COEN 240 Machine Learning Project

Li Yuan Zhi W1588795
Chun-Hao Liu W1588390
Pok Lai Alexander Law W1589101

Abstract:

This project consists of three sections. In the first section, a convolutional neural network (CNN) is built using the cross-error entropy function. Next, several images are selected from the Fashion MNIST dataset. These images are trained and tested using a specified CNN. The performance of the CNN is evaluated. The accuracy rate of the CNN is 91%. In the second section, an image compression system is built using a regular neural network. The same images from the previous Fashion MNIST dataset are used. A specific neural network system containing compression and decompression is used to train images. The peak signal-to-noise ratio (PSNR) is calculated under various number of P nodes and the performance is evaluated using the mean-squared-error function (MSE). The results indicate as the number of nodes increase, the PSNR increases, and the error decreases. In the last section, a color compression video is created using four different methods with a combination of fully connected neural networks and convolutional neural networks. These are selected from three different datasets including Blowing Bubbles, Race Horses and Basketball Drills. The optimal method out of the four methods used is the convolutional neural network which involves up-sampling and many layers.

Task 1:

The goal of this section is to analyze the performance a specific convolutional neural network. The recognition accuracy rate of this network is 0.91 or 91%, which shows that the network mostly works, but has a lower accuracy rate than most networks used in the industry. The confusion matrix, which shows the distribution of how the convolutional neural network selected the labels versus the true labels is shown in Figure x below.

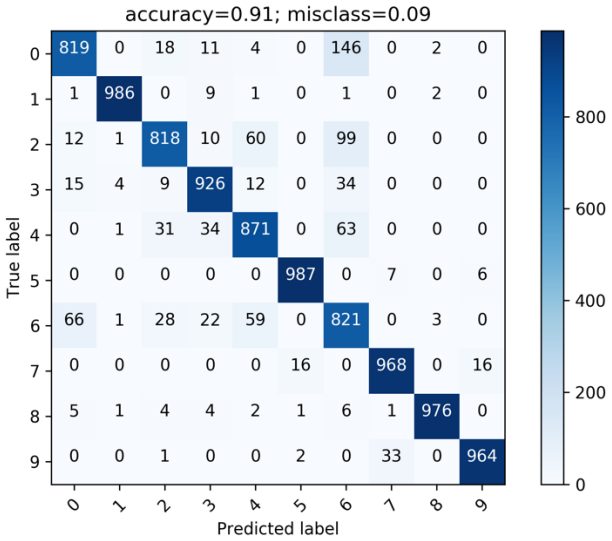

Figure 1: Confusion Matrix

Task 2:

The second task uses a regular neural network to perform compression and decompression using the trained model. We use three different P values (number of nodes) to train the network. The PSNR and loss after 10 epochs are calculated. The PSNR value for the following P values are shown in the table below:

P	PSNR	Loss after 10 Epochs
10	19.47	0.01360
50	22.44	0.00738
200	25.56	0.00364

As the number of nodes increase, the loss decreases and the PSNR increases. This is because with more nodes, we have more data to train the network and hence be more accurate. The following figure contains the original 10 images, the decompressed images with P=10, P=50 and P=200.

Figure 2: Top: Original Images 2nd, 3rd and 4th row: P=10, P=50, P=200

As the P values increase, the visual quality of the decompressed images increase. This is true for all 10 images. There is more depth and detail to the quality of the images as P increases. Within the same P value, the pictures that have higher variation of color distributions with its neighboring pixels are more difficult to decompress. This is because the compression process groups neighboring cells together to a near uniform distribution, and after decompression the variation is lost.

Task 3:

I Introduction:

A color video compression system is built using four different methods which are implemented using combinations of the fully connected neural network and the convolutional neural network. These methods aim to achieve better reconstruction quality (higher PSNR values and visual qualities). The balance between factors such as reconstruction quality, computational complexity, and model size is studied to find an optimal solution.

II Proposed methods

a. Network architectures

Fully Connected Neural Network:

The first network architecture used will be a Fully Connected Neural Network. Figure 3-7 below shows the reconstruction process for various compression ratios. The variables n, c, h and w represent the number of images, number of channels, height and width respectively.

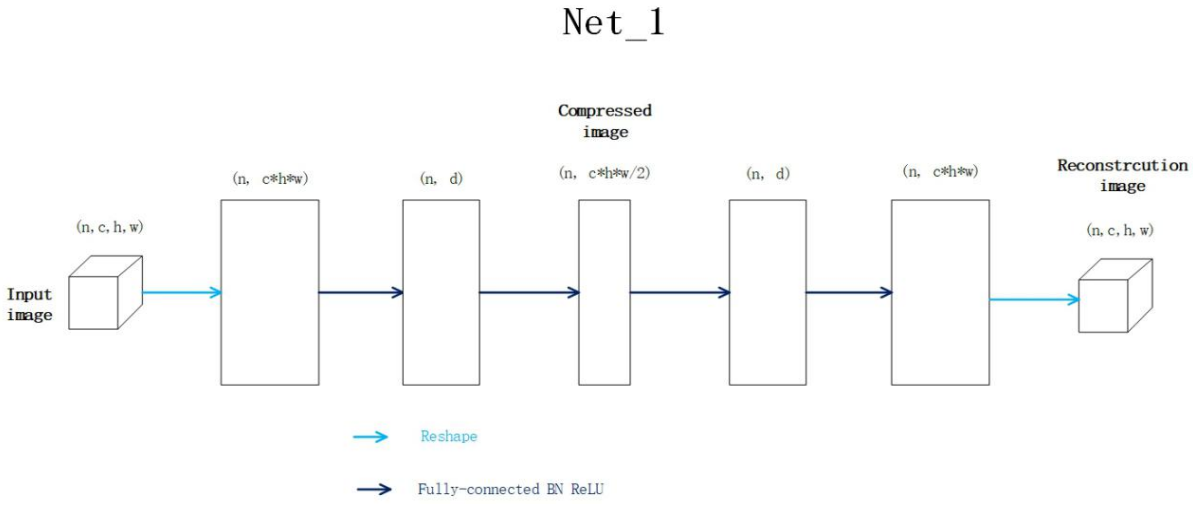

Figure 3: Compression ratio of 1/2

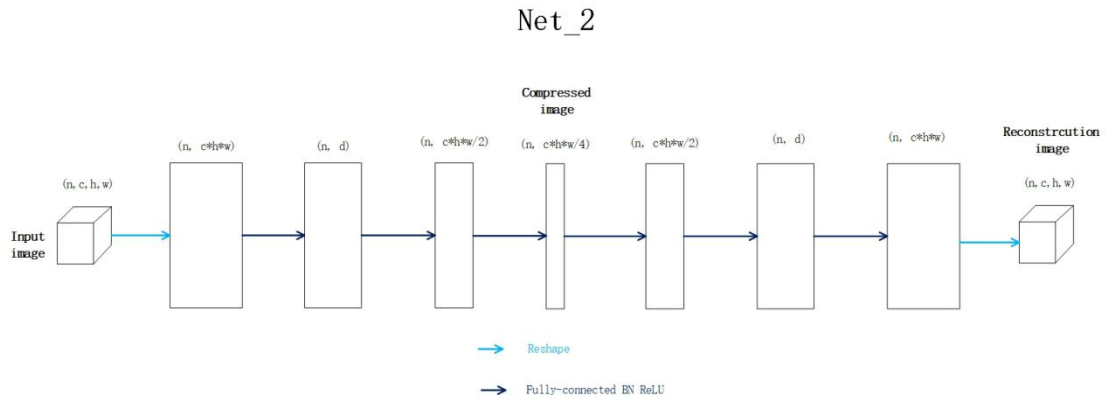

Figure 4: Compression ratio of 1/4

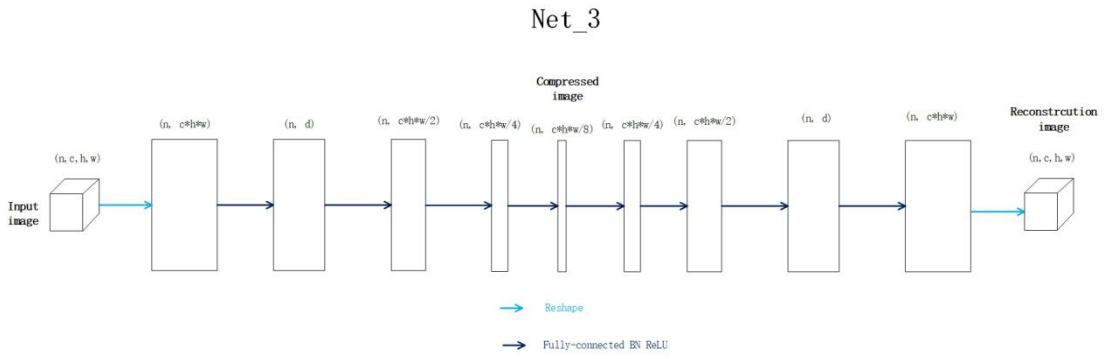

Figure 5: Compression ratio of 1/8

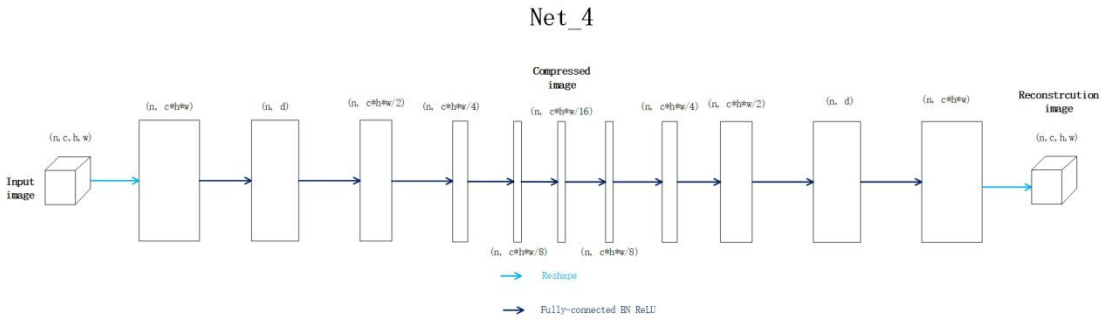

Figure 6: Compression ratio of 1/16

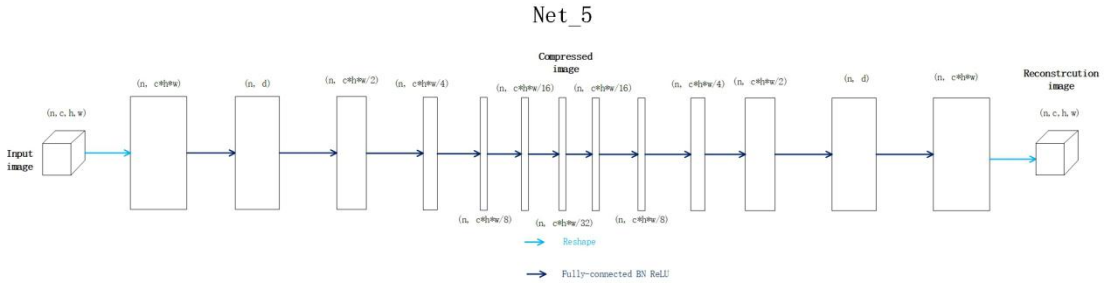

Figure 7: Compression ratio of 1/32

In this process, the input image is first flattened from a three-dimensional matrix to a one-dimensional matrix. As the matrix size of 3x416x200 is too large, we divide the image into 32 smaller images of size 3x41x200/32=9360 (number of total pixels for each color divided by the number of images). After using a RELU function, we convert this to an image size to an intermediate size of 5000 (size d). This is then compressed into half of the original size (9360/2=4680) for a compression ratio of ½ and followed with a RELU function. For a smaller compression ratio of ¼, 1/8, 1/16 and 1/32, we repeat the process of

compressing the image into half of its current size, followed by another RELU function. After shrinking the image to select its most relevant features, we can decompress the image by a multiple of 2 until the image reaches its original size. Finally, we can merge all 32 smaller images, and then perform the sigmoid classifier to produce the reconstruction image. This model uses SGD optimizer with a learning rate of 0.1, batch-size of 1. The model is trained using 10 epochs.

