Meta-USR: A Unified Super-Resolution Network for Multiple Degradation Parameters

Xuecai Hu, Zhang Zhang, *Member, IEEE*, Caifeng Shan, *Senior Member, IEEE*, Zilei Wang, *Member, IEEE*, Liang Wang, and Tieniu Tan, *Fellow, IEEE*

***Abstract*— Recent research on single image super-resolution (SISR) has achieved great success due to the development of deep convolutional neural networks. However, most existing SISR methods merely focus on super-resolution of a single fixed integer scale factor. This simplified assumption does not meet the complex conditions for real-world images which often suffer from various blur kernels or various levels of noise. More importantly, previous methods lack the ability to cope with arbitrary degradation parameters (scale factors, blur kernels, and noise levels) with a single model. A few methods can handle multiple degradation factors, e.g., noninteger scale factors, blurring, and noise, simultaneously within a single SISR model. In this work, we propose a simple yet powerful method termed meta-USR which is the first unified super-resolution network for arbitrary degradation parameters with meta-learning. In Meta-USR, a meta-restoration module (MRM) is proposed to enhance the traditional upscale module with the capability to adaptively predict the weights of the convolution filters for various combinations of degradation parameters. Thus, the MRM can not only upscale the feature maps with arbitrary scale factors but also restore the SR image with different blur kernels and noise levels. Moreover, the light-weight MRM can be placed at the end of the network, which makes it very efficient for iteratively/repeatedly searching the various degradation factors. We evaluate the proposed method through extensive experiments on several widely used benchmark data sets on SISR. The qualitative and quantitative experimental results show the superiority of our Meta-USR.**

***Index Terms*— Arbitrary degradation, blur kernel, deep learning, meta-learning, scale factor, super-resolution.**

I. Introduction

SINGLE image super-resolution (SISR) aims to reconstruct a visually natural high-resolution (HR) image from its degraded low-resolution (LR) image, which is imperative for many devices, such as high-definition (HD) television sets, mobile devices, remote sensing satellites, and medical imaging [computed tomography (CT) and magnetic resonance imaging (MRI)], which is imperative for a wide range of applications in security and surveillance imaging [1], [2], medical imaging [3], satellite and aerial imaging [4], and up-to-date multimedia. It is also a classical and challenging problem [5], [6] in the computer vision community. As there may be multiple HR images corresponding to the same LR image, the SISR is a seriously ill-posed problem.

With the rapid development of deep convolutional neural networks (CNNs), lots of CNN-based SISR methods have been proposed, and state-of-the-art performance has been improved constantly on multiple SISR benchmarks. However, the practicability of these SISR methods is still limited. The main reasons are as follows. First, previous SISR work has a lack of concern for task settings in practice. Existing SISR methods (Enhanced Deep residual network for Super-Resolution (EDSR) [7], residual dense network (RDN) [8], and residual channel attention network (RCAN) [9]) only consider super-resolution of a single fixed integer scale factor ($\times 2$, $\times 3$, and $\times 4$). These methods zoom in the feature maps at the end of networks with the subpixel convolution [10] or transposed convolution [11], termed postupsample methods, as shown in Fig. 1. Since these models can only perform SR with a single fixed integer scale factor, researchers have to design a specific upscale module for each scale factor. However, in real-world scenarios, it is common and necessary to perform SISR with arbitrary scale factors. Second, most existing models [7]–[9] are proposed for image enhancement under a single degradation factor with fixed blurring level or fixed noise level. However, the images captured from real-world scenarios tend to suffer from various degradation factors, various noise levels and various blur kernels, as shown in Fig. 2. Once the real-world images do not follow the degradation assumption in these models, their performances will drop dramatically. To solve this problem, the super-resolution network for multiple degradations with noise-free (SRMDNF) [9] method learns a single model for multiple blur kernels by concatenating the blur maps with the input image. However, when the iterative kernel correction (IKC) [12] method applies the SRMDNF [9] for the blind SR, an LR image has to be repeatedly passed through the whole network SRMDNF by iteratively correcting the blur kernel which makes it very time consuming for practical use. To avoid passing through the entire network repeatedly, a postdeblurring module is

Manuscript received September 10, 2019; revised July 3, 2020; accepted August 8, 2020. Date of publication August 28, 2020; date of current version September 1, 2021. This work was supported in part by the National Key Research and Development Program of China under Grant 2016YFB1001002; in part by the Shandong Provincial Key Research and Development Program (Major Scientific and Technological Innovation Project) under Grant 2019JZZY010119; and in part by the CAS-AIR under Grant 2019-001. *(Corresponding author: Zhang Zhang.)*

Xuecai Hu and Zilei Wang are with the School of Information Science and Technology, University of Science and Technology of China, Hefei 230026, China (e-mail: huxc@mail.ustc.edu.cn, zlwang@ustc.edu.cn).

Zhang Zhang, Liang Wang, and Tieniu Tan are with the Center for Research on Intelligent Perception and Computing, National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, and also with the School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100190, China (e-mail: zzhang@nlpr.ia.ac.cn; wangliang@nlpr.ia.ac.cn; tnt@nlpr.ia.ac.cn).

Caifeng Shan is with the College of Electrical Engineering and Automation, Shandong University of Science and Technology, Qingdao 266590, China, and also with the Artificial Intelligence Research, Chinese Academy of Sciences, Beijing 100190, China (e-mail: shancf@cas-air.cn).

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2020.3016974

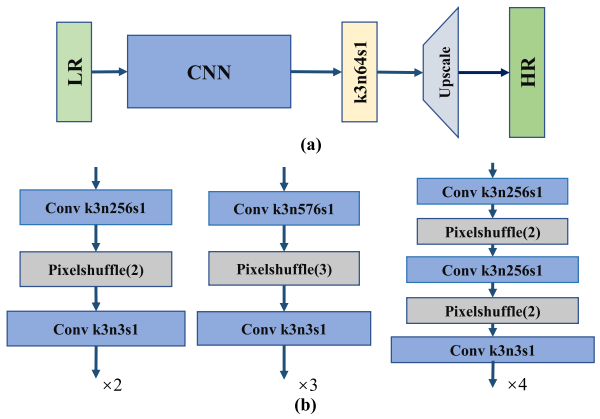

Fig. 1. Framework of a typical SISR method. (a) Typical SISR methods zoom in the feature maps at the end of the network by the upscale module to reduce inference time. (b) Instance of the upscale module based on the subpixel convolution in the RDN [8]. "k" denotes the kernel size of the convolution kernel, "n" denotes the number of the output feature maps, and "s" denotes the stride of the filter. Each scale factor has a specific upscale module. These methods cannot handle noninteger scale factors. (a) Typical super-resolution network. (b) Upscale Modules for $\times 2$, $\times 3$, and $\times 4$.

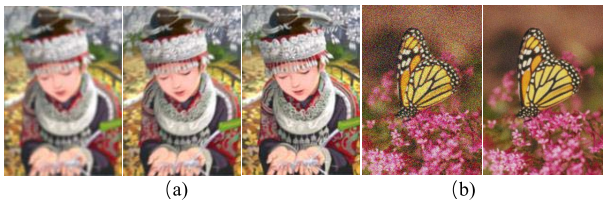

Fig. 2. Example images suffering from various blur kernels and various levels of noises. (a) Various blur kernels. (b) Various noise levels.

necessary, and this module should be lightweight for iterative computation. Finally, to restore real-world low-quality images caused by various degradation factors, it is an intuitive way to train and store multiple models for different degradation factors, noise levels, and blur degrees. However, it is inefficient and overcomplicated due to the combinatorial explosion problem of multiple degradation factors. Thus, it is an essential problem to learn a single unified model for implementing super-resolution for handling multiple degradation factors.

In this study, we unify three degradation factors, i.e., scale factors, blur kernels and noise levels, into one SISR model within a meta-learning framework. Concretely, we propose a meta unified super-resolution (called Meta-USR) method to efficiently handle customized degradation parameters where we take advantage of the meta-learning to dynamically predict the weights for varying permutation of degradation parameters. Thus, we can train a unified model for handling multiple degradation factors.

There are two modules in our Meta-USR; that is, the feature learning module (FLM) and the Meta-Restoration Module (MRM). The MRM is proposed to replace the typical upscale module, which is at the end of the network after FLM. Note that our MRM can not only upscale the feature maps with arbitrary scale factors but also enhance the SR image by deblurring and denoising. Since the MRM is placed behind the network of feature learning, it is a postrestoration module which is suitable for iterative computation.

In the MRM, the key part is a mapping function termed location projection (LP) which is responsible for finding a unique pixel on the LR image for each pixel on the SR image. Besides the LP, two WeightNets are proposed to predict weights for different combinations of the degradation parameters. Finally, the proposed MRM uses the predicted weights of the filters and the features extracted by the FLM to generate an HR image. To the best of our knowledge, this is the first attempt to unify the optimization of all the degradation parameters for SISR via training a single CNN model.

This work is an extension of an earlier and preliminary version presented in [13]. Compared with our previous work, the differences and contributions of this article are summarized as follows.

1) We propose a method Meta-USR to learn a single SISR model for arbitrary degradation parameters (scale factor, blurring, and noise). It is the first model to take all the degradation parameters into consideration in order to enhance the flexibility of the SISR model to complex degradations of images in natural scenarios. Our conference article only focuses on the problem of noninteger scale factor and proposes the meta-upscale layer to solve it.
2) To extend the capability of meta-upscale layer, an MRM is developed, which takes multiple degradation factors, e.g., the scale factor, blur kernel, and noise level, as input to predict the weights of the filters. Our MRM can not only upscale the feature maps with arbitrary scale factors but also restore the SR image with different blur kernels and different noise levels. The proposed MRM is lightweight and flexible to construct various deep image enhancement models. In this work, we share the FLM for all the degradation parameters and place the MRM at the end of the network for high efficiency in inference.
3) We conduct expensive experiments on six popular benchmark data sets on SISR to show the superiority of the meta-upscale layer. Our method can work on arbitrary scale factors, even though some scale factors are unseen in the training stage.

II. Related Work

A. Single Image Super-Resolution

1) Preupsampling Super-Resolution: With the rapid development of deep learning, numerous CNN-based methods have been proposed. A three-layers CNN was first proposed by Dong *et al.* [14] called SRCNN, which upscales the LR image with bicubic interpolation before feeding into the network. Kim *et al.* [15] increased the depth of the network and used the residual learning for stable training. Kim *et al.* [16] first introduced recursive learning to SISR, called DRCN. However, the size of input for these networks [14], [15], [17], [18] needs to be the same size as the final HR image, and these methods are time consuming.

2) Postupsampling Super-Resolution: Shi *et al.* [10] first proposed a real-time super-resolution algorithm ESPCNN by proposing the subpixel convolution layer. The ESPCNN upscales the image at the end of the network to reduce the

computational cost. Ledig *et al.* [19] introduced the residual block and the adversarial learning [20] to make the generated images more realistic and natural. Lim *et al.* [7] used the deeper and wider residual networks called EDSR, which removes the BN layer and uses the residual scaling to speed up training. Zhang *et al.* [8] proposed a RDN which combines the advantages of the residual block and dense connected block. Then, Zhang *et al.* [21] introduced the residual channel attention to the SR framework. Wang *et al.* [22] proposed a novel deep spatial feature transform to recover textures conditioned on the categorical priors. DBPN [23] made use of the mutual dependencies between LT and HR images. DBPN exploited iterative upsampling and downsampling layers to provide an error feedback mechanism for each stage. Although numerous CNN-based methods (RCAN [21], CAscading Residual Network (CARN) [24], SRFeat [25], multi-scale residual network (MSRN) [26], coarse-to-fine learning for super-resolution (CFLSR) [27]–[36]) have been proposed, all these methods only consider several integer scale factors ($\times 2$, $\times 3$, and $\times 4$) and have to design a specific upscale module for each scale factor. A few methods take noninteger scale factors into consideration, and no methods attempt to learn a single SR model for arbitrary scale factors without dynamically zooming in the input image.

