Journal of Advanced College of Engineering and Management, Vol. 3, 2017

DYNAMIC CONVOLUTIONAL NEURAL NETWORK FOR IMAGE SUPER-RESOLUTION

Anil Bhujel[1], Dibakar Raj Pant[2]

[1]Ministry of Information and Communication, Singhdurbar, Kathmandu

Email Address: anil.bhujel@gmail.com

[2]Department of Electronic and Computer Engineering, Pulchowk Campus, Pulchowk, Lalitpur

Email Address: dibakar@gmail.com

Abstract

Single image super-resolution (SISR) is a technique that reconstructs high resolution image from single low resolution image. Dynamic Convolutional Neural Network (DCNN) is used here for the reconstruction of high resolution image from single low resolution image. It takes low resolution image as input and produce high resolution image as output for dynamic up-scaling factor 2, 3, and 4. The dynamic convolutional neural network directly learns an end-to-end mapping between low resolution and high resolution images. The CNN trained simultaneously with images up-scaled by factors 2, 3, and 4 to make it dynamic. The system is then tested for the input images with up-scaling factors 2, 3 and 4. The dynamically trained CNN performs well for all three up-scaling factors. The performance of network is measured by PSNR, WPSNR, SSIM, MSSSIM, and also by perceptual.

Keywords: *Super-resolution, Convolutional Neural Network, dynamic convolutional neural network*

1. Introduction

The generation of high resolution image from the low resolution image is referred as image Super-Resolution (SR). High resolution image contains large number of pixel density carrying more details of real scene, which is very important in image processing and analysis. The application of high resolution image is common in computer vision in pattern recognition and image analysis. It is also very important in biomedical imaging for diagnosis, analysis of satellite imaging, and in image processing. In many application such as surveillance, forensic and satellite imaging, the specific area of image is need to be zoomed for further analysis, at that time the high resolution has great importance. Single image super-resolution (SR) [1], which creates a high-resolution image from a single low-resolution image has a traditional. This problem is inherently ill-posed since a number of similar high resolution pixels exist for any given low-resolution pixel. This problem is reduced by constraining the solution space using strong prior information. According to the image priors, single image super resolution can be classified into four types, they are prediction model, edge based methods, image statistical methods, and patch based methods which are thoroughly examined byYang et. al. [2] and found the patch based methods perform better among others. These four methods either reduce the internal similarities of the same image pixel or learn mapping functions from external low and high-resolution exemplar pairs [3, 4].

2. Literature Review

Deep Learning for Image Super-Resolution

Dong et al. [5], demonstrated a deep learning method for single image super-resolution (SISR), which directly learns an end-to-end mapping between the low resolution and high-resolution images. The end-to-end mapping is represented as a deep convolutional neural network (CNN) which takes the low-resolution image as the input and produce a high-resolution image as output. It further shows that conventional sparse-coding-based super-resolution method can also be viewed as a deep convolutional network. But unlike traditional methods that handle each component separately.

Single Image Super-Resolution

Glasner et al. [1], is a unified framework which combined both classical multi-image super-resolution and example-based super-resolution. In classical multi-image super-resolution, low resolution image is considered from resampling of high resolution image, so the combination of low resolution sequence images generates high resolution image. It further showed how this combined approach can be applied to obtain super resolution from a single image (with no database or prior examples). Glanser et.al. approach is based on the observation that patches in a natural image which tends to recur redundantly many times inside the image, both within the same scale, or in different scales. Recurrence of patches within the same image scale gives rise to the classical super-resolution, whereas recurrence of patches across different scales of the same image gives rise to example-based super-resolution.

3. Related Theory

Super-resolution is a set of image processing techniques that generates a high-resolution image from multiple low-resolution images or from single low resolution image. A high-resolution image retrieves image details which is not visible in any single low-resolution image.

