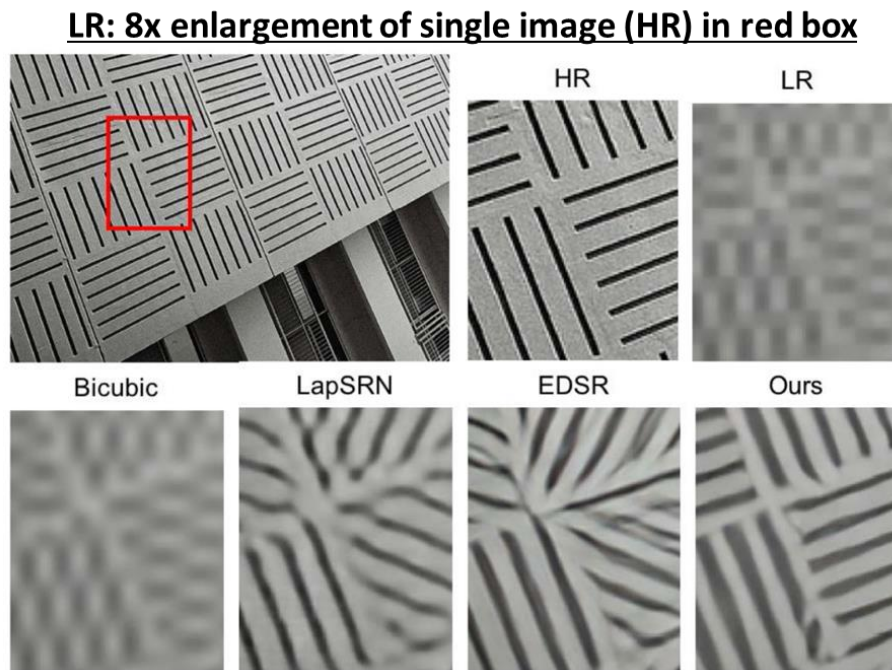


## What is Single Image Super Resolution (SISR)?

SISR is the recovery of high resolution (HR) image from a low resolution (LR) image. This recovery is made possible by learning the non-linear mapping of LR to HR. Till now, learning this non-linear mapping is done via deep neural network (DNN) and non-deep neural network (Non DNN). An example of SISR used to recover HR of a cropped area (magnified 8 times) in the image data is given below [1]:



**Input: LR -> Output: HR**

### **Results of Super-Resolution Methods:**

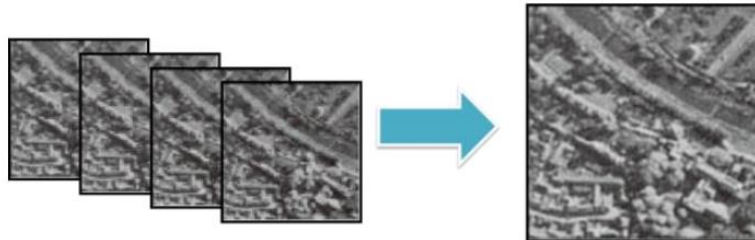
From Bottom Left: Bicubic interpolation, LapSRN [2], EDSR [3], Ours (Author's Method) [1].

In the figure above, the author compared his method (DBPN) with existing ones such as LapSRN and EDSR just to give readers an idea on how much improvement to the resolution of the final output his technique could achieve. Ideally, the LR image (input) should be transformed into the detailed HR image and the difficulty of this increases especially when using large scaling factors on the HR image, in other words, restoring a poorer LR image.

# Importance of SISR

In general, SISR can be applied to any industry that requires improving the resolution of their images. Just to give a few examples, the author from [4] reviewed a few applications of SISR and here are his/her findings:

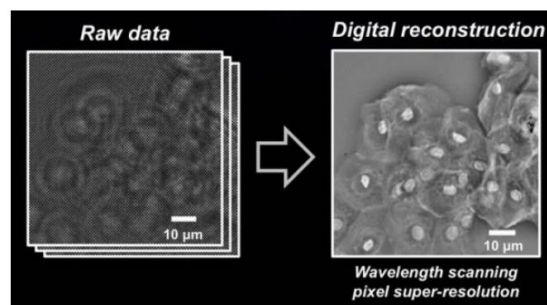
## Satellite Imaging



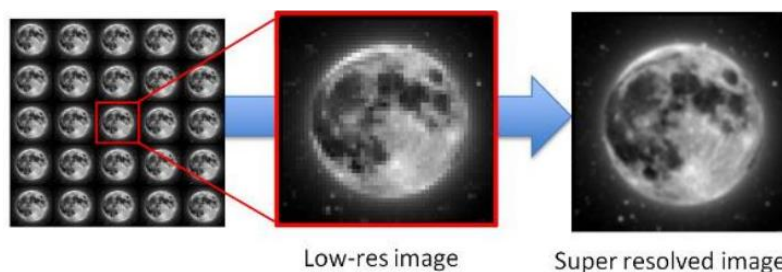
## Medical Imaging



## Microscopy Enhancement



## Astronomy Imaging



What can be deduced from these is if SISR technologies achieve sufficient accuracy and robustness in their performances, they could omit the demand for better hardware for better imaging, which is the current conception in society.

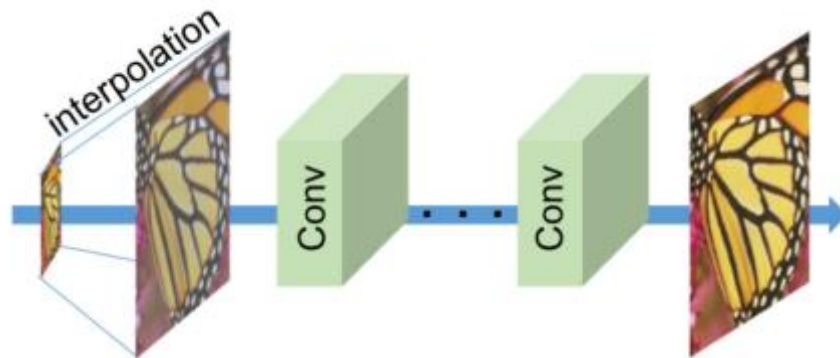
## Related Works on SISR

This section shall explain the past methods used in SISR and describe the author's method.

### Previous Methods:

#### Predefined up-sampling

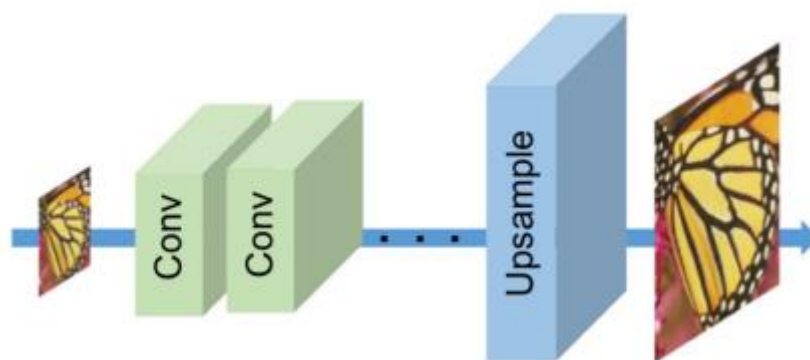
Figure below shows the model architecture for this method:



Predefined up-sampling was first proposed by the authors of SRCNN [5]. The model starts by mapping the LR image to a medium resolution (MR) image using interpolation techniques. The rest of the model can be a deep or non-deep convolutional neural network. The problem with this method is in the usage of MR image, which has the same dimension as the HR image. In other words, MR image input is large and processing this image across the series of convolution is very slow and costly.

#### Single up-sampling

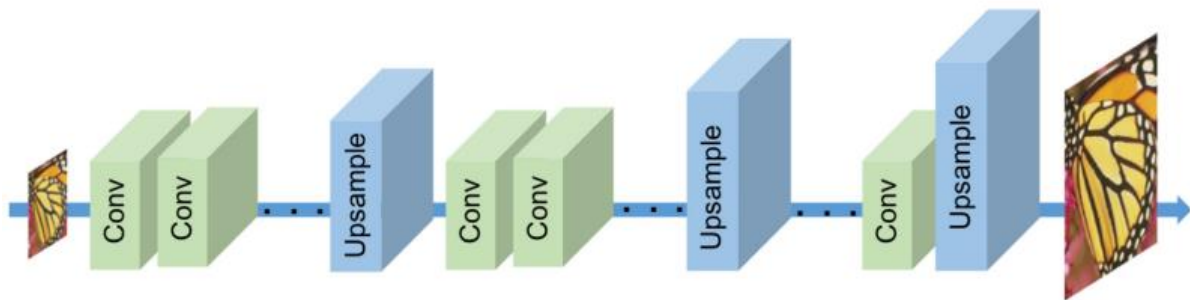
Figure below shows the model architecture for this method:



Proposed in FSRCNN [6] and ESPCN [7]. As depicted in the figure above, the model processes the LR image through a convolutional neural network then finally map the LR image up to the HR image with a single up-sampling layer. The problem with this model is it maps the LR feature space to that of HR very poorly at larger scaling factor as the up-sampling layer relies solely on the low detail LR feature map. Despite that, this method opens door for research in proposing models involving the mutual relation between LR and HR images.

## Progressive up-sampling

Figure below shows the model architecture for this method:

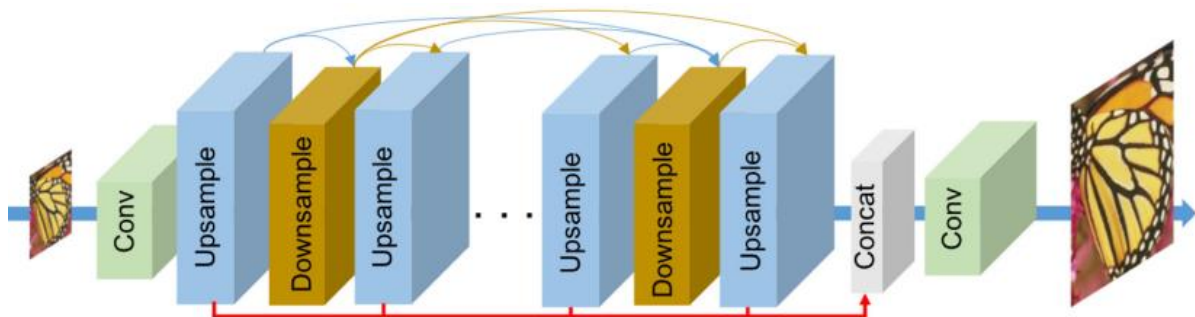


Building up from the single up-sampling method, the authors for LapSRN progressively added up-sampling layer of increasing scale factor at every interval between the convolution layers. However, the use of progressive up-sampling didn't work well as according to the author of DBPN as even the shallower version of DBPN outperformed LapSRN in SISR with 8x scaling factor. This isn't surprising as the progressive up-sampling method, like its predecessor – single up-sampling method, suffers from its reliance on the limited features from the LR image.

## Author's Method:

### Iterative up-/down-sampling

Figure shows the model architecture for the DPBN:



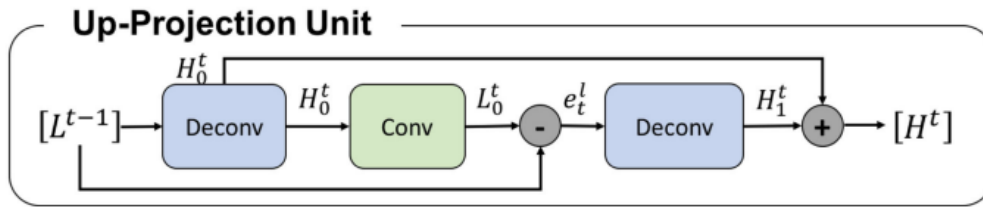
As shown in the title, DBPN stands for Deep Back-Projection Network. The main network of this model is the series of up-sampling and down-sampling layers. The outputs from all up-sampling layers are concatenated to reconstruct the HR image. Moreover, every previous up-sampling/down-sampling layer is concatenated as an input for the next down-sampling/up-sampling layer. The novelty of the author's work doesn't end here. Another significant contribution is the use of back-projection units as the up-sampler and down-sampler, instead of traditional image processing methods. Before explaining the merits of the proposed techniques, it is important to explain the design and concept of the back-projection unit.

