ISSN 2277-2685
IJESR/July. 2020/ Vol-10/Issue-3/1-14
Kadari Neeraja *et. al.*, / International Journal of Engineering & Science Research

"IMAGE SUPER-RESOLUTION USING SUPER-RESOLUTION CONVOLUTIONAL NEURAL NETWORKS"

Kadari Neeraja [1],
Associate Professor,
JNTUH College of Engineering, Hyderabad.
kadari.neeraja@gmail.com

Bommai Meenakshi [2],
meenakshibomma@gmail.com

Abstract

The primary goal of Super-Resolution (SR) is to create a higher resolution image by enhancing lower resolution images. This is crucial because high-resolution images contain more pixels, providing finer details of the original scene. The demand for high resolution is widespread in computer vision applications, as it improves pattern recognition and image analysis performance. In fields like medical imaging, high resolution is essential for accurate diagnosis. Moreover, various applications such as surveillance, forensics, and satellite imaging require high-resolution capabilities for zooming into specific areas of interest. However, obtaining high-resolution images can be challenging and costly due to limitations in sensor and optics manufacturing technology. To address these issues, Super-Resolution Convolutional Neural Networks (SRCNN) have emerged as a cost-effective solution. SRCNN enables the transformation of low-resolution images to high-resolution ones, allowing the utilization of existing low-resolution imaging systems. As implied by its name, SRCNN is a deep convolutional neural network that learns to map low-resolution images to high-resolution ones in an end-to-end manner. This approach significantly improves the quality of low-resolution images. Unlike traditional methods that handle different components separately, SRCNN optimizes all layers together. The performance of the network is assessed using image quality metrics such as peak signal-to-noise ratio (PSNR) and mean squared error (MSE). Additionally, the Open Source Computer Vision Library (OpenCV) is utilized for pre and post-processing of the images

Keywords – Super Resolution, High Resolution Image, Super Resolution Convolution Neural Network, Deep Learning,

INTRODUCTION

With the Internet boom and rapid growth of information technology, the demand for signal and information processing has increased significantly[1-2]. Image processing has become a vital part of this information processing landscape[3]. As displays and screens enable higher resolutions, the consumption of high-resolution content, such as 4K resolution videos and images, is on the rise. To cater to the need for greater fidelity, image super-resolution techniques have been proposed to artificially enhance image resolution [4].One major challenge in computer vision is single image super-resolution (SR), aiming to upscale a high-resolution image from a low-resolution one[5]. This problem is fundamentally ill-posed since

multiple solutions exist for each low-resolution pixel, making it non-unique and undefined[6]. To tackle this, the solution space is usually constrained by leveraging strong prior knowledge[7]. State-of-the-art methods often employ example-based learning, utilizing internal similarities within the same image or mapping functions learned from external low- and high-resolution image pairs[8-10]. For generic super-resolution, external example-based approaches can be formulated or adapted to fit specific tasks, like face hallucination, by configuring the training samples accordingly. As information technology advances, fulfilling the demand for high-definition images using low-resolution ones has become challenging. High-resolution images offer higher pixel density, precise information, and rich data that are essential for practical image analysis and recognition. The single-image super-resolution algorithm can be categorized into bi-cubic interpolation, image reconstruction, and dictionary learning methods. Despite these three types, the debate mainly revolves around learning-based approaches within these categories, leading to two groups of related algorithms based on the source of the training patch: external and internal datasets.

1.1 **Convolution Neural Networks (CNNs)**:

Convolutional Neural Networks are a class of deep learning models designed primarily for image processing tasks, though they are also used for other data types, such as audio and text. CNNs are inspired by the visual processing mechanism of the human brain and have proven highly effective in computer vision tasks like image classification, object detection, image segmentation, and more. In fig.1 CNN architecture is shown.

Different Components of CNNs:

1. **Input Layer:**

- The input layer is the first layer in the CNN and takes the raw input data, typically an image or a tensor representing the data.

2. **Convolutional Layers:**

- Convolutional layers are the core building blocks of a CNN. They consist of learnable filters (kernels) that slide over the input data to perform element-wise multiplications and produce feature maps.

- The convolution operation helps the network identify local patterns and features in the input data. This is illustrated in fig.2.

3**. Activation Function:**

- After each convolution operation, an activation function is applied element-wise to introduce non-linearity into the network.

- Common activation functions include ReLU (Rectified Linear Unit), Leaky ReLU, Sigmoid, and Tanh.

4. **Pooling Layers:**

- Pooling layers downsample the spatial dimensions of the feature maps, reducing computational complexity and making the network more robust to small translations and distortions.

- Max pooling and average pooling are common pooling techniques used to select the maximum or average values within a specific window.

5**. Fully Connected Layers (Dense Layers):**

- Fully connected layers are typically located towards the end of the CNN and connect every neuron in one layer to every neuron in the next layer.

- They are used for making predictions based on the extracted features from earlier layers.

