Medical Imaging Enhancement Using Generative Adversarial Networks

Jacob Beallor – 20020613, Jason Kronick – 20024005,
Alexandre Le Blanc – 20008132, Ethan Shore – 20011044

Abstract—Given the cost and complexity of high-quality medical imaging, our team set out to implement a method to enhance the resolution of medical images using machine learning techniques and, in doing so, provide a possible method of reducing the difficulty and costs involved with medical imaging and analysis. The method implemented was a super-resolution GAN, a variation of GANs used for increasing the resolution of images. An SR-ResNet was also implemented. The performance of these deep learning techniques was compared to basic interpolation methods, which were used as a baseline. Overall, the SR-ResNet performed best, with a 2.17% improvement in SSIM, 7.70% improvement in PSNR, and 80.97% improvement in MOS over the baseline interpolation method.

Index Terms—Group ID: 3, Domain: Life sciences, Task keywords: Medical imaging, super-Resolution, SRGAN, SR-ResNet, interpolation, Type of project: Application project.

1 Introduction

1.1 Problem

Finding signs of disease in medical images, *e.g.* CT scans, MRI or X-rays, can be very difficult due to blurry or unclear images. This is especially true during early stages of the disease when markers are small, few, and get easily lost within the images' details due to lack in image quality [10]. Additionally, the equipment and methods used to obtain high resolution medical images are slow and costly. A single MRI scanner can cost anywhere between $300,000 USD to $1 million USD, which can strain a hospital's budget [2].

1.2 Motivation

Reducing the costs of medical imaging is of utmost importance because everyone is at risk of contracting a disease such as cancer, but not everyone can currently afford the costs of medical imaging and treatment. Additionally, improving the accuracy of medical image analysis is crucial because if a tumour goes unperceived, it may lead to untreated disease, and potentially death.

1.3 Problem Statement

Consequently, for our project, our group's objective was:

To enhance the resolution of medical images using machine learning techniques and, in doing so, provide a possible method of reducing the difficulty and costs involved with medical imaging and analysis.

- *Member 1, 2, 3 and 4 are with School of Engineering at Queen's University*
  E-mail: jacob.beallor@queensu.ca; jason.kronick@queensu.ca; a.leblanc@queensu,ca; ethan.shore@queensu.ca

1.4 Main Contributions

In this paper, we explore multiple machine learning methods to perform image super-resolution, in particular, the super-resolution GAN (SRGAN), which is an advancement in generative adversarial networks for image enhancement proposed by Ledig *et al.* in 2017 [8]. Additionally, Ledig *et al.* also use an SR-ResNet model, which shares the same generative architecture with the SRGAN, but instead uses a supervised approach. Based on their findings, we have developed our own SRGAN, SR-ResNet and interpolation models to enhance the resolution of medical images. Assessed using the commonly-used PSNR, SSIM and MOS evaluation metrics, both the SR-ResNet and SRGAN yielded impressive results, with the SR-ResNet slightly outperforming the SRGAN.

2 Related Work

The original paper by Goodfellow et al. [4], which laid out the framework for generative adversarial networks is a related work. This paper was the foundation on which the SRGAN paper was based and was useful in gaining knowledge on GANs in general. Yang et al. [11] proposed a benchmark for the problem of single image super-resolution. The paper outlines an overview of several super-resolution techniques, and was useful in the evaluation of our methods. The paper by Johnson et al. [7] first proposed the use of a perceptual loss function for deep learning super-resolution techniques, and was useful for this paper. Aftab et al. [1] proposed interpolation techniques for the purpose of super-resolution, which was a technique that was used to determine a baseline of quality for the super-resolved images. Ledig et al. [8] proposed GAN as a solution for single image super-resolution in their original SRGAN paper. This paper was very useful in creating the SRGAN model.

3 Dataset

3.1 Dataset Overview

For our experiments, we used the National Cancer Institute's *Clinical Proteomic Tumor Analysis Consortium Glioblastoma Multiforme* (CPTAC-GBM) image dataset [6], made publicly available by the Cancer Imaging Archive. The dataset includes radiology brain images from cancer patients with Glioblastoma Multiforme, a fast-growing brain tumour. The images are taken at various depths and angles of the patients' brains and at various stages of the tumours' growth. There were 156,943 grayscale images available, with varying resolutions, obtained through various medical imaging techniques, such as computer radiology (*CR*), computerized tomography scan (*CT*) and magnetic resonance imaging (*MRI*).

