Fast and Accurate Image Upscaling with Super-Resolution Forests

Samuel Schulter[†] Christian Leistner[‡] Horst Bischof[†]

[†]Graz University of Technology
Institute for Computer Graphics and Vision
{schulter,bischof}@icg.tugraz.at

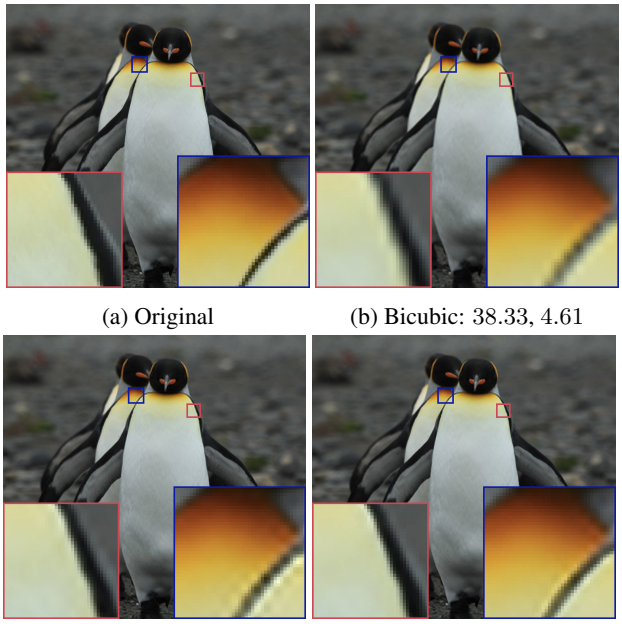
[‡]Microsoft Photogrammetry
Austria
christian.leistner@microsoft.com

Abstract

The aim of single image super-resolution is to reconstruct a high-resolution image from a single low-resolution input. Although the task is ill-posed it can be seen as finding a non-linear mapping from a low to high-dimensional space. Recent methods that rely on both neighborhood embedding and sparse-coding have led to tremendous quality improvements. Yet, many of the previous approaches are hard to apply in practice because they are either too slow or demand tedious parameter tweaks. In this paper, we propose to directly map from low to high-resolution patches using random forests. We show the close relation of previous work on single image super-resolution to locally linear regression and demonstrate how random forests nicely fit into this framework. During training the trees, we optimize a novel and effective regularized objective that not only operates on the output space but also on the input space, which especially suits the regression task. During inference, our method comprises the same well-known computational efficiency that has made random forests popular for many computer vision problems. In the experimental part, we demonstrate on standard benchmarks for single image super-resolution that our approach yields highly accurate state-of-the-art results, while being fast in both training and evaluation.

1. Introduction

Single image super-resolution (SISR) [18, 22] is an important computer vision problem with many interesting applications, ranging from medical and astronomical imaging to law enforcement. The task in SISR is to generate a visually pleasing high-resolution output from a single low-resolution input image. Although the problem is inherently ambiguous and ill-posed, simple linear, bicubic or Lanzcos interpolations [15] are often used to reconstruct the high-resolution image. These methods are extremely fast but typically yield poor results as they rely on simple smoothness assumptions that are rarely fulfilled in real images.



(a) Original (b) Bicubic: 38.33, 4.61

(c) BPJDL [23]: 40.79, 4.57 (d) RFL: 41.55, 5.77

Figure 1: Our super-resolution approach (RFL) compares favorably with related work (upscaling factor is 3). The values define the PSNR and IFC scores, respectively.

More powerful methods rely on statistical image priors [17, 18] or use sophisticated machine learning techniques [14, 37] to learn a mapping from low- to high-resolution patches. Among the best performing algorithms are sparse-coding or dictionary learning approaches, which assume that natural patches can be represented using sparse activations of dictionary atoms. In particular, coupled dictionary learning approaches [35, 38, 39] achieved state-of-the-art results for SISR. Recently, Timofte *et al*. [32] highlighted the computational bottlenecks of these methods and proposed to replace the single dictionary with many smaller ones, thus avoiding the costly sparse-coding step during inference. This leads to a vast computational speed-up while keeping the same accuracy as previous methods.

In this work, we show that the efficient formulation from [32] can be naturally casted as a locally linear multivariate regression problem and that random forests [2, 6, 10] nicely fit into this framework. Random forests are highly non-linear learners that are usually extremely fast during both learning and evaluation. They can easily handle high-dimensional noisy inputs, which has led to their broad dissemination in many computer vision domains. We propose a novel regularized objective function optimized during tree growing that operates not only on the output label domain but also on the input data domain. The goal of the objective is to cluster data samples that have high similarity in both domains. This eases the task for the locally linear regressors that are learned in the leaf nodes of the trees and yields higher quality results for SISR (see Figure 1). We thus coin the approach super-resolution forests (SRF). Inference of SRF is also fast as patches can be routed to leaf nodes using only a few simple feature look-ups plus evaluating a linear regressor.

Our experiments demonstrate the effectiveness of super-resolution forests on different benchmarks, where we present state-of-the-art results. Besides comparing with previous SISR methods including dictionary-based as well as other direct regression-based approaches, we investigate different variants of our random forest model and validate the benefits of the novel regularized objective. Furthermore, we also evaluate the influence of several important parameters of our approach.

2. Related Work

The task of upscaling images, *i.e.*, image super-resolution, has a long history in the computer vision community. While many approaches assume to have multiple views of the scene, either from a stereo setup or via temporal aggregation, this work focuses on single image super-resolution (SISR). We limit the related work to recent dictionary learning and regression-based literature and refer interested readers to a comprehensive survey paper [33].

Dictionary learning approaches for SISR typically build upon sparse coding [27]. Yang *et al*. [38] were one of the first who used a sparse coding formulation to learn dictionaries for the low- and high-resolution domain that are coupled via a common encoding. Zeyde *et al*. [39] improved upon [38] by using a stronger image representation. Furthermore, [39] employed kSVD and orthogonal matching pursuit for sparse coding [1], as well as a direct regression of the high-resolution dictionary for faster training and inference. While both approaches assume strictly equal encodings in the low- and high-resolution domains, Wang *et al*. [35] relaxed this constraint to a linear dependence, which they term semi-coupled. However, all these approaches are still quite slow during both training as well as inference because a sparse encoding is required.