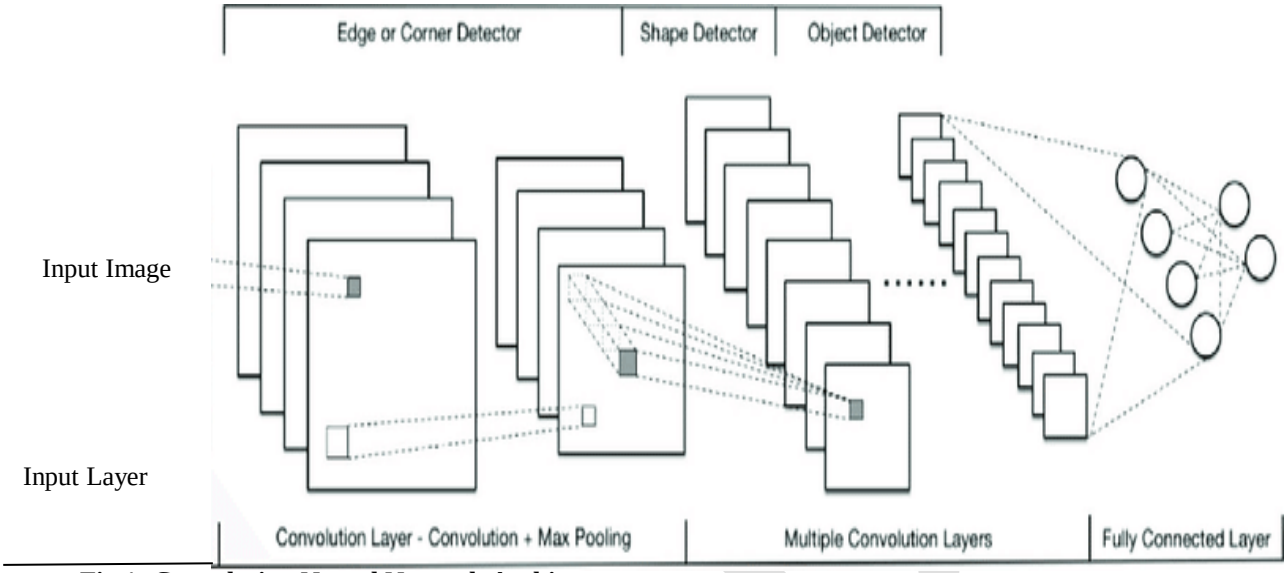

Fig 1: Convolution Neural Network Architecture

6. **Flattening Layer:**

- Before the fully connected layers, the feature maps need to be flattened into a 1D vector.

- This process combines all the spatial information into a single vector, which is then fed into the dense layers.

7. **Output Layer:**

- The output layer produces the final predictions or classifications based on the information learned by the previous layers.

- The number of neurons in the output layer depends on the task at hand; for example, it could be the number of classes in an image classification task.

8**. Loss Function:**

- The loss function measures the difference between the predicted output and the true target labels.

- During training, the CNN aims to minimize this loss function, helping it learn the optimal parameters for making accurate predictions.

9. **Optimization Algorithm:**

- The optimization algorithm is used during training to update the weights and biases of the CNN based on the gradients calculated from the loss function.

- Common optimization algorithms include Stochastic Gradient Descent (SGD), Adam, and RMSprop.

These components work together to process input data, extract meaningful features, and make predictions,

making CNNs powerful tools for various computer vision tasks

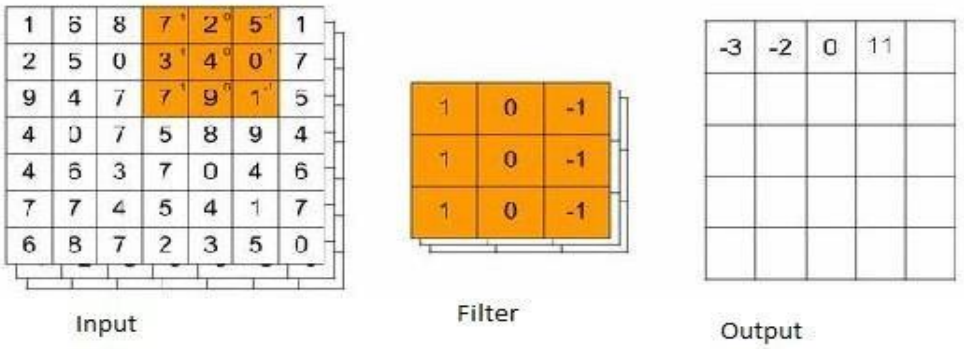

Fig:2. Convolution Operation Illustration

1.2. Image Processing

Image processing is the use of digital computers and algorithms to analyze and manipulate digital images to obtain enhanced images and extract useful information. The process typically involves three main steps:

1. Image acquisition through specialized software.

2. Analysis and manipulation of the image.

3. Output, which may include the modified image or a report based on the analysis results.

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & f(0,2) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & f(1,2) & \dots & f(1,N-1) \\ \vdots & \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & f(M-1,2) & \dots & f(M-1,N-1) \end{bmatrix}$$

RELATED WORK

Single-image super-resolution algorithms can be classified into four groups based on the image priors they use: prediction models, edge-based methods, image statistical methods, and patch (or example-based) methods. Among these, the example-based methods have demonstrated state-of-the-art efficiency. Internal example-based methods leverage self-similarity within the input image to construct exemplary maps, as proposed by Glasner [8] for faster execution. External example-based methods, on the other hand, map external datasets between low and high-resolution image patches. These methods differ in how they connect low and high-resolution patches to dictionary or tuple space and perform visualization in such spaces.Freeman et al. [9] pioneered the use of dictionaries defined as low and high-resolution patch groups, and they locate the nearest neighbor (NN) of the input patch in the low-resolution space, using the corresponding high-resolution patch for restoration. Various mapping functions, such as simple function, kernel regression, random forest, and anchored neighborhood regression, have been proposed to improve mapping efficiency and accuracy.Modern super-resolution techniques often employ sparse coding-based approaches, which focus on enhancing patches. The process involves separate phases of patch extraction and integration, known as pre and post-processing. While most super-resolution algorithms focus on single-channel or grayscale images, methods for color images often convert the problem to another color space (YCbCr or YUV) and apply super-resolution to the luminance space.

Some studies have aimed to super resolve all color channels simultaneously, where each RGB channel is processed independently and then combined for the final output. However, research on the efficiency of different channels and the need to retrieve all three channels remains limited[3-7]. In the context of picture restoration, deep learning approaches have been utilized. For instance, multi-layer perceptron (MLP) models with fully connected layers have been applied for natural image denoising and post-deblurring denoising. In an internal example-based approach, Cui et al. [17] propose incorporating auto-encoder networks into the super-resolution pipeline. However, the deep model in this approach is not explicitly designed as an end-to-end solution, unlike the suggested SRCNN (Super-Resolution Convolutional Neural Network), which optimizes end-to-end mapping and offers higher speed and technical utility.