Convolutional Neural Network (CNN)

Figures 8 to 12 show the number of layers between each step of the convolutional neural network, as well at the reconstruction process.

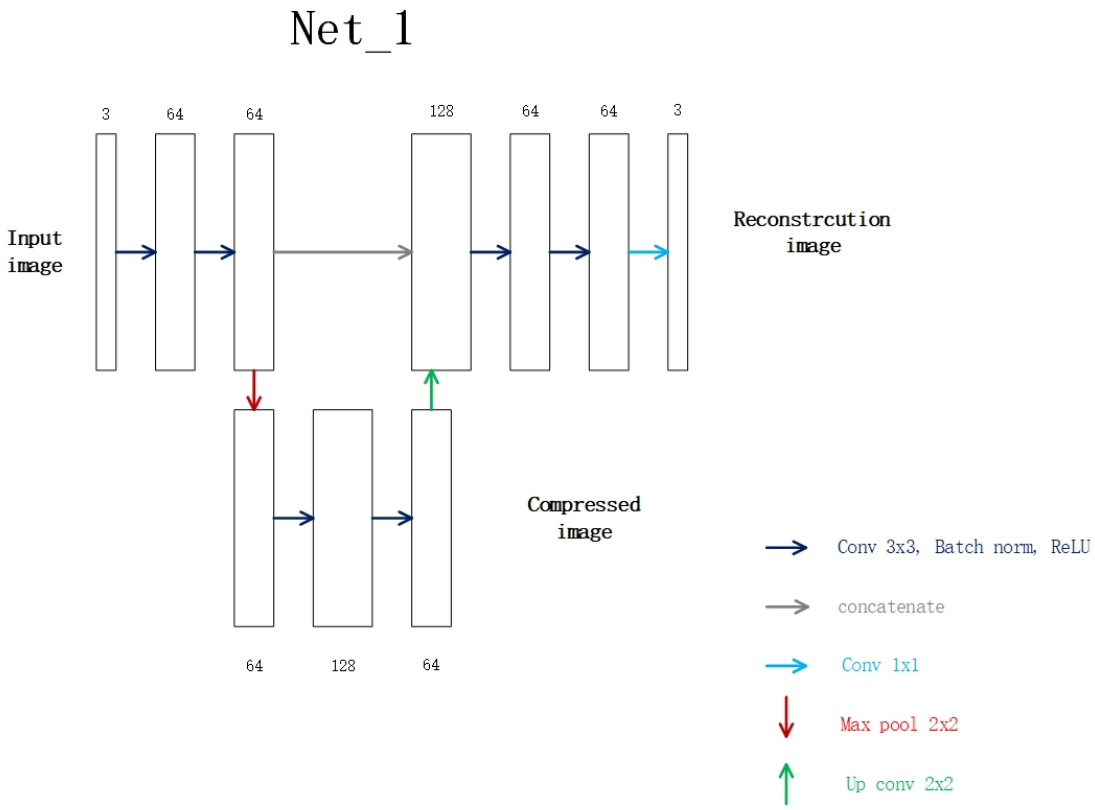

Figure 8: FNN: Compression ratio of 1/2

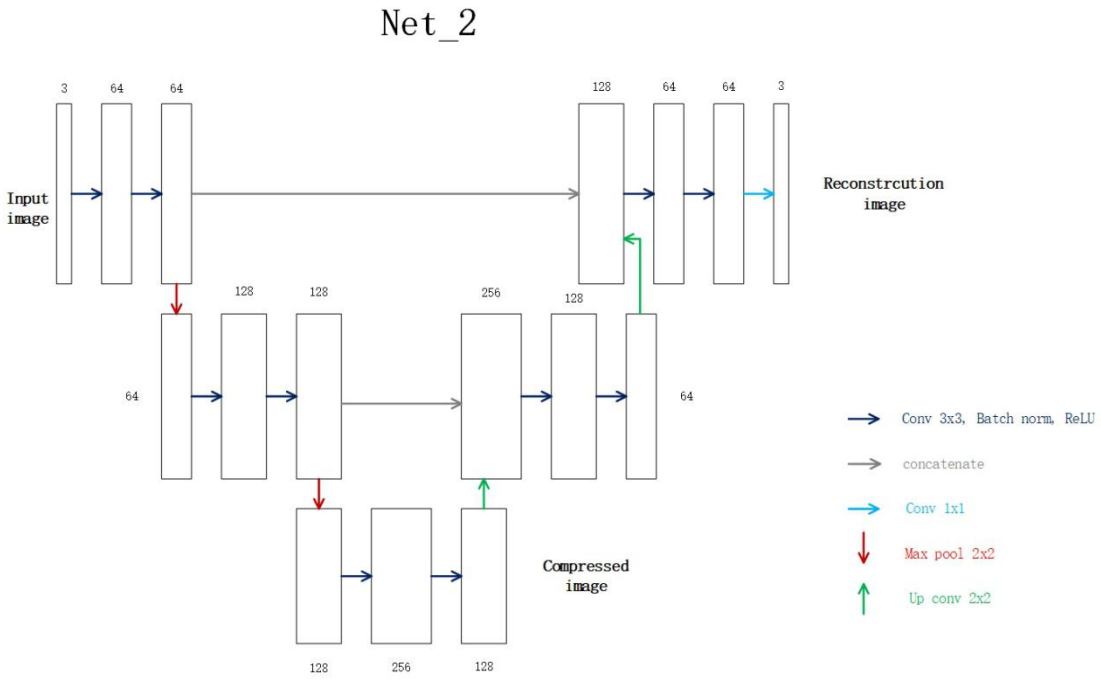

Figure 9: FNN Compression ratio of 1/4

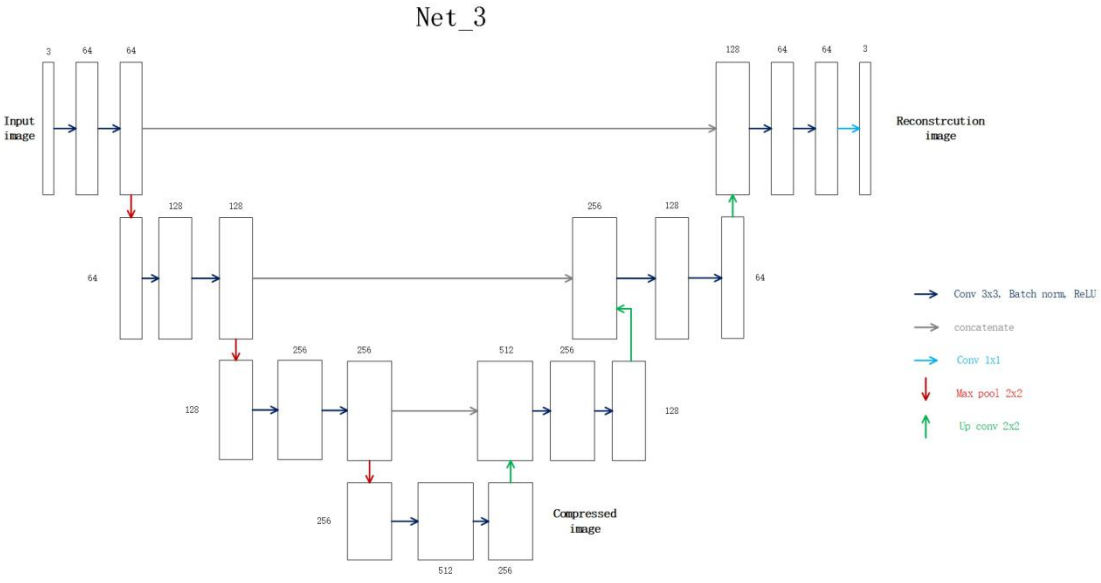

Figure 10: FNN compression ratio of 1/8

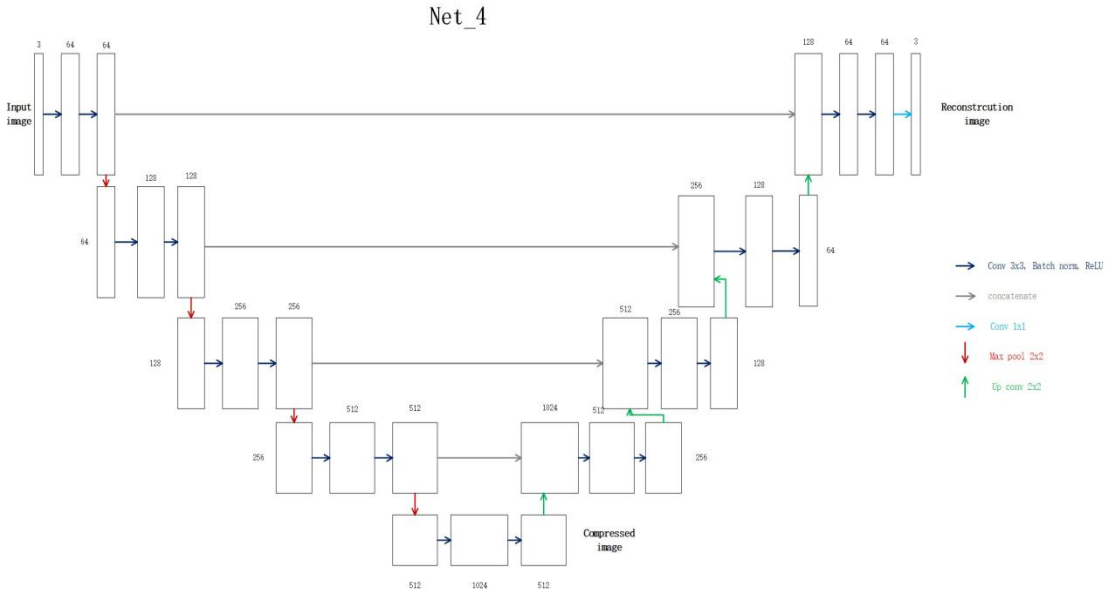

Figure 11: FNN compression ratio of 1/16

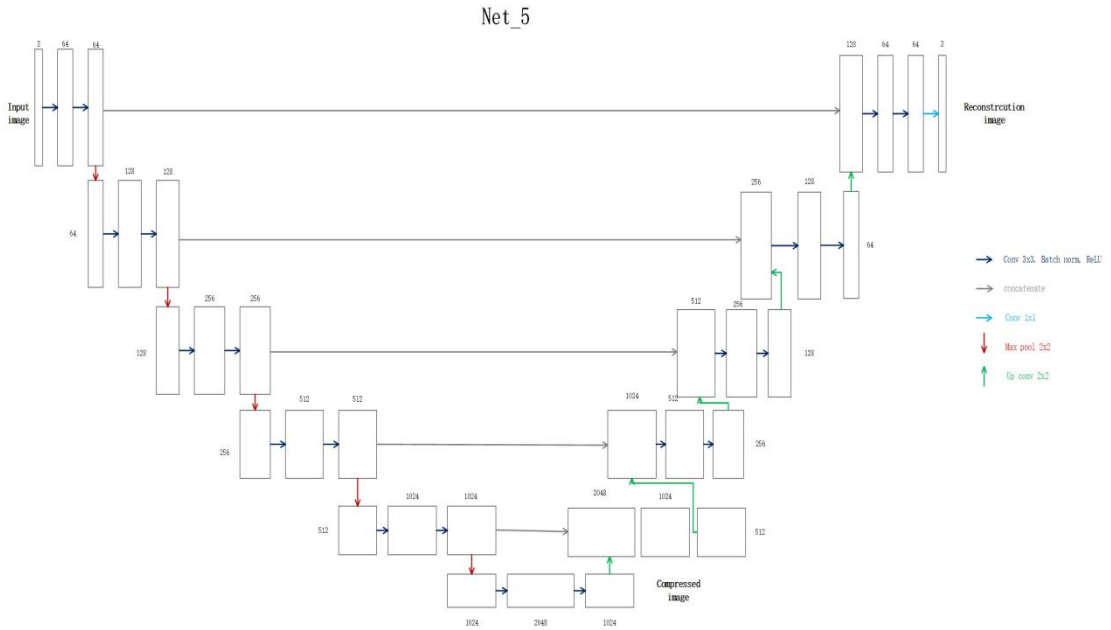

Figure 12: Compression ratio of 1/32

The data is first sorted using a bi-linear kernel function. The original input has 3 channels (the colors red, green and blue). This is filtered using 64 3x3 filters (including 1 layer of padding to keep the image at the

same size), and followed by a batch normalization process and a RELU function. This is followed by another round of the filtering, batch normalization and a RELU function. This current image is copied and saved for later concatenation, then compressed by using by using 2x2 maximum pooling. The number of times the pooling is done affects the compression ratio. The compression ratio is ½ after pooling once; and is multiplied by half after each pooling layer. After each new pooling layer, a similar process where there are two processes of filtering, batch normalization and a RELU function. The only difference from the first pooling is that the number of filters double. For decompression, we start from the bottom of the figure. The network first makes a 2x2 upsampling to features and then combines the final image of the previous pooling layer. We then shrink the size the features by a series of 3x3 convolutions, batch normalization and RELU function twice. This process is repeated until it is back to the original size. This convolutional neural network uses the adam optimizer with the following parameters: learning rate = 0.01, batch-size = 3 and five epochs for each model.

Convolutional Neural Network combined with Fully Connected Neural Network:
The third network architecture involves uses the convolutional neural network, then is combined with the fully connected neural network near the end. Similar to the methods above, the steps are summarized for the method using ½ compression ratio in table 1 below. Compression ratios of ¼, 1/8, 1/16, 1/32 are similar to ½, but include more fully connected layers.

Table 1: CNN+FNN

Layer	Procedure:
First Layer	5x5 convolution, Batch Normalization, RELU, 2x2 maximum pooling
Second Layer	5x5 convolution, Batch Normalization, RELU, 2x2 maximum pooling
Third Layer	3x3 convolution, Batch Normalization, RELU
Fourth Layer	3x3 convolution, Batch Normalization, RELU
Fifth Layer	6144x4680 Fully connected neural network, Batch Normalization, RELU
Sixth Layer	4680x6144 Fully connected neural network, Batch Normalization, RELU
Seventh Layer	6144x9360 Fully connected neural network, Sigmoid function

The fourth network architecture uses another convolution neural network. Take 1/2 compression ratio as an example. The steps are summarized for the method using ½ compression ratio in table 1 below. Compression ratios of ¼, 1/8, 1/16, 1/32 are similar to ½, but include more pooling and up sampling.

Table 2: CNN alternative method

Layer	Procedure:
First Layer	3x3 convolution, Batch Normalization, RELU
Second Layer	3x3 convolution, Batch Normalization, RELU
Third Layer	1x2 maximum pooling
Fourth Layer	1x1 convolution, RELU
Fifth Layer	1x2 deconvolution, Batch Normalization, RELU
Sixth Layer	3x3 convolution, Batch Normalization, RELU
Seventh Layer	1x1 convolution, RELU

III Experimental Studies

Dataset description:

For the fully connected neural network sample, we used the data image set Race Horses. This dataset contains 300 images, 250 of which we will use as our training set, and the rest for the test set. For the convolutional neural network, we used the image set Blowing Bubbles. This dataset contains 500 images, 450 of which we randomly selected as the training set, and the rest as test set. Each image has the size of 416x240 pixels, which can be divided by multiples of 2 up till 16. For a compression ratio of 1/32, we will have to resize the image to 416x224.

Quantitative Evaluation:

Fully Connected Neural Network:

Figures 13-17 show the average PSNR curves on training set and test set for under compression ratios of ½, ¼, 1/8, 1/16 and 1/32.

