Super Resolution Network Fusion

Corey Kurowski, Hershel Millman, Mark Kapron
School of Electrical, Computer, and Energy Engineering
Arizona State University
Tempe, AZ 85281
cjkurows@asu.edu, hmillman@asu.edu, mkapron@asu.edu

Abstract

Super-resolution methods often are not able to recover high frequency components, resulting in blurry or fuzzy images which lack the accurate texture. Recent developments in generative adversarial networks for super-resolution have allowed the recovery of high frequency components, but this comes at a tradeoff of low-frequency information, such as shapes and colors. Whether one is training a network for high frequency component super-resolution or for other super-resolution methods, the amount of training data required is often astronomical, on the order of hundreds of thousands of images. The team's proposed method combines multiple super-resolution architectures to create a hybrid algorithm that outperforms the state of the art in terms of metrics and visual appearance with a fraction of the training images.

1 Introduction and motivation

If every smartphone user in the world today took just one photo and those photos were compiled into a slideshow that advanced one image every second, the slideshow would play for over 85 years nonstop. This is a remarkable way to understand the sheer amount of people in the world with access to a camera, but far more incredible than this, is that this is by far, a low estimate. Not only do billions of people have access to smartphones, but many people have webcams, security cameras, governments have satellites that return very good images, and businesses protect themselves with cameras throughout their locations. With all of these cameras, taking numerous photos and videos every day, it is likely that key events captured will not be the desired quality. Whether it be a recording of a crime being committed, a satellite recon capture, or any other occasion, there is a demand for a way to increase the quality of images and videos post capture. Recently, advances have been made into different methods of approximating a higher-resolution version of a given image. This area of research is termed "super-resolution."

Super-resolution (SR) is a term given to images that have been upscaled to a resolution higher than the original image. The resolution of an image is determined by numerous constraints, including hardware sensitivity, budget, and optical laws of physics. In many applications, including satellite imaging and surveillance, a relatively low resolution (LR) image is the best quality obtainable, which can significantly limit the potential analysis of an image. Additionally, in security camera systems, huge numbers of images must be stored, and the cost of storage space places a limit on the quality of those images. Previously, the only way to obtain high resolution (HR) images was to invest in expensive equipment. This high monetary cost includes computing power, data storage, and physical space. Current research consists of using single or multiple LR images to construct a higher resolution image [1-3]. A super-resolution technique must be able to accomplish two goals: 1) it must be able to extract all information present in the LR image, and 2) it must be able to accurately map that information to a super-resolution image. Using a single-image super-resolution technique requires that technique to be effective at guessing the missing information. A multi-image super

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

resolution technique has the advantage that there is more information about the scene present in the images that the algorithm can utilize to create a higher resolution image. Being able to extract more information from multiple images is trivial, but mapping that information to a super-resolution image is still complex. The effort of this team is devoted towards a single-image super-resolution technique, focusing on optimizing an algorithm which can map extracted information into a super-resolution image.

One of the simplest SR techniques is bicubic upsampling. This involves upscaling by adding pixels which have values chosen to best fit the image gradient. Because this approach is based on a gradient it often leads to blurring effects and loss of sharpness. This produces images that are not photorealistic and can usually be identified as upsampled by a human observer. In order to accomplish more realistic super-resolution, techniques must be used which can take into account more of the information contained in an image. Possibly a more highly desired attribute of a super-resolution method would be its ability to know how many different types of images would be upscaled, as images with different colors, different high-frequency content, or different shapes would need to be upscaled differently. A color LR image of size 256x256x3 is defined by 196,608 numbers, and that image upsampled by a factor of 2 in each direction becomes a 512x512x3 image, which is defined by 786,432 numbers. Trying to create a function which maps such a large input vector to such a large output vector naturally lends itself to machine learning.

Machine learning involves training networks to approximate functions. However, these networks are often very large, and can contain millions of parameters, which can take massive amounts of training data before they can successfully approximate the desired function. The team's work in super-resolution is in the analysis and optimization of machine learning algorithms, so that a hybrid of these methods may be implemented which requires much less data and much less training time.

Presently, there exist several very promising methods to accomplish enhancing an image's quality with super-resolution. Single image techniques often use deep neural networks (DNNs) to reconstruct the high frequency components that are not present in the LR image [1]. On the other hand, another technique of creating an SR image is to use multiple LR images that differ on a sub-pixel level translation [2]. This is known as multi-image super-resolution (MISR). The data from this set of LR images is used to fill in the gaps that are present from a single image. In recent years, many novel methods of both single-image and multi-image SR have been explored. The team took inspiration from a set of these pursuits and worked to combine multiple methods in an effort to enhance the super-resolution quality of the output image.

As will be discussed in detail in the methods and results section, the team attempted several SR enhancement and network fusion methods, but each, until the last, were clearly vain efforts. The first endeavor was: implement the MISR from [2], improve the network's robustness by accommodating translational alteration to the LR inputs, then further improve the quality with a single-image SR on the output image. This approach to the project was quickly abandoned as the pseudo code provided in [2] did not result in a properly reconstructed image, and debugging the implementation fell short. Next, the idea of utilizing a "Pix-2-Pix" generative adversarial network (GAN) to predict the missing information of the HR image was explored. This method was also discarded after several weeks of the team not being able to get the quality of outputs desired. With more tuning and training time, this approach may have yielded promising results. Finally, the act of SR network fusion was pursued. This method consists of using both SRGAN and SRResNet from [4], then creating a fusion of the results that utilized the qualities of each output to produce one objectively better SR output.

2 Methods

2.1 Previous methods

As noted previously, there were three methods in total that the team implemented. The first of which will not be discussed due to the brevity of its pursuit, as well as the lack of results. The second method, however, was deeply explored and will be discussed in full below. In order to provide the method comprehensively though, the concept of a generative adversarial network must be explained first.

Generative Adversarial Networks (GANs) are neural networks that consist of a Generator and a Discriminator. The Generator takes in a vector, usually composed of random numbers, and outputs a

"fake" generated version of whatever it is attempting to learn to produce. The Discriminator attempts to determine whether that sample is a real sample or one that was synthesized by the Generator. In the context of image processing, the Generator creates a "fake" image, and the Discriminator determines whether that image is real or synthesized. The Generator tries its hardest to learn how to trick the Discriminator, and the Discriminator tries its best to not be tricked by the Generator. The term which describes how well this battle is progressing is "adversarial loss," the formula for which is shown in Eq. 1 below.

