Machine Learning-Based Super-Resolution MRI

Team #36: Clayton Green, Mathew Shaffer, and Andrew Musk

1 Introduction

Magnetic resolution imaging (MRI), is one of the greatest technological developments of the 20th century [1]. It provides invaluable images of soft-tissue in the body, which are virtually impossible to obtain using any other medical imaging techniques, in a non-invasive and harmless way [1]. Its use cases are seemingly endless, from torn ligament and tumor detection to anomalies in the brain and spinal cord. Even though MRI scans were developed in the late 1970s, there is still a large amount of research into methods to enhance MRI scans even further, due to their usefulness and prevalence in the medical industry [1]. One of the most important and most recent of these is the use of machine-learning in order to enhance and sharpen the MRI images [2]. These higher quality MRI images provide medical professionals deeper insight into patients' conditions and a greater ability to diagnose specific ailments.

Sharpening and enhancement of images in general has been a very successful use case of machine learning, especially in the social media age when images are taken and distributed more than ever [3]. Things like pinch to zoom are enhanced in real-time using sophisticated machine-learning algorithms which can enhance the quality of the portion of the photo that was zoomed in on [3]. The work done on general image enhancement has been applied to medical imaging, and MRIs specifically, and has been largely successful.

The general idea behind super-resolution imaging is to find an **f** that satisfies **Y = f(X)**, where **X** is the low-resolution input image, **Y** is the high-resolution image that is outputted, and **f** is a mapping function that converts the low-resolution to high-resolution [2]. The high-resolution image will have a greater number of pixels than the low-resolution. The exact increase in the number of pixels is determined by the enhancement factor. Depending on this factor, the super-resolution algorithms will have to convert each pixel in the image into two or more pixels. This shows the difficulty of super-resolution imaging, as the algorithms must try to use information in the image in order to expand it to include many more pixels than the original [2].

Super-resolution without the use of machine learning generally relies on linear methods which use fixed linear combinations of pixels from the low-resolution image to create the increased number of pixels in the high-resolution image [3]. These filters are very fast in runtime as they are a "constant convolution kernel applied uniformly across the image" [3], meaning they do not have any adaptive qualities that are learned, and therefore cannot be applied differently depending on what is included in the pixel, or patch of pixels, being examined. The reason for the linear methods speed is also the reason for their weaknesses, they are too simplistic to bring out vivid detail in the high-resolution images [3].

There are many machine-learning methods currently being studied that implement super-resolution image sharpening. One such method is super-resolution volume reconstruction from slice acquisitions. This method is specialized for taking MRI images of moving subjects, such as infants or animals who do not know to stay still [4]. Motion can cause severe inter-slice artifacting in out-of-plane views, but the use of volume reconstruction can generate a high resolution image [4]. This method opens the door to future research in early brain development and fetal brain anomalies.

While there are many specialized super-resolution imaging techniques like the one above, each with its own advantages, the approach used in this paper to perform super-resolution image enhancement is the Google RAISR algorithm. RAISR stands for Rapid and Accurate Image Super Resolution, and it is a method to quickly increase the pixels and picture quality of an image. The algorithm can be used either on basic low-resolution images, or on images that are first passed through a cheap linear method upscaler. The algorithm works by creating a set of filters from training data, which consists of pairs of low and high quality images. These filters are learned during training, and then used during runtime. When RAISR is used on a new input image, it selects the best filter for each patch of pixels in the image. It will then use the filter to create the enhanced, higher resolution patch that is outputted. A hashing function is used in order to select which filter is used to ensure the runtime is as short as possible without losing accuracy. RAISR has a run-time of one to two orders of magnitude faster than other super-resolution methods, as it was designed to be lightweight with equal or better performance than other current methods. The RAISR method is also capable of producing a sharper image without introducing unwanted effects such as noise amplification, artifacting, and halos. Other image upscaling methods, especially those that use linear interpolation, can cause aliasing artifacts, as they are limited in reconstructing complex structures [5].

