Image Depixelizer Using Enhanced Deep Residual Network

Nagothi Moulika, M.Tech Scholar
Dept. of CS&SE, Andhra University College of Engineering, Visakhapatnam, A.P

***Abstract*— Image reconstruction is challenging now-a-days due to the ill-poseness of the inverse problem and very few number of detected photons. As of late profound deep neural networks have been widely and effectively utilized in PC vision errands and attracted in developing areas like medical imaging, astronomy and many more. In this work, we trained a deep residual convolutional neural network to improve image quality by using the existing training datasets information. We form the target work as an constrained enhancement problem and solve it using the convolutional neural network (CNN) algorithm. The trained datasets are used to evaluate the proposed method. The primary point of Image depixelizer is to change over the given low-resolution image into respective high and super resolution image. Experimental results shows that the embedded Image depixelizer is quite robust in face of various low-resolution images and provides good results in terms of resolution**

***Index Terms*—Image Depixelizer, Convolutional neural network (CNN), Residual Neural Network (ResNET), Enhanced Deep Residual Network for Single Image Super-Resolution.**

I. INTRODUCTION

Image Depixelizer is what can take a super low-resolution pixelate photo of an image and transform it into a high resolution representation photo. Though there are some systems which have high sensitivity compared with other imaging modalities, its image resolution and signal to noice ratio (SNR) are still low because of different actual corruption factors and low incidental photon counts identified. Further developing image quality is fundamental, particularly in applications like small lesion identification, brain imaging and longitudinal examinations.

In examining, deep learning methods are acquiring popularity for their accuracy and effectiveness. In contrast to the previous methodologies, the earlier data and regularization are adapted certainly from information, without indicating them in the preparation objective. However, so far a handless approaches exist for dynamic depixelizer. Thus, the relevance of deep learning methods to this issue is yet to be explored. Furthermore, many proposed deep learning architectures are frequently nonexclusive and are not upgraded for explicit applications. Specifically, a center inquiry for dynamic depixelizer is the means by which to ideally take advantage of redundancy. By designing a network architecture which converts low-resolution image to high-resolution without any noise and redundancy which solve this problem.

In this work, we propose a convolutional recurrent neural network (CNN) method to reconstruct high resolution image from under sampled data (low-resolution images). Firstly, we formulate a general optimization problem for solving a blur image based on variable splitting and alternate minimization. Now, we can see this algorithm as a network architecture. In particular, the proposed method consists of a CNN block which produces recurrent connections across each iteration step, allowing DE-pixelate information to be shared across the multiple iterations of the process inorder to form high resolution image.

II. THEORY

A. Existing model

Existing resolution system uses DRCN model. As, increasing the depth of the network will add more parameters, which will lead to two problems. One problem is that it is easy to over fit, and the other issue is that the model is too large to be stored and reproduced. Hence are current neural network is used. It is similar to the proposed model but it is easy to cause network degradation.

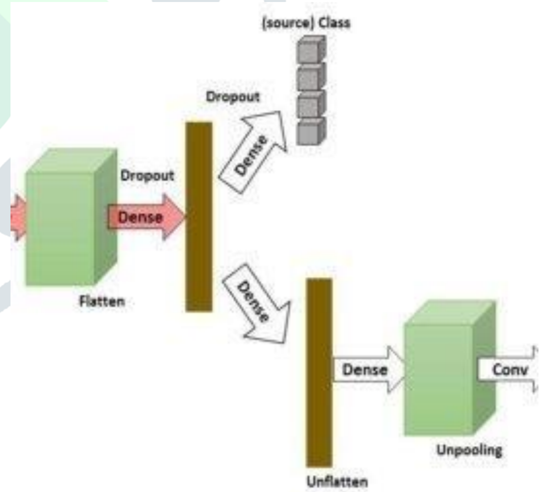


Figure 1 :DRCN model

B. Proposed system using EDSR

By considering the above existing system, a new model is being proposed called, Enhanced Deep Residual Network (EDSR). As we know, most of the information contained in an LR image must be preserved in the HR image. Image resolution model therefore mainly learns the residuals between low-resolution and high-resolution images. This model uses ReLU (Rectified Linear Unit) activation function.