Very recently, two different approaches came up to approximate sparse coding, aiming at much faster inference for different applications [26, 32]. The consent in both works is to replace the single large dictionary and ℓ_1-norm regularization with many smaller sub-dictionaries with ℓ_2-norm regularization. Thus, finding encoding vectors or reconstruction vectors is a quadratic problem for each sub-dictionary and can be solved in closed-form, leading to fast inference. The main effort during inference is shifted to finding the best sub-dictionary. While [26] looks for a sub-dictionary with a small enough reconstruction error, Timofte *et al*. [32] selects the sub-dictionary with highest correlation to the input signal. We provide more details on [32] in Section 3.

As mentioned in the introduction, a different approach to single image super-resolution is to directly learn a mapping from the low- to the high-resolution domain via regression. Yang and Yang [37] propose to tackle the complex regression problem by splitting the input data space and learning simple regressors for each data cell. Dai *et al*. [11] follows a similar approach but jointly learns the separation into cells and the regressors. While the basic idea is similar to our work, both approaches have a key problem: One has to manually define the number of clusters and thus regressors, which directly influences the trade-off between upscaling quality and inference time. However, finding the appropriate number of clusters is typically data-dependent. By using random forests on the other hand, (i) the granularity of the data cells in the input space is found automatically and (ii) the effective size of small linear regressors is typically much larger without increasing inference time.

Another recent work builds on convolutional neural networks to learn the complex mapping function [14], which is termed SRCNN. Our experiments show that our random forest approach can learn a more accurate mapping and outperforms SRCNN. Moreover, our super-resolution forests can be trained within minutes on a single CPU core, while SRCNN takes 3 days to train even with the aid of GPUs.

Because we base our approach on random forests [2, 6, 10], we also have to mention that the training procedure can be easily parallelized on multiple CPU cores, which reduces the training time even further. The highly efficient training (and also inference) is one of the key advantages of this model. Random forests is a very effective machine learning approach [7] and has been successfully applied to many different computer vision tasks. Examples include human pose estimation [21, 30], object detection [19], facial fiducial detection [9, 12], edge prediction [13, 25], semantic image labeling [24, 28], and many more. Random forests are very flexible in the sense that the input and output domains can take almost arbitrary forms. A very general formulation is presented by Dollár and Zitnick [13] where any structured label can be predicted as long as an appropriate distance

function can be formulated for that domain. In contrast, we have a multivariate regression problem where many possible choices for the objective function exist, which we discuss in Section 4. We refer the reader to [10] for a very detailed investigation of random forests.

3. Coupled Dictionary Learning

In recent years, the predominant approaches for dictionary-based single image super-resolution were based on coupled dictionary learning. The first and most generic formulation was proposed by Yang *et al*. [38]. We denote the set of N samples from the low-resolution domain $\mathtt{X_L} \in \mathbb{R}^{D_\mathrm{L} \times N}$ and from the high-resolution domain $\mathtt{X_H} \in \mathbb{R}^{D_\mathrm{H} \times N}$, each column corresponding to one sample $\mathtt{x_L}$ and $\mathtt{x_H}$, respectively. Throughout the paper, we use the notation $\mathtt{X_L}$ and $\mathtt{X_H}$ for both sets and data matrices. Then, the coupled dictionary learning problem is defined as

$$\min_{\mathtt{D_L},\mathtt{D_H},\mathtt{E}} = \frac{1}{D_\mathrm{L}} \|\mathtt{X_L} - \mathtt{D_L E}\|_2^2 + \frac{1}{D_\mathrm{H}} \|\mathtt{X_H} - \mathtt{D_H E}\|_2^2 + \Gamma(\mathtt{E}), \quad (1)$$

where $\mathtt{D_L} \in \mathbb{R}^{D_\mathrm{L} \times B}$ and $\mathtt{D_H} \in \mathbb{R}^{D_\mathrm{H} \times B}$ are the low- and high-resolution dictionaries, respectively. The common encoding $\mathtt{E} \in \mathbb{R}^{B \times N}$ couples both domains. The regularization $\Gamma(\mathtt{E})$ is typically a sparsity inducing norm on the columns of $\mathtt{E}$, *e.g*., the ℓ_0 or ℓ_1 norm.

As mentioned in Section 2, Zeyde *et al*. [39] improved upon [38] with some tweaks on the learning scheme. One main difference to [38] is that they first learn the low-resolution dictionary $\mathtt{D_L}$ via sparse coding and compute the encoding $\mathtt{E}$. Then, they fix $\mathtt{D_L}$ and $\mathtt{E}$ in (1) and directly learn the high-resolution dictionary $\mathtt{D_H}$, which is a simple least squares problem. This combination of low- and high-resolution dictionaries is the basis and the first step for anchored neighborhood regression (ANR) [32] and A+ [31].

The key observation in [32] is that one can replace the sparse coding during inference, which is time-consuming, by pre-selecting so-called anchored points $\bar{\mathtt{D}}_\mathtt{L}^i \in \mathtt{D_L}$ for each dictionary atom $\mathtt{d}^i$ in $\mathtt{D_L}$ already during training. During inference, a new data sample $\mathtt{x}$ selects the closest dictionary atom $\mathtt{d}^*$ in $\mathtt{D_L}$ according to the angular distance and uses only the pre-defined anchor points $\bar{\mathtt{D}}_\mathtt{L}^*$ for computing the encoding $\mathtt{e}$. Thus, no sparse coding with ℓ_1-norm regularizer is required as the dictionary atoms are pre-selected. The problem of finding an encoding $\mathtt{e}$ for sample $\mathtt{x}$ reduces to a least squares problem and can be computed in closed-form

$$\mathtt{e} = \arg\min_{\mathtt{e}} \|\mathtt{x} - \bar{\mathtt{D}}_\mathtt{L}^* \mathtt{e}\|_2^2 = (\bar{\mathtt{D}}_\mathtt{L}^{*\top} \bar{\mathtt{D}}_\mathtt{L}^*)^{-1} \bar{\mathtt{D}}_\mathtt{L}^{*\top} \mathtt{x} = \mathtt{P}^* \mathtt{x} \; . \quad (2)$$

All matrices $\mathtt{P}^i$ for each dictionary atom can be pre-computed in the training phase. Furthermore, using the corresponding high-resolution anchored points $\bar{\mathtt{D}}_\mathtt{H}^*$, the reconstructed high-resolution sample $\hat{\mathtt{x}}_\mathrm{H}$ can be computed as

$$\hat{\mathtt{x}}_\mathrm{H} = \bar{\mathtt{D}}_\mathtt{H}^* \cdot \mathtt{e} = \bar{\mathtt{D}}_\mathtt{H}^* \cdot \mathtt{P}^* \cdot \mathtt{x_L} = \mathtt{W}^* \cdot \mathtt{x_L} \; . \quad (3)$$

As can be seen, the problem reduces to selecting an atom from $\mathtt{D_L}$ and applying the pre-computed mapping $\mathtt{W}^*$. While ANR [32] is limited to selecting close anchor points (angular distance) from the learned dictionary $\mathtt{D_L}$, A+ [31] shows that using a much larger corpus from the full training set yields more accurate predictors and better results.

