CCR: Clustering and Collaborative Representation for Fast Single Image Super-Resolution

Yongbing Zhang, *Member, IEEE,* Yulun Zhang, *Student Member, IEEE,*
Jian Zhang, *Member, IEEE,* and Qionghai Dai, *Senior Member, IEEE*

***Abstract*—Clustering and collaborative representation are recently used in fast single image super-resolution (SR). In this paper, we propose an effective and fast single image super-resolution (SR) algorithm by combining clustering and collaborative representation (CCR). In particular, we first cluster the feature space of low-resolution (LR) images into multiple LR feature subspaces and group the corresponding high-resolution (HR) feature subspaces. The local geometry property learned from the clustering process is used to collect numerous neighbor LR and HR feature subsets from the whole feature spaces for each cluster center. Multiple projection matrices are then computed via collaborative representation to map LR feature subspaces to HR subspaces. For an arbitrary input LR feature, the desired HR output can be estimated according to the projection matrix, whose corresponding LR cluster center is nearest to the input. Moreover, by learning statistical priors from clustering process, our clustering based SR algorithm would further decrease the computational time in the reconstruction phase. Extensive experimental results on commonly used datasets indicate that our proposed SR algorithm obtains compelling SR images quantitatively and qualitatively against many state-of-the-art methods.**

***Index Terms*—Clustering, collaborative representation, feature subspace, projection matrix, statistical prior, super-resolution.**

I. Introduction

SINGLE image super-resolution (SR), an important and classical problem in image processing and computer vision, aims at restoring a high-resolution (HR) image from its degraded low-resolution (LR) measurement, while minimizing visual artifacts as far as possible [1]. SR is a typical ill-posed inverse problem since a multiplicity of solutions exist for any input LR pixel [2]. To mitigate and deal with such a problem, most of SR methods constrain the solution with strong prior information, which can be obtained by learning from a set of training data, using various regularization methods, and assuming a certain distribution of image features [3]. Motivated by those ideas, researchers have proposed lots of SR techniques, which can be broadly grouped into three categories: interpolation-based methods, reconstruction-based methods, and learning-based methods.

Interpolation-based methods [4]–[8], the basic type of SR algorithms, are widely used to produce zoom-in images for its simplicity. Local covariance coefficients, fixed-function kernels [5], or adaptive-structure kernels [6], [7] are usually exploited in the interpolation process. Although these methods are simple and efficient for real-time SR application, in many cases they are more prone to produce visual artifacts, such as aliasing, blurring, blocking, and jagged artifacts, as magnification becomes larger. A further and detailed survey about this type of SR methods and their shortcomings is conducted in [9].

Reconstruction-based methods [10]–[17] assume that the observed LR image is obtained from several degradation factors such as down-sampling and blurring, and emphasize the reconstruction constraint. As a result, the down-sampled version of the target HR image should be close to the corresponding LR one. In order to obtain a reliable solution, various priors, such as edge-directed priors [12], [14]–[16] and similarity redundancy [17], are widely introduced in this type of SR methods. Kernel PCA was utilized to build a prior model for face image SR in [13]. Nonetheless, the reconstruction-based methods sometimes produce HR images with too sharp and unnatural edges, and often introduce ringing artifacts, especially along the salient edges.

Accordingly, by taking the advantage of machine learning (ML) techniques, learning-based SR methods have been obtaining superior and challenging results, and receiving ever-increasing interest in recent years [18]–[39]. Neighbor embedding (NE), sparse coding, and deep learning approaches are the three main types of learning-based SR algorithms.

NE-based methods usually assume that small LR and HR patches form low-dimensional nonlinear manifolds with similar local geometry. One of the representative works is the NE method proposed by Chang *et al.* [18], who used locally linear embedding (LLE) [40] to generate HR patches. LLE assumes that each sample and its neighbors lie on or near a locally linear patch as long as enough samples are available and then finds a low-dimensional embedding that best preserves the local geometry of images. Bevilacqua *et al.* [21] proposed another NE SR algorithm based on nonnegative neighbor embedding and a least squares (LS) approximation of the LR patches with a non-negativity constraint, and proved the

This work was supported in part by the National Natural Science Foundation of China under Grant 61571254, Grant U1301257, Grant U1201255, and Grant 61572047, in part by Guangdong Natural Science Foundation 2014A030313751, and in part by Postdoctoral Science Foundation of China (2015M580018).

Y. B. Zhang and Y. L. Zhang are with Graduate School at Shenzhen, Tsinghua University, Shenzhen 518055, China. Y. L. Zhang is also with Department of Automation, Tsinghua University, Beijing 100084, China (e-mail: zhang.yongbing@sz.tsinghua.edu.cn; zhangyl14@mails.tsinghua.edu.cn).

J. Zhang is with the National Engineering Laboratory for Video Technology, School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China (e-mail: jian.zhang@pku.edu.cn).

Q. Dai is with TNLIST and Department of Automation, Tsinghua University, Beijing 100084, China (e-mail: qhdai@tsinghua.edu.cn).

effectiveness of the choice of having non-negative weights. In the recent dual-geometric neighbor embedding (DGNE) [29] approach for SR, multiview features and local spatial neighbors of patches are explored to find a feature-spatial manifold embedding for SR. In order to optimally represent patches with complicated structures, NE-based algorithms often need to search a vast reference dataset and also lack efficiency in practice.

Coding-based methods have been showing great potential for high-quality image SR by learning an overcomplete LR and HR dictionary pair. Yang *et al.* [19], [30] proposed a sparse coding based SR algorithm, where the HR feature can be recovered as a sparse linear combination of the learned HR dictionary atoms by assuming that LR and HR features share the same sparse reconstruction coefficients. However, the optimization function used in [30], and [19] includes the l_1-norm regularity, making the SR algorithm suffer from highly intensive computation for each input LR feature. This problem was further addressed in [20], where principal component analysis (PCA) and orthogonal matching pursuit (OMP) [41] were used to reduce the dimensionality of LR features and solve sparse representation respectively. Zhang *et al.* [37] proposed a dual sparsity model for image SR. Yang *et al.* advanced support vector regression (SVR) with image sparse representation in [23]. Xiong *et al.* [24] proposed a statistical method for exploiting the one-to-many correspondence between LR and HR patches. Kang *et al.* [25] learned image sparse representations for modeling the relationship between LR and HR image patches in terms of the learned dictionaries for image patches with and without blocking artifacts, respectively. A Bayesian method using a beta process was applied to learn the overcomplete dictionaries in [26] and a statistical prediction model based on sparse representations of LR and HR image patches was used in [27], both of which improved the SR results at the cost of suffering from high computation time both in the dictionary learning and SR phases. At the same time fast SR algorithms were proposed in [22], [31]–[34], [38] by combing dictionary learning or clustering and regression. Anchored neighborhood regression (ANR) [22] learned sparse dictionaries and regressors anchored to the dictionary atoms and its improved variant A+ [31] obtained better quality by learning the regressors in the full training feature spaces. Both ANR [22] and A+ [31] used collaborative representation to compute the mapping matrices offline for fast and accurate SR. Clustering was used in Simple function (SF) [32], which was proposed to cluster feature space into numerous subspaces and learned effective mapping functions by linear regression for each subspace. Zhu *et al.* [33] proposed a novel algorithm for fast single image super-resolution based on self-example learning and sparse representation. Using a similar framework to [32], multiple linear mapping (MLM) [34] learned LR/HR subdictionaries in each clustered feature subspaces, followed by a fast non-local means (NLM) [42] SR enhancement algorithm. Although these fast SR methods are computationally effective, ANR [22], SF [32], and MLM (without NLM enhancement) [34] failed to improve the SR quality significantly for the lack of diversities in each clustered feature subspaces. Apparently, fast and high-quality SR algorithms are highly desirable for practical and real-time applications.

Deep learning techniques have also been attracting increasing interest in its usage for image SR. Cui *et al.* [35] proposed a deep network cascade (DNC) to gradually up-scale LR images layer by layer under the notion internal example-based algorithm [43]. Because independent optimization of the self-similarity search process and auto-encoder was needed in each layer of the cascade, DNC [35] failed to obtain an end-to-end solution. This problem was addressed in the super-resolution convolutional neural network (SRCNN) [28], in which an end-to-end mapping represented as a deep convolutional neural network (CNN) [44] between the LR and HR images was learned. Although SRCNN [28] achieved superior performance compared to most of the existing SR methods, it takes about 3 days in the training phase with a GPU and also be hard for people to design and train his own deep model.

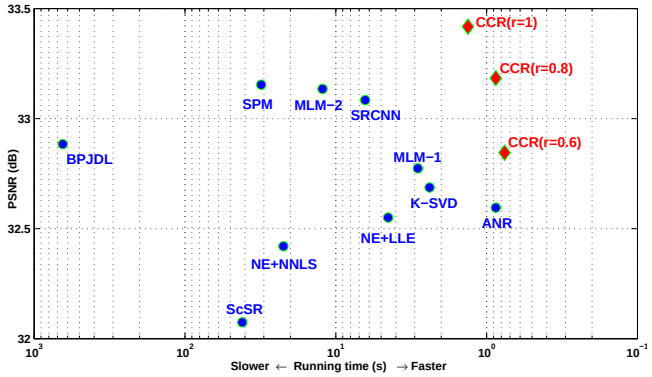


Fig. 1. Average running time versus PSNR on 10 images for the tested methods (scaling factor $s = 3$). The proposed CCR (shown in red) obtains the state-of-the-art SR results and maintains high and competitive speed in comparison to other learning-based methods (**NE+LLE** [18], **ScSR** [19], **K-SVD** [20], **NE+NNLS** [21], **ANR** [22], **BPJDL** [26], **SPM** [27], **SRCNN** [28], **MLM** [34] (including **MLM-1** and **MLM-2**)). More details are shown in Table I and Table II.