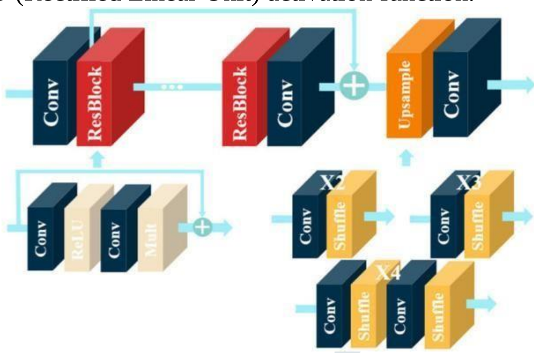
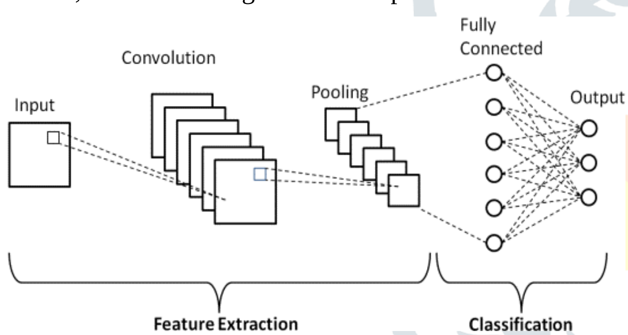


Figure 2:EDSR model

CNN is a kind of deep learning model for preparing information that has a grid design, like pictures, which is motivated by the association of creature visual cortex [13, 14] and intended to naturally and adaptively learn spatial progressive systems of elements, from low to high level examples.



ResNet design utilizes the CNN blocks on various occasions, so let us make a class for CNN block, which takes input channels and output channels. After each and every conv layer, there will be a batchnorm2d. Then, at that point make a ResNet class that takes the contribution of various squares, layers, picture channels, and the quantity of classes.

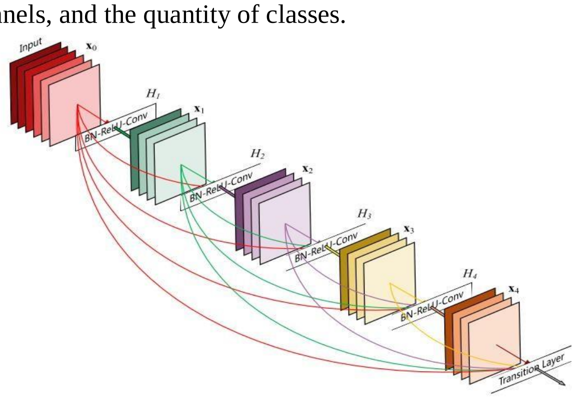


Figure 3 :ResNET model

III. Proposed Algorithm

In the Proposed algorithm we use Rectified linear unit (ReLU) function which is most ordinarily used activation function in deep learning models. The activation function returns 0 if it receives any input as negative, except for any positive value x it returns that value back. So it are often written as f(x)=max(0,x) .

The graphical representation of ReLU is

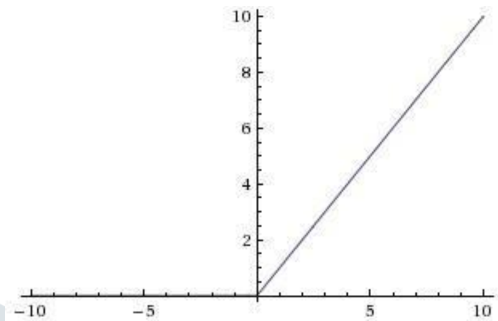


Figure 4 : Graphical Representation of ReLU

Here we have Interaction effects and non-linear effects.

Interactions:

Imagine one node during a neural network model. For simplicity, assume it's two inputs, called A and B. The weights from A and B into our node are 2 and three respectively. therefore the node output is f(2A+3B) . Now apply ReLU function for our f. So, if 2A+3B is positive, the output value of our node is additionally 2A+3B . If 2A+3B is negative, the output value of our node is 0.

For concreteness, consider a case where A=1 and B=1. The output is 2A+3B , and if A increases, then the output increases too. On the opposite hand, if B=-100 then the output is 0, and if A increases moderately, the output remains 0. So A might increase our output, or it'd not. It just depends what the worth of B is.