PROBLEM DEFINITION

The plan is to create and train a fully convolutional neural network (CNN) for image super-resolution. This network will directly learn the mapping between low-resolution and high-resolution images without requiring extensive pre or post-processing steps

PROPOSED APPROACH

The objective is to enhance low-resolution images and generate high-resolution versions from them. To achieve this, we propose using the SRCNN model, which establishes a direct mapping between low-resolution images and their corresponding high-resolution counterparts. Below, we provide an overview of the model.

4.1 Super-Resolution Convolutional Neural Networks (SRCNN Model:

Considering a single low-resolution image, we begin by using bicubic interpolation, the only pre-processing step, to upscale it to the desired scale. The interpolated image is denoted as Y. The objective is to reconstruct an image, F(Y), from Y, which closely resembles the high-resolution picture X of the original ground truth image. Although Y has the same dimensions as X, we still refer to it as a "low-resolution" image for clarity. The learning process involves mapping F, which comprises three conceptual operations:

1) Patch extraction and representation

2) Non-linear mapping

3) Reconstruction.

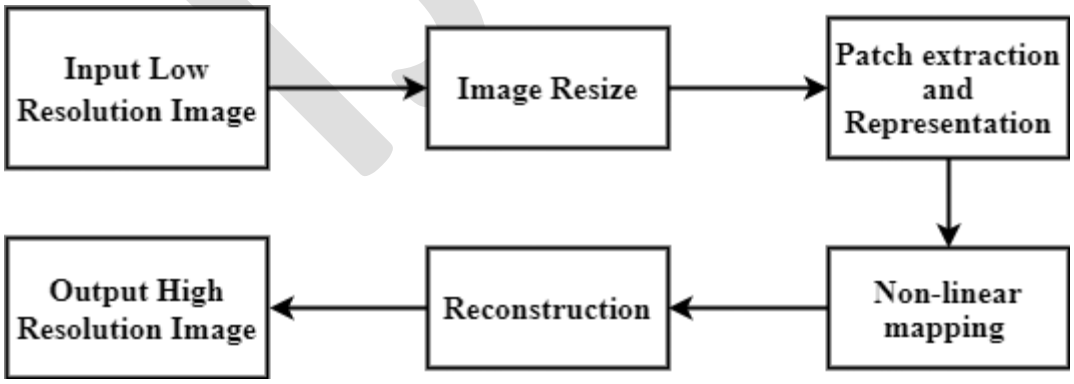

Fig.3 Proposed SRCNN Architecture

4.1.1 Patch extraction and representation:

In this step, patches are extracted from the low-resolution image Y in an overlapping manner. Each patch is then represented as a high-dimensional vector, consisting of a set of feature maps, with the number of maps equal to the vector's dimensions.
Traditionally, image restoration methods involve retrieving patches tightly and representing them using pre-trained bases like PCA, DCT, or Haar. This process can be likened to a series of filters convolving the image, with each filter acting as a basis. However, in our approach, we integrate the enhancement of these bases into the network itself.Formally, the first layer of the network (F1 operation) is expressed as follows:

$$F1(Y) = \max(0, W1 * Y + B1),$$

In this expression, W1 represents a set of n1 support filters of size c x f1 x f1, where c is the number of channels in the input image, and f1 is the spatial size of the filters. W1 applies n1 convolutions to the image, with each convolution having a kernel size of c x f1 x f1. The output consists of feature maps of size n1. B1 is an n1-dimensional vector, where each element corresponds to a filter in the network. The Rectified Linear Unit (ReLU) function, max(0, x), is applied to the filter responses [33].

4.1.2 Non-linear mapping:

In a non-linear fashion, this approach maps each high-dimensional vector to another high-dimensional vector, where the mapped vector conceptually represents a high-resolution patch. For each patch, a n1-dimensional feature is obtained from the first layer. These n1-dimensional vectors are then mapped to n2-dimensional vectors in the next operation. This mapping involves the application of n2 filters with a spatial support of 1x1. While this explanation holds for 1x1 filters, it can easily be extended to broader filters like 3x3 or 5x5. In such cases, the input image patch is not directly involved in the non-linear mapping; rather, it is the 3x3/5x5 patch on the feature map.

The second layer operation is expressed as follows:

$$F2(Y) = \max (0, W2 * F1(Y) + B2)$$

In this expression, W2 comprises n2 n1 x f2 x f2 filters, and B2 is an n2-dimensional vector. Each of the n2-dimensional output vectors represents a visualization of a high-resolution patch used for reconstruction.

To introduce more non-linearity, additional convolutional layers can be added. However, this increases the model's complexity and consequently requires more time for training.

4.1.3 Reconstruction:

In this step, the final high-resolution image is assembled by combining the high-resolution patch-wise representations obtained from the previous stages. The resulting image is expected to closely resemble the Ground Reality X.
Conventional methods for generating the final full image often involve averaging the overlapping high-resolution patches. This averaging process can be seen as using a pre-defined filter over a vector of feature maps. However, in our approach to achieve the desired high-resolution image, we introduce a convolution layer influenced by this process:

$$F(Y) = W3 * F2(Y) + B3$$

Here, W3 consists of c filters with a size of n2 x f3 x f3, and B3 is a c-dimensional vector. This convolution operation effectively combines the high-resolution patch-wise representations to form the final high-resolution image.

Algorithm: SRCNN

Input: Low-resolution image Y (H x W x C), Parameters (W1, W2, W3, B1, B2, B3)

1. Preprocessing:
   - Upscale the low-resolution image Y using bicubic interpolation to obtain Y_interpolated.

2. Patch Extraction and Representation:
   - Initialize an empty list Patches.
   - For each patch P_i of size f1 x f1 extracted from Y_interpolated:
     - Compute the high-dimensional vector V_i using the first layer (F1) of SRCNN:
       V_i = max(0, W1 * P_i + B1).
     - Append V_i to the list Patches.

