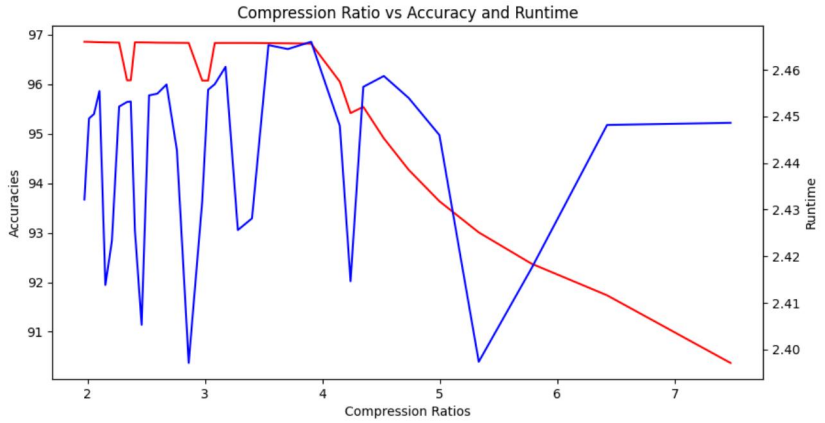

Figure 15: Compression Ratio vs Accuracy and Runtime for the model trained on MNIST

estimate the index for the value in the lookup table which corresponds to the value $log(1 + \frac{b}{a})$ or $log(1 + \frac{a}{b})$. The estimated value extracted from the lookup table is then added to $\mathbf{max}(\mathbf{log}(\mathbf{A}), \mathbf{log}(\mathbf{B}))$

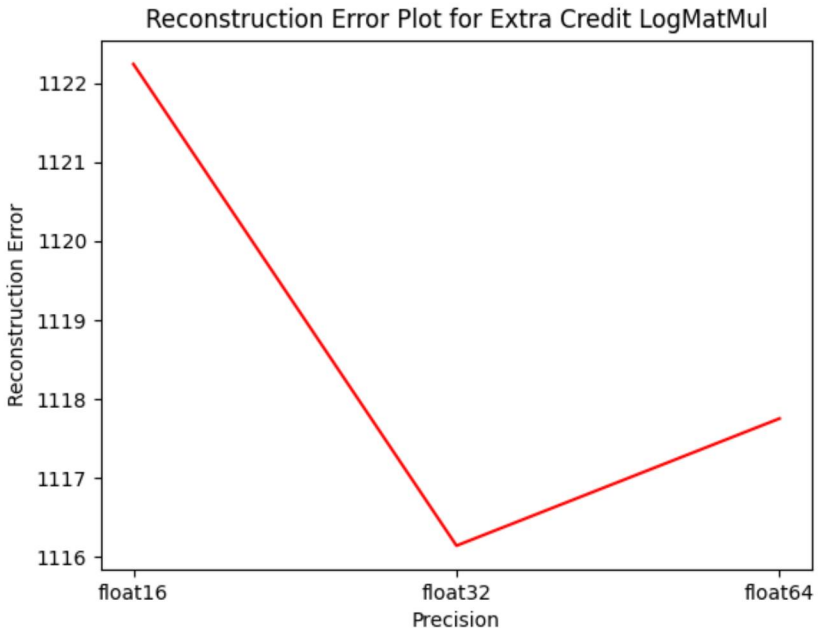

Figure 16: Precision vs Reconstruction Error for Approximated Log Addition

1 Functionality **10 / 10**

✓ **+ 5 pts** *Conv Test all passed*

+ 4 pts Conv Test passed > 60/84

+ 3 pts Conv Test passed > 40/84

+ 2 pts Conv Test passed > 20 / 84

+ 1 pts Conv test failed

✓ **+ 5 pts** *Matmul test all passed*

+ 1 pts Matmul test failed

ML Hardware Assignment 4

Naman Makkar (nbm49)

May 2023

1 Experiments on Numerical Precision

The plots for Numerical Precision vs Reconstruction Error were plotted for Winograd Convolution, Fast Fourier Transform and Log Matrix Multiplication. The reconstruction error utilized here was the L2 loss function. The values of numerical precision used were - **Float 16, Float 32, Float 64**. The maximum reconstruction error was observed in **Float 16 precision** while the lowest reconstruction error was observed for Float 64. It was also observed that Log Matrix Multiplication had a marginally higher reconstruction error than the other 2 methods. Additionally, the reconstruction error reduced with increasing precision. The plots can be observed in **Figures 1-4**.

