Image Super-Resolution

Lasya Manthripragada
NUID : 002130377
Faiz Khwaja
NUID : 002133268

Abstract

The goal of this project is to create an image super resolution model using the current deep learning techniques that can generate high resolution quality images. This project report involves going through the literature survey of the previous methods/models used for the same, and training of CNN based models on a combination of high-resolution and low-resolution images and evaluation of the same models using various metrics such as peak signal-to-noise (PSNR) and similarity index measure (SSIM).

1. Introduction

Image super resolution is a task to enhance the images to high resolution by using deep learning techniques. It is mainly used to detect faces in a low resolution set up, such as surveillance cameras [1]. The authors in [1] very rightly suggested that for watching movies and playing video games, it can offer a superior visual experience. A low-resolution image is simply an image with a fewer number of pixels per inch (PPI). For example, if there is an image of size (60,60) and we want plot it in a canvas of size (6, 10) inches- the number of PPI would be (10,6) whereas if we want to plot the same image but of size (1400,931) in the canvas of same size, the PPI would be (233, 248) thus giving a better-quality image. There is a good explanation given by the author in [2] about the same. Generally, to increase the size of an image, for example we want an image size of 512 x 512 from a size of 256 x 256, we resize it using various interpolation [3] techniques. The most common interpolation technique used is the bicubic interpolation. There is a brief yet very insightful explanation written in [3]. Now, our goal is to build a deep learning model to produce a high-(er) resolution image from a lower resolution image which is better than the image produced by bicubic interpolation or any other interpolation technique for that matter. In this project, we will be presenting a simple CNN model, a concatenation model, and a U-Net architecture model. Concatenation operation [4] which is a "stacking" technique. So, if a layer has the input size (H, W) with 28 filters and some other layer has the same size (H, W) with 28 filters, if we combine these layers the model will learn new features because the feature space is increasing i.e., 56 filters. Please refer to [4], it has a very insightful explanation on why this operation works. In addition to this, we also implemented the U-net architecture, where we will tweak the input and output layers so that it can be trained with the spatial resolution of the dataset we are using. We will further discuss the implementation and results of the said models in Methodology and Experiments sections in detail. Overall, we are taking a dataset with both low-resolution and high-resolution images and training it on models with different architectures and evaluating these models' using metrics like MSE, PSNR and SSIM. To ease the usage of these models.

2. Literature Survey

Deep learning has improved over the years, and it has many working models for image super resolution. We are presenting here 6 papers that are relevant to our project. Wenming Yang et al [6] discussed in their paper about the current models and their challenges. They divided the paper into reviewing the architectures and the optimizing parameters for the same. One of the best architectures was DBPN – deep back projection network which has the highest number of parameters (around 10 M) – in each block of this architecture the input is first down sampled and then the down samples images go through the convolutional layers the output of which is

combined with the input image (which gives the residual image) and this process is called "back projection". This happens in every block of the DBPN architecture. Please refer to the paper for more extensive details. From the paper, we inferred that CNN with a greater number of parameters yields better results. Chao Dong et al [7] proposed a SRCNN – super resolution convolutional neural network. The SRCNN's PSNR value is 27.95 db., so we can infer that the model performed good. The authors also mentioned that their model is way faster. They tested the SRCNN model on Set5 data [8] by using different filter sizes, they concluded that a greater number of filters give good results. Also, larger filter sizes give the better result, but the computational speed decreases. They also tried adding one layer to the architecture to see if it would give better results, it took longer to compute, and eventually converged to the same result, indicating that deeper network did not help here. Xiaole Zhoa et al [9] proposed architecture without the residual connections, instead the skip connections are just the concatenation channels from different layers, hence they named the model FC^2-CN. It has much fewer parameters and gives better results. We are planning to heavily refer from this paper while we are building put concatenation architecture. Jin Yamanaka [10] et al discussed CNN architecture with residual net. The architecture used a combination of deep convolutional layers and skip connections which gave them 10 times faster than a conventional CNN. They also noted that usage of ensemble learning gives better results, so they suggested that a deep learning model should be combined with ensemble learning for complex problems. Donya Khaledyan [11] et al discussed a low-cost implementation of super resolution using interpolation techniques, they proposed an architecture leveraging interpolation techniques and concluded that because of lesser complexity nature of this methodology, it is a viable solution. However, X. Hu [5] et al discussed that using bicubic interpolation may not always give the best results because of the limitation in their ability to learn complex degradations which are non-stationary. They introduced a robust UNet architecture which involves batch normalization, CNN layers and ReLU activation functions, which according to them learns how different complex features to give good results. Overall, we inferred from the above papers that the concatenation works well, and it is important to re-use lower feature space to higher feature by either using skip connections or concatenation. and PSNR is a good measure to check the goodness of the image. We have also concluded that using interpolation techniques solely might not give the best results, and therefore, using a combination of interpolation techniques along with deep learning is advised.