## Back-projection

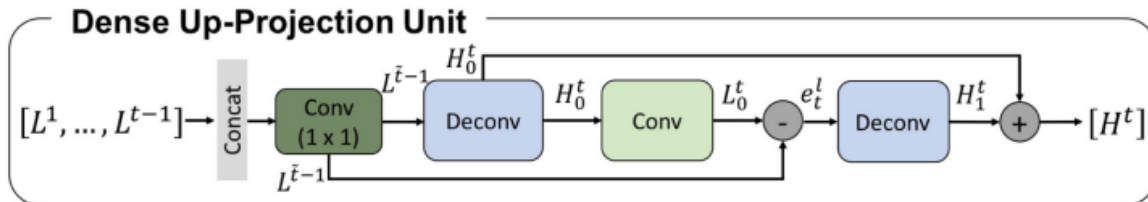
Back-projection is the concept of projecting the output from the model's initial guess back to its input to make corrections to improve the estimation. It was first mentioned in [8] and after that there were a few research works showing the effectiveness of back-projection in super-resolution applications. Most of the previous adopters of back-projection used constant predefined parameters such as the single sampling filter and blur operator. This report will not go through explicitly on the past work and disadvantage of traditional back-projection. In the case of the DBPN author's contribution, a trainable back-projection unit by learning the parameters of the up- and down-sampling operators was implemented.

## Up-projection: The up-sampling operator

This is the projection unit used as an up-sampling operator in DBPN:

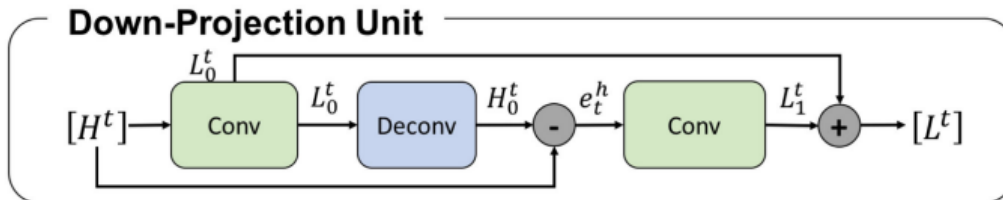


Below is the up-projection unit for the dense DBPN, which will be explained in the later sections:

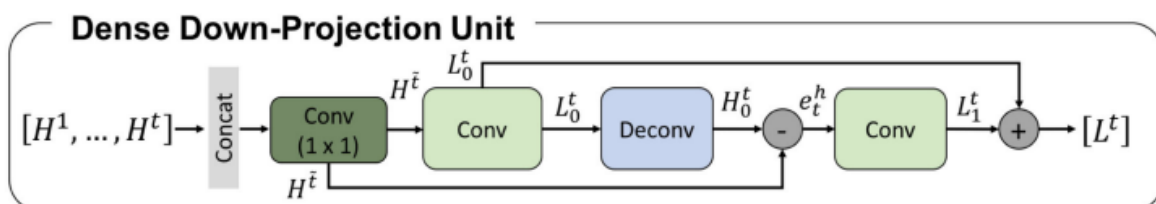


## Down-projection: The down-sampling operator

This is the projection unit used as the down-sampling operator in DBPN:



Below is the down-projection unit for the dense DBPN:



## Details of the projection units

From the figures shown, up-projection projects LR to HR and down-projection projects HR to LR. To make the projection units trainable, a convolution layer is used to produce LR images from HR images whereas a deconvolution layer is used to project LR images to HR images.

The architecture of a projection unit isn't complicated. Below is the explanation of the procedure in an up-projection unit:

1. LR image is first projected to an initial HR image via deconvolution.
2. The estimated HR image is projected to a LR image via convolution.
3. The current LR image will differ from the input LR image due to reconstruction error. This LR reconstruction error, found through the difference between the input LR and the estimated LR, is projected to the HR space to attain a HR reconstruction error.
4. The estimated reconstruction error for the HR space is added to the estimated HR image to further improve the features of the estimated HR image.

The same applies to the down-projection unit, only that the input, output and the order of convolution/deconvolution procedures are different.

