Pushing and Bounding Loss for Training Deep Super-Resolution Network

Shang Li[1,2], Guixuan Zhang[3], Jie Liu[4], Shuwu Zhang[1,2]
[1] School of Artificial Intelligence, University of Chinese Academy of Sciences (UCAS)
[2] Institute of Automation, Chinese Academy of Sciences (CASIA)
[3] Key Laboratory of Digital Rights Services (KLDRS)
[4] Beijing Engineering Research Center of Digital Content Technology (BJDCRC)
{lishang2018, guixuan.zhang, jie.liu, shuwu.zhang}@ia.ac.cn

***Abstract*—As deep neural networks (DNNs) are hard to be trained due to gradient vanishing, intermediate supervision is typically used to help earlier layers to be better optimized. Such deeply supervised methods have proved to be beneficial to various tasks such as classification and pose estimation, but it is rarely used for image super-resolution (SR). This is because intermediate supervision needs a set of intermediate labels, but in SR, these labels are hard to be defined. Experiments show that identity labels across the whole network, which are used for classification, will cause inconsistence and harm the final performance. We argue that 'mediately accurate' labels, *i.e.*relatively soft labels, are more suitable for intermediate supervision on SR networks. But labels in SR networks are of either completely high resolution or completely low resolution. To address this problem, we propose what we call pushing and bounding loss, which forces the network to learn better features as it goes deeper. In this way, we do not need to explicitly give any 'mediately accurate' labels but all internal layers can also be directly supervised. Extensive experiments show that deep SR networks trained in this scheme will receive a stable gain without adding any extra modules.**

I. Introduction

Super-resolution (SR) task aims to restore high-resolution (HR) images from their corresponding low-resolution (LR) versions. In this paper, we are interested in single image super-resolution (SISR), in which case, only one LR image is given. Since there are infinite HR solutions for a given LR image, this problem is ill-posed and challenging. But due to its wide applications in security surveillance, medicine imaging, video enhancement and so on, SR is always an active research filed.

With the development of convolutional neural networks (CNNs), methods based on CNN, especially deep CNN, have also achieved remarkable results on SR. Since CNN is firstly applied to SR in [1], SR networks have become much larger and deeper. However, it is usually hard to train deep neural networks (DNNs) because of gradient vanishing or explosion. To address this problem, deep supervision is usually used in many tasks, such as classification [2], [3], pose estimation [4], [5] and edge detection [6].

However, although benefits of deep supervision have been proved in various models [3], [7], [8], it is rarely used in SR networks. This is because labels for intermediate supervision in SR networks are hard to be defined. In classification networks, intermediate labels are the same as the final ground truth, but in SR networks, we experimentally find that this kind of supervision does not work. We infer that in classification networks, the optimizing targets of early and deep features are the same, but features in SR networks have more diverse representations and thus cannot be supervised by the same HR ground truth. The same problem is also encountered in pose estimation and edge detection. Thus, a set of 'progressively accurate' labels are required for coarse-to-fine intermediate supervision. In pose estimation networks, 'mediately accurate' labels are generated by adjusting the size of gaussian kernel of heatmaps [8], [9], and in edge detection networks [6], relatively coarse labels are generated by adjusting the threshold of Canny algorithm [10]. But in SR networks, labels are of either completely HR or complete LR. Those mediately high resolution labels are hard to be defined.

To address this problem, we try to supervise internal layers in SR networks without explicitly giving intermediate labels. Firstly, we reconstruct the internal features to HR images, which are called intermediate results. Intuitively speaking, we hope these intermediate results become better and better as the network goes deeper and deeper. Thus, we calculate respectively the distances between these intermediate results and ground truth, and we force these distances to become smaller and smaller via what we call pushing and bounding loss. In other words, as the network goes deeper, we force these intermediate results to get closer to the ground truth, and the final result will be the best one. This training scheme has two benefits. Firstly, internal layers can be directly supervised, and even early layers can be fully optimized. Secondly, we do not force the network to give a best result at once. Instead, we lead it to give better and better result via our pushing and bounding loss. This supervision is relatively soft, and makes it easy for the network to converge. As a result, without adding any extra modules, deep residual networks trained via our pushing and bounding loss can receive stable gain. Although the training process becomes more complex, the inference cost keeps the same and the model will have better performance.