3) Super-Resolution for Multiple Degradations: The real-world LR images suffer from various blur kernels and various levels of noise. However, typical SISR methods only consider the bicubic degradation. In order to make SISR methods more realistic, several methods took multiple blur kernels and multiple levels of noises into consideration. The plug-and-play framework (CNN for image restoration (IRCNN) [35]) was proposed which used the iterative method half-quadratic splitting [36]. However, the IRCNN for SISR iteratively passed through the whole network for 30 times which is very time consuming. The deep plug-and-play super-resolution (DPSR) [37] was also a plug-and-play framework, while it only works for the degradation model which downsamples the images at first and then blurs the LR images. Furthermore, it is also time consuming. Super-resolution network for multiple degradations (SRMD) [9] learned a single convolutional super-resolution network for multiple degradations (blurring and noise) by using the dimensionality stretching strategy where the SRMD [9] concatenates the degradation maps with the LR image as a new input to pass through the whole network. In [38], the zero-shot SR (ZSSR) trained image-specific DNN on the testing LR image. But it requires thousands of gradient updates in the inference stage. Our Meta-USR takes the degradation factors to predict the weights of the model instead of concatenating the degradation maps with the input LR image.

Moreover, a few works consider learning a single model for super-resolution of multiple degradation parameters (multiple scale factors, multiple blur kernels, and multiple levels of noises). Our Meta-USR is the first attempt to take all the degradation parameters into consideration for a flexible solution of SISR. If the blur kernel was changed, the DPSR [37], SRMD [9], and ZSSR [38] have to go through the entire network again and again. Different from them, our MRM is placed at the end of the SISR network, and it is lightweight. Since we only need to repeatedly execute MRM instead of the whole SISR network, it is more efficient if we use our Meta-USR to solve the blind super-resolution by iteratively updating the blur kernel.

B. Meta Learning

Meta-learning, or learning to learn, studies how learning systems can increase efficiency through experience. The meta-learning is mainly used in a few-shot/zero-shot learning [39]–[41] and transfer learning [42]. The detailed survey of meta-learning can be found in [45] and [46]. The weight prediction is one of the meta-learning strategies in the neural network [44]. The weights of the neural network models are predicted by another neural network rather than directly learned from training data sets. In recent years, the weight prediction has been applied to various tasks, such as detection [45], instance segmentation [46], video super-resolution [47], and image filtering [48].

Unlike the previous work [48], we take advantage of the meta-learning to predict weights of the restoration filters for different degradation parameters. And we mainly focus on the reformulation of the upscale module and propose the MRM to replace it. Our MRM takes both the coordinate and degradation information as input for the weight prediction network. Our Meta-USR can train a unified model for super-resolution with arbitrary scale/blur kernels/noise levels which can be applied more conveniently and efficiently for practical applications.

III. Our Approach

In this section, we elaborate on the proposed Meta-USR. Before we describe the proposed approach, it is essential to first introduce the details of the degradation model. After that, we describe our model architecture, as shown in Fig. 3. In our Meta-USR, the FLM extracts the feature of the LR image. The MRM zooms in the feature maps with arbitrary scale factors and enhances the SR image by deblurring and denoising.

A. Problem Formulation

1) Degradation Model: SISR aims to reconstruct a visually natural HR image from its degraded LR image. However, it is challenging to collect LR–HR image pairs from the real world. Typically, previous methods are trained on synthetic data sets. Most existing methods degrade the visually natural HR image and generate the LR image with the following simple degradation function. Formally

$$\mathbf{I}_{\mathrm{LR}} = (\mathbf{I}_{\mathrm{HR}}) \downarrow_s \tag{1}$$

where $\mathbf{I}_{\mathrm{LR}}$ is the LR image. $\mathbf{I}_{\mathrm{HR}}$ is the HR image. $\downarrow_s$ represents the bicubic interpolation downsampling with scale factor $\mathbf{s}$.

However, real-world LR images suffer from various factors, e.g., blurring, sensor noise, and read–write noise. In order to cope with real-world images, a more flexible model is needed to take various degradation factors into considerations. We choose the model [9] to describe image degradation as the combinations of scale factor, blurring and noise as follows:

$$\mathbf{I}_{\mathrm{LR}} = D(\mathbf{I}_{\mathrm{HR}}; \delta) = (\mathbf{I}_{\mathrm{HR}} \otimes \mathbf{k}) \downarrow_s + \mathbf{n} \tag{2}$$

where D denotes the degradation function. δ denotes the set of the degradation parameters. $\mathbf{k}$ represents a blur kernel, $\mathbf{n}$

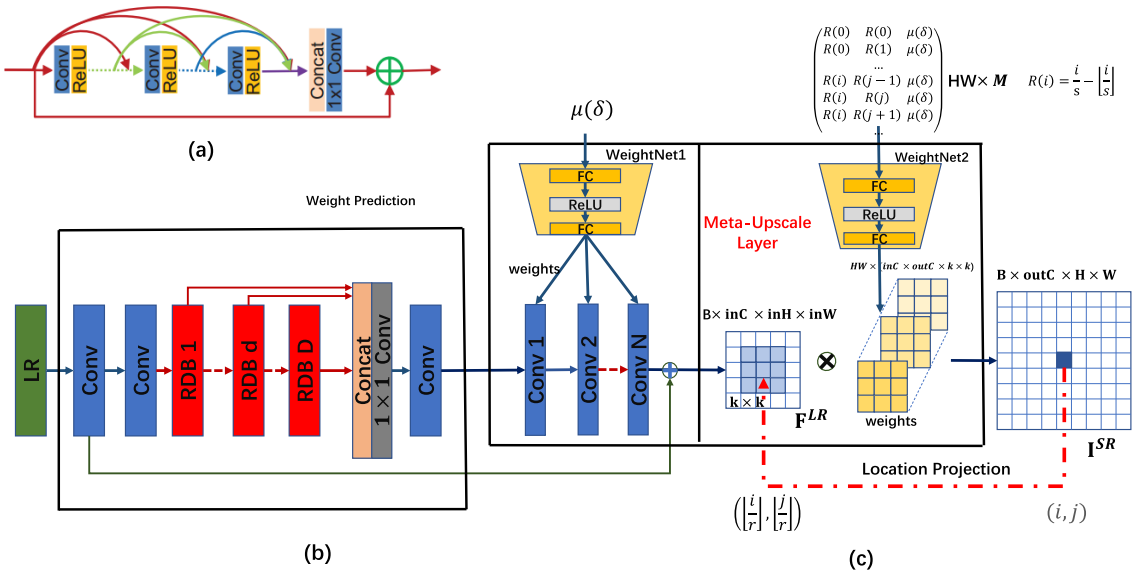

Fig. 3. Instance of our Meta-USR based on RDN [8]. (a) RDB proposed by RDN [8]. (b) FLM which generates the shared feature maps for arbitrary scale factors. (c) For each pixel on the SR image, we project it onto the LR image. The proposed MRM takes a sequence of coordinate-related and scale-related vectors as input to predict the weights for convolution filters. With the convolution operation, the MRM finally generates the HR image. (a) RDB. (b) FLM. (c) MRM.

denotes noise, and $\downarrow_s$ represents the downsampling operation with scale factor $\mathbf{s}$.

2) Meta-USR Formulation: Although we could train and store a specific model for each fixed scale factor $\mathbf{s}$, fixed blur kernel $\mathbf{k}$, and fixed noise $\mathbf{n}$ based on previous methods [8], [24], these SR models perform poorly when the degradation parameters of the testing image deviate from that in the training stage.

Our Meta-USR is proposed to learn a single SR model for arbitrary degradation parameters (scale factor, blurring, and noise). Suppose $\mathbf{I}^{\mathrm{LR}}$ denotes an LR image which is degraded from the corresponding original HR image $\mathbf{I}^{\mathrm{HR}}$, our Meta-USR reconstructs an HR image $\mathbf{I}^{\mathrm{SR}}$ from $\mathbf{I}^{\mathrm{LR}}$. We formulate our Meta-USR as

$$\mathbf{I}^{\mathrm{SR}} = G\left(\mathbf{I}^{\mathrm{LR}}, \delta\right) \tag{3}$$

where both the $\mathbf{I}^{\mathrm{LR}}$ and the degradation parameters δ are the input of the proposed Meta-USR network G. The architecture of the Meta-USR network is shown in Fig. 3, where the FLM and MRM are two key parts. The details of the two parts will be presented in Sections III-B and C.

B. Feature Learning Module

Most state-of-the-art SISR methods [7]–[10], [19] are pos-tupsampling methods which zoom in the feature maps at the end of networks with the upscale module. In our Meta-USR, the FLM is responsible for extracting feature of the LR image. Thus, these state-of-the-art methods could be selected as our FLM by simply removing the traditional upscale module.

We choose the state-of-the-art SISR network, namely, RDN [8] as our FLM. Note that our Meta-USR can also work with EDSR [7], MDSR [7], or RCAN [21]. For the RDN [8], there are 3 convolutional layers and 16 residual dense blocks (RDBs). Each RDB has eight convolutional layers. The growth rate for the dense block is 64. And the channel number of extracted feature maps is 64. The detailed structure is shown in Fig. 3(b). More details can be found in RDN [8].

C. Meta-Restoration Formulation

The typical upscale module, based on subpixel convolution [10] or transposed convolution [11], can only zoom in the feature maps with a fixed integer scale factor. Thus, we proposed the MRM to replace the typical upscale module. Our MRM can not only upscale the LR feature maps with arbitrary scale factors but also restore the LR image with different blur kernels and noise levels. In MRM, these are two types of convolution layers. The first type of convolution layers is a regular convolution layer with stride $= 1$, which does not change the size of input feature maps. We directly learn the weights of these convolution layers by the WeightNet1, as shown in Fig. 3. The second type of convolution layers is responsible for zooming in the feature maps with arbitrary scale factors, and the weights of this convolution layer are learned by the WeightNet2 in Fig. 3. Here, we focus on how to formulate the second type of convolution layers (**Meta-Upscale Layer**).

1) Meta-Upscale Layer: Let $\mathbf{F}^{\mathrm{LR}}$ denote the extracted feature. Suppose the degradation parameters is $\delta = (s, \mathbf{k}, \mathbf{n})$, where $(s, \mathbf{k},$ and $\mathbf{n})$ denote the scale factor, blur kernel, and noise level, respectively.