3. Non-linear Mapping:
   - Initialize an empty list Representations.
   - For each high-dimensional vector V_i in Patches:
     - Compute another high-dimensional vector U_i using the second layer (F2) of SRCNN:
       U_i = max(0, W2 * V_i + B2).
     - Append U_i to the list Representations.

4. Reconstruction:
   - Initialize an empty list HighResPatches.
   - For each high-dimensional vector U_i in Representations:
     - Reconstruct the high-resolution patch using the third layer (F) of SRCNN:
       HighResPatch_i = W3 * U_i + B3.
     - Append HighResPatch_i to the list HighResPatches.

5. Output:
   - Combine the high-resolution patches from HighResPatches to form the final high-resolution image F(Y_interpolated).

End of Algorithm.

In this algorithm, the SRCNN model is used to process the low-resolution image and learn an end-to-end mapping to generate the final high-resolution image. The patch extraction, non-linear mapping, and reconstruction steps are performed using the three layers (F1, F2, F) of the SRCNN model. The algorithm aims to enhance the resolution of the input image efficiently and produce a high-quality high-resolution output.

EXPERIMENTAL RESULTS

During the pre-processing stage, the input data is prepared for training on a machine learning model. In this particular model, the 91 selected images are decomposed into 22,000 sub-images, each of size 32. These sub-images are then converted into arrays and stored as datasets for further use in the training process.

5.1 Building & Training SRCNN Model

The model, known as the 'Super Resolution Convolution Neural Network (SRCNN),' comprises three main parts:

1) Patch extraction and representation

2) Non-linear mapping

3) Reconstruction

The training of the model was performed using the TensorFlow library. Each part of the model consists of one convolutional layer. The details of each layer are discussed in section 3.1. The first layer serves as the input layer with a size of 32x32 and includes 128 filters, each of size 9x9. The second layer has 64 filters of size 3x3, while the output layer (third layer) consists of a single filter of size 5x5. All layers use Rectified Linear Unit (ReLU) as the activation function. The model is trained for 200 epochs using the Mean Squared Error (MSE) as the loss function.

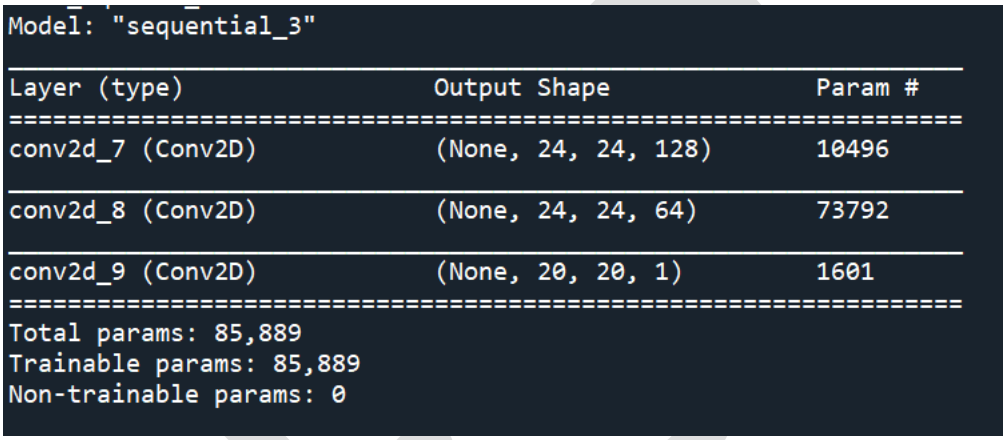

Fig 4: As mentioned in section 5.2, SRCNN Model is formed with three convolution layers. The above picture shows the layers in the trained model. Output shape represents the output size of each input from the layer. Param represents the total no of weights and biases.

```
Epoch 00192: loss did not improve from 0.00280
Epoch 193/200
23504/23504 [==============================] - 99s 4ms/step - loss: 0.0028 - mean_squared_error: 0.0028

Epoch 00193: loss improved from 0.00280 to 0.00280, saving model to weights.h5
Epoch 194/200
23504/23504 [==============================] - 98s 4ms/step - loss: 0.0028 - mean_squared_error: 0.0028

Epoch 00194: loss did not improve from 0.00280
Epoch 195/200
23504/23504 [==============================] - 97s 4ms/step - loss: 0.0028 - mean_squared_error: 0.0028

Epoch 00195: loss did not improve from 0.00280
Epoch 196/200
23504/23504 [==============================] - 95s 4ms/step - loss: 0.0028 - mean_squared_error: 0.0028

Epoch 00196: loss did not improve from 0.00280
Epoch 197/200
23504/23504 [==============================] - 97s 4ms/step - loss: 0.0028 - mean_squared_error: 0.0028

Epoch 00197: loss did not improve from 0.00280
Epoch 198/200
23504/23504 [==============================] - 99s 4ms/step - loss: 0.0028 - mean_squared_error: 0.0028

Epoch 00198: loss did not improve from 0.00280
Epoch 199/200
23504/23504 [==============================] - 113s 5ms/step - loss: 0.0028 - mean_squared_error: 0.0028

Epoch 00199: loss did not improve from 0.00280
Epoch 200/200
23504/23504 [==============================] - 126s 5ms/step - loss: 0.0028 - mean_squared_error: 0.0028

Epoch 00200: loss improved from 0.00280 to 0.00280, saving model to weights.h5
23504/23504 [==============================] - 35s 1ms/step

mean_squared_error: 0.28%
```

Fig 5: The above figure shows the output after each epoch. The model is trained with 200 epochs with the loss function of MSE.

