Fast Direct Super-Resolution by Simple Functions

Chih-Yuan Yang and Ming-Hsuan Yang
Electrical Engineering and Computer Science, University of California at Merced
{cyang35,mhyang}@ucmerced.edu

Abstract

The goal of single-image super-resolution is to generate a high-quality high-resolution image based on a given low-resolution input. It is an ill-posed problem which requires exemplars or priors to better reconstruct the missing high-resolution image details. In this paper, we propose to split feature space into numerous subspaces and collect exemplars to learn priors for each subspace, thereby creating effective mapping functions. The use of split input space facilitates both feasibility of using simple functions for super-resolution, and efficiency of generating high-resolution results. High-quality high-resolution images are reconstructed based on the effective learned priors. Experimental results demonstrate that the proposed algorithm generates high-quality super-resolution images efficiently and effectively against state-of-the-art methods.

1. Introduction

Single-image super-resolution (SISR) aims to generate a visually pleasing high-resolution (HR) image from a given low-resolution (LR) input. It is a challenging and ill-posed problem because numerous pixel intensities need to be predicted from limited input data. To alleviate this ill-posed problem, it is imperative for most SISR algorithms to exploit additional information such as exemplar images or statistical priors. Exemplar images contain abundant visual information which can be exploited to enrich the super-resolution (SR) image details [4, 1, 5, 19, 6, 17, 3, 18]. However, numerous challenging factors make it difficult to generate SR images efficiently and robustly. First, there exist fundamental ambiguities between the LR and HR data as significantly different HR image patches may generate very similar LR data as a result of downsampling process. That is, the mapping between HR and LR data is many to one and the reverse process from one single LR image patch alone is inherently ambiguous. Second, the success of this approach hinges on the assumption that a high-fidelity HR patch can be found from the LR one (aside from ambiguity which can be alleviated with statistical priors), thereby entailing a large and adequate dataset at our disposal. Third, the ensuing problem with a large dataset is how to determine similar patches efficiently.

In contrast, the performance stability and low computational cost are the marked advantages of statistical SISR approaches [2, 16, 15, 9, 24, 20]. Since the priors are learned from numerous examples, they are statistically effective to represent the majority of the training data. The computational load of these algorithms is relatively low, as it is not necessary to search exemplars. Although the process of learning statistical priors is time consuming, it can be computed offline and only once for SR applications. However, statistical SISR algorithms are limited by specific image structures modeled by their priors (e.g., edges) and ineffective to reconstruct other details (e.g., textures). In addition, it is not clear what statistical models or features best suit this learning task from a large number of training examples.

In this paper, we propose a divide-and-conquer approach [25, 23] to learn statistical priors directly from exemplar patches using a large number of simple functions. We show that when sufficient amount of data is collected, the ambiguity problem of the source HR patches is alleviated. When the input space of LR source images is divided, the mapping between LR and HR patches of each subspace can be effectively modeled by a linear function. The use of simple functions also facilitates the process to generate high-quality HR images efficiently.

The contributions of this work are summarized as follows. First, we demonstrate a direct single-image super-resolution algorithm can be simple and fast when effective exemplars are available in the training phase. Second, we effectively split the input domain of low-resolution patches based on exemplar images, thereby facilitating learning simple functions for effective mapping. Third, the proposed algorithm generates favorable results with low computational load against existing methods. We demonstrate the merits of the proposed algorithm in terms of image quality and computational load by numerous qualitative and quantitative comparisons with the state-of-the-art methods.

2. Related Work and Problem Context

The SISR problem has been intensively studied in computer vision, image processing, and computer graphics. Classic methods render HR images from LR images through certain mathematical formulations [13, 11] such as bicubic interpolation and back-projection [8]. While these algorithms can be executed efficiently, they are less effective in generating high-quality edges and textures because no information of image structure is used.

Recent methods exploit rich visual information contained in a set of training images or from the input frame. Markov random fields are used in [4, 19] to reduce the ambiguity problem between LR and HR patches by minimizing the difference of overlapping HR patches. To alleviate the dataset problem in terms of texture diversity and richness, user guidance is exploited [19, 6] to prepare precise exemplar images for super resolution. The locally linear embedding algorithm is incorporated to address both ambiguity and dataset adequateness problems by integrating HR exemplar patches with weights computed from their LR exemplar patches [1]. Instead of searching for exemplars from an external dataset, self exemplars are exploited from the input image [5, 3] to find patches difficult found from external training set. In addition to images patches, segment exemplars are proposed [17, 6] for upsampling textures because the consistency of HR textures is better enforced by matching large segments than small independent patches.

Statistical SISR algorithms learn priors from numerous feature vectors to generate a function mapping features from LR images to HR images. A significant advantage of this approach is the low computational complexity as the load of searching exemplars is alleviated. Global distributions of gradients are used [15] to regularize a deconvolution process for generating HR images. Edge-specific priors are developed to model the mapping of edges from LR to HR to reconstruct sharp HR edges [2, 16]. In addition, priors of patch mapping from LR to HR are developed based on dictionaries via sparse representation [24, 21] , support vector regression [12], or kernel ridge regression [9].

Notwithstanding much demonstrated success of the algorithms in the literature, existing methods require computationally expensive processes in either searching exemplars [4, 1, 5] or extracting complex features [12, 16, 17, 6]. In contrast, we present a fast algorithm based on intensity values mapped through simple functions for efficient and effective SISR. Instead of using one or a few mapping functions, we learn a large number of simple functions. We show this divide-and-conquer algorithm is effective and efficient for SISR when the right components are properly integrated.