There are some unavoidable errors occur when an optical image is converted into a digital image, due to conversion of a continuously varying light intensity into a set of pixels each measuring the average amount of light on the small area of each pixel. The reproduction of those digital image are excellent whose optical intensity varies slowly, but the reproduction of those images which contain high frequency feature like edges, corners, zigzags are altered due to aliasing effect. Aliasing effect is the folding or overlapping of high-resolution image information back onto the low-resolution information. It is useful to think the resolution of image in terms of a spatial frequency, which is the number of lines per inch. If we want to record a digital image containing information of a particular spatial frequency the pixels size should be less than half the spatial wavelength of that information. One thousand lines per inch resolution would require pixels less than 0.0005 inches wide, which is called the Nyquist condition.

Convolutional Neural Network

The convolutional layer receives a single input, the feature maps from the previous layer. The layer computes feature maps as its output by convolving filters across the feature maps from the previous layer. These filters are the parameters of the convolutional layer and are learned during training by using back-propagation. During testing, they are held fixed and do not change from one sample to another.

Forward Pass: The learning of network is usually done in batches of T sample. It shall denote by x_i^t, the ith input feature map of sample t and by y_j^t the jth output feature map of sample t. The filters would be denoted by k_{ij}. In the forward pass of the convolutional layer, the output feature maps are calculated using the convolutional operator (denoted by *):

$$y_j^t = \sum_i k_{ij} * x_i^t \qquad (1)$$

Backward Pass: during the backward pass, the convolution layer computes the gradient of the network's loss function l with respect to x_i^t:

$$\frac{\partial l}{\partial x_i^t} = \sum_j \left(\frac{\partial l}{\partial y_j^t}\right) \underline{*} \left(k_{ij}\right) \qquad (2)$$

Where, $\underline{*}$ represents the convolution with zero padding. It uses back-propagation algorithm, so the values of the gradient $\frac{\partial l}{\partial x_i^t}$ are passed to the previous layer which computed x_i^t. Additionally, the gradient of the loss function with respect to k_{ij} is computed:

$$\frac{\partial l}{dk_{ij}} = \frac{1}{T}\sum_t \left(\frac{\partial l}{\partial y_j^t}\right) * \left(\widetilde{x_i^t}\right) \qquad (3)$$

Where $\widetilde{x_i^t}$ is the row/column flipped version of x_i^t. After computing $\frac{\partial l}{dk_{ij}}$, the parameters k_{ij} of the layer are updated by using gradient descent:

$$k_{ij} = k_{ij} - \propto . \frac{\partial l}{dk_{ij}} \qquad (4)$$

Where, $\propto$ is the learning rate.

Dynamic Convolutional Neural Network

In contrast to the convolutional layer, the dynamic convolution layer [8] receives two inputs. The first input is the feature map from the previous layer and the second is the filters. The feature maps are obtained from the input by following a sub-network A. The filters are the result of applying a separate convolutional sub-network B on the input. The output of the layer is computed by convolving the filters across the features maps from the previous layer in the same way as in the convolution layer but here the filters are a function of the input and therefore vary from one sample to other. The whole system is a directed acyclic graph of layers and therefore the training is done by using the back-propagation algorithm.

Forward Pass: During the forward pass, the two networks compute separately. Network A computes the feature maps from the input image which is given to the dynamic convolution network as first input and the separate sub convolution network B computes the filter that will be given to the dynamic convolution network as the second input as shown in Fig 1. The output feature maps are calculated as follows:

$$y_j^t = \sum_i k_{ij}^t * x_i^t \qquad (5)$$

Notice that in contrast to the conventional convolution layer, in the dynamic convolution layer every sample has a different kernel k_{ij}^t.

Backward Pass: In the backward pass, the dynamic convolution layer computes the gradient of the loss function l with respect to x_i^t similarly to before:

$$\frac{\partial l}{\partial x_i^t} = \sum_j \left(\frac{\partial l}{\partial y_j^t}\right) \underline{*} \left(k_{ij}^t\right) \qquad (6)$$

The values of the gradient $\frac{\partial l}{\partial x_i^t}$ are passed to the layer in network A that produces x_i^t. Additionally, and similarly to the conventional convolutional layer, the gradient of the loss function with respect to k_{ij}^t is computed:

$$\frac{\partial l}{\partial k_{ij}^{t}} = \frac{1}{T}\sum_{t}\left(\frac{\partial l}{\partial y_{j}^{t}}\right) * \left(\widetilde{x_{i}^{t}}\right) \qquad (7)$$

In contrast to the convolution layer, k_{ij}^{t} are not parameters of the layer, they are a function of the input t that are passed from a previous layer in network B. Therefore, the values of the gradient $\frac{\partial l}{\partial k_{ij}^{t}}$ are passed to the layer that computed k_{ij}^{t} as part of the back-propagation algorithm.