2 Experiments on SVD Reconstruction Error and Speedup

For the first set of experiments with SVD, the reconstruction error of matrices was plotted with the ranks of the low rank approximations of the matrix. The results of these experiments can be observed in **Figures 5-9** The second set of experiments, the runtime speedup and the FLOPs speedup of SVD was calculated on a matrix multiplication operation between the weights of a linear layer and the activation of the previous layer. The plots for the second set of experiments can be observed in **Figures 10-12** where the rank vs runtime speedup and rank vs FLOPs speedup have been plotted for the matrix multiplication. It was observed that the lowest rank values achieved the greatest FLOPs speedup whereas the greatest runtime speedup was observed for ranks between $\mathbf{2^4}$ and $\mathbf{2^5}$.

3 MNIST Training

The given fully connected neural network was trained on the MNIST dataset and low rank approximation was carried out on the model with the help of SVD. SVD was utilised to carry out low rank approximation on the first and the second fully connected layers of the model. For the first fully connected layer ranks were

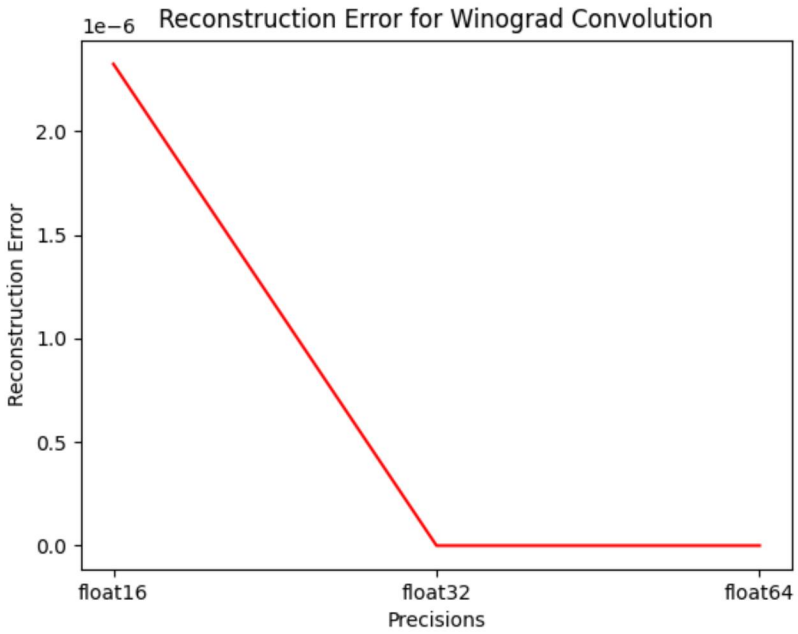

Figure 1: Precision vs Reconstruction Error for Winograd Convolution

Figure 2: Precision vs Reconstruction Error for Fast Fourier Transform

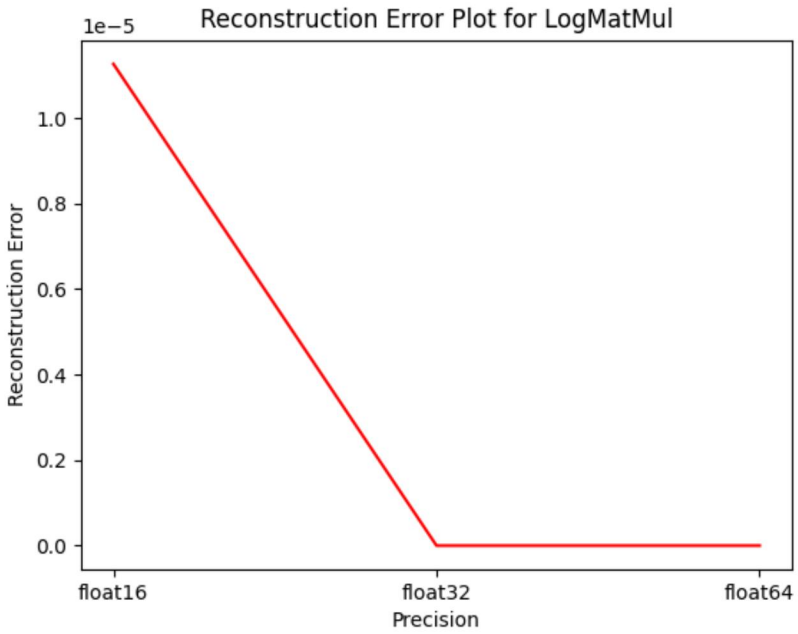

Figure 3: Precision vs Reconstruction Error for Log Matrix Multiplication

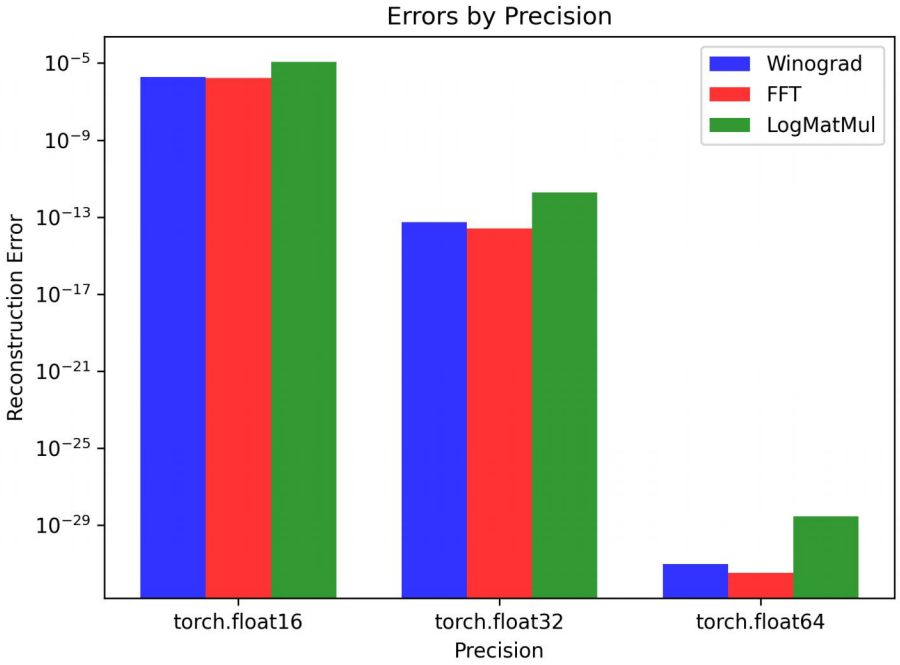

Figure 4: Precision vs Reconstruction Error comparison between Winograd Convolution, Fast Fourier Transform and Log Matrix Multiplication

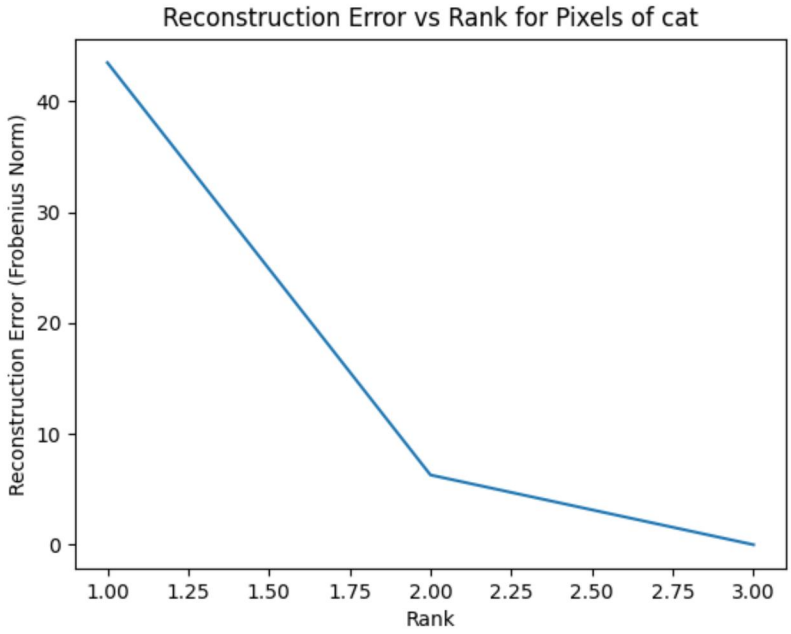

Figure 5: Reconstruction Error vs Rank for low rank approximation of the pixels of a cat

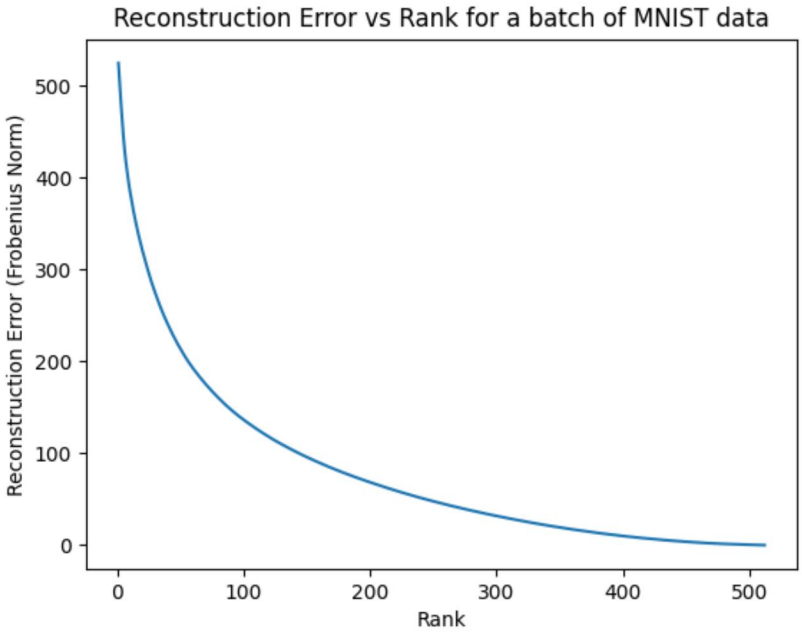

Figure 6: Reconstruction Error vs Rank for low rank approximation of a batch of MNIST digits

