# Final Project: Super resolution using a convolutional neural network

**HENRY MORRIS,**[1]

[1]*Dartmouth College, Hanover, NH 03755*

In this project, I explore the applicability of using a convolutional neural network to perform super-resolution imaging on 20,000 grayscale images from the STL-10 dataset. With all the images originally being 96 by 96 in pixel size, I down sampled them by a scale of 2 and by a scale of 3, before then with bicubic interpolation up sampled the images back to 96 by 96, which cause the images to have a lower resolution. In the experiments, we also introduce various levels of gaussian noise to these upscaled, lower resolution images. Ultimately, using an 80/10/10 training, validation, and testing split with a U-net architecture for my DNN, I found that the super-resolution performed here did a reasonably well job in reconstructing the ground truth images as these images do captures the overall shape and structure of the original as well as recovering some of the fine details not present in the initial guesses. As expected, the scale of down sampling and the amount of noise added are inversely related to the accuracy of the reconstructed images.

## Introduction

Single image super-resolution (SR) is the set of techniques aiming to solve the classic problem in computer image processing of recovering a high-resolution image from a single low-resolution image. Its importance cannot be undersold as it has applications in a multitude of fields including medical imaging, surveillance, satellite imagery, and photography to name just a few. The SR problem is trivially ill-posed as the scarcity of finer detailed information in a low-resolution image leads to an infinite number of possibly solutions. Up until recently, researchers have employed exclusively classical methods to achieve image enhancements such as the usage of priors with Bayesian probability or a frequency domain approach that utilizes Fourier transforms. However, these approaches are limited as they rely on assumptions about the images in the first place and thus are often inadequate when it comes to handling complex, real-world scenarios. However, the rise of deep learning in tandem with a public availability of large datasets has revolutionized the field, leading to the widespread use of convolutional neural networks (CNN) which are able to capture and learn complex patterns within images. In other words, CNNs learns to map low-resolution inputs to high-resolution outputs through a series of convolutional layers, enabling it to discern and recreate fine details that are not initially present in the lower-resolution images. The first known published paper using a super-resolution convolutional neural network (SRCNN) was published in 2014 by Dong et al.[1] Since then, research in this field has exploded and a variety of other architectural structures using neural networks have been introduced which have continued to push the boundaries of SR.

A convolutional neural network is hierarchical in nature which allows it to learn and capture different features at different levels of abstraction, i.e. as different layers. An example of one of these networks can be seen in figure 1. Here, we see the network consists of a contracting path (left side) and an expansive path (right side), giving us the U-shape from which the name U-net architecture comes from. The contracting path consists of the repeated application of two 3 by 3 convolutions, each followed by a rectified linear unit (ReLU) as the activation function, and a 2 by 2 max pooling operation with stride 2 for down sampling. At each down sampling step, we double the number of feature channels. On the expansive path, we are in essence going backwards to get back to where we started. Every step consists of up sampling the feature map followed by a 2 by 2 convolution that halves the number of feature channels, a concatenation with the correspondingly cropped feature map from the contracting path, and two 3x3 convolutions, each followed by a ReLU. At the final layer a 1x1 convolution is used to map each 64- component feature vector to the desired number of classes. In total the network has 23 convolutional layers.

In this report and throughout the design process, I attempted to recreate a version of the CNN introduced by Dong as well taking inspiration from the CNNs introduced more recently that attempt to improve on robustness as well as accuracy.[3,4] However, due to computational efficiency issues, many of the more complicated features I added were removed so I could run my code on Google Colab in a reasonable time. The final architecture I settled with is very similar to the one in figure 1. Like it, I have 2 convolutions at each layer but to improve on computational time even more, I reduced the number of layers, only going up to 512 by 512 instead of 1024 by 1024.