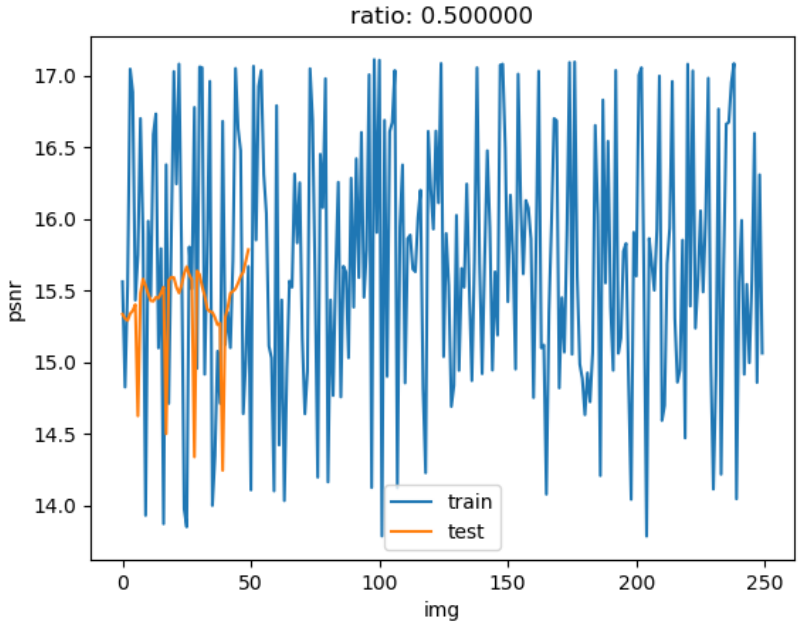

Figure 13: PSNR under a compression ratio of 1/2

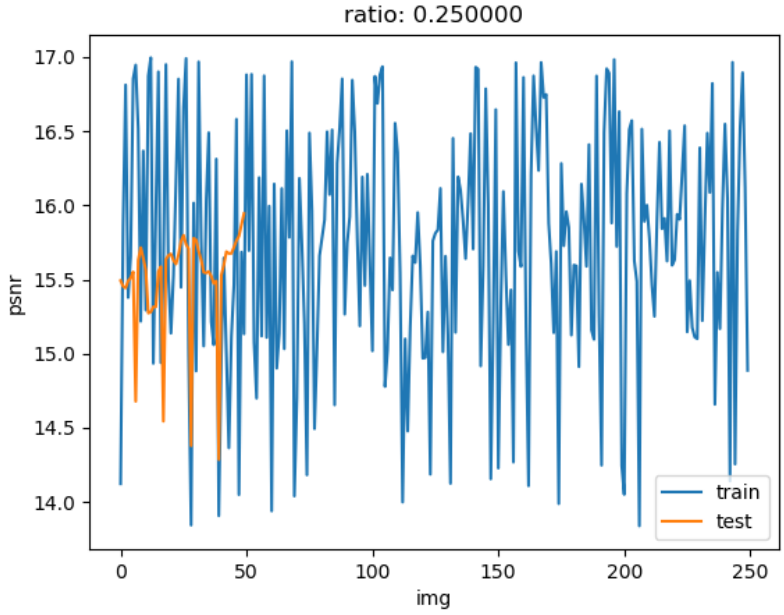

Figure 14: PSNR under a compression ratio of 1/4

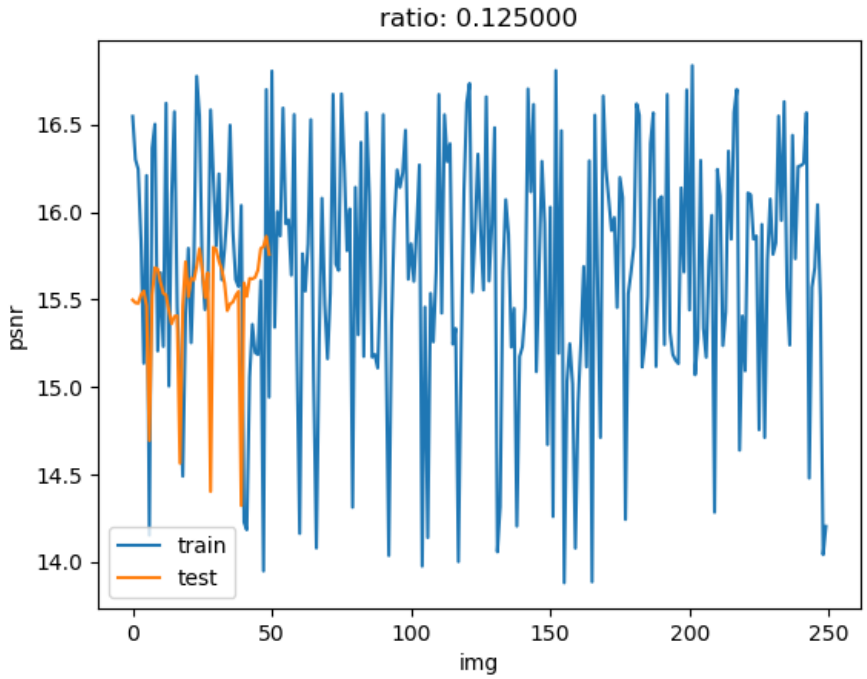

Figure 15: PSNR under a compression ratio of 1/8

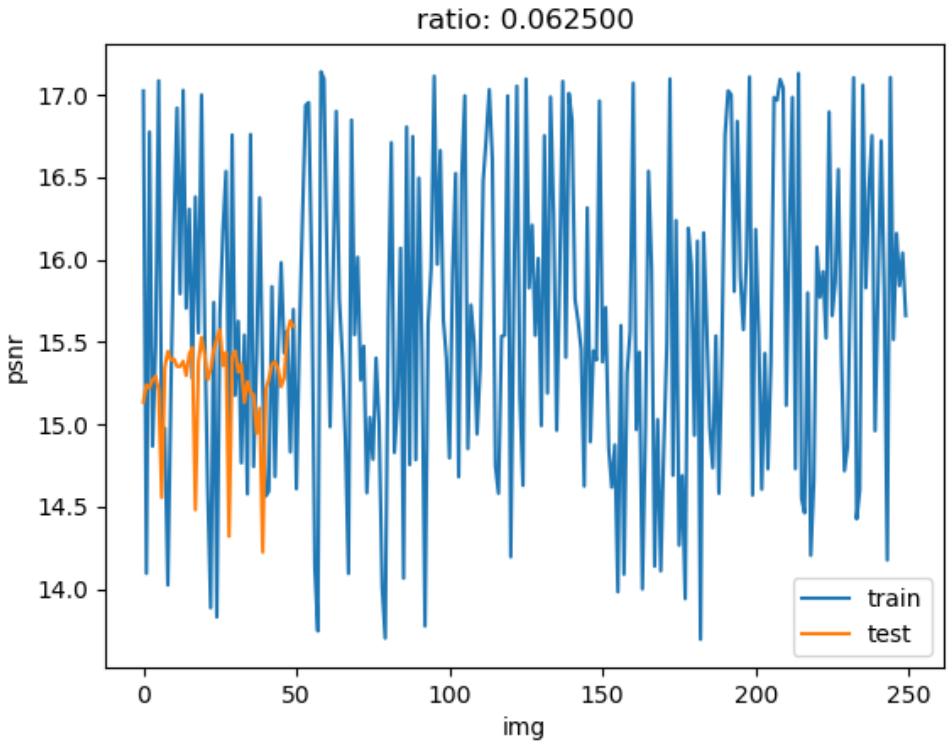

Figure 16: PSNR under a compression ratio of 1/16

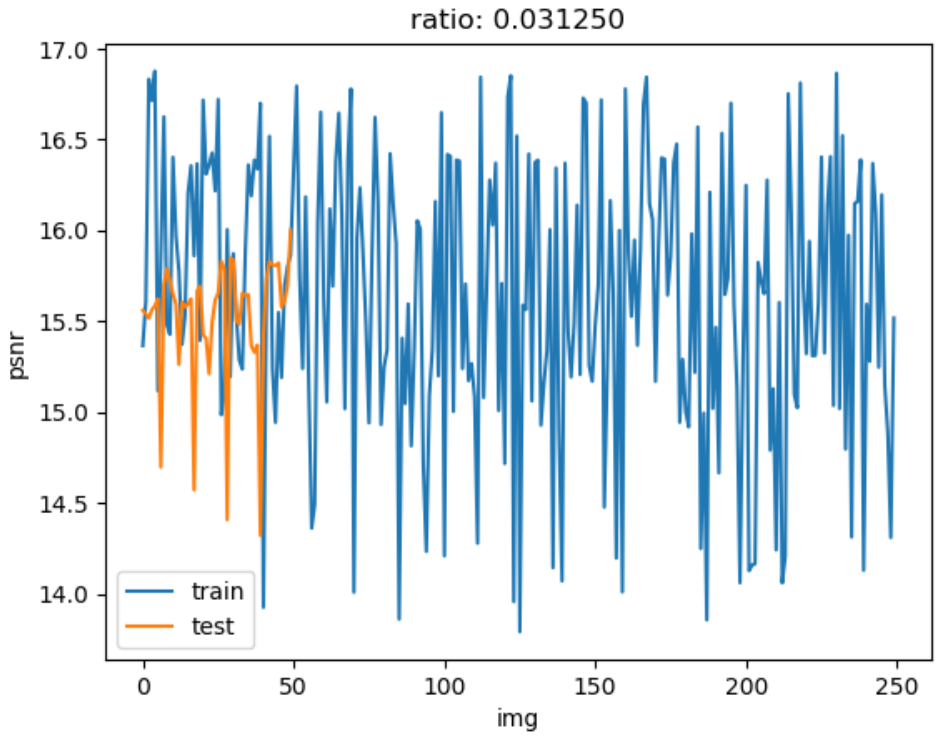

Figure 17: PSNR under a compression ratio of 1/32

From the figures above, we can see that the PSNR values increase only slightly if we used a higher compression ratio, signifying that the compression ratio does not have a large impact. The PSNR value is also relatively low, implying that the neural network does not have a good reconstruction quality after compression.

Convolutional Neural Network:

The average PSNR curves on training set and test set for compression ratios of ½, ¼, 1/8, 1/16 and 1/32 are show in figures 18-22 below.