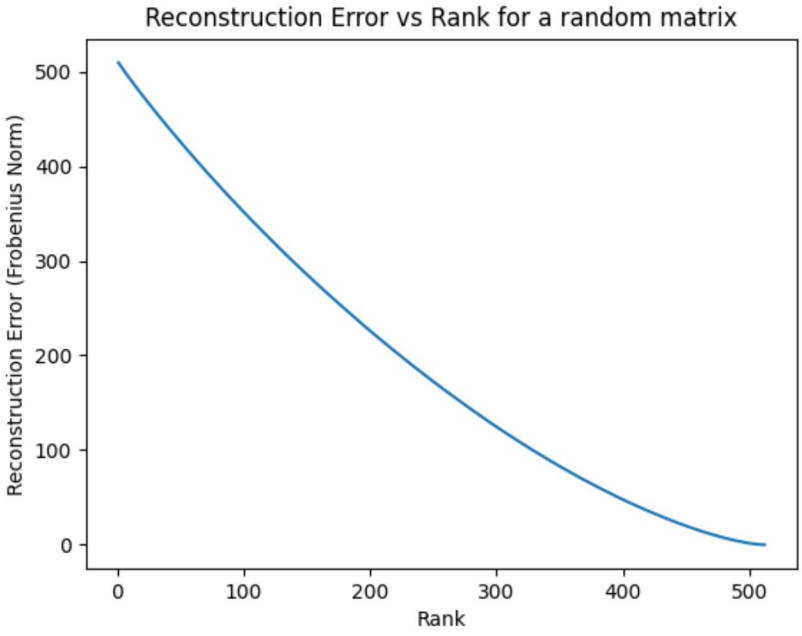

Figure 7: Reconstruction Error vs Rank for low rank approximation of a random matrix

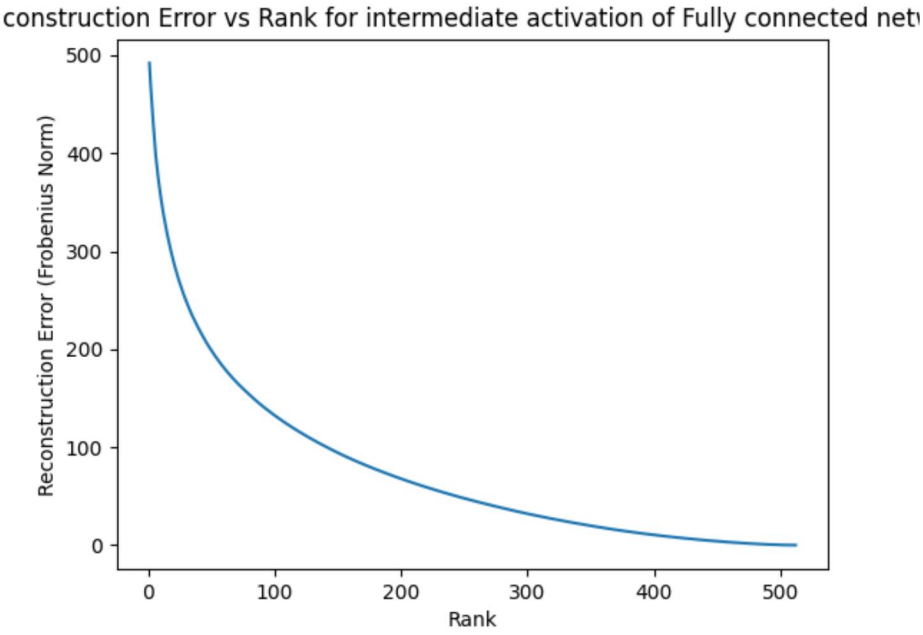

Figure 8: Reconstruction Error vs Rank for low rank approximation of an intermediate activation of a fully connected network

