A Bayesian Nonparametric Approach to Image Super-Resolution

Güngör Polatkan, Mingyuan Zhou, *Member, IEEE*, Lawrence Carin, *Fellow, IEEE*, David Blei, and Ingrid Daubechies, *Fellow, IEEE*

Abstract—Super-resolution methods form high-resolution images from low-resolution images. In this paper, we develop a new Bayesian nonparametric model for super-resolution. Our method uses a beta-Bernoulli process to learn a set of recurring visual patterns, called dictionary elements, from the data. Because it is nonparametric, the number of elements found is also determined from the data. We test the results on both benchmark and natural images, comparing with several other models from the research literature. We perform large-scale human evaluation experiments to assess the visual quality of the results. In a first implementation, we use Gibbs sampling to approximate the posterior. However, this algorithm is not feasible for large-scale data. To circumvent this, we then develop an online variational Bayes (VB) algorithm. This algorithm finds high quality dictionaries in a fraction of the time needed by the Gibbs sampler.

Index Terms—Bayesian nonparametrics, factor analysis, dictionary learning, variational inference, gibbs sampling, stochastic optimization, image super-resolution

1 Introduction

The sparse representation of signals with a basis is important in many applications. It has been extensively used in image denoising [1], [2], inpainting [3], [4], and classification [5], [6].

Many real data sets can be sparsely represented in some basis; typically this basis itself has to be learned from the data [7], [8], [9], [10], [11], [12]. For example, an image can be represented by weighted combinations of recurrent patterns of pixels. This construction may be beneficial, both while building a model for more accurate representation of the data (e.g., superior image denoising models) and while deriving and implementing an inference procedure for more efficient algorithms.

In this paper we consider image super-resolution (SR), the problem of recovering a high-resolution (HR) image from a low-resolution (LR) image. It has many applications, e.g., to smart phones, surveillance cameras, medical imaging, and satellite imaging.

There are a variety of approaches for image super-resolution. In general, rendering an HR image from an LR image has many possible solutions. We must use regularization of some form, i.e., prior information about the HR, to guarantee uniqueness and stability of the extension. For this purpose, researchers have proposed several methods [13], [14]. *Interpolation-based methods*, such as the Bicubic method and Bilinear method, often over-smooth images, losing detail. *Example-based approaches* use machine learning to avoid this [15], [16], [17]; they train on ground-truth HR and LR (either single image or multiple images), learning a statistical relationship between the two. These relationships are later used to reconstruct unknown HR images from corresponding LR images. Freeman et al. [15] proposes a method that stores a training set of preprocessed patches and uses a nearest-neighbor search to super-resolve. Kim and Kwon [16] proposes using kernel ridge regression with a regularized gradient descent. Another class of SR algorithms use texture similarity to match image regions with known textures [18], [19]. SR algorithms can also be classified as SR using single-image versus multiple images. Many machine learning based techniques fall into the multiple image category. One classic single-image SR example is [20] which uses recurring patterns at same and different scales in a single image. Another such example is [21], which uses a Gaussian process regression.

In this work, our focus is on SR via example-based sparse coding. Super-resolution via Sparse representation (ScSR) is such an algorithm pioneered in [22]. ScSR is based on sparse coding via L1 regularized optimization. In [22], image data are represented using a collection of dictionary elements (recurring patterns of pixels) that are weighted across different positions. Although very powerful, this model requires one to specify the number of dictionary elements and the variance of the noise model in advance—parameters that may be difficult to assess for real-world images. It also only provides a batch learning algorithm, i.e., computing model parameters via a gradient descent algorithm on a small subset of the data.

Bayesian nonparametric methods circumvent all these limitations. These methods adapt the structure of the latent space

- *G. Polatkan is with Twitter Inc., San Francisco, CA. E-mail: gungorpolatkan@gmail.com.*
- *M. Zhou is with the Department of Statistics at the University of Texas at Austin, Austin, TX. E-mail: mz31@duke.edu.*
- *L. Carin is with the Department of Electrical Engineering, Duke University, Durham, NC. E-mail: lcarin@duke.edu.*
- *D. Blei is with the Department of Computer Science at Princeton University, Princeton, NJ. E-mail: blei@cs.princeton.edu.*
- *I. Daubechies is with the Department of Mathematics, Duke University, Durham, NC. E-mail: ingrid@math.duke.edu.*

Manuscript received 8 Sept. 2012; revised 6 Apr. 2014; accepted 13 Apr. 2014. Date of publication 30 Apr. 2014; date of current version 14 Jan. 2015. Recommended for acceptance by R.P. Adams, E. Fox, E. Sudderth, and Y.W. Teh.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPAMI.2014.2321404

to the data and provide a powerful representation because they infer parameters that otherwise have to be assigned a priori [23], [24], [25], [26], [27]. The full posterior distribution can be approximated via MCMC or variational inference, yielding sparse representations and learned dictionaries.

Bayesian nonparametric methods have been used in many image analysis applications: to learn deep architectures used for object recognition in [28], for image inpainting and denoising in [29], [30], for image segmentation in [27], [31], and to learn nonparametric multiscale representations of images in [32].

In this paper, we develop a Bayesian nonparametric method for super-resolution. We show that inference in our model is feasible, performing super-resolution with both a sampling based algorithm [33] and an online variational inference algorithm [34]. In the latter, we approximate the posterior distributions via a stochastic gradient descent over a variational objective that enables us to use the full data set and process the data segment by segment. We also provide human evaluation experiments which shows that signal-to-noise ratio (a typical quantitative measure of success in image analysis applications) is not necessarily consistent with human judgement. We devise a new model, new algorithms, and study a human-based evaluation. We make the following contributions:

- We develop a sparse Bayesian nonparametric model for SR, learning the number of dictionary elements and the noise variance from the data.
- We develop an online variational Bayes (VB) algorithm finding high quality "coupled dictionaries" in a fraction of the time needed by traditional inference.
- We devise large scale human evaluation experiments to explicitly assess the visual quality of results.

Our approach to SR gives a rich nonparametric representation with scalable learning.

The remainder of the paper is organized as follows: Section 2 describes the proposed super-resolution model and non-parametric prior, Section 3 contains the derivation of the posterior inference algorithms, Section 4 presents the experimental results and implementation details, Section 5 includes the discussion and future work.

2 Proposed Approach

Bayesian factor analysis can be used to learn factors / dictionaries from natural images. Zhou et al. [29] used beta process factor analysis in image denoising, inpainting and compressive sensing. These models learn both the dictionary elements and their number from the data.

We build here a nonparametric factor analysis model that couples a high-resolution image to a corresponding low-resolution image. In training, we learn the HR/LR relationship from observed HR/LR pairs. Once trained, we perform super-resolution by conditioning on an observed LR image and computing the expectation of its corresponding HR image.

A more detailed description of the training process is as follows: We create training data by taking observed HR images and forming corresponding LR images. Fig. 1 depicts the preprocessing and data extraction steps. We first down-sample the HR images. Then, we up-sample those by interpolating with a deterministic weighting function (e.g., bicubic interpolation). We extract same-sized patches from the same locations of both the HR and interpolated LR images, and consider those patches as coupled to each other. These are the data on which we train the model.

Fig. 1. Depicting the observations extracted (e.g., image patches) from high and low resolution images.

In the model, each small patch is generated from latent global dictionary elements—small images functioning as factor loadings—using local sparse weights and Gaussian noise. We will first explain how these latent variables are generated and then present how they are used to generate the observations.

We learn two dictionaries: one for HR images and one for LR images. In terms of notation, $\mathbf{d}_k^{(l)}$ represents the LR dictionary element, and $\mathbf{d}_k^{(h)}$ is the HR dictionary element. $v^{(l)}$ and $v^{(h)}$ represent the dimensionality of the low and high resolution dictionary elements, respectively. $\mathbf{I}_{v^{(h)}}$ represents $v^{(h)} \times v^{(h)}$ identity matrix. To model each dictionary element, we use a zero-mean Gaussian distribution,

$$\mathbf{d}_k^{(l)} \sim \mathcal{N}\big(0, {v^{(l)}}^{-1}\mathbf{I}_{v^{(l)}}\big) \quad \mathbf{d}_k^{(h)} \sim \mathcal{N}\big(0, {v^{(h)}}^{-1}\mathbf{I}_{v^{(h)}}\big).$$

The matrix form of the dictionaries are $\mathbf{D}^{(l)}$ and $\mathbf{D}^{(h)}$ where kth columns are $\mathbf{d}_k^{(l)}$ and $\mathbf{d}_k^{(h)}$, respectively.

Following [22], we assume that the sparse weights are shared by both resolution levels for combining dictionary elements to produce images. This is the key property of the model that allows us to frame super-resolution as inference. Sparse weights have two components: real valued weights s_{ik} and binary valued assignments z_{ik}. To model the weights s_{ik}, we use a zero-mean Gaussian distribution with precision γ_s. $\mathbf{z}_i$ is a binary vector that encodes which dictionary elements are activated for the observation i. $p(\mathbf{z})$ represents the prior of $\mathbf{z}$ and we will elaborate on this in the next section. These are given as

$$s_{ik} \sim \mathcal{N}(0, 1/\gamma_s) \quad z_{ik} \sim p(z_{ik}).$$

We place Gamma priors on the precisions of the sparse weights and observation noise (γ_s and γ_ϵ). The two resolution levels share these variables as well,

$$\gamma_\epsilon \sim \text{Gamma}(c, d), \quad \gamma_s \sim \text{Gamma}(e, f),$$

where c, d, e, f are constant hyper-parameters.