3. Methodology

In this project, we are experimenting with four different architectures including baseline - the CNN model using concatenation technique, UNet model and simple CNN model. We will demonstrate these the architectures in the upcoming sections. Firstly, we implemented a baseline model, which is a simple CNN autoencoder model. We are using convolutional layers as they perform good on images and we are using the autoencoder architecture because we need the output of the model to be an image, since it is not a classification or regression task – an autoencoder architecture was the best to use in our case. We resized the lower resolution images into the same size as the higher resolution image, in our case, the higher resolution image is of size 512 x 512. We are resizing it using bicubic interpolation – since we are using mean square error and mean absolute error (MSE, MAE) for ease of comparison between input and output images we make both the images to be of same size.

3.1 Baseline Model

In the baseline model, the encoder reduces the input image to 16 x 16 in the latent space, and the decoder up samples the image to 512 x 512 – which is our desired output. In the next three sections, we will discuss the CNN model using concatenation technique and UNet model architecture, and the simple CNN model.

3.2 CNN Model using concatenation.

The CNN model we've experimented with uses the concatenation technique. Concatenation is a technique to combine the high-level features with the low-level features. It basically concatenates the channels from the two layers together, that way the model will be able to learn new features that might be dependent on the features both in high and low feature space. Figure 1 – gives the architecture of the model. We are using bicubic interpolation to resize the lower resolution images to 512 x 512. The model takes in an image of size 512 x 512 and outputs an image of the same size. It takes in the images of 512 x 512 x 3 size and in the encoder – it is reduced to 16 x 16 x 512. The architecture of this model is like that of our baseline model expect we are using the concatenation technique to combine the feature spaces. In decoder phase – the images are reduced to a certain size – let's say to 32 x 32 x 512, the two layers of the same spatial dimension in encoder and decoder are combined, the resulting

next layer would be 32 x 32 x 1024. This worked better because of the reasons mentioned above. We heavily borrowed the architecture from [14]. The results of this architecture will be discussed in detail in the next section.

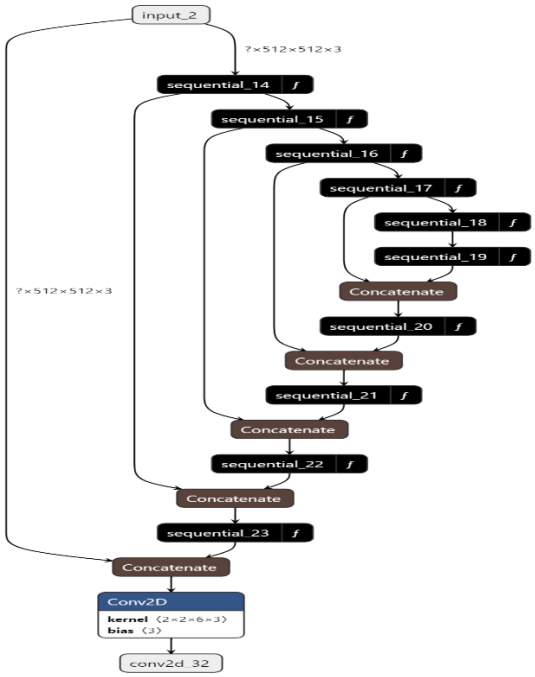

Figure 1 CNN with Concatenation

3.3 UNet Model

UNet model was originally developed for bio-medical image segmentation tasks [12]. UNet consists of an encoder and decoder along with skip connections. Skip connections help pass the information directly from the encoder to the decoder – which helps in smoothening the training process and using the lower feature space with the higher feature space. This process helps preserve the details that might be lost during the down sampling process, therefore resulting in better/higher quality images. The UNet model we have experimented with has a combination of convolutional layers, batch normalization layers, max pooling and ReLU activation functions. Our model takes in pairs of lower resolution and higher resolution images and outputs the predicted higher resolution image from the lower resolution image. Figure 2 gives the architecture of the UNet model [19] we are using, as mentioned, the model brings down the image to 16 x 16 in the latent space and then up samples it to our desired output size. The results of the model will be discussed in the next section. We extensively borrowed from [18] to implement this model.