4. Methodology

Fig 2, shows the purposed system block diagram of Dynamic Convolutional Neural Network (DCNN) for Image Super resolution. The filter generating network [9] determines the size of filter used in convolutional neural network based on the input image. Thus the CNN used here makes adaptive depending upon the input images. CNN is used to generate high-resolution image from the single low-resolution image.

Patch Extraction and Representation

A popular strategy in image restoration is to densely extract patches and then represent them by a set of pre-trained bases such as PCA, DCT, Haar, etc. This is equivalent to convolving the image by a set of filters. The first layer of Convolutional Neural Networkis expressed as an operation F_1:

$$F_1(Y) = \max\,(0, W_1 * Y + B_1) \qquad (8)$$

Where W_1 and B_1 represent the filters and biases respectively, and '*' denotes the convolution operation.

Non-Linear Mapping

The first layer extracts an n_1-dimensional feature maps for an input imageand each of these n_1-dimensional feature maps mapped into an n_2-dimensional one in second layer. This is equivalent to applying n_2 filters which have a trivial spatial support 1 by 1 filters. It is easy to generalize to larger filter sizes like 3 by 3 or 5 by 5. In that case, the non-linear mapping is on a 3 by 3 or 5 by 5 patch of the feature map rather than on a patch of the input image. The operation of the second layer is:

$$F_2(Y) = \max\,(0, W_2 * Y + B_2) \qquad (9)$$

Here,W_2 contains n_2 filters of size n_1 x f_2 x f_2, and B_2 is n_2-dimensional.