Let $\mathbf{x}_i^{(h)}$ and $\mathbf{x}_i^{(l)}$ represents patches extracted from HR and LR images, respectively, as shown in Fig. 1. Given the (global) dictionary elements and (local) sparse weights, the observations are modeled as

$$\boldsymbol{\epsilon}_i^{(l)} \sim \mathcal{N}\big(\mathbf{0}, \gamma_\epsilon^{-1}\mathbf{I}_{v^{(l)}}\big) \quad \boldsymbol{\epsilon}_i^{(h)} \sim \mathcal{N}\big(\mathbf{0}, \gamma_\epsilon^{-1}\mathbf{I}_{v^{(h)}}\big)$$
$$\boldsymbol{x}_i^{(l)} = \mathbf{D}^{(l)}(\mathbf{s}_i \odot \mathbf{z}_i) + \boldsymbol{\epsilon}_i^{(l)} \quad \boldsymbol{x}_i^{(h)} = \mathbf{D}^{(h)}(\mathbf{s}_i \odot \mathbf{z}_i) + \boldsymbol{\epsilon}_i^{(h)},$$

where $\{(l), (h)\}$ represents LR and HR, respectively. Here, N is the total number of patches, $\boldsymbol{\epsilon}_i^{(h)}$ is the observation

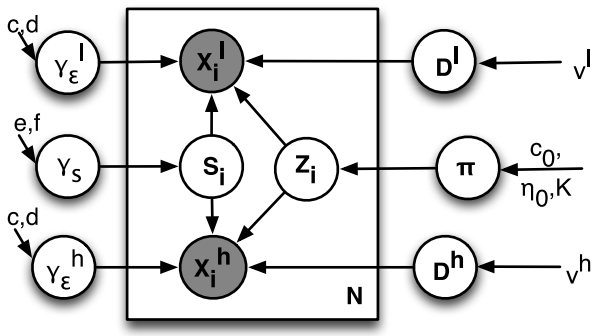


Fig. 2. Graphical model.

noise, and $\odot$ represents the element-wise multiplication of two vectors. Fig. 2 illustrates the graphical model.

To use this model in SR, we must be able to compute the posterior distributions of the hidden variables. In the training phase, we must compute the posterior distributions $p(\mathbf{D}^{(h)}, \mathbf{D}^{(l)} | \{\mathbf{x}_i^{(h)}, \mathbf{x}_i^{(l)}\})$ of the dictionaries, given a collection of HR/LR image pairs. In testing, we use their posterior expectation to reconstruct a held-out HR image from an LR image,

$$\mathbb{E}\left[\mathbf{x}_j^{(h)} \mid \mathbf{x}_j^{(l)}, \left\{\mathbf{x}_i^{(h)}, \mathbf{x}_i^{(l)}\right\}\right] \approx \hat{\mathbf{D}}^{(h)}(\hat{\mathbf{s}}_j \odot \hat{\mathbf{z}}_j), \tag{1}$$

where $\hat{\mathbf{D}}^{(h)}$ is the mean of the posterior distribution $p(\mathbf{D}^{(h)} | \{\mathbf{x}_i^{(h)}, \mathbf{x}_i^{(l)}\})$ and $(\hat{\mathbf{s}}_j \odot \hat{\mathbf{z}}_j)$ are the posterior expectation of the sparse weights from the LR image patches $(\boldsymbol{x}_j^{(l)})$ via posterior inference. (We discuss algorithms for posterior inference in Section 3.)

2.1 Beta-Bernoulli Process Prior (BP)

We now discuss the prior for the factor assignments $\mathbf{z}_i$. We use a beta-Bernoulli process [23], [24], [25], [26], [35], a prior on infinite binary matrices which is connected to the Indian buffet process (IBP). Each row encodes which dictionary elements are activated for the corresponding observation; columns with at least one active cell correspond to factors. A distinguishing characteristics of this prior is that the number of these factors is not specified a priori. Conditioned on the data, we examine the posterior distribution of the binary matrix to obtain a data-dependent distribution of how many components are needed.

The IBP metaphor gives the intuition. Consider a buffet of dishes at a restaurant. Suppose there are infinite number of dishes and we are trying to specify the infinite binary matrix indicating which customers (observations) choose which dishes (factors/dictionary elements). In the Indian buffet process, N customers enter the restaurant sequentially. Each customer chooses dishes in a line from a buffet. The first customer starts from the beginning of the buffet and takes from each dish, stopping after Poisson(τ) number of dishes where τ represents the arrival rate. The ith customer starts from the beginning as well, but decides to take from dishes in proportion to their popularity within the previous $i-1$ customers. This proportionality can be quantified as $\frac{m_k}{i}$ where m_k is the number of previous customers who took this kth dish. After considering the dishes previously taken by other customers, the ith customer tries a Poisson($\frac{\tau}{i}$) number of new dishes. Which customers chose which dishes is recorded by the infinite binary matrix with N rows (indicating the customers/observations) and infinite columns (indicating the dishes/factors/dictionary elements). One important property of this process is that the joint probability of final assignment is independent of the order of customers getting into the restaurant which is called exchangeability property of the prior [36].

The probabilistic process is as follows. Each observation i is drawn from a Bernoulli process (a sequence of independent identically distributed Bernoulli trials), $\mathbf{x}_i \sim \mathrm{BeP}(B)$ where B is drawn from a beta process $B \sim \mathrm{BP}(c_0, B_0)$. B_0 represents the base measure with $B_0 = \mathcal{N}(0, 1/\beta\mathbf{I})$. As $K \to \infty$, the ith observation is $\mathbf{x}_i = \sum_{k=1}^{\infty} z_{ik}\delta_{d_k}$ where z_{ik} denotes whether the dictionary element $\mathbf{d}_k$ is used while representing the ith observation or not, and the sample from the beta process is given by $B = \sum_{k=1}^{\infty} \pi_k \delta_{d_k}$. Here, π_k represents the usage probability of element $\mathbf{d}_k$.

In inference, we use a finite beta-Bernoulli approximation [25]. The finite model truncates the number of dictionary elements to K and is given by

$$\pi_k \sim \mathrm{Beta}(c_0\eta_0, c_0(1-\eta_0)),\ z_{ik} \sim \mathrm{Bernoulli}(\pi_k),$$

where c_0 and η_0 are scalars and $k \in 1, \ldots, K$. As K tends to infinity, the finite beta-Bernoulli approximation approaches the IBP/BP. If the truncation is large enough, data analyzed with this prior will exhibit fewer than K components [23].

2.2 Super-Resolution via Posterior Distributions

Our algorithm has two stages: fitting the model on pairs of HR and LR images, and super-resolving new LR images to create HR versions.

Training: coupled dictionary learning stage. In training, we observe $\mathbf{x}_i^{(h)}$ and $\mathbf{x}_i^{(l)}$. All other random variables are latent. The key inference problem to be solved is the computation of the posterior distributions of the hidden variables. In the training phase, we must compute the posterior distributions $p(\mathbf{D}^{(h)}, \mathbf{D}^{(l)} | \{\mathbf{x}_i^{(h)}, \mathbf{x}_i^{(l)}\})$ of the dictionaries given a collection of HR/LR image pairs. We rewrite the coupled model in a form similar to the single scale model:

$$\boldsymbol{x}_i^{(c)} = \begin{pmatrix} \boldsymbol{x}_i^{(l)} \\ \boldsymbol{x}_i^{(h)} \end{pmatrix}, \mathbf{d}_k^{(c)} = \begin{pmatrix} \mathbf{d}_k^{(l)} \\ \mathbf{d}_k^{(h)} \end{pmatrix}, \boldsymbol{\epsilon}_i^{(c)} = \begin{pmatrix} \boldsymbol{\epsilon}_i^{(l)} \\ \boldsymbol{\epsilon}_i^{(h)} \end{pmatrix}, \tag{2}$$

where the superscript (c) corresponds to combination of (l) and (h). Writing the fully-observed model in this way reveals that we can train the dictionaries with similar methods as for the single-scale base model. (Training amounts to approximating the posteriors of these values.) The differences are that we use combined patches $\boldsymbol{x}_i^{(c)}$ and combined dictionaries $\mathbf{d}_k^{(c)}$. This leads to shared sparse weights for the two resolution levels. (The details of computing the distribution $p(\mathbf{D}^{(h)}, \mathbf{D}^{(l)} | \{\mathbf{x}_i^{(h)}, \mathbf{x}_i^{(l)}\})$ are discussed in Section 3.)

Super-resolving a low resolution image. Fitted dictionaries in hand, we now show how to form HR images from LR images via posterior computation.

In this prediction setting, the HR image $\mathbf{x}_i^{(h)}$ is unknown; the goal is to reconstruct it from the LR image patches $\mathbf{x}_i^{(l)}$, the posterior estimates of the dictionaries $(\hat{\mathbf{D}}^{(h)}, \hat{\mathbf{D}}^{(l)})$, and the precisions $\hat{\gamma}_\epsilon, \hat{\gamma}_s$ of the noise and the sparse weights,

$$\boldsymbol{x}_i^{(c)} = \begin{pmatrix} \boldsymbol{x}_i^{(l)} \\ - \end{pmatrix}, \mathbf{d}_k^{(c)} = \begin{pmatrix} \mathbf{d}_k^{(l)} \\ \mathbf{d}_k^{(h)} \end{pmatrix}, \ \boldsymbol{\epsilon}_i^{(c)} = \begin{pmatrix} \boldsymbol{\epsilon}_i^{(l)} \\ - \end{pmatrix}.$$

First we find estimates of the sparse factor scores, $(\hat{\mathbf{s}}_i \odot \hat{\mathbf{z}}_i)$, by using the LR image patches $\boldsymbol{x}_i^{(l)}$ and posterior estimates of the dictionaries and precisions γ_ϵ and γ_s. The fitted value of γ_s determines the strength of a "regularization term" that controls the sparsity of the factor scores.

More precisely, this prediction setting has three steps. The input is a set of held-out LR image patches $\mathbf{x}_i^{(l)}$, the posterior estimates of the dictionaries $(\hat{\mathbf{D}}^{(h)}, \hat{\mathbf{D}}^{(l)})$, and the precisions $\hat{\gamma_\epsilon}, \hat{\gamma_s}$ of the noise and the sparse weights. The steps are as follows:

1) We find estimates of the sparse factor scores, $(\hat{\mathbf{s}}_i \odot \hat{\mathbf{z}}_i)$, conditioned on the LR image patches $\mathbf{x}_i^{(l)}$ and estimates $(\hat{\mathbf{D}}^{(h)}, \hat{\mathbf{D}}^{(l)}), \hat{\gamma_\epsilon}, \hat{\gamma_s}$ from the training stage.
2) Eq. (1) determines the HR patches $\hat{\mathbf{x}}_i^{(h)}$.
3) We replace each $\mathbf{x}_i^{(l)}$ by its corresponding collocated $\hat{\mathbf{x}}_i^{(h)}$; the whole HR image, $\hat{\mathbf{X}}^{(h)}$, is the pixel-wise average of those overlapping reconstructions.