Both works [32, 31] still make a detour via sparse-coded dictionaries. However, one can also see this problem more directly as non-linear regression. In this work, we thus reformulate the problem of learning the mapping from the low- to the high-resolution domain via locally linear regression. To do so, we realize that the mapping function $\mathtt{W}$ in (3) depends on the data $\mathtt{x_L}$, which yields

$$\hat{\mathtt{x}}_\mathrm{H} = \mathtt{W}(\mathtt{x_L}) \cdot \mathtt{x_L} \; . \quad (4)$$

Now, training only requires to find the function $\mathtt{W}(\mathtt{x_L})$. Two recent works [14, 37], already mentioned in Section 2, also attack the super-resolution task with a direct regression formulation to learn $\mathtt{W}(\mathtt{x_L})$. In the next section, we detail our approach and show how random forests [2, 6, 10] can be used as an effective algorithm to learn this mapping, which has some critical benefits over previous works.

4. Random Forests for Super-Resolution

In this section, we investigate the problem of learning the data-dependent function $\mathtt{W}(\mathtt{x_L})$ defined in Equation (4). Assuming a regular squared loss the learning problem can be formulated as

$$\arg\min_{\mathtt{W}(\mathtt{x_L})} \sum_{n=1}^{N} \|\mathtt{x}_\mathrm{H}^n - \mathtt{W}(\mathtt{x}_\mathrm{L}^n) \cdot \mathtt{x}_\mathrm{L}^n\|_2^2 \; . \quad (5)$$

We can generalize this model with different basis functions $\phi(\mathtt{x})$ resulting in

$$\arg\min_{\mathtt{W}_j(\mathtt{x_L}) \; \forall j} \sum_{n=1}^{N} \|\mathtt{x}_\mathrm{H}^n - \sum_{j=0}^{\gamma} \mathtt{W}_j(\mathtt{x}_\mathrm{L}^n) \cdot \phi_j(\mathtt{x}_\mathrm{L}^n)\|_2^2 \; , \quad (6)$$

where the goal is to find the data-dependent regression matrices $\mathtt{W}_j(\mathtt{x_L})$ for each of the $\gamma+1$ basis functions. While the standard choice for $\phi_j(\mathtt{x})$ is the linear basis function, *i.e*., $\phi_j(\mathtt{x}) = \mathtt{x}$, one can also opt for polynomial ($\phi_j(\mathtt{x}) = \mathtt{x}^{[j]}$, where $^{[\cdot]}$ denotes the point-wise power operator) or radial basis functions ($\phi_j(\mathtt{x}) = \exp\left(\frac{\|\mathtt{x}-\mu_j\|^2}{\sigma_j}\right)$, where μ_j and σ_j are parameters of the basis function) [5]. We evaluate different choices of $\phi_j(\mathtt{x})$ in our experiments. Either way, the objective stays linear in the parameters to be learned, if the data-dependence is neglected for a moment. The question remaining is how to define the data-dependence of $\mathtt{W}$ on $\mathtt{x_L}$.

In this work we employ random forests (RF) [2, 6, 10] to create the data dependence. RF are ensembles of T binary trees $\mathcal{T}_t(\mathtt{x}) : \mathcal{X} \rightarrow \mathcal{Y}$, where $\mathcal{X} \in \mathbb{R}^D$ is the input feature

space and $\mathcal{Y}$ is the output label space. The output space $\mathcal{Y}$ always depends on the task at hand. For classification, $\mathcal{Y} = \{1, \dots, C\}$ with C being the number of classes. For our task, *i.e.*, multi-variate regression with D_H dimensions, we can define $\mathcal{Y} = \mathbb{R}^{D_\mathrm{H}}$ (and $\mathcal{X} = \mathbb{R}^{D_\mathrm{L}}$). At this point, we postpone the description of the training procedure of our super-resolution forests to Section 4.1 and assume that the structure of each single tree already exists.

Each tree $\mathcal{T}_t$ independently separates the data space into disjoint cells, which correspond to the leaf nodes. Using more than one tree, *i.e.*, forests, leads to overlapping cells. This separation defines our data dependence for $\mathtt{W}(\mathtt{x}_\mathrm{L})$ and each leaf l can learn a linear model

$$m_l(\mathtt{x}_\mathrm{L}) = \sum_{j=0}^{\gamma} \mathtt{W}_j^l \cdot \phi_j(\mathtt{x}_\mathrm{L}) \; . \tag{7}$$

Finding all $\mathtt{W}_j^l$ requires solving the regularized least squares problem which can be done in closed-form as $\mathtt{W}^{l\top} = (\Phi(\mathtt{X}_\mathrm{L})^\top \Phi(\mathtt{X}_\mathrm{L}) + \lambda \mathtt{I})^{-1} \Phi(\mathtt{X}_\mathrm{L})^\top \cdot \mathtt{X}_\mathrm{H}$, where we stacked all $\mathtt{W}_j^l$ and the data into matrices $\mathtt{W}^l$, $\Phi(\mathtt{X}_\mathrm{L})$, and $\mathtt{X}_\mathrm{H}$. The regularization parameter λ has to be specified by the user. Now, we can easily see the relation to the locally linear regression model from Equation (4). Because we average predictions over all T trees during inference, the data-dependent mapping matrix $\mathtt{W}(\mathtt{x}_\mathrm{L})$ is modeled as

$$\hat{\mathtt{x}}_\mathrm{H} = m(\mathtt{x}_\mathrm{L}) = \mathtt{W}(\mathtt{x}_\mathrm{L}) \cdot \mathtt{x}_\mathrm{L} = \frac{1}{T} \sum_{t=1}^{T} m_{l(t)}(\mathtt{x}_\mathrm{L}) \; , \tag{8}$$

where $l(t)$ is the leaf in tree t the sample $\mathtt{x}_\mathrm{L}$ is routed to.