3. Proposed Algorithm

One motivation of this work is to handle the ambiguity problem efficiently in the test phase. In order to generate HR images efficiently, we do not search for a large set for similar patches at the test phase as the computational load is high [5, 3]. Furthermore, we do not use complicated features which require intensive operations to extract [16, 24, 21]. To handle the ambiguity problem using simple features, we collect a large set of LR patches and their corresponding HR source patches. We divide the input space into a large set of subspaces from which simple functions can be used to model the mapping between LR and HR patches effectively and efficiently. Although the proposed algorithm entails processing a large set of training images, it is only carried out offline in batch mode.

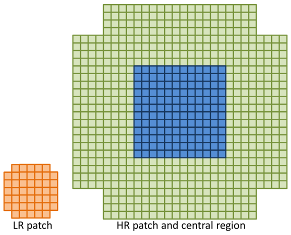


Figure 1. Training LR and HR pairs (four corner pixels are discarded). A set of functions is learned to map a LR patch to a set of pixels at the central (shaded) region of the corresponding HR patch (instead of the entire HR patch).

Given a HR training image I_h, we generate the LR image I_l by

$$I_l = (I_h \otimes G) \downarrow_s, \tag{1}$$

where $\otimes$ is a convolution operator, G is a Gaussian kernel, $\downarrow$ is a downsampling operator and s is the scaling factor. From each I_h and the corresponding I_l image, a large set of corresponding HR and LR patch pairs are generated. Let P_h and P_l be two paired patches. We compute the patch mean of P_l as μ, and extract the features of P_h and P_l as the intensities subtracting μ to present the high-frequency signals. For patch P_h, we only extract the features of a central region of P_h (e.g., the shaded region of HR patch in Figure 1) and discard four boundary pixels. We do not learn mapping functions to predict the HR boundary pixels as the LR patch P_l does not carry sufficient information and is less likely to predict the HR boundary pixels correctly.

We collect a large set of LR patches from natural images to learn K cluster centers of their extracted features. Figure 2 shows 4096 cluster centers learned from 2.2 million natural patches. Similar to gradients of natural images which can be modeled by a heavy-tail distribution [7], more populous cluster centers correspond to smoother patches as shown in Figure 3. These K cluster centers can be viewed as a set of anchor points to represent the feature space of natural image patches for super resolution.

For some regions in the feature space where natural patches appear fairly rarely, it is unnecessary to learn mapping functions to predict patches of HR from LR. Since each cluster represents a subspace, we collect a certain number of exemplar patches in the segmented space to training a mapping function. Since natural images are abundant and easily acquired, we can assume that there are sufficient exemplar

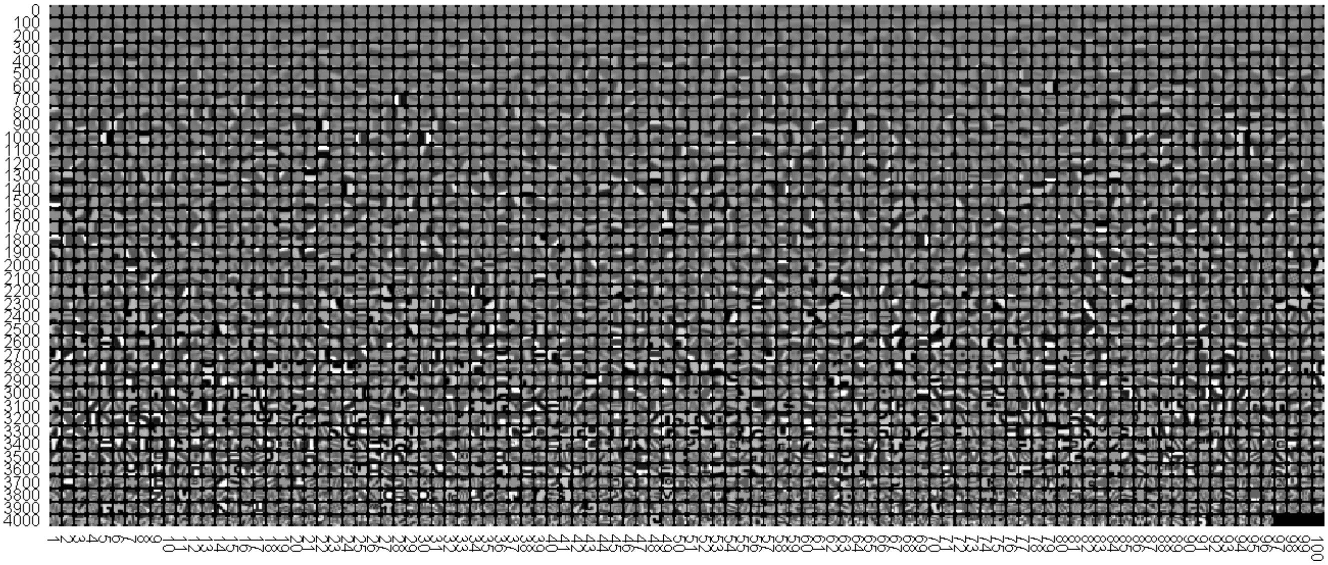


Figure 2. A set of 4096 cluster centers learned from 2.2 million natural patches. As the features for clustering are the intensities subtracting patch means, we show the intensities by adding their mean values for visualization purpose. The order of cluster centers is sorted by the amounts of clustered patches, as shown in Figure 3. Patches with more high-frequency details appear less frequently in natural images.

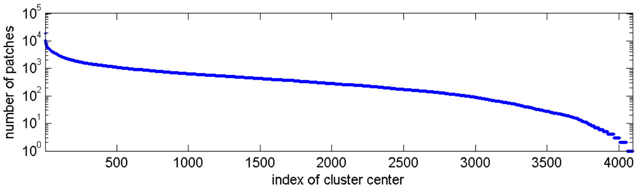


Figure 3. Histogram of clustered patches from a set of 2.2 million natural patches with cluster centers shown in Figure 2. While the most populous cluster center consists of 18489 patches, the 40 least popular clusters only have one patch. A cluster center has 537 patches on average.

patches available for each cluster center.