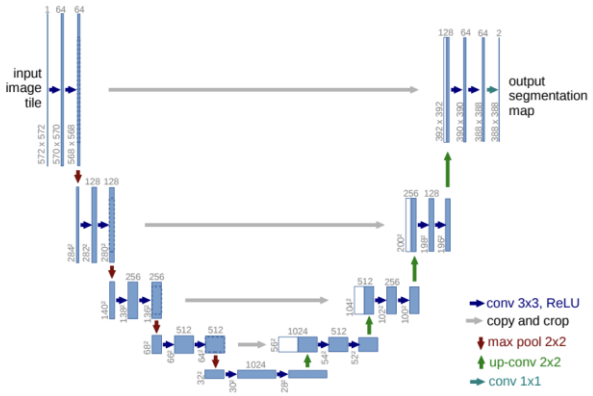


Fig. 1. U-net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations[2]

## Experimental Results

For the final project, I created 8 image datasets, each derived originally from the 20,000 grayscale images in the STL-10 dataset, which would then be used to train their own SRCNN function that maps the low-resolution images to high-resolution images. The 8 image datasets were divided into 2 subsets, one being down sampled by a scale of 2 and the other being down sampled by a scale of 3. Within each of these subsets, different levels of Gaussian noise were added to examine the effects noise would have in the reconstruction process. In Table 1, the numerical results of the experiments can be seen. In figure 2, we have the input and output images of the same randomly chosen image from the test data for each dataset.

In terms of the design of the experiment, some choices were made due to time or computational limitations that may warrant changes in further work. For example, it may have been helpful to have the broader training datasets that contain low-resolution images which each have been toyed with differently, i.e. different images in the training dataset would have different levels of noises and may also be down sampled at different scales. By doing this, we could have built a function whose inputs could be any low-resolution grey-scale image. On that note, it may also have been useful to perform these experiments with RGB-colored images instead of grey-scale images.

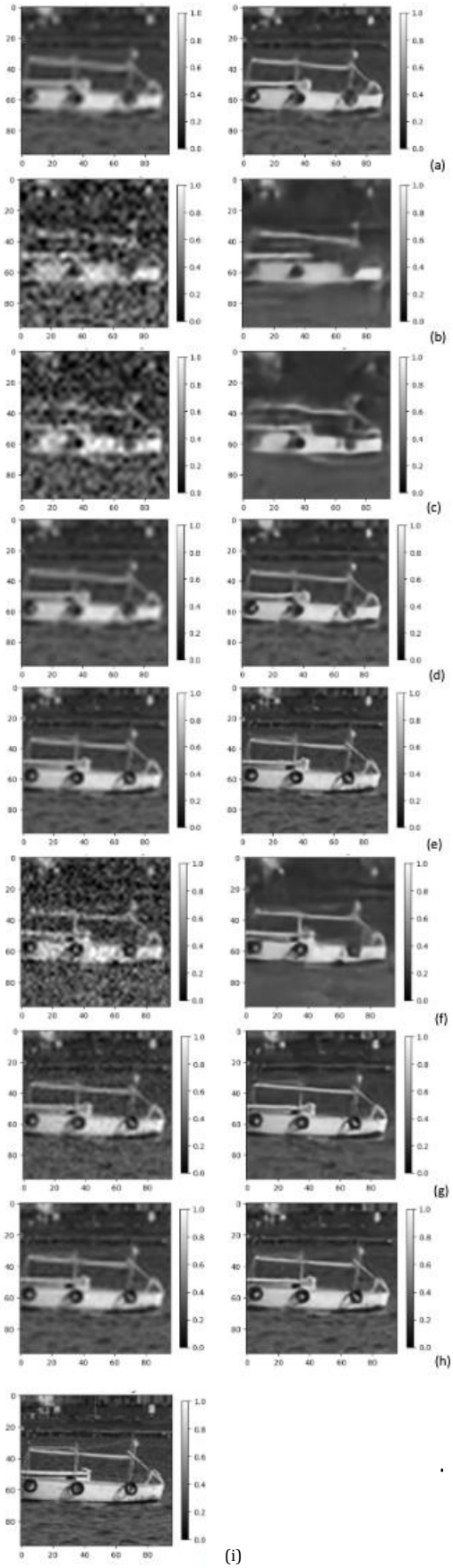


Fig. 2. The graphs to the right are the initial guesses and predicted images for the image indexed with 42 in the test dataset. (a) shows Image Dataset #1, (b) shows Image Dataset #2, (c) shows Image Dataset #3, (d) shows Image Dataset #4, (e) shows Image Dataset #5, (f) shows Image Dataset #6, (g) shows Image Dataset #7, (h) shows Image Dataset #8, and (i) is the ground truth image.