The use of RAISR and its exceptional runtime compared to other super-resolution algorithms is what guides this study. It is beneficial to medical professionals, medical institutions, and patients to be able to enhance MRI images in a cheap, quick way. The RAISR algorithm is able to be

run on a mobile device, which shows its beauty and the advantages it can bring to the medical field [3].

2 METHODS

For this project, we intended to explore the use of RAISR as a method for restoration of low resolution images. There are currently a wide range of restoration methods that exist, however, each with their own set of weaknesses. Simple Image Super Resolution (SISR) is the process of generating a high resolution image from a low resolution image. SISR is widely used and there are a variety of methods that aim to use it.

Some of the simplest methods for upscaling images include nearest-neighbor, bicubic and bilinear [5]. These methods are collectively called linear interpolators. These are attractive for their minimal complexity. However, they often result in artifacts in the produced image as they do not consider the content of the image [5]. Other methods include self-similarity methods, sparsity methods, and Gaussian Mixtures, all of which produce higher resolution images at the cost of higher complexity. Another range of methods is that of example-based methods which emphasis learning a mapping between low resolution and high resolution images. This is done by learning from a database of images and is a category that RAISR falls into.

RAISR fundamentally differs from these methods in that it aims to provide great restoration but with much greater time and memory efficiency. The way in which RAISR does this can be broken into three steps which are described in the following sections.

2.1 Upscaling of LR image with cheap interpolation

The first step in RAISR is to choose a cheap interpolation method and learn a dxd filter h for a patch of the image. This is done by minimizing the euclidean distance upscaled version of the image and the high resolution training image. This is done according to the following formula [5]:

$$min_h \sum_{i=1}^{L} || A_i b - b_i ||_2^2$$

Here A is matrix of patches extracted from the upscaled images with each row representing a patch. B_i is a vector of pixels from the high resolution image. This algorithm is updated to make it faster by taking advantage of matrix multiplication and writing this optimization as [5]:

$$min_h || Q_h - V ||_2^2$$

In this case Q ($A^T A$) and V ($A^T b$) can be calculated by cumulatively summing dot products of rows [5]. Therefore RAISR makes the global filtering calculations of tables very efficient by exploiting parallelization and make it very memory efficient. This filtering achieves much better performance than standard linear upscaling methods through its learning algorithm and though it doesnt perform as well as state-of-the-art algorithms, it achieves much better efficiency [5].

2.2 Generation of Hash Table of Filters

The next part of the RAISR method is to generate a hash table of filters to be used by test images. The filtering of test images can be fast as it is simply the application of a filter for each patch. However, two things are done in terms of hashing, to make this more efficient.

Firstly, considering the enormous range of filters that could be created, filters are keyed based on local gradient statistics. All the filters with the same statistics are grouped into buckets [5]. Through this, the number of filters is reduced, and hashing is done more efficiently. The 3 statistics considered are angle, strength and coherence.

Secondly, considering that a large dataset is required to produce a reliable filter and that there are common hash values that are seen in most patches, more data is required. This is solved using patch symmetry. Patches can be rotated and mirrored on each of their axes, resulting in x8 the training data [5].

2.3 Selective Blending of Previous Steps

As a result of the previous two images, we gain a new high resolution image. However, through the previous two steps we can have structure deformations from sharpening. As a way to avoid this, we introduce blending. Through blending, areas where the structure of the original image and the produced image are similar, we keep the produced image. However, in areas where there is a significant difference, we keep the initial upscaled image. As a result of this, we end up using a weighted average of the produced RAISR image and the initially upscaled image [5].

2.4 Experimentation

For this project, a data set consisting of MRI images from twenty patients was used. These images were provided in the Digital Imaging and Communications in Medicine (DICOM) format. These images were converted to JPEG format, which is the format that the RAISR algorithm accepts, using the xnViewMP application. Each patient had around fifteen different MRI images. The resulting data set consisted of approximately three hundred separate images.

Because each patient's data came from a single MRI session, most images for a given patient were fairly similar. This meant that any patient used in the training set could not be included in the test set. If some images from a patient were included in the training set and some in the test set, the filter will have been built on MRI images that are very similar to images in the test set, resulting in inaccurate performance on the test data.