Our contributions can be summarized as two points:

1. We propose a method to better optimize deep super-resolution networks, and deep SR networks trained via our method can receive stable gain without adding any extra modules.
2. We propose pushing and bounding loss which can super-

vise internal layers without explicitly giving any intermediate labels. And this supervision is soft but direct, and can help deep SR networks converge to a better point.

II. Related Work

A. Deep Learning for Single Image Super Resolution

Since the proposal of SRCNN in [1], convolutional neural networks (CNNS) for super resolution (SR) have become much deeper and larger. Although deep CNNs are hard to be trained, many techniques are proposed to solve this problem. In [11], a global residual connection is added from the beginning to the end, and thus gradient can flow directly to early layers. In [12], residual learning is formally introduced to image recognition and [13] immediately applies this structure to SR networks. Multiple residual connections encourage smoother gradient flow and these connections make it possible to train CNN over 100 layers without obvious optimizing problem. As a result, later SR networks, such as EDSR [14], RDN [15], DBPN [16], RCAN [17] *et al*., all have more than 100 layers. RCAN, with the help of residual in residual (RIR) structure, can even have more 500 layers. It is true that various residual connections can guarantee a good result, however, we experimentally show that better results can be achieved if very deep SR networks can be further optimized.

B. Deeply Supervised Convolutional Neural Networks

Deep supervision is firstly introduced in [2] to help train a very deep neural network. In this paper, not only the final output features are supervised, but also the internal features at different depths are supervised by individual classifiers. Later in [3], the benefit of deep supervision is formally illustrated. It proves that intermediate supervisions not only solve the problem of gradient vanishing or explosion, but also act as a kind of regularization and help the network converge to a better point. Given the remarkable achievement of deep supervision in image classification, it has also been applied to various other tasks. In [4], intermediate supervision is used to train a multi-stage networks for pose estimation. And later in [5], intermediate supervision is furtherly developed to a default setting for pose estimation. In the field of edge detection, [6] adopts intermediate supervision to help the network learn to reduce the false positive edges in final edge detection maps. Although deep supervision has excellent performance in various task, it is rarely used in deep super resolution networks because the intermediate labels for SR networks are hard to defined. Thus, we propose the pushing and bounding loss, which share the benefit of deep supervision but do not need explicit intermediate labels. And this loss can help train deep SR networks to a better state.

III. Proposed Method

A. Formulation

To make a clear illustration, we firstly formulate general SR networks. As shown in Figure 1 (a), a plain SR network consists of three parts, *i.e*.head, body and tail. The head,