**Table 1.** Image dataset #1-4 have been down sampled by scale of 3, with #2-4 also having different levels of Gaussian noise added. Image dataset #5-8 have been down sampled by scale of 2, with #6-8 also having different levels of Gaussian noise added.

| Image dataset | SNR to ground truth | Test MSE |
|---|---|---|
| Image dataset #1 | 5.509 | 3.342e-3 |
| Image dataset #2 | 2.361 | 8.056e-3 |
| Image dataset #3 | 2.994 | 7.221e-3 |
| Image dataset #4 | 4.112 | 3.505e-3 |
| Image dataset #5 | 7.628 | 1.563e-3 |
| Image dataset #6 | 2.361 | 4.693e-3 |
| Image dataset #7 | 2.995 | 2.317e-3 |
| Image dataset #8 | 4.112 | 1.741e-3 |

## Discussion and Conclusion

Looking at figure 2, we can analyze the effectiveness of our SRCNN in outputting an image visually similar to the ground truth. In (a), we see that the boat is deblurred with strong edge detection between where the boat ends and where the water begins, something that was not in the input. In other words, there is no longer a grey-ish transition area between where the white of the boat and the dark regions depicting the water is. In (b) and (c), the input image is very noisy and thus the output image is not that similar to the ground truth. Nevertheless, the pixelation from the noise is able to be removed with the deep convolution network and we are able to recreate the general framework of the boat. In (d) where only a little noise was added, the SRCNN is easily able to handle the noise with the outputted image in the figure looking almost identical to the output image in (a). In e-h, the story is similar but the output images are closer to the ground truth because we down sampled at only a scale of 2. In (e) where no noise is added, the output image is almost identical to the ground truth with only a slight bit of fine detail lost when it comes to the light white ropes on the boat. Additionally, despite the SNR of Image Dataset #2 and #3 being very close to the SNR of Image Dataset #6 and $7, their output images are much more accurate. For example, in (f), we recover the outline for most of the front circular black object, with the rest of the image illustrated pretty well. The only issue in this super resolution is the detail of the current and land parts. In (g), we once again are able to remove the pixelation that occurred because of the noise that was added and end up with an output image that contains most of the fine detail of the ground truth image.

Ultimately, from the results of the experiments, the main takeaway is that the scale of down sampling impacts the ability to recover the ground truth much more than the amount of Gaussian noise added. As the scale of down sampling increases, which causes more fine details to be lost, the model, trying to predict the fine details lost, will generally assume these regions are constant which may not be true. Meanwhile, utilizing the training data, the model can learn to recognize where noise is added and then smooth out the region of the image to all be the same color. On that same point, it can learn the absolute difference between pixels where it is fair to assume the difference in color is due not to noise. With that being said, one issue in reconstruction that occurred due to the added noise was in identifying the front black circular object which was both next to the white ship and dark current. The reason I expect that the black circular object was sometimes misidentified as water is because of the clipping of grey-scale values to be in range [0,1]. Since the pixels that make up the black circular object had values already close to 1, the added noise only served to lower the grey-scale values of some of these pixels.

## References


1. C. Dong, C. Loy, K. He, and X. Tang, “Image super-resolution using deep convolutional networks,” IEEE Trans. Pattern Anal. Mach. Intell. **38**(2), 295–307 (2016), https://doi.org/10.48550/arXiv.1501.00092.
2. Ronneberger, O., Fischer, P., Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In: Navab, N., Hornegger, J., Wells, W., Frangi, A. (eds) Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. MICCAI 2015. Lecture Notes in Computer Science(), vol 9351. Springer, Cham. https://doi.org/10.1007/978-3-319-24574-4_28.
3. S. Anwar, S. Khan, and N. Barnes, “A deep journey into super-resolution: A survey,” ACM Comput. Surv. **53**(3), 1–34 (2020), https://doi.org/10.48550/arXiv.1904.07523.
4. J. Kim, J. K. Lee and K. M. Lee, "Accurate Image Super-Resolution Using Very Deep Convolutional Networks," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016, pp. 1646-1654, doi: 10.1109/CVPR.2016.182.