3.2 Preprocessing

Given the limited computing power and time restrictions of the project, only 24,000 images were initially collected from the dataset for preprocessing. The goal of the super-resolution task was to perform an upscaling of a factor of four, as was done in the SRGAN paper. As such, the first step of preprocessing was to select images that were large enough so that the corresponding downsampled images were detailed enough to learn from, but not too large, since this would require more time and computing power to train on. Therefore, images of size 512×512 pixels were selected, as this was the minimal size deemed adequate for the selected upscaling factor. Many of the remaining images contained objects that were small and undetailed, surrounded by large black backgrounds. Super-resolving these types of images would produce uninteresting results and improvement in resolution would be difficult to evaluate. Therefore, a filter was applied to the remaining dataset to remove images of this type. This process began with devising a way to quantify the detail of an image. The proposed method of quantification was to calculate the sum of differences between the values of adjacent pixels in the images, referred to as the pixel difference score. The formula used for this calculation is shown in Equation 1.

$$P_D = \sum_{i=1}^{512}\sum_{j=1}^{511} abs(I_{i,j} - I_{i,j+1}) + \sum_{j=1}^{512}\sum_{i=1}^{511} abs(I_{i,j} - I_{i+1,j}) \tag{1}$$

Where, $I_{i,j}$ is the value of the pixel in the *i*th row and *j*th column of the image. The motivation behind this method was that undetailed images would contain large sections of pixels with similar value causing the sum of differences between adjacent pixel values to be small. By contrast, more detailed images would contain a high number of sections in which the pixel value varied greatly, causing the sum of differences between adjacent pixel values to be larger. One issue with this method, however, is that images containing a lot of noise would obtain the highest scores. To solve this problem, a Gaussian filter was first applied to each image to remove noise before the pixel difference score was calculated. After all preprocessing was conducted, roughly 5000 images remained in the final dataset.

4 Methodology

4.1 Overview of Models

Once the data preprocessing was completed, the methods for super-resolution were selected. The selected methods can be broken into two categories: interpolation and deep learning. The interpolation methods that were used are nearest-neighbour interpolation, bilinear interpolation, and bicubic interpolation. The deep learning methods were SRResNet and SRGAN.

4.2 Interpolation

Interpolation is a method for estimating unknown values based on nearby pixel values and can be useful for upsampling. Nearest-neighbour interpolation replicates the value of the nearest pixel. It is the simplest but the least effective of the methods. Bilinear interpolation uses linear interpolation in both the y-axis and x-axis. A weighted average of four nearby pixels is used. This method is more complex but tends to yield better results than nearest-neighbour interpolation. Bicubic interpolation is the most complex but tends to provide the best results. It uses up to 16 pixels by performing cubic interpolation in both axes. The interpolation methods were performed by downsampling the original images, and then upsampling the images using the OpenCV python package.

4.3 SRGAN

4.3.1 Architecture

Like most variations of GANs, the SRGAN consists of two neural networks: a generative network and a discriminative network. Given a low-resolution input image, the generator attempts to output a realistic version of the image super-resolved to a certain target resolution. Thus, in line with the adversarial nature of a GAN, the discriminator attempts to learn how to distinguish between true high-resolution images and images that were super-resolved by the generator.

The generator consists of a combination of convolutional, batch normalization and PReLU activation layers, however it also uses several residual blocks. A residual block consists of two Convolutional-Batch Normalization-PReLU layers and a skip connection from the input to the output. A diagram of a single residual block can be seen in Figure 1 below.

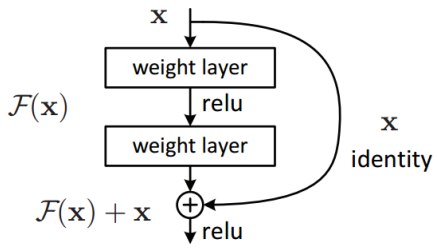

Fig. 1. Residual block architecture [9]

Instead of trying to learn a direct mapping from input to output, the residual block aims to predict the residuals and then adds back the input using the skip connection to