Post-processing. Following [22], we apply a post-processing step that, when down-sampled, the reconstructed HR image, $\hat{\mathbf{X}}^{(h)}$, should match the given LR image $\mathbf{X}^{(l)}$. Specifically, we solve the following:

$$\hat{\mathbf{X}}^{(h)*} = \underset{\mathbf{X}}{\operatorname{argmin}} \|f(\mathbf{X}) - \mathbf{X}^{(l)}\|_2^2 + c\|\mathbf{X} - \hat{\mathbf{X}}^{(h)}\|_2^2,$$

where $f()$ is a linear operator consisting of an anti-aliasing filter followed by down-sampling. This optimization problem is solved with gradient descent.

3 Posterior Inference

In the proposed approach, all of the priors are in the conjugate exponential family. In a first implementation, we use Gibbs sampling. We iteratively sample from the conditional distribution of each hidden variable given the others and the observations. This defines a Markov chain whose stationary distribution is the posterior [33]. The corresponding sampling equations are analytic and provided in the Appendix A-B (Appendix is in the supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2014.2321404).

The Gibbs sampler has difficulty with scaling to large data, because it must go through many iterations, each time visiting the entire data set before the sampler mixes. For this reason, both our Gibbs sampler and ScSR use 10^5 patches sampled from 3×10^6. We now develop here an alternative algorithm to Gibbs sampling for SR that scales to large and streaming data. Specifically, we develop an online variational inference algorithm.

Variational inference is a deterministic alternative to MCMC that replaces sampling with optimization [34]. The idea is to posit a parameterized family of distribution over the hidden variables and then optimize the parameters to minimize the KL divergence to the posterior of interest [37]. Our algorithm iteratively tracks an approximate posterior distribution, which improves as more data are seen.

In typical applications, the variational objective is optimized with coordinate ascent, iteratively optimizing each parameter while holding the others fixed. However, in Bayesian settings, this suffers from the same problem as Gibbs sampling—the entire data set must be swept through multiple times in order to find a good approximate posterior. In the algorithm we present here, we replace coordinate ascent optimization with *stochastic optimization*—at each iteration, we subsample our data and then adjust the parameters according to a noisy estimate of the gradient. Because we only subsample the data at each iteration, rather than analysing the whole data set, the resulting algorithm scales well to large data. This technique was pioneered in [38] and was recently exploited for online learning of topic models [39] and hierarchical Dirichlet processes [40]. For the general algorithm in conditionally conjugate models, a class of latent variable models that contains ours, see [34].

We first develop the coordinate ascent algorithm for the coupled model. Then we derive the online variational inference algorithm, which can more easily handle large data sets.

3.1 Variational Inference for the Coupled Model

We use the coupling perspective in Section 2.2 to derive the batch variational Bayes algorithm. The single-scale base model is the BPFA model of [35], which gives a mean-field variational inference algorithm. The batch VB algorithm derived here is the coupled version of that.

We first define a parametrized family of distributions over the hidden variables. Let $\mathbf{Q} = \{\boldsymbol{\pi}, \mathbf{Z}, \mathbf{S}, \mathbf{D}, \gamma_\epsilon, \gamma_s\}$ denote the hidden variables for all i, k. We write coupled data as in Equation (2); in the new set-up the variables to be learned become $\mathbf{Q} = \{\boldsymbol{\pi}, \mathbf{Z}, \mathbf{S}, \mathbf{D}^{(c)}, \gamma_\epsilon, \gamma_s\}$. We use a fully factorized variational distribution,

$$q(\mathbf{Q}) = q_\tau(\boldsymbol{\pi}) q_\phi(\mathbf{D}^{(c)}) q_\nu(\mathbf{Z}) q_\omega(\mathbf{S}) q_\lambda(\boldsymbol{\gamma}_\epsilon) q_\epsilon(\boldsymbol{\gamma}_\mathbf{s}).$$

Each component of this distribution is governed by a free variational parameter,

$$\begin{aligned} q_{\tau_k}(\pi_k) &= \text{Beta}(\tau_{k1}, \tau_{k2}) \quad q_{\nu_{ik}}(z_{ik}) = \text{Bernoulli}(\nu_{ik}) \\ q_{\phi_{kj}}(d_{kj}) &= \mathcal{N}(\phi_{kj}, \Phi_{kj}) \quad q_\lambda(\gamma_\epsilon) = \text{Gamma}(\lambda_1, \lambda_2) \\ q_{\omega_{ik}}(s_{ik}) &= \mathcal{N}(\omega_{ik}, \Omega_{ik}) \quad q_\epsilon(\gamma_s) = \text{Gamma}(\epsilon_1, \epsilon_2). \end{aligned}$$

We optimize these parameters with respect to a bound on the marginal probability of the observations. This bound is equivalent, up to a constant, to the negative KL divergence between q and the true posterior. Thus maximizing the bound is equivalent to minimizing KL divergence to the true posterior. Let $\boldsymbol{\Xi} = \{c_0, \eta_0, c, d, e, f\}$ be the hyper-parameters. The variational lower bound is

$$\begin{aligned} \log(p(\mathbf{X}^{(c)} \mid \boldsymbol{\Xi})) \geq H(q) + \sum_{k=1}^{K} \Bigg\{ & \mathbb{E}_\mathbf{q}[\log p(\pi_k \mid c_0, \eta_0, K)] \\ & + \sum_{i=1}^{N} \mathbb{E}_\mathbf{q}[\log p(z_{ik} \mid \boldsymbol{\pi})] + \sum_{j=1}^{J} \mathbb{E}_\mathbf{q}[\log \left(p(d_{kj} \mid \beta_{kj})\right)] \\ & + \sum_{i=1}^{N} \mathbb{E}_\mathbf{q}[\log\big(p(s_{ik} \mid \gamma_s) p(\gamma_s | e, f)\big)] \Bigg\} \\ & + \sum_{i=1}^{N} \big\{ \mathbb{E}_\mathbf{q}[\log p\big(\mathbf{x}_i^{(c)} \mid \mathbf{Z}, \mathbf{S}, \mathbf{D}^{(c)} \gamma_\epsilon\big)] + \mathbb{E}_\mathbf{q}[\log p(\gamma_\epsilon \mid c, d)] \big\}, \end{aligned} \tag{3}$$

where $\mathrm{H}(q)$ is the entropy of the variational distribution and dimensionality of the dictionary elements J is twice as big as the single-scale model. We denote this function $\mathcal{L}(q)$.

Holding the other parameters fixed, we can optimize each variational parameter exactly; this gives an algorithm that goes uphill in $\mathcal{L}(q)$ [41]. (Further, this will provide the algorithmic components needed for the online algorithm of Section 3.2.)

Update equations for each free parameter optimizing this bound are given below. In all equations, $\tilde{\mathbf{x}}_{i(-k)}^{(c)}$ represents the reconstruction error using all but the kth dictionary element, that is

$$\tilde{\mathbf{x}}_{i(-k)}^{(c)} = \mathbf{x}_i^{(c)} - \mathbf{D}^{(c)}(\mathbf{s}_i \odot \mathbf{z}_i) + \mathbf{d}_k^{(c)}(s_{ik} \odot z_{ik}).$$

The expectation based on the variational distribution is then given by

$$\mathbb{E}_q\big[\tilde{\mathbf{x}}_{i(-k)}^{(c)}\big] = \mathbf{x}_i^{(c)} + \boldsymbol{\phi}_k^{(c)}(\omega_{ik}\nu_{ik}) - \sum_{k=1}^{K} \boldsymbol{\phi}_k^{(c)}(\omega_{ik}\nu_{ik}).$$

Update for the binary factor assignment z_{ik}. The variational parameter for z_{ik} is ν_{ik}. We first consider two values of the variational distribution for two values (0, 1) of z_{ik},

$$\begin{aligned} q(z_{ik}=1) \;&\propto\; \exp(\mathbb{E}_q[\ln(\pi_k)]) \\ &\times \exp\left(\frac{\frac{-\lambda_1}{\lambda_2}\left(\left(\omega_{ik}^2+\Omega_{ik}\right)\left(\boldsymbol{\phi}_k^{(c)^T}\boldsymbol{\phi}_k^{(c)}+\sum_j \Phi_{kj}\right) - 2\omega_{ik}\boldsymbol{\phi}_k^{(c)^T}\mathbb{E}_q\big[\tilde{\mathbf{x}}_{i(-k)}^{(c)}\big]\right)}{2}\right) \\ q(z_{ik}=0) \;&\propto\; \exp(\mathbb{E}_q[ln(1-\pi_k)]), \text{ where} \end{aligned}$$

$$\begin{aligned} \mathbb{E}_q[\ln(\pi_k)] &= \psi\left(c_0\eta_0 + \sum_i \nu_{ik}\right) - \psi(c_0+N) \\ \mathbb{E}_q[\ln(1-\pi_k)] &= \psi\left(c_0(1-\eta_0) - \sum_i \nu_{ik} + N\right) - \psi(c_0+N). \end{aligned}$$

Then the update equation for ν_{ik} is given as

$$\nu_{ik} = \frac{q(z_{ik}=1|-)}{q(z_{ik}=1|-) + q(z_{ik}=01|-)}.$$

Update for the shared sparse weight s_{ik}. The variational distribution for s_{ik} is Gaussian parametrized with mean ω_{ik} and variance Ω_{ik}. Coordinate ascent update equation for these free parameters are

$$\begin{aligned} \Omega_{ik} &= \left(\frac{\epsilon_1}{\epsilon_2} + \frac{\lambda_1}{\lambda_2}\nu_{ik}\left(\boldsymbol{\phi}_k^{(c)^T}\boldsymbol{\phi}_k^{(c)} + \sum_j \Phi_{kj}\right)\right)^{-1}, \\ \omega_{ik} &= \frac{\lambda_1}{\lambda_2}\Omega_{ik}\nu_{ik}\boldsymbol{\phi}_k^{(c)^T}\mathbb{E}_q\big[\tilde{\mathbf{x}}_{i(-k)}^{(c)}\big]. \end{aligned}$$