Motivated by the previous fast learning-based work [22], [31], [32], [34], we propose a fast SR algorithm based on clustering and collaborative representation (CCR). The newly proposed SR algorithm consists of two phases: learning and reconstruction. In the learning phase, we first extract and obtain two pools of co-occurrence LR-HR features, a small one for finding cluster centers and PCA transform matrix of LR feature space, and a large one for grouping neighbor LR-HR feature subsets. Different from [32], [34] that learned multiple mapping functions or subdictionaries in the clustered subspaces, the proposed method collects neighbor LR-HR feature subsets from more than one clustered subspaces by considering the local geometry between feature subspaces. Once the neighbor LR-HR feature subsets are grouped, multiple projection matrices are then computed to each LR feature cluster center by utilizing collaborative representation [45]. In the reconstruction phase, for each LR feature extracted from the LR input image, we search the most matched projection matrix and multiply it by the LR feature to obtain the corresponding HR feature. Due to simple projection matrices grouped by considering local geometry of data, our CCR-based SR approach achieves superior high quality results compared with SF [32] and ANR [22], while maintaining computation

efficiency (see Fig. 1) without additional enhancement algorithm, like back-projection [46] in [30], [19], and NLM [42] in [34]. More details about the differences between our CCR-based algorithm and other state-of-the-art SR methods will be discused in Section II-E.

Overall, the contributions of this work are mainly summarized as follows:

1) We cluster the LR feature spaces into numerous subspaces and group the neighbor LR/HR feature subsets by considering local geometry of data without learning dictionaries or searching neighbors in only one cluster for each LR cluster center.
2) We further speed up our SR method by considering a small part of LR cluster centers, since in the clustering stage we have learned statistical priors that most of the LR/HR features fall into a small number of feature cluster centers.
3) We demonstrate that CCR is simple and useful in SR and can achieve high-quality and speed efficiently without using extra enhancement algorithms.

The remainder of the paper is organized as follows. The proposed fast CCR-based SR algorithm is detailed in Section II. We report extensive experimental results and conclude the paper in Section III and Section IV successively.

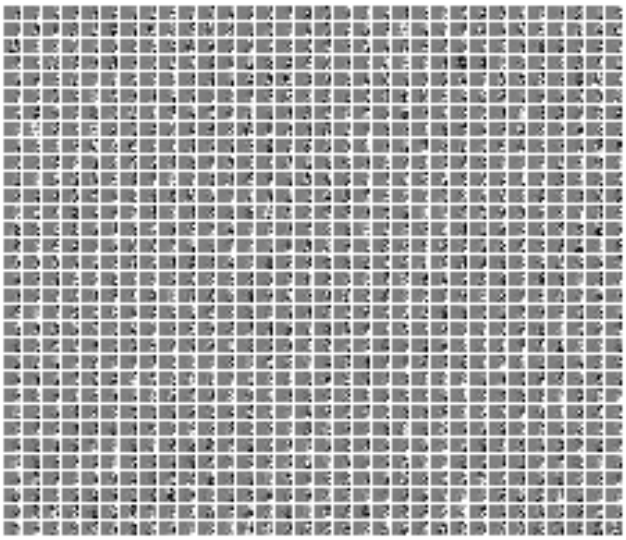


Fig. 2. A set of 1024 cluster centers learned from 135,581 LR features. The length of each LR feature vector is 30 and we show it in a small 5×6 box. The order of cluster centers is sorted by the number of clustered features and presented from left to right and top to bottom.

II. Fast SR Based on Clustering and Collaborative Representation

In this section, we first present how to extract two pools of LR/HR feature sets from a training image set and cluster them into numerous subspaces. We then use the clustered feature subspaces to group neighbor features, followed by computing multiple projection matrices. Finally, the CCR-based SR algorithm is outlined.

In the following discussion, we denote $\boldsymbol{I}_{L/H}$, $\boldsymbol{p}_{L/H}$, and $\boldsymbol{y}_{L/H}$ as an LR/HR image, patch, and feature, respectively. $\boldsymbol{c}$ and $\boldsymbol{F}$ denote a cluster center in the LR feature space and its corresponding projection matrix. Bold lowercase letters denote vectors. Bold uppercase letters denote regular matrices. Plain lowercase letters are used as scalars.

A. Feature Extraction and Clustering

Let $\{\boldsymbol{I}_L^i\}_{i=1}^{N_I}$ and $\{\boldsymbol{I}_H^i\}_{i=1}^{N_I}$ be the LR and HR training images, we first upscale $\{\boldsymbol{I}_L^i\}_{i=1}^{N_I}$ to $\{\boldsymbol{I}_B^i\}_{i=1}^{N_I}$ using Bicubic interpolation with a scaling factor s. Using $\{\boldsymbol{I}_B^i\}_{i=1}^{N_I}$ and $\{\boldsymbol{I}_H^i\}_{i=1}^{N_I}$, we collect two pools of samples, a small one $\{\boldsymbol{p}_L^i\}_{i=1}^{N_1}$ and a larger LR-HR patch pairs $\{\boldsymbol{p}_L^i\}_{i=1}^{N_2}$ and $\{\boldsymbol{p}_H^i\}_{i=1}^{N_2}$. To obtain better performance and enhance robustness, we do not operate with the LR/HR patches, but extract features from them. For HR patch $\boldsymbol{p}_H$, its corresponding feature $\boldsymbol{y}_H$ is obtained by

$$\boldsymbol{y}_H = \boldsymbol{p}_H - \boldsymbol{p}_L, \tag{1}$$

where $\boldsymbol{y}_H$ is the high-frequency details of the HR patch $\boldsymbol{p}_H$, and $\boldsymbol{p}_L$ is the corresponding patch from $\boldsymbol{I}_B$ and stands for the low-frequency components.

In order to extract local LR features corresponding to their high-frequency content, similar to [20], we use the first- and second- order gradients in the horizontal and vertical directions from the Bicubic interpolation result $\boldsymbol{I}_B^i$ by

$$\boldsymbol{G}_j^i = \boldsymbol{f}_j * \boldsymbol{I}_B^i, \quad j = 1, ..., 4, \tag{2}$$

where $\boldsymbol{G}_j^i$ is the filtered image. $\boldsymbol{f}_1$ and $\boldsymbol{f}_2$ are gradient high-pass filters in horizontal and vertical directions. $\boldsymbol{f}_3$ and $\boldsymbol{f}_4$ are Laplacian high-pass filters in horizontal and vertical directions. The operator $*$ stands for convolution.

Then we obtain the original high-dimensional LR features $\{\boldsymbol{g}_L^i\}_{i=1}^{N_1}$ from $\{\boldsymbol{p}_L^i\}_{i=1}^{N_1}$ by

$$\boldsymbol{g}_L^i = \left[\boldsymbol{g}_1^i;\ \boldsymbol{g}_2^i;\ \boldsymbol{g}_3^i;\ \boldsymbol{g}_4^i\right], \tag{3}$$

where $\boldsymbol{g}_j^i$ comes from the filtered image $\boldsymbol{G}_j^i$ with the same positions.

In order to enhance efficiency and robustness, we compute a PCA transform matrix $\mathbf{P}$ in $\{\boldsymbol{g}_L^i\}_{i=1}^{N_1}$ to reduce the dimensionality of $\boldsymbol{g}_L^i$ while preserving 99.9% of the average energy. The final low-dimensionality LR feature is obtained by

$$\boldsymbol{y}_L^i = \mathbf{P} \cdot \boldsymbol{g}_L^i, \quad i = 1, ..., N_1. \tag{4}$$

Then we divide $\{\boldsymbol{y}_L^i\}_{i=1}^{N_1}$ into K_1 subsets using the K-means clustering algorithm, which leads to K_1 cluster centers $\{\boldsymbol{c}_i\}_{i=1}^{K_1}$ of their dimensionality-reduced features. Fig. 2 shows $K_1 = 1024$ LR cluster centers learned from $N_1 = 135,581$ LR features. Fig. 4 shows the distribution of clustered features and the center preserving ratio, which was defined as

$$r = \frac{\sum_{k=1}^{K'} |\Omega_k|}{N_1}, \tag{5}$$

where K' denotes the number of LR cluster centers or projection matrices used in the SR phase. Because we sort the LR cluster centers by the number of clustered features in a descending order, K' also means K' clusters which have the most LR features. Ω_k stands for the specified index set of the kth clustered subset from $\{\boldsymbol{y}_L^i\}_{i=1}^{N_1}$, $|\cdot|$ denotes the cardinality

Fig. 3. 10 test images from left to right: **bike**, **bird**, **birdpeak**, **foreman**, **hat**, **penguin**, **pepper**, **skiing**, **text**, and **woman**.

of a set. The center preserving ratio is a useful statistical prior learned from clustering, from which we know which cluster centers are more populous and make our proposed SR algorithm faster with a relatively small number of populous cluster centers.

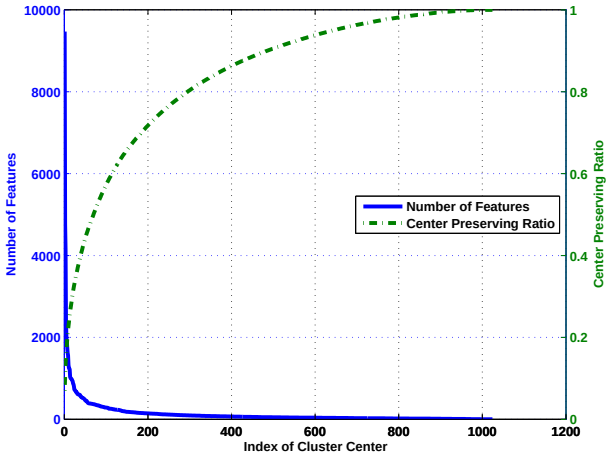


Fig. 4. Histogram of the clustered LR features from LR feature space containing about 135,581 features. Center preserving ratio shows a statistical prior that most of the training LR features fall into a relatively small number of popular clusters, with which we can further speed up our proposed SR method.