Once the patients were split into training and test sets, the model was trained using the train.py file. The high resolution images could be directly inputted for training, as the training file performs the image compression so that the model can be trained using a high and low resolution version of the images. The test set, however, must be manually re-sized before testing can begin. The upscaling factor must be determined during training, as the model must know what factor the image size will by increased by. The test images were compressed by the corresponding upscaling factor used during training. When testing is performed,

using the test.py file, the image size will then be increased by this factor, allowing for performance measurements comparing the original high resolution images and the upscaled version of the compressed images passed to the testing stage.

Due to the limited amount of training data, we wanted to perform cross-validaton on the data set so that we could include all 20 patients in training to hopefully get a more accurate measure on performance. However, because the publicly released version of Google's RAISR implementation was used, which does not have the code built for speed of runtime, it was unrealistic. Training on a single image took anywhere from 3-5 minutes on our testing platform, so performing training on the images for a single iteration of cross-validation would take anywhere from 10-20 hours, and this would have to be repeated up to 20 times. For this reason, we did not perform cross-validation.

3 Results

The RAISR algorithm showed promising results for applications in MRI super resolution. After running test.py on low-resolution MRI images, high-resolution equivalents of the test images were generated. These output images were twice the resolution of the input when using an upscaling factor of two. Upon initial visual observation of the images, the algorithm appeared to have successfully generated accurate upscaled images with minimal to no loss. When manually compared to the original high-resolution image, the output images maintained all major features of the MRI. Figure 1 shows a comparison of the output to the input when the input is zoomed to be of the same size. To quantify the accuracy of the RAISR algorithm, we used three main evaluation factors. We measured the Mean Squared Error (MSE), Peak Signal to Noise Ratio (PSNR), and Structural Similarity Index (SSIM) by comparing the original image to the algorithms output. These three factors sought to verify the success observed in our qualitative comparison.

3.1 MSE

The Mean Squared Error is a widely used loss function to evaluate error in applications. The MSE, however, heavily weighs outliers in the data which can cause inaccurate results regarding the similarity between the original image and the RAISR output [6]. For example, if a line of white pixels is off by one, it could cause the entire row to have the maximum error possible under MSE. Using the MSE is a good start, but it cannot be the only method used.

$$MSE = 1/n * \sum_{i=1}^{n} (orig_i - enhanced_i)^2$$

3.2 PSNR

Peak Signal to Noise Ratio is normally used as a measurement for the quality of reconstruction in lossy compression [7]. The end result of the RAISR algorithm is very similar to a compressed version of the original input image, so PSNR is a good measurement for error. PSNR is a logarithmic decibel scale where the higher the value, the better the quality of the output image [7]. If the images are identical, the MSE will be 0 and the PSNR will go to infinity. The equation for PSNR is [7]:

$$PSNR = 10 * \log_{10} \frac{Max_I^2}{MSE}$$

3.3 SSIM

The Structural Similarity Index is a method for predicting the quality of digital images and video. The SSIM index is a full reference metric which compares windows of an image to an uncompressed original image as reference [8]. The SSIM formula is based on luminance, contrast, and structure of the image. The SSIM index is intended to be an improvement on MSE and PSNR. While those other methods estimate absolute errors in pixel intensity, SSIM is used to analyze changes in structural information [8]. A value of 0 represents no structural similarity between the two images being compared while a value of 1 represents that the images are the same image [8].

$$SSIM_{(x,y)} = \frac{(2\mu_x\mu_y+c_1)(2\sigma_{xy}+c_2)}{(\mu_x^2+\mu_y^2+c_1)(\sigma_x^2+\sigma_y^2+c_2)}$$

3.4 Performance

The first test that was conducted used images from three patients to train the model. The default parameters of the RAISR implementation were used which gave an upscaling factor of two with a patch size of eleven. This test resulted in an MSE of 25.67 which would imply that the output image and the original image are very different. However, MSE is not the best indicator of perceptual similarity. The PSNR when comparing the output image to the original was 32.60 which is comparable to using a lossy image compression algorithm on the original high-resolution image. This means that while the quality of the output image is not perfect, it is on par with a jpeg image. Lastly, the Structural Similarity Index between the output and the original was 0.89. This indicates that the image maintained most of its structural integrity after the upscaling process.