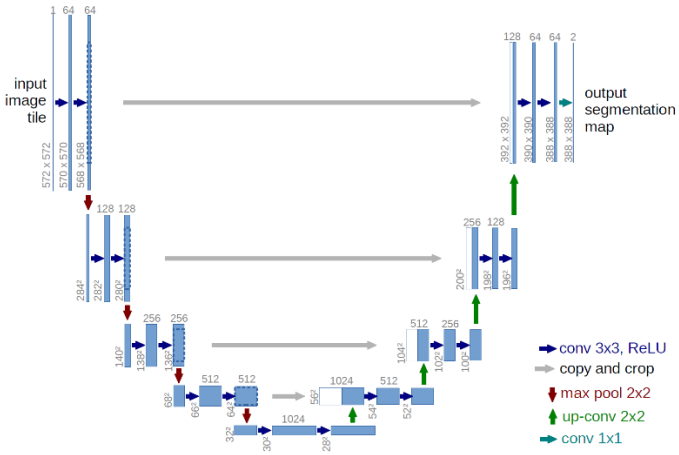

Figure 2 – UNet Architecture

3.4 Simple CNN model

After many trials, we decided to implement a simple CNN with 3 convolutional layers, and one max pool layer. Figure 3 gives the architecture of the model. It takes in the resized image, using bicubic interpolation, as input and gives the output of the desired size. It was worth mentioning that it worked unexpectedly well. More information on the model's performance will be discussed in the next section.

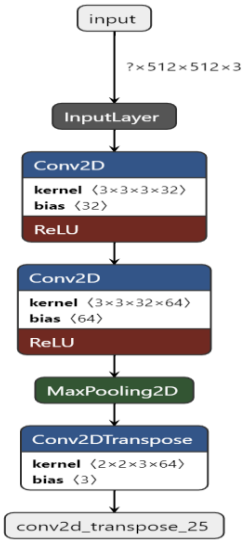

Figure 3 Simple CNN architecture

3.5 Loss Function and Metrics

The lower resolution images are up sampled to the size of the higher resolution while producing the output of the model, here we are comparing pixel – to – pixel between the two sets. This is the L1 loss - it is the absolute difference between the pixels (Corresponding), and it is known to be less affected by the outliers, unlike MSE loss so it gives better results.

The metrics we are using are PSNR and SSIM. PSNR -which is the log ratio of the maximum pixel (in our case, it 1 since we are normalizing the image between 0-1) and the MSE, the higher the better

SSIM - which gives the measure of how similar two images are in terms of luminance, contrast and structure, the metric is explained in further detail below.

$$SSIM(x, y) = (l(x, y))^{\alpha} * (c(x, y))^{\beta} * (s(x, y))^{\gamma},$$

where x,y are the predicted o/p and the ground truth respectively. l(x,y) gives the luminance, c(x,y) gives the contrast and s(x,y) gives the structure. Calculations on how to calculate each term is explained in this very insightful article [13].

Overall, we are taking a dataset with both low-resolution and high-resolution images and training it on models with mentioned in above sections. and evaluating these models' using metrics like MSE, PSNR and SSIM.

4. Experimental Results

4.1 Dataset

We are taking data from two sources and combining them. [15] has 100 images in low resolution and 100 images in high resolution. [17] has 855 images in each low resolution and high-resolution image sets. We are going to combine this data and split it into training, validation, and test sets. [15] is a subset of the set5 [8] dataset. We will first train on these images, and if the model doesn't perform well will use the ImageNet [16] dataset and convert the HR images into LR. Figure 4 is a sample of our dataset. Table 1 gives the train – test – validation classification.

Dataset	Percentage
Train	70%
Test	20%
Validation	10%

Table 1 Train – Test – Validation Split

Figure 4 HR image v/s LR image

4.2 Baseline Model

Baseline model results were not as significant – it did not perform better than the interpolation technique. Table 2 gives the results of the baseline model and Table 3 gives details of the hyperparameters. We did not report validation results for baseline model.

Metrics	Train
MAE	0.98
PSNR	25.4
SSIM	0.72

Table 2 Results for baseline

Hyperparameters	Setting
Epochs	20
Batch Size	16
Learning Rate	0.001
Optimizer	Adam

Table 3 Settings for baseline

4.3 Concatenation Technique

The mentioned CNN model with concatenation technique was tested on the dataset in the above section and compared with the baseline model results. The PSNR and SSIM of this model exceeded that of baseline's model in just the first few epochs. The results of this concatenation technique are presented in Table 4. Figure 8 shows the results of the baseline model versus the concatenation model, and the results of concatenation model have done visibly better. Table 5 gives the settings of our model. Baseline model has the same settings.