Suppose there are l LR exemplar patches belonging to the same cluster center. Let $\mathbf{v}_i$ and $\mathbf{w}_i$ $(i = 1, \ldots, l)$ be vectorized features of the LR patches and of the central region of HR patches, and suppose their dimensions are m and n. We propose to learn a set of n linear regression functions to individually predict the n feature values in HR. Let $\mathbf{V} \in \mathbb{R}^{m \times l}$ and $\mathbf{W} \in \mathbb{R}^{n \times l}$ be the matrices of $\mathbf{v}_i$ and $\mathbf{w}_i$. We compute the regression coefficients $\mathbf{C}^* \in \mathbf{R}^{n \times (m+1)}$ by

$$\mathbf{C}^* = \underset{\mathbf{C}}{\operatorname{argmin}} \left\| \; \mathbf{W} - \mathbf{C} \begin{pmatrix} \mathbf{V} \\ \mathbf{1} \end{pmatrix} \; \right\|^2, \tag{2}$$

where $\mathbf{1}$ is a $1 \times l$ vector with all values as 1, and this linear least-squares problem is easily solved.

Given a LR test image, we crop each LR patch to compute the LR features and search for the closest cluster center. According to the cluster center, we apply the learned coefficients to compute the HR features by

$$\mathbf{w} = \mathbf{C}^* \begin{pmatrix} \mathbf{v} \\ 1 \end{pmatrix}. \tag{3}$$

The predicted HR patch intensity is then reconstructed by adding the LR patch mean to the HR features.

The proposed method generates effective HR patches because each test LR patch and its exemplar LR patches are highly similar as they belong to the same compact feature subspace. The computational load for generating a HR image is low as each HR patch can be generated by a LR patch through a few additions and multiplications. The algorithm can be easily implemented with GPU parallel processors because all LR patches are upsampled individually. In addition, the proposed method is suitable for hardware implementations as only few lines of code are required.

4. Experimental Results

Implementation: For color images, we apply the proposed algorithm only on grayscale channel and upsample frames in the color channels by bicubic interpolation as human vision is more sensitive to brightness change. For a scaling factor 4, we set the Gaussian kernel width in Eq. 1 to 1.6 as commonly used in the literature [16]. The LR patch size is set as 7×7 pixels, but the four pixels at the four corners are discarded and thus the LR feature dimension is 45. The central region of a HR patch is set as the 12×12 pixels (See Figure 1). Since the scaling factor is 4, a pixel in HR is covered by 9 LR patches and the output intensity is generated by averaging 9 predicted values. We collect a set of 6000 HR natural images from the Berkeley segmentation [10] and LabelMe [14] datasets to generate the LR training image set containing 679 million patches.

Number of clusters: Due to the memory limitation on a machine (24 GB), we randomly select 2.2 million patches to learn a set of 4096 cluster centers, and use the learned

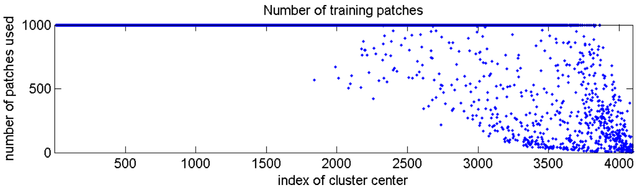
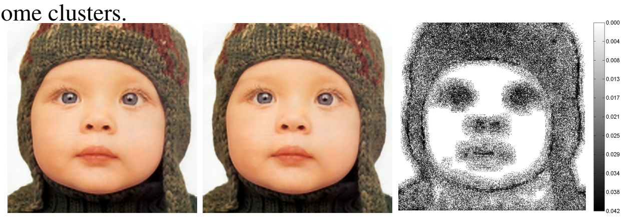


Figure 4. Number of training patches used to train regression coefficients in our experiments. Since some patches are rarely observed in natural images, there are fewer than 1000 patches in some clusters.



(a) 512 clusters (b) 4096 clusters (c) Difference map

Figure 5. Super resolution results with different numbers of cluster centers. Images best viewed on high-resolution display where each image is shown with at least 512 × 512 pixels (full resolution).

cluster centers to label all LR patches in training image set. As the proposed function regresses features from 45 dimensions to one dimension only (each row of $\mathbf{C}^*$ in Eq. 2 is assumed to be independent) and most training features are highly similar, a huge set of training instances is unnecessary. We empirically choose a large value, e.g., 1000, as the size of training instances for each cluster center and collect training instances randomly from the labeled patches. Figure 4 shows the actual numbers of training patches. Since some patches are rarely observed in natural images, there are fewer than 1000 patches in a few clusters. For such cases we still compute the regression coefficients if there is no rank deficiency in Eq. 2, i.e., at least 46 linear independent training vectors are available. Otherwise, we use bilinear interpolation to map LR patches for such clusters.

The number of cluster centers is a trade-off between image quality and computational load. Figure 5 shows the results generated by 512 and 4096 clusters with the same settings. While the low-frequency regions are almost the same, the high-frequency regions of the image generated by more clusters are better in terms of less jaggy artifacts along the face contour. With more clusters, the input feature space can be divided into more compact subspaces from which the linear mapping functions can be more effectively learned.

In addition, we also evaluate the use of a support vector regressor (SVR) as the mapping function, with a Radial Basis Function kernel and a linear kernel. With the same setup, the images generated by SVRs and linear regressor are visually almost the same (See the supplementary material for examples). However, the computational load of using SVRs are much higher due to the load to compute the similarity of each support vector to the test vector. While an image can be generated by the proposed method in 14 seconds, it take 1.5 hours by using SVRs.