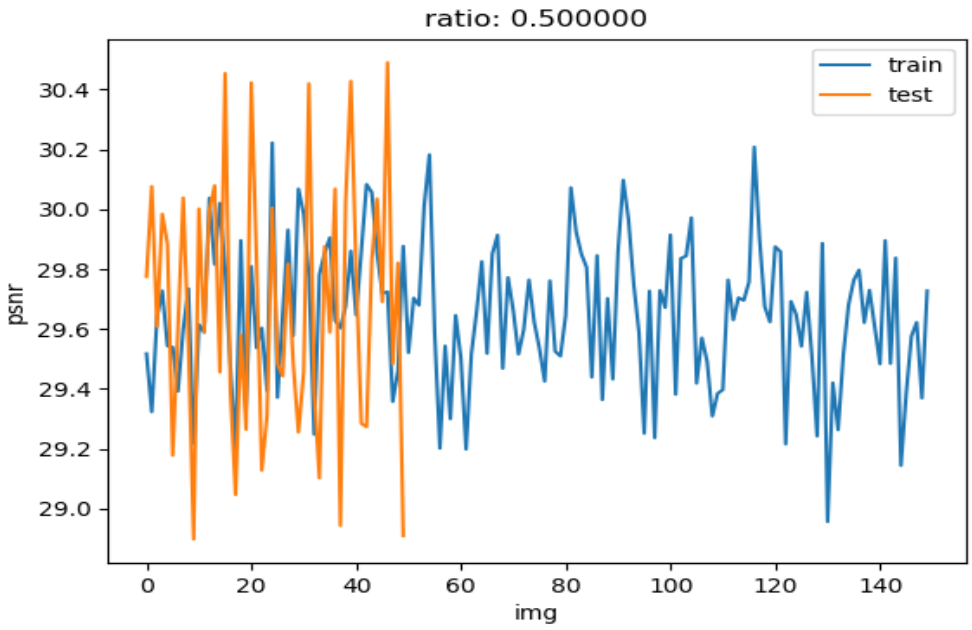

Figure 18: PSNR under a compression ratio of 1/2

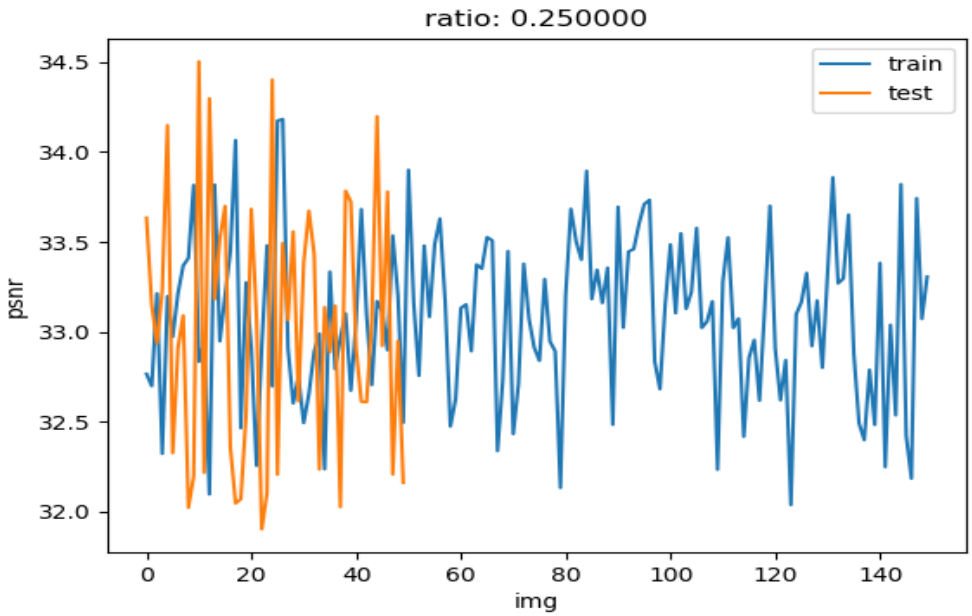

Figure 19: PSNR under a compression ratio of 1/4

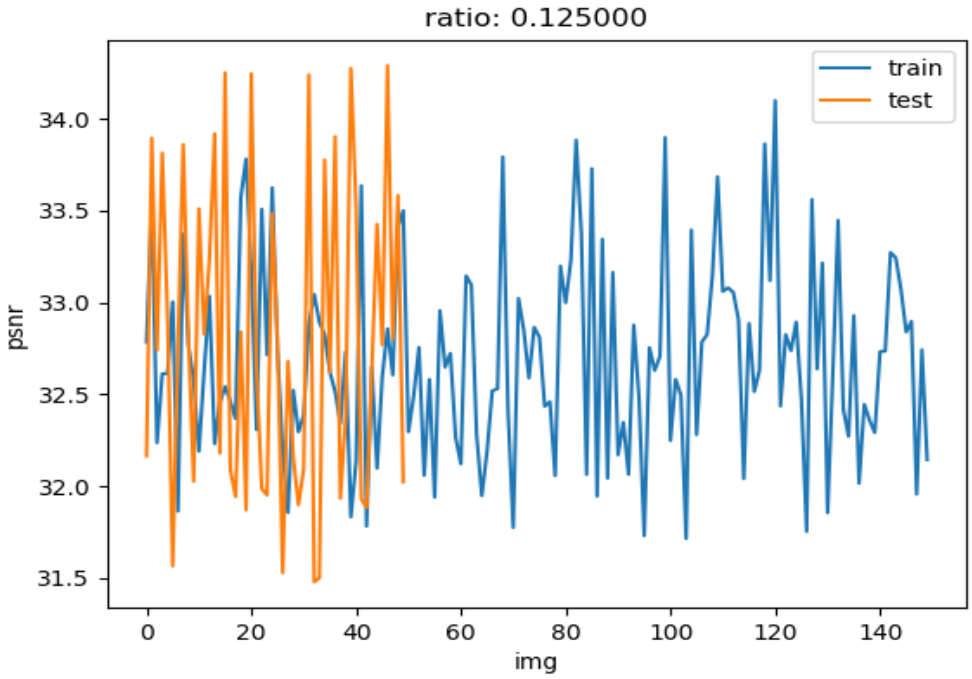

Figure 20: PSNR under a compression ratio of 1/8

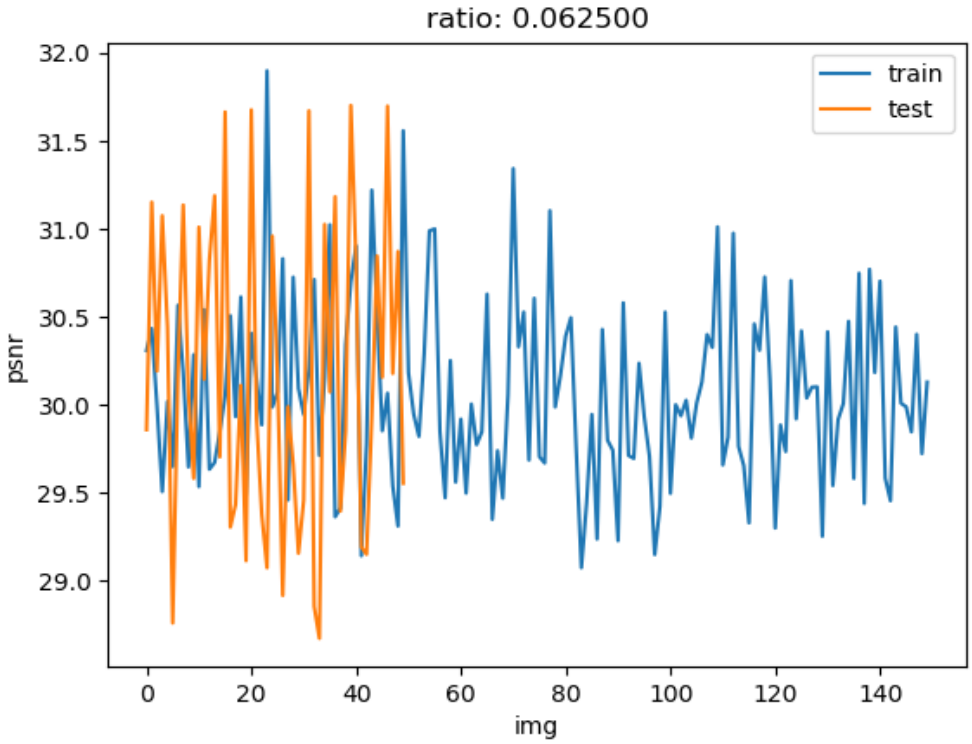

Figure 21: PSNR under a compression ratio of 1/16

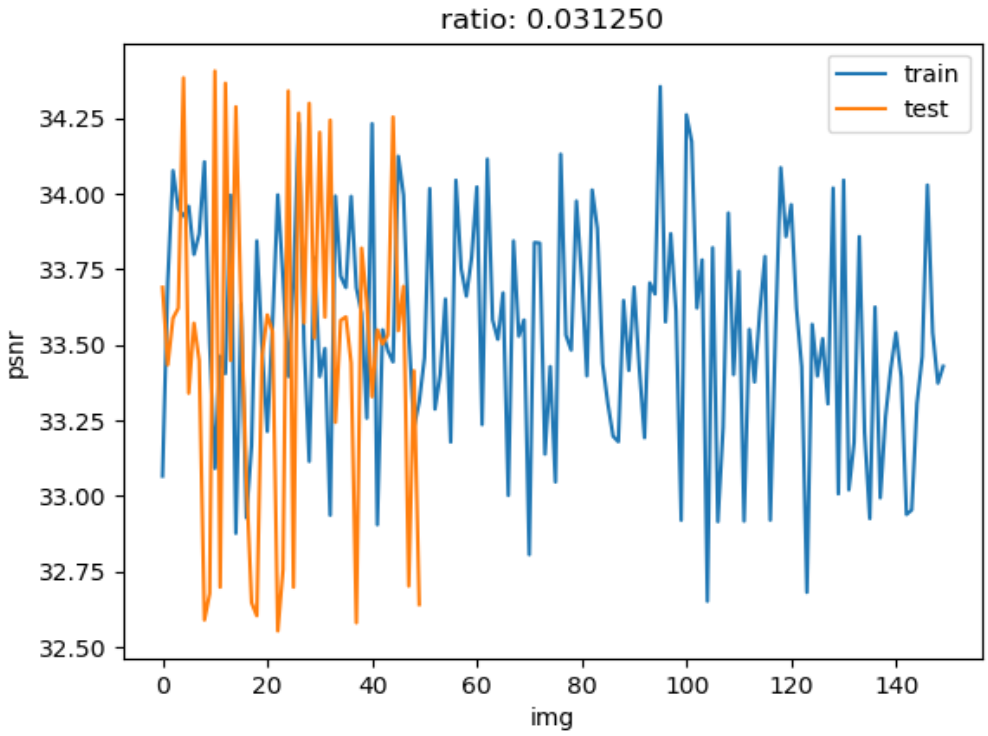

Figure 22: PSNR under a compression ratio of 1/32

The PSNR values are much higher in this method, and with a deeper network the PSNR value seem to increase on average. The shows that the picture reconstruction quality is good.

Figures 23-27 show the average PSNR curves on training set and test set for under compression ratios of 1/2, 1/4, 1/8, 1/16, and 1/32 for the method using a combination of convolutional neural network and fully connected neural network.