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[logD(x)] + \mathbb{E}_{z\sim p_z(z)}[log(1-D(G(z)))] \tag{1}$$

By (minimizing / maximizing) this minmax function, the Generator can become very efficient at synthesizing images, and a Discriminator can become very efficient at detecting whether an image is real or fake. This minmax "game" is usually played by first training the Discriminator so that it does very well at determining whether the image is real, and then the Generator gets it chance to trick the Discriminator. For each failed attempt by the Generator, it receives feedback from the Discriminator and it works to improve. As it starts to consistently trick the Discriminator, the Discriminator then begins to improve its detecting skills even further, and the process repeats.

Pix-2-Pix is a GAN that performs image translation from one domain to another. These domains include, but are not limited to, ground truth and silhouette, black and white, and day and night. Instead of using a hand-picked loss function, an architecture was chosen which allows the network to learn the loss function. The setup also uses a conditional GAN, and employs a U-net architecture. The conditional GAN works by allowing both networks, the Discriminator and the Generator, to see the class label while training. The loss function for a conditional GAN is shown in Eq. 2.

$$\pounds_{cGAN}(G,D) = \mathbb{E}_{x,y}[logD(x,y)] + \mathbb{E}_{x,z}[log(1-D(x,G(x,z)))] \tag{2}$$

This allows the Pix-2-Pix network to learn how to generate and discriminate images from multiple domains without having to train multiple sets of networks. The loss function used in these networks is conditional, but also involves L1 distance (Eq. 3) and is shown below in Eq. 4.

$$\pounds_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x,z)\|_1] \tag{3}$$

$$G^* = arg\min_G \max_D \pounds_{cGAN}(G,D) + \lambda \pounds_{L1}(G) \tag{4}$$

L1 distance is preferable to L2 distance because L2 distance tends to cause blurriness in output images. The U-net architecture involves skip connections. This means that instead of one output being produced by the Generator and being analyzed by the discriminator, multiple layers of the Generator are connected to mirroring layers of the Discriminator. This allows the networks in tandem to learn on multiple levels, where there is less of an information bottleneck, and allows the discriminator to learn from the intermediate stages of generation. Pix-2-Pix also makes use of a Markovian Discriminator, which they call PatchGAN. PatchGAN is a Discriminator which analyzes patches of images for structural similarity. This is advantageous because although different domains may be similar in structure on a large scale, on the pixel level, they are often quite different. It is analogous to a loss function that captures style or texture.

With some background on GANs (specifically Pix-2-Pix), the method, which will from here on be referred to as the "edge method," was as follows:

Step 1: The CelebA dataset, consisting of over 204,000 images, is reduced to a more manageable scale of 5000 images for this project. Each of these images is 178x218 pixels (a reasonable size for any number of GANs), and these are deemed the HR images for this method.

Step 2: The entire HR dataset of 5000 images is input into a new dataset creation script that first takes the HR image, downsamples it to a fourth of the size (forming the base LR image), creates sixteen copies of the LR image, shifts each LR copy by one pixel more than the last all the way to, at most, N-1 pixels down and to the right (where N is the downsample factor – 4 here). This yields sixteen LR images that are all offset by a single pixel. Each LR image is then upsampled to the original resolution (with zero padding added post-upscale as an intentional curb toward the GAN)

using a bicubic interpolation scheme, and finally, the images are merged together. The idea here was to simulate a sub-pixel offset, as the downscaled shifts and upscale before merging should have given an average to each pixel. This means the theoretical in-between pixel values should contain additional information about the original pixel value from the HR image. In hindsight, the team determined that a more practical approach would have been to simply take an average of a neighboring range of pixels and created a new image out of it, which would have provided the same theoretical concept.

The dataset creation script had an additional stage beyond the "sub-pixel offset" image generation. This stage was to subtract these generated sub-pixel offset images from the base HR image and obtain the missing "edges." These edge images are not entirely just the high frequency components of the image though, as miscellaneous artifacts exist, which is entirely intentional, as this pursuit aimed at being able to create a SR image based off multiple frames of a video capture (i.e. a slow-motion video could have 16 frames compiled to get a SR image). Figure 1 below shows an example of each of these three images, with the left image being the HR base image (Figure 1A), the middle showing the merged sub-pixel offset image (Figure 1B), and the right displaying the "edge" image (Figure 1C). Each of these two new types of images were generated for every HR base image, forming a new collective dataset that had 5000 sub-pixel offset images and 5000 edge images.

Figure 1: Dataset examples: Showing the HR base image (left), merged subpixel offset image (middle), and the "edge" image (right)

Once again, it should be noted that the zero padding was added post-upscale. This was done to force the generator to recognize that gray and white pixels needed to be added to the black background. More on this exact matter will be discussed later, and it is specifically the leading reason on why this method was stopped.

Step 3: The Pix-2-Pix GAN is ready to train at this stage. The input into the GAN is a sub-pixel offset image (taken from the sub-dataset that Figure 1B is from) and the desired output is the edge image (taken from the sub-dataset that Figure 1C is from). Training is done for 200 epochs with a batch size of 1 and a learning rate of 0.0002. The Adam optimizer is used here too with β_1 being 0.9, β_2 being 0.999, and epsilon set to $1x10^{-8}$. Finally, the learning rate would start an exponential decay at 100 epochs and decay to zero at the end of the 200^{th} epic.

Step 4: The edge output of the GAN would be added to the sub-pixel offset image that served as the input of the GAN, and the resulting image (the HR reconstruction image), theoretically, should closely match the HR base image. If, subjectively, the HR reconstruction was visually appeasing, it could be subtracted from the HR base image to obtain a difference image (with total pixel value of the difference image being a gauge on performance), or various metrics could be tested (e.g. PSNR, SSID, ect.).

For a simple interpretation of this method, the diagram in Figure 2 shows each of the steps described above.