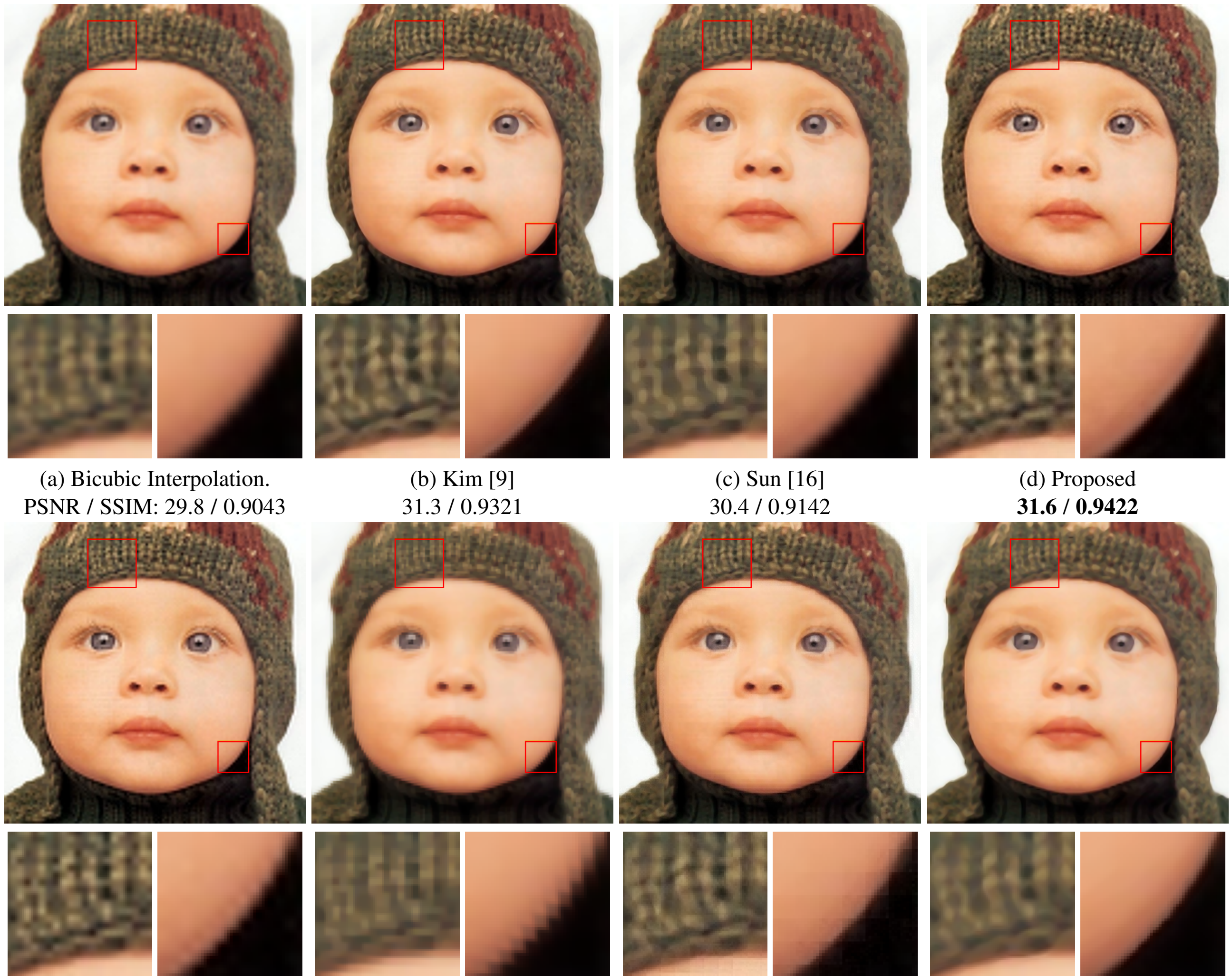
Table 1. Average evaluated values of 200 images from the Berkeley segmentation dataset [10]. While the generated SR images by the proposed method are comparable to those by the self-exemplar SR algorithm [5], the required computational load is much lower (14 seconds vs. 10 minutes).

Algorithm	PSNR	SSIM [22]
Bicubic Interpolation	24.27	0.6555
Back Projection [8]	25.01	0.7036
Sun [16]	24.54	0.6695
Shan [15]	23.47	0.6367
Yang [24]	24.31	0.6205
Kim [9]	25.12	0.6970
Wang [21]	24.32	0.6505
Freedman [3]	22.22	0.6173
Glasner [5]	**25.20**	0.7064
Proposed	25.18	**0.7081**

Evaluation and analysis: We implement the proposed algorithm in MATLAB, which takes 14 seconds to upsample an image of 128 × 128 pixels with scaling factor 4 on a 2.7 GHz Quad Core machine. The execution time can be further reduced with other implementations and GPU. We use the released code [15, 24, 21] to generate HR images, and implement other state-of-the-art algorithms [8, 16, 5, 3] as the source codes are not available. Using the same setup, it takes 10 minutes for the exemplar-based method [5] to process a 128×128 image. All the source codes and datasets will be made available to the public.

Figure 6-11 show SR results of the proposed algorithm and the state-of-the-art methods. More results are available in the supplementary material. Numerical evaluations in terms of PSNR and SSIM index [22] are computed when the ground truth images are available. Table 1 shows averaged indexes for a set of 200 natural images. The evaluations are presented from the four perspectives with comparisons to SR methods using statistical priors [9, 16], fast SR algorithms [8, 15], self-exemplar SR algorithms [5, 3], and SR approaches with dictionary learning [24, 21].