According to [32], simple functions mapping from LR feature space to HR feature space can be learned for fast SR when abundant and effective exemplars are available in the training phase. This inspires us to collect a larger training samples $\{\boldsymbol{p}_L^i\}_{i=1}^{N_2}$ and $\{\boldsymbol{p}_H^i\}_{i=1}^{N_2}$, from which we extract co-occurrence LR-HR feature spaces $\{\boldsymbol{y}_L^i\}_{i=1}^{N_2}$ and $\{\boldsymbol{y}_H^i\}_{i=1}^{N_2}$. In order to preserve the local geometry and group the local manifold around the LR cluster centers $\{\boldsymbol{c}_i\}_{i=1}^{K_1}$, we also employ K-means algorithm to split the LR feature spaces into K_2 subspaces $\{\boldsymbol{Y}_L^i\}_{i=1}^{K_2}$ and cluster centers $\left\{\boldsymbol{c}_i^{'}\right\}_{i=1}^{K_2}$, where $\boldsymbol{Y}_L^i = \left\{\boldsymbol{y}_L^j\right\}_{j\in\Omega_i}$ is the ith subset of $\{\boldsymbol{y}_L^i\}_{i=1}^{N_2}$.

B. Grouping Neighbor Features

Having prepared LR and HR feature cluster centers and subspaces, we have to group LR-HR neighbor features $\mathbf{N}_L$ and $\mathbf{N}_H$ for each cluster center in $\{\boldsymbol{c}_i\}_{i=1}^{K_1}$.

For each LR cluster center $\boldsymbol{c}_i$, we first find d nearest centers $\left\{\boldsymbol{c}_j^{'}\right\}_{j=k_1}^{k_d}$ in $\left\{\boldsymbol{c}_j^{'}\right\}_{j=1}^{K_2}$, and obtain the LR-HR feature searching subspaces $\boldsymbol{Y}_{Lsub}^i = \bigcup_{j=k_1}^{k_d} \boldsymbol{Y}_L^j$ corresponding to $\left\{\boldsymbol{c}_j^{'}\right\}_{j=k_1}^{k_d}$. Then we group neighbor feature subset $\mathbf{N}_{L,i}$ from $\boldsymbol{Y}_{Lsub}^i$ by

$$\boldsymbol{N}_{L,i} = \begin{cases} \boldsymbol{Y}_{Lsub}^i, & \text{if } \left|\boldsymbol{Y}_{Lsub}^i\right| \le max \\ max \text{ NN in } \boldsymbol{Y}_{Lsub}^i, & \text{if } \left|\boldsymbol{Y}_{Lsub}^i\right| > max \end{cases} \tag{6}$$

where max is a threshold to control the scale of $\mathbf{N}_{L,i}$ for effective computation, NN stands for nearest neighborhood. Correspondingly, we group HR neighbor feature subset $\mathbf{N}_{H,i}$ using the corresponding HR features in $\{\boldsymbol{y}_H^i\}_{i=1}^{N_2}$.

The reason why we use the rule presented in (6) is that we want to utilize the local geometry of the LR feature space. If we do not set a threshold to control the scale of $\mathbf{N}_{L,i}$, for some LR cluster centers whose element number is very small, the characteristics of these cluster centers and their corresponding LR features will be neglected by those cluster centers who have a large number of LR features.

Furthermore, a small d would help the searching procedure to be time-saving and also take advantage of the local geometry of $\{\boldsymbol{y}_L^i\}_{i=1}^{N_2}$. An appropriate value of d would also advance the performance of our proposed SR method, more details about selecting the optimal value of d will be stated in Section III-C.

C. Computing Projection Matrices

Once the LR-HR neighbor feature subsets $\{\boldsymbol{N}_{L,i}, \boldsymbol{N}_{H,i}\}_{i=1}^{K_1}$ are estimated, multiple projection matrices are computed via collaborative representation [47] to map LR feature spaces to HR feature spaces.

Different from most coding-based SR methods that solve the sparse coefficients by combining the reconstruction error with l_0-norm [20] or l_1-norm [30] regularization term, we use collaborative representation [47] with l_2-norm regularized least squares regression to obtain the projection matrix for each LR-HR neighbor feature subsets $\{\boldsymbol{N}_{L,i}, \boldsymbol{N}_{H,i}\}_{i=1}^{K_1}$. The problem becomes

$$\boldsymbol{x}_i = arg\ \min_{\boldsymbol{x}_i} \left\|\boldsymbol{y}_L^i - \boldsymbol{N}_{L,j} \cdot \boldsymbol{x}_i\right\|_2^2 + \lambda \left\|\boldsymbol{x}_i\right\|_2^2, \tag{7}$$

where $\boldsymbol{y}_L^i \in \mathbb{R}^{n\times 1}$ is an arbitrary LR feature, $\boldsymbol{N}_{L,j}$ is the jth LR neighbor subset whose corresponding cluster center $\boldsymbol{c}_j$ is closet to $\boldsymbol{y}_L^i$, $\boldsymbol{x}_i$ is the coefficient vector of $\boldsymbol{y}_L^i$ over $\boldsymbol{N}_{L,j}$, and $\lambda > 0$ is a weighting parameter allowing us to alleviate the ill-posed problem. The problem above becomes to have a closed-form solution given by

$$\boldsymbol{x}_i = \left(\boldsymbol{N}_{L,j}^T \boldsymbol{N}_{L,j} + \lambda \boldsymbol{I}\right)^{-1} \boldsymbol{N}_{L,j}^T \boldsymbol{y}_L^i. \tag{8}$$

Assuming that $\boldsymbol{y}_L^i$ and $\boldsymbol{y}_H^i$ share the same coefficient vector $\boldsymbol{x}_i$ over $\boldsymbol{N}_{L,j}$ and $\boldsymbol{N}_{H,j}$ respectively, the desired HR feature $\boldsymbol{y}_H^i \in \mathbb{R}^{m\times 1}$ can be obtained by

$$\boldsymbol{y}_H^i = \boldsymbol{N}_{H,j} \cdot \boldsymbol{x}_i \tag{9}$$

Algorithm 1 The Learning Stage of CCR-Based SR Algorithm

Input: N_I LR and HR training images $\left\{\boldsymbol{I}_L^i, \boldsymbol{I}_H^i\right\}_{i=1}^{N_I}$, scaling factor s.
Prepocessing: Upscale $\left\{\boldsymbol{I}_L^i\right\}_{i=1}^{N_I}$ to $\left\{\boldsymbol{I}_B^i\right\}_{i=1}^{N_I}$ using Bicubic interpolation with a scaling factor s;
Obtain filtered images $\left\{\boldsymbol{G}_j^i\right\}_{i=1}^{N_I}, j = 1, ..., 4$, from $\left\{\boldsymbol{I}_B^i\right\}_{i=1}^{N_I}$ by (2).
Feature Extraction: Extract a small LR feature set $\left\{\boldsymbol{g}_L^i\right\}_{i=1}^{N_1}$ from $\left\{\boldsymbol{G}_j^i\right\}_{i=1}^{N_I}, j = 1, ..., 4$ by (3);
Compute the PCA transform matrix $\mathbf{P}$ from $\left\{\boldsymbol{g}_L^i\right\}_{i=1}^{N_1}$;
Get dimensionality-reduced LR feature set $\left\{\boldsymbol{y}_L^i\right\}_{i=1}^{N_1}$, from which we obtain K_1 LR cluster centers $\{\boldsymbol{c}_i\}_{i=1}^{N_1}$;
Collect a larger LR-HR feature sets $\left\{\boldsymbol{y}_L^i, \boldsymbol{y}_H^i\right\}_{i=1}^{N_2}$, and cluster $\left\{\boldsymbol{y}_L^i\right\}_{i=1}^{N_2}$ into K_2 subspaces $\left\{\boldsymbol{c}_i^{'}, \boldsymbol{Y}_L^i\right\}_{i=1}^{K_2}$.
for $i = 1$; $i \leq K_1$; $i++$ **do**
    Find d nearest LR cluster centers of $\boldsymbol{c}_i$ from $\left\{\boldsymbol{c}_i^{'}\right\}_{i=1}^{K_2}$;
    Obtain LR feature searching spaces $\boldsymbol{Y}_{Lsub}^i = \bigcup_{j=k_1}^{k_d} \boldsymbol{Y}_L^j$;
    Group LR neighbor feature subset $\boldsymbol{N}_{L,i}$ from $\boldsymbol{Y}_{Lsub}^i$ by (6);
    Group HR neighbor feature subset $\boldsymbol{N}_{H,i}$ with the corrsponding HR features in $\left\{\boldsymbol{y}_H^i\right\}_{i=1}^{N_2}$.
    Compute projection matrix $\boldsymbol{F}_i$ by (11).
end for
Output: PCA transform matrix $\mathbf{P}$, relationship projecting models $\{\boldsymbol{c}_i, \boldsymbol{F}_i\}_{i=1}^{K_1}$.

By combing (8) and (9), we can obtain:

$$\boldsymbol{y}_H^i = \boldsymbol{N}_{H,j}\left(\boldsymbol{N}_{L,j}^T\boldsymbol{N}_{L,j} + \lambda\boldsymbol{I}\right)^{-1}\boldsymbol{N}_{L,j}^T\boldsymbol{y}_L^i, \tag{10}$$

where we further compute the corresponding projection matrix $\boldsymbol{F}_j \in \mathbb{R}^{m\times n}$ offline by

$$\boldsymbol{F}_j = \boldsymbol{N}_{H,j}\left(\boldsymbol{N}_{L,j}^T\boldsymbol{N}_{L,j} + \lambda\boldsymbol{I}\right)^{-1}\boldsymbol{N}_{L,j}^T, \quad j = 1, ..., K_1, \tag{11}$$

which converts a given LR feature vector into a desired HR version by utilizing matrix multiplication. The learning stage of our CCR-based SR algorithm is summarized in **Algorithm** 1.