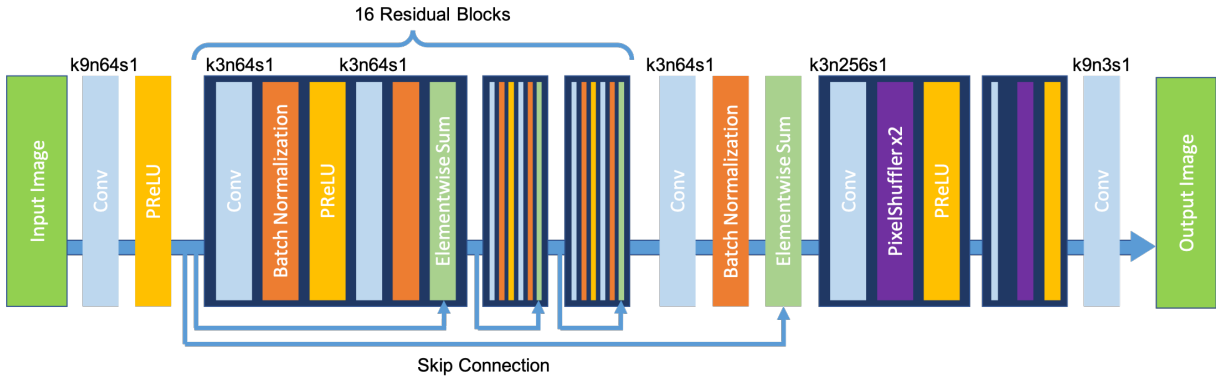

Fig. 2. Generator network architecture

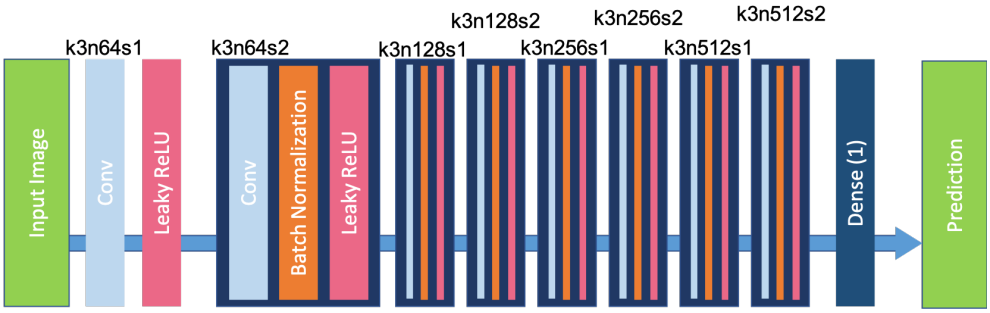

Fig. 3. Discriminator network architecture

achieve the desired output. This skip connection allows for quicker learning due to a stronger gradient and also allows for the identity mapping to be easily learned [9]. The last type of layer involved in the generator architecture is the pixel shuffle layer. This type of layer is added twice near the end of the network to expand the deep feature maps from the convolutional layers to feature maps with greater height and width. By applying 2 pixel shuffle layers, the feature maps are expanded to feature maps with dimensions equal to the target resolution.

To better explain the generator architecture, define kXnYsZ to be a convolutional or transpose convolutional layer with a kernel size of X, Y filters and stride Z. A diagram of the generator architecture using this nomenclature can be seen in Figure 2.

The discriminator consists of convolutional, batch normalization and leaky RELU activation layers followed by a fully connected dense layer with a sigmoid activation to obtain the desired prediction probability output. Using the same nomenclature as above, the discriminator network architecture is shown in Figure 3.

4.3.2 Losses

To learn how to properly super-resolve an image, the generator undergoes two rounds of training. The first round implements a supervised learning method in which the generator is given a downsampled version of an image and then attempts to super-resolve it back to the origin resolution. The original image is used as the "label" of the downsampled image and thus the loss is quantified by simply taking the mean squared error between the original image on the super-resolved image. The resulting trained network is referred to as the SR-ResNet. The SR-ResNet is then fine-tuned using adversarial training by incorporating the discriminator network. While adversarial training implements the traditional cross-entropy adversarial losses to the discriminator and generator networks, a second type of loss called the content loss is also used.

The content loss is calculated by first using a VGG19 network to obtain feature maps of both the original and super-resolved images. The content loss is defined as the mean squared error between the two resulting feature maps. Any of the convolutional layers of the VGG19 network can be selected to obtain the feature map. As such, the layer utilized is given by the j^{th} convolutional layer before the i^{th} maxpooling layer. As recommended by the SRGAN paper, an ideal content loss is calculated using the values 5 and 2 for i and j, respectively. The total loss used to train the generator network is shown in Equation 2 below

$$l_G = l_c + 10^{-3} l_{adv} \tag{2}$$

where l_c and l_{adv} are the content loss and adversarial loss, respectively. The network obtained after the fine-tuning training is completed is referred to as the SRGAN.