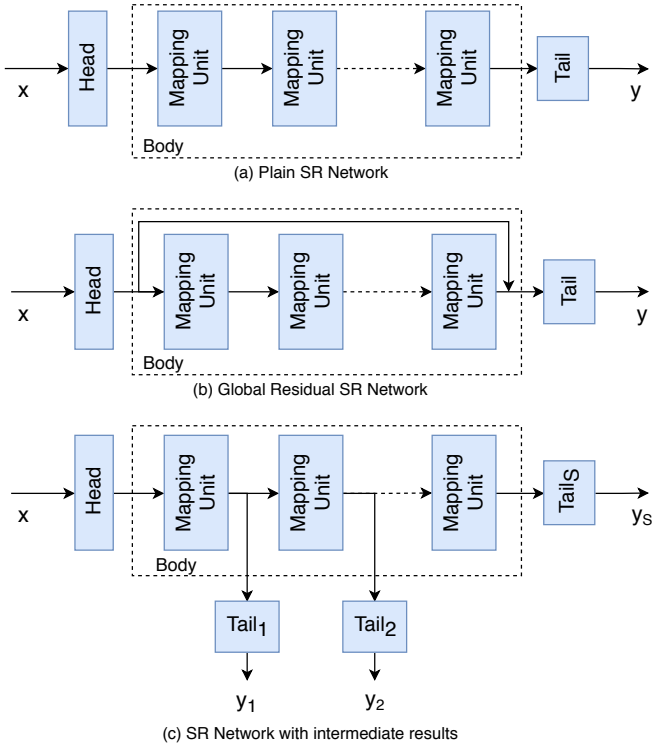

Fig. 1. Dfferent SR network structures

denoted as $H(\cdot)$, extract low-resolution (LR) features from the original LR image. This stage can be expressed as

$$f_{LR} = H(x) \tag{1}$$

where f_{LR} denotes LR features and x denotes the LR image. Then the body, which usually consists of S mapping units, maps LR features to high-resolution (HR) features. This stage can be expressed as

$$f_i = M_i(f_{i-1}) \qquad 1 \leq i \leq S \tag{2}$$

where $M_i(\cdot)$ denotes the i^{th} mapping unit and f_i denotes the its output. And the input of the first mapping unit is also the extracted LR features. The output of the last mapping unit is also the HR features. So we have $f_0 = f_{LR}$ and $f_S = f_{HR}$.

At last, the tail, denoted as $T(\cdot)$, reconstructs HR features to the HR image y. This process can be expressed as

$$y = T(f_{HR}) \tag{3}$$

The reconstructed result is then supervised by the ground truth GT.

$$J = |GT - y| \tag{4}$$

where J denotes the final reconstruction loss and $|\cdot|$ denotes L1 norm.

B. Different Kinds of Deep Supervision

1) Deeply Supervised by the Ground Truth: As we have mentioned above, deep supervision has proved to be beneficial to training deep convolutional neural networks (CNNs). However, this technique is rarely used in SR models. If we want to

take the advantages of deep supervision in SR networks, the most direct way is supervising internal features with ground truth, just like the way adopted in classification networks. Firstly, we need to reconstruct internal features f_i to 'HR images', which are called intermediate results. Then we have

$$y_i = T_i(f_i) \tag{5}$$

where y_i denotes the intermediate result of the i^{th} internal feature, and $T_i(\cdot)$ denotes the tail we used to reconstruct the i^{th} internal feature. If all of these intermediate results are supervised by the ground truth, then the intermediate loss I can be written as

$$I = \sum_{1}^{S} |y_i - GT| \tag{6}$$

For convenience, this is called IGT loss.

2) Deeply Supervised by Blurred Ground Truth: The first kind of deep supervision seems to be fair reasonable, since it works for image classification and pose estimation. However, we experimentally find if all hierarchical features are supervised by the same label, it will cause inconsistence across the whole network and harm the final performance. Thus, it may be more reasonable to supervise these intermediate results with different labels. Intuitively, as the network goes deeper, the corresponding intermediate results should become better, and so are the labels. Motivated by this assumption, we blur GT with Gaussian kernels of different sizes to generate intermediate labels with different accuracies. In formula,

$$L_i = B(GT, K_i), \quad s.t. \quad 1 \leq i < S \tag{7}$$

where L_i denotes the i^th intermediate label and $B(\cdot)$ denotes the Gaussian blur function. K_i denotes the kernel size that used to generate L_i. Since we want the labels to get better as i increases, we need to add a constraint $K_{i+1} < K_i$, with $1 \leq i < S$. Then we supervise y_i with L_i and the intermediate loss I can be written as

$$I = \sum_{1}^{S} |y_i - L_i| \tag{8}$$

For convenience, this is called IBGT loss.