Update for the kth coupled dictionary element $\mathbf{d}_k^{(c)}$. The variational distribution for the couple dictionary element $\mathbf{d}_k^{(c)}$ is Gaussian parametrized with mean $\boldsymbol{\phi}_k^{(c)}$ and variance $\boldsymbol{\Phi}_k^{(c)}$. Coordinate ascent update equation for these free variational parameters are

$$\begin{aligned} \boldsymbol{\Phi}_k^{(c)} &= \left(2v\mathbf{I}_{2v} + \frac{\lambda_1}{\lambda_2}\sum_{i=1}^{N}\left(\omega_{ik}^2+\Omega_{ik}\right)\nu_{ik}^2\right)^{-1}, \\ \boldsymbol{\phi}_k^{(c)} &= \frac{\lambda_1}{\lambda_2}\boldsymbol{\Phi}_k^{(c)}\sum_{i=1}^{N}\omega_{ik}\nu_{ik}\mathbb{E}_q\big[\tilde{\mathbf{x}}_{i(-k)}^{(c)}\big]. \end{aligned}$$

Update for the dictionary usage probabilities π_k. The variational distribution for π_k is a beta distribution parametrized with the shape parameters (τ_{k1}, τ_{k2}). Coordinate ascent update equation for these free parameters are

$$\begin{aligned} \tau_{k1} &= c_0\eta_0 + \sum_{i=1}^{N}\nu_{ik,} \\ \tau_{k2} &= N - \sum_{i=1}^{N}\nu_{ik} + c_0(1-\eta_0). \end{aligned}$$

Update for the precision γ_ϵ. The variational distribution for γ_ϵ of the observation noise $\boldsymbol{\epsilon}_i$ is a gamma distribution parametrized with (λ_1, λ_2). Coordinate ascent equation for these free parameters are

$$\begin{aligned} \lambda_1 &= c + NP, \\ \lambda_2 &= d + \frac{1}{2}\sum_{i=1}^{N}\left\{\left\|\mathbf{x}_i^{(c)} - \sum_{k=1}^{K}\boldsymbol{\phi}_k^{(c)}(\omega_{ik}\nu_{ik})\right\|_2^2\right. \\ &\quad + \sum_{k=1}^{K}\nu_{ik}\left(\omega_{ik}^2+\Omega_{ik}\right)\left(\boldsymbol{\phi}_k^{(c)^T}\boldsymbol{\phi}_k^{(c)} + \sum_j \Phi_{kj}\right) \\ &\quad \left. - \sum_{k=1}^{K}\nu_{ik}\boldsymbol{\phi}_k^{(c)^T}\boldsymbol{\phi}_k^{(c)}\omega_{ik}^2\right\}. \end{aligned}$$

Update for the precision γ_s. The variational distribution for γ_s of the sparse weights s_{ik} is a gamma distribution parametrized with (ϵ_1, ϵ_2). Coordinate ascent equation for these free parameters are

$$\begin{aligned} \epsilon_1 &= e + \frac{1}{2}NK, \\ \epsilon_2 &= f + \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{K}\left(\omega_{ik}^2+\Omega_{ik}^2\right). \end{aligned}$$

Algorithm 1 Batch VB

Sample N observations from the data. Initialize $\boldsymbol{\tau}, \boldsymbol{\nu}, \boldsymbol{\phi}, \boldsymbol{\Phi}, \boldsymbol{\omega}, \boldsymbol{\Omega}, \boldsymbol{\lambda}, \boldsymbol{\epsilon}$ using Gibbs sampler.
for $t = 1$ to T **do**
  Init. local variables $\nu_{n_k}, \boldsymbol{\omega}_{n_k}, \boldsymbol{\Omega}_{n_k}$ using Gibbs sampler.
  while relative improvement in ℓ is large **do**
    for $k = 1$ to K **do**
      for $n = 1$ to N **do**
        update $\nu_{nk}, \boldsymbol{\omega}_{nk}, \boldsymbol{\Omega}_{nk}$ by using batch updates.
      compute $\Phi_k, \boldsymbol{\phi}_k, \boldsymbol{\tau}_k, \boldsymbol{\lambda}, \boldsymbol{\epsilon}$ by batch updates.

3.2 Online Variational Inference

We now develop online variational inference. We divide the variational parameters into *global* variables and *local* variables. Global variables depend on all of the images. These are the dictionary probabilities $\boldsymbol{\pi}_k$, dictionary elements $\mathbf{d}_k$, precisions $\boldsymbol{\gamma}_\mathbf{s}$ and γ_ϵ. Local variables are the ones drawn for each image. These are the weights $\mathbf{s}_i$, binary variables $\mathbf{z}_i$. The algorithm iterates between optimizing the local variables using local (per-image) coordinate ascent, and optimizing the global

variables. This same structure is found in many Bayesian nonparametric models [23], [42].

The basic idea is to optimize Equation (3) via stochastic optimization [43]. This means we repeatedly follow noisy estimates of the gradient with decreasing step sizes ρ_t. If the step sizes satisfy $\sum_t \rho_t = \infty$ and $\sum_t \rho_t^2 < \infty$ then we will converge to the optimum of the objective. (In variational inference, we will converge to a local optimum.)

The noisy estimates of the gradient are obtained from subsampled data. We write the objective $\mathcal{L}$ as a sum over data points. Defining the distribution $g(n)$ which uniformly samples from the data, we can then write $\mathcal{L}$ as an expectation under this distribution,

$$\mathcal{L} = \sum_{n=1}^{N} \ell(\boldsymbol{\tau}, \boldsymbol{\nu}_n, \boldsymbol{\phi}, \boldsymbol{\omega}_n, \boldsymbol{\lambda}_n, \boldsymbol{\epsilon}_n, \mathbf{X}_n) \tag{4}$$

$$= N\mathbb{E}_g[\ell(\boldsymbol{\tau}, \boldsymbol{\nu}_n, \boldsymbol{\phi}, \boldsymbol{\omega}_n, \boldsymbol{\Omega}_n, \boldsymbol{\lambda}_n, \boldsymbol{\epsilon}_n, \mathbf{X}_n)]. \tag{5}$$

The gradient of the objective can be written as a similar expectation. Thus, sampling data at random and computing the gradient of ℓ_n gives a noisy estimate of the gradient.

There are two further simplifications. First, when we subsample the data we optimize the local variational parameters fully and compute the gradient of ℓ_n with respect to only the global variational parameters. Second, we use the natural gradient [44] rather than the gradient. In mean field variational inference, this simplifies the gradient step as follows. Suppose we have sampled an image n and fitted its local variational parameters given the current settings of the global variational parameters. Let $\tilde{\boldsymbol{\tau}}, \tilde{\boldsymbol{\phi}}, \tilde{\boldsymbol{\Phi}}, \tilde{\boldsymbol{\lambda}}, \tilde{\boldsymbol{\epsilon}}$ be the global variational updates from Section 3.1 as though we observed N copies of that image. (Note that these depend on its local variational parameters.) Following a noisy estimate of the natural gradient of $\mathcal{L}$ is equivalent to taking a weighted average of the current and the newly fitted global parameters, e.g.,

$$\boldsymbol{\phi} = (1 - \rho_t)\boldsymbol{\phi} + \rho_t \tilde{\boldsymbol{\phi}}. \tag{6}$$

It follows that there is no additional computational cost to optimizing the global variational parameters with stochastic optimization versus coordinate ascent.

We decrease the step-size ρ_t by $\rho_t = (\rho_0 + t)^{-\kappa}$. The learning rate parameter ρ_0 down-weights early iterations; the parameter κ controls the speed of forgetting previous values of the global variables.

Algorithm 2 Online VB with mini-batches

Define $\rho_t = (r + t)^{-\kappa}$, Initialize $\boldsymbol{\tau}, \boldsymbol{\nu}, \boldsymbol{\phi}, \boldsymbol{\Phi}, \boldsymbol{\omega}, \boldsymbol{\Omega}, \boldsymbol{\lambda}, \boldsymbol{\epsilon}$ using Gibbs sampler.
for $t = 1$ to $\frac{N}{N_S}$ **do**
  Sample N_S new observations from the data. Initialize local variables $\nu_{n_k}, \boldsymbol{\omega}_{n_k}, \boldsymbol{\Omega}_{n_k}$ using Gibbs sampler.
  while relative improvement in ℓ is large **do**
    for $k = 1$ to K **do**
      for $n_t = (t-1) \times N_S + 1$ to $t \times N_S$ **do**
        update $\nu_{n_t k}, \boldsymbol{\omega}_{n_t k}, \boldsymbol{\Omega}_{n_t k}$ by using batch updates.
      compute $\tilde{\Phi}_k, \tilde{\boldsymbol{\phi}}_k, \tilde{\boldsymbol{\tau}}_k, \tilde{\boldsymbol{\lambda}}, \tilde{\boldsymbol{\epsilon}}$ by batch VB updates as if there are N/N_S copies of the images.
  for $k = 1$ to K **do**
    update $\Phi_k, \boldsymbol{\phi}_k, \boldsymbol{\tau}_k, \boldsymbol{\lambda}, \boldsymbol{\epsilon}$ by Equation 6

The full online VB algorithm is listed in Algorithm 2. (Note that we sample the data in mini-batches, rather than one at a time. When the mini-batch size is equal to one data point, we recover the algorithm as described above.)

3.3 Initialization with MCMC

We initialize both the batch and online VB with a few iterations (e.g., 5) of MCMC.[1] This is useful for two reasons: (1) It provides a good initialization and thus faster convergence, (2) Noisy random-walks of MCMC help VB avoid low-quality local optima: at the beginning of each e-step, MCMC initializes $\mathbf{s}_i$ and $\mathbf{z}_i$ by sampling from their approximate posterior distribution, given the most recent global variables. These samples are noisy estimates of the sparse weights near their posterior means. For instance, when the factor assignment z_{ik} equals 0, the MCMC draws the sparse weight s_{ik} from the prior $\mathcal{N}(0, 1/\gamma_s)$ whereas in VB it would be exactly 0. Providing the freedom to "jiggle" gives the algorithm the opportunity to jump away from one local optimum to reach a better optimum.