4.4 Technical Requirements

To implement the SRGAN and SR-ResNet, Python was used as the base programming language and a combination of the Keras and TensorFlow libraries were used to create the deep learning networks. Additional libraries used include NumPy, Matplotlib, Scikit-Learn and Scikit-Image.

Due to the complexity of the implemented networks, training could not be conducted on any of our local machines. Therefore, execution of network training was performed using a deep learning virtual machine on Google Cloud Platform. The virtual machine was deployed with 60 GB of RAM and four NVIDIA Tesla K80 GPUs.

4.5 Training Procedures

To train the SR-ResNet and SRGAN models, the dataset was first split into training, validation and testing sets. Due to the complexity of these models and the limited available resources, 1400 images were randomly selected from the filtered dataset, of which 1000 were used for training, 200 for validation, and 200 for testing. These datasets were comprised of high-resolution images, which were then downsampled by a factor of four to obtain the desired low-resolution input shapes for the models. The SR-ResNet model was trained for 100 epochs using MSE loss and the Adam optimizer with a learning rate of 0.0001. During training the per epoch loss, SSIM and PSNR scores were tracked on both the training data and the validation data. Upon completion of the SR-ResNet training, this model was then further fine-tuned using the SRGAN architecture and losses for an additional 50 epochs. Similarly, the per epoch losses, SSIM and PSNR scores were tracked during training.

5 Experiments and Results

5.1 Evaluation Metrics

To evaluate each of our super-resolution models and corresponding outputs, we used three commonly-used metrics:

- Peak signal-to-noise ratio (PSNR)
- Structural similarity index (SSIM)
- Mean-opinion-score (MOS)

PSNR estimates the absolute pixel-wise errors between a reference image f and a test image g, both of equal dimensions $m \times n$, using equation 3 below [5]:

$$PSNR = 20log_{10}\left(\frac{MAX_f}{\sqrt{MSE}}\right) \tag{3}$$

where (i, j) represents the individual pixel coordinates and

$$MSE = \frac{1}{mn}\sum_{m=0}^{m-1}\sum_{n=0}^{n-1}\|f(i,j) - g(i,j)\|$$

PSNR values range from zero to infinity. The higher the score, the less the difference between the test and the reference images.

SSIM measures the similarity between two images. It is a better indication of perceived changes between two images, taking into account more factors such as loss of correlation, luminance distortion and contrast distortion. For two images f and g of equal dimensions, SSIM is computed with the following equation:

$$SSIM = \frac{(2\mu_f\mu_g + C_1)(2\sigma_{fg} + C_2)}{(\mu_f^2 + \mu_g^2 + C_1)(\sigma_f^2 + \sigma_y^2 + C_2)} \tag{4}$$

where μ_f and μ_g represent the images' mean luminances, σ_f and σ_g are the images' standard deviations, and σ_{fg} is the covariance between the two images [5]. The C_1 and C_2 terms are constants put in place to avoid a null denominator. SSIM scores range between 0 and 1, where 0 signifies no correlation between the images and 1 indicates the images are identical. To compute the PSNR and SSIM scores, we implemented predefined functions from the Scikit-Image library [3].

While PSNR and SSIM are both purely quantitative metrics, MOS is more qualitative and is a direct representation of an image's quality perceived by the human eye. To obtain the MOS, we designed and conducted a survey and shared it to a random sample of Queen's students. Survey participants were shown multiple sets of four images containing the original image (labeled) and the three super-resolved replications (unordered and unlabeled) generated by the SRGAN, SR-ResNet and the best interpolation models. For each set of images, participants were asked to rate the quality of the three enhanced images on a scale of 1-to-5, where 1 was a poor quality reconstruction and 5 was a high quality reconstruction. The overall average survey scores for each super-resolution method were calculated, yielding the final MOS scores.

5.2 Experimental Setup

Since the three interpolation methods do not require any training, each method was used directly on the 200 validation images and the SSIM and PSNR scores were calculated. Upon analyzing the results, the best performing model by both SSIM and PSNR metrics was the bicubic interpolation, as seen in Table 1. Additionally, samples of the three interpolation methods can be seen in Figure 4.