3) Deeply Supervised by Pushing and Bounding Loss: Although we can generate different labels by blurring the ground truth with different kernel sizes, there are two concerns about it. Firstly, the kernel sizes are manually set, and different setting may lead to very different result. Since we do not know the best setting in advance, it may take a lot of time to adjust these hyper-parameters. Secondly, these blurred labels actually force the network to learn to deblur, instead of magnifying the resolution, and these two processes are essentially different. Experimental results also suggest that this kind of supervision does not work.

To solve these problems, we need to come back to the original motivation, without introducing any other unnecessary assumptions. The only hypothesis is that the intermediate results should become better as the network goes deeper, and we only need to add this constraint to the SR network. As the quality of a reconstructed HR image is defined via its distance between the ground truth, we simply force the distances between these intermediate results and the ground truth to become smaller and smaller. In formula,

$$d_i = |y_i - GT| \tag{9}$$

And to implement the constraint $d_{i+1} \leq d_i$, where $1 \leq i < S$, we propose the pushing loss, which can be written as

$$P = \sum_{1}^{S-1} max(0, d_{i+1} - d_i + m_i) \tag{10}$$

where m_i is the maximum margin between the i^{th} and i^{i+1} intermediate results. In practical, we simply choose $m_i = m$, and m is a constant number. We experimentally find that as long as m is not too small, we can always get a robust result. Of course, this constraint is not strong enough, since it only restricts the order of $\{d_i\}$, but do not restrict their ranges. Thus, we need to add one upper bound for d_1. The bounding loss B can be written as

$$B = max(0, d_1 - U) \tag{11}$$

where U is the upper bound of d_1. In practical, we set $U = (S-1)m$. As a result, the optimizing target for d_S becomes 0, which is exactly the desired case. The total loss is

$$\begin{aligned} loss =& \alpha(P + B) + \beta J \\ =& \alpha(\sum_{1}^{S-1} max(0, d_{i+1} - d_i + m) \\ &+ max(0, d_1 - (S-1)m)) + \beta d_S \end{aligned} \tag{12}$$

where α and β are the weight of intermediate loss and reconstructing loss respectively. In practical, we set α as 1 and β as 10.

The pushing loss forces the network to learn better and better features, and the bounding loss restricts the worst case. These two losses together force the final result to get closer to ground truth, and their optimizing target is the same as the reconstructing loss, which forces the network to converge to a better point. In this way, we do not explicitly give any intermediate labels but we do force the network to learn better and better intermediate results, without introducing any other assumptions. Although there are also some hyper-parameters, their influences are limited. We also experimentally prove that as long as the margin m is not too small, the final result is not sensitive to these hyper-parameters.

C. Model Setting

Above discussion does not involve any details about the SR network. In fact, the losses mentioned above can be applied to general deep SR networks. Without loss of generality, we conduct all of our experiments on the state-of-the-art model, RCAN, which is proposed in [17]. We choose RCAN because it has simple and flexible structure, which is easy to be modified. Additionally, improvement on state-of-the-art result is more meaningful.

As show in Figure 1 (b), RCAN uses a global skip connection to help the gradient flow to the early layers. The mapping unit of RCAN is called residual group (RG), which consists of a sequential of residual channel attention blocks (RCABs). This structure is called residual in residual structure.The head of RCAN is simply one convolutional layer with 3×3 kernel size. It tail is composed by PixelShuffle [18] layers. In our experiments, since we have introduced intermediate loss to help train the network, we omit the global skip connection, which serves similar roles. We keep the structure of residual group and residual channel attention block. Although in original RCAN, multiple skip connections greatly benefit the training process, experiments show that its result can be further improved. We denote different RCAN structure as RCAN-$g \times b$-wc, which means that it has g residual groups and each group has b RCABs. And the number of channels is c. If the RCAN is trained by pushing and bounding loss, it is called PB-RCAN.