For each pixel (i, j) on the SR image, its value can be determined by the feature of the pixel (i', j') on the LR image and the weights of the corresponding filter. The meta-upscale layer can be seen as a mapping function between $\mathbf{I}^{\mathrm{SR}}$ and $\mathbf{F}^{\mathrm{LR}}$. At first, the meta-upscale layer localizes the coordinate (i', j') on the LR image for each pixel (i, j) on the SR image with the given scale factor. Then, the value of pixel (i, j) on the SR

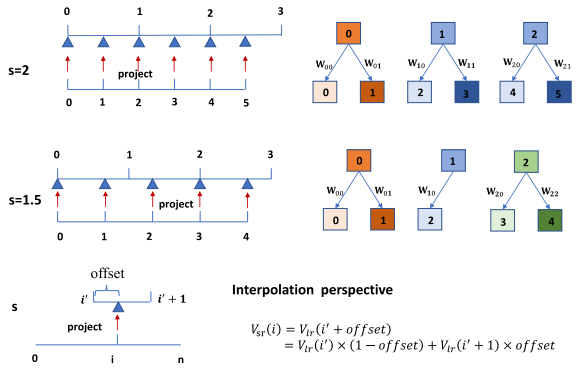

Fig. 4. Schematic on how to upscale the feature map with the noninteger scale factor $r = 1.5$. Here, we only show the 1-D case for simplicity.

image will be calculated according to a specific filter and the feature of pixel (i', j') on the LR image. Thus, we formulate the meta-upscale layer as

$$\mathbf{I}^{\mathrm{SR}}(i, j) = \Phi\left(\mathbf{F}^{\mathrm{LR}}\left(i', j'\right), \mathbf{W}(i, j, \delta)\right) \tag{4}$$

where $\mathbf{I}^{\mathrm{SR}}(i, j)$ denotes the pixel value at (i, j) on SR image. $\mathbf{F}^{\mathrm{LR}}(i', j')$ denotes the feature of pixel (i', j') on the LR image. $\mathbf{W}(i, j, \delta)$ is the weights of filter for pixel (i, j) when the degradation parameters are $\delta = (s, \mathbf{k}, \mathbf{n})$. $\Phi(.)$ is the feature mapping function to calculate the pixel value.

For the meta-upscale layer, there are three important functions, i.e., the LP, the weight prediction and the feature mapping. The LP maps pixel (i, j) on the SR image to pixel (i', j') on the LR image by projection. And the Weight Prediction Network predicts the weights of the filter for each pixel on the SR image. At last, the feature mapping function maps the feature on the LR image with the predicted weights of the filter back to the SR image to calculate the value of the pixel.

2) Location Projection: For each pixel (i, j) on the SR image, the LP finds the (i', j') on the LR image. The value of the pixel (i, j) is decided by the feature of (i', j') on the LR image. We do the following projection operator to map these two pixels:

$$\left(i', j'\right) = T(i, j) = \left(\left\lfloor \frac{i}{s} \right\rfloor, \left\lfloor \frac{j}{s} \right\rfloor\right) \tag{5}$$

where T is the transformation function. $\lfloor x \rfloor$ is the floor function that gives the largest integer less than or equal to x.

The proposed LP mapping function can be seen as a kind of variable fractional stride [49] mechanism which could upscale the feature maps with arbitrary scale factors. The fractional stride varies with coordinate and scale factor. As shown in Fig. 4, if the scale factor s is 2, each pixel (i', j') on the LR image generates two pixels. However, if the scale factor is noninteger, such as $s = 1.5$, some pixels generate two pixels (stride $= 1/2$) and some pixels determine one pixel (stride $=$ 1). For each pixel (i, j) on the SR image, our LP function can find a unique pixel (i', j') on the LR image that is most related to pixel (i, j).

3) Weight Prediction: In order to learn a single SR model for multiple degradation parameters, our meta-upscale layer uses a subnetwork to adaptively predict the weights of the filters for different degradation parameters. As shown in Fig. 4, one pixel on the LR image may be involved in the calculations of multiple pixels on the SR image. Thus, for each pixel (i, j) on the SR image, the input of the weight prediction network WeightNet2 should be related with the coordinate (i, j) and degradation parameters δ. We formulate the weight prediction network as follows:

$$\mathbf{W}(i, j, \delta) = \varphi\left(\mathbf{v}_{ij\delta}; \theta\right) \tag{6}$$

where $\mathbf{W}(i, j, \delta)$ denotes the weights of filter for pixel (i, j) on the SR image when the customized degradation parameters are δ. $\mathbf{v}_{ij\delta}$ is a vector related with i, j, δ. $\varphi(.)$ is the weight prediction network and takes the $\mathbf{v}_{ij\delta}$ as input. θ is the parameters of the weight prediction network.

The input $\mathbf{v}_{ij\delta}$ is related with coordinate and degradation parameters. But different factors have different numerical range. These factors should be resized to the same numerical range. Next, we will present how to encode the degradation factors and coordinate to get the input vector $\mathbf{v}_{ij\delta}$.

For the coordinate information, we choose the offset $((i/s) - \lfloor (i/s) \rfloor, (j/s) - \lfloor (j/s) \rfloor)$ as the encoding function which is inspired by typical linear interpolation. For the typical linear interpolation method, given two points (x_0, y_0) and (x_1, y_1), the linear interpolation is the straight line between these points. For given x in the interval (x_0, x_1), the value of x along the straight line is given by

$$y = y_0\left(1 - \frac{x - x_0}{x_1 - x_0}\right) + y_1\frac{x - x_0}{x_1 - x_0} \tag{7}$$

where $(x - x_0)/(x_1 - x_0)$ is the offset. Here, the offset determines the weight of the y_0 and the y_1, as shown in Fig. 4. Thus, we also choose the offset to encode the coordinate information. We formulate the input vector $\mathbf{v}_{ij\delta}$ as follows:

$$\mathbf{v}_{ij\delta} = \left(\frac{i}{s} - \left\lfloor \frac{i}{s} \right\rfloor, \frac{j}{s} - \left\lfloor \frac{j}{s} \right\rfloor, \mu(\delta)\right) \tag{8}$$

where the $\mu(\delta)$ denotes the encoding function for degradation parameters δ. We use the degradation and offset coordinate information to predict the weights. And then we use the weights of filter to restore the LR image by deblurring, denoising, and upscaling.

For scale factor s, the encoding function can be formulated as follows:

$$e_s = \frac{1}{s}. \tag{9}$$

For the blur kernel $\mathbf{k}$, these are two choices, that is, the blur kernel $\mathbf{k}$ by reshaping it to a vector or the kernel width k_w. To encode the blur kernel $\mathbf{k}$, the elements in the $\mathbf{k}$ are reshaped into a vector. Using the blur kernel $\mathbf{k}$ is suitable for most types of blur, while the kernel width k_w only work for Gaussian blur. In this article we choose the blur kernel as the blur information. Thus, the encoding function for the blur kernel can be formulated as follows:

$$\mathbf{e}_{\mathbf{k}} = \mathrm{vec}(\mathbf{k}) \tag{10}$$

vec(.) denotes to vectorize the matrix $\mathbf{k}$.

For the noise **n**, we follow the SRMD [9] which considers the additive white Gaussian with standard deviation n_σ. The standard deviation n_σ denotes the noise level. However, the noise level ranges from 0 to dozens (such as 40). It is better to normalize it. The encoding function for the noise can be formulated as follows:

$$e_{\mathbf{n}} = \frac{n_\sigma}{\lambda} \tag{11}$$

where the λ is used to resize the noise level.

To take these degradation parameters into consideration, our $\mu(\delta)$ concatenates the encoding of these three degradation factors into a vector and can be formulated as

$$\mu(\delta) = (e_s, e_{\mathbf{n}}, e_{\mathbf{k}}). \tag{12}$$

Note that we also choose $\mu(\delta)$ as the input for the WeightNet1, as shown in the Fig. 3.

4) Feature Mapping: We extract the feature of the pixel (i', j') on the LR image from $\mathbf{F}^{\mathrm{LR}}$. We then predict the weights of the filters with weight prediction network in (6). Finally, we map the feature to the value of the pixel on the SR image. We choose the convolution operation as the feature mapping function. We formulate the $\Phi(.)$ as follows:

$$\Phi\left(\mathbf{F}^{\mathrm{LR}}(i', j'), \mathbf{W}(i, j, \delta)\right) = \mathbf{F}^{\mathrm{LR}}(i', j') \otimes \mathbf{W}(i, j, \delta). \tag{13}$$

Note that, we actually extract the feature on a $k \times k$ window centered at (i', j'). And the $k \times k$ represents the kernel size of the convolution filter.

D. Settings in Meta-Restoration Module

In MRM, there are two weight prediction networks, **N** regular convolution layers followed by ReLU and one meta-upscale layer. The WeightNet1 and WeightNet2 have the same structure, as shown in Fig. 3. The weight prediction network outputs a group of weights with the shape $(\mathrm{inC}, \mathrm{outC}, k, k)$. Here, the inC is the number of channels of the feature maps, and we set $\mathrm{inC} = 64$ in this article for RDN [8]. In the meta-upscale layer, the outC is the number of channels for the predicted HR image. That is, outC is set as 3 for color images, outC is set as 1 for grayscale image. For regular convolution layers, the outC equals to inC. The k represents the size of the convolution kernel and is chosen as 3×3 empirically.

Since the output size of the weight prediction network is $k^2 \times \mathrm{inC} \times \mathrm{outC}$ and is very large compared with the input size (vary from 3 to 228), there should be sufficient hidden neurons to encode the variations in the weight prediction. Here, we set the number of hidden neurons as 128 for WeightNet2 and 64 for WeightNet1. Our experiments show that more hidden neurons have no improvements. We use the ReLU as the activation function. We conduct experiments and find that the best number of the fully connected layer is 2 with the balance of speed and performance.

IV. Experimental Results

A. Data Sets and Metrics

1) Data Sets: In the NTIRE 2017 Challenge on SISR, a high-quality data set DIV2K [51] is released. There are 1000 images in the DIV2K database, 800 images for training, 100 images for validation, and 100 images for the test. All of our models are trained with the DIV2K training set. The LR images are degraded from the DIV2K HR images by following the degradation function (2). In test stage, besides the DIV2K, we also perform cross data sets validations on five other standard benchmark data sets, including Set5 [52], Set14 [53], B100 [50], Manga109 [54], and Urban100 [55]. Note that the ground truth of the DIV2K test set is not publicly available. Therefore, we report the results on the DIV2K validation set.

2) Metrics: The super-resolution results are evaluated with peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM) [56]. Following the setting in [8], we only consider the PSNR and SSIM [56] on the Y channel of the transformed YCbCr color space.

B. Training Details

In the SISR, the traditional loss function is L2 loss. Following the settings of [7], we train our network using L1 loss instead of the L2 for better convergence.

During the training of the network, we randomly extract 32 LR RGB patches with a size of 50×50 as a batch input for all experiments. As presented in [8], we randomly augment the patches by flipping horizontally or vertically and rotating 90°. The optimizer is Adam. The learning rate is initialized as 10^{-4} for all layers and decreased by half for every 100 epochs. All experiments run in parallel on four GPUs. Each patch image in a batch has the same scale factor. Our Meta-USR is trained from scratch.

C. Experimental Settings

In order to prove the superiority of the proposed MRM, we conduct three groups of experiments, as shown in Table I. First, we learn a single SR model for arbitrary scale factors, called Meta-SR. To evaluate the effectiveness of the meta-upscale layer in solving SR for arbitrary scale factors, we directly downsample the HR images without blurring and noise. That is, we only consider the scale factor. The range of scale factor is set as (1, 4]. We uniformly sample the scale factor s with stride 0.1. Second, we learn a single SR model for multiple blur kernels, called Meta-DeBlur. To evaluate the effectiveness of the MRM in solving SR for multiple blur kernels, we only vary the blur kernels and fix the scale factor as 2, or 3, or 4 and noise level as 0. We only vary the blur kernels. Finally, we take multiple scale factors, multiple blur kernels and multiple noise levels together into consideration to learn a unified SR model for multiple degradation parameters, called Meta-USR.