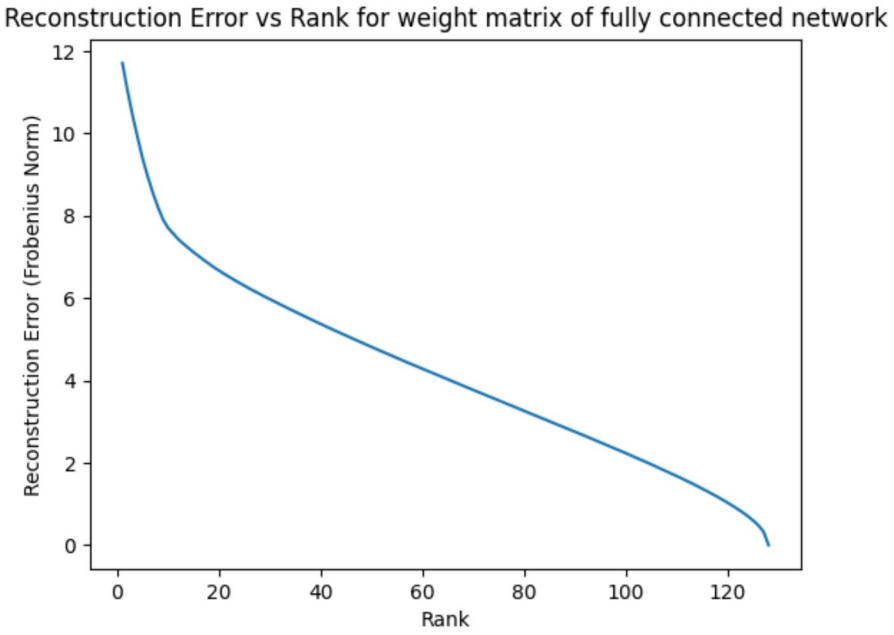

Figure 9: Reconstruction Error vs Rank for low rank approximation of the weight matrix of a linear layer of a fully connected network