IV. Experiments

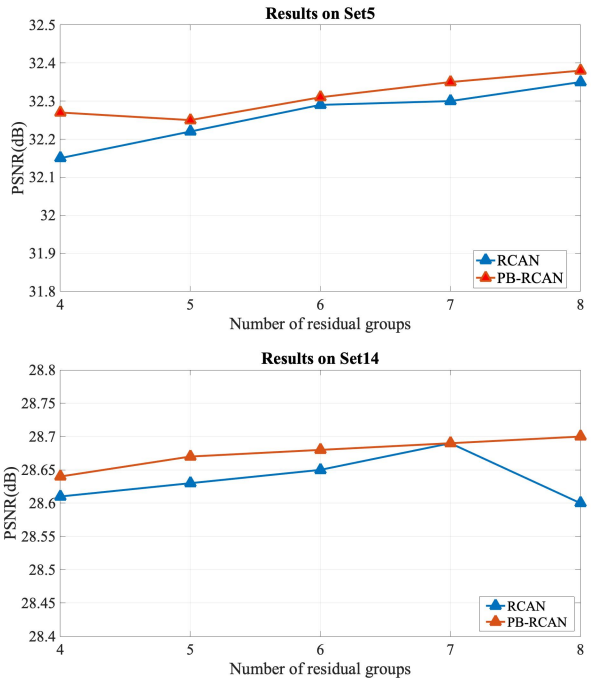

Fig. 2. Results of different RCANs trained by pushing and bounding loss.

A. Data and Training

Following the setting of [17], [15], 800 training images from DIV2K dataset [19] are used as training set, and Set5 [20], Set14 [21], BSDS100 [22] and Urban100 [23] are used as validation sets. LR images are generated by bicubic downscaling and data augmentation is done by randomly rotating input images by $90°$ or $180°$, and vertically or horizontally flip. We train our models with RGB images and our results are evaluated on Y channel of transformed YCbCr space.

The size of our input is 48×48 and the batch size is 32. We train each model for 1000 epochs. We choose Adam [24] as our optimizer, with $\beta_1 = 0.9$, $\beta_2 = 0.99$, and $\epsilon = 10^{-8}$. Our initial learning rate is $2e-4$ and we decay it by half every 200 epochs.

TABLE I
Results RCAN trained by different losses. Scale factor is $\times 4$. Best average PSNR/SSIM is chosen in 8×10^5 iterations. Best result is highlighted.

Datasets	Set5	Set14
RCAN baseline	32.22/0.8960	28.63/0.7821
IGT loss	32.19/0.8947	28.58/0.7817
IBGT loss	32.16/0.8944	28.61/0.7822
Pushing and bounding loss	**32.25/0.8960**	**28.67/0.7833**

B. Comparison with Different Losses

To investigate the superiority of the proposed pushing and bounding loss, we train the same network with different kinds of losses, *i.e.*IGT loss, IBGT loss and pushing and bounding loss. We choose RCAN as the baseline and the scale factor is 4. it is important to note that the RCAN here has 5 residual groups and each group contains 10 residual channel attention blocks (RCABs). Besides, the number of feature channels is set to 32. We train the baseline with only final reconstructing loss. For IBGT loss, the Gaussian kernel sizes K_i is set as

$$K_i = \begin{cases} 0 & i = 5 \\ 2 \times (5 - i + 1) - 1 & 1 \leq i \leq 5 \end{cases} \tag{13}$$

For pushing and bounding loss, the maximum margin m is set as 1×10^{-2}. As shown in Table I, except for pushing and bounding loss, RCAN trained by the other two losses will have worse performance. This also proves our previous arguments that IGT loss will confuse the network and IBGT loss actually damages the original optimizing target. IGT loss may be useful to image recognition and pose estimation, because these two tasks are essentially classification tasks. The results of both tasks are not sensitive to the value of features, so even if all internal features are supervised by the same ground truth, it will not confuse the network. But SR is actually a regression task, and the value of each pixel in feature maps will influence the final performance. As a result, it is more reasonable to supervise the network by a set of 'progressively accurate' labels. Although IBGT loss satisfies this condition, it actually leads the network to learn to deblur, instead of enhancing the resolution. Pushing and bounding loss does not explicitly give any intermediate labels, but it still forces the network to learn better features as the network goes deeper. Thus, it works for deeply supervised SR networks.