As noted, this method was stopped after several weeks of parameter tuning and training. It was clear that the team's inexperience with GANs, as well as the specific task the GAN was asked to perform, was resulting in images that were not even close to increasing the resolution of the sub-pixel offset images. In fact, it appeared that the results were of lower quality than that being input into the GAN. The generator consistently learned that the output of completely black images would fool the discriminator roughly 25% of the time. This progressed to the generator being able to fool the discriminator even more often by roughly estimating the shape of the edges and essentially applying a very large gaussian blur kernel on the edge to just have a gradient-like edge image. The team was unable to get the discriminator to learn that these types of generated images were fake. If this hurdle

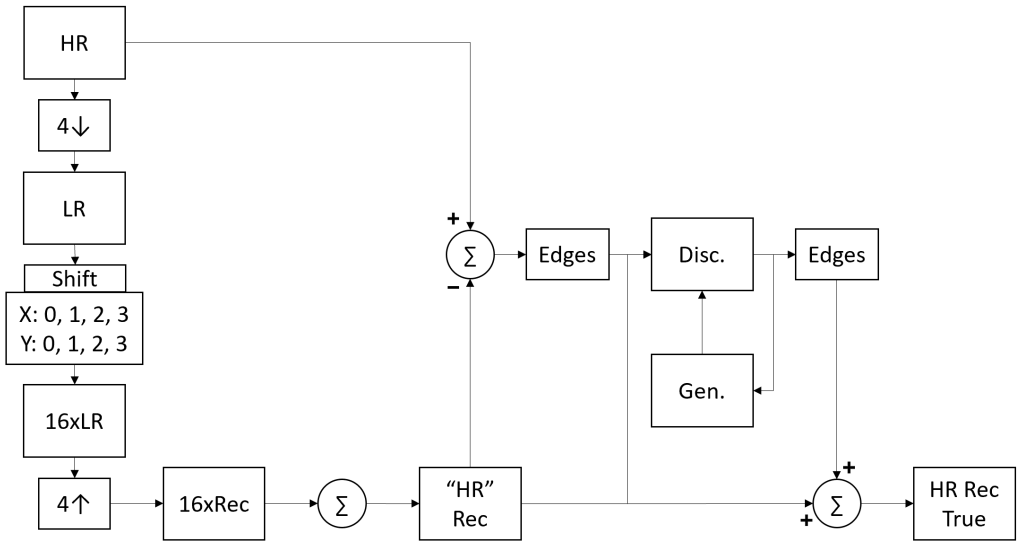


Figure 2: Edge method flow diagram

could have been overcome, perhaps this method would have worked. The better approach would be to input the edges to output the sub-pixel offset images, but in the efforts of wanting to be able to merge video frames, this approach would be pointless, as those frames would not just be edges, and no access to the HR images would be available.

2.2 Current method

In the efforts of completing the "edge method," the team came across numerous other super-resolution GANs that all seemed quite promising. Each of these outside GANs showed great promise, but none seemed to entirely work great at every aspect of super-resolution. This non-ideal set of results, as well as the work done in generating the dataset from the previous method, inspired the current approach that the team has been working at. This method is what the team considers to be a proof-of-concept network fusion. In order to fully describe the efforts, convolution neural networks (CNNs) must be defined, just as GANs were previously.

The use of a convolutional neural network (CNN) is advantageous for applications in image processing. Similar to traditional neural networks, a CNN is constructed of layers connecting the input and output through sets of weighted parameters. The novelty of using convolutional layers is that the network can capture spatial features and patterns, which does not happen with a flattened input. Furthermore, stacking convolutional layers together to make a deeper network was seen to provide better results but performance would plateau after a long enough training time. Initially, the solution to this problem was to continue to stack layers until the desired performance was reached. When training times of these deep networks increased, the gradients of the loss function would approach zero and performance would be stagnant or degrade rapidly. Recent research introduced the ResNet architecture that provided a solution to the vanishing gradient problems by introducing an identity mapping. This interconnection of layers would allow the network to only need to learn the residual information while passing through the known data. Less deep ResNet architectures continuously out performed deeper CNNs, establishing ResNet as a state-of-the-art design for image processing applications. With the understanding of CNNs and ResNet, an application specific neural network design was constructed for Super Resolution.

The specific method for Super Resolution within this project seeks to utilize a CNN architecture that is built specifically for the task of single-image super resolution. There exists a trade-off between the accuracy and quality of the results and the training time needed to obtain these results. Creating a deeper network offers a more dynamic range of model maps that in turn provide higher quality outputs. A consequence of having this deeper framework is the significant increase of parameters in the network, which in turn requires a longer training time to minimize the network's loss function.

Methods to combat the longer training time which maintaining the high accuracy outputs have been researched and produced novel discoveries. The introduction of residual blocks and skip-connections have proven to decrease the training required for deep network designs. These blocks achieve a better result by preventing the network from degrading through feeding forward an identity mapping and having the network learn the residual information. When employing this type of network for a super resolution application, the low resolution input data is preserved within these identity mappings.

With the establishment as to how the networks actually function, the current approach can now be described in a similar fashion to how the edge method was. First though, it is important to give insight on how the team opted to choose to fuse SRResNet and SRGAN. Throughout the extensive literature review, as well as parallel research while implementing the edge method, it was clear that two specific networks were doing very well at super-resolution. These were SRResNet and SRGAN, but each clearly had pitfalls, and these poor performing attributes were even emphasized in [4]. Further elaborated on in [4], the aspects that the networks could not get correct were not a matter of training, but rather the actual structure of the network.

In SRResNet, the task of filling in the missing information within the framework first starts with upsampling the low resolution input with bicubic interpolation to provide a starting place. For a simple network, an MSE loss function (Eq. 5) is typically employed to further estimate the missing information.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \widehat{y}_i)^2 \tag{5}$$

Unfortunately, this approach leads to averaging the pixel values throughout the image which creates a smooth appearing output with missing high frequency data. In brief, the output of the network tends to be essentially a low-pass filtered image of the ground truth. More advanced approaches utilize GANs to learn and interpolate the missing high frequency data inherent in higher resolution images. Additionally, some techniques aim to use the feature space to compute loss through calculating Euclidean distances while training a GAN. The combination of a ResNet architecture and a GAN in [4] was found to have successful results but required significant training data and time.

