Image Interpolation and Super resolution

Report for IITB EE610 Image Processing 2021

Kruthagnya Myana
Electrical Engineering Dept.
18D070015
kruthagnya.m@iitb.ac.in

Sudheeradh K
Electrical Engineering Dept.
18D070016
sudheeradh@iitb.ac.in

Neha Jahnavi N
Electrical Engineering Dept.
18D070017
18D070017@iitb.ac.in

***Abstract*—The project's objective is to implement Image interpolation and Super-resolution algorithms and compare visual results between them. We plan to implement a GUI to apply these algorithms easily on required Images. Image super-resolution and interpolation techniques that enhance the images' quality and dimensions, which makes the images visually appealing and properly visible on our HD and 4K devices. In this project the features of Nearest Neighbor Interpolation, Bilinear Interpolation and Image super-resolution using deep learning were analysed.**

***Index Terms*—Bilinear, Nearest Neighbor, Image Interpolation**

I. Introduction

It is a chellenging task to reconstruct a High resolution image from a Low resolution image counterpart. Though there are numerous algorithms which include simply upscaling the image and estimating the pixels by using Nearest Neigbour or Bilinear algorithms, there has been tremendous success by using Convolution neural networks also. We chose this project to learn and implement the algorithms and to learn how the MSE is minimised as we progress to FSRCNN.

II. Background and Prior Work

Here is some background on what Interpolation is:
The interpolation is the process of finding out the unknown pixels of the image. There are various methods for interpolation. Image interpolation algorithms directly affect the quality of image magnification.Image interpolation is the process of transferring image from one resolution to another without losing image quality. When a small image is enlarged, for example if an image is zoomed to 400 shown in Fig. 1., the color values of original 4 adjacent pixels marked A, B, C, and D in (a) were filled in the new A, B, C, and D locations in (b) accordance with the magnification factor. But there are a large number of pixels which values are unknown between A, B, C, and D, such as P. so the values of these pixels should be calculated through interpolating estimation

III. Data and Methodology

We have trained our ML model using the DIV2K dataset, as the original dataset used by the authors require their permission to access them. All of the tests are performed with a scale factor of 4 between low and high-resolution images. Our FRSCNN model file is named model.h5, and it contains the different parameters and learned weights.

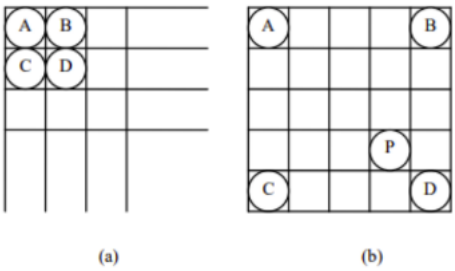

Fig. 1. Effect of Interpolation

A. Nearest Neighbour Interpolation

Nearest-neighbor interpolation is a simple method of multivariate interpolation in one or more dimensions. The nearest neighbor algorithm selects the value of the nearest point and does not consider the values of neighboring points at all, yielding a piecewise-constant interpolant.

B. Bilinear Interpolation

Bilinear interpolation takes a weighted average of the 4 neighborhood pixels to calculate its final interpolated value. The result is much smoother image than the original image. Bilinear interpolation is performed using linear interpolation first in one direction, and then again in the other direction. Although each step is linear in the sampled values and in the position, the interpolation as a whole is not linear but rather quadratic in the sample location.

C. Fast Super-Resolution CNN

FSRCNN is done in 5 steps which include upscaling our images to our desired resolution by using convolution(feature extraction), shrinking the image to reduce the number of feature maps, then we non-linearly map the low and high resolution image vectors, expanding the image to increase the number of feature maps again, and then reconstructing the High-resolution image using Deconvolution.

Fig. 2. Original Low-Resolution Image

Fig. 3. After Nearest Neighbour Interpolation

Fig. 4. After Bilinear Interpolation

Fig. 5. After Fast Super-resolution CNN

Fig. 6. Original Hi-Resolution Image

IV. Experiments and Results

We learned and implemented the Interpolation algorithms (Nearest neighbor, Bilinear), and a Super-resolution algorithm(FSRCNN)

Fig. 7. Image 1

Fig. 8. Image 2

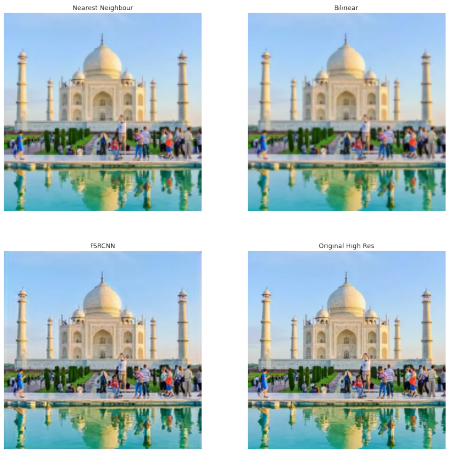

Fig. 9. Image 3

Fig. 10. Image 4

We Calculated the PSNR values of the images

	PSNR in dB		
	NN	Bilinear	FRSCNN
Image 1	19.5776	18.4848	20.1031
Image 2	26.4024	24.1314	29.1077
Image 3	26.7096	24.6258	28.2342
Image 4	22.7018	20.9643	25.4426

V. Learning, Conclusions, and Future Work

We see that FSRCNN model has higher PSNR values, hence better quality of the reconstructed images. By using interpolation algorithms, since they estimate the pixel values of the upscaled images either by weighted average, or normalising, we observe that the images lack the texture and detail that is taken care of in FSRCNN.

The training model, at first crops out a 648x648 size image from our original High resolution image, performs operations like flipping, and increasing saturation, to make it more accessible to detect the error. Then it downscales the image by 4x, to 162x162 size and gives this as input and defines the 648x648 image as the output. This part is called augmentation.

The model then proceeds to test this with the images in the dataset DIV2K. We used zeroes for padding, and we kept the learning rate as 0.001, we considered MSE (Mean squared error) to compare between the output and the HIgh-resolution image.

Contribution of Team Members

We have first gone through some research papers and have learned about the algorithms of interpolation and super-resolution. The report is written by all three of us, as for the code, Kruthagnya has worked on Interface.py, the GUI base for this project. Neha has worked on the Nearest neighbor interpolation. Sudheeradh has written code for Bilinear interpolation. We have also trained the ML model for FSRCNN on the DIV2K dataset, which gave us the PSNR values of the algorithms, for comparison.

References

[1] **Accelerating the Super-Resolution Convolutional Neural Network**

[2] **GitHub - FRSCNN Pytorch**

[3] **GitHub - FRSCNN Tensorflow**

[4] **Keras Documentation**

[5] **Nearest Neighbor Interpolation**

[6] **Bilinear Interpolation**