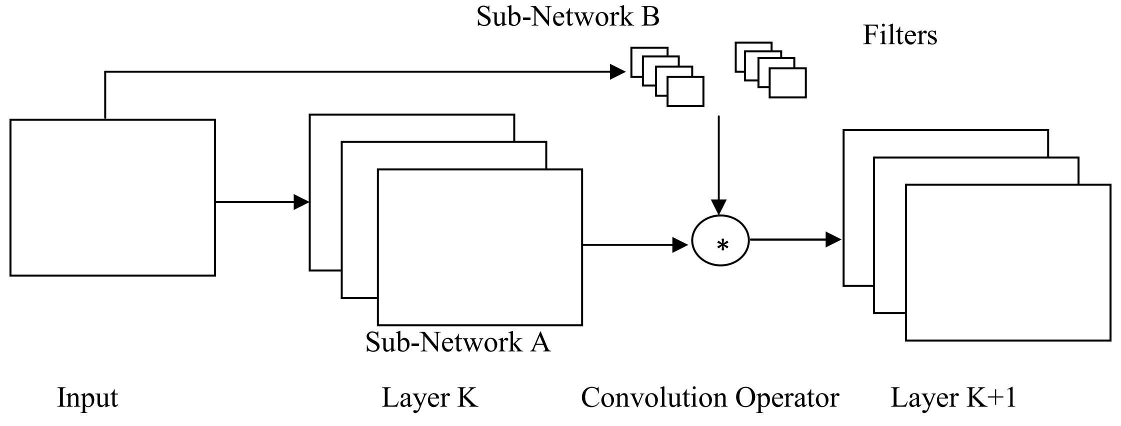


Fig 1 Dynamic Convolutional Layer

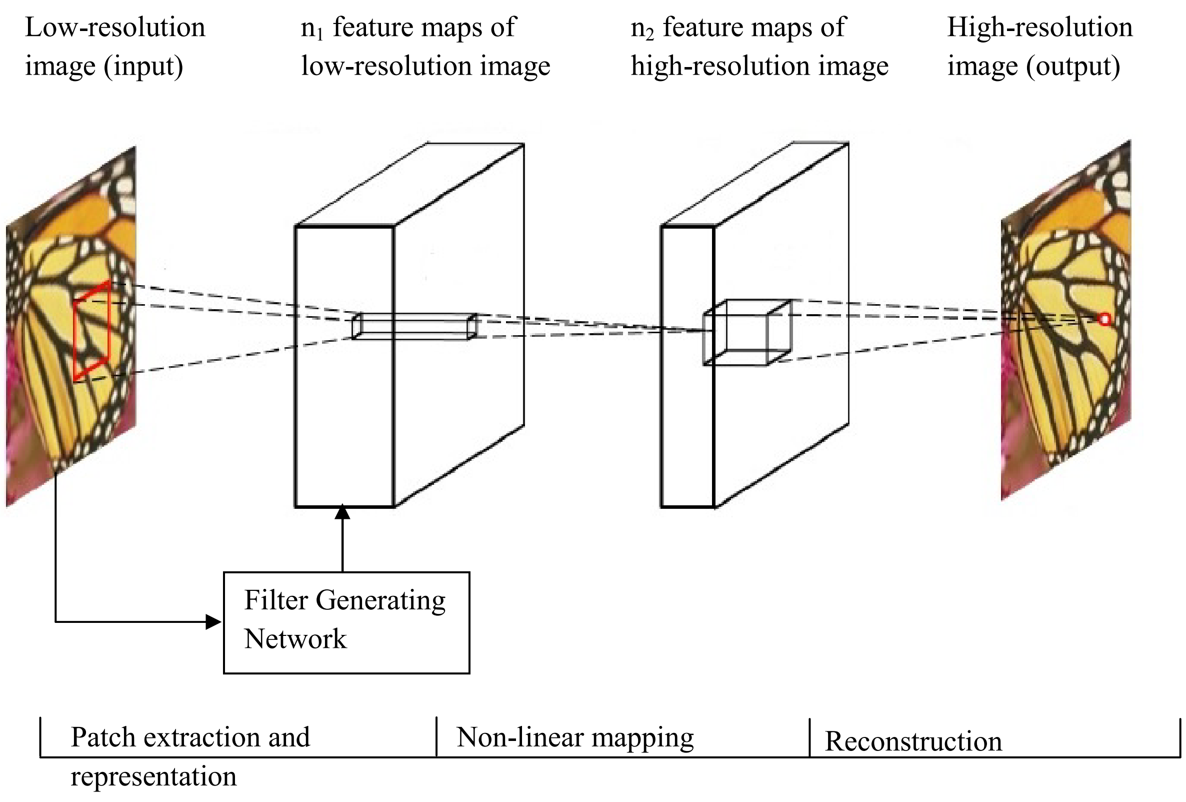

Low-resolution image (input)
n_1 feature maps of low-resolution image
n_2 feature maps of high-resolution image
High-resolution image (output)
Filter Generating Network
Patch extraction and representation
Non-linear mapping
Reconstruction

Fig 2 Dynamic Convolutional Neural Network for Image Super-Resolution Block Diagram

Reconstruction

During reconstruction, the overlapping high-resolution patches that are predicted from the low resolution input image are averaged to produce the final full image. The averaging can be considered as a pre-defined filter on a set of feature maps (where each position is the flattened vector form of a high resolution patch). This motivated to define a convolutional layer to produce the final high resolution image.

$$F(Y) = W_3 * F_2(Y) + B_3 \qquad (10)$$

Here, W_3 corresponds to c filters of a size n_2 x f_3 x f_3, and B_3 is a c-dimensional vector. If the representation of the high resolution patches are in the image domain, then the filter act like an averaging filter.

5. Experiment and Results

A. Training of Network

The training data sets used in convolutional neural network is data sets of 91 images, each images then sub-sampled into sub-images of size 33 by 33, which produces around 24800 sub-images. The network is trained for the up-scaling factor of 2, 3 and 4. For the analysis of effects of large training data sets over limited data sets, the network has been trained with large ImageNet data. There was around 0.39 million images in ImageNet which has been further decomposed around 5 million sub-images.

To synthesize the low resolution samples, the sub images are blur by a Gaussian Kernel, sub sample it by the up-scaling factor, and up-scaled it by same factor via bicubic interpolation. The output images are quite smaller due to avoiding the border pixels during training, whereas the border pixels are padded with zero during testing that makes the same image size with input.