Metrics	Train	Validation
MAE	0.0103	-
PSNR	42.8	44.2
SSIM	0.975	0.967

Table 4 Results for CNN model with concatenation

Hyperparameters	Setting
Epochs	20
Batch Size	16
Learning Rate	0.001
Optimizer	Adam

Table 5 Settings for CNN model with Concatenation technique

4.4 UNet Model

The described UNet model was tested on the same dataset and used PSNR and SSIM as the evaluation metrics with MAE as the loss function. It did not work better than the CNN model with concatenation, however, it worked better than the baseline model. Figure 8 gives the comparison of the results, and there is no significant difference between the results – [5] mentioned that UNet gave visually better results but PSNR and SSIM were the lowest for the same. Table 6 gives the setting of the model. Table 7 – gives the results of the model.

Hyperparameters	Setting
Epochs	20
Batch Size	64
Learning Rate	0.001
Optimizer	Adam

Table 6 Setting for UNet model.

Table 7 gives the results of PSNR and SSIM of the unet model.

Metrics	Train	Validation
MAE	0.0373	-
PSNR	36.7	30.2
SSIM	0.923	0.928

Table 7 Results for UNet model

4.5 Simple CNN model

It is worth mentioning that a simple model with 3 convolutional years and 1 max pool layer performed better. However, it might be because of the limited data we had. It gave better results than the baseline model, and as you can see the Figure 8, the results are quite good when compared with the other two models. It has the same settings as the CNN model with concatenation technique. Table 8 – gives the results of the simple CNN model.

Metrics	Train	Validation
MAE	0.0147	-
PSNR	40.55	40.61
SSIM	0.95	0.94

Table 8 Results for Simple CNN model

4.6 Results

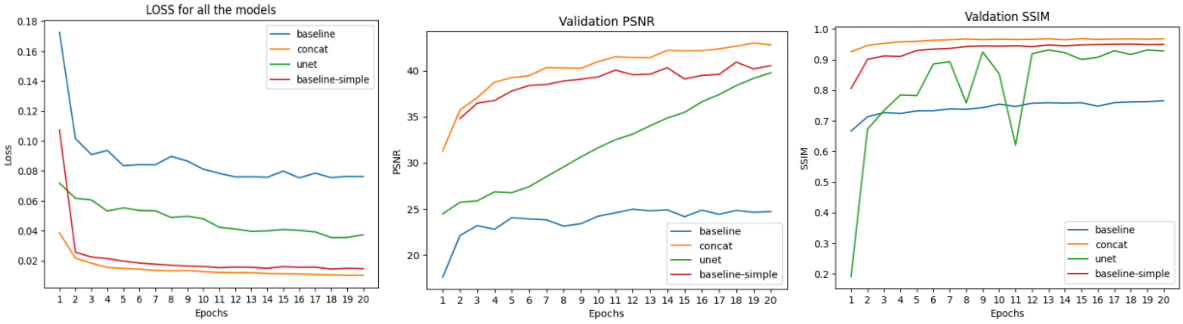

Figure 5 – 7 Loss, Validation - PSNR, Validation – SSIM

As expected, the Concatenation technique model performed the best. The simple CNN model surprisingly stood out and gave high PSNR. Although the UNet model did not give the highest PSNR or SSIM – the results were significantly better than the interpolation technique. PSNR and SSIM – validation results are reported above.

Figure 8 gives the results of the performed model along with LR image– Concatenation model performed the best.