The control method, which involved manually upscaling images using Paint.net, a simple image editing software which uses bicubic interpolation for enhancement [9]. Upscaling images in Paint.net is extremely fast and requires no model training or datasets. Surprisingly, the results of the control were slightly better than the RAISR algorithm. With an upscaling factor of two, the resulting output image had an MSE of only 14.751, a PSNR of 35.953, and a SSIM of 0.927. This trend continued for the 3x and 4x cases where the control continued to outperform the algorithm, even though these differences are essentially undetectable to the human eye.

When the upscaling factor was increased from 2x to 3x or 4x, the amount of error measured by each of the three methods increased. The most notable was the drop in SSIM. Between 2x and 3x, the SSIM dropped from 0.883 to 0.777 and then further dropped to 0.645 at 4x upscaling. When looking at the 4x upscaled image, it is immediately apparent that most of the fine detail was lost. Large sections of the MRI images lose a lot of their texture while maintaining shape.

Figure 1 shows the visual results of RAISR with a x2 upscaling factor. The original image is half the size of the

enhanced version, so the original was zoomed in to be the same size as the enhanced for comparison. As we can see, the enhanced version of the image has smoothed out the pixelation found in the original. The results for testing on upscaling factors of x2, x3, and x4 as well as 3, 5, and 11 patients in the training data can be found in figure 3.

3.5 Alternative Implementation

We attempted to use an alternative implementation of RAISR created by the Jalali-Lab at UCLA [10]. This implementation is designed to be a faster version than the publicly released version of RAISR that was used in this project. While it uses the same method for image super-resolution of learning a set of filters from training data that can then be applied to low resolution images, it employs many specialized coding techniques meant to speed up both training and testing. Just-in-time compilation employing JIT numba is used to speed up all of the Python code [10]. Additionally, parallelized coding libraries are employed in order to reap the benefits of multi-core machines, which speeds up execution significantly as Python normally executes in a single-threaded manner [10]. This gains a significant performance advantage over the implementation used in this project as the publicly released version of the Google RAISR algorithm is not optimized for speed. Our testing showed these performance enhancements resulted in run times an order of magnitude smaller than the alternative implementation.

However, the results were less than desirable. The outputted images were visibly blurry and had a mean square error compared to the original image of 62.07, a PSNR value of 20.793, and a SSIP of 0.459. The output from this implementation compared to the original high-resolution image can be found in figure 2.

We were not able to determine why the performance for this implementation was so inferior, as the documentation suggests that performance should be comparable, if not better, than the implementation used in this project. We initially believed that the problems were arising from machine incompatibility, as the documentation says that the code had been tested on MacOS-X, but we were testing on Windows machine. The program seemed to infinitely create and destroy threads, and was only able to be shut down by performing a full reboot.

We then tested on a machine running MacOS-x. While the program did run successfully, the results were not nearly as good as the alternative implementation. One possible explanation is that we were forced to use different release versions of packages that were required by the program. Some of the packages required release versions that were not compatible with Python3, the python version that RAISR runs on, and we were forced to instead run the program with later versions of those packages.

4 Discussion

4.1 Analysis

As our results show, RAISR is effective at performing super-resolution on MRI data, but it does not seem to perform better than common image upscaling methods. There could be several reasons for this. One is the limited amount of training data. While this project used over 300 MRI images, it came from only 20 patients. Because some patients could not be included in training as they were needed to perform testing, we were limited even further.

However, the limitations on the amount of data may not have been the main problem as the performance of RAISR did not seem to vary much depending on the amount of training data. The reason for this is that although patients had different MRI images, many still had very similar appearances. This means that RAISR would encounter many patches that have very similar characteristics, resulting in many filters that are very similar to one another. Because RAISR uses rotations and mirroring to increase the number of patches it can analyze for a single image, it seems that RAISR can be trained for use on MRI data with relatively few training images.