The initial filter weights are chosen randomly from a Gaussian distribution with zero mean and standard deviation 0.001 and the biases are set 0. The learning rate is different for different layer; it is 0.0001 for the first two layers and 0.00001 for the last layer. The smaller learning rate in the last layer helps the network to converge.

B. *Testing of Network*

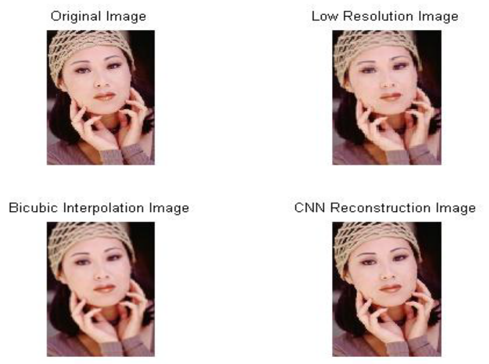
After successfully trained the network, the weights of filters and biases of first layer, second layer and third layer network are fixed. These parameters are extracted and used during the image super-resolution. In the static CNN, the network is trained separately for different up-scaling factor and separate network is used during image super-resolution. But the network is modified to cope with multiple up-scaling factors (2, 3, and 4) and is trained simultaneously with these up-scaling factors, so that a common parameters are extracted which is used during image super-resolution. The trained network is tested using dataset5 (5 pictures) and dataset14 (14 pictures).

C. *CNN over Bicubic*

Table 1 shows the quantitative measurement of bicubic interpolated image and CNN reconstructed image with the ground truth image. It can be seen that the CNN reconstructed image has better quality than bicubic interpolated image for all up-scaling factor 2, 3 and 4. As the number of up-scaling factor increases, the quality of reconstruction decreases. Table 1 shows, CNN reconstructed image has 2.997 dB higher PSNR, 9.262 dB higher WPSNR, 0.0243 dB higher SSIM, and 0.0031 dB higher MSSSIM than bicubic interpolation for up-scaling 2. Similarly, the CNN reconstructed image showed better quality than bicubic interpolation for up-scaling 3, and 4 as well. The quality of images seen degraded as the up-scaling factor increases; this is because large number of high resolution pixels has to be predicted from few number of input image pixels.

D. *CNN on Different Color Model*

The images in different color model are tested. The results obtained from Ycbcr, HSV and RGB color model is shown in Table 2. The network exhibited a bit better in Ycbcr model than HSV and RGB, since the network is trained by Ycbcr model. The CNN reconstructed image of butterfly has PSNR 6.551 dB higher than HSV and 4.423 dB higher than RGB color space, similarly we can see in all other measurement indices, the value obtained from the Ycbcr color model has higher than that of HSV and RGB color model. Fig 3 is the output obtained in RGB color model, whereas the Fig 4 represents the output in Ycbcr color model.



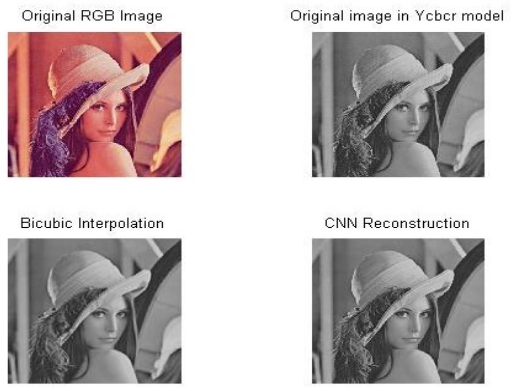
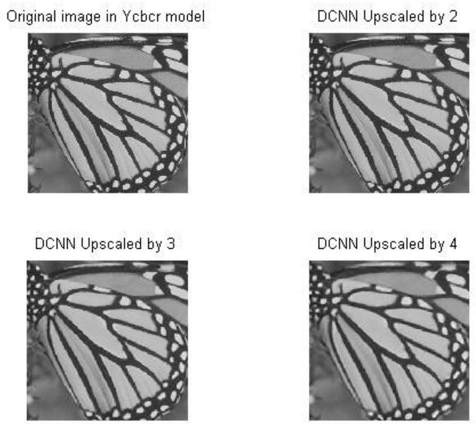