SRGAN is a GAN to translate images from low resolution to super-resolution. It uses a perceptual loss function which attempts to emulate how a human would perceive images. Perceptual loss combines the MSE loss in Eq. 5 (called content loss), as well as the traditional adversarial loss (Eq. 1), to form Eq. 6.

$$l^{SR} = l_X^{SR} + 10^{-3} l_{Gen}^{SR} \tag{6}$$

A residual network is one which allows connections between layers, usually called "skip connections." These extra connections allow layers to share information and can help prevent important information from being lost as it progresses through the layers of the network. The Generator is actually built on top of SRResNet. This is done as a form of cross-entropy, essentially allowing the generator to start from an image much closer to the ground truth than just starting with random noise. In this sense, SRGAN expands on the capabilities of SRResNet by building the Generator network on top of SRResNet. The Discriminator network in SRGAN uses a pre-trained 19-layer VGG network for feature detection. These features allow for the perceptual loss function, meaning that instead of just being scored on absolute error, the generator's output is scored based on its similarity in feature space. This is highly advantageous for photorealistic super resolution, as the human eye is much more sensitive to features such as edges and dots than it is to slight differences in coloration or small shifts in pixels. In other words, the human eye has a severe sensitivity for high-frequency components of images, which SRGAN does very well at replicating.

As noted above, SRResNet does quite well at constructing the general shapes, colors, and base structure of the ground truth image (essentially forming the low-pass filtered ground truth image) from a LR input. SRGAN uses this base image to start, and it works to improve the edges and high-frequency parts of the image, but in doing so, loses some precision in shapes, color, and general structure when comparing it to both the ground truth and output of SRResNet. In what is an intuitive approach, the team saw the opportunity to utilize the best performing parts of both of these networks, combine them in a clean fashion [while dropping their poor attributes], and theoretically, improve the

total output quality. This proof-of-concept network fusion performed very well, and importantly, was very minimal in terms of training. The steps of the network fusion were as follows:

Step 1: In the first stage of the network fusion method, a practical approach to training was first established. Instead of training SRResNet, then SRGAN, and then maximizing qualities of each network for fusion, the combination of all sections was done. That is, SRResNet and SRGAN were fused into one parallel network, whilst the outputs were fused with a set of well performing parameters. Two important notes regarding this parallel network arrangement. First, since SRGAN is built on top of SRResNet, both pre-trained and untrained versions of the network were trained, and improvements were minimal in contrasting the two versions. Since this is a proof-of-concept fusion, the latter SRGAN results were used alongside the [originally] untrained SRResNet results. As for the second note, ideally, the fusion should be done with a neural network too, but the team sought to establish the realistic possibility of isolating quality components from each network. For this reason, a signal processing mindset was adopted in the isolation of these components. A network would be able to learn this isolation technique and take it even further to maximize the structural similarity index metric (SSIM), which will be how the team later analyzes the success of the fusion.

Step 2: After the networks were trained in parallel, the latter part of the script is implemented, which seeks to isolate the aforementioned qualities of each network's outputs. To do this, first the attributes are enhanced and a mute is applied to their poor performing elements too. For SRResNet, since it can be thought of as a low-passed filtered version of the ground truth with some artifacts in the high-frequencies, the team sought to eliminate the artifacts with a gaussian filter (Eq. 7).

$$g(i,j) = \sum_{k,l} f(i+k, j+l)h(k,l) \tag{7}$$

Where $h(k,l)$ is the "kernel" of the filter, and is defined as follows:

$$h(k,l) = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

With the artifacts of the SRResNet output minimal, and further establishing the desire for just the low-frequency elements from it, SRGAN can now be "enhanced." For SRGAN, since the team sought the higher frequency parts of the image (e.g. edges, dots, and the general "sharp" parts of the image), a "sharpening" filter can be applied by using Eq. 7, but using a kernel such as:

$$h(k,l) = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

This will, however, not eliminate the low frequency parts of the image. A quick way to do this would be to apply a high-pass filter to the enhanced "SRGAN" output. This was the initial approach the team took, but it was clear that SRGAN was actually establishing quality transitions from high-frequency to low-frequency parts of the image. That is, SRGAN's outputs had pixels around the edges (on average at 10 pixels out from edges) that were more accurate to the ground truth than SRResNet. To take advantage of these pixels, it was decided that a mask would be applied to the enhanced SRGAN output. This mask took, at first, just the set of the high-frequency components of SRGAN, but to account for the 10 or so pixels around the edges, dilation, a morphological operation (typically used to eliminate "pepper" noise), was used for a single iteration. This essentially expanded the width of all edges in the mask. To allow for a smooth transition from the pixels in SRGAN to SRResNet, the mask itself was also low-pass filtered [with a gaussian blur].

Step 3: With the best qualities from SRResNet and SRGAN enhanced and isolated, the actual fusion could take place. This was done with the composite function from Python's Pillow (PIL) library, which simply allows the combination of two images with a transparency mask. Which means that the exact desired result can be done, where the mask can be applied to the enhanced SRGAN output, and those isolated pixels can be put directly onto the enhanced SRResNet output (i.e. the high-frequency components are added to the low-frequency components).

Step 4: This step was only done initially, but it is important to note that here is where optimization was done. Parameters such as each gaussian blur's radius, the sharpening kernel, the mask dilation, and others were optimized to find an all-around well performing set. This, once again, was merely because it was a proof-of-concept to successfully fuse these networks' outputs with emphasis on their defining qualities. As will be discussed more below, a neural network would be much more ideal to implement the entire output fusion.

The network described above was constructed by first establishing the data set. The high-resolution images that are taken to be ground truth were first sent through a Gaussian filter and then down sampled to construct the low-resolution inputs. This process ensures an objective baseline for network performance. These low-resolution input images are then sent through a feed-forward CNN generator network to be then compared to an adversarial network as a component of SRGAN. Figure 3 shows the architecture of the generator network. It can be seen that the ResNet framework ensures the existing low-resolution data is mapped through the network by skip connections to avoid the problem of degradation after many training epochs. As a result, the network only needs to learn to generate and adjust the uncertain pixels after upsampling the low-resolution input. By focusing the learning on the residual information, the network can more effectively approximate the high-frequency information seen in the ground truth that would be missing in a traditional CNN approach.