The main problem seems to be in the algorithm itself. RAISR's strength is in being able to enhance straight edges, and is able to synthesize certain textures, however, RAISR is not as effective at recovering natural high-resolution details [11]. Because this project is centered around MRI images, specifically of the brain, the dataset will not cater to RAISR's strengths. MRI's have very few straight edges and mostly consist of curved edges.

5 Conclusion

Our results show that the Google RAISR algorithm is adequately successful when used to perform super-resolution enhancement on MRI data. One of the greatest benefits to its use is that it requires a relatively small amount of MRI training data when compared to other machine learning tasks. This is shown in figure 3 which shows the insensitivity of performance to the number of patients included in training data.

While this project did not use Google's in-house implementation and therefore was not able to take advantage of the speed that the high-performance implementation would have brought, the speed of the RAISR algorithm and its lightweight use of memory are also advantages.

RAISR does provide visible enhancement and can be used to create super-resolution MRI data, but it does not seem that its performance will be any better than other simpler methods like that used in the control group. This is most likely do the fact that RAISR performs best on images that contain many straight edges and a high variation in texture [11]. MRI images generally do not fit this description as almost all edges are curves and not a great variation in texture in different parts of the image, although there is some.

While RAISR is a viable candidate for super-resolution of MRI data and differences between enhancement using RAISR and the bicubic control group are indistinguishable to the human eye, it seems that there are other algorithms which may perform better. To gain performance better than simple algorithms like that used in our control group, other algorithms such as Google's multi-frame Super Res Zoom will likely be better at as it focuses on recovering natural high-resolution details rather than simple enhancements of visual qualities in the images [11].

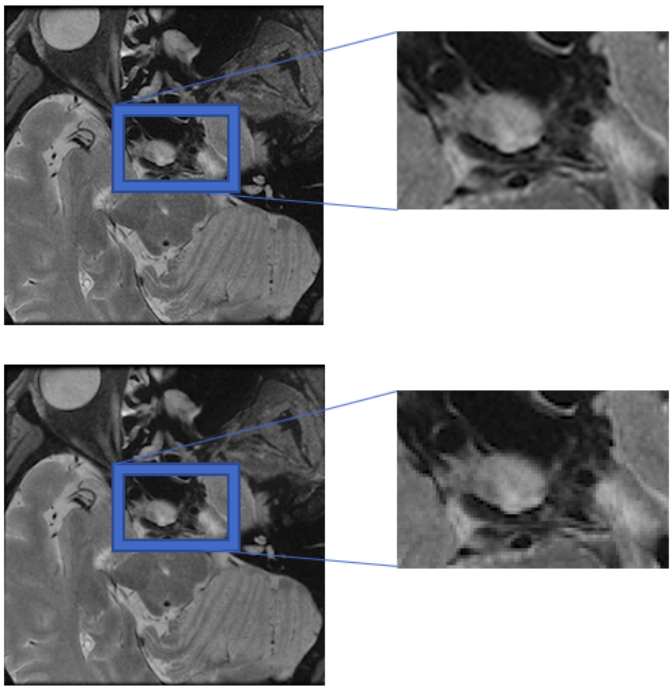

Fig. 1. Comparison of enhanced (top) and original (bottom) with original enlarged to be of the same size. The zoom in on a particular patch (right) shows the enhancement as the original shows pixelation and the enhanced has a much smoother patch