We also note that the recently presented filter forests [16] have similar leaf node models $m_l(\mathtt{x}_\mathrm{L})$. They use the linear basis function and learn the model for a single output dimension within the context of image denoising. Our leaf node model from (7) can be considered as a more general formulation. Another option for $m_l(\mathtt{x}_\mathrm{L})$, which is typically used in random regression forests, is a constant model, *i.e.*, $m_l(\mathtt{x}_\mathrm{L}) = \hat{m} = \text{const}$. The constant prediction $\hat{m}$ is often the empirical mean over the labels $\mathtt{x}_\mathrm{H}$ of the training samples falling into that leaf. We evaluate different versions of $m_l(\mathtt{x}_\mathrm{L})$ in our experiments.

4.1. Learning the Tree Structure

All trees in the super-resolution forests (SRF) are trained independently from each other with a set of N training samples $\{\mathtt{x}_\mathrm{L}^n, \mathtt{x}_\mathrm{H}^n\} \in \mathcal{X} \times \mathcal{Y}$. Training a single random tree involves recursively splitting the training data into disjoint subsets by finding splitting functions

$$\sigma(\mathtt{x}_\mathrm{L}, \Theta) = \begin{cases} 0 & \text{if } r_\Theta(\mathtt{x}_\mathrm{L}) < 0 \\ 1 & \text{otherwise} \end{cases} \; , \tag{9}$$

for all internal nodes in $\mathcal{T}_t$. Splitting starts at the root node and continues in a greedy manner down the tree until a maximum depth ξ_max is reached and a leaf is created. The values of Θ define the response function $r_\Theta(\mathtt{x}_\mathrm{L})$. It is often defined as $r_\Theta(\mathtt{x}_\mathrm{L}) = \mathtt{x}_\mathrm{L}[\Theta_1] - \Theta_\text{th}$, where the operator $[\cdot]$ selects one dimension of $\mathtt{x}_\mathrm{L}$, $\Theta_1 \in \{1, \dots, D_\mathrm{L}\} \subset \mathbb{Z}$, and $\Theta_\text{th} \in \mathbb{R}$ is a threshold. Another option, which we also adopt in this work, are pair-wise differences for the response function, *i.e.*, $r_\Theta(\mathtt{x}_\mathrm{L}) = \mathtt{x}_\mathrm{L}[\Theta_1] - \mathtt{x}_\mathrm{L}[\Theta_2] - \Theta_\text{th}$, where also $\Theta_2 \in \{1, \dots, D_\mathrm{L}\} \subset \mathbb{Z}$.

The typical procedure for finding good parameters Θ for the splitting function $\sigma(\cdot)$ is to sample a random set of parameter values Θ_k and choosing the best one Θ^* according to a quality measure. The quality for a specific splitting function $\sigma(\mathtt{x}_\mathrm{L}, \Theta)$ is computed as

$$\mathrm{Q}(\sigma, \Theta, \mathtt{X}_\mathrm{H}, \mathtt{X}_\mathrm{L}) = \sum_{c \in \{Le, Ri\}} |\mathtt{X}^c| \cdot \mathrm{E}(\mathtt{X}_\mathrm{H}^c, \mathtt{X}_\mathrm{L}^c) \; , \tag{10}$$

where Le and Ri define the left and right child nodes and $|\cdot|$ is the cardinality operator. With a slight abuse of notation, we define for both domains $\mathtt{X}_{\{\mathrm{H,L}\}}^{Le} = \{\mathtt{x}_{\{\mathrm{H,L}\}} : \sigma(\mathtt{x}_\mathrm{L}, \Theta) = 0\}$, $\mathtt{X}_{\{\mathrm{H,L}\}}^{Re} = \{\mathtt{x}_{\{\mathrm{H,L}\}} : \sigma(\mathtt{x}_\mathrm{L}, \Theta) = 1\}$. The function $\mathrm{E}(\mathtt{X}_\mathrm{H}, \mathtt{X}_\mathrm{L})$ aims at measuring the compactness or the purity of the data. The intuition is to have similar data samples falling into the same leaf nodes, thus, giving coherent predictions.

For our task, we define a novel regularized quality measure $\mathrm{E}(\mathtt{X}_\mathrm{H}, \mathtt{X}_\mathrm{L})$ that not only operates on the label space $\mathcal{Y}$ but also on the input space $\mathcal{X}$. We define it as

$$\mathrm{E}(\mathtt{X}_\mathrm{H}, \mathtt{X}_\mathrm{L}) = \frac{1}{|\mathtt{X}|} \sum_{n=1}^{|\mathtt{X}|} \big(\|\mathtt{x}_\mathrm{H}^n - m(\mathtt{x}_\mathrm{L}^n)\|_2^2 + \kappa \cdot \|\mathtt{x}_\mathrm{L}^n - \bar{\mathtt{x}}_\mathrm{L}\|_2^2 \big) \; , \tag{11}$$

where $m(\mathtt{x}_\mathrm{L}^n)$ again is the prediction for the sample $\mathtt{x}_\mathrm{L}^n$, $\bar{\mathtt{x}}_\mathrm{L}$ is the mean over the samples $\mathtt{x}_\mathrm{L}^n$, and κ is a hyper-parameter.

The first term in (11) operates on the label space and we get different variants depending on the choice of $m(\mathtt{x}_\mathrm{L})$. If $m(\mathtt{x}_\mathrm{L})$ is the average over all samples in $\mathtt{X}_\mathrm{H}$, *i.e.*, a constant model, we have the reduction-in-variance objective that is employed in many works, *e.g.*, in Hough Forests [19]. Assuming a linear model $m(\mathtt{x}_\mathrm{L})$, we have a reconstruction-based objective as in [16], which is typically more time consuming than the reduction-in-variance option. The model chosen during growing the trees and for the final leaf nodes does not necessarily has to be the same. For instance, one could assume a constant model during growing the trees, but a linear one for the final leaf nodes.

The second term in (11) operates on the data space and can be considered as a clustering objective that is steered with κ. The intuition of this regularization is that we not only require to put samples in a leaf node that have similar

labels $\mathbf{x}_\mathrm{H}$. We also want those samples themselves (*i.e.*, $\mathbf{x}_\mathrm{L}$) to be similar, which potentially eases the task for the linear regression model $m_l(\mathbf{x}_\mathrm{L})$ in the leaf.

After fixing the best split $\sigma(\cdot)$ according to (10), the data in the current node is split and forwarded to the left and right children respectively. Growing continues until one of the stopping criteria is met and the final leaf nodes are created.

5. Single Image Super-Resolution

In this section, we assess the performance of our direct regression-based approach, SRF, for single image super-resolution on different data sets as well as 3 upscaling factors. After outlining our experimental setup including details on data sets and evaluation metrics, we compare with different random forest variants as well as several state-of-the-art SISR approaches. To provide more insights into the proposed method, we finally investigate several properties and parameter choices in more detail.