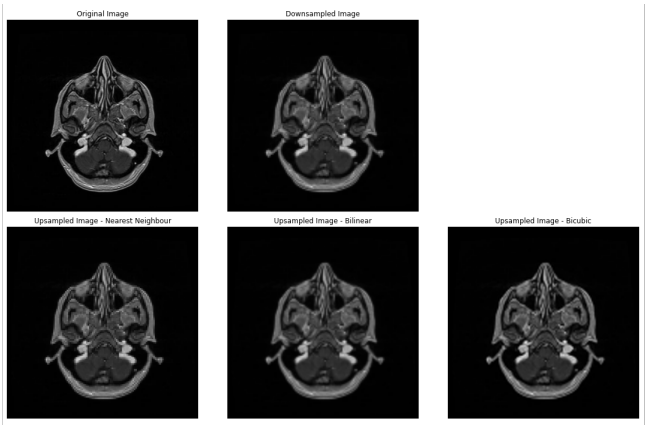

Fig. 4. Sample image reproduction using the three interpolation methods.

Interpolation Method	SSIM	PSNR
Nearest-neighbour	0.89	29.27
Bilinear	0.90	30.72
Bicubic	0.92	32.19

TABLE 1
Interpolation SSIM and PSNR scores on validation data.

Due to the adversarial nature of the SRGAN, the model yielded at the end of all 50 epochs may not be the best performing version. Although the loss may be lowest during

a given epoch, this is not reliable since the strength of the discriminator is constantly changing as well. As such, the validation SSIM and PSNR scores obtained during training were used to select the model at its best performing epoch. Given the bicubic interpolation, SR-ResNet, and the optimal SRGAN model as the top candidates, these models were then further analyzed using the testing dataset to determine the best overall model. For each method the SSIM, PSNR and MOS scores were calculated and compared to select the best model.

5.3 Experimental Results

A summary of the bicubic interpolation, SR-ResNet, and SRGAN performance on the testing dataset can be seen in Table 2. Note that the bicubic interpolation method was used as the baseline to which the other models were compared.

	SSIM	**% Impovement**
Bicubic interpolation	0.92	N/A
SR-ResNet	0.94	+2.17%
SRGAN	0.90	-2.17%
	PSNR	**% Impovement**
Bicubic interpolation	32.19	N/A
SR-ResNet	34.67	+7.70%
SRGAN	30.42	-5.50%
	MOS	**% Impovement**
Bicubic interpolation	2.26	N/A
SR-ResNet	4.09	+80.97%
SRGAN	3.40	+50.44%

TABLE 2
Model performance scores on testing data.

Overall, the SR-ResNet performed best, with a 2.17% improvement in SSIM, 7.70% improvement in PSNR, and 80.97% improvement in MOS over the baseline bicubic interpolation. Sample results of the generated superresolution images from each of the three methods can be found in Figure 5.

Despite the high performance of these models, there appeared to be the possibility of overfitting occurring when training the SR-ResNet. Upon taking a closer look at the SSIM and PSNR histories, the training scores continued to increase as training progressed whereas the model's scores on the validation data did not improve. The SSIM and PSNR history can be seen in Figures 6 and 7, respectively.

Fig. 6. SSIM results on training and validation datasets throughout training.

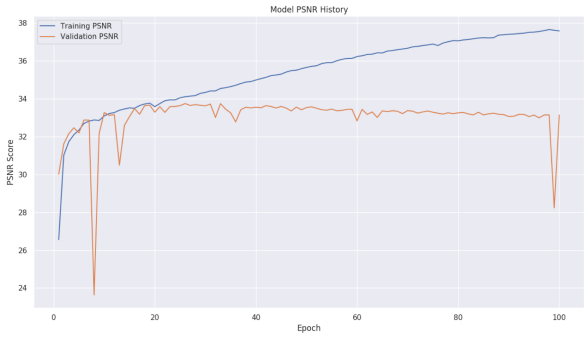

Fig. 7. PSNR results on training and validation datasets throughout training.

6 Discussion

A number of roadblocks were run into throughout the experiments and model building, the greatest of which were memory errors with the GPUs. Originally we intended to use a batch size of 32 for our SRGAN, however this far exceeded what the GPUs could handle. As a results, the batch size was reduced to one. Additionally, we tried using a dense layer with 1024 nodes in our discriminator, as proposed in the original SRGAN paper, but this had to be removed as well due to limited memory. Unfortunately, the only real way to avoid these obstacles would be to obtain access to significantly better computing power, which would be too costly.