Fig 3.1 Reconstruction on RGB Color Model

Fig 3.2 Reconstruction in Ycbcr Model



Courtesy: Dong et.al. Fig 3.3 Output of DCNN

E. Static CNN vs Dynamic CNN

In the conventional CNN, the network is trained separately for separate up-scaling factor (x_2, x_3, x_4) and parameters for each up-scaling factor is separate. During the testing particular parameters has to be used to get better result. But the network is modified to cope with dynamic up-scaling factor (x_2, x_3, x_4) and network is trained with multiple up-scaled training data to get a common parameters for these three up-scaling factor. Once the network is trained it works finely in all three up-scaling factors. Table 3 demonstrates the PSNR and SSIM value of output generated by static CNN and dynamic CNN.

The results in Table 3 are compared for the training and testing of network by different up-scaling factors. The performance of static CNN found better when the training and testing up-scaling factor matched and degraded the results when the up-scaling factor between training and testing is mismatched. But the dynamic network is trained by the multiple up-scaling factor simultaneously gives the satisfactory result in all three up-scaling factors. The performance of DCNN for a particular up-scaling factor is closed to the performance of static CNN for the same training and testing up-scaling factor but far better than static CNN in different training and testing up-scaling.

The bold values in static CNN show the better result when network is matchedfor up-scaling factor, for example, network trained by up-scaling factor 2 is tested by image up-scaled by 2.

From the Table 3, the PSNR and SSIM value for the static CNN trained and test by same up-scaling factor has higher than that of all, but the value degraded drastically for the mismatch between training and testing up-scale factor. Unlike in static CNN, the dynamic CNN performs well in all three up-scaling factors. The PSNR for testing up-scale 2 of static CNN trained by up-scale 2 has 36.659 dB whereas the DCNN has 36.344 dB. When the testing up-scale factor is 2, the PSNR of static CNN dropped to 29.435 dB but the DCNN has 32.394 dB, similarly for up-scale factor 4, PSNR of static CNN further dropped to 25.267 dB whereas the DCNN drop few and becomes 30.086 dB. The output result is shown in Fig 5.

Table 1 Image super-resolution using CNN and Bicubic Interpolation on Set 5 dataset

Input Images	Up-scale	PSNR (dB)		WPSNR (dB)		SSIM (dB)		MSSSIM (dB)	
		Bicubic	CNN	Bicubic	CNN	Bicubic	CNN	Bicubic	CNN
baby	2	37.066	38.537	53.512	61.812	0.9514	0.9651	0.9958	0.9979
Bird	2	36.808	40.915	51.016	61.142	0.9720	0.9859	0.9974	0.9991
butterfly	2	27.434	32.753	43.714	54.744	0.916	0.9652	0.9932	0.9980
Head	2	34.858	35.723	54.383	61.597	0.862	0.8862	0.9891	0.9932
Woman	2	32.145	35.365	48.002	57.642	0.9478	0.9686	0.9952	0.9981
average	**2**	**33.662**	**36.659**	**50.125**	**59.387**	**0.9299**	**0.9542**	**0.9942**	**0.9973**
baby	3	33.911	35.250	44.975	49.408	0.9030	0.9233	0.9832	0.9889
Bird	3	32.576	35.475	42.509	47.704	0.9257	0.9550	0.9856	0.9931
butterfly	3	24.038	27.953	35.252	41.838	0.8241	0.9121	0.9724	0.9898
Head	3	32.880	33.712	46.171	50.192	0.7991	0.8267	0.9709	0.9786
Woman	3	28.564	31.371	39.354	45.603	0.891	0.9297	0.9797	0.9901
average	**3**	**30.394**	**32.752**	**41.652**	**46.949**	**0.8686**	**0.9094**	**0.9784**	**0.9881**
baby	4	31.777	33.126	40.253	43.292	0.8556	0.8815	0.9691	0.9793
Bird	4	30.180	32.520	38.032	41.645	0.8737	0.9115	0.9715	0.9843
butterfly	4	22.099	25.459	30.921	36.200	0.7407	0.8620	0.9501	0.9808
Head	4	31.590	32.444	42.009	44.641	0.7513	0.7784	0.9559	0.9666
Woman	4	26.464	28.895	34.890	39.124	0.8350	0.8868	0.9623	0.9798
average	**4**	**28.422**	**30.489**	**37.221**	**40.981**	**0.8113**	**0.8641**	**0.9618**	**0.9782**