## Merits of Author's Work

Here the importance and benefits of the author's proposal are explained as given below:

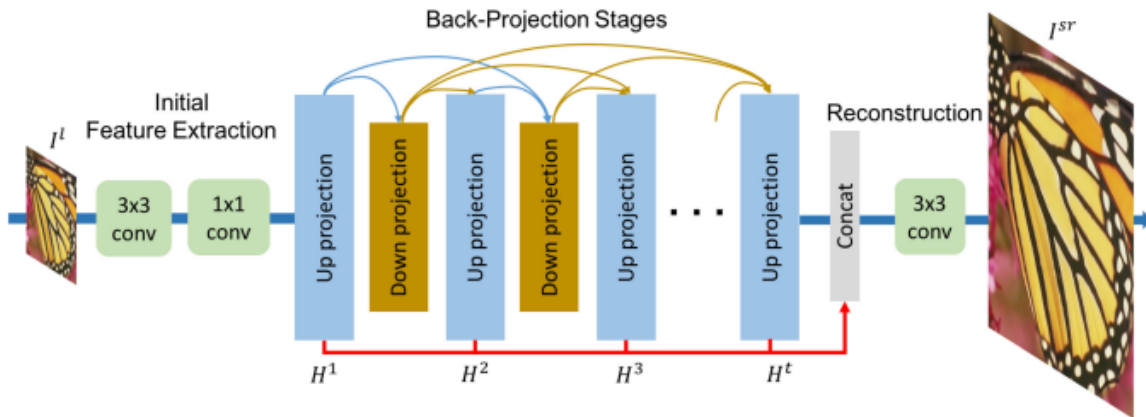
1. **Concatenation of up-sampling layers in reconstructing the output HR**  
Each up-sampling layer carries their own HR feature map refined throughout the training process. By concatenating all the up-sampling layers, all the unique HR feature maps are combined. Different feature map makes up for each other's loss of details. This maximises the important feature mapping from LR to HR.
2. **Concatenation of previous up-/down-sampling layers as inputs for subsequent down-/up-sampling layers**  
Like before, concatenating the previous up-/down-sampling layers maximise the important feature mapping from LR to HR or from HR to LR. Therefore, the subsequent sampling layer could receive input with rich features and this feature richness increases progressively as the concatenation involves more sampling layers.
3. **Trainable back-projection as the sampling layer**  
With the up-/down-projection units, the author of DBPN was the first to introduce error feedback in super-resolution at the time of preparing his journal. Error feedback allows refining the estimated feature mapping via the mutual dependence of HR and LR features within each projection unit.



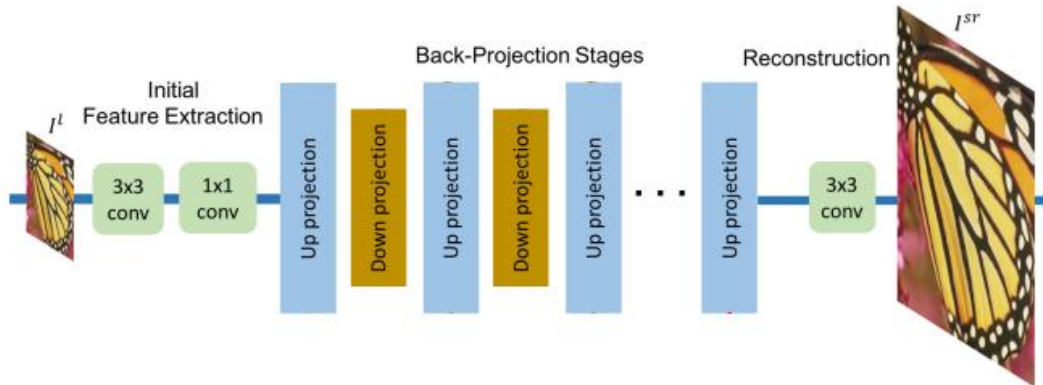
# Modelling of DBPN

The model architecture of the DBPN introduced in the earlier section is a dense DBPN, such that the term dense refers to interconnected sampling layers with concatenation. In the DBPN paper, a lightweight version of DBPN was also introduced, which is the non-dense version. Each version has their own projecting units, as described in the back-projection section. Below shows the dense and non-dense DBPN for comparison purpose:

## Dense DBPN:



## Non-dense DBPN:



There are many variants of DBPN introduced by the author. To reimplement the author's work for demonstration purpose, the DBPN-M variant for the non-dense DBPN and the D-DBPN variant for the dense DBPN were chosen. In addition to reimplementing work, the DBPN-M shall be trained with and without the error feedback to evaluate the effectiveness of the error feedback. In the next subsection, the configurations of each chosen DBPN model are described.

# Model Configuration

This is the configuration for each DBPN variant as highlighted in the author's paper.