D. Fast CCR-based SR

As we have obtained a set of relationship projecting models $\{\boldsymbol{c}_k, \boldsymbol{F}_k\}_{k=1}^{K_1}$, the SR problem can then be efficiently solved in a patch-wise manner. For an LR input image $\boldsymbol{I}_L$, we first upscale it to the size of the desired HR image $\boldsymbol{I}_H$ by Bicubic interpolation. A set of LR features are then extracted from the interpolated result $\boldsymbol{I}_B$. For each LR feature $\boldsymbol{y}_L^i$, we compute the Euclidean distances to K_1 LR cluster centers $\{\boldsymbol{c}_k\}_{k=1}^{K_1}$ and select the most matched projection matrix $\boldsymbol{F}_j$ whose corresponding cluster center $\boldsymbol{c}_j$ is nearest to $\boldsymbol{y}_L^i$. Then we use $\boldsymbol{F}_j$ to project $\boldsymbol{y}_L^i$ to its desired $\boldsymbol{y}_H^i$, which is added with $\boldsymbol{p}_L^i$ to form the HR patch $\boldsymbol{p}_H^i$. Finally, we combine these HR patches to a whole HR image $\boldsymbol{I}_H$ by averaging the intensity values over the overlapping regions.

Algorithm 2 Proposed CCR-Based SR Algorithm

Input: LR image $\boldsymbol{I}_L$, scaling factor s, PCA transform matrix $\mathbf{P}$, the number of used projection matrices $K^{'}$, and the projecting models $\{\boldsymbol{c}_k, \boldsymbol{F}_k\}_{k=1}^{K^{'}}$.
Initialization:
Upscale $\boldsymbol{I}_L$ to $\boldsymbol{I}_B$ using Bicubic interpolation with a scaling factor s;
Set the desired HR image $\boldsymbol{I}_H = \mathbf{0}$.
Feature Extraction:
Filter $\boldsymbol{I}_B$ to obtain four filtered images $\boldsymbol{G}_j,\ (j = 1, ..., 4)$ with four high-pass filters $\boldsymbol{f}_j,\ (j = 1, ..., 4)$ by (2);
Collect LR patches $\left\{\boldsymbol{p}_L^i\right\}_{i=1}^{N}$ from $\boldsymbol{I}_B$, and the corresponding high-dimensional LR features $\left\{\boldsymbol{g}_L^i\right\}_{i=1}^{N}$ from $\boldsymbol{G}_j,\ (j = 1, ..., 4)$ by (3);
Obtain the low-dimensional LR features $\left\{\boldsymbol{y}_L^i\right\}_{i=1}^{N}$ by reducing the dimensionality of $\left\{\boldsymbol{g}_L^i\right\}_{i=1}^{N}$ with PCA transform matrix $\mathbf{P}$ by (4).
for $i = 1$; $i \leq N$; $i++$ **do**
    Find the nearest LR cluster center of $\boldsymbol{y}_L^i$ from $\{\boldsymbol{c}_k\}_{k=1}^{K^{'}}$ by
$$j = arg\min_{j=1,..,K^{'}} \left\|\boldsymbol{y}_L^i - \boldsymbol{c}_j\right\|_2;$$
    Obtain the corresponding HR feature as $\boldsymbol{y}_H^i = \boldsymbol{F}_j \cdot \boldsymbol{y}_L^i$;
    Recovery the corresponding HR patch $\boldsymbol{p}_H^i$ by
$$\boldsymbol{p}_H^i = \boldsymbol{y}_H^i + \boldsymbol{p}_L^i;$$
    Add $\boldsymbol{p}_H^i$ to the corresponding pixels in $\boldsymbol{I}_H$.
end for
Average $\boldsymbol{I}_H$ in the overlap area to create the resulting image.
Output: The desired HR image $\boldsymbol{I}_H$.

Moreover, our CCR-based SR algorithm can further speed up by utilizing a small number of projection matrices. As we can see from Fig. 4 that most of the LR features belong to a very small number of LR cluster centers. This is a useful statistical prior for us to design a faster SR algorithm while maintaining well performance both quantitatively and visually. More details about the running time are given in Section III-D and the CCR-based SR algorithm is presented in **Algorithm** 2.

E. Discussions with Related Works

Our work is different from prior learning-based SR methods such as SF [32], MLM [34], ANR [22], and A+ [31].

CCR is different from SF [32] in the following aspects: First, CCR uses similar LR features to Zeyde's methods in [20], while SF used different LR features extracted by subtracting the patch mean to represent the high-frequency signals. As a result, the histogram of the clustered LR features in SF is also different from ours shown in Fig. 4 . Second, for each LR subspace, SF learned mapping function in just one set of corresponding cluster samples. CCR would learn each mapping function in more than one set of cluster samples. Third, as the mapping functions were bounded to the cluster samples in SF, which needed a large set of training data

(more than 91 images [30] commonly used in SR) to obtain considerable performance.

CCR is different from MLM [34] in the following aspects: First, in each LR-HR feature subspaces, MLM learned LR and HR subdictionaries, which were then used to compute the mapping matrix. Second, as each mapping matrix was computed in one pair of subdictionaries bounded to one set of corresponding cluster samples, a larger set of training images (e.g., 300 images in BSD300 [48]) were used in MLM. Third, MLM used a fast non-local means (NLM) as a post-processing step to achieve better performance, while CCR does not need such post-process.

CCR is different from ANR [22] in the following aspects: First, ANR needs to learn a pair of LR-HR dictionaries in the LR-HR feature spaces, which fails to learn the geometry property among the dictionary atoms and the statistical priors for each anchored point. While CCR is able to learn these two important properties in the clustering process. Second, when grouping neighbors for each anchored point (For ANR, it is the dictionary atom. For CCR, it is the cluster center), ANR searches its LR/HR neighbor atoms from a constant LR/HR dictionary. Because the dictionary atoms sample the LR/HR feature spaces in a very sparse way and the optimal number of neighbor atoms for each anchored point is also limited, the projection matrices computed in ANR have limited ability to reconstruct most of HR features extracted from the LR input. While CCR searches more neighbor samples from a part of training samples, allowing to represent LR features more faithfully and reconstruct more accurate HR features. Third, ANR did not investigate the importance of each anchored dictionary atom, so in the reconstruction phase ANR had to search the whole anchored atoms which is time consuming, especially for real-time applications. However, by using the statistical property learned from the clustering process, CCR can achieve better SR results with less number of projection matrices.

CCR is different from A+ [31] in the following aspects: First, A+ learned an LR dictionary only in the LR feature space and also failed to learn the geometry property among the dictionary atoms and the statistical priors for each LR dictionary atom. Second, when grouping neighbor features for each anchored dictionary atom, A+ used the whole training samples, which is time consuming and memory wasting. However, we cluster the whole training samples into K_2 (e.g., $K_2 = 1024$) subspaces and find d (e.g., $d = 6$) nearest centers, the neighbor feature subset for each anchored cluster center is grouped from the d clusters. By using the rule detailed in Section II-B, CCR would save training time and memory utilization, while achieving comparable results with A+. Third, similar as ANR, A+ also had to use the whole anchored dictionary atoms in the reconstruction phase, while CCR is able to use far less number of anchored cluster centers and projection matrices, saving running time and memory utilization.

Overall, compared with clustering and regression based methods, like SF [32] and MLM [34], which learned the mapping functions bounded in each insular cluster samples, CCR group neighbor feature sets from more than one cluster. When it comes to dictionary learning and regression based methods, like ANR [22] and A+ [31], which group neighbor features for each anchored dictionary atom from either the whole dictionary atoms (for ANR) or the whole training samples (for A+). CCR group neighbor features from a smaller set of training samples compared with that in A+.

III. Experimental Results

In this section, a set of experiments are reported to demonstrate the efficiency and robustness of our proposed CCR-based SR algorithm.

A. Experimental Settings

In our experiments, for a fair comparison, we use the training set [30] which consists of 91 images. These training images are also used in our compared methods [18]–[22], [26]–[28]. To validate the effectiveness and robustness of our CCR-based SR method, we use 10 images shown in Fig. 3 to conduct our SR experiments.

We compare our CCR-based SR algorithm with Bicubic interpolation and eight state-of-the-art learning based SR methods: **NE+LLE**: neighbor embedding + locally linear embedding method [18]. **ScSR**: sparse coding-based SR method using l_1-optimization in Yang *et al.* [19]. **K-SVD**: sparse coding-based SR method using l_0-optimization in Zeyde *et al.* [20]. **NE+NNLS**: neighbor embedding + non-negative least squares method [21]. **ANR**: anchored neighborhood regression method [22]. **BPJDL**: a beta process prior is applied to learn the over-complete dictionaries in a Bayesian SR method [26]. **SPM**: a statistical prediction model based on sparse representation for SR [27]. **SRCNN**: SR using convolutional neural network [28]. **MLM**: learning LR-HR dictionary pairs and mapping fuctions in each cluster [34]. The implementations are all from the publicly available codes provided by the authors[1].

Peak signal-to-noise ratio (PSNR), structural similarity (SSIM) [49], and visual information fidelity (VIF) [50] are employed in our experiments to evaluate the quality of SR results by different methods. All of these evaluation metrics are performed between the luminance channel of the original HR image and the reconstructed one.

The LR images for training and testing are generated from the original HR images down-sampled by Bicubic interpolation and the down-sampled image is then up-sampled using Bicubic interpolation with the same scaling factor to extract LR features. Since human visual system is more sensitive to the luminance component than the chrominance components [51], [52], RGB images are converted to YCbCr color space. The SR process is only performed on the luminance channel (i.e., Y), Bicubic interpolation is used to interpolate the chromatic channels (i.e., Cb and Cr), and the final results are obtained by converting three channels back to the RGB color space. We set scaling factor $s = 3$ and center preserving ratio $r = 1$ throughout the paper except in Section III-D, where we show the running time comparisons with $s \in \{2, 3, 4\}$,

[1]The results of MLM are provided by its authors and the source code of the proposed **CCR** will be available after this paper is published.