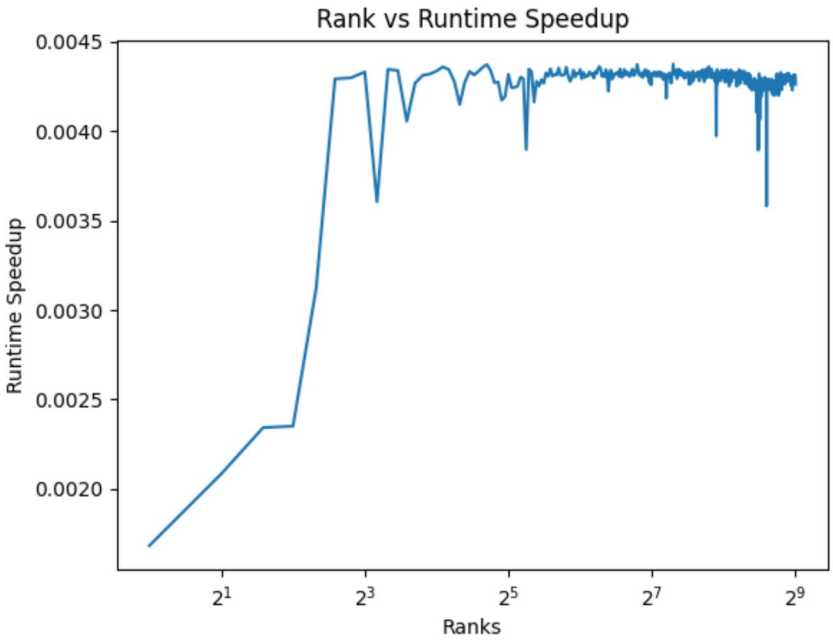

Figure 10: Rank vs Runtime Speedup for matrix multiplication

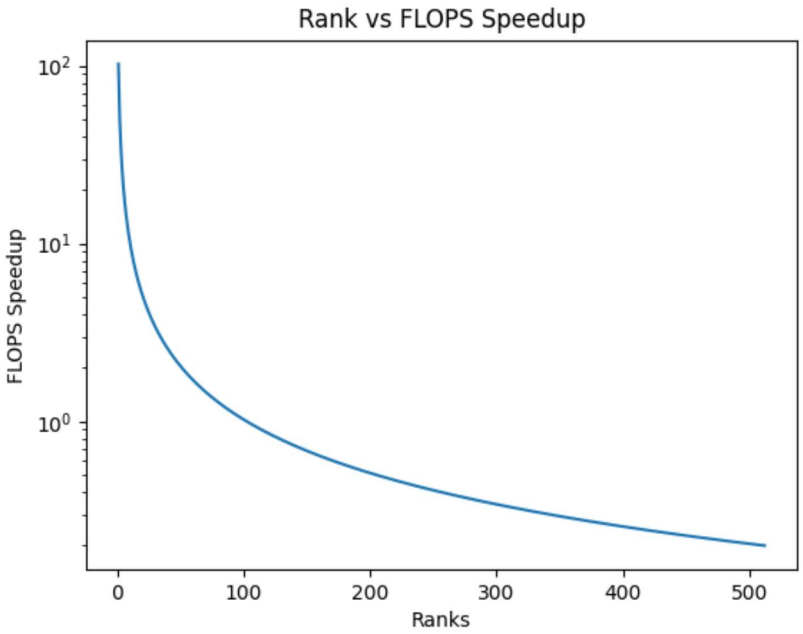

Figure 11: Rank vs FLOPs Speedup for matrix multiplication

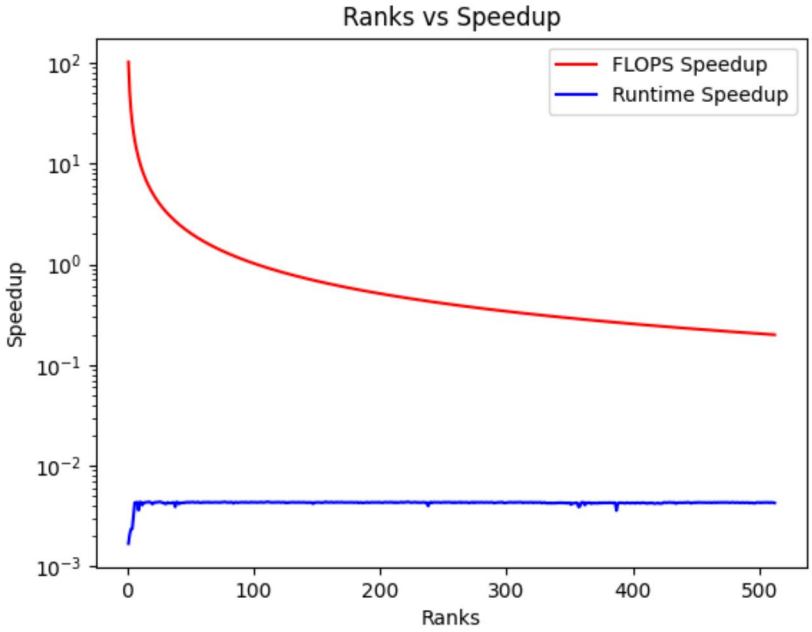

Figure 12: Rank vs Runtime and FLOPs Speedup for matrix multiplication

chosen from **8-128** with a separation of **32** between the different ranks, while for the second fully connected layer, ranks were chosen from **1-33** with a separation of **15** between the different ranks. After the model had finished training, the low rank approximation was carried out and the compression ratios, accuracies and runtimes for each low rank approximated model were collected. The plots for Compression Ratio vs Accuracy and Compression Ratio vs Runtime can be observed in **Figures 13-15**. It was observed that the accuracy uniformly reduces and reaches below 90% as the compression ratios increase and reach values greater than 7. However, no consistent trend can be observed in the runtime of the models, only a sharp increase and decrease can be observed for various compression ratios. The lowest runtime of **2.39s** was observed for a compression ratio of **2.86**.

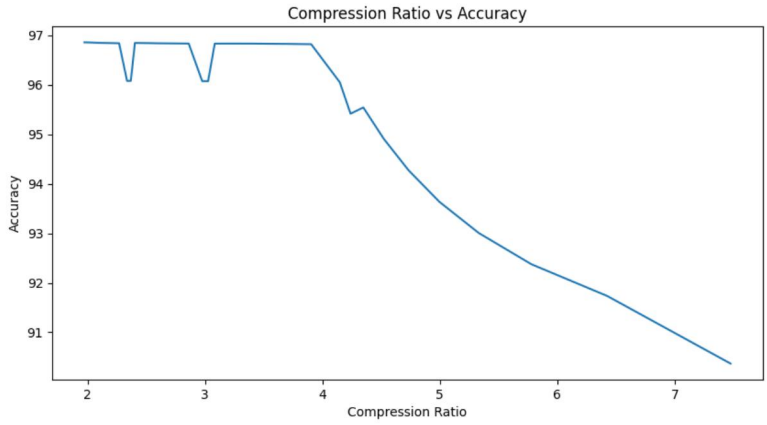

Figure 13: Compression Ratio vs Accuracy for the model trained on MNIST

4 Extra Credit Attempt

For the extra credit a new implementation for Log Matrix Multiplication was tried which aims to approximate the log addition step of the LogMatMul with the help of a look-up table.

$log(a * b) = log(a) + log(b) = log(a * (1 + \frac{b}{a})) = log(b * (1 + \frac{a}{b})) = log(a) + log(1 + \frac{b}{a}) = log(b) + log(1 + \frac{a}{b})$

The lookup table stores the values of $log(1 + \frac{b}{a})$ or $log(1 + \frac{a}{b})$, with the denominator being larger. The values are stored in negative powers of 2, given the size of the lookup table it can store powers of 2 from -1 to -size. Then the absolute difference between the logs of the matrices A and B is calculated and the difference is scaled according to the size of the lookup table in order to