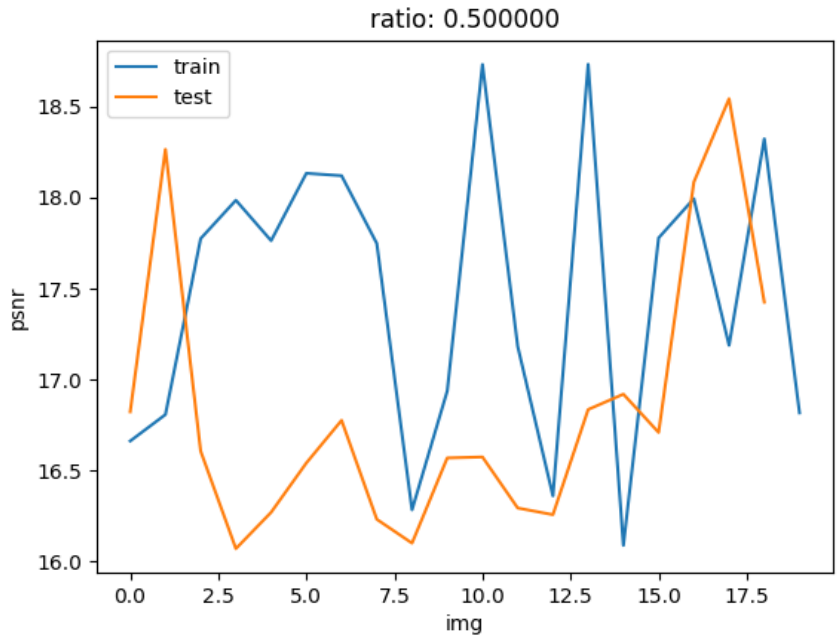

Figure 23: PSNR under a compression ratio of 1/2

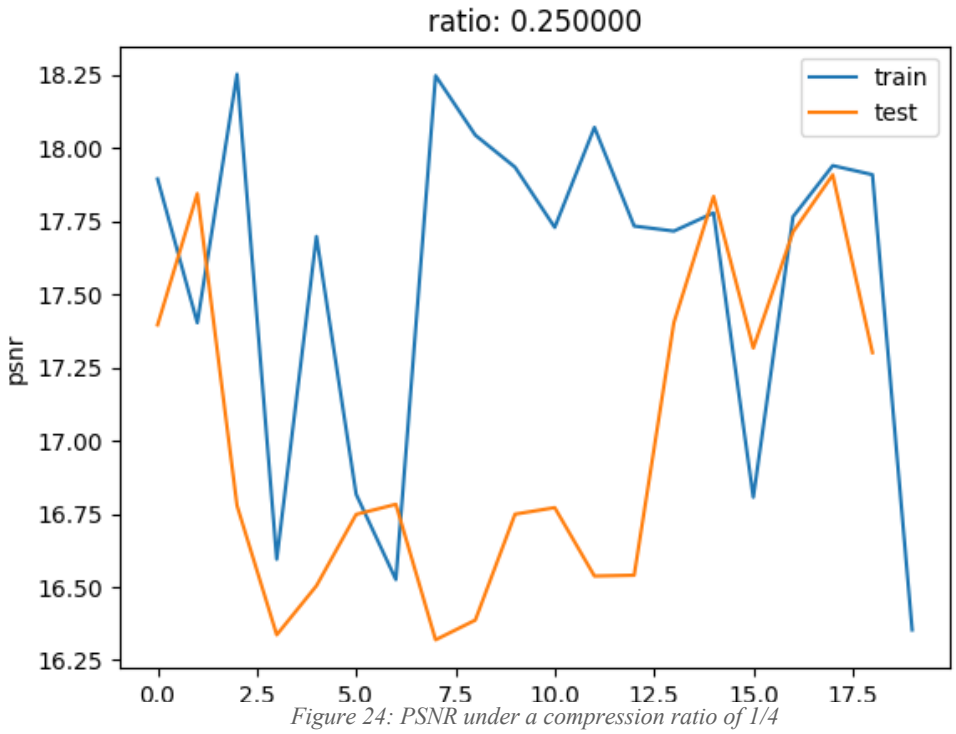

Figure 24: PSNR under a compression ratio of 1/4

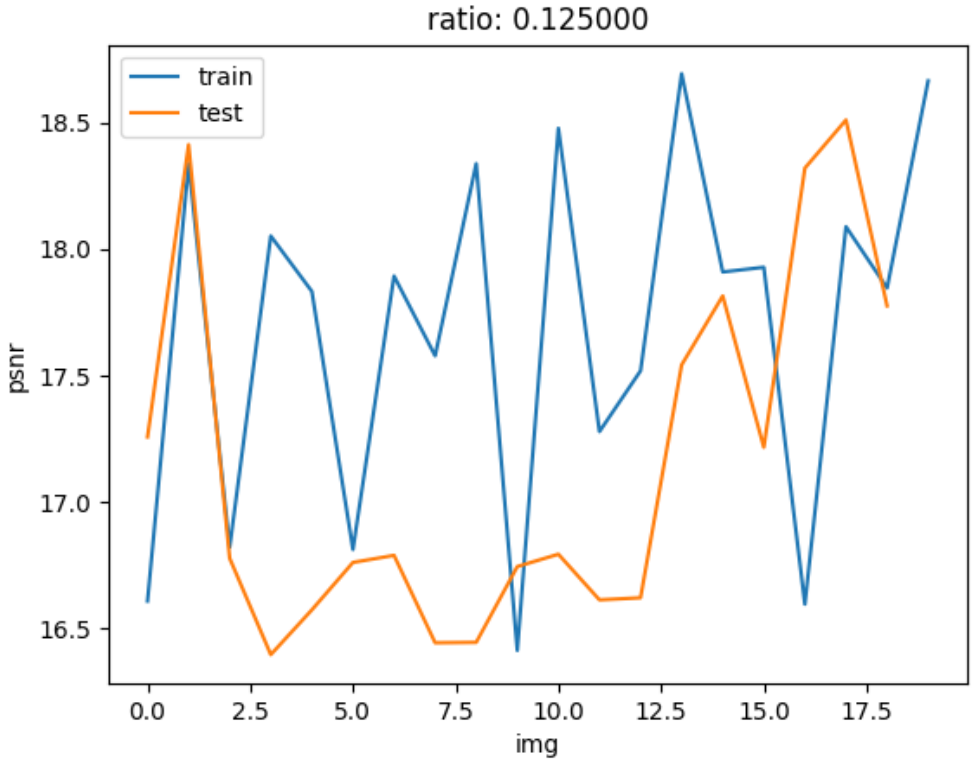

Figure 25: PSNR under a compression ratio of 1/8

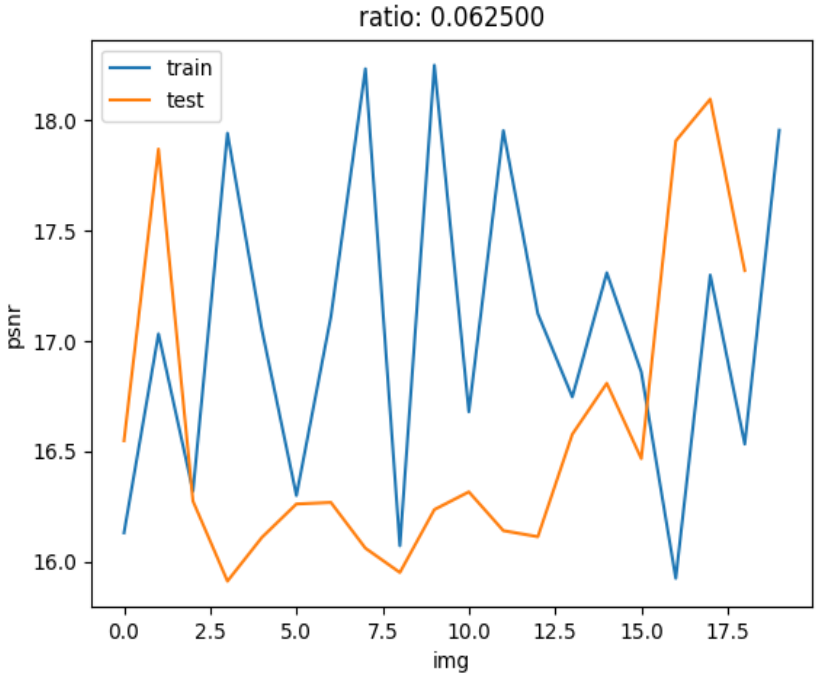

Figure 26: PSNR under a compression ratio of 1/16

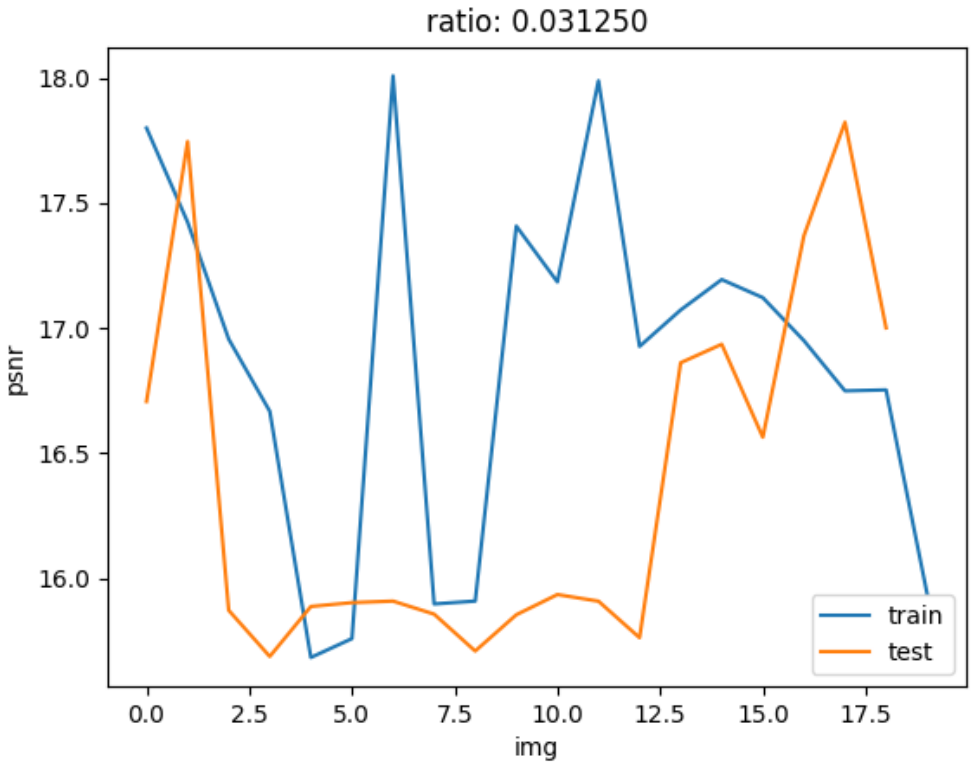

Figure 27: PSNR under a compression ratio of 1/32

From the figures above, we can see that the PSNR values vary erratically with a value of around 17, with no clearly identifiable pattern, showing that the compression ratio does not influence the image quality. The PSNR value is a little on the lower end, showing that the reconstruction quality after compression is not that good.

For the final method (the CNN alternative), the average calculated PSNR is around 7, and in some cases is not even able to calculate an answer or produce an image. This is the worst performing method out of the four.

Between the various compression ratios, generally speaking the PSNR value increases with more compression. We can see that when compression ratio is 1/32, the average PSNR reaches the highest value. This is because the model learns more useful features through the deep network. However, this relationship is not very strong as the plots with ¼ compression ratio and and 1/8 compression ratio seem to do relatively well. The 1/16 compression ratio and ½ compression ratio have the lowest PSNR values, and therefore the worst performance. When compared to the fully connected neural network, we notice that the PSNR values in the convolutional neural network are much higher. The performance of the convolutional neural network is much better.