	Scale	DBPN-M		D-DBPN	
1 <sup>st</sup> Conv block		Filter size:	3	Filter size:	3
		No. of filter:	128	No. of filter:	256
		Stride:	1	Stride:	1
		Padding:	1	Padding:	1
2 <sup>nd</sup> Conv block		Filter size:	1	Filter size:	1
		No. of filter:	32	No. of filter:	64
		Stride:	1	Stride:	1
		Padding:	0	Padding:	0
Back-projection (BP) Stages (Conv, Deconv)	2x	Filter size:	6	Filter size:	6
		No. of filter:	32	No. of filter:	64
		Stride:	2	Stride:	2
		Padding:	2	Padding:	2
	4x	Filter size:	8	Filter size:	8
		No. of filter:	32	No. of filter:	64
		Stride:	4	Stride:	4
		Padding:	2	Padding:	2
	8x	Filter size:	12	Filter size:	12
		No. of filter:	32	No. of filter:	64
		Stride:	8	Stride:	8
		Padding:	2	Padding:	2
Reconstruction Conv block		Filter size:	1	Filter size:	3
		No. of filter:	1*	No. of filter:	3
		Stride:	1	Stride:	1
		Padding:	0	Padding:	1
Parameters	2x	779		5819	
	4x	1381		10426	
	8x	3101		23205	
Depth		24		52	
No. of BP stage		4		7	
Dense connection		No		Yes	

All convolution and deconvolution blocks except the reconstruction block uses Parametric Rectified Linear Unit (PReLU) as their activation function. No activation function is used for the reconstruction block.

\* In the author's paper, the author used only 1 filter at the reconstruction layer as DBPN-M is only trained on luminance-based data. In this reimplementaion, 3 number of filters are used instead as the DBPN-M is directly trained with RGB data.

Another difference between the author's implementation and mine is that the author used PyTorch as the development framework whereas the latter is done with TensorFlow and



Keras. Hence, the padding layers might not be the same as the author's one as padding can be added automatically in TensorFlow/Keras via setting padding argument to "same" or "valid", instead of specifying a padding size like in PyTorch.

## Model Training

Below shows the model training scheme employed by the author and me.

	<b>Author's</b>	<b>Mine</b>
Dataset	DIV2K	DIV2K
Data Augmentation	Rotation, scaling, cropping and flipping	None
Optimizer (parameter)	Adam (learning rate: 0.0001) (momentum: 0.9)	Adam (learning rate: 0.0004) (momentum: 0.9)
Epoch	1,000,000	100
Scheduling	Reduce learning rate by factor of 10 after half of total epochs	None
Loss	L1 loss	Mean Absolute Error (Mean of L1)




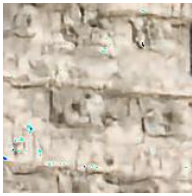













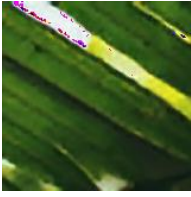


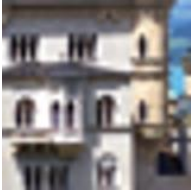
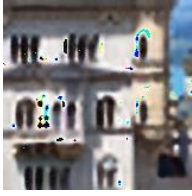
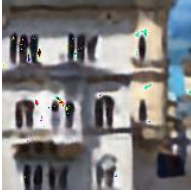


The dataset used is the DIVERse 2K resolution image dataset (DIV2K) [9] which was used in NTIRE 2017 [10] and NTIRE 2018 [11], a Super Resolution challenge. In total, there are 1000 images, 800 for training, 100 for validation and 100 for testing. At the time of accessing the dataset, the last 100 images for testing weren't available. Hence, only 80 images were used for validation and the last 20 images were allocated for testing. All the data used in the reimplementation were cropped in such a way that the input LR images are 40 x 40 pixels whereas the output HR images are 160 x 160 pixels in dimensions. In other words, only a scaling factor of x4 is studied in this reimplementation. None of the image resolution is affected by the cropping process. The goal of this reimplementation is to validate the author's method, so a lightweight training scheme for the SISR models is chosen.

For more information on the reimplementation code and trained models, kindly check out my GitHub repository [here](#).