As mentioned earlier, overfitting was also observed in the SR-ResNet. To fix this we could attempt to make the model more robust by implementing dropout layers or, once again, increasing the batch size. Furthermore, we found it interesting that the SR-ResNet outperformed the SRGAN, as it seemed the addition of the adversarial training to the SR-ResNet did not improve the results. This may be attributed to the aforementioned limitations in the SRGAN, and due to the fact that only 50 epochs of training were performed.

Finally, it is important to note the limitations in the evaluation metrics being used. PSNR and SSIM are known for being somewhat flawed, since "although blurry images are not perceptually realistic enough, they can still achieve relatively high PSNR/SSIM" [12]. This could explain why the bicubic model outperformed the SRGAN in both SSIM and PSNR, despite being noticeably worse. This is a common discovery in many papers discussing super-resolution. For this reason, we the mean-opinion-score was also implemented. However, although MOS is supposed to be a more accurate representation of the quality of an image perceived by humans, we were limited in our ability to perform a reliable MOS test. We were only able to obtain 68 survey responses, and it was challenging to properly train survey respondents in terms of rating the generated images. Some participants had completely misunderstood the assignment and ranked the images as 1 being the best, 2 being the second best and 3 being the worst. Rating the images was also challenging for survey participants, as many completed the survey on their mobile devices and their screens would be too small to notice the subtle differences and missing features in the generated images.

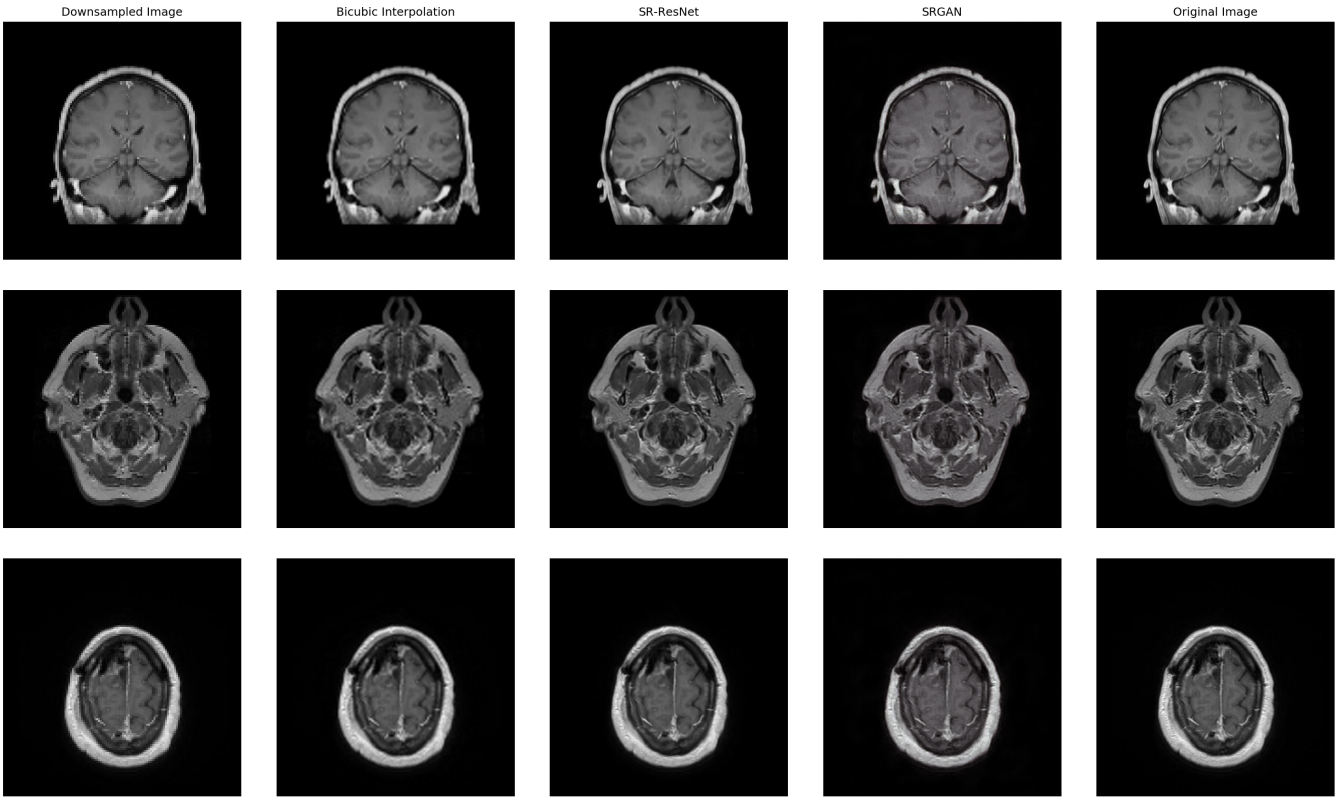

Fig. 5. Sample results of the generated superresolution images from the bicupic interpolation, SR-ResNet, and SRGAN models