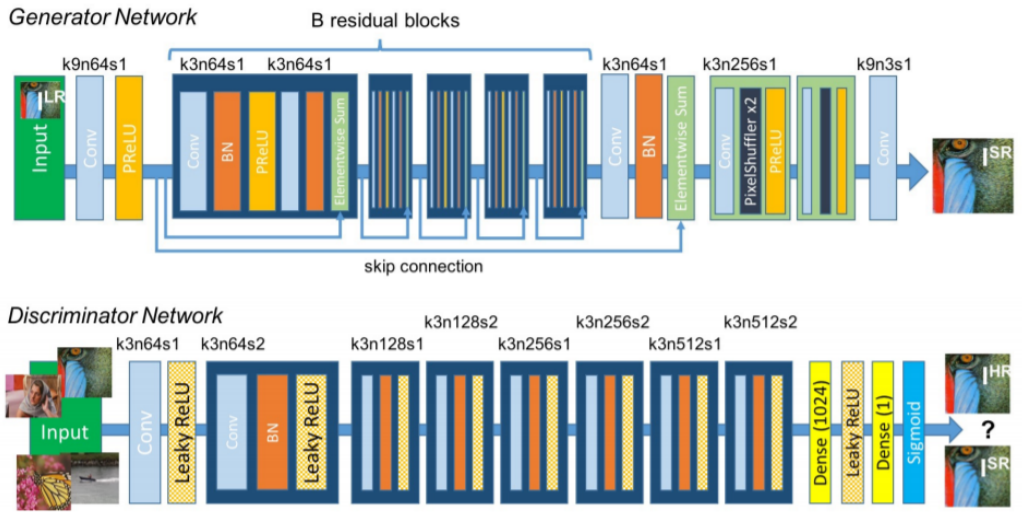


Figure 3: Architecture of Generator and Discriminator Network with corresponding kernel size (k), number of feature maps (n) and stride (s) indicated for each convolutional layer [4]

3 Results

Well the results will heavily focus on the proof-of-concept network fusion method, it will start by showing some of the results that the edge method obtained.

3.1 Edge method

In the edge method, as was established previously, the aim was to allow the Pix-2-Pix GAN to generate the "edges" that were missing from the LR inputs. Once again, these were not entirely just edges, but rather any pixels that were simply different from the ground truth (HR base image). These were, however, mostly the edges or high-frequency components. Figure 1 showed the training images that the Pix-2-Pix GAN was trained on. Figure 4 shows an example of the network's output at 200 epochs (left) versus the actual desired edge (middle), with the LR input image also shown (right). The training had a batch size of 1 and a learning rate of 0.0002. The Adam optimizer is used with β_1 being 0.9, β_2 being 0.999, and epsilon set to $1x10^{-8}$. Finally, the learning rate would start an exponential decay at 100 epochs and decay to zero at the end of the 200^{th} epic.

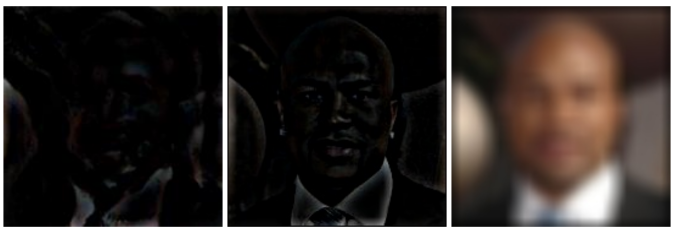


Figure 4: Edge method results: network's output at 200 epochs (left) vs. the actual desired edge (middle) vs. the LR input image (right)

Analyzing the results here, it is quite clear that the network was struggling to obtain the exact "edges" that the team was hoping the Generator could generate. The collar is about the only part of the image that can partially be made out, but only really due to having the true edges right along side. These results, with no true improvement made it easy for the team to realize that a new method was needed to make a contribution to the arena of super-resolution.

3.2 Network fusion - current method

With the team's proof-of-concept network fusion, very good results were obtained. When looking at the images as they compare to a ground truth image, there are several metrics that can be used. Common signal processing metrics, including the MSE and peak signal to noise ratio (PSNR), are not representative of how the human eye interprets an image though. That is, these metrics are not ideal for grading the visual quality of the image itself. When looking at the success of the super-resolution network fusion image output, the team sought for an analytical representation of how a person might grade the output in comparison to the ground truth image. Two specific metrics do exactly this: Mean Opinion Score (MOS) and structural similarity index metric (SSIM). The mean opinion score is taking a subjective score from many individuals where they are each rating the quality of the output compared to the ground truth. This obviously requires a large survey size in order to get an accurate analytical score. The SSIM does the exact task that is desired though, where it looks to give a numeric value to an image compared to a ground truth. This value will range between 0 and 1, with 1 being identical to the ground truth. This is done by looking at the luminance, contrast, and structure of the images with the following equation:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{8}$$

Where μ_x is the average of x, μ_y is the average of y, σ_x is the variance of x, σ_y is the variance of y, σ_{xy} is the covariance of x and y, c_1 and c_2 are two variables that stabilize division with a weak denominator, L is the dynamic range of the pixels, and $k_1 = 0.01$ and $k_2 = 0.03$ by default.

With this metric, the actual success of the network fusion can be objectively analyzed.

Below, in Figures 5 and 6, the results of the proof-of-concept network fusion can be seen. For each, the left image is the ground truth, the second from the left is the network fusion method, the middle is SRGAN, the second from the right is SRResNet, and the right image is a bicubic upscale. The network fusion is done with the following parameters: gaussian blur on SRResNet with a radius of 4, sharpening of SRGAN with the sharpen filter from the PIL library, a mask developed from subtracting a gaussian blurred (with a radius of 4) enhanced SRGAN image from the enhanced SRGAN base image, the mask was dilated once with a 5x5 kernel, and finally, the mask was blurred with a gaussian blur with a radius of 9.