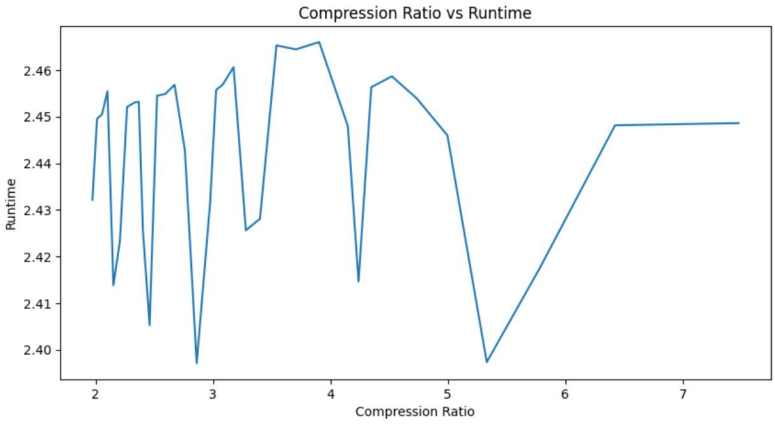

Figure 14: Compression Ratio vs Runtime for the model trained on MNIST

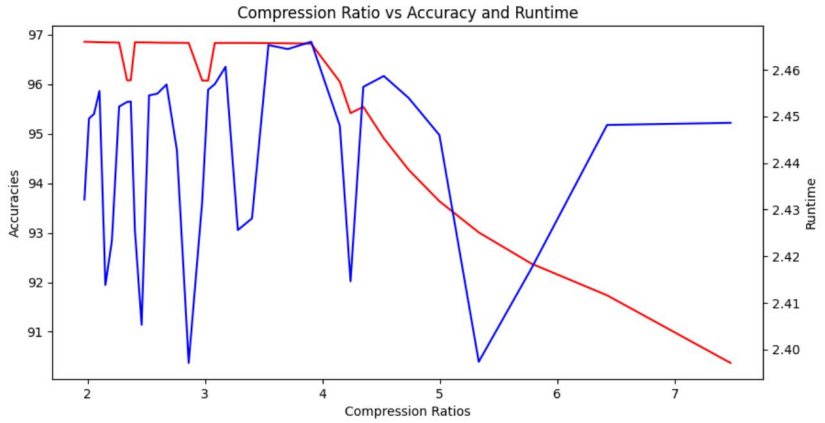

Figure 15: Compression Ratio vs Accuracy and Runtime for the model trained on MNIST

estimate the index for the value in the lookup table which corresponds to the value $log(1 + \frac{b}{a})$ or $log(1 + \frac{a}{b})$. The estimated value extracted from the lookup table is then added to $\mathbf{max}(\mathbf{log}(\mathbf{A}), \mathbf{log}(\mathbf{B}))$

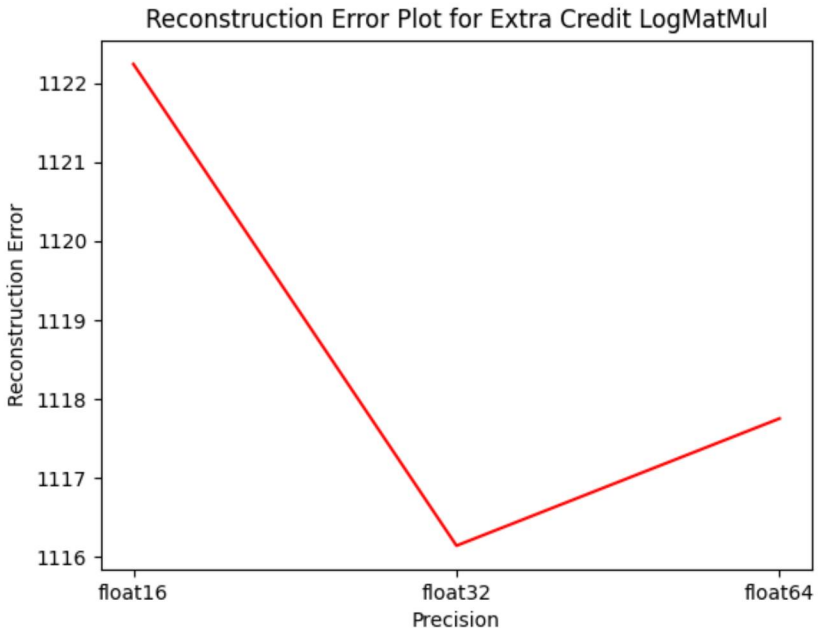

Figure 16: Precision vs Reconstruction Error for Approximated Log Addition

2 Report **5 / 6**

+ 2 pts Q3 (quantization) Complete

✓ **+ 1 pts** *Q3 (quantization) quality*

✓ **+ 2 pts** *Q4 (SVD) complete*

+ 1 pts Q4 (SVD) quality

✓ **+ 2 pts** *Q5 (SVD + NN) complete*

+ 1 pts Q5 (SVD + NN) Quality

+ 0 pts Not complete

chosen from **8-128** with a separation of **32** between the different ranks, while for the second fully connected layer, ranks were chosen from **1-33** with a separation of **15** between the different ranks. After the model had finished training, the low rank approximation was carried out and the compression ratios, accuracies and runtimes for each low rank approximated model were collected. The plots for Compression Ratio vs Accuracy and Compression Ratio vs Runtime can be observed in **Figures 13-15**. It was observed that the accuracy uniformly reduces and reaches below 90% as the compression ratios increase and reach values greater than 7. However, no consistent trend can be observed in the runtime of the models, only a sharp increase and decrease can be observed for various compression ratios. The lowest runtime of **2.39s** was observed for a compression ratio of **2.86**.