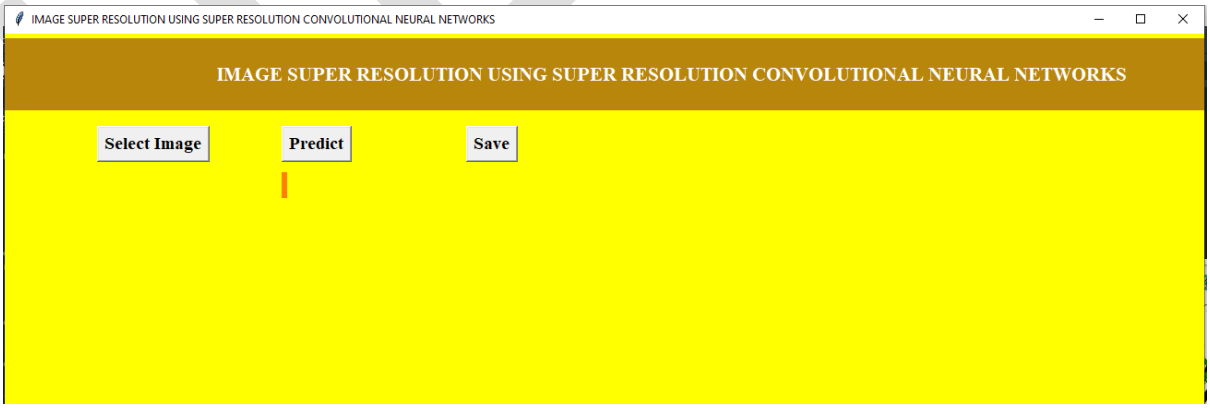

Fig 6: From above screen, click on ‘Select Image’ button to select the Image

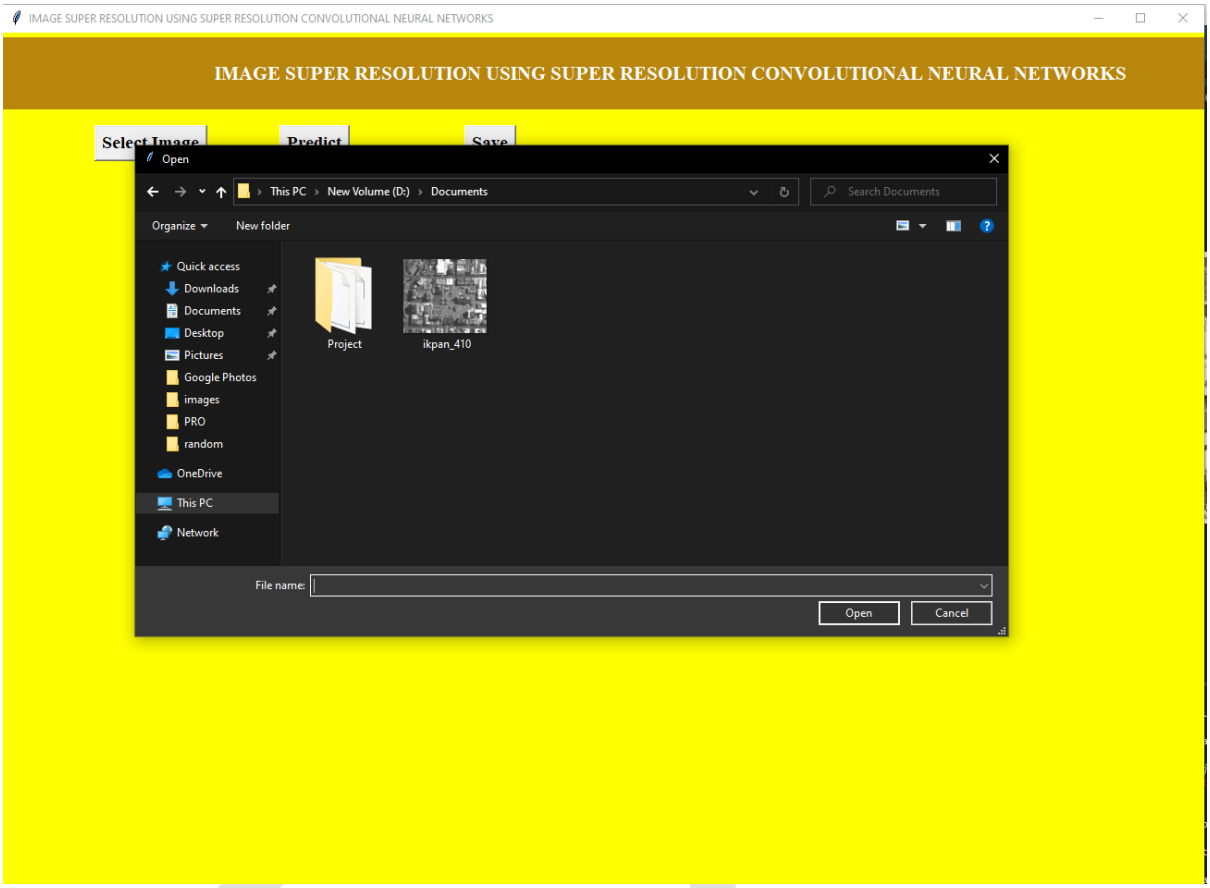

Fig 7: Select an Image from the file folders

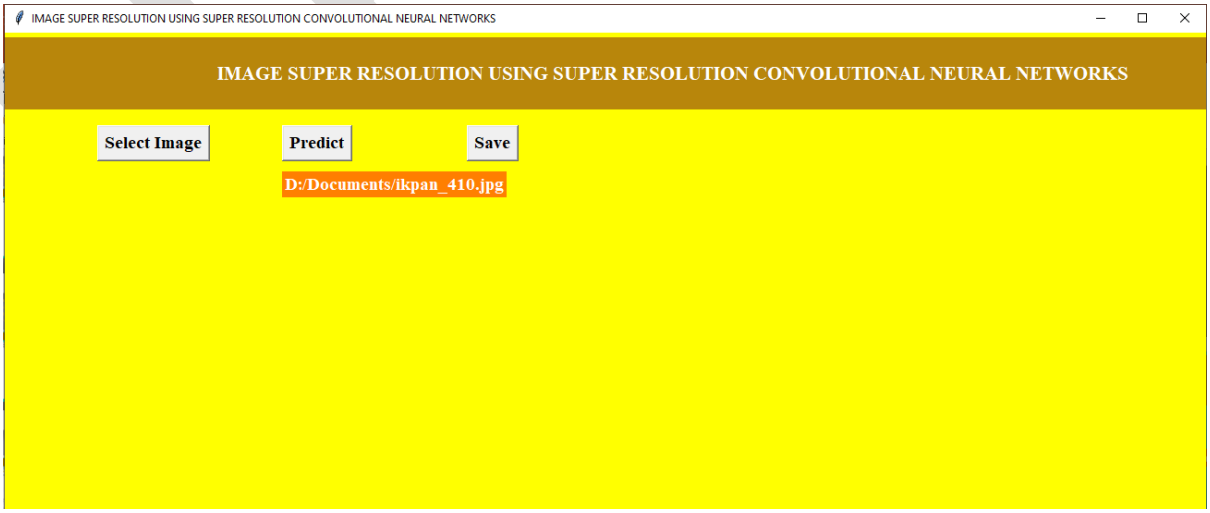