TABLE I
SUMMARY OF PSNR(DB), SSIM, AND VIF RESULTS OF 10 TEST IMAGES FOR 3× MAGNIFICATION VIA DIFFERENT METHODS. FOR EACH IMAGE, WE HAVE THREE ROWS: PSNR, SSIM, AND VIF. THE BEST RESULT FOR EACH IMAGE IS **HIGHLIGHTED**.

Images	**Bicubic**	**NE+LLE** [18]	**ScSR** [19]	**K-SVD** [20]	**NE+NNLS** [21]	**ANR** [22]	**BPJDL** [26]	**SPM** [27]	**SRCNN** [28]	**MLM-1** [34]	**MLM-2** [34]	**CCR** $r = 0.8$	**CCR** $r = 1$
bike	22.826	23.901	23.967	23.908	23.810	23.990	24.226	24.446	24.513	24.317	**24.745**	24.488	24.662
	0.7040	0.7655	0.7669	0.7649	0.7597	0.7698	0.7812	0.7880	0.7871	0.7822	**0.8017**	0.7915	0.7957
	0.3103	0.3835	0.3540	0.3815	0.3758	0.3890	0.4044	0.4088	0.3974	0.4052	**0.4276**	0.4136	0.4200
bird	32.627	34.590	34.197	34.587	34.280	34.606	34.879	35.146	34.967	34.632	34.787	34.984	**35.363**
	0.9262	0.9481	0.9399	0.9481	0.9437	0.9491	0.9504	0.9517	0.9498	0.9465	0.9501	0.9515	**0.9533**
	0.5517	0.6599	0.5867	0.6577	0.6413	0.6652	0.6746	0.6683	0.6582	0.6593	0.6695	0.6760	**0.6827**
bird-peak	36.246	37.780	37.260	38.097	37.764	37.784	38.229	38.368	38.552	38.256	38.606	38.682	**38.881**
	0.9387	0.9449	0.9333	0.9466	0.9450	0.9459	0.9461	0.9460	0.9474	0.9438	0.9475	0.9489	**0.9493**
	0.5024	0.5964	0.4159	0.5958	0.5869	0.5996	0.6056	0.5749	0.6036	0.5993	0.6065	0.6206	**0.6246**
fore-man	31.222	33.199	32.101	33.221	33.018	33.229	33.283	33.889	33.409	33.486	34.086	34.136	**34.275**
	0.9059	0.9287	0.9133	0.9293	0.9250	0.9295	0.9305	0.9352	0.9322	0.9269	0.9372	0.9375	**0.9386**
	0.5310	0.6394	0.5327	0.6347	0.6229	0.6452	0.6517	0.6493	0.6397	0.6439	0.6579	0.6675	**0.6706**
hat	29.329	30.432	30.510	30.630	30.383	30.497	30.839	30.989	30.929	30.943	31.265	31.242	**31.339**
	0.8353	0.8621	0.8548	0.8652	0.8598	0.8629	0.8683	0.8685	0.8666	0.8678	0.8750	0.8767	**0.8777**
	0.4019	0.4882	0.4211	0.4930	0.4791	0.4907	0.5034	0.4958	0.4964	0.5115	0.5211	0.5306	**0.5339**
penguin	38.326	40.571	39.127	40.804	40.356	40.631	40.900	41.205	41.219	40.470	40.971	41.090	**41.455**
	0.9680	0.9750	0.9614	0.9761	0.9742	0.9761	0.9752	0.9753	0.9766	0.9714	0.9759	0.9770	**0.9774**
	0.6461	0.7486	0.5443	0.7473	0.7336	0.7540	0.7527	0.7234	0.7491	0.7342	0.7473	0.7603	**0.7652**
pepper	32.481	33.881	33.443	34.117	33.759	33.872	34.173	34.359	34.422	34.168	34.454	34.647	**34.716**
	0.8721	0.8863	0.8692	0.8881	0.8839	0.8869	0.8885	0.8881	0.8894	0.8845	0.8905	0.8923	**0.8927**
	0.4930	0.5897	0.4543	0.5920	0.5762	0.5928	0.6037	0.5876	0.5979	0.5937	0.6065	0.6180	**0.6208**
skiing	29.729	30.679	30.552	30.742	30.693	30.724	30.933	31.390	31.172	30.884	31.211	31.287	**31.391**
	0.8890	0.9050	0.8951	0.9055	0.9039	0.9062	0.9071	0.9098	0.9094	0.9032	0.9094	0.9108	**0.9115**
	0.4604	0.5529	0.4528	0.5448	0.5398	0.5553	0.5550	0.5522	0.5541	0.5420	0.5506	0.5657	**0.5700**
text	28.510	30.243	29.600	30.393	30.190	30.301	30.654	31.053	30.736	30.186	30.608	30.580	**31.024**
	0.9329	0.9473	0.9292	0.9504	0.9466	0.9477	0.9492	0.9537	0.9518	0.9388	0.9520	0.9519	**0.9555**
	0.4437	0.5530	0.4369	0.5527	0.5455	0.5545	0.5733	0.5717	0.5622	0.5392	0.5663	0.5625	**0.5794**
woman	28.569	30.222	29.986	30.370	29.950	30.314	30.727	30.693	30.923	30.395	30.619	30.706	**31.071**
	0.8897	0.9161	0.9043	0.9176	0.9128	0.9167	0.9209	0.9227	0.9237	0.9126	0.9194	0.9235	**0.9263**
	0.8897	0.9161	0.9043	0.9176	0.9128	0.9167	0.9209	0.9227	0.9237	0.5668	0.5841	0.5894	**0.9263**
Aver-age	30.986	32.550	32.074	32.687	32.420	32.595	32.884	33.154	33.084	32.774	33.135	33.184	**33.418**
	0.8862	0.9079	0.8968	0.9092	0.9055	0.9091	0.9117	0.9139	0.9134	0.9078	0.9159	0.9162	**0.9178**
	0.4795	0.5769	0.4680	0.5761	0.5649	0.5807	0.5904	0.5812	0.5830	0.5795	0.5937	0.6004	**0.6068**

and $r \in (0, 1]$ for further speeding up our SR method. In the training phase, $N_1 = 135,581$ LR features are clustered into $K_1 = 1024$ subspaces, $N_2 = 5,000,000$ LR-HR feature pairs are collected from the same training images and clustered into $K_2 = 1024$ subsets in LR feature space. The dimensionality of LR and HR feature vectors is $n = 30$ (applying PCA to reduce dimensionality), and $m = 81$, respectively. More discussions and experiments on the selection of the parameters K_1, max, d, and λ will be conducted in Section III-C. We set $K_1 = 1024$, $max = 2048$, $d = 6$, and $\lambda = 0.03$ as the optimal for image SR in a set of experimental results.

B. Comparison Results

In Table I, we compare the proposed algorithm with the state-of-the-art methods in terms of PSNR, SSIM, and VIF. Our algorithm obtains the best results for most of the test images, achieving best results on average and obvious advantage over the baselines. The average PSNR gain of our CCR ($r = 1$) over the second best method SPM [27] are 0.264dB. It also can be found that the average SSIM and VIF gains of our CCR ($r = 1$) on the second best method MLM [34] is 0.0019 and 0.0131 respectively. These quantitative results are noticeable and demonstrate that our CCR-based SR algorithm not only obtains smaller reconstruction error but also preserves better structural details than the other compared methods. Furthermore, even we use a much lower center preserving ratio (e.g., $r = 0.8$, namely only 295 projection matrices are used in the reconstruction phase), our CCR-based method outperforms other competing methods. This fact tells that the statistical priors learned from clustering are useful and important, obtaining a much smaller number of projection matrices and saving the utilization of memory. More details about center preserving ratio (r) are discussed in Section III-D.

To further demonstrate the effectiveness of our proposed method, we compare our visual results with other state-of-the-art methods with upscaling factor $s = 3$ in Fig. 5, 6, 7, and 8. We select these images based on the variety of the scenes including animal (see Fig. 6), plant (see Fig. 8), instrument (see Fig. 5), architecture, and man (see Fig. 7). We can see that the Bicubic interpolation produces the worst results with seriously blurring effects along the edges and over-smoothed textures. NE+LLE [18] can alleviate the blurring effects along the edges in Fig. 5(c) and Fig. 7(c) by partially reconstructing the high frequency components of the HR images. But it also produces ringing effects along edges in Fig. 8(c) by introducing inaccurate neighbors in SR. ScSR [19] learns HR and LR dictionaries by taking advantage of l_1-norm regularization term and also generates ringing effects and blurred results in Fig. 5(d) and Fig. 8(d). The reason is mainly that the dictionaries learned by ScSR [19] are too compact and fail to capture the details from the whole diverse and some infrequent training patches. K-SVD [20] has a relatively fast implementation by using a PCA dimensionality-reduction strategy and l_0-norm regularization term. But, as shown in Fig. 5(e) and Fig. 6(e), it generates some noticeable blurred details along the dominant edges. In Fig. 7(f) and Fig. 8(f), NE+NNLS [21] produces both jaggy edge artifacts and annoying textural details. Because of using only a small number of dictionary atoms, ANR [22] would achieve very fast SR while generating some unpleasing artifacts in Fig. 7(g)

and blurred edges in Fig. 8(g). BPJDL [26] suppresses ringing effects significantly, however, it also produces jaggy artifacts in Fig. 5(h) and smoothing details in Fig. 6(h). From the visual results shown in Fig. 5(i) and Fig. 7(i), SPM [27] shows its strong performance on deblurring along the main edges. While it generates over-sharped edges and obvious ringing effects, and also fails to recover detailed textures in Fig. 8(i). Although SRCNN [28] obtains relatively high values in terms of PSNR, SSIM, and VIF as presented in Table I, it generates results possessing serious jaggy artifacts along the dominant edges in Fig. 5(j) and Fig. 7(j), which helps to explain why SRCNN achieves relatively high values of PSNR on 10 test images while failing to obtain similar advantages in terms of SSIM and VIF. Because MLM-2 [34] obtained better performance than MLM-1 [34] mainly owing to the usage of post-processing. While most of other compared methods here did not use post-process. We only compare visual results with MLM-1 [34]. From Fig. 5(k) and Fig. 7(k), MLM-1 produced ringing artifacts and some unpleasing artifacts along the edges. As can be observed, the CCR-based SR method produces sharper edges without obvious ringing effects (e.g. Fig. 5(l) and Fig. 7(l)) , less blurring effects (e.g. Fig. 6(l)), and finer textures with more details (e.g. Fig. 8(l)).