Table 2 Image Super Resolution on Different Color Spaces

Input Images	Up-scale	Measurement	Ycbcr		HSV		RGB	
			Bicubic	CNN	Bicubic	CNN	Bicubic	CNN
Butterfly	3	PSNR (dB)	**24.038**	**32.753**	22.768	26.202	24.038	28.330
		WPSNR (dB)	**35.252**	**41.838**	-	-	34.153	41.057
		SSIM (dB)	**0.8241**	**0.9121**	0.8282	0.9109	0.8282	0.9110
		MSSSIM (dB)	**0.9724**	**0.9898**	0.9741	0.9900	-	-
Woman	3	PSNR (dB)	**28.564**	**31.371**	27.216	30.077	28.563	31.311
		WPSNR (dB)	**39.354**	**45.603**	-	-	38.043	44.110
		SSIM (dB)	**0.891**	**0.9297**	0.8748	0.9158	0.8748	0.9158
		MSSSIM (dB)	**0.9797**	**0.9901**	0.9765	0.9880	-	-

The bold values represent the better results in Ycbcr color model among three other color models.

Table 3 Result of static CNN and dynamic CNN for up-scaling factor 2, 3 and 4

Test/ Train	Static CNN x_2		Static CNN x_3		Static CNN x_4		DCNN $x_{2,3,4}$	
	PSNR (dB)	SSIM (dB)	PSNR (dB)	SSIM (dB)	PSNR (dB)	SSIM (dB)	PSNR (dB)	SSIM (dB)
x_2	**36.659**	**0.954**	30.575	0.874	28.442	0.813	**36.344**	**0.952**
x_3	29.435	0.884	**32.752**	**0.909**	29.002	0.830	**32.394**	**0.904**
x_4	25.267	0.766	28.722	0.847	**30.489**	**0.8641**	**30.086**	**0.854**

6. Conclusion

We designed a convolutional neural network to cope with dynamic range of up-scaling factors, and named the system dynamic convolutional neural network (dynamic CNN), the performance of dynamic CNN is close to the static CNN tested for same training up-scale, but improves quality of super-resolution than that of static CNN implemented in mismatched training and testing up-scale. This algorithm provides solution to arduous task in designing separate convolutional neural network for different up-scaling factors.

References

1. Glasner D., Bagon S., and Irani M., *"Super-resolution from a single image."* In ICCV, 2009.
2. Yang C. Y., Ma C., and Yang M. H., *"Single image super-resolution: a benchmark."* In ECCV, pages 372-386, 2014.
3. Chang H., Yeung D. Y., and Xiong Y., *"Super-resolution throughneighbor embedding."* In: IEEE Conference on Computer Vision and Pattern Recognition, 2004.
4. Kim K. I. and Kwon Y., *"Single Image super-resolution using sparse regression and natural image prior."* IEEE TPAMI, 32(6): 1127-1133, 2010.
5. Dong C., C. Loy C., He K., and Tang X., *"Image super-resolution using deep convolutional neural networks."* IEEE TPAMI, 2015.
6. Yang J., Wang Z., Lin Z., Cohen S., and Huang T., *"Coupled dictionary training for image super-resolution."* IEEE TIP, 19(11): 1-8, 2010.
7. Yang J., Wright J., Huang T., and Ma Y., *"Image super-resolution via sparse-representations."* IEEE TIP, 19(11): 1-8, 2010.
8. Klein B., Wolf L., and Afek Y., *"A dynamic convolutional layer for short range weather prediction."* In CVPR, pages 4840-4848, IEEE, 2015.
9. Brabandare B. D., Jia X., Tyutelaars T., and Gool L. V., *"Dynamic Filter Networks."* arXiv preprint arXiv:1605.09673v2, 2016.