Fig 8: After selecting Image, It will show the path to the Image at the bottom

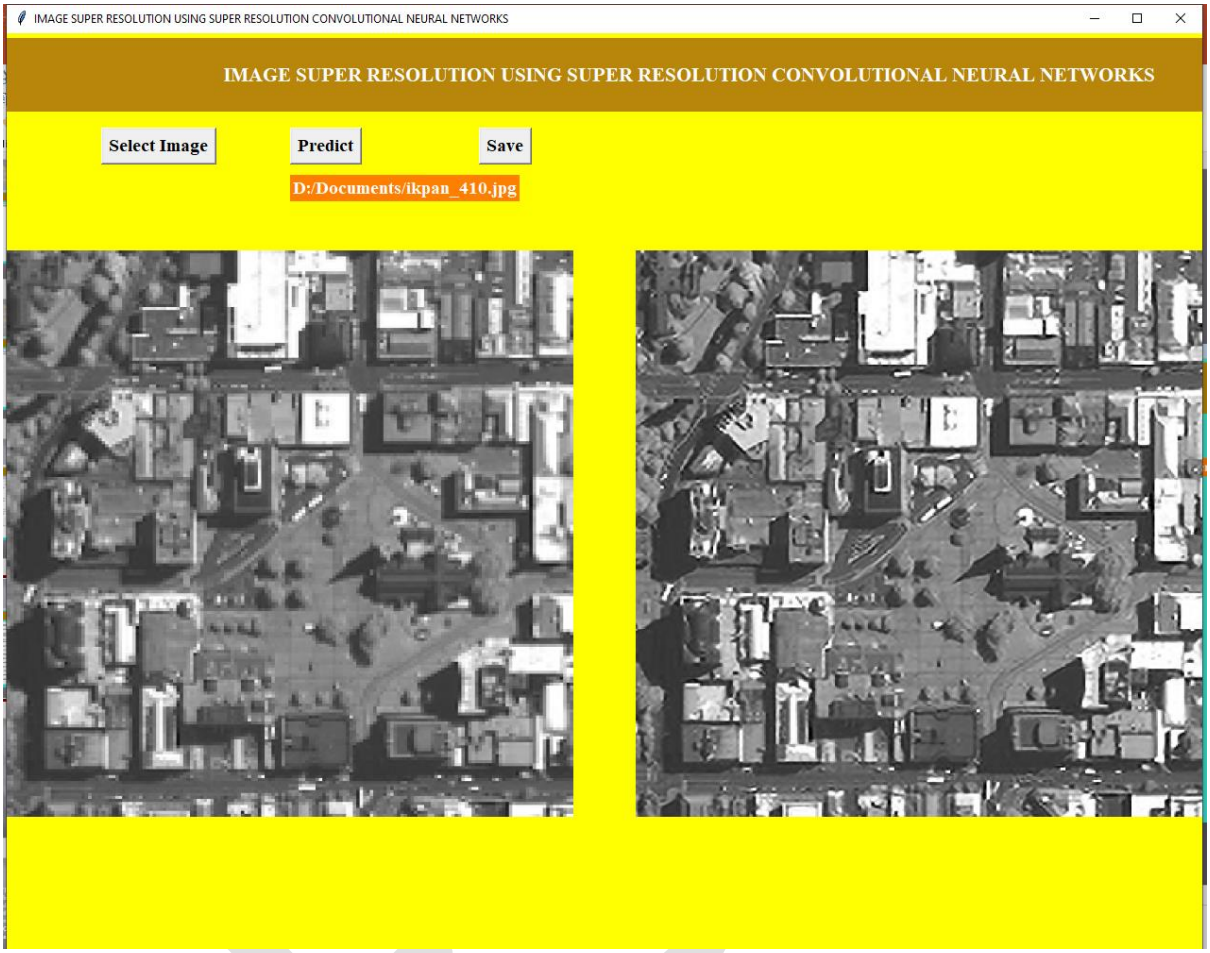

Fig 9: Click on 'Predict' button to get high resolution Image. Left image is the original low resolution image and Right image is recovered high resolution image. Click 'Save' to save the image