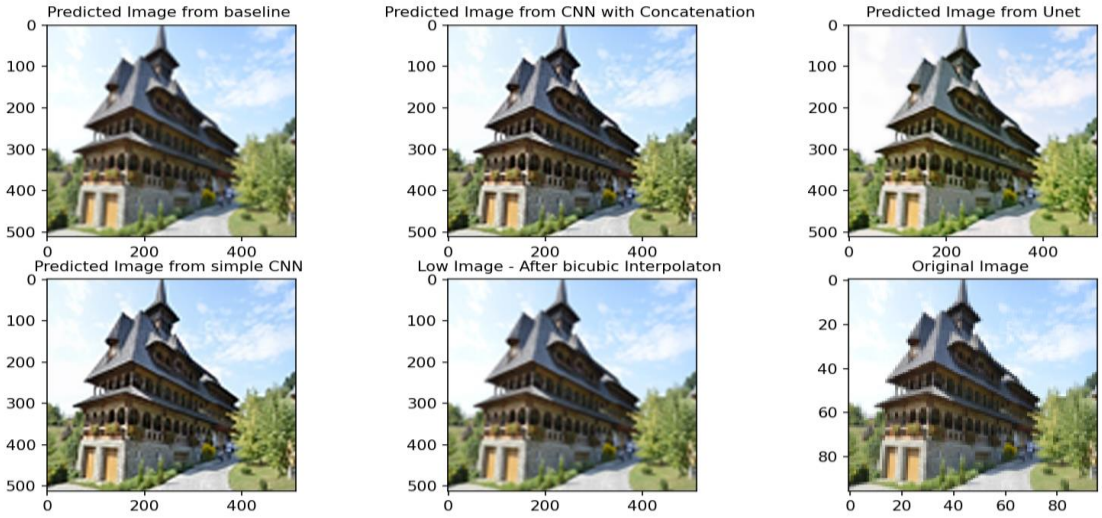

Figure 8 - Results of the all models.

5. Conclusion

In this project, we implemented three different models. The CNN model with concatenation technique to be the best. The simple CNN model surprisingly did well, and the results were also significantly good. However, there was no evident difference between the results. [5] suggested to try different loss functions to optimize the difference between HR and LR images. Our future scope would be to try different metrics such as perceptual loss to experiment. Additionally, making vaster datasets, the dataset we used has 1000 images – getting a bigger dataset and the right hardware resources – the above-mentioned models can be experimented on this new dataset. Overall, the CNN model with concatenation was our best model. [20] has the code for our project.

6. References

1. M. Yang and J. Qi, "Reference-based Image Super-Resolution by Dual-Variational AutoEncoder," 2021 International Conference on Communications, Computing, Cybersecurity, and Informatics (CCCI), Beijing, China, 2021, pp. 1-5, doi: 10.1109/CCCI52664.2021.9583193.
2. https://www.linkedin.com/pulse/what-difference-between-low-high-resolution-image-vala-vincent/
3. https://annmay10.medium.com/resizing-images-using-various-interpolation-techniques-4b99800999f2
4. https://www.researchgate.net/post/What-are-the-operations-of-concatenation-or-combination-exist-in-the-literature-for-CNN-architecture-of-Deep-Learning
5. X. Hu, M. A. Naiel, A. Wong, M. Lamm and P. Fieguth, "RUNet: A Robust UNet Architecture for Image Super-Resolution," 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Long Beach, CA, USA, 2019, pp. 505-507, doi: 10.1109/CVPRW.2019.00073.
6. Wenming Yang, Xuechen Zhang, Yapeng Tian, Wei Wang, Jing-Hao Xue, and Qingmin Liao. 2019. Deep Learning for Single Image Super-Resolution: A Brief Review. Trans. Multi. 21, 12 (Dec. 2019), 3106–3121. https://doi.org/10.1109/TMM.2019.2919431
7. Dong, C., Loy, C.C., He, K., Tang, X. (2014). Learning a Deep Convolutional Network for Image Super-Resolution. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds) Computer Vision – ECCV 2014. ECCV 2014. Lecture Notes in Computer Science, vol 8692. Springer, Cham. https://doi.org/10.1007/978-3-319-10593-2_13
8. https://huggingface.co/datasets/eugenesiow/Set5
9. Zhao, X., Liao, Y., He, T., Zhang, Y., Wu, Y., & Zhang, T. (2021). FCN: Fully Channel-Concatenated Network for Single Image Super-Resolution.
10. Yamanaka, Jin & Kuwashima, Shigesumi & Kurita, Takio. (2017). Fast and Accurate Image Super Resolution by Deep CNN with Skip Connection and Network in Network. 217-225. 10.1007/978-3-319-70096-0_23.
11. Khaledyan, Donya & Amirany, Abdolah & Jafari, Kian & Moaiyeri, Mohammad & Zargari, Abolfazl & Mashhadi, Najmeh. (2020). Low-Cost Implementation of Bilinear and Bicubic Image Interpolation for Real-Time Image Super-Resolution. 1-5. 10.1109/GHTC46280.2020.9342625.
12. 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
13. https://medium.com/srm-mic/all-about-structural-similarity-index-ssim-theory-code-in-pytorch-6551b455541e
14. https://www.kaggle.com/code/nirawitkanthachai/low-resolution-images-to-high-resolution/notebook
15. https://www.kaggle.com/datasets/akhileshdkapse/super-image-resolution.
16. https://image-net.org/download-images.php
17. https://www.kaggle.com/datasets/adityachandrasekhar/image-super-resolution
18. https://www.kaggle.com/code/harshraone/super-resolution-u-net
19. https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/
20. https://github.com/lasyamanthri/CS7150_DeepLearning/tree/master