As can be seen above, the SSIM in Figure 5 was improved 12% and in Figure 6, 9%. This trend held true in all results, with the SSIM improvement ranging from 5% to 12%. These are very good improvements, and it shows that even with just minimal training, the network fusion method can come very close to replicating the ground truth image. With a neural network, it is likely that this can be improved further, as various pixel correlations between the optimal features in the SRResNet and SRGAN outputs can be learned. Looking at the SRGAN outputs, between the toucan's feet and also on the lady's cheek, there is what is known as a "hotspot." This takes away from the SSIM in

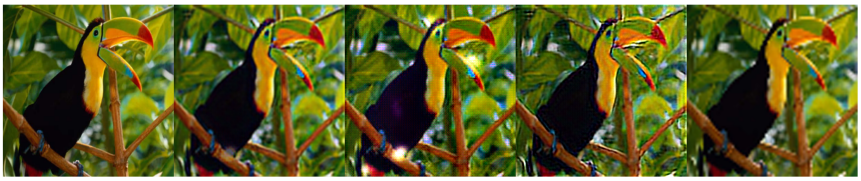


Figure 5: Fusion method results example 1

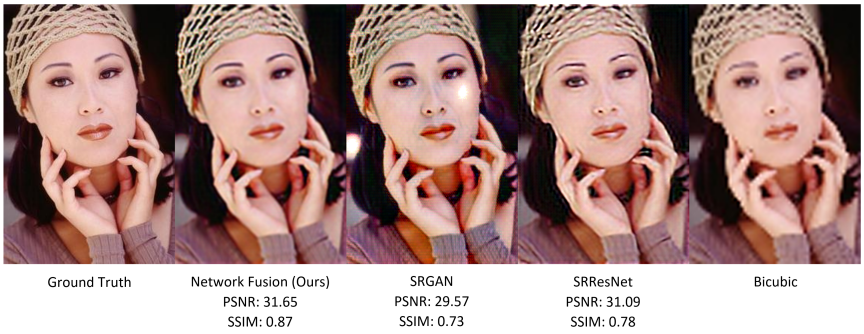


Figure 6: Fusion method results example 2

both images, and it is a cause from how the SRGAN implements what is called a "pixel shuffle" to attempt to generate better results. In SRResNet, it is clear that the edges are a weakness in the image, but the colors and structure are very good. Finally, in the bicubic interpolation upscale, the image is simply choppy in nature, which is a major pitfall in how the interpolation is performed. More results and the masks used to isolate the higher frequency components can be found in Appendix A and B, respectively. To further showcase these weaknesses in the SRGAN and SRResNet outputs, Figure 7 highlights a boy's cheek and eye.

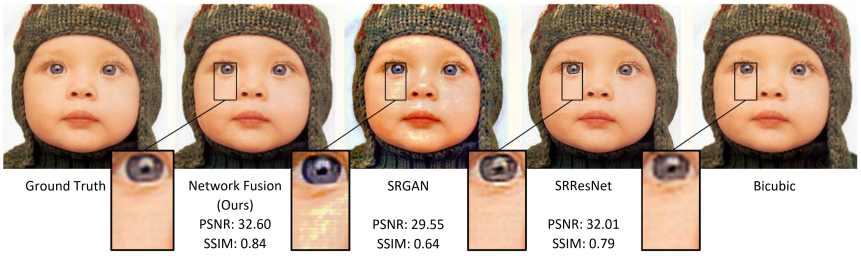


Figure 7: Fusion method results example 3

The cheek itself is smooth, which is a strong spot for SRResNet, but the eye must have definition, which severely hurts the image. SRGAN on the flipside shows great promise on the eye's definition, but it has artifacts on the cheek. When looking at the team's network fusion, it can be easily seen that the eye has much better definition when compared to SRResNet (but slightly worse than SRGAN) and also does not have artifacts on the cheek like those seen in SRGAN. Well a bit of definition is sacrificed in the network fusion, the negation of artifacts while maintaining most of the high frequencies is a careful tradeoff made while optimizing the parameters of the network fusion. With more parameter tuning or with the implementation of the network fusion neural network discussed previously, the results are very likely to improve even more.

4 Discussion

From the results above, the Network Fusion approach outperforms both SRResNet and SRGAN in terms of quantifiable metrics. When looking at the outputs of the SRResNet alone, the network does a decent job of recovering the general shapes of images but fails to recover the high frequency components accurately. Additionally, the SRGAN detects and fills in the edges of the network but introduces image artifacts that create patches of unrealistic colors and textures. To achieve better performance in these two networks, a longer training time is required with the addition of a larger training dataset. Unfortunately, when training SRGAN for a longer period of time, the image artifacts that are seen in the results persist.

To produce visually realistic Super Resolution images, the SRResNet and SRGAN need to individually train for an extensive amount of time. Extending the training time and requiring a large training set becomes very costly in real world applications that have restrictive budgets. It is shown that the Network Fusion approach significantly reduces the cost to the user and provides an affordable and practical solution for Super Resolution imaging. In many situations the reduction of cost comes with the trade-off of performance but this is not the case for the Network Fusion architecture. To expand upon and further compare the results of the different networks, the MOS of the output images can be collected and compared to see how the networks perform in terms of human perception. Additionally, the Network Fusion could be implemented on the networks that were trained for a longer period of time to test the upper limit of performance given that training cost is not restrictive.

5 Conclusion

The need for robust and accurate super resolution imaging persists, as does the need for this training to be efficient in both time and resources. Several methods of accomplishing this task were investigated in this project. The final method was a product of the lessons learned from each shortcoming and a deeper understanding of network architectures. The results of this project showed that initial training of the SRResNet provided a good starting point to act as the input of the SRGAN. When compared the network fusion to these super resolution networks, the fusion method outperforms in terms of quantifiable metrics such as PSNR and SSIM, with the SSIM improving anywhere from 5% to 12%. The most significant result from this project is not necessarily that this network outperformed other state-of-the-art methods, but rather it did so with only a fraction of the training time and data. The super resolution network fusion approach accomplishes the goal of high accuracy and low-cost super resolution without any notable trade-offs. In the future, it is the team's desire to fully implement a network to perform the network fusion itself, and from there, maximize the performance while minimizing the training.

References

[1] W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang, "Fast and accurate image super-resolution with deep laplacian pyramid networks," *arXiv:1710.01992 [cs]*, Oct. 2017.