C. Study on Different Model Depths

To investigate the effectiveness of pushing and bounding loss on different model depths, we conduct experiments on PB-RCANs with different number of residual groups. Original RCANs with only final reconstructing loss are chosen as the baseline. We change the number of residual groups from 4 to 8, and each residual group consists of 10 residual channel

attention blocks (RCABs) . The number of channels is set as 32 and the scale factor is 4. Results on Set5 and Set14 are shown in Figure 2. Interestingly, the results do not always get better as the number of residual groups increases. This may be caused by fluctuation of training process. Considering this fluctuation, extensive experiments are done on PB-RCANs with different depths. As shown in the figure, PB-RCANs can always achieve better results than original ones on different datasets.

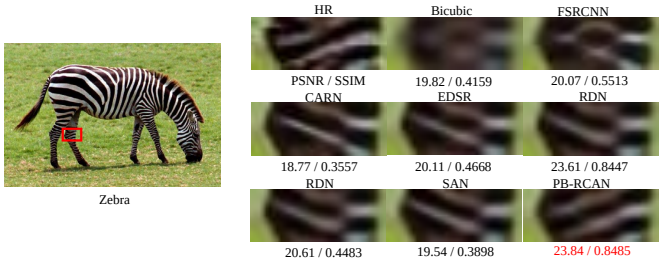

Fig. 3. Visual results of *Zebra* from Set14.

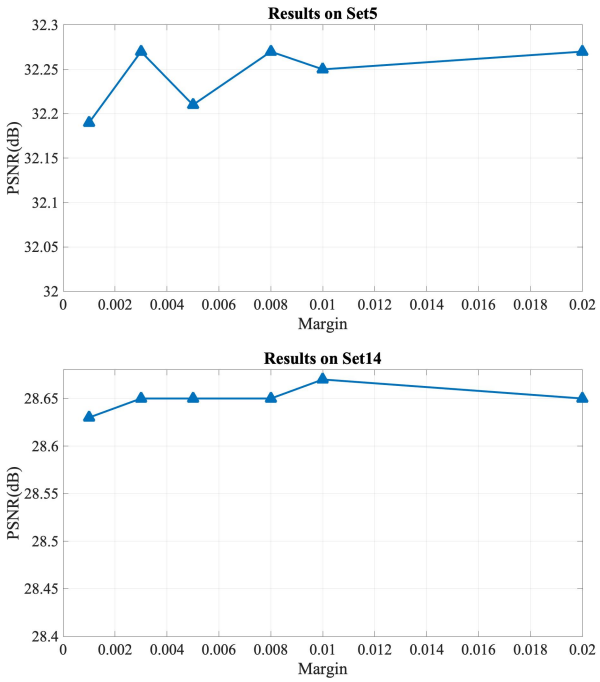

Fig. 4. Results of RCAN trained by pushing and bounding loss with different maximum margins.

D. Study of Maximum Margin

To investigate the influence of maximum margin m between different intermediate results, experiments are conducted with different m. We choose six typical values, *i.e.*1×10^{-3}, 3×10^{-3}, 5×10^{-3}, 8×10^{-3}, 1×10^{-2} and 2×10^{-2}. Experiments are done on PB-RCAN. The scale factor is 4. As shown in Figure 4, as the m increases to 5×10^{-3}, the final results almost keep the same. It can be inferred that as long the maximum is great enough, the final results is not so sensitive to this hyper-parameter.