Perceptual Quality Evaluation

Fully Connected Neural Network:

In order to accelerate the model training, we normalize the input values of the pixels of the images to between 0 and 1. We randomly pick one original image and its corresponding reconstruction image and PSNR for each compression ratio. Figures 28-32 below show the original image and reconstructed image.

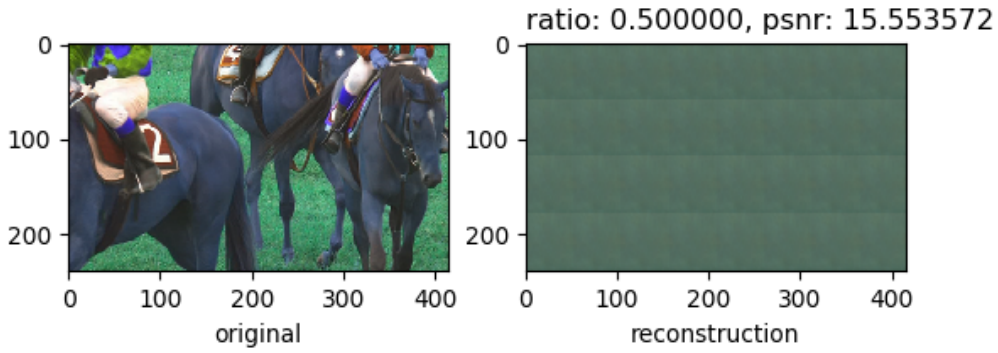

Figure 28: Compression ratio of 1/2

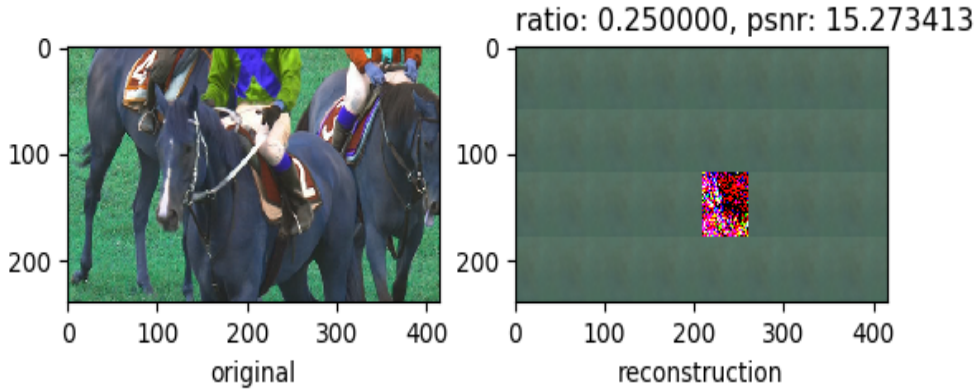

Figure 29: Compression ratio of ¼

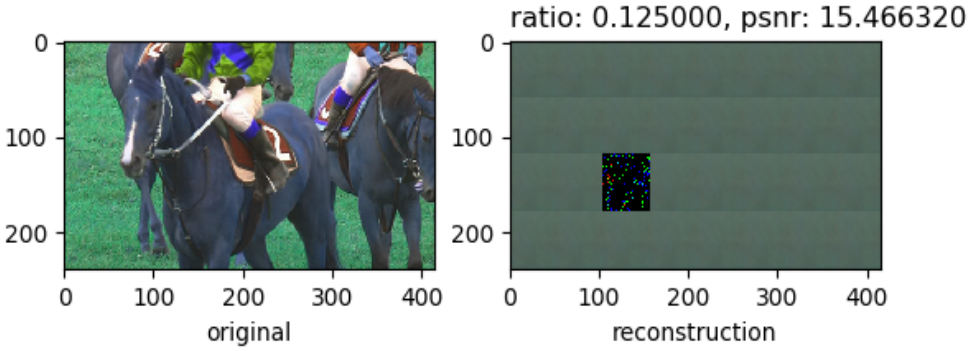

Figure 30: Compression ratio of 1/8

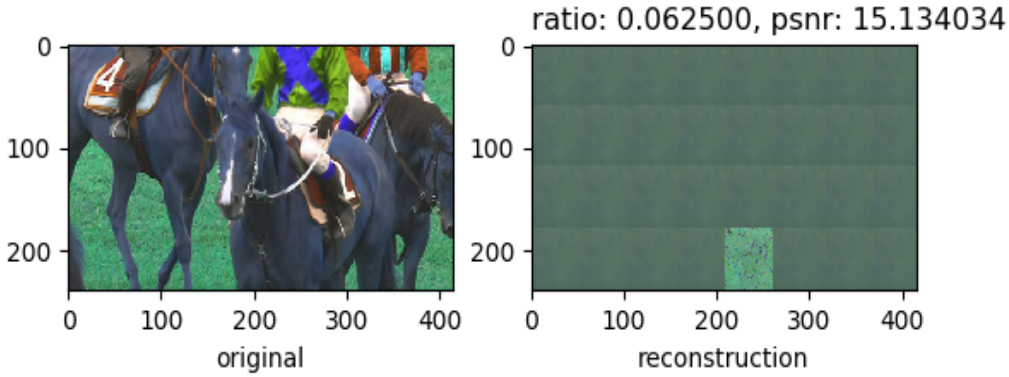

Figure 31: Compression ratio of 1/16

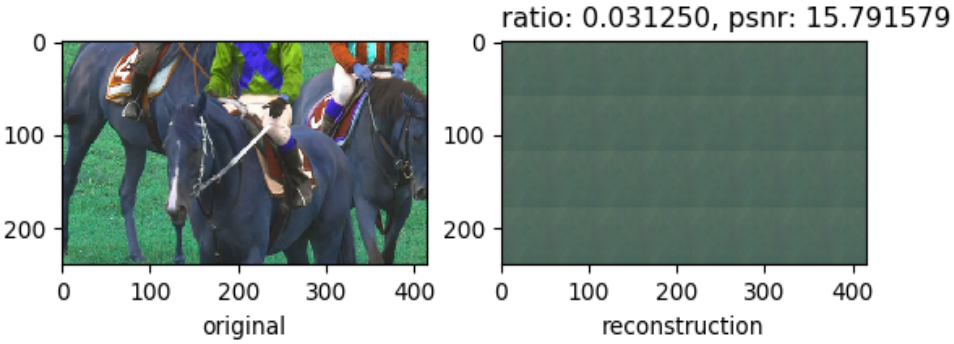

Figure 32: Compression ratio of 1/32

Visually we can see that the reconstruction quality of our model is not very good. The reconstruction image manages to get the background color and the color of the horses, but is not able to differentiate them. Rather, it mixes the two colors together to form an overall background color. The other details are either not portrayed or portrayed randomly and erratically. The fully connected neural network architecture does not have a good result, we suppose that the bad image effect is due to the connection of the pixels, convolution could connect well those connections and relations, but directly extract images into vectors to a fully connected neural network will not learn the relationship between pixels well. We then improve this architecture by adding some convolutions before fully connected layers to extract more features.