D. Meta-SR: Single Model for Arbitrary Scale Factors

1) Settings: We directly downsample the HR images without blurring and noise. The range of scale factor is set as (1, 4]. In this section, we train all these models on arbitrary scale factors together. Since a few previous approaches focus on the super-resolution of arbitrary scale factors with a single model, we need to design six baselines first, as shown in Table II.

TABLE I

SETTINGS OF THREE EXPERIMENTS. "[n m] / k" REPRESENTS THE DEGRADATION FACTOR RANGES FROM n TO m WITH STRIDE = k. WE COMPARE OUR META-SR WITH OTHER BASELINES TO PROVE THE SUPERIORITY OF THE PROPOSED META-UPSCALE LAYER. WE COMPARE OUR META-DEBLUR WITH OTHER BASELINES TO PROVE THE SUPERIORITY OF THE PROPOSED MRM. AND META-USR IS THE FIRST UNIFIED SR MODEL FOR MULTIPLE DEGRADATION FACTORS

Methods \ δ	scale factors	blur kernels	noise
Meta-SR	multiple scale factors, (1, 4] / 0.1	✗	✗
Meta-DeBlur	single fixed scale factor, 2 or 3 or 4	multiple blur kernels, (0.2, 3] / 0.1	✗
Meta-USR	multiple scale factors, [2, 3] / 0.1)	multiple blur kernels, (0.2, 3] / 0.1	multiple levels of noise [0, 40] / 1

TABLE II

DETAILS OF BASELINES WE DESIGN TO SOLVE SUPER-RESOLUTION OF ARBITRARY SCALE FACTORS **s**."(BICUBIC, s)" DENOTES THAT WE UPSCALE INPUT IMAGES (FEATURE MAPS) OR DOWNSCALE OUTPUT IMAGES s TIMES BASED ON BICUBIC INTERPOLATION. FOR UPSCALE MODULE, "(SUBPIXEL CONV, k)" DENOTES THAT THE UPSCALE MODULE UPSCALES THE INPUT FEATURE MAPS k TIMES BASED ON SUBPIXEL CONVOLUTION. "LP CONV" DENOTES THAT OUR META-UPSCALE LAYER UPSCALES THE INPUT FEATURE MAPS BASED ON LP MAPPING

Baselines \ Module	Upscale Input Image	FLM	Upscale Feature Maps	Weight Prediction	Upscale Module	Downscale Output Image
bicubic	(bicubic, **s**)	✗	✗	✗	✗	✗
RDN(x1)	(bicubic, **s**)	RDN [8]	✗	✗	(sub-pixel conv, **1**)	✗
RDN(x2)	(bicubic, **s/2**) when s>2	RDN [8]	✗	✗	(sub-pixel conv, **2**)	(bicubic, **2/s**) when s<2
RDN(x4)	✗	RDN [8]	✗	✗	(sub-pixel conv, **4**)	(bicubic, **4/s**)
BicuConv	✗	RDN [8]	(bicubic, **s**)	✗	(sub-pixel conv, **1**)	✗
Meta-Bicu	✗	RDN [8]	(bicubic, **s**)	✓	(sub-pixel conv, **1**)	✗
Meta-RDN	✗	RDN [8]	✗	✓	(LP conv, **s**)	✗

TABLE III

PSNR RESULTS OF ARBITRARY UPSCALE ON DIFFERENT METHODS. THE EDSR IS BASED ON RESIDUAL BLOCK. AND THE RDN IS BASED ON THE DENSE CONNECTION BLOCK. THE TEST DATA SET IS B100 [50]. THE BEST RESULTS ARE IN **BLACK**

Methods \ Scale	x1.1	x1.2	x1.3	x1.4	x1.5	x1.6	x1.7	x1.8	x1.9	x2.0	x2.1	x2.2	x2.3	x2.4	x2.5
bicubic	36.56	35.01	33.84	32.93	32.14	31.49	30.90	30.38	29.97	29.55	29.18	28.87	28.57	28.31	28.13
RDN(x1)	42.41	39.76	38.00	36.68	35.57	34.64	33.87	33.19	32.60	32.08	31.63	31.23	30.86	30.51	30.23
RDN(x2)	41.84	39.34	37.87	36.63	35.56	34.63	33.83	33.1	32.52	32.11	31.61	31.24	30.82	30.44	30.23
RDN(x4)	39.71	38.48	37.33	36.29	35.34	34.52	33.81	33.14	32.60	32.09	31.61	31.23	30.88	30.52	30.31
BicuConv	41.86	39.16	37.88	29.86	35.68	34.77	33.95	33.18	32.60	31.85	31.53	31.11	30.87	30.38	30.16
Meta-Bicu	42.11	39.58	38.07	36.83	35.81	34.86	34.03	33.24	32.63	32.18	31.59	31.21	30.91	30.54	30.34
Meta-RDN(**our**)	**42.94**	**40.14**	**38.33**	**36.96**	**35.88**	**34.94**	**34.14**	**33.48**	**32.87**	**32.36**	**31.88**	**31.48**	**31.09**	**30.78**	**30.49**
EDSR(x1)	42.42	39.79	38.08	36.73	35.65	34.73	33.83	33.27	32.67	32.15	31.69	31.29	**30.91**	30.56	30.28
EDSR(x2)	41.79	39.11	37.79	36.51	35.40	34.49	33.81	33.11	32.57	32.09	31.57	31.15	30.81	30.47	30.22
EDSR(x4)	39.61	38.41	37.27	36.24	35.30	34.46	33.75	33.09	32.56	32.04	31.56	31.17	30.82	30.46	30.24
Meta-EDSR(**our**)	**42.72**	**39.92**	**38.16**	**36.84**	**35.78**	**34.83**	**34.06**	**33.36**	**32.78**	**32.26**	**31.73**	**31.31**	30.87	**30.60**	**30.40**

Methods \ Scale	x2.6	x2.7	x2.8	x2.9	x3.0	x3.1	x3.2	x3.3	x3.4	x3.5	x3.6	x3.7	x3.8	x3.9	x4.0
bicubic	27.89	27.66	27.51	27.31	27.19	26.98	26.89	26.59	26.60	26.42	26.35	26.15	26.07	26.01	25.96
RDN(x1)	29.95	29.68	29.45	29.21	29.03	28.81	28.67	28.47	28.30	28.15	28.00	27.86	27.72	27.59	27.47
RDN(x2)	29.71	29.65	29.43	29.20	29.05	28.71	28.69	28.51	28.49	28.18	28.17	28.09	27.84	27.61	27.51
RDN(x4)	29.99	29.75	29.53	29.26	29.14	28.89	28.75	28.57	28.42	28.19	28.16	27.93	27.81	27.70	27.64
BicuConv	29.81	29.55	29.28	29.05	28.91	28.64	28.51	28.28	28.13	27.91	27.84	27.61	27.49	27.37	27.29
Meta-Bicu	30.01	29.76	29.54	29.28	29.22	28.89	28.75	28.54	28.39	28.17	28.11	27.87	27.75	27.64	27.58
Meta-RDN(**our**)	**30.21**	**29.95**	**29.71**	**29.48**	**29.30**	**29.08**	**28.91**	**28.73**	**28.56**	**28.41**	**28.28**	**28.14**	**28.02**	**27.88**	**27.75**
EDSR(x1)	29.98	29.73	29.49	29.25	29.07	28.85	28.69	28.51	28.36	28.18	28.04	27.91	27.77	27.64	27.52
EDSR(x2)	29.91	29.66	29.45	29.19	29.09	28.78	28.64	28.45	28.34	28.11	28.06	27.83	27.73	27.61	27.49
EDSR(x4)	29.93	29.68	29.47	29.20	29.08	28.82	28.69	28.49	28.35	28.13	28.09	27.85	27.73	27.63	27.56
Meta-EDSR(**our**)	**30.09**	**29.83**	**29.61**	**29.34**	**29.22**	**28.95**	**28.82**	**28.63**	**28.48**	**28.27**	**28.21**	**27.98**	**27.86**	**27.75**	**27.67**

Then, we compare our approach with these baselines to prove the superiority of our method. Our Meta-SR replace the typical upscale module in EDSR [7] and RDN [8] with MRM, called Meta-EDSR and Meta-RDN, respectively.

2) Experimental Results on Arbitrary Scale Factors: The experimental results are shown in Table III. For the bicubic interpolation baseline, simply upscaling the LR image with bicubic interpolation could not introduce any texture or other details to the HR images. It has very limited performance compared with other methods. For the RDN($\times$1) and EDSR($\times$1), they use a CNN (RDN [8] or EDSR [7]) to enhance local details of the preupscaling input images. Compared with the bicubic baseline, they greatly improve performance. But they have a low performance on the large scale factors. Our

TABLE IV

PSNR AND SSIM RESULTS OF OUR META-RDN ON UNSEEN ARBITRARY SCALE FACTORS. WE TRAIN META-RDN ON SCALE FACTORS FROM 1 TO 4 WITH STRIDE 0.1. THE TEST DATA SET IS B100 [50]. THE UNSEEN SCALE FACTORS ARE IN **BLACK**.THE BEST RESULTS ARE UNDERLINE

Methods \ Scale	Metric	x1.40	**x1.42**	**x1.44**	**x1.46**	**x1.48**	x1.50	**x1.52**	**x1.54**	**x1.56**	**x.1.58**	x1.60	**x1.62**
bicubic	PSNR	32.93	**32.78**	**32.61**	**32.46**	**32.31**	32.14	**32.01**	**31.88**	**31.74**	**31.61**	31.49	**31.37**
	SSIM	0.9644	**0.9624**	**0.9605**	**0.9284**	**0.9255**	0.9226	**0.9111**	**0.9083**	**0.9054**	**0.9026**	0.8993	**0.8967**
RDN(x1)	PSNR	36.68	**36.47**	**36.22**	**36.01**	**35.79**	35.57	**35.37**	**35.21**	**35.02**	**34.85**	34.64	**34.49**
	SSIM	0.9631	**0.9610**	**0.9591**	**0.9569**	**0.9548**	0.9527	**0.9507**	**0.9486**	**0.9465**	**0.9453**	0.9418	**0.9397**
Meta-RDN(**our**)	PSNR	36.96	**36.73**	**36.50**	**36.27**	**36.05**	35.88	**35.66**	**35.48**	**35.28**	**35.11**	34.94	**34.76**
	SSIM	0.9644	**0.9624**	**0.9605**	**0.9585**	**0.9564**	0.9546	**0.9524**	**0.9503**	**0.9482**	**0.9461**	0.9439	**0.9419**

TABLE V

COMPARED WITH THE STATE-OF-THE-ART METHODS ON ×2, ×3, AND ×4. THE REPORTED RESULTS OF THE STATE-OF-THE ART METHODS ARE INDEPENDENTLY TRAINED FOR EACH SCALE FACTOR. THE BEST RESULTS ARE IN **BLACK**

Methods	Metric	Set14			B100			Manga109			DIV2K		
		x2	x3	x4	x2	x3	x4	x2	x3	x4	x2	x3	x4
bicubic	PSNR	30.24	27.55	26.00	29.56	27.21	25.96	30.80	26.95	224.89	31.35	28.49	26.92
	SSIM	0.8688	0.7742	0.7227	0.8431	0.7385	0.6675	0.9339	0.8556	0.7866	0.9076	0.8339	0.7774
RDN	PSNR	34.01	**30.57**	28.81	32.34	29.26	27.72	**39.18**	34.13	31.00	35.17	31.39	29.34
	SSIM	0.9212	**0.8468**	0.7871	0.9017	0.8093	0.7419	0.9780	**0.9484**	0.9151	0.9483	0.8931	0.8446
Meta-RDN	PSNR	**34.04**	30.55	**28.84**	**32.36**	**29.30**	**27.75**	39.18	**34.14**	**31.03**	**35.18**	**31.42**	**29.36**
	SSIM	**0.9213**	0.8466	**0.7872**	**0.9019**	**0.8096**	**0.7423**	**0.9782**	0.9483	**0.9154**	**0.9484**	**0.8935**	**0.8448**

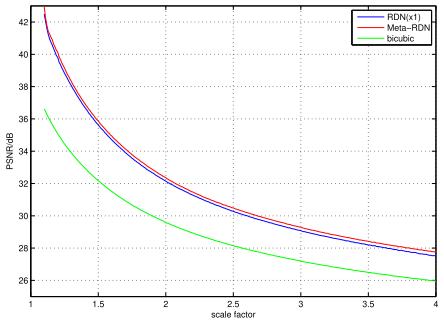

Fig. 5. Curve of PSNR as the scale factor changing. The test data set is B100 [50].