E. Comparison with State-of-the-art Models

To further prove the effectiveness of our pushing and bounding loss, we train a PB-RCAN with complete RCAN structure in [17]. We compare the results with seven other models, *i.e.*FSRCNN [25], LapSRN [26], CARN [27], D-DBPN [16], EDSR [14], RDN [15], RCAN [17] and SAN [28]. As shown in Table II, RCAN trained by the proposed pushing and bounding loss outperform the original RCAN on all four public validation datasets. It indicates that original RCAN is not fully optimized and its results can be further improved. Also, it outperforms all previous methods. The visual result is shown in Figure 3

V. Analysis

Deep supervision has its benefits in twofold. Firstly, early layers in deeply supervised networks are directly connected to the loss layer, and thus gradients can easily flow to all layers. However, in networks with only final reconstructing loss, gradients received by early layers are not so strong enough to optimize them. Although multiple skip connections can largely resolve this problem, it can be further improved [7], [29]. Secondly, as pointed out in [3], deep supervision can be viewed as a kind of regularization, which reduces the potential optimizing directions of the whole network and make it easier for the model to find a global optimal solution. Compared with other networks, deeply supervised networks are forced to progressively learn better and better results. This is a kind a greedy strategy, and it largely simplifies this problem. Thus, deeply supervised networks can more easily find better solution. There is one concern that a greedy strategy will disable the model to find global optimal solution. Although the solution may be not global optimum, experiments show that deeply supervised networks perform better that original ones.

Despite its solid benefits, deep supervision is rarely applied to SR networks. But in fact, deep supervision is very suitable for SR networks. Firstly, The overall structures of SR networks are usually plain. They have on no pooling layers or extra branches like segmentation networks or detection networks. Thus, it is easy to modify them to deeply supervised versions. Secondly, depth plays important role in SR networks, and deeper SR networks usually have better performances. Thus, optimizing problem is more significant in SR networks.

But in reality, intermediate labels in SR networks are hard to be defined and inappropriate intermediate labels will even harm the performance. The proposed pushing and bounding loss circumvent this problem and only forces the network to learn better features as the network goes deeper. We implement deep supervision to SR networks without explicitly giving any intermediate labels and achieve stable improvements on original models.

TABLE II
RESULTS OF DIFFERENT $\times 4$ MODELS ON FOUR PUBLIC DATASETS. BEST RESULTS ARE HIGHLIGHTED.

Model	Set5		Set14		B100		Urban100	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Bicubic	28.42	0.8104	26.00	0.7027	25.96	0.6675	23.14	0.6577
FSRCNN [25]	30.71	0.8657	27.59	0.7535	26.98	0.7150	24.62	0.7280
LapSRN [26]	31.54	0.8850	28.19	0.7720	27.32	0.7280	25.21	0.7560
CARN [27]	32.13	0.8937	28.60	0.7806	27.58	0.7349	26.07	0.7837
EDSR [14]	32.46	0.8968	28.80	0.7876	27.71	0.7420	26.64	0.8033
D-DBPN [16]	32.47	0.8980	28.82	0.7860	27.72	0.7400	26.38	0.7946
RDN [15]	32.47	0.8990	28.81	0.7871	27.72	0.7419	26.61	0.8028
RCAN [17]	32.63	0.9002	28.87	0.7889	27.77	0.7436	26.82	0.8087
SAN [28]	32.64	0.9003	28.92	0.7888	27.78	0.7436	26.79	0.8068
PB-RCAN	**32.65**	**0.9005**	**28.92**	**0.7890**	**27.82**	**0.7440**	**26.83**	**0.8093**

VI. CONCLUSION

In this paper, we are interested in solving the optimizing problem in deep super resolution (SR) convolutional neural networks (CNNs). Although deep supervision can greatly help train deep models, it is rarely applied in SR networks, because the required intermediate labels are hard to be defined for the SR problem. To tackle this matter, we propose the pushing and bounding loss, which can directly supervise all internal layers without explicitly giving any intermediate labels. We only guide the network to learn better features as the network goes deeper. It also severs as the role of regularization, leading the network to search for the optimal solution and largely accelerate the process of finding better results. As a result, Extensive experiments show that our method achieves stable improvement on original models.