Convolutional Neural Network:

Similar to the fully connected neural network, we normalize the input pixels of the following images to be between 0 and 1. We randomly pick one original image and its corresponding reconstruction image and PSNR for each compression ratio. Figures 33-37 below show the original image and reconstructive image with varying compression ratios.

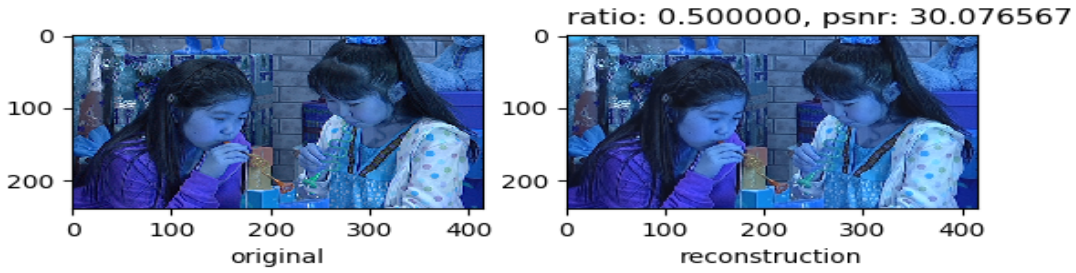

Figure 33: Compression ratio of 1/2

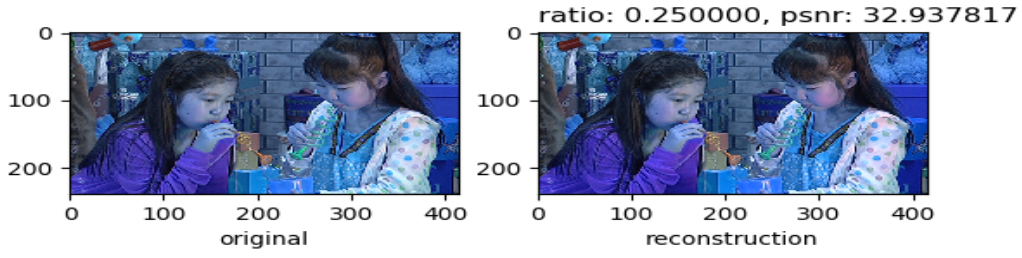

Figure 34: Compression ratio of 1/4

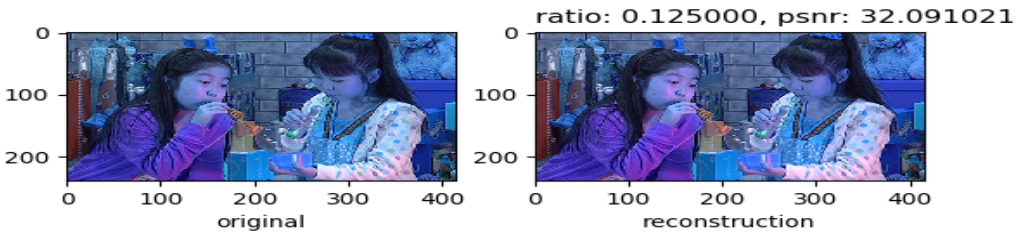

Figure 35: Compression ratio of 1/8

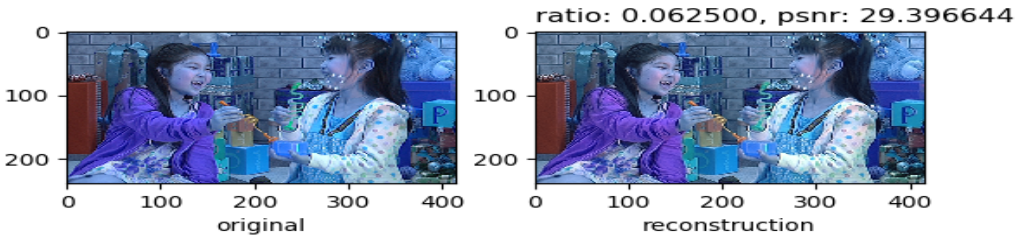

Figure 36: Compression ratio of 1/16

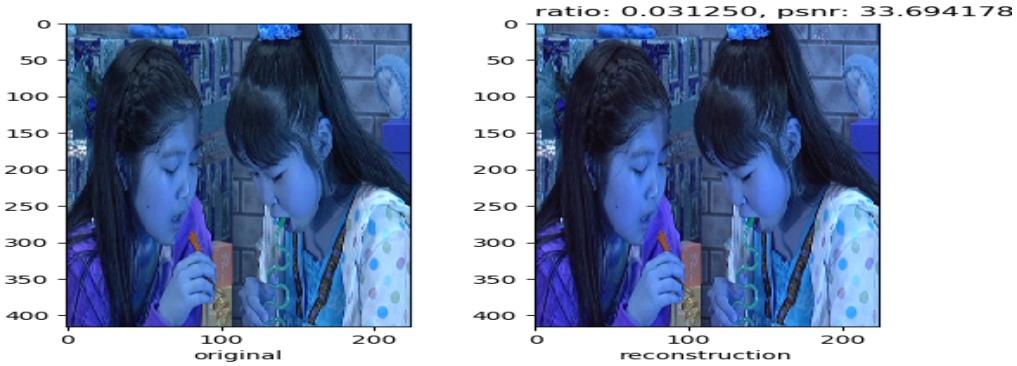

Figure 37: Compression ratio of 1/32

We can see visually that each model has a much better reconstruction quality than compared with the fully connected neural network. The features are shown in the correct location, as well as the correct color. As with a deeper compression ratio, the image quality seems to improve.

Convolutional Neural Network and Fully Connected Neural network:
Similar to the above, figures 38 to 42 below show the reconstructed image using the combination of convolutional neural network and fully connected neural network.