4 Experiments

We use three data sets. To train, we use the set of 68 images collected from the web by [22]. We test on the natural images of [28] (20 100×100 images) and a benchmark set of images (11 images of various size)[2] used by the community to evaluate SR algorithms.[3] These images provide a rich set of HR-LR pairs.

Throughout this work, unless otherwise mentioned we use the same parameters (without any tuning): we set the SR ratio to 2 or 4 and the patch size to 8×8.[4] The hyperparameters are $c = d = e = f = 10^{-6}$ and $c_0 = 2, \eta_0 = 0.5$, these are standard uninformative priors used in, e.g., [29]. The truncation level K in BP is set to 512. Most images use fewer factors, e.g., Baboon uses 487, House 438 and Barbara 471 factors. Illuminace channel includes the information in an image that is sensitive to human eye and it is a common practice in SR literature to apply and measure the performance of algorithms only in this channel [22]. Following this, we apply all algorithms only to the illuminance channel and use Bicubic interpolation for the color layers (Cb, Cr) for all compared methods.

We study our methods with two kinds of posterior inference—Gibbs sampling (BP) and online variational inference (O-BP), which scales to larger data sets.[5] To

1. For batch VB, these MCMC samples are collected on the same subset of the data on which batch VB will process. For online VB, they are collected from the mini-batches. Scale is not an issue here because we only collect five samples.

2. The quality of reconstructions by our patch based approach does not depend on the size of the image as can be seen in results. Size of the image only matters for the time complexity.

3. We are using SR ratio = 2 or 4. For SR ratio 2, the images which do not have an even number of rows/columns are cut to have even number of rows/column to prevent any possible mismatch and error in computing PSNR in all algorithms. For instance the last column of pixels from an image of size 330×171 is excluded so the corresponding image has the size 330×170.

4. The visual results for SR ratio 4 are in the Appendix G, available in the online supplemental material, and quality does not depend on the image size.

5. The software and the visual results can be found at http://www.gungorpolatkan.com/ImageSuperresolution.

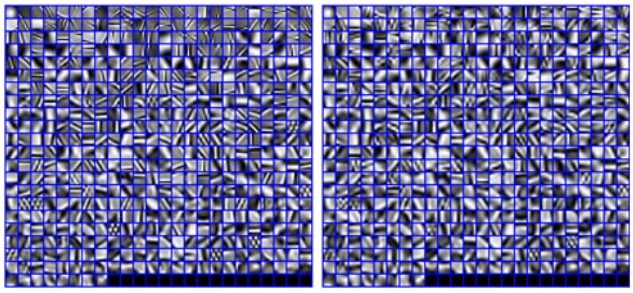


Fig. 3. Dictionary trained in batch mode on lluminance channel with SR ratio $= 2$. (Top) HR Dictionary, (Bottom) LR Dictionary. Every square represents a dictionary element and the HR-LR pairs are co-located. HR dictionary consists of sharper edges.

compare, we study both interpolation and example-based algorithms. Bicubic interpolation is the gold standard in the SR literature. We also study nearest neighbor interpolation, bilinear interpolation and sparse mixing estimation (SME) [45]. To compare with an example-based method, we use super-resolution via sparse representation (ScSR, [22]).[6] Both BP's and ScSR's dictionary learning stages use 10^5 patches sampled from the training data, however O-BP uses the whole set in an online fashion. The HR and LR dictionaries trained by our approach are shown in Fig. 3. The HR dictionary consists of sharper edges. Only few unmatched pairs occur due to aliasing.[7]

As a quantitative measure of performance we compute the signal to noise ratio (PSNR), a measure that is widely used in image recovery applications. We present the PSNR results for benchmark images in Table 1 and natural images in Table 2. These PSNR based results can be summarized as: (1) The online learning algorithm and ScSR perform similarly, (2) They both perform slightly better than the Gibbs sampler. (3) All of the example based algorithms perform better than the interpolation based techniques. We also provide PSNR results before and after post-processing in Appendix E, available in the online supplemental material. These additional results indicate that post-processing contributes 0.1dB on average to the final accuracy.

4.1 Crowdsourcing via Mechanical Turk

Though signal to noise ratio (PSNR), is a widely used metric in image recovery applications, this is not enough to measure human judgement. For this purpose, we also performed human evaluation experiments on Amazon Mechanical Turk (MTurk, http://www.mturk.com).

6. The dependent hierarchical Beta process (dHBP), another Bayesian nonparametric prior, is proposed in [30]. It removes the exchangeability assumption of beta-Bernoulli construction. This prior assumes that each observation i has a covariate $\ell_i \in \mathbb{R}^{\mathcal{L}}$. In this model, the closer the two sparse factor assignments z_i and z_j in the covariate space, the more likely they share similar dictionary elements. In [30] the authors apply dHBP to image inpainting and spiky noise removal, and show improvement over BP. We obtained preliminary results with dHBP for super-resolution. However, in this setting BP performed better.

7. When the original HR image is down sampled and interpolated to form LR images in the training set, aliasing occurs at the edges with certain thickness. This causes discrepancy between the HR and LR images at those edges. Model, which trained on this data set, learns unmatched pairs of patches corresponding to these certain unmatched edges as well. Note that this has nothing to do with the probabilistic model or inference algorithm. It is a property of the formed data set.

TABLE 1
Test Results with SR Ratio $= 2$

PSNR	Bic.	SME	ScSR1	ScSR2	BP	OBP
Baboon	23.63	23.10	24.33	24.36	24.27	**24.39**
Barbara	25.35	24.42	25.88	25.89	25.98	**25.99**
Boat	29.95	29.72	31.23	31.29	31.17	**31.31**
Camera	30.32	26.33	30.68	30.46	31.51	30.94
House	32.79	33.28	34.26	**34.31**	34.08	34.27
Peppers	31.99	33.06	33.05	33.06	32.45	**33.08**
Parthen.	28.12	27.28	29.05	**29.10**	28.96	29.06
Girl	34.76	33.98	35.57	35.58	35.62	**35.66**
Flower	40.04	39.72	41.06	41.11	41.26	**41.33**
Lena	32.83	33.57	34.47	34.54	34.56	**34.68**
Raccoon	30.95	31.73	32.39	32.43	32.43	**32.62**

PSNR for the illuminance channel is presented (the higher the better) BP: Proposed algorithm trained via Gibbs sampler, O-BP Proposed algorithm trained via Online VB, seeing more data, ScSR: Super-Resolution via Sparse Representation [22], SCSR1 represents K = 256, SCSR2 represents K = 512, NNI: Nearest neighbor interpolation, SME: Sparse Mixing Estimation [45].

The Amazon Mechanical Turk is a web interface for deploying small tasks to people, called *Turkers*. Typically an MTurk experiment works as follows: the *requesters*, people organizing the experiments and paying Turkers, prepare tasks called Human Intelligence Tasks (HITs). In our case each HIT is an image comparison problem. Once the HITs are completed, requesters can approve or reject the HITs based on their reliability measures. For instance trivial solution HITs, as we explain next, and the time spend on each HIT are frequently used measures for reliability. Approved results are acquired to be used in the analysis.

While preparing HITs, we used the natural image data. We asked Turkers to visually assess and select the better of two HR reconstructions of each image. We considered all ordered combinations of the algorithms, each equally likely, e.g., BP versus ScSR, BP versus Bicubic etc. We initially collected $42{,}807$ decisions from 208 unique Turkers. For quality control we gave test pairs in which a ground truth HR image was used, i.e., a comparison of an algorithmic reconstruction versus a true HR image. All of the judgments of the Turkers who failed to pass this test (Turkers who selected the algorithmic

TABLE 2
Test Results on Natural Images with SR Ratio $= 2$

	Bic.	NNI	ScSR1	ScSR2	BP	O-BP
N1	29.74	27.44	31.52	31.55	31.52	**31.56**
N2	29.52	27.71	31.16	**31.20**	31.17	**31.20**
N3	22.97	21.95	23.94	**24.00**	23.80	23.94
N4	21.63	20.98	22.59	**22.66**	22.38	22.41
N5	24.85	23.85	26.01	**26.06**	25.77	25.90
N6	25.34	24.61	26.20	**26.26**	26.08	26.07
N7	26.66	25.43	27.92	27.92	27.77	**27.97**
N8	26.08	24.71	27.27	**27.43**	27.01	27.26
N9	26.02	25.29	26.82	**26.89**	26.58	26.73
N10	24.79	24.07	26.23	**26.25**	25.91	26.16
N11	26.86	25.22	28.06	28.04	27.99	**28.16**
N12	28.16	26.65	29.63	29.66	29.78	**29.86**
N13	25.15	24.18	26.40	**26.36**	26.31	26.33
N14	26.82	25.98	27.99	**28.01**	27.86	27.94
N15	25.78	24.64	27.00	27.04	26.90	**27.06**
N16	27.28	25.85	28.88	**29.01**	28.83	28.96
N17	27.79	26.33	29.21	**29.24**	29.02	29.16
N18	29.13	27.75	30.38	30.41	30.25	**30.43**
N19	24.57	23.19	26.07	**26.10**	25.92	26.02
N20	22.00	21.13	23.26	23.28	23.26	**23.29**

Refer to Table 1 for abbreviations.