5.1. Experimental Setup and Benchmarks

The publicly available framework of Timofte *et al*. [32] is a perfect base for evaluating single image super-resolution approaches. It includes state-of-the-art results of different dictionary learning [32, 38, 39] as well as neighborhood embedding [4, 8] approaches allowing for a fair and direct comparison on the same code basis. As in many other super-resolution works [8, 22, 39], this framework also applies regular bicubic upsampling on the color components and the sophisticated upsampling models only on the luminance component of an image. The reason is the human visual perception that is much more sensitive to high frequency changes in intensity than in color.

Upscaling the luminance component of an image consists of two steps [32, 39]. First, regular bicubic upsampling is applied, which results in a rather low-frequency upscaled image I_{low}. To make up for the high-frequency components, the second step is to apply a model that predicts the high-frequency components I_{high} of the image. The final upscaled image is then given as $I_{final} = I_{low} + I_{high}$.

The above mentioned upsampling methods included in the framework of [32] all operate on small image patches and learn a mapping from the low- to the high-frequency component. All patches in a low-resolution image (typically overlapping) are upsampled to their corresponding high-resolution patches. Overlapping high-resolution patches are simply averaged to produce the final output. The main task is thus to find a mapping function $P_{high} = f_{L\to H}(F(P_{low}))$, where P_{low} and P_{high} are low- and high-frequency patches and $F(\cdot)$ is a feature extractor. The most basic feature is the patch itself. However, better results can be achieved by using first- and second-order derivatives of the patch followed by a basic dimensionality reduction, as was used in [32, 39]. In this work we use the exact same features allowing for a direct comparison of the upsampling method itself.

Set	Set5				Set14			
Factor	x2		x3		x2		x3	
RFC	35.3	0.6	31.3	0.3	31.4	1.2	28.2	0.6
RFL	36.4	0.7	32.1	0.4	32.2	1.5	28.9	0.8
RFP	36.4	0.9	32.2	0.5	32.2	1.7	28.9	1.0

Table 1: Comparison of different random forest variants on two data sets and for different upscaling factors. For each setting, the left column presents the PSNR in dB and the right column the upscaling time in seconds.

For our evaluations, we use three different sets of images, `Set5` from [4], `Set14` from [39], and `BSDS` from [3], consisting of 5, 14, and 200 images, respectively. Unless otherwise noted, the standard settings for our SRF are: $T = 15$, $\xi_{\text{max}} = 15$, $\lambda = 0.01$, and $\kappa = 1$.

5.2. Random Regression Forest Variants

Before comparing with state-of-the-art SISR approaches, we evaluate different variants of our super-resolution forests. In Section 4, we formulated our approach as a locally linear regressor, where locality is defined via a random forest. Thus, the random forest holds a linear prediction model in each leaf node, akin to filter forests [16]. For the linear model, we can choose between different basis functions $\phi(\mathbf{x})$. We evaluate the linear (RFL) and the polynomial (RFP) cases. In contrast, one can also store a constant model (RFC) in each leaf node by finding a mode in the continuous output space of the incoming data, *e.g.*, by taking the average (see Section 4). Another choice is using structured forests from Dollár and Zitnick [13], which learn a mapping from input to arbitrary structured outputs. However, this option makes little sense for this task because the output space is continuous and a quality measure can be defined easily also for high-dimensional output spaces, see Section 4.1.

For the comparison between these different random forest choices, we set the number of trees to $T = 8$ and keep the remaining standard settings. For the constant leaf node model (RFC), we fix the minimum number of samples per leaf node to 5. For the two linear models (RFL and RFP), this parameter is set higher (64) as a linear regression model has to be learned. We evaluate on `Set5` and `Set14` for two different magnification factors. All results are presented in Table 1. We report the upscaling quality (PSNR in dB) as well as the upscaling time (in seconds), respectively. While the performance gap between RFL and RFP is almost zero, one can clearly observe the inferior results of the constant leaf node model.

data set	factor	Bicubic PSNR/IFC/Time	Zeyde [39] PSNR/IFC/Time	ANR [32] PSNR/IFC/Time	NE+LLE PSNR/IFC/Time	RFL PSNR/IFC/Time	ARFL PSNR/IFC/Time
Set5	x2	33.66/6.08/0.0	35.77/7.86/3.9	35.82/8.07/0.7	35.76/7.86/5.4	36.55/8.52/1.2	**36.70/8.56**/1.2
	x3	30.39/3.58/0.0	31.91/4.51/1.8	31.91/4.60/0.4	31.84/4.51/2.5	32.46/4.92/0.9	**32.58/4.92**/1.0
	x4	28.42/2.33/0.0	29.73/2.96/1.1	29.73/3.03/0.3	29.62/2.96/1.4	30.15/**3.22**/0.8	**30.21**/3.19/0.7
Set14	x2	30.23/6.07/0.0	31.81/7.64/8.0	31.78/7.81/1.3	31.75/7.65/11.1	32.26/8.14/2.2	**32.37/8.14**/2.1
	x3	27.54/3.47/0.0	28.68/4.23/3.6	28.64/4.31/0.7	28.60/4.24/5.2	29.05/**4.54**/1.7	**29.13**/4.53/1.7
	x4	26.00/2.24/0.0	26.91/2.75/2.3	26.87/2.81/0.5	26.82/2.76/3.0	27.24/**2.95**/1.3	**27.30**/2.92/1.3
BSDS	x2	29.70/5.68/0.0	30.99/7.10/19.5	30.94/7.24/0.9	30.91/7.08/10.8	31.50/**7.58**/2.3	**31.62**/7.56/2.3
	x3	27.26/3.21/0.0	28.05/3.87/5.8	27.99/3.91/0.5	27.97/3.85/4.4	28.39/**4.15**/2.2	**28.46**/4.14/2.3
	x4	25.97/2.04/0.0	26.60/2.50/3.6	26.56/2.53/0.3	26.53/2.49/2.7	26.86/**2.67**/2.1	**26.90**/2.64/2.1

Table 2: Results of the proposed method compared within the framework of [32] on 3 data sets for 3 upscaling factors. The results of RFL are averaged over 3 independent runs. We omit the standard deviation in the table as it was negligibly low (0.0059 and 0.0058 on average for PSNR and IFC, respectively).