SR methods based on statistical priors: As shown in Figure 6(b)(c), Figure 8(a), Figure 10(c), and Figure 11(b)(c), the proposed algorithm generates textures with better contrast than existing methods using statistical priors [9, 16]. While a kernel ridge regression function is learned in [9] and a gradient profile prior is trained in [16] to restore the edge sharpness based on an intermediate bicubic interpolated images, the high-frequency texture details are not generated due to the use of the bicubic interpolated intermediate image. Furthermore, a post-processing filter is used in [9] to suppress median gradients in order to reduce image noise generated by the regression function along edges. However, mid-frequency details at textures may be wrongly reduced and the filtered textures appear unrealistic. There are four differences between the proposed method and the



(a) Bicubic Interpolation.
PSNR / SSIM: 29.8 / 0.9043

(b) Kim [9]
31.3 / 0.9321

(c) Sun [16]
30.4 / 0.9142

(d) Proposed
31.6 / 0.9422

(e) Back Projection [8]
PSNR / SSIM: 31.1 / 0.9391

(f) Shan [15]
27.8 / 0.8554

(g) Yang [24]
30.1 / 0.9152

(h) Wang [3]
29.5 / 0.8859

Figure 6. Child. Results best viewed on a high-resolution display with adequate zoom level where each image is shown with at least 512 × 512 pixels (full resolution).

existing methods based on statistical priors. First, the proposed method upsamples the LR patches directly rather than using an intermediate image generated by bicubic interpolation, and thus there is no loss of texture details. Second, the proposed regressed features can be applied to any types of patches, while existing methods focus only on edges. Third, no post-processing filter is required in the proposed method to refine the generated HR images. Fourth, existing methods learn a single regressor for the whole feature space, but the proposed method learns a large number of regressors (one for each subspace), thereby making the predicting results more effective.

Fast SR methods: Compared with existing fast SR methods [8, 15] and bicubic interpolation, Figure 6(a)(e)(f), Figure 7, and Figure 11(a)(b) show that the proposed method generates better edges and textures. Although the bicubic interpolation is the fastest method, the generated edges and textures are always over-smoothed. While the back-projection approach [8] enforces the HR image constrained by the LR image through Eq. 1, contrast along edges is enhanced but the contours are not. These results are caused by the fixed back-projection kernel, which is assumed to be a Gaussian kernel to isotropically back-project the difference from a LR pixel to its source HR pixels in Eq. 1. However, the image structures along sharp edges are highly anisotropic, and thus an isotropic Gaussian kernel is less effective in such patches. Thus, the back-projected edges are jaggy and displeasing. A global gradient distribution is exploited as constraints in [15] to achieve fast SR. However, although the global gradient distribution is reconstructed by [15] in Figure 6(f) and Figure 7(c), the local gradients are not constrained. Thus, over-smoothed textures as well as jaggy edges are generated by this method. The pro-

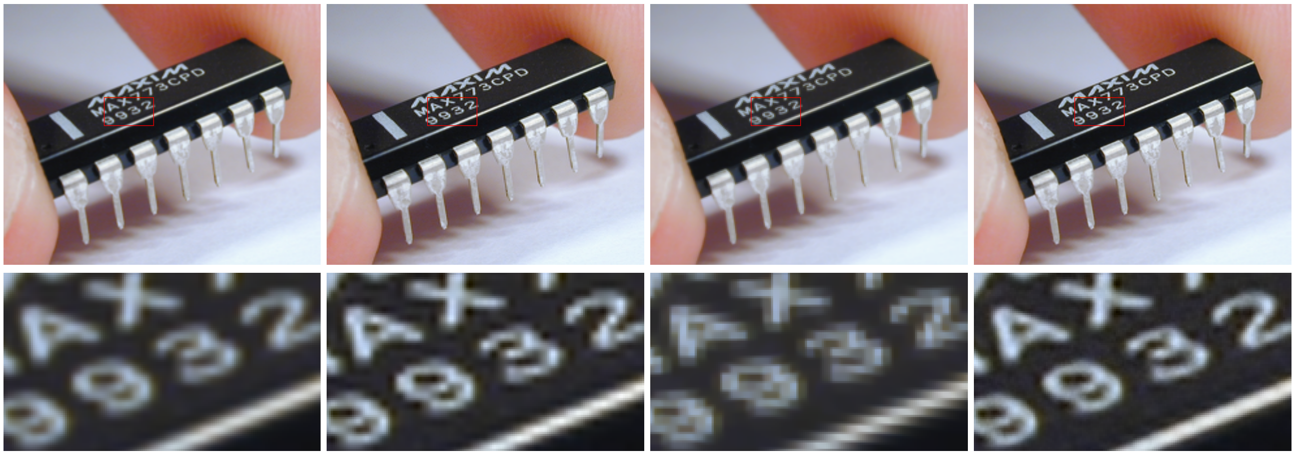

(a) Bicubic Interpolation (b) Back Projection [8] (c) Shan [5] (d) Proposed

Figure 7. IC. Results best viewed on a high-resolution display with adequate zoom level where each image is shown with at least 974 × 800 pixels (full resolution). Because the ground truth image does not exist, the PSNR and SSIM indexes can not be computed.