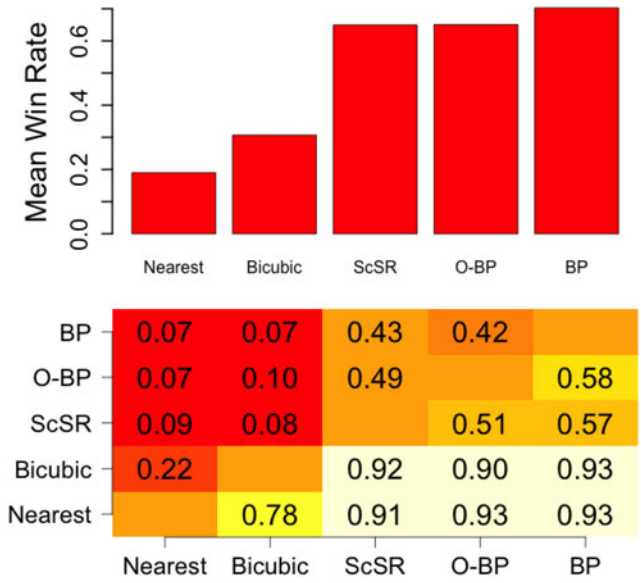


Fig. 4. Human evaluation via mechanical turk. (Top) Average win rate in one-to-one comparisons. (Bottom) Win rates for each one-to-one comparison. Each number represents the winning rate of the method in the column, e.g., 0.57 for BP versus ScSR (BP is on the column and ScSR on the row) means that on average, 0.57 of the times Turkers voted in favor of BP.

reconstruction instead of true HR) were removed. For instance if a user has been asked 200 cases and one of them was the control case and the Turker wrongly decided to an inferior reconstruction instead of a ground truth HR image, then we decided that this Turker is not reliable and we eliminated all of his/her remaining 200 comparisons. This reduced the data to $20{,}469$ decisions from 161 unique reliable Turkers. Further details of MTurk experiments can be found in Appendix I, available in the online supplemental material.

The results of the human evaluation are in Fig. 4. In the bottom table, we provide win rates for each one-to-one comparisons. Each number represents the winning rate of the method in the column. For instance, 0.93 for O-BP versus Nearest (O-BP is on the column and Nearest on the row) means that out of 100 binary comparisons of O-BP and Nearest, 93 Turkers voted in favor of O-BP. In general, we observe that example-based methods perform significantly better than interpolation-based methods. Within the example-based approaches, the models are similar. However, note that ScSR uses the first and second-order derivative filters for the LR patches. Our method does not use these features, yet we perform similarly. Further, ScSR requires setting the noise precision and the number of dictionary elements. (In ScSR comparisons, we used the parameters provided by [22].) Our method does not require to set those parameters.

In the PSNR results, ScSR and O-BP seem to perform similarly and both slightly better than BP. However, in the human evaluation we observed that BP reconstructions are found to be better. (Based on 95 percent confidence intervals, both the BP versus O-BP and BP versus ScSR results are statistically significant. The O-BP versus ScSR difference is statistically insignificant.) This shows that PSNR is not necessarily consistent with the human assessment of images [46]. Sample visual results are shown in Figs. 5, 6 and 9. (The remaining results are in the Appendix E and F, available in the online supplemental material.) Note that BP and O-BP are based on the model of Section 2.

4.2 Nonparametric Property of the Model

In this section, we demonstrate the importance of a Bayesian nonparametric method for image super-resolution. As we mentioned in Section 2.1, we use a beta-Bernoulli process for the factor assignments $\mathbf{z}_i$ that encodes which dictionary elements are activated for the corresponding observation. In

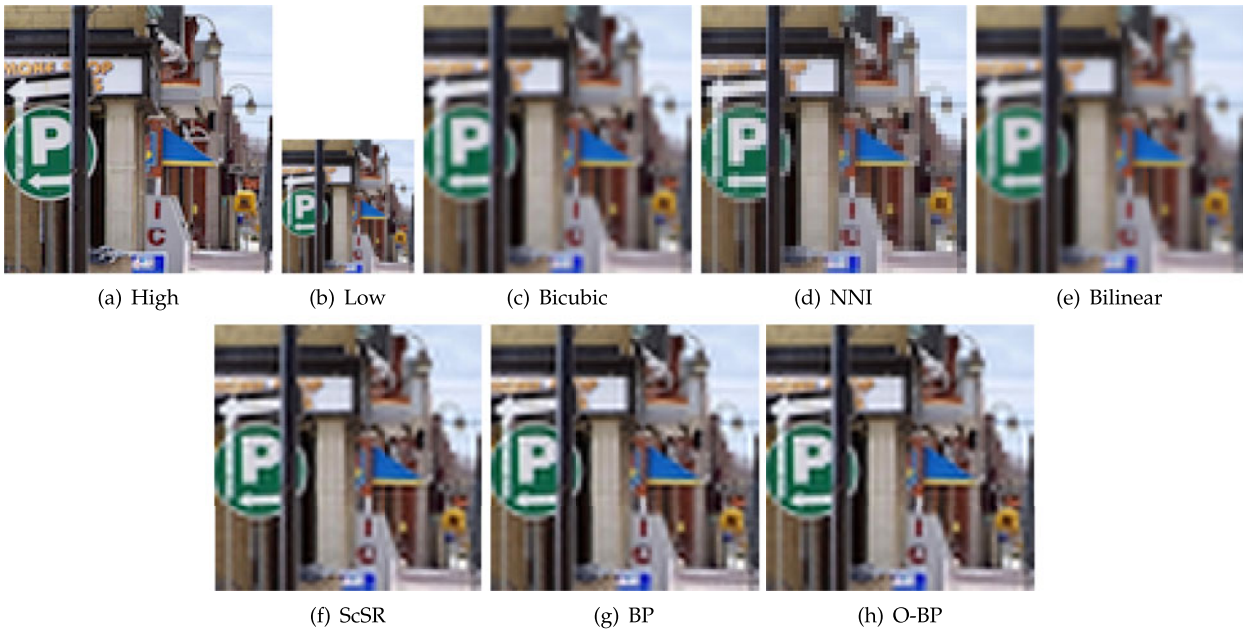


Fig. 5. Reconstruction of natural image 3. **BP**: Algorithm presented in this work trained via Gibbs sampler, **O-BP** Algorithm presented in this work trained via Online VB, **ScSR**: Super-Resolution via Sparse Representation. Example based approaches are superior to interpolation techniques, ScSR and our approach perform similarly.

Fig. 6. Reconstruction of Parthenon image. BP: Algorithm presented in this work trained via Gibbs sampler, O-BP Algorithm presented in this work trained via Online VB, ScSR: Super-Resolution via Sparse Representation. SME: Sparse Mixing Estimation [45].

the binary matrix (whose rows are the factor assignment $\mathbf{z}_i$'s), the columns with at least one active cell correspond to factors that are used.

A distinguishing characteristic of this prior is that the number of the factors to be learned is not specified a priori. Conditioned on the data, we examine the posterior distribution of the binary matrix to obtain a data-dependent distribution of how many components are needed. For the parametric ScSR, the number of dictionary elements must be set a priori. This is illustrated by the following experiment. For both model, we train on 10^4 patches, for different values of K (starting from scratch each time); for ScSR, K is the target number (which needs to be set before starting the algorithm), while for our approach, K functions as an upper bound on the number of dictionary elements (which should not be too low). Fig. 7 shows that, unlike ScSR, our approach is less sensitive to the value of K if it is sufficiently large. The Barbara image uses 700, 801 and 816 factors in our approach for K equals to $1{,}024, 2{,}048$ and $4{,}096$ respectively.

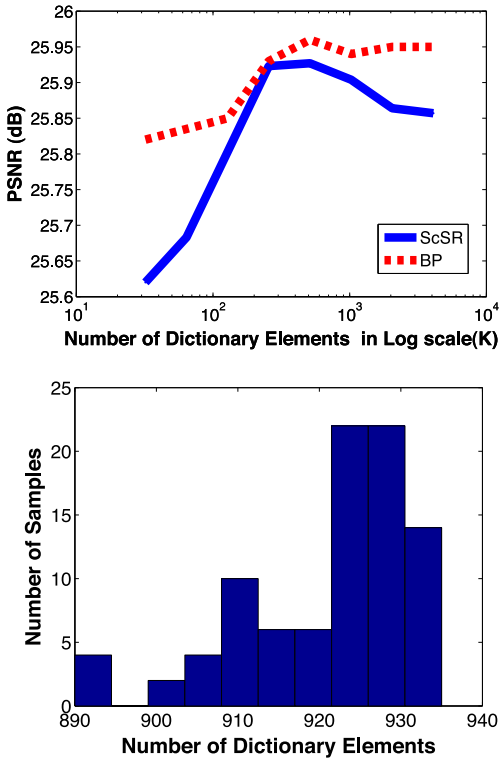


Fig. 7. Learning the number of dictionary elements from the data. (Top) PSNR of the reconstruction of the Barbara image by nonparametric BP and parametric ScSR with different number of dictionary elements. (Bottom) Histogram of the number of dictionary elements for BP when $K = 1{,}024$ over 100 samples.

4.3 Online Learning, Computational Time and Scaling

In this section, we present how well our algorithms scale. In online learning, instead of subsampling the patches during the dictionary learning stage, we use the full data set and process it segment by segment (so called "mini-batches"). We use the training data of Section 4. The learning parameters are set to $\kappa = 0.501$ and $\rho_0 = 3$.

Fig. 8 shows the evolution of the mean PSNR on the held-out natural image data set by the online and the batch algorithms as a function of the number of image patches seen (visualizations of the learned dictionaries are provided in Appendix D, available in the online supplemental material). The number of patches seen represents the computational time since both algorithms' time complexity is linear with number of observations. For online VB, the number of patches seen represents the total number of data seen after each iteration. For batch VB, this represents cumulative sum of the number of same data seen after each variational-EM iteration. Even before the second iteration of the batch VB (100K) is completed, online VB with 5K mini-batch converges—reaches to a local optima better than batch VB. This means that the online algorithm finds dictionaries at least as good as those found by the batch VB in only a fraction of the time. As also shown in Table 1, it finds high quality dictionaries. This may be because stochastic gradient is robust to local optima [47].