data set	factor	A+ [31] PSNR/IFC/Time	SRCNN [14] PSNR/IFC/Time	BPJDL [23] PSNR/IFC/Time	RFL PSNR/IFC/Time	RFL+ PSNR/IFC/Time	ARFL+ PSNR/IFC/Time
Set5	x2	36.55/8.47/0.8	36.34/7.52/3.0	36.41/7.77/129.8	36.55/8.52/1.1	36.73/8.66/2.0	**36.89/8.66**/2.1
	x3	32.59/4.93/0.5	32.39/4.31/3.0	32.10/4.50/110.0	32.46/4.92/1.0	32.63/**5.00**/1.6	**32.72**/4.99/1.7
	x4	30.28/3.25/0.3	30.09/2.84/3.2	-	30.15/3.22/0.8	30.29/**3.27**/1.5	**30.35**/3.24/1.5
Set14	x2	32.28/8.10/1.5	32.18/7.23/4.9	32.17/7.60/243.8	32.26/8.14/2.3	32.41/**8.28**/3.9	**32.52**/8.25/3.9
	x3	29.13/4.53/0.9	29.00/4.03/5.0	28.76/4.17/217.7	29.05/4.54/1.8	29.17/**4.60**/2.5	**29.23**/4.57/2.5
	x4	27.32/2.96/0.6	27.20/2.61/5.2	-	27.24/2.95/1.3	27.35/**2.98**/2.1	**27.41**/2.96/2.1
BSDS	x2	31.44/7.46/1.2	31.38/6.77/3.4	31.35/6.92/144.1	31.50/7.58/2.5	31.52/**7.61**/2.8	**31.66**/7.60/3.1
	x3	28.36/4.09/0.6	28.28/3.69/3.4	28.10/3.72/137.6	28.39/**4.15**/2.3	28.38/4.15/2.0	**28.45**/4.13/2.0
	x4	26.83/2.63/0.4	26.73/2.38/3.5	-	26.86/**2.67**/2.1	26.85/2.66/1.7	**26.89**/2.63/1.7

Table 3: Results of the proposed method compared with state-of-the-art works on 3 data sets for 3 upscaling factors.

5.3. Comparison with State-of-the-art

We split our comparison with related work into two parts. First, we compare with methods that build upon the exact same framework as [32, 39] in order to have a dedicated comparison of our regression model. Besides standard bicubic upsampling, we compare with a sparse coding approach [39], ANR [32], as well as neighborhood embedding [8]. We use the same 91 training images as in [32] for `Set5` and `Set14`. For `BSDS`, we use the provided 200 training images. Second, we compare with state-of-the-art methods in SISR that eventually build upon a different framework and use different sets of training images. We compare with A+ [32], SRCNN [14], and BPJDL [23]. We do not include the results of [11] as they are slightly worse than A+ [31]. Beside our standard model (RFL), we also evaluate an improved training scheme for random forests presented in [29], which we denote ARFL. This algorithm integrates ideas of gradient boosting in order to optimizes a global loss over all trees (we use the squared loss) and can be readily replaced with standard random forests. Additionally, we add two variants (RFL+ and ARFL+) which use an augmented training set consisting of the union of the 200 BSDS and the 91 images from [32].

Tables 2 and 3 show the quantitative results for both parts of our evaluation on different upscaling factors and image sets, respectively. We report both PSNR (in dB) and the IFC score, which was shown to have higher correlation with the human perception compared to other metrics [36]. As can be seen in Table 2, our super-resolution forests with linear leaf node models, RFL and ARFL, achieve better results than all dictionary-based and neighborhood embedding approaches. Furthermore, all our models compare favorably with state-of-the-art methods, see Table 3. We can even outperform the complex CNN, although our approach can be trained within minutes while [14] takes 3 days on a GPU. Note however that training the models with the scheme of [29] takes longer as intermediate predictions have to be computed, which involves solving a linear system. The inference times of SRCNN (in the table) differ from the ones reported in [14] as only the slower Matlab implementation is publicly available. Qualitative results can be found in Figures 1 and 2. As can be clearly seen, the proposed RFL produces sharp edges with little artifacts.

In Figure 3 we visualize the trade-off between upscaling quality and inference time for different methods. We include two additional neighborhood embedding approaches [4, 8] (NE+LS, NE+NNLS) as well as different

(a) Original (b) Bicubic: 28.64, 3.82 (c) ANR [32]: 30.08, 4.72 (d) A+ [31]: 31.54, 5.04

(e) SRCNN [14]: 31.35, 4.38 (f) BPJDL [23]: 30.53, 4.44 (g) RFL: 31.33, 5.03 (h) RFL+: 31.82, 5.18

Figure 2: Some qualitative results of state-of-the-art methods for upscaling factor x3 on image `legs`. The numbers in the subcaptions refer to PSNR and IFC scores, respectively. Best viewed in color and digital zoom.

variants of our random forest (with different number of trees $T = \{1, 5, 10, 15\}$). The figure shows the average results from `Set5` for an upscaling factor of 2. One can see that RFL provides a good trade-off between accuracy and inference time. Already the variant with a single tree (RFL-1) gives better results than many related methods. Using 5 trees improves the results significantly with only a slight increase in inference time. SRCNN [14] is clearly the fastest method because no feature computation is required (timings taken from [14]), which could potentially be sped-up in the framework of Timofte [32].

Finally, we compare different dictionary sizes for dictionary-based approaches with our random forest in Figure 4a. For our model, we include four variants with different number of trees $T = \{1, 5, 10, 15\}$. As can be seen, our weakest model ($T = 1$) already outperforms dictionary based models up to a dictionary size of 2048 while being almost as fast as GR [32] (Figure 4b). When using more trees, RFL outperforms even larger dictionaries. Figure 4c shows that our random forest model (trained from 91 images) takes less time to train than ANR [32], even though we do not train our trees in parallel yet.