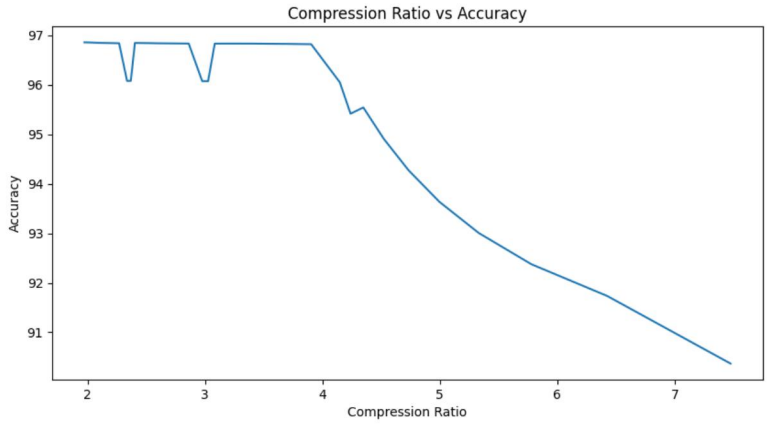

Figure 13: Compression Ratio vs Accuracy for the model trained on MNIST

4 Extra Credit Attempt

For the extra credit a new implementation for Log Matrix Multiplication was tried which aims to approximate the log addition step of the LogMatMul with the help of a look-up table.

$log(a * b) = log(a) + log(b) = log(a * (1 + \frac{b}{a})) = log(b * (1 + \frac{a}{b})) = log(a) + log(1 + \frac{b}{a}) = log(b) + log(1 + \frac{a}{b})$

The lookup table stores the values of $log(1 + \frac{b}{a})$ or $log(1 + \frac{a}{b})$, with the denominator being larger. The values are stored in negative powers of 2, given the size of the lookup table it can store powers of 2 from -1 to -size. Then the absolute difference between the logs of the matrices A and B is calculated and the difference is scaled according to the size of the lookup table in order to

estimate the index for the value in the lookup table which corresponds to the value $log(1 + \frac{b}{a})$ or $log(1 + \frac{a}{b})$. The estimated value extracted from the lookup table is then added to $\mathbf{max}(\mathbf{log}(\mathbf{A}), \mathbf{log}(\mathbf{B}))$

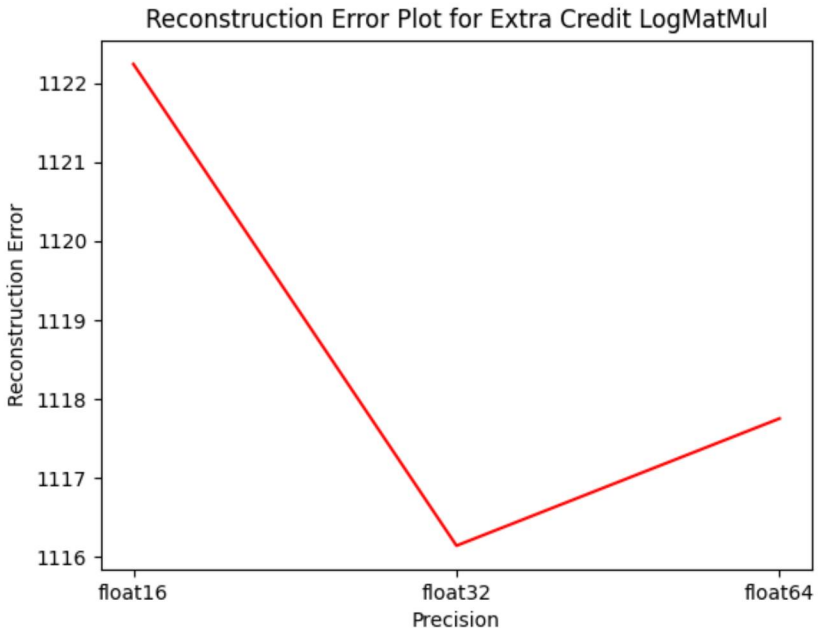

Figure 16: Precision vs Reconstruction Error for Approximated Log Addition

3 Logmul Extra Credits **1 / 1**

✓ **+ 1 pts** *Attempt logmult extra credits.*

+ 0 pts Click here to replace this description.