C. Investigation of the Parameters in CCR

In the proposed CCR-based SR algorithm, there are four critical parameters: the number of LR cluster centers K_1, the scale controlling threshold max, number of nearest centers d, and the weighting parameter λ. In order to observe how each parameter affects the performance of our proposed method and effectively select the parameter set $\{K_1, max, d, \lambda\}$, we conduct extensive SR experiments on 30 images in terms of average values of PSNR, SSIM, and VIF. Among the four parameters, we found K_1, max, and d play more important roles in our method than λ, which can be set to be a small positive constant when we select other parameters. We also use the prior that the larger K_1 and max, the better the results will often be.

Firstly, we fix $\{K_1, max, \lambda\} = \{1024, 2048, 0.1\}$, and d ranges from 1 to 50. In Fig. 9(a-c), when the parameter value of d is smaller than 3, PSNR, SSIM, and VIF values are all low. Although the PSNR value becomes to be stable as d becomes larger, the corresponding SSIM and VIF values reduce obviously. Because the parameter d takes advantage of the local geometry of the data, which affects the structural details in the reconstructed results. Thus, we choose 6 as the optimal value of the parameter d.

Secondly, we fix $\{K_1, max, d\} = \{1024, 2048, 6\}$, and λ ranges from 0.01 to 0.2. In Fig. 9(d-f), when the parameter λ becomes larger, the PSNR value increases a little and then reduce slightly and the values of SSIM and VIF would present a monotonic reduction slightly. Because in (7), the value of parameter λ affects the estimation accuracy of $\boldsymbol{x}_i$. When $\lambda \to \infty$, then $\boldsymbol{x}_i \to \mathbf{0}$, and the estimation error of $\boldsymbol{x}_i$ becomes larger. Thus we choose 0.03 as the optimal value of the parameter λ.

Thirdly, we fix $\{max, d, \lambda\} = \{2048, 6, 0.03\}$, and K_1 ranges from 32 to 4096. In Fig. 9(g-i), when the parameter K_1 becomes larger, the values of PSNR, SSIM, and VIF would increase. This is reasonable, because the LR feature space is divided into more subspaces, for an input LR feature, it is projected to HR feature with a more correlative LR cluster center and projection matrix pair. However, a larger value of the parameter K_1 would also induce longer training and SR time and more memory utilization. By taking a tradeoff between running time and reconstruction quality and also making a fair comparison with other learning-based SR methods, we set $K_1 = 1024$ throughout the paper.

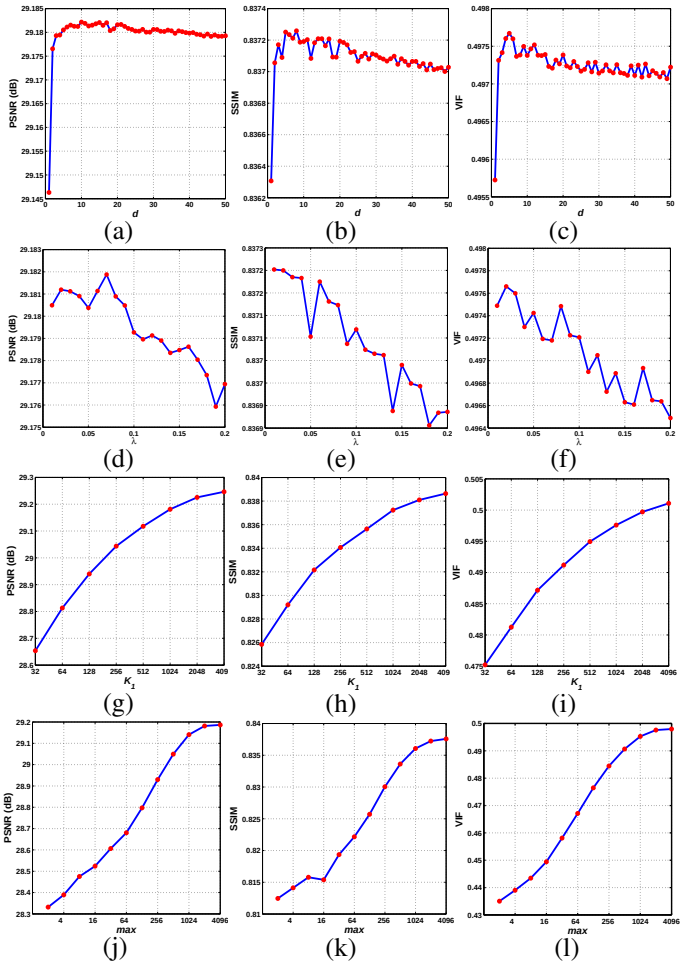


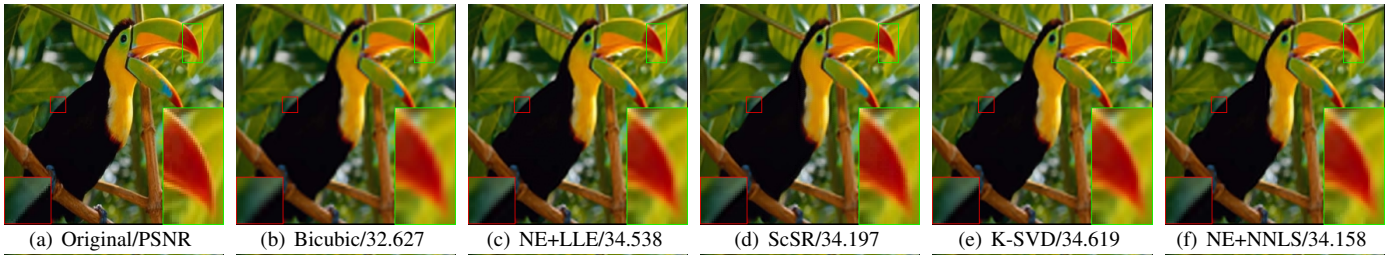
Fig. 9. Average performance of CCR on PSNR, SSIM, and VIF values versus different values of d (the number of nearest centers), λ (a weighting parameter), K_1 (the number of LR cluster centers), and max (a scale controlling threshold) on 30 images.

Finally, we fix $\{K_1, d, \lambda\} = \{1024, 6, 0.03\}$, and max ranges from 2 to 4096. In Fig. 9(j-l), when the parameter max becomes larger in the range from 2 to 2048, the values of PSNR, SSIM, and VIF increase obviously. Because in (7), when the LR neighbor feature subset $\boldsymbol{N}_L^j$ consists of more LR features similar to the input LR feature $\boldsymbol{y}_L^i$, the reconstruction error of which would be smaller. Another important reason is that there are many LR feature clusters, the number of whose elements is to be very small. If we do not set a rule like (6), some LR neighbor feature subsets $\boldsymbol{N}_L^j$ and its corresponding LR feature cluster center $\boldsymbol{c}_j$ will not be well matched, because most of elements in $\boldsymbol{N}_L^j$ may correspond to other LR feature cluster centers. When the value of the parameter max becomes

(a) Original/PSNR (b) Bicubic/22.826 (c) NE+LLE/23.890 (d) ScSR/23.967 (e) K-SVD/23.895 (f) NE+NNLS/23.759

(g) ANR/23.978 (h) BPJDL/24.226 (i) SPM/24.446 (j) SRCNN/24.513 (k) MLM-1/24.317 (l) CCR/**24.662**

Fig. 5. Visual comparisons with different SR results on **bike** by using different methods (scaling factor $s = 3$). (Zoom in for better view.)



(a) Original/PSNR (b) Bicubic/32.627 (c) NE+LLE/34.538 (d) ScSR/34.197 (e) K-SVD/34.619 (f) NE+NNLS/34.158

(g) ANR/34.633 (h) BPJDL/34.879 (i) SPM/35.146 (j) SRCNN/34.967 (k) MLM-1/34.632 (l) CCR/**35.359**

Fig. 6. Visual comparisons with different SR results on **bird** by using different methods (scaling factor $s = 3$). (Zoom in for better view.)

to be larger than 2048, the values of PSNR, SSIM, and VIF seem to be stable, and the training time would increase. Thus we set the value of the parameter max to be 2048 throughout this paper.

D. Running Time

Table II shows the running time comparisons of numerous state-of-the-art methods, along with the reconstruction quality on 10 images in terms of PSNR, SSIM, and VIF with different scaling factors ($\times 2, \times 3, \times 4$). We profile the running time of all the SR methods in a MATLAB 2012a environment using the same machine (3.20 GHz Core(TM) i5-3470 with 16 GB RAM). According to Table II, our CCR ranks the second place with a very small gap between ANR, although the number of projection matrices in CCR and ANR is set to be 1024. This is because in the SR stage, the distance between the input LR feature and the dictionary atoms is measured by inner product in ANR, while in CCR the distance between the input LR feature and cluster centers is measured by Euclidean distance. In our experiments, we find that when the number of projection matrices is smaller than 295, CCR is faster than ANR, otherwise, CCR is slower than ANR slightly. This is mainly because MATLAB is good at vector operation. Although our method requires a little more running time than ANR, a large average PSNR, SSIM, and VIF gains of our CCR ($r = 1$) over ANR can be up to 0.823 dB, 0.0087, and 0.0261. We can also learn from Table II that when the scaling factor becomes larger, the methods, like NE+LLE, K-SVD, NE+NNLS, ANR, and CCR, conducting SR in a patch-wise way would also run faster, while SRCNN remains almost unchanged speed and SPM even increases the SR running time.