5.4. Influence of the Tree Structure

The main factor influencing the tree structure of random forests, beside the inherent randomness induced, is the objective function used to evaluate potential splitting functions. We investigate the influence of six different choices. First, a fully random selection of the splitting function (Ra), *i.e.*, extremely randomized trees [20]. Second, an objective that prefers balanced trees (Ba), which we define as

$$\mathrm{Q}(\sigma, \Theta, \mathtt{X}) = \left(|\mathtt{X}^{Le}| - |\mathtt{X}^{Ri}|\right)^2 , \qquad (12)$$

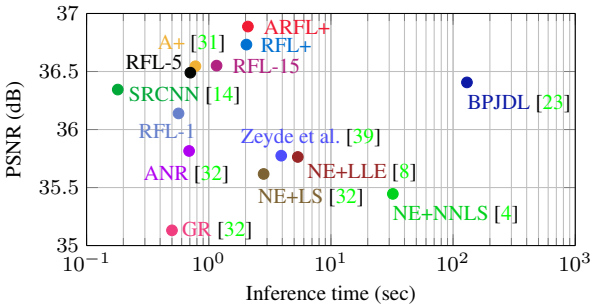


Figure 3: Visualization of the trade-off between accuracy and inference time for different methods. The results are the average over the images from `Set5`.

where $\mathtt{X}^{Le}$ and $\mathtt{X}^{Ri}$ are the set of samples falling into left and right child node according to splitting function σ. The remaining options are reduction-in-variance with $\kappa = 0$ (Va) and $\kappa = 1$ (VaF) and the reconstruction-based objective, again with $\kappa = 0$ (Re) and $\kappa = 1$ (ReF), respectively.

Another parameter in our implementation is the number of samples $\hat{N}$ considered for finding a splitting function σ in each node. We use reservoir sampling [34] to shrink the data $\mathtt{X}$ to $\min(|\mathtt{X}|, \hat{N})$ samples and use those to find a splitting function σ, which significantly reduces the training time without sacrificing quality. After fixing σ, all the data is forwarded to the left and right for further growing.

We present our results in Figure 5. The upscaling scores (Figure 5a) reveal that Va and Re are similarly good, while VaF and ReF give better results confirming the importance of the regularization in the objective function (11). While Ba and Ra are inferior, it is worth mentioning that the simple balanced objective function (Ba) (12) gives relatively

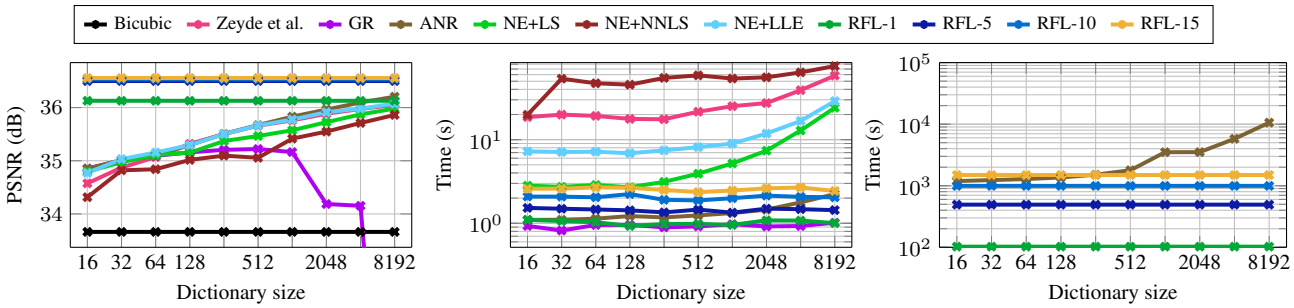


Figure 4: Our random forest model with different number of tress (1, 5, 10, and 15) compared to dictionary learning based approaches with different sizes of the dictionary `D`. We present the upscaling quality (a), the inference time (b), and the training time for ANR [32] (c) compared to our approach.

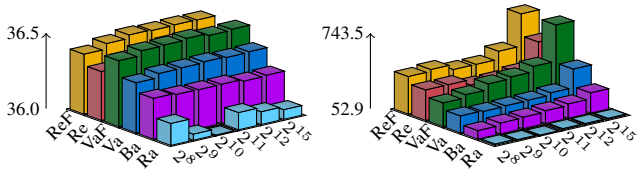


Figure 5: Influence of the tree structure (splitting objective and subsample size $\hat{N}$) on (a) the upscaling quality and (b) the training time of the trees. See text for more details.

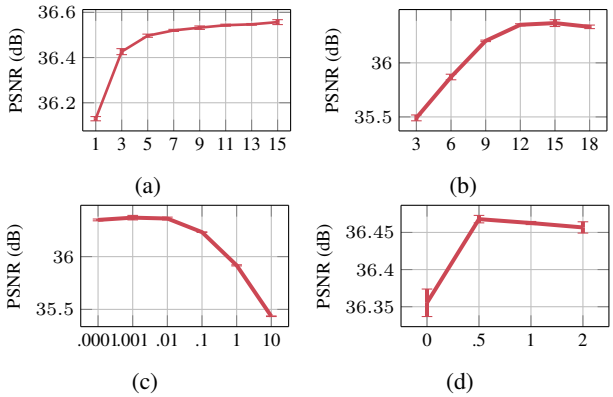


Figure 6: Random forest parameter evaluation on `Set5`: (a) number of trees T, (b) maximum tree depth $\xi_{\max}$, regularization parameters (c) λ, and (d) κ.

good results and being faster during training, *c.f.*, Figure 5b. On the other hand, the parameter $\hat{N}$ (evaluated for $2^{\{8,9,10,11,12,15\}}$) has little effect on the scores.

5.5. Important Random Forest Parameters

Our final set of experiments on single image super-resolution investigate several important parameters of our super-resolution forests (beside those that have already been investigated). These parameters include the number of trees T in the ensemble, the maximum tree depth $\xi_{\max}$, the regularization parameter λ for the linear regression in the leaf nodes, and, finally, the regularization parameter κ for the splitting objective. Figure 6a shows the expected behavior of the parameter T for the random forest approach. The performance steadily increases with increasing T until saturation, which is at around $T = 13$ for this particular application. The second parameter in our evaluation is the maximum tree depth $\xi_{\max}$, which has strong influence on training and inference times. From Figure 6b, we can see a saturation of the accuracy at depth $\xi_{\max} = 12$ and even a slight drop in performance with too deep trees. Figure 6c indicates to use a rather low regularization parameter λ. In Figure 6d we can again see that the regularization in Equation (11) is important and should be activated.

6. Conclusion

In this work, we present a new approach for single image super-resolution via random forests. We show the close relation between recent sparse coding based approaches and locally linear regression. We exploit this connection and avoid the detour of using a sparse-coded dictionary to learn the mapping from low- to high-resolution images. Instead, we follow a more direct approach with a random regression forest formulation. Our super-resolution forests build on linear prediction models in the leaf nodes instead of typically used constant models. Additionally, it employs a new regularization on the splitting objective function which operates on the output as well as the input domain of the data. Our results confirm the effectiveness of this approach on different benchmarks, where we outperform the current state-of-the-art. The inference of our random forest model is among the fastest methods and the training time is typically one or several orders of magnitude less than related approaches.