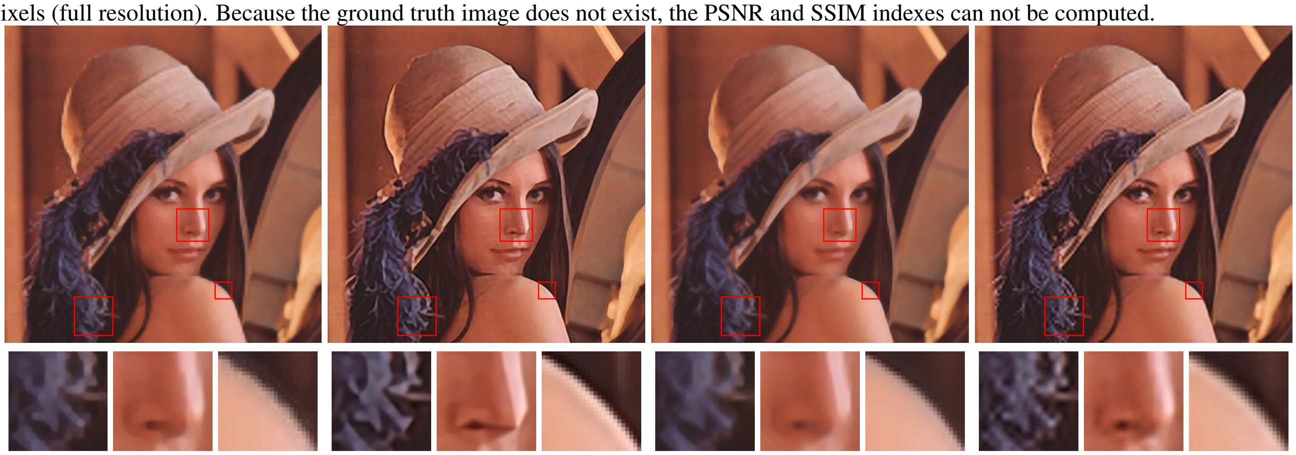
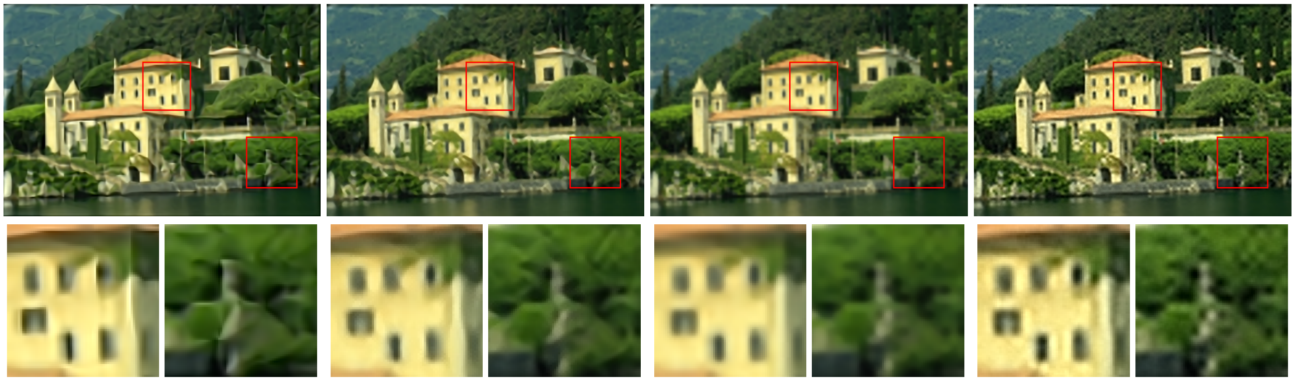

(a) Sun [16] PSNR / SSIM: 28.7 / 0.8827 (b) Glasner [5] 29.1 / 0.9002 (c) Wang [21] 28.5 / 0.8642 (d) Proposed **29.6 / 0.9071**

Figure 8. Lena. Results best viewed on a high-resolution display with adequate zoom level where each image is shown with at least 512 × 512 pixels (full resolution).

posed method generates better edges and textures as each LR patch is upsampled by a specific prior learned from a compact subspace of similar patches. Thus, the contrast and local structures are preserved and less artifacts are generated.

SR methods based on self exemplars: Figure 8(b)(d), Figure 9(a)(d), Figure 10(b)(d), and Figure 11(c)(d) show the results generated by self-exemplar SR methods and the proposed algorithm. Self-exemplar SR algorithms [5, 3] iteratively upsample images with a small scaling factor (e.g., 1.25). Such an approach has an advantage of generating sharp and clear edges because it is easy to find similar patches in some slightly downsampled input images, but difficult to do so in other image datasets. However, the number of patches generated by the input image is still limited. To reduce the errors caused by using a small exemplar set, the back-projection technique is used in [5] to compensate the generated image in each upsampling loop. However, over-compensated artifacts may also be generated, as shown in Figure 8(b) and Figure 11(c), the edges and textures are over-sharpened and unnatural. In addition, it entails computationally expensive patch match processes that prevents it from real-time applications although this problem is alleviated by local approximate operations and GPU [3]. However, such fast approximation methods [3] are likely to exacerbate the problems with the self-exemplar SR method [5] and generate more artifacts. As shown in Figure 9(a), the structure of windows and the texture of bushes are distorted. The details of nose structure are almost lost in Figure 10(b) with unrealistic stripes near the hand and rock region. In contrast, the proposed method overcomes the difficulty of finding rare patches by using a huge exemplar set. Even the probability of finding similar edge patches is low in a few training images, the chances are improved significantly with thousands of images. Although the computational load to collect those exemplar patches is high in the offline train-