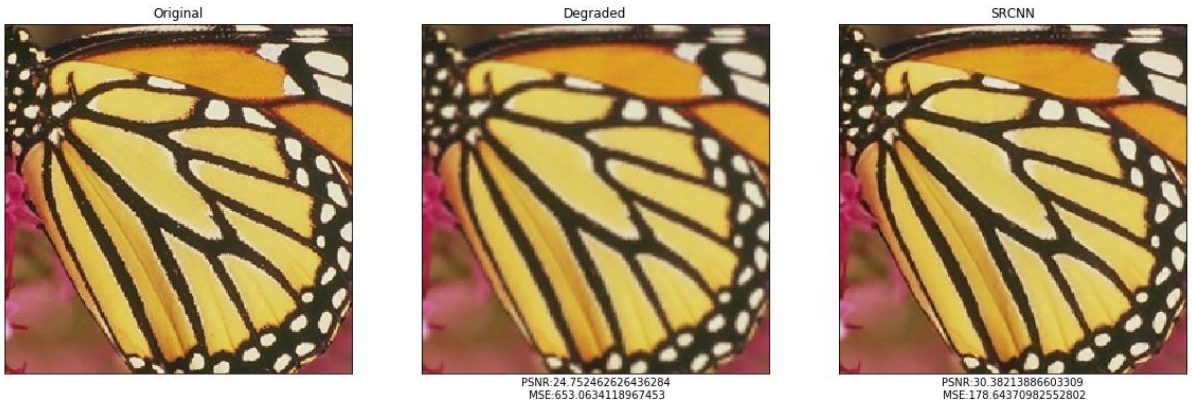

Fig 10: The above images are original image, downscaled low resolution image and Predicted High Resolution Image respectively from left to right.

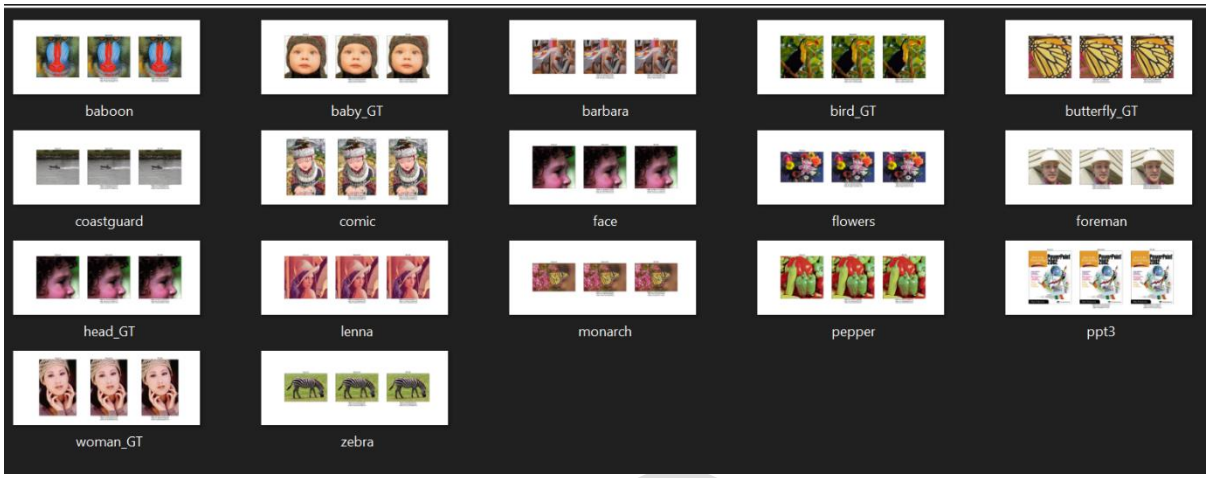

Fig 11: The above image shows Validation Image set outputs

```
baboon.bmp
PSNR:22.285122135474143 MSE:1152.628628072958 SSIM:0.6327198184502538
PSNR:23.035623553580113 MSE:969.7041633624108 SSIM:0.7090932944586413
baby_GT.bmp
PSNR:34.2527045770483 MSE:73.27083787867768 SSIM:0.9344362511483775
PSNR:36.25917073887409 MSE:46.16199203251638 SSIM:0.9549899421914523
barbara.bmp
PSNR:25.888852808173493 MSE:502.7086229899586 SSIM:0.8086912237988951
PSNR:26.575777516739723 MSE:429.1645726122764 SSIM:0.8550465047651278
bird_GT.bmp
PSNR:32.9717779215716 MSE:98.40650856389986 SSIM:0.9532838164378017
PSNR:36.54718663485879 MSE:43.1999209486166 SSIM:0.9693544478647688
butterfly_GT.bmp
PSNR:24.752462626436284 MSE:653.0634118967453 SSIM:0.8788410517538944
PSNR:30.38213886603309 MSE:178.64370982552802 SSIM:0.9519721343623928
coastguard.bmp
PSNR:27.368831974620207 MSE:357.53488122802503 SSIM:0.7527808342151686
PSNR:28.674064088913997 MSE:264.725152216791 SSIM:0.8292474459490644
comic.bmp
PSNR:23.635798521218994 MSE:844.5429938632378 SSIM:0.8295735523314516
PSNR:25.882429134012234 MSE:503.4527323202805 SSIM:0.8994648135506398
face.bmp
PSNR:30.882219119160833 MSE:159.213431846987 SSIM:0.8003579419510646
PSNR:31.641199608086662 MSE:133.68468429542574 SSIM:0.8283444292801055
flowers.bmp
PSNR:27.248686459559124 MSE:367.56400047398984 SSIM:0.8690622024599293
PSNR:29.66020048831105 MSE:210.95098945372675 SSIM:0.8989100264554274
foreman.bmp
PSNR:32.0448352333889 MSE:121.81985893148101 SSIM:0.9383957200607694
PSNR:36.14700153176391 MSE:47.369790326730126 SSIM:0.9624930464724937
head_GT.bmp
PSNR:30.914629910015442 MSE:158.0296668637887 SSIM:0.8008776169731272
PSNR:31.66977921232533 MSE:132.80783419222212 SSIM:0.8286942162476625
lenna.bmp
PSNR:31.35857283101428 MSE:142.6738341696767 SSIM:0.8459403992986435
PSNR:33.10844852555662 MSE:95.35791859591282 SSIM:0.8682031606172121
monarch.bmp
PSNR:30.04415908317135 MSE:193.1015266363467 SSIM:0.9432861129427074
PSNR:35.13288997238322 MSE:59.82912045533126 SSIM:0.9658650027517576
```