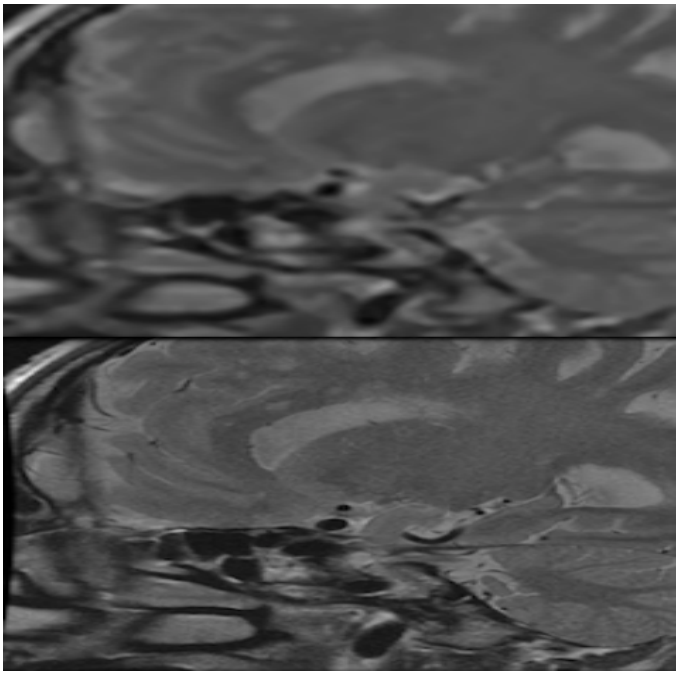

Fig. 2. Results from alternative RAISR implementation. ”Enhanced” (top) is of much worse quality than original (bottom)

Upscale	Test	MSE	PSNR	SSIM
x2	Control	14.751	35.953	0.927
	RAISR, 3 patients	24.728	32.455	0.883
	RAISR, 5 patients	24.878	32.417	0.881
	RAISR, 11 patients	24.700	32.457	0.883
x3	Control	25.377	32.604	0.838
	RAISR, 3 patients	32.937	30.418	0.777
x4	Control	37.745	29.891	0.712
	RAISR, 3 patients	44.934	27.777	0.645

Fig. 3. Results for testing of control (bicubic) and RAISR for x2, x3, x4 upscaling and varying number of patients in training data

REFERENCES

[1] T. Geva, "Magnetic resonance imaging: Historical perspective," *Journal of Cardiovascular Magnetic Resonance*, 2016. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/16869310

[2] A. Ducournau, R. Fablet, C. Pham, and F. Rousseau, "Brain mri super-resolution using deep 3d convolutional networks," *IEEE 14th International Symposium on Biomedical Imaging*, 2017. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7950500

[3] P. Milanfa, "Enhance! raisr sharp images with machine learning," *Google AI Blog*, 2016. [Online]. Available: https://ai.googleblog.com/2016/11/enhance-raisr-sharp-images-with-machine.html

[4] J. Estroff, A. Gholipour, and S. Warfield, "Robust super-resolution volume reconstruction from slice acquisitions: Application to fetal brain mri," *IEEE Transactions on Medical Imaging*, vol. 29, 2010. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/5482022

[5] J. Isidoro, P. Milanfar, and Y. Romano, "Raisr: Rapid and accurate image super resolution," *IEEE Transactions on Computational Imaging*, vol. 3, 2017. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7744595

[6] W. Kurt, "A deeper look at mean squared error," *Count Bayesie*, 2019. [Online]. Available: https://www.countbayesie.com/blog/2019/1/30/a-deeper-look-at-mean-squared-error

[7] "Peak signal-to-noise ratio as an image quality metric," *National Instruments*, 2019. [Online]. Available: http://www.ni.com/en-us/innovations/white-papers/11/peak-signal-to-noise-ratio-as-an-image-quality-metric.html

[8] Z. Wang, A. Bovic, H. Sheikh, and E. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE TRANSACTIONS ON IMAGE PROCESSING*, 2004. [Online]. Available: https://www.cns.nyu.edu/pub/eero/wang03-reprint.pdf

[9] "Paint.net image resize," 2019. [Online]. Available: https://www.getpaint.net/doc/latest/ImageMenu.html

[10] S. He and B. Jalali, "Jalali-lab implementation of raisr algorithm," 2018. [Online]. Available: https://github.com/JalaliLabUCLA/Jalali-Lab-Implementation-of-RAISR

[11] B. Wronski and P. Milanfar, "See better and further with super res zoom on the pixel 3," *Google AI Blog*, 2018. [Online]. Available: https://ai.googleblog.com/2018/10/see-better-and-further-with-super-res.html