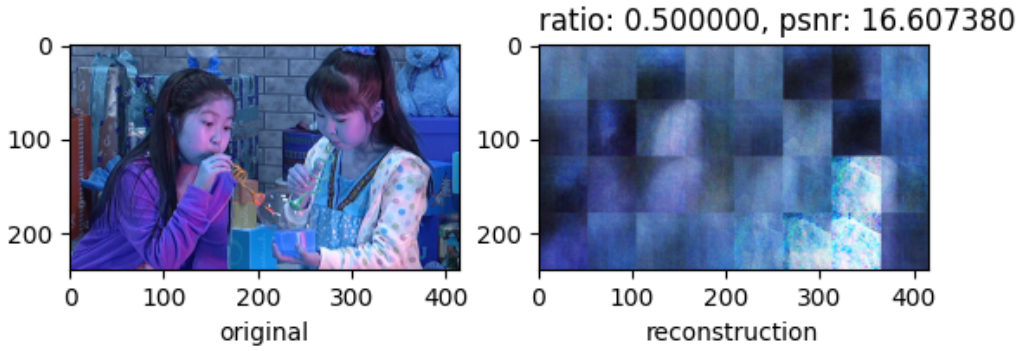

Figure 38: Compression ratio of 1/2

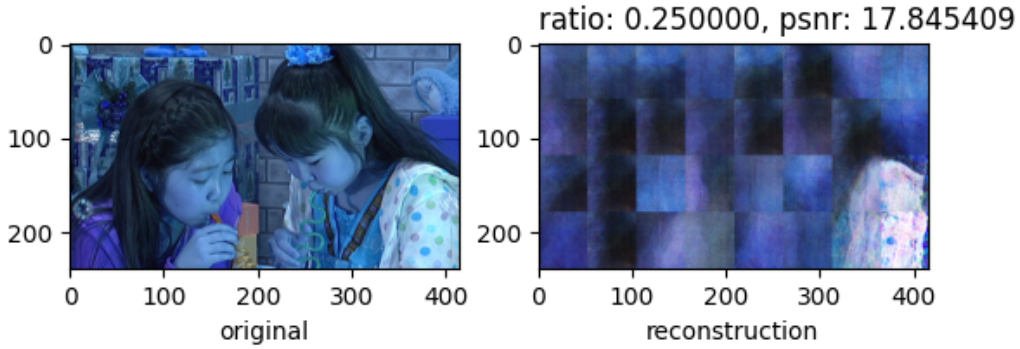

Figure 39: Compression ratio of 1/4

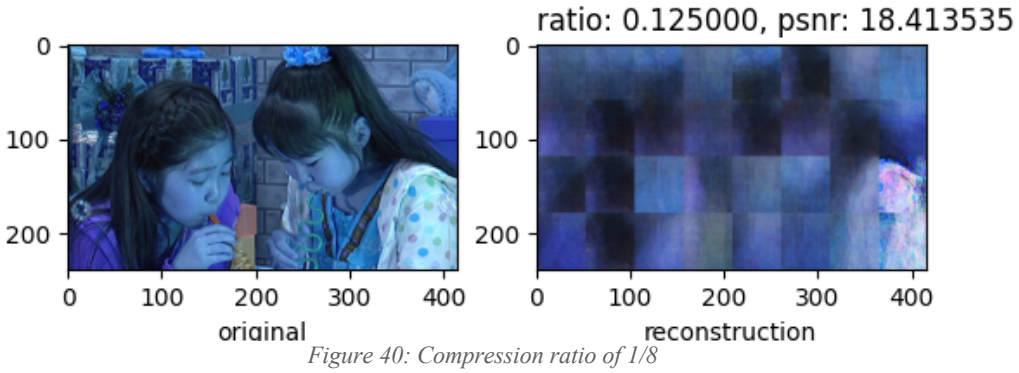

Figure 40: Compression ratio of 1/8

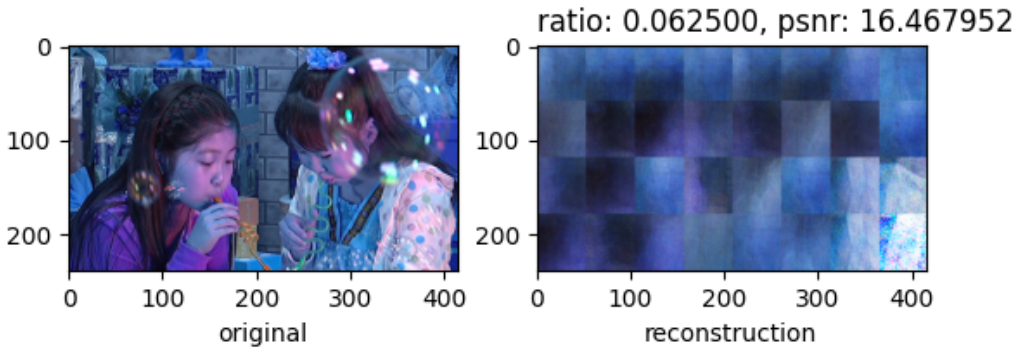

Figure 41: Compression ratio of 1/16

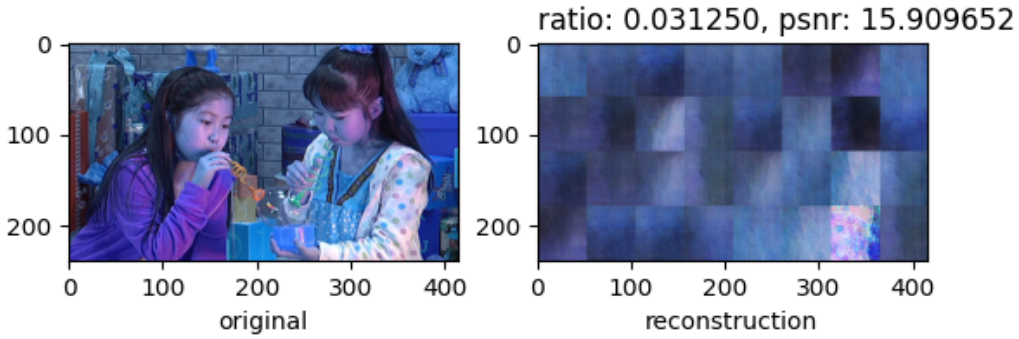

Figure 42: Compression ratio of 1/32

This form of the convolutional neural network combined with the fully connected neural network is not as good as the previous example, but it produces reasonable results. Most of the features are shown in a poor detail, but the approximate color scheme looks correct. The reconstructed image looks ‘pixelated’ as a result.

Convolutional Neural network Alternative
Similarly, figures 43 to 46 below show the reconstructed image using the alternative convolutional neural network.

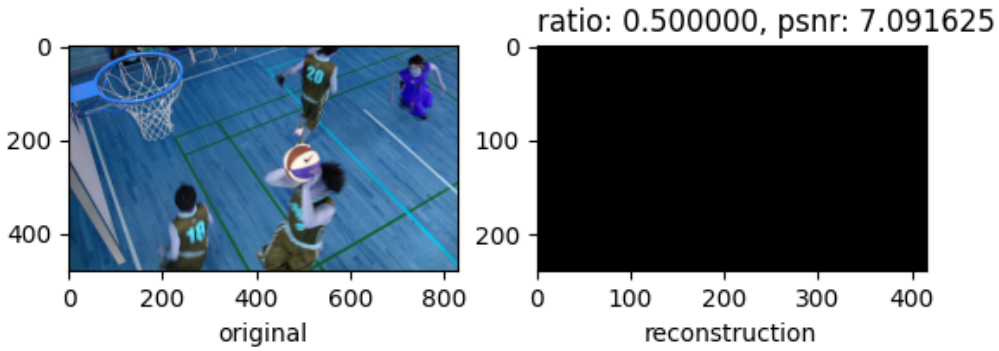

Figure 43: Compression ratio of 1/2

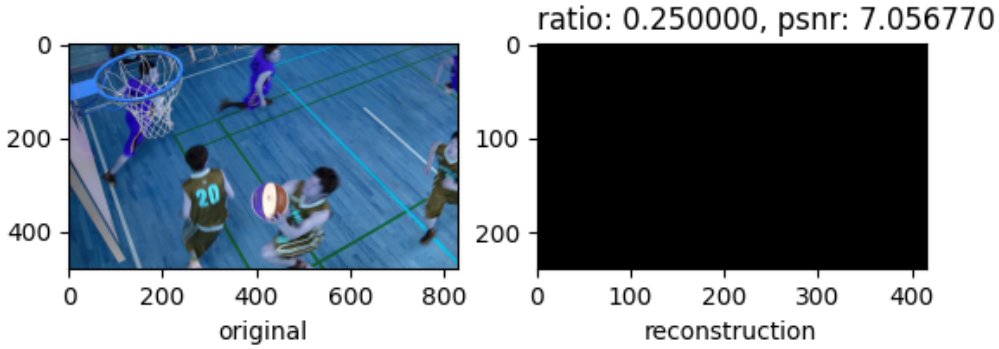

Figure 44: Compression ratio of 1/4

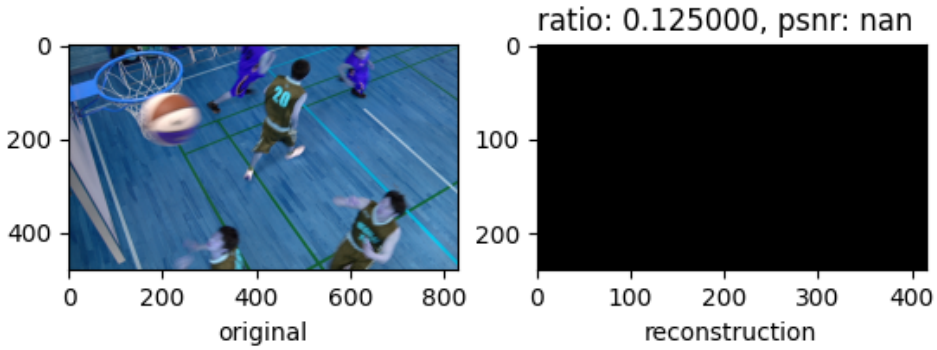

Figure 45: Compression ratio of 1/8

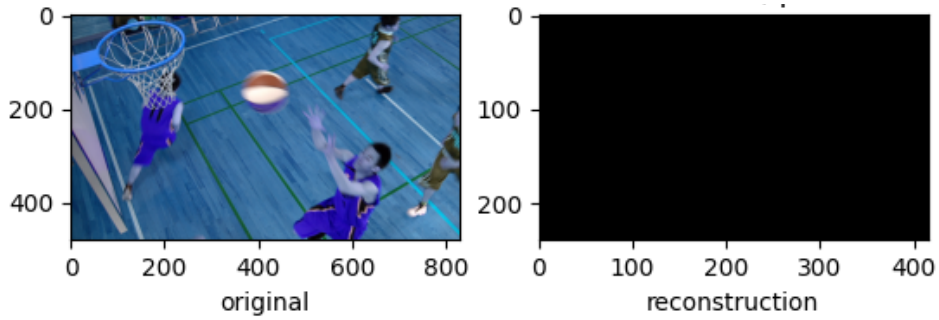

Figure 46: Compression ratio of 1/16

The fourth architecture uses convolutions and pooling exclusively. This method achieves very poor results, and should not be recommended. The reconstruction image is almost not able to represent anything on the actual image.

Complexity and model size analysis

We can calculate the number of floating-point operations by a series of steps. For a two dimensional convolution, we define the input size of the image to have C_{in} channels, height of H and width W. We let K be the number of filters. The vertical padding is P_h, horizontal padding is width P_w, vertical stride be S_h, horizontal stride to be S_w, and let the group number be g. The output size of the convolution layer is K x H' x W', where

$$H' = \frac{(H + 2 \times P_h - k \times H)}{S_h} + 1$$

$$W' = \frac{(W + 2 \times P_w - k \times W)}{S_w} + 1$$

The total number of parameters are:

$$\frac{H' \times W' \times C_{in} \times K \times k_H \times k_W}{g}$$

The number of operations in the convolutional neural network is far more than in the fully connected neural network, by an order of magnitude of two. The number of parameters in the fully connected neural network remain roughly the same for various compression ratios, whilst the number of parameters increase exponentially as the compression ratios are deeper.

Fully Connected Neural Network:

MODEL	Number of Parameters	Floating-point Operations
Net_1	140,453,400	142,132,840
Net_2	162,376,860	164,716,180
Net_3	167,862,990	170,532,250
Net_4	169,237,155	172,071,385
Net_5	169,582,599	172,499,361

Convolutional Neural Network:

MODEL	Number of Parameters	Floating-point Operations
Net_1	445,315	14,023,589,888
Net_2	2,068,867	26,064,224,256
Net_3	8,560,003	38,096,027,648
Net_4	34,518,403	50,123,415,552

Net_5	138,339,715	62,148,595,712

Convolutional Neural Network + Fully Connected Neural Network

MODEL	Number of Parameters	Floating-point Operations
Net_1	115,277,640	169,804,760
Net_2	137,201,100	192,388,100
Net_3	142,687,230	198,204,170
Net_4	144,061,395	199,743,305
Net_5	144,406,839	200,171,281

Convolutional Neural Network Alternative

This method is unable to be completed as some images could not be reproduced.

IV Conclusions and Future Work

The fully connected neural network does not reconstruct images well as the average PSNR value is low. Additionally, the reconstructed image does not look nearly close enough to the original image. However, with the convolutional neural network, the experimental results show that our network can reconstruct images well visually, and a relatively high PSNR also supports this conclusion. This convolutional neural network can be improved further by adding more convolutions, enhancing the ability of extracting and integrating features. The convolutional neural network combined with the fully connected neural network

generally shows the correct features but does not reproduce the image at a high quality. Finally, the alternative convolutional neural network does not produce reasonable results and can be discarded.

References

Y. Yao, Y. Wei, F. Gao and G. Yu, "Anomaly Intrusion Detection Approach Using Hybrid MLP/CNN Neural Network," Sixth International Conference on Intelligent Systems Design and Applications, Jinan, 2006, pp. 1095-1102.

C. Yakopcic, M. Z. Alom and T. M. Taha, "Memristor crossbar deep network implementation based on a Convolutional neural network," *2016 International Joint Conference on Neural Networks (IJCNN)*, Vancouver, BC, 2016, pp. 963-970.

T. Roska et al., "A hardware accelerator board for cellular neural networks: CNN-HAC," IEEE International Workshop on Cellular Neural Networks and their Applications, Budapest, Hungary, 1990, pp. 160-168.

J. Sim, J. Park, M. Kim, D. Bae, Y. Choi and L. Kim, "14.6 A 1.42TOPS/W deep convolutional neural network recognition processor for intelligent IoE systems," 2016 IEEE International Solid-State Circuits Conference (ISSCC), San Francisco, CA, 2016, pp. 264-265.

S. Albawi, T. A. Mohammed and S. Al-Zawi, "Understanding of a convolutional neural network," 2017 International Conference on Engineering and Technology (ICET), Antalya, 2017, pp. 1-6.

Contribution of team members

Li Yuan Zhi 33%

Chun-Hao Liu 33%

Pok Lai Alexander Law 33%