For dictionary training, the convergence time for online VB with 5K mini-batch size is 16 hours. In Gibbs sampling, we throw away the first 1,500 samples for the burn-in period and later collect 1,500 samples to approximate the

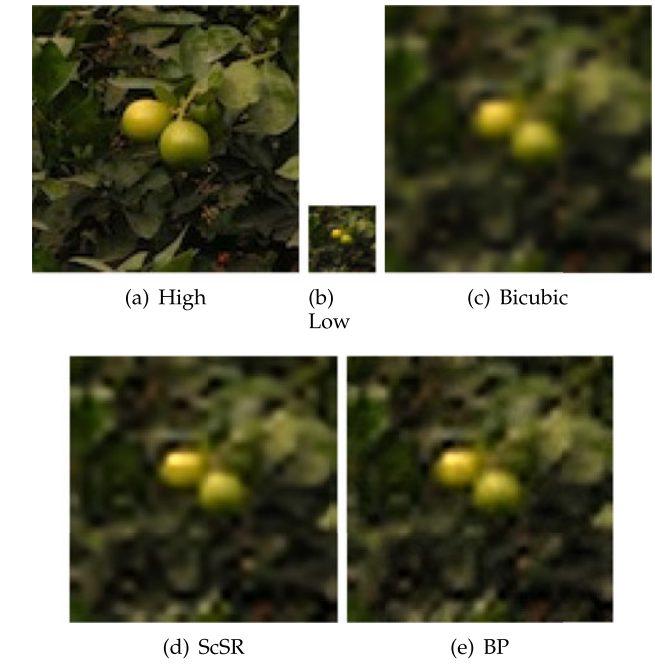


Fig. 9. (Natural image 18) test results, SR ratio = 4.

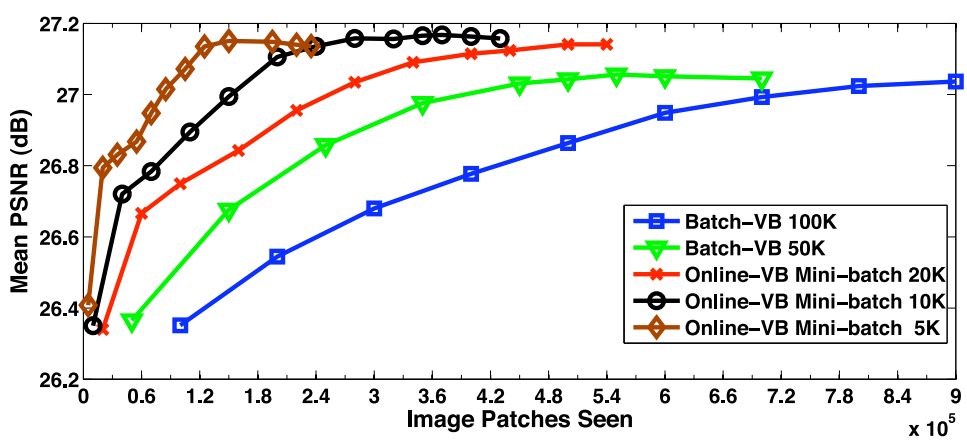


Fig. 8. Held-out prediction performances of online learning with different mini-batch sizes. Online-VB run on the whole data set is compared with the Batch-VB run on a subset of the data. The online algorithms converge much faster than the batch algorithm does.

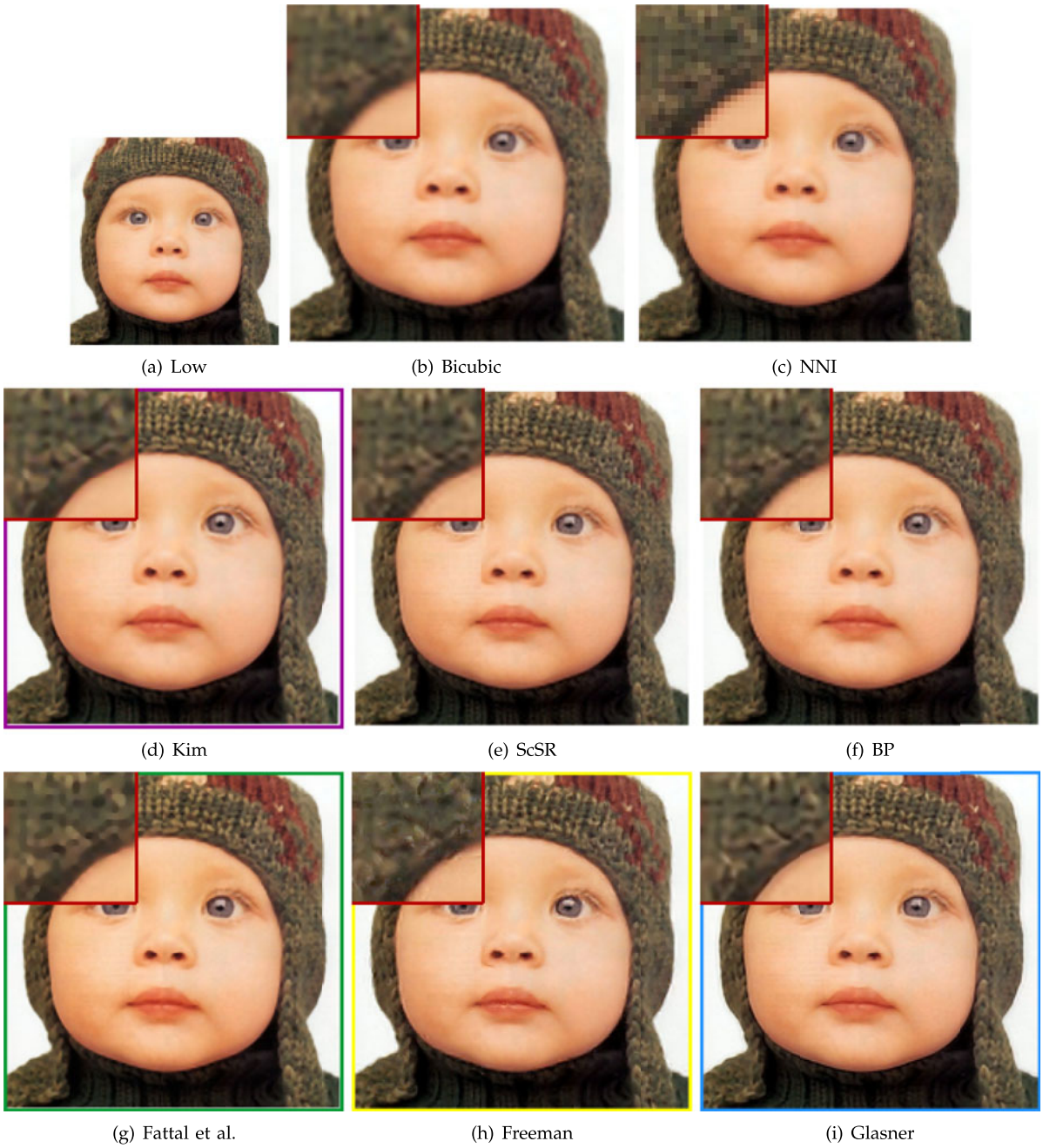


Fig. 10. (Baby) Test set results with SR ratio $= 4$.

posterior distributions. This takes approximately 50 hours on the same machine with an unoptimized Matlab implementation on 10^5 number of patches. Running Gibbs sampling same amount of time with online VB, i.e., collecting less number of samples such as 500, reduces held-out PSNR between 0.2 dB to 0.5 dB, depending on the image. This is consistent with the findings in [29]. We run ScSR with 10^5 and 1.5×10^5 training samples. The PSNR of images changes at the level of 0.001 dB. For instance for the Lena image, 10^5-trained model gave 34.541 dB and 1.5×10^5-trained model gave 34.548 dB.[8]

8. Running time for ScSR is approximately 10 hours for 10^5 patches and 16 hours for 1.5×10^5 patches. However, ScSR uses special linear algebra packages, so we believe computing speed comparison between the algorithms explained here and ScSR is not an ideal one.

4.4 Visual Comparisons with Other Algorithms

In Fig. 10, we provide visual comparisons with algorithms of Freeman et al. [15], Kim and Kwon [16], Glasner et al. [20], Fattal [48] using the SR ratio 4. (the images courtesy of [15], [16], [20], [48].) More results can be found in Appendix H, available in the online supplemental material. Since the HR images do not exist, we cannot compute the pSNR for these comparisons. As we observe, Glasner et al. [20] provides very sharp edges by artificially enhancing them. However, this makes images unrealistic (looking like graphically rendered). Sparse coding techniques are not as good with edges but performs well in textures and look more realistic. Glasner et al. can be used to boost the edge sharpness of the sparse coding methods. Example based sparse coding techniques (ScSR and our method BP) allow any single-image

SR algorithm as a pre-processing step. Instead of bicubic interpolation (see Fig. 1), one can use Glasner et al. with sparse coding and the dictionary training will learn the relationship between the HR and Glasner et al. (In ScSR and BP, it learns the relationship between the HR and Bicubic interpolation and it boosts the bicubic's performance.).

5 Discussion

We developed a new model for super-resolution based on Bayesian nonparametric factor analysis, and new algorithms based on Gibbs sampling and online variational inference. With online training, our algorithm scales to very large data sets. We evaluated our method against a leading sparse coding technique [22] and other state-of-the-art methods. We evaluated both with traditional PSNR and by devising a large scale human evaluation. This is a new real-world application of online variational methods.

The choice of the inference algorithm depends on the usage. Our results suggest that with more computation time Gibbs sampling performs slightly better (based on human evaluation). If speed is important, our online algorithms can be used without much loss.

Regarding the evaluation metric, the standard in image analysis has been signal-to-noise ratio. However, its practical relevance has been questioned [46]. The human eye is sensitive to details which are not always captured in this metric, and that is why we ran a human evaluation. Our experiments show that the signal-to-noise ratio is not necessarily consistent with human judgement. Though online variational inference does its job quite well in terms of held-out PSNR, this metric does not perfectly correlate with how human eyes judge quality. It is interesting that Gibbs samplers give poorer PSNR but are better with respect to human evaluation.

As future work, our approach can be used as a building block in more complicated probabilistic models. For example, our approach could be developed into a time series to perform SR on video or a hierarchical model that explicitly models images as collections of patches. With the latter there is a potential improvement to the model if reconstructions can be forced to match the low resolution input within the model instead of a post-processing step.

References

[1] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006.