[2] J. Wu, T. Yue, Q. Shen, X. Cao, and Z. Ma, "Multiple-image super resolution using both reconstruction optimization and deep neural network," *2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2017.

[3] L. Yue, H. Shen, J. Li, Q. Yuan, H. Zhang, and L. Zhang, "Image super-resolution: The techniques, applications, and future," *Signal Processing*, vol. 128, pp. 389–408, Nov. 2016.

[4] C. Ledig et al., "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network," *arXiv:1609.04802 [cs, stat]*, May 2017.

6 Appendix A

Further results from the network fusion method are shown below, with the same image scheme as shown in Figures 5-7, but with the removal of the bicubic upscale interpolation on the far right. Those labeled “ResNet” are SRResNet’s outputs.

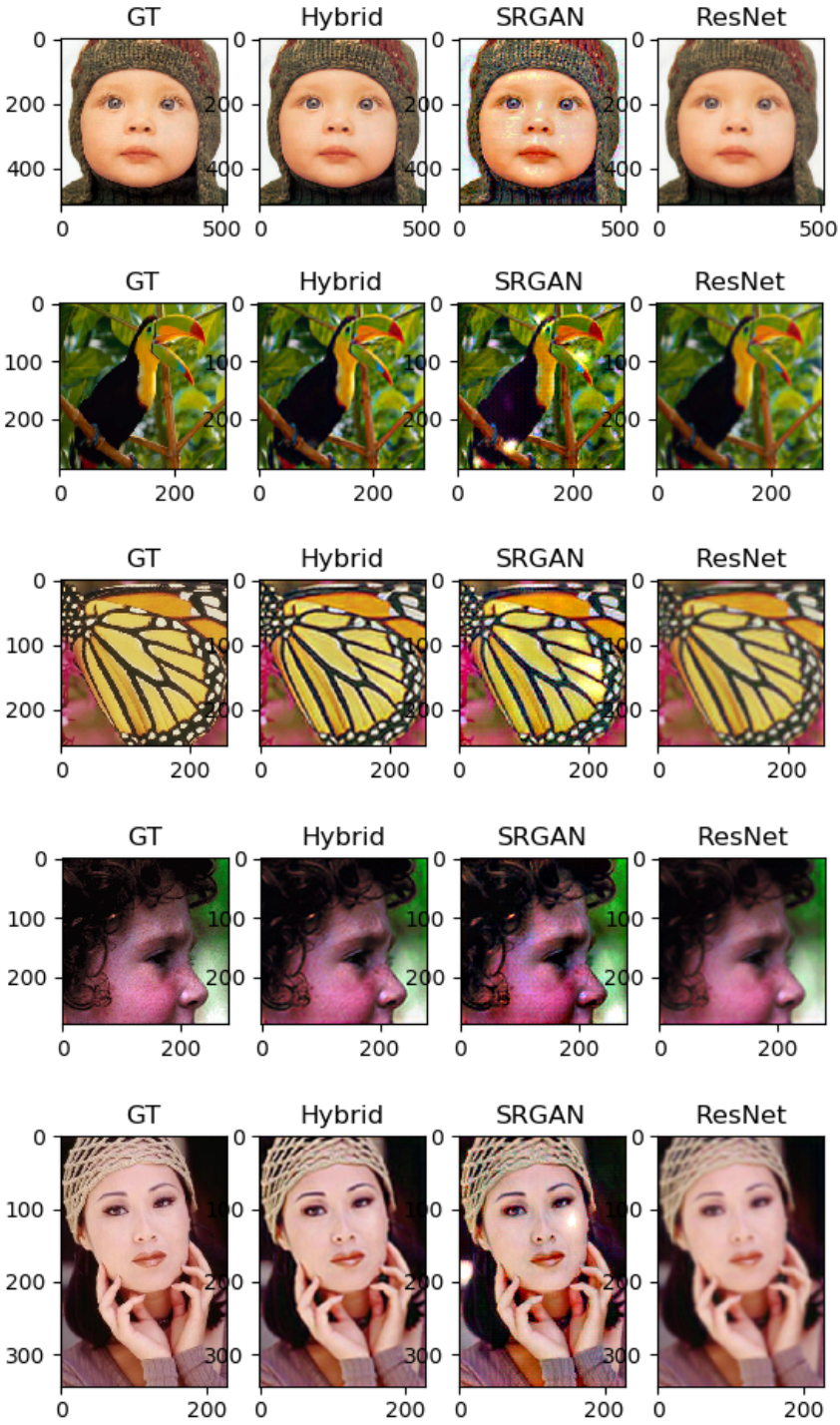


Figure 8: Fusion method addition results: Examples 1-5

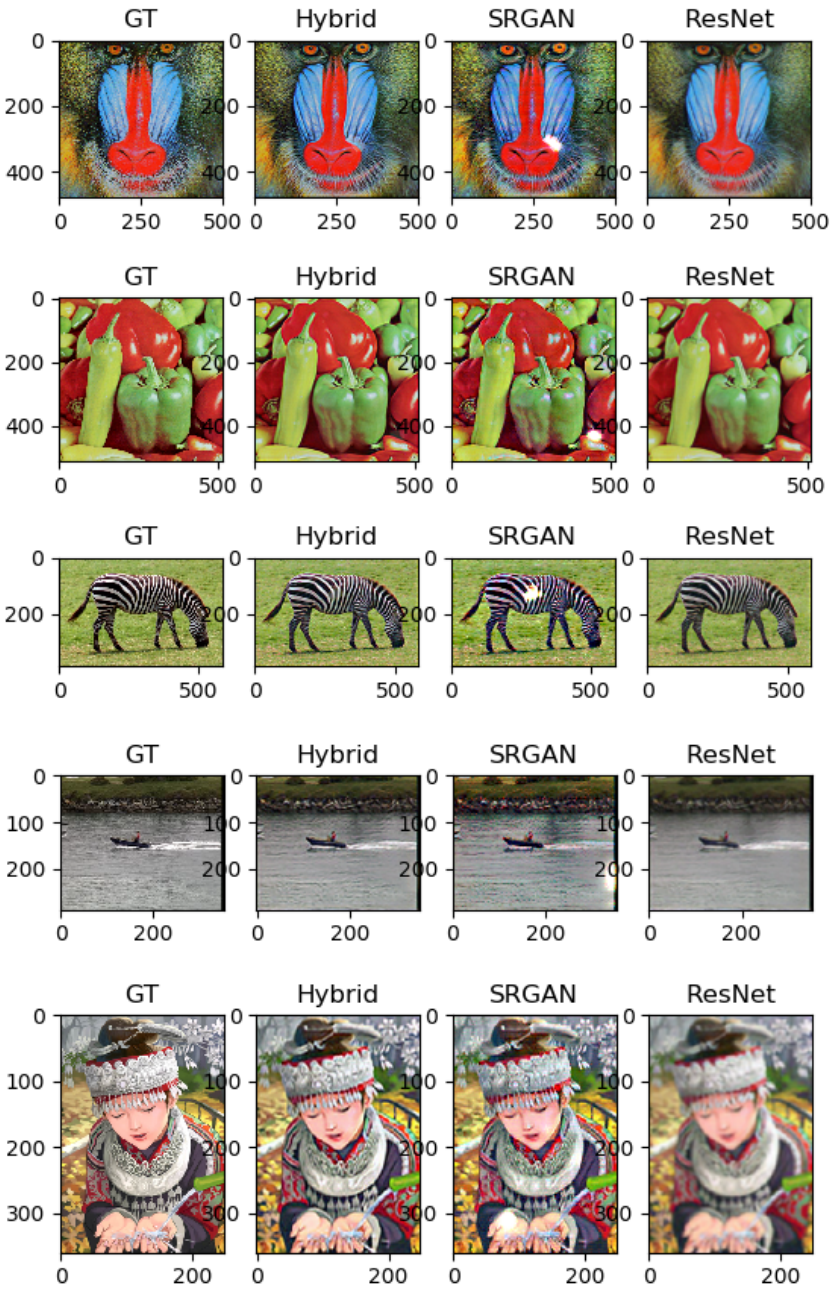


Figure 9: Fusion method addition results: Examples 6-10

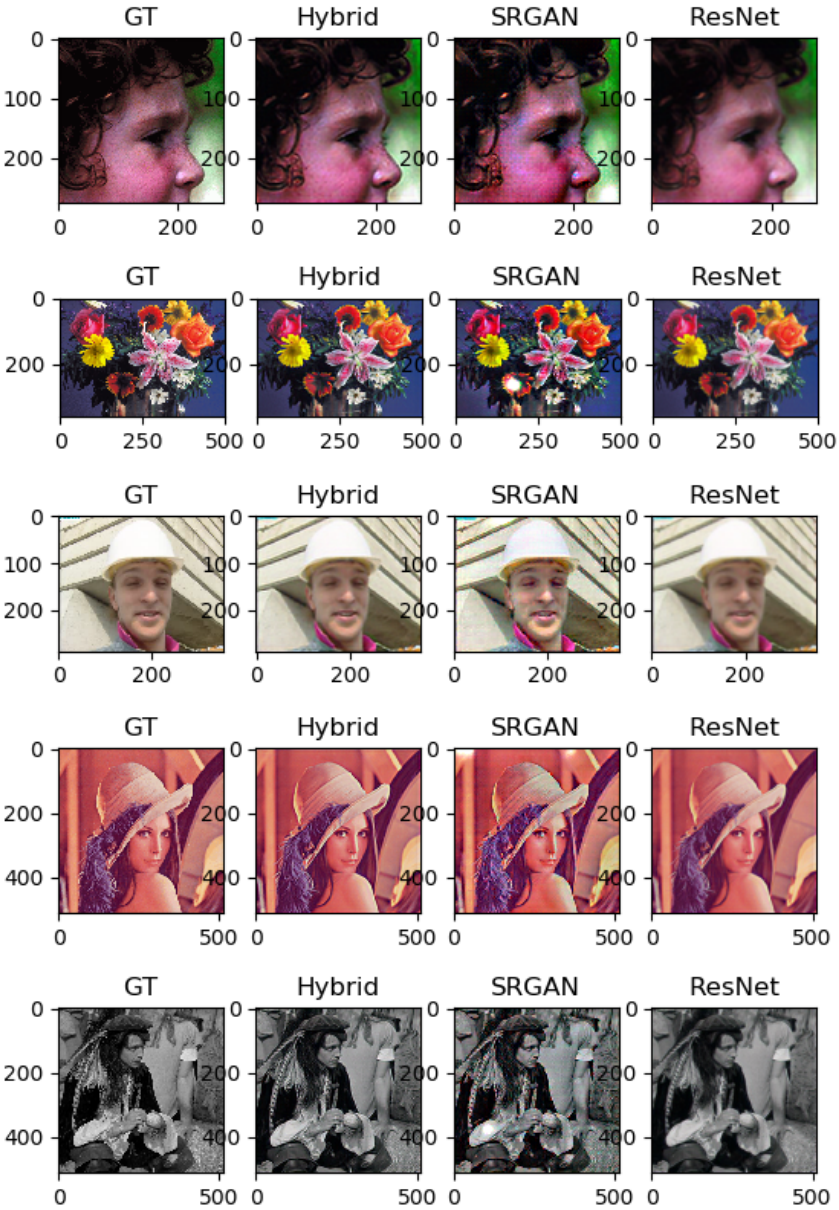
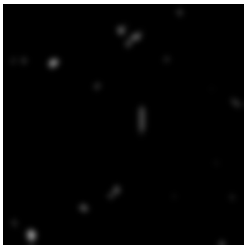


Figure 10: Fusion method addition results: Examples 11-15

7 Appendix B

Below are the masks used for each of the fifteen images that the network fusion was tested on. These masks were used to isolate the pixels from SRGAN (mostly high frequencies).

(a) Mask 1



(b) Mask 2

(c) Mask 3

(d) Mask 4

(e) Mask 5

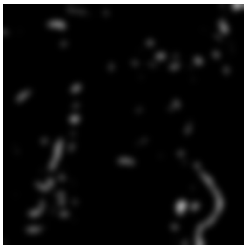
(f) Mask 6

(g) Mask 7

(h) Mask 8

(i) Mask 9

(j) Mask 10



(k) Mask 11

(l) Mask 12

(m) Mask 13

(n) Mask 14

(o) Mask 15