7 Conclusion and Future Work

7.1 Conclusion

In conclusion, medical images can be difficult to examine due to low image resolution, and medical imaging equipment is expensive. Hence our objective was to enhance the resolution of medical images using machine learning techniques and, in doing so, provide a possible method of reducing the difficulty and costs involved with medical imaging and analysis.

To achieve this, we produced several super-resolution models. First was an assortment of interpolation methods, which acted as a good baseline to our deep learning models. The deep learning models produced were the SR-ResNet and SRGAN. Both models significantly outperformed the interpolation methods overall, however, the SRGAN surprisingly did not improve upon the SRResNet.

Overall, we were impressed with the results and believe this is a good proof of concept and a step towards providing improved medical imaging at a reduced cost.

7.2 Future Work

As next steps for the project, we believe the SRGAN's performance would greatly benefit from the inclusion of the 1024 dense layer and dropout layers, increasing the batch size to 32, and running 100 epochs of training instead of 50. To do so, improved computing power would be required. Collaborating with the Queen's School of Computing would be an ideal option to obtain the needed compute power, otherwise we would have to significantly increase the capital invested into cloud computing.

Additionally, to further facilitate the task of analysing medical images, we could look into the Lesion-Focused Super-Resolution GAN (LF-SRGAN), a recent state-of-the-art advancement proposed by Jin *et al*. In the LF-SRGAN, a medical image first goes through lesion-detection to determine the particular region of interest, then the SRGAN is applied to this smaller region, as opposed to the whole image, which provides no additional information [12]. This method shows promise and would be a great future step towards solving our problem.

8 Group Member Contributions

A breakdown of each group member's contribution is presented below:
Jacob Beallor:

- Google Cloud Platform environment setup
- Interpolation method implementation
- SR-ResNet and SRGAN development and training

Jason Kronick:

- Google Cloud Platform environment setup
- SR-ResNet and SRGAN development and training
- Data preprocessing

Alexandre Le Blanc:

- Researched evaluation metrics and determined the PSNR and SSIM Python implementation methods
- Created and distributed the MOS evaluation survey

- Error handling and debugging, assisted with SRGAN development
- Google Cloud Platform environment setup

Ethan Shore:

- Helped research evaluation methods
- Helped in downsampling the images
- Helped in debugging the SRGAN code
- Google Cloud Platform environment setup
- Github repository setup

9 Replication Package

The replication package and dataset source can be found in the following Github repository:

https://github.com/ethanshore/SRGAN-Medical-Imaging

References

[1] H. Aftab, A. B. Mansoor, and M. Asim. A new single image interpolation technique for super resolution, 2008.

[2] J. R. Corea, A. M. Flynn, B. Lechêne, G. Scott, G. D. Reed, P. J. Shin, M. Lustig, and A. C. Arias. Screen-printed flexible mri receive coils. nature communications, 2016.

[3] CVNotes. Psnr and ssim metric: Python implementation, 2020.

[4] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets, 2014.

[5] A. Horé and D. Ziou. Image quality metrics: Psnr vs. ssim, 2010.

[6] N. C. Institute. Cptac-gbm. cancer imaging archive, 2020.

[7] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution, 2016.

[8] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi. Photo-realistic single image super-resolution using a generative adversarial network, 2017.

[9] S. Sahoo. Residual blocks – building blocks of resnet, 2018. Accessed: 04/08/20.

[10] N. A. Shonka, J. S. Loeffler, D. P. Cahill, and J. de Groot. Primary and metastatic brain tumors. cancer network, 2015.

[11] C.-Y. Yang, C. Ma1, and M.-H. Yang. Single-image super-resolution: A benchmark, 2014.

[12] J. Zhu, G. Yan, and P. Lio. How can we make gan perform better in single medical image super- resolution? a lesion focused multi-scale approach, 2019.