TABLE VI

COMPARED WITH THE STATE-OF-THE-ART METHODS ON ×2, ×3, AND ×4. THE REPORTED RESULTS OF THE STATE-OF-THE ART METHODS ARE INDEPENDENTLY TRAINED FOR EACH SCALE FACTOR. THE BEST RESULTS ARE IN **BLACK**

Methods	Metric	Set5			Urban100		
		x2	x3	x4	x2	x3	x4
bicubic	PSNR	33.66	30.39	28.42	26.88	24.46	23.14
	SSIM	0.9299	0.8682	0.8104	0.8403	0.7349	0.6577
RDN	PSNR	**38.24**	**34.71**	32.47	32.89	28.80	26.61
	SSIM	**0.9614**	0.9296	**0.8990**	0.9353	0.8653	0.8028
Meta-RDN	PSNR	38.23	**34.73**	**32.51**	**33.03**	**28.94**	**26.71**
	SSIM	0.9610	**0.9296**	0.8986	**0.9360**	**0.8673**	**0.8050**

Meta-RDN improves PSNR with 0.27 dB and 0.28 dB compared with the RDN(×1) on scale factor 3 and 4, respectively. And the upscaled input makes this baseline time consuming. For the RDN(×4) and EDSR(×4), our Meta-SR methods achieve better performance than RDN(×4) and EDSR(×4) on all scale factors. When the targeted scale factor is far away from the integral scale factor 4, the scale factor is close to 1, and the performance gaps between our methods and these two baselines. The largest gaps reach to 3.11 and 3.23 dB for the EDSR(×4) and RDN(×4), respectively.

When the scale factor is close to 1, there are larger gaps between our methods and these baselines. The largest gaps reach to 3.11 and 3.23 dB for the EDSR(×4) and RDN(×4), respectively, as the scale factor is 1.1.

Due to the use of weight prediction, both meta-bicu (one of the designed baseline) and our Meta-USR could learn the optimal weights of the filter for each scale factor, while BicuConv shares the same weights of the filter for all the scale factors. The experimental results show that meta-bicu is significantly better than BicuConv, which proves the superiority of the weight prediction module. At the same time, our Meta-RDN is also better than meta-bicu. The main reason lies in the variations of valid field of view (FOV) which can be preserved as a constant value for our Meta-RDN. While for the meta-bicu, the valid FOV will be smaller with the increase of scale factor. It proves the superiority of the LP Mapping, which is better than the bicubic interpolation.

We also test the Meta-RDN on finer variations of scale factors with a smaller stride 0.02. As most test scale factors never occur in the training stage, the experimental results are shown in Table IV and Fig. 5. Thanks to the meta-learning, the weight prediction network learns the experience from trained scale factors and uses this experience to predict weights for unseen scale factors. Thus, our Meta-RDN can work with unseen scale factors and achieve satisfying, continuous, and monotone performance.

Since our Meta-RDN is modified on RDN [8] by replacing the typical upscale module with MRM, however, the RDN learned a separate model for each scale factor with different upscale modules, including ×2, ×3, and ×4. We test our Meta-RDN on six different benchmarks with PSNR and SSIM metrics to compare with RDN on these integer scale factors. As shown in Tables V and VI, the Meta-RDN achieves the comparable or even better results compared with the corresponding baseline RDN [8] on integer scale factors. Since the proposed meta-upscale layer could dynamically predict

Fig. 6. Visual comparison of a single image upsampled with different scale factors by our Meta-RDN. We upscale these images to the same size in the bottom row for more convenient comparison.

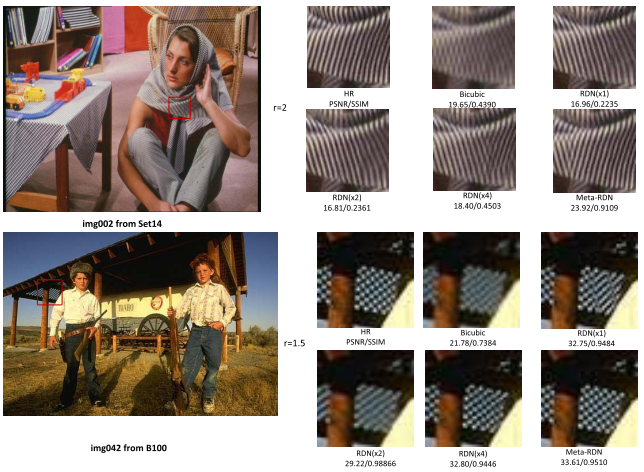

Fig. 7. Visual comparison with four baselines. Our Meta-RDN has a better performance.

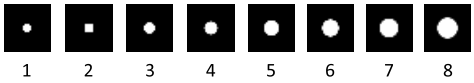

Fig. 8. Specification of eight disk blur kernels. The kernel size is 15.

weights of the filter for each scale, we can train a single model for multiple scale factors and get the best performance at arbitrary scale factors. Moreover, our Meta-RDN network only needs to save one model for test. Our Meta-RDN network is a more efficient method to learn a single SISR model for multiple scale factors and use it to restore better local details.

3) Visualized Comparison: For qualitative and intuitive comparison, we show the visualized results in Figs. 6 and 7. As shown in Fig. 6, we zoom in the same LR image with different scale factors (including noninteger scale factors) based on the proposed Meta-RDN. In Fig. 7, we compare the proposed Meta-RDN with the RDN($\times$1), RDN($\times$2), and RDN($\times$4) for super-resolution of arbitrary scale factors. In local texture regions, the SR images generated by our Meta-USR are sharper and clearer. Since the RDN($\times$1), RDN($\times$2), and RDN($\times$4) share the same weights of filters for all scale factors, the textures of the SR image generated by

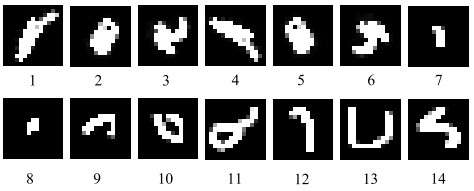

Fig. 9. Specification of 16 motions blur kernels. The kernel size is 15.

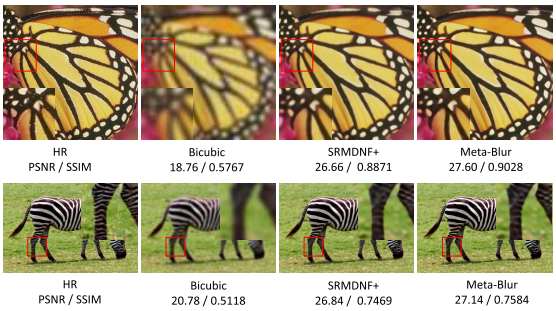

Fig. 10. Visual comparison with the SRMDNF+. When the LR images are badly blurry, our Meta-DeBlur achieve better results than SRMDNF+.

these baseline methods are more blurry than our Meta-RDN. Benefiting from the weight prediction, our Meta-USR can predict a group of independent weights for each scale factor.

In summary, benefited from the proposed MRM, our Meta-RDN achieves better performance on almost all scale factors than other baselines, which demonstrate the effectiveness of the proposed SISR model for arbitrary scale factors.

E. Meta-DeBlur: Single Model for Arbitrary Blur Kernels

1) Settings: In this experiment, our Meta-DeBlur only considers the blur kernel to evaluate the effectiveness of our MRM for multiple blur kernels. Here, we set the noise level as 0. For the isotropic Gaussian, we choose the isotropic Gaussian blur parameterized by the kernel width k_w to compare with the SRMDNF [9], where k_w is set to [0.2, 2] for the scale factor 2 with stride 0.1. For the scale factor 3 or 4, the kernel width is set to [0.2, 3] with stride 0.1. For disk blur, we choose eight disk blur kernels with radius uniformly sampled from

TABLE VII

COMPARISON WITH SRMDNF+ FOR THE SUPER-RESOLUTION OF ARBITRARY ISOTROPIC GAUSSIAN BLUR. THE BACKBONE IS THE RDN [8]. WE REPORT THE PSNR ON B100 [50]. THE BEST RESULTS ARE IN **BLACK**

Methods \ k_w	Scale	0.2	0.4	0.6	0.8	1.0	1.2	1.4	1.6	1.8	2.0
bicubic	x2	29.56	29.41	28.88	28.37	27.85	27.35	26.88	26.45	26.07	25.74
RDN	x2	32.16	32.08	31.03	29.89	28.89	28.03	27.32	26.74	26.26	25.86
SRMDNF+	x2	32.30	32.31	32.36	32.39	32.34	32.18	31.18	31.40	**30.85**	30.28
Meta-DeBlur(3 convs)	x2	**32.30**	**32.33**	**32.30**	**32.41**	**32.36**	**32.20**	**31.82**	**31.40**	30.84	**30.29**
Methods \ k_w	Scale	1.2	1.4	1.6	1.8	2.0	2.2	2.4	2.6	2.8	3.0
bicubic	x3	26.42	26.21	26.03	25.89	25.77	25.67	25.59	25.53	25.48	25.44
RDN	x3	27.47	26.99	26.55	26.15	25.80	25.48	25.20	24.95	24.73	24.54
SRMDNF+	x3	29.28	29.10	29.21	29.14	29.04	28.89	28.41	28.20	27.97	27.72
Meta-DeBlur(3 convs)	x3	**29.29**	**29.25**	**29.23**	**29.16**	**29.06**	**28.91**	**28.71**	**28.51**	**28.27**	**28.02**
bicubic	x4	25.60	25.47	25.32	25.17	25.01	24.85	24.69	24.54	24.40	24.27
RDN	x4	26.84	26.51	26.19	25.89	25.61	25.35	25.11	24.88	24.68	24.50
SRMDNF+	x4	**27.75**	27.71	**27.75**	**27.74**	**27.71**	**27.68**	27.51	27.44	27.33	27.21
Meta-DeBlur(3 convs)	x4	27.74	**27.72**	27.74	27.73	**27.71**	**27.68**	**27.61**	**27.54**	**27.44**	**27.32**