## Discussions

From the model training, DBPN-M without error feedback is the fastest to train, followed by DBPN-M with error feedback and the slowest is none other than D-DBPN. Unsurprisingly, adding an error feedback calculation or a densely connected projection units will increase the computation time. As no data augmentation was executed and with the limited number of data available from the DIV2K dataset, all three models seemed to exhibit overfitting after the 50<sup>th</sup> epoch, as after this epoch, the validation loss doesn't seem to improve anymore, as opposed to the steadily decreasing training loss. Therefore, in this reimplementation, it is

justifiable not to use long training iterations. Despite that, there are, although sometimes subtle and sometimes distinctive, differences in the transformation results using each model. The table below shows the comparison of the quality of HR image produced from each model using the LR image from the test set. To give a more explicit comparison, the LR image is resized to its HR counterpart using bicubic interpolation, as a representation of what the LR image would look like in the same dimension as the HR image.

	LR (Bicubic)	DBPN-M (No EF)	DBPN-M (with EF)	D-DBPN	HR
Image 1					
Image 2					
Image 3					
Image 4					
Image 5					
Avg. PSNR	22.527	22.925	22.656	22.162	-

Avg. PSNR is the average Peak Signal-to-Noise Ratio (PSNR) usually used to compare the similarity between two images, with the reference image being the HR image in this case. The higher the value of this metric, the closer the image is to its reference. PSNR can be calculated using the formula below:

$$PSNR(Img1,Img2) = 10 \log \left( \frac{max\_pixel\_value^2}{MSE(Img1,Img2)} \right)$$

The *max\_pixel\_value* here is 255 since that is the maximum value a pixel can have in a RGB image. The average PSNR was calculated by using the PSNR for the 20 LR-to-HR transformed images in the test set. Based on the 5 test images in the table above, the average PSNR values depicted in the table shouldn't be taken literally. One of the reasons is due to the lack of proper model training and data augmentation, it can be shown that the three SISR models tend to regress the pixel values erroneously at certain spots. These pixels will affect the mean square error in the PSNR calculation. Besides, with the naked eye, it is quite distinctive that the D-DBPN produces the best resolution, followed by DBPN-M with error feedback and then DBPN-M without error feedback. Therefore, under conducive training conditions, one should observe an increasing trend in PSNR from the non-dense model to the dense model.

## Conclusion

From the results of my reimplementation, there are two deductions can be made:

- Using the reconstruction error to enhance the quality of the output images from each projection unit does produce a better HR or LR feature map. This is proven as DBPN-M performed better with the error feedback than without.
- The concept of densely connected projection units through the concatenation of all HR feature maps across the up-projection layers for the HR reconstruction and the concatenation of previous feature maps as the input of subsequent projection layers were also proven to work in SISR as D-DBPN yielded the best resolution among the three DBPN variants tested.

From these deductions, the author's method is valid. Due to the ingenuity of his method, it is no wonder that DBPN won the NTIRE 2018 and PIRM 2018 challenge. This report serves as a summary of SISR and a lightweight reimplementation of the author's work. For more details, it is highly recommended to give the author's paper a good read.

## References

1. M. Haris, G. Shakhnarovich and N. Ukita, "Deep Back-ProjectiNetworks for Single Image Super-Resolution," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 12, pp. 4323-4337, 2021.
2. W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang, "Deep laplacian pyramid networks for fast and accurate super-resolution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5835–5843.
3. B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced deep residual networks for single image super-resolution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2017, pp. 1132–1140.
4. A. Singh and J. Singh, "Super Resolution Applications in Modern Digital Image Processing", *International Journal of Computer Applications*, vol. 150, no. 2, pp. 6-8, 2016.
5. C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 295–307, 2016.
6. C. Dong, C. C. Loy, and X. Tang, "Accelerating the super-resolution convolutional neural network," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 391–407.
7. W. Shi et al., "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 1874–1883.
8. M. Irani and S. Peleg, "Motion analysis for image enhancement: Resolution, occlusion, and transparency," *J. Vis. Commun. Image Representation*, vol. 4, no. 4, pp. 324–335, 1993.
9. E. Agustsson and R. Timofte, "NTIRE 2017 Challenge on Single Image Super-Resolution: Dataset and Study," *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017, pp. 1122-1131.
10. R. Timofte et al., "NTIRE 2017 Challenge on Single Image Super-Resolution: Methods and Results," *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017, pp. 1110-1121.
11. R. Timofte et al., "NTIRE 2018 Challenge on Single Image Super-Resolution: Methods and Results," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2018, pp. 965-96511.