ACKNOWLEDGMENT

This work was supported by the National Key R&D Program of China (2018YFB1403900) and the Science and Technology Program of Beijing (Z201100001820002). It was also the research achievement of the Key Laboratory of Digital Rights Services, which is one of the National Science and Standardization Key Labs for Press and Publication Industry.

REFERENCES

[1] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *European conference on computer vision*. Springer, 2014, pp. 184–199.

[2] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[3] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, "Deeply-supervised nets," in *Artificial intelligence and statistics*, 2015, pp. 562–570.

[4] V. Ramakrishna, D. Munoz, M. Hebert, J. A. Bagnell, and Y. Sheikh, "Pose machines: Articulated pose estimation via inference machines," in *European Conference on Computer Vision*. Springer, 2014, pp. 33–47.

[5] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *European conference on computer vision*. Springer, 2016, pp. 483–499.

[6] Y. Liu and M. S. Lew, "Learning relaxed deep supervision for better edge detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 231–240.

[7] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[8] L. Ke, H. Qi, M.-C. Chang, and S. Lyu, "Multi-scale supervised network for human pose estimation," in *2018 25th IEEE International Conference on Image Processing (ICIP)*. IEEE, 2018, pp. 564–568.

[9] Y. Chen, Z. Wang, Y. Peng, Z. Zhang, G. Yu, and J. Sun, "Cascaded pyramid network for multi-person pose estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7103–7112.

[10] J. Canny, "A computational approach to edge detection," *IEEE Transactions on pattern analysis and machine intelligence*, no. 6, pp. 679–698, 1986.

[11] J. Kim, J. Kwon Lee, and K. Mu Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1646–1654.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[13] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4681–4690.

[14] B. Lim, S. Son, H. Kim, S. Nah, and K. Mu Lee, "Enhanced deep residual networks for single image super-resolution," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2017, pp. 136–144.

[15] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, "Residual dense network for image super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2472–2481.

[16] M. Haris, G. Shakhnarovich, and N. Ukita, "Deep back-projection networks for super-resolution," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1664–1673.

[17] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, "Image super-resolution using very deep residual channel attention networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 286–301.

[18] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1874–1883.

[19] E. Agustsson and R. Timofte, "Ntire 2017 challenge on single image super-resolution: Dataset and study," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 126–135.

[20] X. Chen and C. Qi, "Low-rank neighbor embedding for single image super-resolution," *IEEE Signal Processing Letters*, vol. 21, no. 1, pp. 79–82, 2013.

[21] R. Zeyde, M. Elad, and M. Protter, "On single image scale-up using sparse-representations," in *International conference on curves and surfaces*. Springer, 2010, pp. 711–730.

[22] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 5, pp. 898–916, 2010.

[23] J.-B. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5197–5206.
[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[25] C. Dong, C. C. Loy, and X. Tang, "Accelerating the super-resolution convolutional neural network," in *European conference on computer vision*. Springer, 2016, pp. 391–407.
[26] W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang, "Deep laplacian pyramid networks for fast and accurate super-resolution," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 624–632.
[27] N. Ahn, B. Kang, and K.-A. Sohn, "Fast, accurate, and lightweight super-resolution with cascading residual network," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 252–268.
[28] T. Dai, J. Cai, Y. Zhang, S.-T. Xia, and L. Zhang, "Second-order attention network for single image super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 065–11 074.
[29] Y. Chen, J. Li, H. Xiao, X. Jin, S. Yan, and J. Feng, "Dual path networks," in *Advances in Neural Information Processing Systems*, 2017, pp. 4467–4475.