Moreover, since we have learned the statistical prior in the clustering phase that most of the LR features belong to a relatively small number of LR feature cluster centers, we can use a small number of projection matrices in our SR phase to further speed up our CCR. When we focus on the running time results for a scaling factor of 3, center preserving ratio $r = 0.7$, our CCR reaches a speedup of $51\times$ when compared to ScSR [19], $798\times$ when compared to BPJDL [26], $39\times$ when compared to SPM [27], and $8\times$ when compared to SRCNN [28].

We also investigate how the center preserving ratio (r) or the number of projection matrices (K') used in the SR phase would affect the performance of our CCR on the 10 images in terms of the average values of PSNR, SSIM, VIF, and running

(a) Original/PSNR (b) Bicubic/31.222 (c) NE+LLE/33.172 (d) ScSR/32.101 (e) K-SVD/33.249 (f) NE+NNLS/32.889

(g) ANR/33.165 (h) BPJDL/33.283 (i) SPM/33.889 (j) SRCNN/33.409 (k) MLM-1/33.486 (l) CCR/**34.278**

Fig. 7. Visual comparisons with different SR results on **foreman** by using different methods (scaling factor $s = 3$). (Zoom in for better view.)

(a) Original/PSNR (b) Bicubic/32.481 (c) NE+LLE/33.887 (d) ScSR/33.443 (e) K-SVD/34.118 (f) NE+NNLS/33.730

(g) ANR/33.896 (h) BPJDL/34.173 (i) SPM/34.359 (j) SRCNN/34.422 (k) MLM-1/33.730 (l) CCR/**34.168**

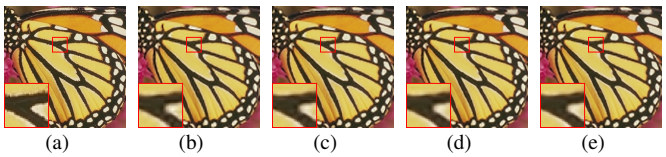
Fig. 8. Visual comparisons with different SR results on **pepper** by using different methods (scaling factor $s = 3$). (Zoom in for better view.)

TABLE II
RUNNING TIME (SECONDS) COMPARISONS OF DIFFERENT METHODS FOR 3 DIFFERENT SCALING FACTORS ($\times 2$, $\times 3$, $\times 4$) ON 10 IMAGES. THE BEST RESULT FOR EACH SCALE IS **HIGHLIGHTED**.

Scale	**NE+LLE** [18]	**ScSR** [19]	**K-SVD** [20]	**NE+NNLS** [21]	**ANR** [22]	**BPJDL** [26]	**SPM** [27]	**SRCNN** [28]	**MLM-1** [34]	**MLM-2** [34]	**CCR** $r = 0.6$	$r = 0.8$	$r = 1$
$\times 2$	10.70	-	5.42	54.39	1.59	712.99	15.63	6.34	-	-	**1.36**	1.62	2.65
$\times 3$	4.50	41.71	2.39	22.25	0.87	646.46	31.20	6.39	2.86	12.25	**0.76**	0.87	1.33
$\times 4$	3.59	-	1.62	14.01	0.72	-	-	6.42	-	-	**0.61**	0.67	0.96

time. In Table III, as r or K' becomes larger, the values of PSNR, SSIM, and VIF would all increase, which demonstrates that it is reasonable for us to cluster the LR feature space to numerous subspaces. When r = 0.6, the number of projection matrices we use to perform SR is only K' = 114, much smaller than 350 used in MLM [34], while obtaining the best average results in terms of PSNR and SSIM compared with most of other state-of-the-art methods (e.g., NE+NNLS [21] and ANR [22]) in Table I. When r = 0.8003, the number of projection matrices we use to perform SR is only K' = 295, obtaining the best average performance in terms of PSNR, SSIM, and VIF compared with other competing methods in our experiments.

A more intuitionistic and detailed results are shown in Fig. 10, from which we can learn that r = 0.8 is a good choice for SR to obtain high-quality reconstructed results and save running time. In order to illustrate the influence of r in a visualized way, we give a group of experimental results on butterfly image in Fig. 11. Among the four images shown in Fig. 11, the visual quality of the resulted images ($r \geq 0.8$) remains almost unchanged. Thus we can use a smaller r or K' in the SR phase when the speed of SR is considered to be the most important aspect.

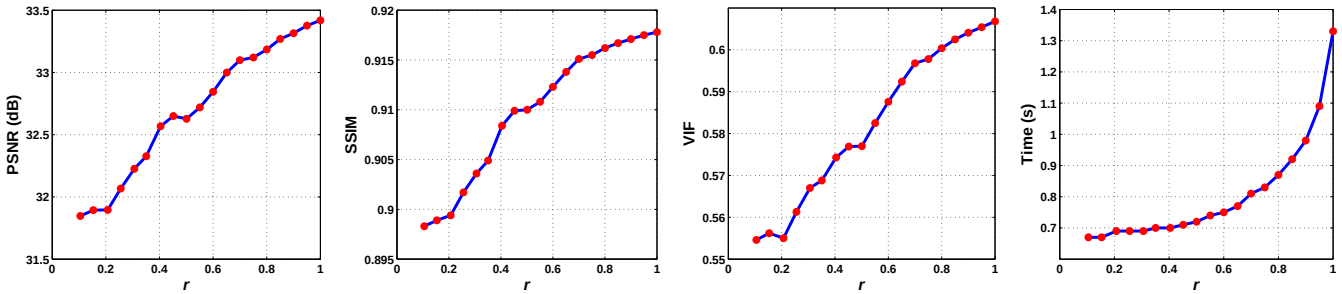


(a) (b) (c) (d) (e)

Fig. 11. Visual quality of the **butterfly** image versus different values of center preserving ratio r (scaling factor $s = 3$). (a) Original image. (b) $r = 0.6$, PSNR: 26.112, SSIM: 0.8844, VIF: 0.4797. (c) $r = 0.8003$, PSNR: 26.760, SSIM: 0.8993, VIF: 0.5068. (d) $r = 0.9001$, PSNR: 27.086, SSIM: 0.9044, VIF: 0.5186. (e) $r = 1$, PSNR: 27.272, SSIM: 0.9077, VIF: 0.5270.

TABLE III
AVERAGE PERFORMANCE OF CCR ON 10 TEST IMAGES FOR SCALING FACTOR ($\times 3$) WITH DIFFERENT VALUES OF CENTER PRESERVING RATIO r.

r	0.1050	0.2070	0.3062	0.4043	0.5008	0.6000	0.7000	0.8003	0.9001	1.0000
K'	2	8	20	40	70	114	184	295	485	1024
PSNR	31.847	31.896	32.226	32.568	32.627	32.845	33.099	33.184	33.315	33.418
SSIM	0.8983	0.8994	0.9036	0.9084	0.9100	0.9123	0.9151	0.9162	0.9171	0.9178
VIF	0.5546	0.5550	0.5670	0.5743	0.5770	0.5876	0.5968	0.6004	0.6041	0.6068
Time	0.67	0.69	0.69	0.70	0.72	0.75	0.81	0.87	0.98	1.33



(a) (b) (c) (d)

Fig. 10. Average performance of CCR on PSNR, SSIM, and VIF values with different values of r (center preserving ratio) on 10 test images.

IV. CONCLUSION

In this paper, we propose a simple yet fast and novel image super-resolution (SR) algorithm, which belongs to the family of learning-based SR algorithms, using clustering and collaborative representation (CCR). The algorithm employs clustering method and collaborative representation [47] to learn numerous projection matrices from the LR feature spaces to their HR feature spaces. When compared with other state-of-the-art SR methods, our CCR-based algorithm shows the best performance both in terms of objective evaluation metrics and subjective visual results. As for objective evaluation, our algorithm obtains a large gains over other competing methods in PSNR, SSIM, and VIF values and also consumes the least running time. When it comes to visual SR results, our algorithm also turns out to reconstruct results that are more faithful to the original HR images with sharper edges and finer details. Moreover, our approach can further speed up the SR procedure by using a very small number of projection matrices while maintaining high-quality SR results, which would be very useful and adaptive for real-time applications.

REFERENCES

[1] M. V. W. Zibetti and J. Mayer, "A robust and computationally efficient simultaneous super-resolution scheme for image sequences," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 10, pp. 1288–1300, Oct. 2007.

[2] N. A. Dodgson, "Quadratic interpolation for image resampling," *IEEE Trans. Image Process.*, vol. 6, no. 9, pp. 1322–1326, Sep. 1997.

[3] X. Lu, Y. Yuan, and P. Yan, "Image super-resolution via double sparsity regularized manifold learning," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 12, pp. 2022–2033, Dec. 2013.

[4] H. Takeda, S. Farsiu, and P. Milanfar, "Kernel regression for image processing and reconstruction," *IEEE Trans. Image Process.*, vol. 16, no. 2, pp. 349–366, Feb. 2007.

[5] R. Keys, "Cubic convolution interpolation for digital image processing," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 29, no. 6, pp. 1153–1160, Dec. 1981.

[6] X. Li and M. T. Orchard, "New edge-directed interpolation," *IEEE Trans. Image Process.*, vol. 10, no. 10, pp. 1521–1527, Oct. 2001.

[7] L. Zhang and X. Wu, "An edge-guided image interpolation algorithm via directional filtering and data fusion," *IEEE Trans. Image Process.*, vol. 15, no. 8, pp. 2226–2238, Aug. 2006.

[8] M. Li and T. Q. Nguyen, "Markov random field model-based edge-directed image interpolation," *IEEE Trans. Image Process.*, vol. 17, no. 7, pp. 1121–1128, Jul. 2008.

[9] P. Thévenaz, T. Blu, and M. Unser, "Image interpolation and resampling," *Handbook of Medical Imaging, Processing and Analysis*, pp. 393–420, 2000.