[2] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. Image Proc.*, vol. 15, no. 12, pp. 3736–3745, Dec. 2006.

[3] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Nonlocal sparse models for image restoration," in *Proc. Int. Conf. Comput. Vis.*, 2009, pp. 2272–2279.

[4] J. Mairal, M. Elad, and G. Sapiro, "Sparse representation for color image restoration," *IEEE Trans. Image Process.*, vol. 17, no. 1, pp. 53–69, Jan. 2008.

[5] M. Ranzato, C. S. Poultney, S. Chopra, and Y. Lecun, "Efficient learning of sparse representations with an energy-based model," in *Proc. Neural Inf. Process. Syst.*, 2006, pp. 1137–1144.

[6] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 210–227, Feb. 2009.

[7] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Supervised dictionary learning," *Comput. Res. Repository*, vol. abs/0809.3, pp. 1033–1040, 2008.

[8] J. Mairal, G. Sapiro, and M. Elad, "Learning multiscale sparse representations for image and video restoration," *Multiscale Model. Simul.*, vol. 7, pp. 214–241, 2008.

[9] A. M. Bruckstein, D. L. Donoho, and M. Elad, "From sparse solutions of systems of equations to sparse modeling of signals and images," *Siam Rev.*, vol. 51, pp. 34–81, 2009.

[10] E. J. Cands and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?" *IEEE Trans. Inf. Theory*, vol. 52, no. 12, pp. 5406–5425, Dec. 2006.

[11] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proc. Int. Conf. Mach. Learn.*, 2009, pp. 87–696.

[12] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, "Self-taught learning: Transfer learning from unlabeled data," in *Proc. Int. Conf. Mach. Learn.*, 2007, pp. 759–766.

[13] S. Farsiu, M. Robinson, M. Elad, and P. Milanfar, "Fast and robust multiframe super resolution," *IEEE Trans. Image Process.*, vol. 13, no. 10, pp. 1327–1344, Oct. 2004.

[14] M. Tipping and C. Bishop, "Bayesian image super-resolution," in *Proc. Adv. Neural Inf. Process. Syst.*, 2003, pp. 1303–1310.

[15] W. T. Freeman, T. R. Jones, and E. C. Pasztor, "Example-based super-resolution," *IEEE Comput. Graph. Appl.*, vol. 22, no. 2, pp. 56–65, Mar./Apr. 2002.

[16] K. I. Kim and Y. Kwon, "Single-image super-resolution using sparse regression and natural image prior," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 6, pp. 1127–1133, Jun. 2010.

[17] J. Sun, N. Zheng, H. Tao, and H. Shum, "Image hallucination with primal sketch priors," in *Proc. Comput. Vis. Pattern Recog.*, 2003, pp. 729–736.

[18] Y. HaCohen, R. Fattal, and D. Lischinski, "Image upsampling via texture hallucination," in *Proc. IEEE Int. Conf. Comput. Photography*, 2010, pp. 1–8.

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

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

[21] H. He and W.-C. Siu, "Single image super-resolution using Gaussian process regression," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2011, pp. 449–456.

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

[23] F. Doshi-Velez, K. T. Miller, J. Van Gael, and Y. W. Teh, "Variational inference for the indian buffet process," in *Proc. Artif. Intell. Statist.*, 2009, pp. 137–144.

[24] D. Knowles and Z. Ghahramani, "Infinite sparse factor analysis and infinite independent components analysis," in *Proc. Ind. Compon. Anal. Signal Separation*, 2007, pp. 381–388.

[25] T. L. Griffiths and Z. Ghahramani, "Infinite latent feature models and the Indian buffet process," in *Proc. Adv. Neural Inf. Process. Syst.*, 2005, pp. 475–482.

[26] R. Thibaux and M. I. Jordan, "Hierarchical beta processes and the Indian buffet process," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2007, pp. 564–571.

[27] S. Ghosh, A. B. Ungureanu, E. B. Sudderth, D. M. Blei, and M. Stanley, "Spatial distance dependent Chinese restaurant processes for image segmentation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 1–9.

[28] B. Chen, G. Polatkan, G. Sapiro, D. Dunson, and L. Carin, "The hierarchical beta process for convolutional factor analysis and deep learning," in *Proc. Int. Conf. Mach. Learn.*, Jun. 2011, pp. 361–368.

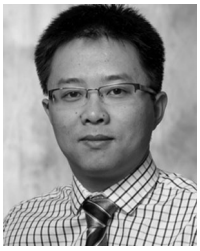
[29] M. Zhou, H. Chen, J. Paisley, L. Ren, G. Sapiro, and L. Carin, "Non-parametric Bayesian dictionary learning for sparse image representations 1," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 2295–2303.

[30] M. Zhou, H. Yang, G. Sapiro, D. Dunson, and L. Carin, "Dependent hierarchical beta process for image interpolation and denoising," in *Proc. Artif. Intell. Statist.*, 2011, pp. 883–891.

[31] S. Ghosh and E. B. Sudderth, "Nonparametric learning for layered segmentation of natural images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2012, pp. 2272–2279.

[32] J. J. Kivinen, E. B. Sudderth, and M. I. Jordan, "Learning multiscale representations of natural scenes using Dirichlet processes," presented at the IEEE 11th Int. Conf. Computer Vision, Rio de Janeiro, Brazil, 2007..
[33] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods*. New York, NY, USA: Springer, 2004.
[34] M. Hoffman, D. Blei, J. Paisley, and C. Wang, "Stochastic variational inference," *J. Mach. Learn. Res.*, vol. 14, pp. 1303–1347, 2013.
[35] J. Paisley and L. Carin, "Nonparametric factor analysis with beta process priors," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, 2009, pp. 777–784.
[36] T. L. Griffiths and Z. Ghahramani, "The Indian buffet process: An introduction and review," *J. Mach. Learn. Res.*, vol. 12, pp. 1185–1224, 2011.
[37] M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul, "An introduction to variational methods for graphical models," *Mach. Learn.*, vol. 37, pp. 183–233, 1999.
[38] M. Sato, "On-line model selection based on the variational Bayes," *Neural Comput.*, vol. 13, pp. 1649–1681, 2001.
[39] M. D. Hoffman, D. M. Blei, and F. Bach, "Online learning for latent Dirichlet allocation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 856–864.
[40] C. Wang, J. Paisley, and D. Blei, "Online variational inference for the hierarchical Dirichlet process," in *Proc. Artif. Intell. Statist.*, 2011, pp. 752–760.
[41] C. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.
[42] Y. Teh, M. Jordan, M. Beal, and D. Blei, "Hierarchical Dirichlet processes," *J. Amer. Statist. Assoc.*, vol. 101, pp. 1566–1581, 2006.
[43] H. Robbins and S. Monro, "A stochastic approximation method," *Ann. Math. Statist.*, vol. 22, no. 3, pp. 400–407, 1951.
[44] S. Amari, "Natural gradient works efficiently in learning," *Neural Comput.*, vol. 10, pp. 251–276, 1998.
[45] S. Mallat and G. Yu, "Super-resolution with sparse mixing estimators," *IEEE Trans. Image Process.*, vol. 19, no. 11, pp. 2889–2900, Nov. 2010.
[46] Z. Wang and A. Bovik, "Mean squared error: Love it or leave it? A new look at signal fidelity measures," *IEEE Signal Process. Mag.*, vol. 26, no. 1, pp. 98–117, Jan. 2009.
[47] L. Bottou, *Online Learning and Stochastic Approximations*. Cambridge, U.K.: Cambridge Univ. Press, 1998.
[48] R. Fattal, "Image upsampling via imposed edge statistics," *ACM Trans. Graph.*, vol. 26, p. 95, 2007.

Güngör Polatkan received the BS degree from the Electrical and Electronics Engineering Department at Boğaziçi University in 2007, and the MA and PhD degrees from the Electrical Engineering Department at Princeton University in 2009 and 2012, respectively. He is currently a data scientist at Twitter Inc. His main research interests include machine learning, probabilistic modeling and applications into recommendation systems, dictionary learning, and topic modeling.



Mingyuan Zhou received the BSc degree from Nanjing University, Nanjing, China, in 2005, the MSc degree from the Chinese Academy of Sciences, Beijing, China, in 2008, and the PhD degree in electrical and computer engineering from Duke University, Durham, NC, in May 2013. He is an assistant professor of statistics at the University of Texas at Austin, TX. He is with the Department of Information, Risk, and Operations Management at the McCombs School of Business and is also a core faculty member in the Department of Statistics and Data Sciences. His current research interests lie in Bayesian statistics and machine learning. He is a member of the IEEE.

Lawrence Carin (SM'96-F'01) received the BS, MS, and PhD degrees in electrical engineering from the University of Maryland, College Park, in 1985, 1986, and 1989, respectively. In 1989, he joined the Electrical Engineering Department at Polytechnic University (Brooklyn) as an assistant professor, and became an associate professor there in 1994. In September 1995, he joined the Electrical Engineering Department at Duke University, where he is currently the William H. Younger professor of Engineering, and is currently serving as the Department chair. He is a co-founder of Signal Innovations Group, Inc. (SIG), a small business, where he serves as the director of Technology. His current research interests include signal processing, sensing, applied statistics and machine learning. He has published more than 250 peer-reviewed papers, and he is a member of the Tau Beta Pi and Eta Kappa Nu honor societies. He is a fellow of the IEEE.

David Blei received the PhD degree from U.C. Berkeley in 2004 and was a postdoctoral fellow at Carnegie Mellon University. He is an associate professor of computer science at Princeton University. His research focuses on probabilistic topic models, Bayesian nonparametric methods, and approximate posterior inference. He works on a variety of applications, including text, images, music, social networks, and scientific data.

Ingrid Daubechies received the BS and PhD degrees from the Vrije Universiteit Brussel (Belgium), both in theoretical physics. From 1987 until 1994, she was a member of Technical Staff at AT&T Bell Laboratories; she has held academic positions at the University of Michigan, Rutgers University and Princeton University. Since January 2011, she has been on the faculty of the Mathematics Department at Duke University. She is a fellow of the IEEE, an AAAS fellow, and a member of the National Academy of Sciences.