TABLE VIII

AVERAGE PSNR AND SSIM RESULTS FOR BICUBIC DEGRADATION ON DATA SET SET5, SET14, B100. [n m] / 0.1 REPRESENTS THE KERNEL WIDTH RANGES FROM n TO m WITH STRIDE = 0.1. THE REPORTED RESULTS ARE THE AVERAGE PSNR/SSIM ON DIFFERENT KERNEL WIDTH. THE BEST RESULT IS **BLACK**

Dataset	Scale	k_w	bicubic	ZSSR	RCAN	RDN	SRMDNF+	Meta-DeBlur(3 convs)
Set5	x2	[0.2 2] / 0.1	30.77 / 0.8756	32.47 / 0.8970	32.55 / 0.8975	32.50 / 0.8973	37.18 / 0.9519	**37.34 / 0.9527**
	x3	[0.2 3] / 0.1	28.09 / 0.8043	29.66 / 0.8386	29.75 / 0.8396	29.68 / 0.8389	33.84 / 0.9174	**34.04 / 0.9201**
	x4	[0.2 3] / 0.1	27.05 / 0.7679	28.94 / 0.8224	29.11 / 0.8253	29.00 / 0.8231	32.20 / 0.8929	**32.12 / 0.8927**
Set14	x2	[0.2 2] / 0.1	28.05 / 0.7908	29.43 / 0.8230	29.57 / 0.8247	29.45 / 0.8236	33.19 / 0.9034	**33.28 / 0.9053**
	x3	[0.2 3] / 0.1	25.99 / 0.7017	27.05 / 0.7395	27.13 / 0.7414	27.10 / 0.7403	30.03 / 0.8287	**30.13 / 0.8325**
	x4	[0.2 3] / 0.1	25.09 / 0..6616	26.45 / 0.7115	26.56 / 0.7150	26.51 / 0.7122	28.55 / 0.7791	**28.62 / 0.7817**
B100	x2	[0.2, 2] / 0.1	27.66 / 0.7579	28.82 / 0.7938	28.88 / 0.7950	28.82 / 0.7946	31.75 / 0.8835	**31.85 / 0.8863**
	x3	[0.2 3] / 0.1	26.19 / 0.6823	26.76 / 0.7039	26.81 / 0.7057	26.78 / 0.7043	28.90 / 0.7916	**29.00 / 0.7963**
	x4	[0.2 3] / 0.1	25.25 / 0.6283	26.19 / 0.6716	26.26 / 0.6749	26.22 / 0.6722	27.64 / 0.7335	**27.66 / 0.7359**

the interval [1.8, 6], as shown in Fig. 8. For motion blur, we choose 14 realistic-looking motion blur kernels generated by the released code, as shown in Fig. 9. The kernel size is set to 15. For fair comparisons, we keep the backbone of the SRMDNF as same as our Meta-DeBlur, called this baseline SRMDNF+.

2) Experimental Results on Multiple Blur Kernels: The experimental results of single SR model for multiple blur kernels are shown in Tables VII–X. As shown in Tables VII and VIII for Gaussian blur since the RDN only trains on the bicubically downsampled data sets without any blur operations, the RDN obtains poor performance when the test images are blurry. Both our Meta-DeBlur and the SRMDNF+ [9] learn a single model for multiple blur kernels. Different from the SRMDNF+ [9] which uses the dimensionality stretching strategy to concatenate degradation maps with the LR image as the new input, our Meta-DeBlur takes advantage of the meta-learning strategy to predict the weights of the MRM for each blur kernel. As shown in Table IX for disk blur, our meta-blur achieves higher performance than SRMDNF+.The more blurry the blur kernel is, the bigger the performance gap is. For motion blur, the results are shown in Table X. Our meta-blur is slightly better than the SRMDNF+ on most motion blur kernels.

Compared with SRMDNF+ [9], which needs to concatenate deblurring parameters in the front of the network, our MRM is placed at the end of the network and uses the deblurring parameters to predict suitable weights. Although we share the FLM and only change the weights of MRM for each blur kernel, our Meta-DeBlur still achieves slightly higher performance than the SRMDNF+. As shown in Table VII, our Meta-DeBlur achieves better performance than SRMDNF+ when the kernel width is large.

As shown in Fig. 3, there are N regular convolution layers in our MRM. Here, we study how the number of regular convolution layers in the WeightNet1 affects the performance. The results are shown in Table XI. The experimental results show that adding regular convolution layers and predicting the weights for these layers can improve the performance. Note that, the weights of the meta-upscale layer are always learnable in all experiments. For this ablation study, we choose the EDSR as FLM. We only use 10 residual blocks in the ablation study stage. Since the size of the blur kernel is 15×15, if we only predict the weights of the meta-upscale layer, the valid kernel size for deblurring will be limited. Thus, we add the number of the regular convolution layers, so that the size of the learnable kernel for deblurring will be greatly enlarged, which may improve the performance. Here, we choose $\mathbf{N} = 3$ in this article.

Since the MRM can not only be placed at the end of the network but also can be placed at the beginning of the network, we conduct a controlled experiment by placing the MRM at the beginning of the network to compare with our Meta-DeBlur. The results are shown in Table XII. Although the Meta-BeBlur

TABLE IX

COMPARISON WITH SRMDNF+ FOR THE SUPER-RESOLUTION OF ARBITRARY DISK BLUR. THE BACKBONE IS THE EDSR [7]. WE REPORT THE PSNR ON B100 [50]. THE BEST RESULT IS **BLACK**

Methods \ Kernels	scale	1	2	3	4	5	6	7	8
SRMDNF+	x3	28.92	28.86	28.39	27.54	26.77	26.13	25.59	25.12
Meta-Blur	x3	**28.94**	**28.88**	**28.40**	**27.55**	**26.79**	**26.18**	**25.64**	**25.17**

TABLE X

COMPARISON WITH SRMDNF+ FOR THE SUPER-RESOLUTION OF ARBITRARY MOTION BLUR. THE BACKBONE IS THE EDSR [7]. WE REPORT THE PSNR ON SET14 [53]. THE BEST RESULT IS **BLACK**

Methods \ Kernels	scale	1	2	3	4	5	6	7
SRMDNF+	x3	25.71	24.79	**23.48**	21.49	21.47	20.55	**20.58**
Meta-Blur	x3	**25.82**	**24.99**	23.47	**21.62**	**21.66**	**20.64**	20.58
Methods \ Kernels	scale	8	9	10	11	12	13	14
SRMDNF+	x3	20.57	20.20	**20.09**	20.23	19.95	20.20	20.01
Meta-Blur	x3	**20.62**	**20.23**	20.09	**20.28**	**19.99**	**20.28**	**20.08**

TABLE XI

AVERAGE PSNR AND SSIM RESULTS FOR DIFFERENT NUMBER OF REGULAR LEARNABLE CONVOLUTION LAYER ON DATA SET SET14, B100. THE SCALE FACTOR IS 2 AND THE KERNEL WIDTH RANGES [0.2 2]

Dataset \ Number	0	1	2	3	4
B100	31.483 / 0.8815	31.547 / 0.8814	31.562 / 0.8815	31.585 / 0.8819	31.600 / 0.8821
Set14	32.658 / 0.9003	32.796 / 0.9006	32.822 / 0.9008	32.858 / 0.9011	32.881 / 0.9012

TABLE XII

COMPARISON OF MRM MODULE PLACED AT THE BEGINNING OR THE END OF THE NETWORK. THE BACKBONE IS THE RDN [10]. WE REPORT THE PSNR ON B100 [50]. THE BEST RESULTS ARE IN **BLACK**

Methods \ k_w	Scale	0.2	0.4	0.6	0.8	1.0	1.2	1.4	1.6	1.8	2.0
RDN	x2	32.16	32.08	31.03	29.89	28.89	28.03	27.32	26.74	26.26	25.86
Meta-DeBlur(begin)	x2	**32.32**	**32.34**	**32.32**	**32.42**	**32.37**	**32.21**	**31.85**	**31.44**	**30.86**	**30.30**
Meta-DeBlur(end)	x2	32.30	32.33	32.30	32.41	32.36	32.20	31.86	31.40	30.84	30.29

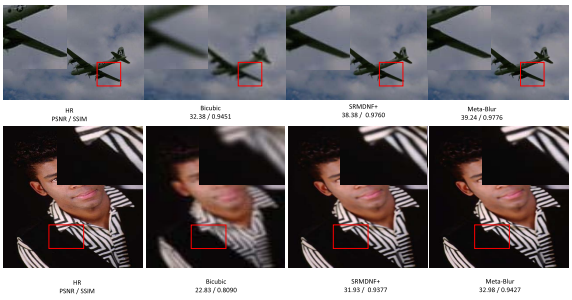

Fig. 11. Visual comparison with the SRMDNF+. The blur kernel is disk blur kernel.

(begin) is consistently better than the Meta-DeBlur (end), the Meta-DeBlur (begin) is inefficient for iterative inference. Thus, we place the MRM at the end of the network for high efficiency.

3) Visualized Comparison: The visualized comparisons are presented in Figs. 10 and 11. Since the LR image is blurry, the SR image upscaled by bicubic interpolation is still blurry. Both our Meta-DeBlur and SRMDNF+ can restore the details of the images by deblurring. Our Meta-DeBlur generates clearer and more natural SR images than SRMDNF+.

TABLE XIII

TIME COST COMPARISON OF OUR META-DEBLUR WITH SRMD ON B100 DATA SET

Methods	FLM	MRM	1 iteration	7 iteration
SRMD	3.71e-2 s		3.71e-2 s	2.60e-1 s
Meta-DeBlur	3.62e-2 s	1.12e-3 s	3.73e-2 s	4.32e-2 s

4) Computational Cost for Blind SR: We also test the running time of each module in our Meta-DeBlur (FLM and MRM) using Tesla P40 with Intel Xeon E5-2670v3 @2.30 GHz. The test data set is B100 (the size of the LR image is 240 × 160), and the scale factor equals to 2. The results are shown in Table XIII. The run time of FLM (3.62e-2 s per image) is about 30 times of the MRM (1.12e-3 s per image). Compared with the FLM, the running time of the MRM can be neglected. And the run time of the SRMDNF+ is 3.71e-2 s per image.