[10] S. Baker and T. Kanade, "Limits on super-resolution and how to break them," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 9, pp. 1167–1183, Sep. 2002.

[11] Z. Lin and H.-Y. Shum, "Fundamental limits of reconstruction-based superresolution algorithms under local translation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 1, pp. 83–97, Jan. 2004.

[12] R. Fattal, "Image upsampling via imposed edge statistics," in *ACM Trans. Graph.*, vol. 26, no. 3, Jul. 2007, pp. 95:1–95:8.

[13] A. Chakrabarti, A. Rajagopalan, and R. Chellappa, "Super-resolution of face images using kernel pca-based prior," *IEEE Trans. Multimedia*, vol. 9, no. 4, pp. 888–892, Jun. 2007.

[14] J. Sun, J. Zhu, and M. F. Tappen, "Context-constrained hallucination for image super-resolution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 231–238.

[15] H. Xu, G. Zhai, and X. Yang, "Single image super-resolution with detail enhancement based on local fractal analysis of gradient," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 10, pp. 1740–1754, Oct. 2013.

[16] L. Wang, S. Xiang, G. Meng, H.-Y. Wu, and C. Pan, "Edge-directed single-image super-resolution via adaptive gradient magnitude self-interpolation." *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 8, pp. 1289–1299, Aug. 2013.

[17] K. Zhang, X. Gao, D. Tao, and X. Li, "Single image super-resolution with non-local means and steering kernel regression," *IEEE Trans. Image Process.*, vol. 21, no. 11, pp. 4544–4556, Nov. 2012.

[18] H. Chang, D.-Y. Yeung, and Y. Xiong, "Super-resolution through neighbor embedding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun./Jul. 2004, pp. 1–6.

[19] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Trans. Image Process.*, vol. 19, no. 11, pp. 2861–2873, Nov. 2010.

[20] R. Zeyde, M. Elad, and M. Protter, "On single image scale-up using sparse-representations," in *Proc. 7th Int. Conf. Curves Surf.*, Jun. 2010, pp. 711–730.

[21] M. Bevilacqua, A. Roumy, C. Guillemot, and M. L. Alberi-Morel, "Low-complexity single-image super-resolution based on nonnegative neighbor embedding," in *British Machine Vision Conference*, Sep. 2012.

[22] R. Timofte, V. De, and L. V. Gool, "Anchored neighborhood regression for fast example-based super-resolution," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 1920–1927.

[23] M.-C. Yang and Y.-C. F. Wang, "A self-learning approach to single image super-resolution," *IEEE Trans. Multimedia*, vol. 15, no. 3, pp. 498–508, Apr. 2013.

[24] Z. Xiong, D. Xu, X. Sun, and F. Wu, "Example-based super-resolution with soft information and decision," *IEEE Trans. Multimedia*, vol. 15, no. 6, pp. 1458–1465, Oct. 2013.

[25] L.-W. Kang, C.-C. Hsu, B. Zhuang, C.-W. Lin, and C.-H. Yeh, "Learning-based joint super-resolution and deblocking for a highly compressed image," *IEEE Trans. Multimedia*, vol. 17, no. 7, pp. 921–934, Jul. 2015.

[26] L. He, H. Qi, and R. Zaretzki, "Beta process joint dictionary learning for coupled feature spaces with application to single image super-resolution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 345–352.

[27] T. Peleg and M. Elad, "A statistical prediction model based on sparse representations for single image super-resolution." *IEEE Trans. Image Process.*, vol. 23, no. 6, pp. 2569–2582, Jun. 2014.

[28] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2014, pp. 184–199.

[29] S. Yang, Z. Wang, L. Zhang, and M. Wang, "Dual-geometric neighbor embedding for image super-resolution with sparse tensor," *IEEE Trans. Image Process.*, vol. 23, no. 7, pp. 2793–2803, Jul. 2014.

[30] J. Yang, J. Wright, T. Huang, and Y. Ma, "Image super-resolution as sparse representation of raw image patches," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.

[31] R. Timofte, V. De Smet, and L. Van Gool, "A+: Adjusted anchored neighborhood regression for fast super-resolution," in *Proc. IEEE Asian Conf. Comput. Vis.*, Nov. 2014.

[32] C.-Y. Yang and M.-H. Yang, "Fast direct super-resolution by simple functions," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 561–568.

[33] Z. Zhu, F. Guo, H. Yu, and C. Chen, "Fast single image super-resolution via self-example learning and sparse representation," *IEEE Trans. Multimedia*, vol. 16, no. 8, pp. 2178–2190, Oct. 2014.

[34] K. Zhang, D. Tao, X. Gao, X. Li, and Z. Xiong, "Learning multiple linear mappings for efficient single image super-resolution," *IEEE Trans. Image Process.*, vol. 24, no. 3, pp. 846 – 861, Mar. 2015.

[35] Z. Cui, H. Chang, S. Shan, B. Zhong, and X. Chen, "Deep network cascade for image super-resolution," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2014, pp. 49–64.

[36] J. Zhang, C. Zhao, R. Xiong, S. Ma, and D. Zhao, "Image super-resolution via dual-dictionary learning and sparse representation," in *Proc. IEEE Int. Symposium Circuits and Syst.*, May. 2012, pp. 1688–1691.

[37] Y. Zhang, Y. Zhang, and Q. Dai, "Single depth image super resolution via a dual sparsity model," in *Proc. IEEE Int. Conf. Multimedia & Expo Workshops*, Jun./Jul. 2015, pp. 1–6.

[38] Y. Zhang, Y. Zhang, J. Zhang, H. Wang, and Q. Dai, "Single image super-resolution via iterative collaborative representation," in *Advances in Multimedia Information Processing–PCM 2015*. Springer, Sep. 2015, pp. 63–73.

[39] Y. Zhang, Y. Zhang, J. Zhang, H. Wang, X. Wang, and Q. Dai, "Adaptive local nonparametric regression for fast single image super-resolution," in *Proc. IEEE Int. Conf. Visual Commun. Image Process.*, Dec. 2015.

[40] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[41] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Inf. Theory*, vol. 53, no. 12, pp. 4655–4666, Dec. 2007.

[42] J. Darbon, A. Cunha, T. F. Chan, S. Osher, and G. J. Jensen, "Fast nonlocal filtering applied to electron cryomicroscopy," in *Proc. 5th IEEE Symp. Biomed. Imag.*, May 2008, pp. 1331–1334.

[43] D. Glasner, S. Bagon, and M. Irani, "Super-resolution from a single image," in *Proc. IEEE Int. Conf. Comput. Vis.*, Sep./Oct. 2009, pp. 349–356.

[44] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, Dec. 1989.

[45] R. Timofte and L. Van Gool, "Adaptive and weighted collaborative representations for image classification," *Pattern Recognition Letters*, vol. 43, pp. 127–135, Jul. 2014.

[46] M. Irani and S. Peleg, "Motion analysis for image enhancement: Resolution, occlusion, and transparency," *J. Vis. Commun. Image Represent.*, vol. 4, no. 4, pp. 324–335, 1993.

[47] L. Zhang, M. Yang, and X. Feng, "Sparse representation or collaborative representation: Which helps face recognition?" in *Proc. IEEE Int. Conf. Comput. Vis.*, Nov. 2011, pp. 471–478.

[48] 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. IEEE Int. Conf. Comput. Vis.*, Jul. 2001, pp. 416–423.

[49] 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.

[50] H. R. Sheikh and A. C. Bovik, "Image information and visual quality," *IEEE Trans. Image Process.*, vol. 15, no. 2, pp. 430–444, Feb. 2006.

[51] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael, "Learning low-level vision," *Int. J. Comput. Vis.*, vol. 40, no. 1, pp. 25–47, Jun. 2000.

[52] Y. Tang, P. Yan, Y. Yuan, and X. Li, "Single-image super-resolution via local learning," *Int. J. Mach. Learn. Cybern.*, vol. 2, no. 1, pp. 15–23, Mar. 2011.

Yongbing Zhang received the B.A. degree in English and the M.S. and Ph.D degrees in computer science from the Harbin Institute of Technology, Harbin, China, in 2004, 2006, and 2010, respectively. He joined Graduate School at Shenzhen, Tsinghua University, Shenzhen, China in 2010, where he is currently an associate Research Fellow. He was the receipt of the Best Student Paper Award at IEEE International Conference on Visual Communication and Image Processing in 2015. His current research interests include video processing, image and video coding, video streaming, and transmission.

Yulun Zhang received B.E. degree from School of Electronic Engineering, Xidian University, China, in 2013. He is currently working toward the M.E. degree in Department of Automation, Tsinghua University, China. He was the receipt of the Best Student Paper Award at IEEE International Conference on Visual Communication and Image Processing in 2015. His research interests include sparse representation, super-resolution, and transfer learning.

Jian Zhang (M'14) received B.Sc. degree from Department of Mathematics, Harbin Institute of Technology (HIT), Harbin, China, in 2007, and received M.Eng. and Ph. D degrees from School of Computer Science and Technology, HIT, in 2009 and 2014, respectively. Currently, he is working as a postdoctoral fellow at the Institute of Digital Media (IDM), Peking University (PKU), Beijing, China. His research interests include image/video compression and restoration, compressive sensing, sparse representation, and dictionary learning. He was the recipient of the Best Paper Award and Best Student Paper Award at IEEE International Conference on Visual Communication and Image Processing in 2011 and 2015 respectively.

Qionghai Dai received the M.S. and Ph.D. degrees in computer science and automation from Northeastern University, Shenyang, China, in 1994 and 1996, respectively. He is currently a professor in the Department of Automation and is the Director of the Broadband Networks and Digital Media Laboratory at Tsinghua University, Beijing. He has authored or co-authored over 200 conference and journal papers and two books. His research interests include computational photography and microscopy, computer vison and graphics, intelligent signal processing. He is Associate Editor of JVCI, IEEE TNNLS, and IEEE TIP.