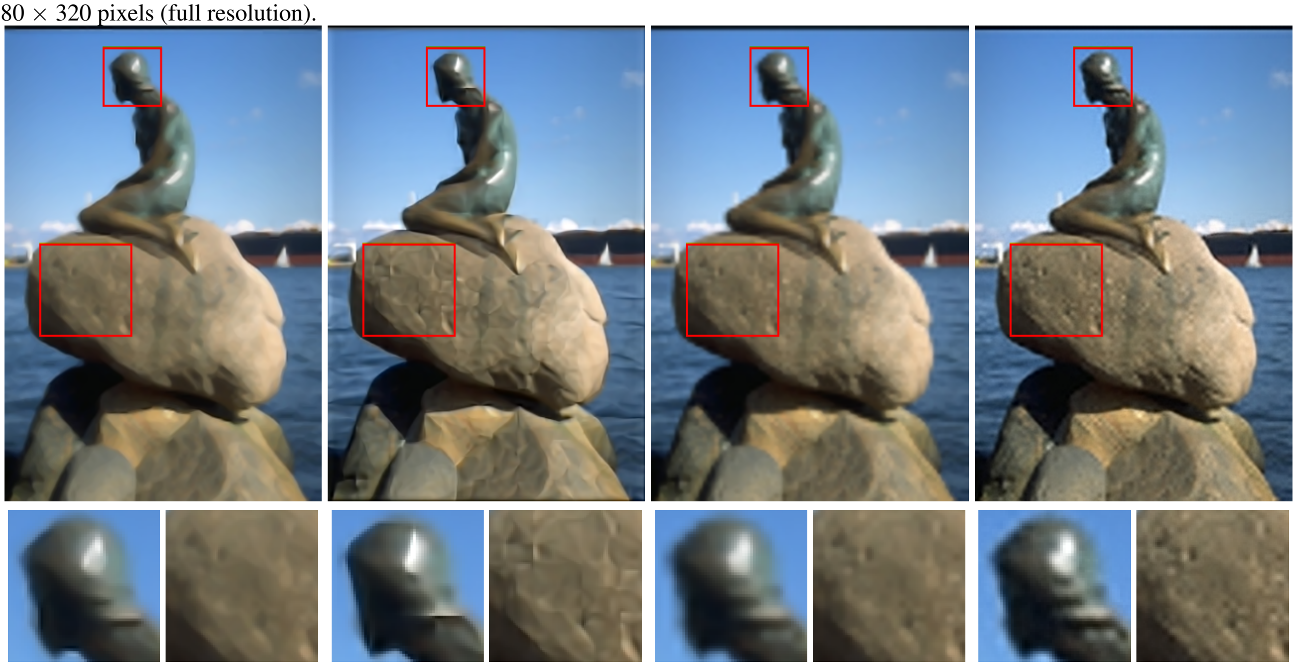
(a) Freedman [3]
PSNR / SSIM: 18.7 / 0.3876

(b) Kim [9]
20.4 / 0.4699

(c) Sun [16]
20.1 / 0.4395

(d) Proposed
20.5 / 0.4921

Figure 9. Mansion. Results best viewed on a high-resolution display with adequate zoom level where each image is shown with at least 480 × 320 pixels (full resolution).



(a) Wang [21]
PSNR / SSIM: 25.9 / 0.6746

(b) Freedman [3]
23.2 / 0.6374

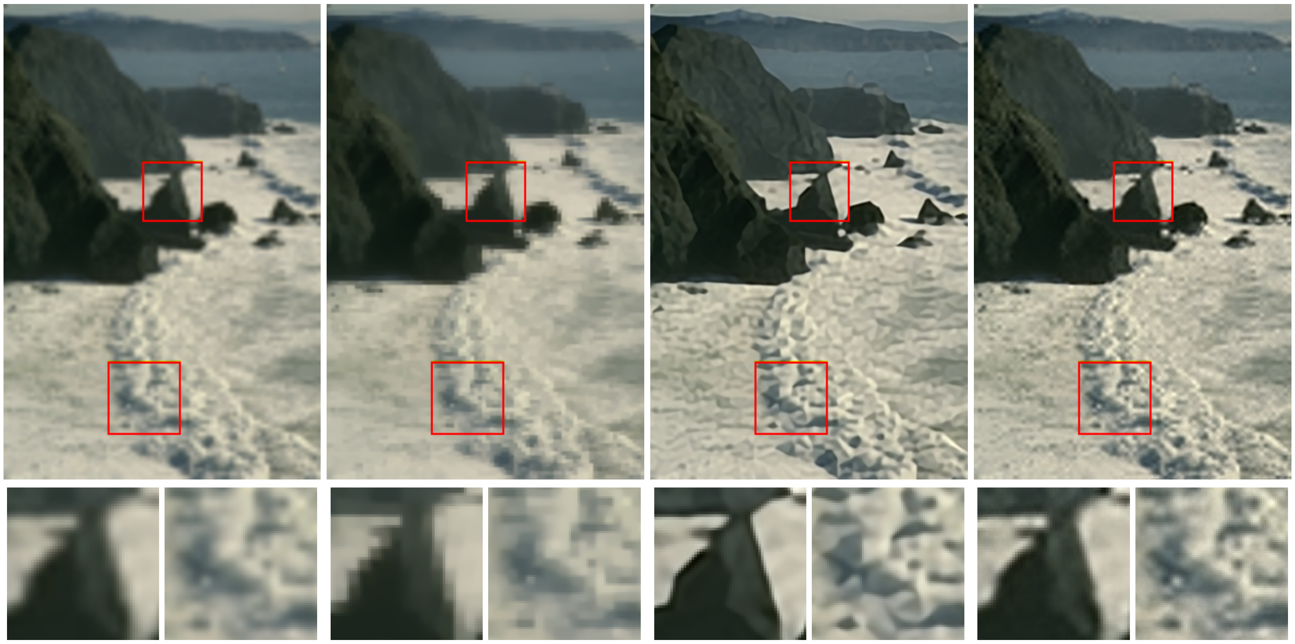
(c) Sun [16]
25.8 / 0.6606

(d) Proposed
26.5 / 0.7133

Figure 10. Mermaid. Results best viewed on a high-resolution display with adequate zoom level where each image is shown with at least 320 × 480 pixels (full resolution).

ing phase, a SR image can be generated very efficiently. As shown in Figure 6(d), Figure 7(d), Figure 8(d), Figure 10(d), and Figure 11(d), the proposed method generates SR images with sharp edges effectively.