Acknowledgment: This work was supported by the Austrian Science Foundation (FWF) project Advanced Learning for Tracking and Detection (I535-N23) and by the Austrian Research Promotion Agency (FFG) projects Vision+ and IKT4QS1.

References

[1] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation. *TSP*, 54(11):4311–4322, 2006. 2

[2] Y. Amit and D. Geman. Shape Quantization and Recognition with Randomized Trees. *NECO*, 9(7):1545–1588, 1997. 2, 3

[3] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour Detection and Hierarchical Image Segmentation. *PAMI*, 33(5):898–916, 2011. 5

[4] M. Bevilacqua, A. Roumy, C. Guillemot, and M.-L. Alberi Morel. Low-Complexity Single-Image Super-Resolution based on Nonnegative Neighbor Embedding. In *BMVC*, 2012. 5, 6, 7

[5] C. M. Bishop. *Pattern Recogntion and Machine Learning*. Springer, 2007. 3

[6] L. Breiman. Random Forests. *ML*, 45(1):5–32, 2001. 2, 3

[7] R. Caruana, N. Karampatziakis, and A. Yessenalina. An empirical Evaluation of Supervised Learning in High Dimensions. In *ICML*, 2008. 2

[8] H. Chang, D.-Y. Yeung, and Y. Xiong. Super-Resolution Through Neighbor Embedding. In *CVPR*, 2004. 5, 6, 7

[9] T. F. Cootes, M. Ionita, C. Lindner, and P. Sauer. Robust and Accurate Shape Model Fitting using Random Forest Regression Voting. In *ECCV*, 2012. 2

[10] A. Criminisi and J. Shotton. *Decision Forests for Computer Vision and Medical Image Analysis*. Springer, 2013. 2, 3

[11] D. Dai, R. Timofte, and L. van Gool. Jointly Optimized Regressors for Image Super-resolution. In *Eurographs*, 2015. 2, 6

[12] M. Dantone, J. Gall, G. Fanelli, and L. v. Gool. Real-time Facial Feature Detection using Conditional Regression Forests. In *CVPR*, 2012. 2

[13] P. Dollár and L. Zitnick. Structured Forests for Fast Edge Detection. In *ICCV*, 2013. 2, 5

[14] C. Dong, C. Change Loy, K. He, and X. Tang. Learning a deep convolutional network for image super-resolution. In *ECCV*, 2014. 1, 2, 3, 6, 7

[15] C. E. Duchon. Lanczos Filtering in One and Two Dimensions. *JAM*, 18(8):1016–1022, 1979. 1

[16] S. Fanello, C. Keskin, P. Kohli, J. Izadi, Shahram Shotton, A. Criminisi, U. Pattaccini, and T. Paek. Filter Forests for Learning Data-Dependent Convolutional Kernels. In *CVPR*, 2014. 4, 5

[17] R. Fattal. Upsampling via Imposed Edges Statistics. *TOG*, 26(3):95, 2007. 1

[18] W. T. Freeman, T. R. Jones, and E. C. Pasztor. Example-Based Super-Resolution. *CGA*, 22(2):56–65, 2002. 1

[19] J. Gall and V. Lempitsky. Class-Specific Hough Forests for Object Detection. In *CVPR*, 2009. 2, 4

[20] P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *ML*, 63(1):3–42, 2006. 7

[21] R. Girshick, J. Shotton, P. Kohli, A. Criminisi, and A. Fitzgibbon. Efficient Regression of General-Activity Human Poses from Depth Images. In *ICCV*, 2011. 2

[22] Glasner, Daniel, Bagon, Shai and Irani, Michal. Super-Resolution From a Single Image. In *ICCV*, 2009. 1, 5

[23] L. He, H. Qi, and R. Zaretzki. Beta Process Joint Dictionary Learning for Coupled Feature Spaces with Application to Single Image Super-Resolution. In *CVPR*, 2013. 1, 6, 7

[24] P. Kontschieder, P. Kohli, J. Shotton, and A. Criminisi. GeoF: Geodesic Forests for Learning Coupled Predictors. In *CVPR*, 2013. 2

[25] J. J. Lim, C. L. Zitnick, and P. Dollár. Sketch Tokens: A Learned Mid-level Representation for Contour and Object Detection. In *CVPR*, 2013. 2

[26] C. Lu, J. Shi, and J. Jia. Abnormal Event Detection at 150 FPS in MATLAB. In *ICCV*, 2013. 2

[27] B. A. Olshausen and D. J. Field. Sparse Coding with an Overcomplete Basis Set: A Strategy Employed by V1? *VR*, 37(23):3311–3325, 1997. 2

[28] S. Rota Bulò and P. Kontschieder. Neural Decision Forest for Semantic Image Labelling. In *CVPR*, 2014. 2

[29] S. Schulter, C. Leistner, P. Wohlhart, P. M. Roth, and H. Bischof. Alternating Regression Forests for Object Detection and Pose Estimation. In *ICCV*, 2013. 6

[30] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-Time Human Pose Recognition in Parts from a Single Depth Image. In *CVPR*, 2011. 2

[31] R. Timofte, V. De Smet, , and L. Van Gool. A+: Adjusted Anchored Neighborhood Regression for Fast Super-Resolution. In *ACCV*, 2014. 3, 6, 7

[32] R. Timofte, V. De Smet, and L. Van Gool. Anchored Neighborhood Regression for Fast Example-Based Super-Resolution. In *ICCV*, 2013. 1, 2, 3, 5, 6, 7, 8

[33] J. van Ouwerkerk. Image super-resolution survey. *IVC*, 24(10):1039–1052, 2006. 2

[34] J. S. Vitter. Random Sampling with a Reservoir. *TOMS*, 11(1):37–57, 1985. 7

[35] 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. 1, 2

[36] C.-Y. Yang, C. Ma, and M.-H. Yang. Single-Image Super-Resolution: A Benchmark. In *ECCV*, 2014. 6

[37] C.-Y. Yang and M.-H. Yang. Fast Direct Super-Resolution by Simple Functions. In *ICCV*, 2013. 1, 2, 3

[38] J. Yang, J. Wright, T. Huang, and Y. Ma. Image Super-Resolution Via Sparse Representation. *TIP*, 19(11):2861–2873, 2010. 1, 2, 3, 5

[39] R. Zeyde, M. Elad, and M. Protter. On Single Image Scale-Up using Sparse-Representations. In *Curves and Surfaces*, 2010. 1, 2, 3, 5, 6, 7