Fig 12: The above image compares the downscaled & recovered high resolution image metrics with respect to the original image

CONCLUSION AND FUTURE WORK

In conclusion, the proposed Super Resolution Convolutional Neural Network (SRCNN) model successfully recovers high-resolution images from their low-resolution counterparts. Through meticulous training and evaluation using metrics like Mean Squared Error (MSE) and Peak Signal-to-Noise Ratio (PSNR), the model demonstrates its effectiveness in enhancing image quality.

The simplicity and robustness of the SRCNN model make it applicable to various low-level vision tasks, such as image deblurring and denoising. By exploring additional filters and advanced training methods, we can further optimize the model's efficiency and performance.

Moreover, this Paper opens up promising opportunities in different sectors. In surveillance applications, the model can assist in improved detection and facial recognition from low-resolution surveillance camera images. In the medical field, the model's ability to generate high-resolution MRI images from lower resolution scans can aid in overcoming imaging challenges. Additionally, in media applications, super resolution offers cost-effective ways to store and deliver media content at different resolutions as needed.

Overall, the SRCNN model proves to be a valuable tool for image enhancement, and its versatility extends to various domains, presenting exciting avenues for future research and real-world implementations.

References:

1. Chao Dong, Chen Change Loy and Xiaoou Tang, " ImageSuperResolution Using Deep Convolutional Networks"arXiv:1501.00092v3[cs.CV] 31 Jul 2015(pp:1-14).
2. Yang, C.Y., Ma, C., Yang, M.H.: Single-image super-resolution: A benchmark. In: European Conference on Computer Vision, pp.372–386
3. Kadari Neeraja and G Narsimha, "Multi-Objective Optimal Path Planning for Autonomous Robots with Moving Obstacles Avoidance in Dynamic Environments" International Journal of Advanced Computer Science and Applications (IJACSA), 13(12), 2022. http://dx.doi.org/10.14569/IJACSA.2022.0131226.
4. Kadari Neeraja and G Narsimha, "A MULTI OBJECTIVE HYBRID COLLISION-FREE OPTIMAL PATH FINDER FOR AUTONOMOUS ROBOTS IN KNOWN STATIC ENVIRONMENTS", Scalable Computing: Practice and Experience, ISSN 1895-1767 , Volume 23, Issues 4, pp. 389–402 https://doi.org/10.12694/scpe.v23i4.2049.
5. Kadari Neeraja, N., Narsimha, G. (2023). A Multi Objective Hybrid Collision-Free Near-Optimal Dynamic Path Planner for Autonomous Robots in Dynamic Environments. In: Garg, L., et al. Key Digital Trends Shaping the Future of Information and Management Science. ISMS 2022. Lecture Notes in Networks and Systems, vol 671. Springer, Cham. https://doi.org/10.1007/978-3-031-31153-6_30.
6. Kadari Neeraja, Dr. G. Narsimha. (2022). Hybrid Dynamic Path Planning Approach for Autonomous Robots in partially Known Dynamic Environments. Computer Integrated Manufacturing Systems, 28(12), 361–380. Retrieved from http://cims-journal.com/index.php/CN/article/view/415.
7. Kadari Neeraja, Dr. G. Narsimha, "Multi Objective Hybrid Dynamic Path Planning Approach for Autonomous Robots in Partially Known Dynamic Environments with Moving Obstacle avoidance"- -2022, Vol,54,Issue:3.pg.No: 805-821, https://www.advancedengineeringsciences.com/article/301.html.
8. Glasner, D., Bagon, S., Irani, M.: Super-resolution from a single image. In: IEEE International Conference on Computer Vision. pp.349–356
9. Freeman, W.T., Jones, T.R., Pasztor, E.C.: Example-based superresolution. Computer Graphics and Applications 22(2), 56–65
10. Kim, K.I., Kwon, Y.: Single-image super-resolution using sparse regression and natural image prior. IEEE Transactions on Pattern Analysis and Machine Intelligence 32(6), 1127–1133
11. Dai, S., Han, M., Xu, W., Wu, Y., Gong, Y., Katsaggelos, A.K.: Softcuts: a soft edge smoothness prior for color image superresolution. IEEE Transactions on Image Processing 18(5), 969–981
12. Burger, H.C., Schuler, C.J., Harmeling, S.: Image denoising: Canplain neural networks compete with BM3D? In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 2392–2399
13. Schuler, C.J., Burger, H.C., Harmeling, S., Scholkopf, B.: A machine learning approach for non-blind image deconvolution. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 1067–1074
14. Ma, Yufeng& Xiang, Zheng & Du, Qianzhou& Fan, Weiguo. (2018). Effects of user-provided photos on hotel review helpfulness: An analytical approach with deep leaning. International Journal of Hospitality Management. 71. 120-131. 10.1016/j.ijhm.2017.12.008.

15. Lakkavaram, V & Raghuveer, Lakkavaram& Kumar, Chatarband& Sri, G & Habeeb, Shaik. (2019). A review on Practical Diagnostic of Tomato Plant Diseases.
16. Gupta, Disha, S. June 29, 2017. Architecture of Convolutional Neural Networks (CNNs) demystified. URL: https:// www.analyticsvidhya.com/blog /2017/06/architecture-of-convolutional-neural-networks-simplified-demystified/
17. Cui, Z., Chang, H., Shan, S., Zhong, B., Chen, X.: Deep network cascade for image super-resolution. In: European Conference of Computer Vision, pp. 49–64 (2014)