However, in the real world, the blur kernel of the testing LR image is often unknown, namely, Blind SR, some work, e.g., IKC [12], has proposed to iteratively correct the blur kernel. The IKC [12] which is based on the SRMDNF solves the blind SR by iteratively correcting the blur kernel which

TABLE XIV
EXPERIMENTAL RESULTS OF OUR META-USR WHEN THE NOISE LEVEL = 0. THE BACKBONE IS THE RDN [8]. WE REPORT THE PSNR ON B100 [50]

scale \ k_w	k1.0	k1.1	k1.2	k1.3	k1.4	k1.5	k1.6	k1.7	k1.8	k1.9	k2.0
x2.0	32.137	32.080	31.981	31.835	31.646	31.419	31.165	30.890	30.596	30.305	30.003
x2.1	31.694	31.663	31.596	31.496	31.361	31.190	30.990	30.762	30.510	30.227	29.893
x2.2	31.295	31.274	31.224	31.151	31.049	30.910	30.743	30.552	30.338	30.103	29.843
x2.3	30.765	30.739	30.693	30.623	30.561	30.511	30.410	30.324	30.210	30.041	29.691
x2.4	30.579	30.583	30.558	30.513	30.450	30.363	30.252	30.119	29.958	29.780	29.572
x2.5	30.302	30.313	30.300	30.268	30.218	30.152	30.064	29.956	29.831	29.680	29.512
x2.6	30.017	30.031	30.028	30.008	29.973	29.922	29.856	29.773	29.669	29.549	29.408
x2.7	29.753	29.768	29.769	29.761	29.737	29.697	29.643	29.573	29.492	29.392	29.274
x2.8	29.508	29.526	29.530	29.524	29.505	29.472	29.428	29.373	29.302	29.218	29.115
x2.9	29.260	29.283	29.294	29.296	29.284	29.262	29.226	29.181	29.124	29.053	28.970
x3.0	29.076	29.098	29.111	29.117	29.111	29.097	29.070	29.038	28.994	28.941	28.877

TABLE XV
AVERAGE PSNR/SSIM OF OUR META-USR ON DIFFERENT BLUR KERNELS. THE BACKBONE IS THE RDN [8]. WE REPORT THE PSNR/SSIM ON B100 [50]. THE KERNEL WIDTH RANGES FROM 1 TO 2 WITH STRIDE 0.1

noise level \ scale	x2.0	x2.2	x2.4	x2.6	x2.8	x3.0
0	31.26 / 0.8685	30.77 / 0.8581	30.25 / 0.8421	29.84 / 0.8293	29.41 / 0.8132	29.05 / 0.8032
10	28.00 / 0.7445	27.76 / 0.7345	27.55 / 0.7248	27.35 / 0.7163	27.143 / 0.7067	26.97 / 0.7033
20	26.87 / 0.6939	26.65 / 0.6842	26.46 / 0.6755	26.28 / 0.6674	26.10 / 0.6590	25.93 / 0.6566
30	26.14 / 0.6611	25.93 / 0.6519	25.75 / 0.6438	25.57 / 0.6363	25.40 / 0.6286	25.24 / 0.6267
40	25.61 / 0.6377	25.40 / 0.6289	25.22 / 0.6213	25.05 / 0.6142	24.88 / 0.6071	24.72 / 0.6054

repeatedly passes through the SRMDNF network seven times. Repeatedly passing through the whole SRMDNF+ seven times will cost 2.601e-1 s per image. Different from the SRMDNF+, when we apply Meta-DeBlur to solve Blind SR as the same as IKC [12], our Meta-DeBlur only needs to recalculate the MRM seven times to generate a new SR image. The running time is 4.31e-2 s per image. It is more efficient. The SRMDNF+ has to concatenate the new blur kernel with the input image at first. And then it has to pass a new input image through the whole network again. Thus, the blind SR based on the SRMDNF+ is very time consuming. But, the MRM in our Meta-DeBlur is lightweight for iteratively searching or iteratively correcting the blur kernel.

Compared with the SRMDNF+, our Meta-DeBlur achieves slightly higher results and is less time consuming for real-world application. We argue that our Meta-DeBlur is the first postdeblurring method which performs deblurring at the end of the network.

F. Meta-USR: Single Model for Multiple Degradation Factors

1) Settings: In this experiment, we learn a single model for super-resolution of arbitrary degradation parameters (multiple scale factors, multiple blur kernels, and multiple noise levels). The range of scale factor is set to [2, 3] with stride 0.1. For the blur kernel, we choose the Gaussian Blur with kernel width [1, 2] whose stride is 0.1. For the noise level, we choose the additive white Gaussian noise. The level range of the noise level is set to [0, 40] with stride 1. Our Meta-USR is the first method to learn a single model for all these degradation parameters.

2) Experimental results on Multiple Degradation Factors: Our Meta-USR is the first unified super-resolution network for arbitrary degradation parameters. Here, we focus on how the different degradation factors affect PSNR. The experimental results of Meta-USR are shown in Tables XIV and XV. As shown in Table XIV and Fig. 12, when the scale factor is small, the kernel width has a bigger impact on the PSNR. When the width increases, the PSNR drops very fast (about 0.2–0.3 dB). The blur kernel is uppermost degradation. On the contrary, the scale factor is the uppermost degradation factor when the scale factor is large. That is, the PSNR drops along with the increase of the width very slowly (about 0.04 dB). As shown in Table XV, if we take the noise level into consideration, the PSNR drops sharply as the noise level increases.

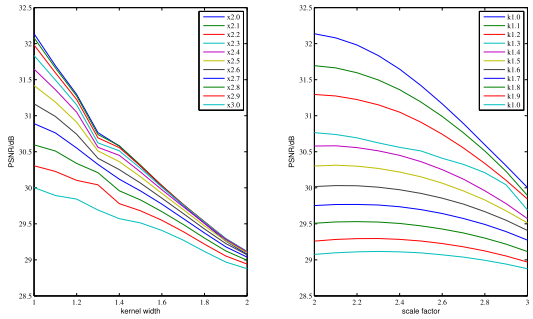

Fig. 12. PSNR curve of our Meta-USR. The test data set is B100 [50]. "×2.5" denotes that we fix scale factor as 2.5 and varies the kernel width. "k1.5" denotes that we fix the kernel width as 1.5 and vary the scale factor.

In the Meta-USR, the meta-upscale layer can upscale the LR images, while the regular convolution layers in MRM do not change the size of the LR image. What if WeightNet2 only take the scale factor as input without blur kernel or noise to predict weights for the meta-upscale layer? We design a

TABLE XVI

COMPARISON OF EXPERIMENTAL RESULTS FOR OUR META-USR BETWEEN SEPARATING THE DEGRADATION PARAMETERS OR COMBINING THE DEGRADATION PARAMETERS FOR EACH MODULE. THE BACKBONE IS THE RDN [8]. WE REPORT THE PSNR/SSIM ON B100 [50]

Methods \ Degradation (s,k_w,n)	(2, 1.0, 0)	(2.5, 1.0, 0)	(2.5, 1.0, 20)	(3, 1.0, 20)
Meta-USR-separate	31.920 / 0.8946	30.071 / 0.8443	26.861 / 0.6996	26.272 / 0.6765
Meta-USR	32.156 / 0.8961	30.302 / 0.8478	26.900 / 0.7005	26.320 / 0.6778

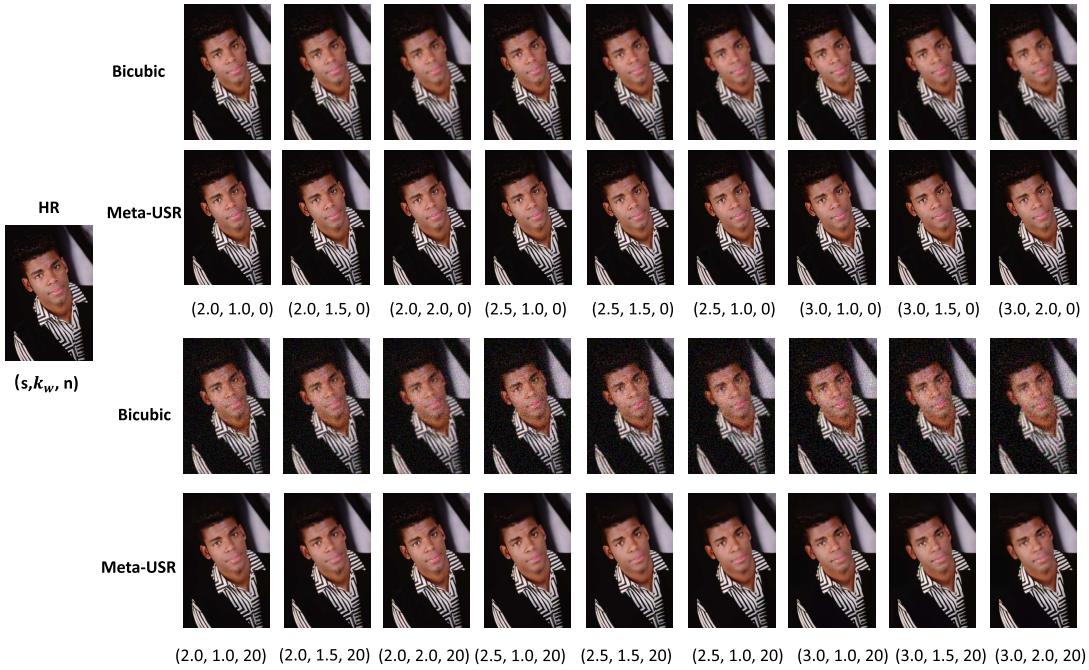

Fig. 13. Visual results of our Meta-USR to upscale a single LR image with various degradation parameters. (s, k_w, n) denotes the scale factor is s, the kernel width of the Gaussian blur kernel is k_w, and the level of noise is n.

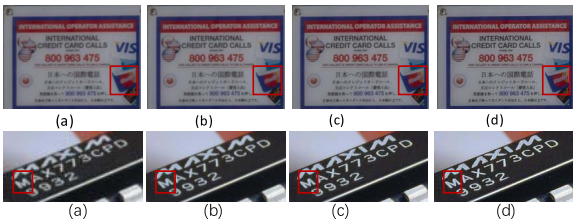

Fig. 14. Visual comparison with the SRMDNF and RDN. The scale factor is 2. (a) LR. (b) RDN. (c) SRMDNF. (d) Ours.

new baseline, the WeightNet1 takes the blur kernel, and noise level as input and the WeightNet2 takes coordinates and scale factor as input, called Meta-USR-Separate. The experimental results are shown in Table XVI. As shown in Table XVI, our Meta-SR achieves better performance than the Meta-USR-separate. This result shows that solving the deblurring, denoising, and upscaling with divided modules is not better than considering all these factors with one module.

3) Visualized Comparison: The visual results are shown in Fig. 9. For high level of noise, these generated SR images are smooth and miss some details.

As we all know, real-world images are complex and suffer from various blur kernels and various levels of noise. Both the blur kernel and the noise level are unknown. And our Meta-USR is a single model for multiple degradation parameters, and it no longer needs to train and store hundreds of models for different degradation factors and noise levels. For different scenarios, the user only needs to customize the scale factor, blur kernel, and noise which is suitable for the scenarios. It will be very useful in practice. Since there are no ground-truth HR images, we only provide the visualized comparison. To find the degradation parameters with good visual quality, we follow the SRMD [35] to adopt a grid search strategy.

Fig. 14 shows two real LR images. The RDN [9] only considers the scale factor. We can observe that both SRMD [35] and Meta-USR can produce more natural HR images than RDN. The images produced by RDN are affected by compression artifacts, and it cannot recover sharp edges. Compared with SRMD, our Meta-USR produces more sharp edges.

V. CONCLUSION

In this work, we propose a meta-learning-based SISR model for solving the SR problem with arbitrary degradation parameters. The key module, i.e., the MRM, can jointly predict the weights of the filters for upscaling, deblurring, and denoising. Furthermore, as the MRM is a lightweight network and can be placed at the end of the network, it will be very efficient for repeatedly calculating the MRM in the case of blind SR. Thus, the real-time iterative methods for blind SR can benefit from our Meta-USR, which will be the research problems in future

work. The extensive experimental results demonstrate the superiority of Meta-USR in performance and computational efficiency.