This is an easy case where the node captured an interaction. As you add more nodes and more layers, the potential complexity of interactions only increases.This is how the activation function helps in capturing an interaction.

Non-linearities:

If the slope is not constant then the function is non-linear . So, the ReLU function is 0 for non-linear, but the slope is either 0 for negative or 1 for positive values.

That's a really limited sort of non-linearity.But two facts about deep learning models allow us to make many various sorts of non-linearities from how we combine ReLU nodes.

First, most models include a bias term for every node. The bias term is simply a continuing number that's determined during model training. For simplicity, consider a node with one input called A, and a bias. If the bias term takes a worth of seven , then the node output is f(7+A). during this case, if A is a smaller amount than -7, the output is 0 and therefore the slope is 0. If A is bigger than -7, then the node's output is 7+A, and therefore the slope is 1.

So the bias term allows us to maneuver where the slope changes. So far, it still appears we will have only two different slopes.However, real models have many nodes. Each node (even within one layer) can have a special value for it's bias, so each node can change slope at different values for our input.When we add the resulting functions copy , we get a combined function that changes slopes in many places.

These models have the pliability to supply non-linear functions and account for interactions well (if which will give better predictions). As we add more nodes in each layer (or more convolutions if we are employing a convolutional model) the model gets even greater ability to represent these interactions and non-linearities.

A. Working :

This structure straightforwardly convolves the first low-resolution small image, remakes it with a deconvolution layer, and adds a fix extraction, non-direct planning and extending layer to accomplish high working velocity without loss of recovery. The working of an image depixelizer is explained in the following steps:

Step1: Upload a low-resolution image.
Step2: Convert the input image into a vector notation.
Step3: Extract the features from the vector.
Step4: Map the pixels in the input image to the images in dataset.
Step5: Reconstruct the image vector after non-linear mapping of pixels in the image.
Step6: Obtained vector result is converted into image format.
Step7: Generation of a high-resolution image.

B. EDSR Model:

In image-resolution, most of the information contained in an LR image must be preserved in the HR image. Image resolution models accordingly chiefly become familiar with the residuals among LR and HR pictures. Therefore, Residual network designs are preferred. EDSR is developed using ResNet architecture. Personality data is passed on through skip associations where as remaking of high frequency content is done on the principle way of the organization.

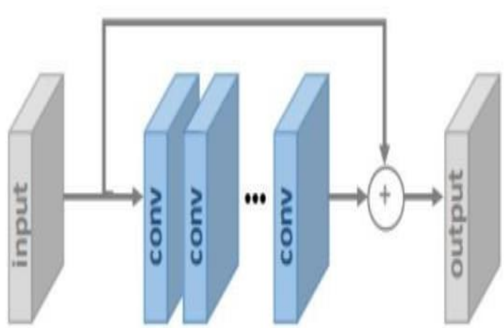


Figure 5 : EDSR model

C. Datasets :

Here we use DIV2K dataset which is high-quality (2K resolution) image dataset for image reconstruction tasks. This dataset consists of 100 validation images, 100 test images and 800 training images. This dataset performs only on the validation datasets and the results are compared.

D. Training :

RGB input patches of 4848 pixels from the LR image, along with the associated HR patches. All of the photos are pre-processed by removing the DIV2K dataset's mean RGB value.Random horizontal flips and 90-degree rotations are added to the training data.For EDSR, the minibatch with a randomly selected scale among 2, 3, and 4 is used for each update. Only the modules that match the selected criteria are shown.

E. Geometric Calculations :

- ILR to create seven augmented inputs from the input image:

$$I_{n,i}^{LR} = T_i\left(I_n^{LR}\right)$$

- Ti denotes the eighth geometric transformation.
- All photos are inverse translated to their original geometry after being super resolved:

$$\tilde{I}_{n,i}^{SR} = T_i^{-1}\left(I_{n,i}^{\hat{SR}}\right)$$

- To produce the self-ensemble result, all of the changed outputs are averaged together:

$$I_n^{SR} = \frac{1}{8}\sum_{i=1}^{8} \tilde{I}_{n,i}^{SR}.$$

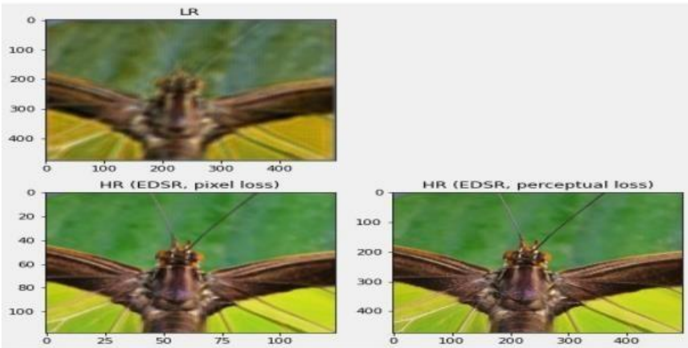
- This self-ensemble method has an advantage over other ensembles in that it does not necessitate further model training. It's especially useful when the model's size or training time is a factor.
- When compared to the traditional model ensemble method, which requires separately trained models, it delivers about the same performance boost.

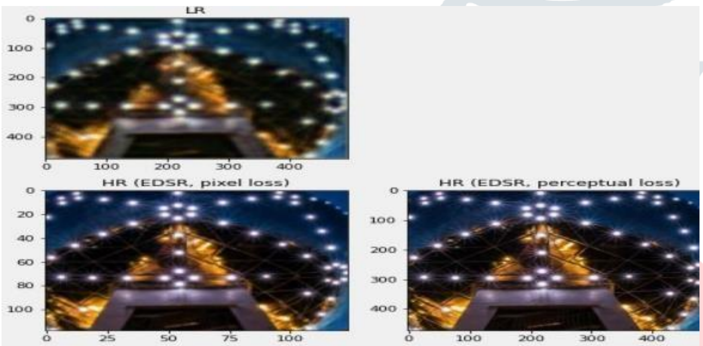
IV. Experimental Results and Testing

The test set forth is evaluation experiment image is randomly selected from the demo folder in the system. Jupyter Notebook software platform is use to perform the experiment. The PC for experiment is equipped with an Intel i3 2.2GHz Personal laptop and 4GBRAM.

The proposed plot is tried utilizing customarily image processing. From the simulation of the experiment results, we can draw to the conclusion that this method is robust to many kinds of testing images.

RESULT :





V. Conclusion

The proposed approach for image resolution attains a remarkable success than the traditional-learning based models. Image depixelizer may be used for many image processing applications, which includes natural image enhancement, surveillance so that automatic recognition systems to improves their performance on low resolution images. Image depixelizer performance can be further increased by exploring more filters while upscaling and different training strategies includes changing the model .

VI. References

[1] R. Timofte, R. Rothe, and L. Van Gool, “Seven ways to improve example-based single image super resolution,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 1865–1873.

[2] S. C. Park, M. K. Park, and M. G. Kang, “Super-resolution image reconstruction: a technical overview,” IEEE Signal Processing Magazine, vol. 20, no. 3, pp. 21–36, 2003.

[3] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” nature, vol. 521, no. 7553, p. 436, 2015.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in Proceedings of the Advances in Neural Information Processing Systems, 2012, pp. 1097– 1105.

[5] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath et al., “Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups,” IEEE Signal Processing Magazine, vol. 29, no. 6, pp. 82–97, 2012.

[6] R. Collobert and J. Weston, “A unified architecture for natural language processing: Deep neural networks with multitask learning,” in Proceedings of the International Conference on Machine Learning, 2008, pp. 160–167.

[7] C.-Y. Yang, C. Ma, and M.-H. Yang, “Single-image super-resolution: A benchmark,” in Proceedings of the European Conference on Computer Vision, 2014, pp. 372–386.

[8] R. Timofte, R. Rothe, and L. Van Gool, “Seven ways to improve example-based single image super resolution,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 1865–1873.

[9] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” arXiv preprint arXiv:1412.6980, 2014.

[10] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification,” in Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1026–1034.

[11] S. C. Park, M. K. Park, and M. G. Kang, “Super-resolution image re construction: a technical overview,” IEEE Signal Processing Magazine, vol. 20, no. 3, pp. 21–36, 2003.

[12] R. Keys, “Cubic convolution interpolation for digital image processing,” IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 29, no. 6, pp. 1153–1160, 1981