SR methods based on dictionary learning: Figure 6(g)(h), Figure 8(c), and Figure 10(a) show images generated by SR algorithms based on dictionary learning [24, 21]. The proposed algorithm is different from these algorithms in several aspects. Numerous mapping functions are learned individually by the proposed algorithm from sets of training patches grouped by similarity, but only one mapping function (the paired dictionaries) is trained by all patches in the existing SR methods based on sparse dictionary learning [24, 21]. Therefore, the learned dictionaries may not capture the details from the all diverse and some infrequent patches. Blocky edges can be observed in Figure 6(g) as such patches of sharp edges appear less frequently than other smooth patches in natural images. An additional transform matrix is used in [21] to reduce the problem by increasing the flexibility of mapping the LR sparse coefficients to HR sparse coefficients. As shown in Figure 6(h), Figure 8(c) and Figure 10(a), the edges are sharp without blocky artifacts. However, the additional transform matrix blurs textures because the mapping of sparse coefficients becomes many-to-many rather than one-to-one, which results in effects of averaging. In contrast, the pro-



(a) Bicubic Interpolation
PSNR / SSIM: 23.1 / 0.5241

(b) Shan [15]
22.4 / 4908

(c) Glasner [5]
23.7 / **0.5839**

(d) Proposed
23.8 / 0.5835

Figure 11. Shore. Results best viewed on a high-resolution display with adequate zoom level where each image is shown with at least 320 × 480 pixels (full resolution).

posed method exploits the advantage of the divide-and-conquer approach to ensure each linear function effectively works in a compact feature subspace. Using simple feature and linear functions, the proposed method generates sharper edges than [24] and richer textures than [21], as shown in Figure 6(d)(g)(h), Figure 8(c)(d), and Figure 10(a)(d).

5. Conclusions

In this paper, we propose a fast algorithm which learns mapping functions to generate SR images. By splitting the feature space into numerous subspaces and collecting sufficient training exemplars to learn simple regression functions, the proposed method generates high-quality SR images with sharp edges and rich textures. Numerous experiments with qualitative and quantitative comparisons against several state-of-the-art SR methods demonstrate the effectiveness and stability of the proposed algorithm.

6. Acknowledge

This work is supported in part by the NSF CAREER Grant #1149783 and NSF IIS Grant #1152576.

References

[1] H. Chang, D.-Y. Yeung, and Y. Xiong. Super-resolution through neighbor embedding. In *CVPR*, 2004.

[2] R. Fattal. Image upsampling via imposed edge statistics. In *SIGGRAPH*, 2007.

[3] G. Freedman and R. Fattal. Image and video upscaling from local self-examples. *TOG*, 30(2):1–11, 2011.

[4] W. T. Freeman, T. R. Jones, and E. C. Pasztor. Example-based super-resolution. *IEEE Computer Graphics and Applications*, pages 56–65, 2002.

[5] D. Glasner, S. Bagon, and M. Irani. Super-resolution from a single image. In *ICCV*, 2009.

[6] Y. HaCohen, R. Fattal, and D. Lischinski. Image upsampling via texture hallucination. In *ICCP*, 2010.

[7] J. Huang and D. Mumford. Statistics of natural images and models. In *CVPR*, 1999.

[8] M. Irani and S. Peleg. Improving resolution by image registration. *CGVIP*, 53(3):231–239, 1991.

[9] K. I. Kim and Y. Kwon. Single-image super-resolution using sparse regression and natural image prior. *PAMI*, 32(6):1127 –1133, 2010.

[10] 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 *ICCV*, volume 2, pages 416–423, 2001.

[11] P. Milanfar, editor. *Super-Resolution Imaging*. CRC Press, 2010.

[12] K. Ni and T. Nguyen. Image superresolution using support vector regression. *IEEE TIP*, 16(6):1596–1610, 2007.

[13] S. C. Park, M. K. Park, and M. G. Kang. Super-resolution image reconstruction: A technical overview. *IEEE Signal Processing Magazine*, pages 21–36, 2003.

[14] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. LabelMe: A database and web-based tool for image annotation. *IJCV*, 77(1-3):157 –173, 2008.

[15] Q. Shan, Z. Li, J. Jia, and C.-K. Tang. Fast image/video upsampling. In *SIGGRAPH Asia*, 2008.

[16] J. Sun, J. Sun, Z. Xu, and H.-Y. Shum. Image super-resolution using gradient profile prior. In *CVPR*, 2008.

[17] J. Sun, J. Zhu, and M. F. Tappen. Context-constrained hallucination for image super-resolution. In *CVPR*, 2010.

[18] L. Sun and J. Hays. Super-resolution from internet-scale scene matching. In *ICCP*, 2012.

[19] Y.-W. Tai, S. Liu, M. S. Brown, and S. Lin. Super resolution using edge prior and single image detail synthesis. In *CVPR*, 2010.

[20] J. Wang, S. Zhu, and Y. Gong. Resolution enhancement based on learning the sparse association of image patches. *Pattern Recognition Letters*, 31(1):1–10, 2010.

[21] S. Wang, L. Zhang, Y. Liang, and Q. Pan. Semi-coupled dictionary learning with applications to image super-resolution and photo-sketch synthesis. In *CVPR*, 2012.

[22] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE TIP*, 13(4):600 –612, 2004.

[23] H. Wei, X. Liang, W. Zhihui, F. Xuan, and W. Kai. Single image super-resolution by clustered sparse representation and adaptive patch aggregation. *China Communications*, 10(5):50–61, 2013.

[24] J. Yang, J. Wright, T. Huang, and Y. Ma. Image super-resolution via sparse representation. *IEEE TIP*, 2010.

[25] S. Yang, M. Wang, Y. Chen, and Y. Sun. Single-image super-resolution reconstruction via learned geometric dictionaries and clustered sparse coding. *IEEE TIP*, 21(9):4016–4028, 2012.