REFERENCES

[1] B. K. Gunturk, Y. Altunbasak, and R. M. Mersereau, "Super-resolution reconstruction of compressed video using transform-domain statistics," *IEEE Trans. Image Process.*, vol. 13, no. 1, pp. 33–43, Jan. 2004.

[2] W. W. W. Zou and P. C. Yuen, "Very low resolution face recognition problem," *IEEE Trans. Image Process.*, vol. 21, no. 1, pp. 327–340, Jan. 2012.

[3] W. Shi *et al.*, "Cardiac image super-resolution with global correspondence using multi-atlas patchmatch," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.* Cham, Switzerland: Springer, 2013, pp. 9–16.

[4] D. Yıldırım and O. Güngör, "A novel image fusion method using IKONOS satellite images," *J. Geodesy Geoinformation*, vol. 1, no. 1, pp. 27–34, 2012.

[5] Y. Bai, Y. Zhang, M. Ding, and B. Ghanem, "SOD-MTGAN: Small object detection via multi-task generative adversarial network," in *Proc. ECCV*, 2018, pp. 206–221.

[6] M. Haris, G. Shakhnarovich, and N. Ukita, "Task-driven super resolution: Object detection in low-resolution images," 2018, *arXiv:1803.11316*. [Online]. Available: http://arxiv.org/abs/1803.11316

[7] 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 (CVPRW)*, Jul. 2017, pp. 136–144.

[8] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, "Residual dense network for image super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2472–2481.

[9] K. Zhang, W. Zuo, and L. Zhang, "Learning a single convolutional super-resolution network for multiple degradations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3262–3271.

[10] 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. (CVPR)*, Jun. 2016, pp. 1874–1883.

[11] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. ECCV*, 2014, pp. 818–833.

[12] J. Gu, H. Lu, W. Zuo, and C. Dong, "Blind super-resolution with iterative kernel correction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1604–1613.

[13] X. Hu, H. Mu, X. Zhang, Z. Wang, T. Tan, and J. Sun, "Meta-SR: A magnification-arbitrary network for super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1575–1584.

[14] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *Proc. ECCV*, 2014, pp. 184–199.

[15] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1646–1654.

[16] J. Kim, J. K. Lee, and K. M. Lee, "Deeply-recursive convolutional network for image super-resolution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1637–1645.

[17] Y. Tai, J. Yang, X. Liu, and C. Xu, "MemNet: A persistent memory network for image restoration," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 4539–4547.

[18] Y. Tai, J. Yang, and X. Liu, "Image super-resolution via deep recursive residual network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3147–3155.

[19] C. Ledig *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4681–4690.

[20] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[21] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, "Image super-resolution using very deep residual channel attention networks," 2018, *arXiv:1807.02758*. [Online]. Available: http://arxiv.org/abs/1807.02758

[22] X. Wang, K. Yu, C. Dong, and C. Change Loy, "Recovering realistic texture in image super-resolution by deep spatial feature transform," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 606–615.

[23] M. Haris, G. Shakhnarovich, and N. Ukita, "Deep back-projection networks for super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1664–1673.

[24] N. Ahn, B. Kang, and K.-A. Sohn, "Fast, accurate, and lightweight super-resolution with cascading residual network," in *Proc. ECCV*, 2018, pp. 252–268.

[25] S.-J. Park, H. Son, S. Cho, K.-S. Hong, and S. Lee, "SRFEAT: Single image super-resolution with feature discrimination," in *Proc. ECCV*, 2018, pp. 439–455.

[26] J. Li, F. Fang, K. Mei, and G. Zhang, "Multi-scale residual network for image super-resolution," in *Proc. ECCV*, 2018, pp. 517–532.

[27] K. Zhang, D. Tao, X. Gao, X. Li, and J. Li, "Coarse-to-fine learning for single-image super-resolution," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 5, pp. 1109–1122, May 2017.

[28] C. Chen, Z. Xiong, X. Tian, Z.-J. Zha, and F. Wu, "Camera lens super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1652–1660.

[29] X. Xu, Y. Ma, and W. Sun, "Towards real scene super-resolution with raw images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1723–1731.

[30] X. He, Z. Mo, P. Wang, Y. Liu, M. Yang, and J. Cheng, "ODE-inspired network design for single image super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1732–1741.

[31] Z. Li, J. Yang, Z. Liu, X. Yang, G. Jeon, and W. Wu, "Feedback network for image super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3867–3876.

[32] T. Dai, J. Cai, Y. Zhang, S.-T. Xia, and L. Zhang, "Second-order attention network for single image super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 11065–11074.

[33] W. Zhang, Y. Liu, C. Dong, and Y. Qiao, "RankSRGAN: Generative adversarial networks with ranker for image super-resolution," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3096–3105.

[34] J. Cai, H. Zeng, H. Yong, Z. Cao, and L. Zhang, "Toward real-world single image super-resolution: A new benchmark and a new model," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3086–3095.

[35] K. Zhang, W. Zuo, S. Gu, and L. Zhang, "Learning deep CNN denoiser prior for image restoration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3929–3938.

[36] M. V. Afonso, J. M. Bioucas-Dias, and M. A. T. Figueiredo, "Fast image recovery using variable splitting and constrained optimization," *IEEE Trans. Image Process.*, vol. 19, no. 9, pp. 2345–2356, Sep. 2010.

[37] K. Zhang, W. Zuo, and L. Zhang, "Deep plug-and-play super-resolution for arbitrary blur kernels," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1671–1681.

[38] A. Shocher, N. Cohen, and M. Irani, "'zero-shot' super-resolution using deep internal learning," in *Proc. IEEE CVPR*, Jun. 2018, pp. 3118–3126.

[39] M. Andrychowicz *et al.*, "Learning to learn by gradient descent by gradient descent," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3981–3989.

[40] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017.

[41] Q. Cai, Y. Pan, T. Yao, C. Yan, and T. Mei, "Memory matching networks for one-shot image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4080–4088.

[42] Y.-X. Wang and M. Hebert, "Learning to learn: Model regression networks for easy small sample learning," in *Proc. ECCV*, 2016, pp. 616–634.

[43] J. Vanschoren, "Meta-learning: A survey," 2018, *arXiv:1810.03548*. [Online]. Available: http://arxiv.org/abs/1810.03548

[44] C. Lemke, M. Budka, and B. Gabrys, "Metalearning: A survey of trends and technologies," *Artif. Intell. Rev.*, vol. 44, no. 1, pp. 117–130, Jun. 2015.

[45] T. Yang, X. Zhang, W. Zhang, and J. Sun, "Metaanchor: Learning to detect objects with customized anchors," in *Proc. NIPS*, 2018, pp. 320–330.

[46] R. Hu, P. Dollár, K. He, T. Darrell, and R. Girshick, "Learning to segment every thing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 4233–4241.

[47] Y. Jo, S. W. Oh, J. Kang, and S. J. Kim, "Deep video super-resolution network using dynamic upsampling filters without explicit motion compensation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3224–3232.

[48] Q. Fan, D. Chen, L. Yuan, G. Hua, N. Yu, and B. Chen, "Decouple learning for parameterized image operators," in *Proc. ECCV*, 2018, pp. 442–458.

[49] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.
[50] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. 8th IEEE Int. Conf. Comput. Vis. (ICCV)*, 2001, pp. 416–423.
[51] E. Agustsson and R. Timofte, "NTIRE 2017 challenge on single image super-resolution: Dataset and study," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 126–135.
[52] M. Bevilacqua, A. Roumy, C. Guillemot, and M. L. Alberi-Morel, "Low-complexity single-image super-resolution based on nonnegative neighbor embedding," in *Proc. BMVC*, 2012, pp. 135.1–135.10.
[53] R. Zeyde, M. Elad, and M. Protter, "On single image scale-up using sparse-representations," in *Proc. Int. Conf. Curves Surf.*, 2010, pp. 711–730.
[54] A. Fujimoto, T. Ogawa, K. Yamamoto, Y. Matsui, T. Yamasaki, and K. Aizawa, "Manga109 dataset and creation of metadata," in *Proc. 1st Int. Workshop coMics Anal., Process. Understand. (MANPU)*, 2016, pp. 1–5.
[55] J.-B. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 5197–5206.
[56] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

Xuecai Hu received the B.Sc. degree from the University of Science and Technology of China, Hefei, China, in 2015, where he is currently pursuing the Ph.D. degree.

His research interests include computer vision, image enhancement, and semantic segmentation.

Zhang Zhang (Member, IEEE) received the B.Sc. degree in computer science and technology from the Hebei University of Technology, Tianjin, China, in 2002, and the Ph.D. degree in pattern recognition and intelligent systems from the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2008.

He is currently an Associate Professor with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences. His research interests include human attribute recognition, person re-identification, and human activity recognition.

Caifeng Shan (Senior Member, IEEE) received the B.Eng. degree from the University of Science and Technology of China, Hefei, China, in 2001, the M.Eng. degree from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2004, and the Ph.D. degree in computer vision from the Queen Mary University of London, London, U.K., in 2007.

He has authored more than 100 articles and 60 patent applications. His research interests include computer vision, pattern recognition, image and video analysis, machine learning, biomedical imaging, and related applications.

Dr. Shan has served as an Associate Editor or a Guest Editor for several scientific journals, including the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY and the IEEE JOURNAL OF BIOMEDICAL AND HEALTH INFORMATICS.

Zilei Wang (Member, IEEE) received the B.S. and Ph.D. degrees in control science and engineering from the University of Science and Technology of China (USTC), Hefei, China, in 2002 and 2007, respectively.

He was a Post-Doctoral Research Fellow with the National University of Singapore, Singapore. He is currently an Associate Professor with the Department of Automation, USTC, where he is also the Founding Leader of the Vision and Multimedia Research Group. His current research interests include computer vision, multimedia, and deep learning.

Liang Wang received the B.Eng. and M.Eng. degrees from Anhui University, Hefei, China, in 1997 and 2000, respectively, and the Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences (CASIA), Beijing, China, in 2004.

From 2004 to 2010, he was a Research Assistant with the Imperial College London, London, U.K., and Monash University, Clayton, VIC, Australia, a Research Fellow with the University of Melbourne, Parkville, VIC, Australia, and a Lecturer with the University of Bath, Bath, U.K., respectively. He is currently a Full Professor of the Hundred Talents Program, National Laboratory of Pattern Recognition, CASIA. His major research interests include machine learning, pattern recognition, and computer vision.

Dr. Wang is the International Association of Pattern Recognition (IAPR) Fellow.

Tieniu Tan (Fellow, IEEE) received the B.Sc. degree in electronic engineering from Xi'an Jiaotong University, Xi'an, China, in 1984, and the M.Sc. and Ph.D. degrees in electronic engineering from the Imperial College London, London, U.K., in 1986 and 1989, respectively.

He is currently a Professor with the Center for Research on Intelligent Perception and Computing, National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China. His research interests include biometrics, image, and video understanding, information hiding, and information forensics.

Dr. Tan is a fellow of Chinese Academy of Sciences (CAS), The World Academy of Sciences (TWAS), Brazilian Academy of Sciences (BAS), the International Association of Pattern Recognition (IAPR), and the U.K. Royal Academy of Engineering. He is the Past President of the IEEE Biometrics Council.