22nd *Computer Vision Winter Workshop*
Nicole M. Artner, Ines Janusch, Walter G. Kropatsch (eds.)
Retz, Austria, February 6–8, 2017

Loss-Specific Training of Random Forests for Super-Resolution

Alexander Grabner, Georg Poier, Michael Opitz, Samuel Schulter, Peter M. Roth
Graz University of Technology
Institute of Computer Graphics and Vision
{alexander.grabner, poier, michael.opitz, schulter, pmroth}@icg.tugraz.at

Abstract. *Super-resolution addresses the problem of image upscaling by reconstructing high-resolution output images from low-resolution input images. One successful approach for this problem is based on random forests. However, this approach has a large memory footprint, since complex models are required to achieve high accuracy. To overcome this drawback, we present a novel method for constructing random forests under a global training objective. In this way, we improve the fitting power and reduce the model size. In particular, we combine and extend recent approaches on loss-specific training of random forests. However, in contrast to previous works, we train random forests with globally optimized structure and globally optimized prediction models. We evaluate our proposed method on benchmarks for single image super-resolution. Our method shows significantly reduced model size while achieving competitive accuracy compared to state-of-the-art approaches.*

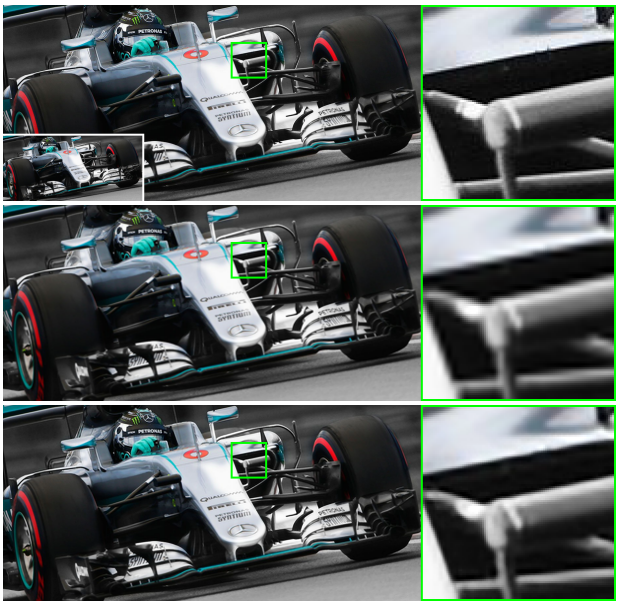


Figure 1: Example for super-resolution using an upscaling factor of 3. From top to bottom: the ground truth image, the result of bicubic upsampling and the result of our approach.

1. Introduction

Single image super-resolution (SR) is a subdomain of image reconstruction which addresses the problem of enhancing image resolution [15, 17, 27]. The goal is to estimate a visually pleasing high-resolution output image starting from a single low-resolution input image (see Figure 1). However, this upscaling is nontrivial, because one pixel in the low-resolution input image has to account for multiple pixels in the high-resolution output image. Therefore, SR is an ill-posed problem for which no unique solution exists.

As a result, SR is an active research area and many different approaches have been proposed. The most popular class of SR algorithms are interpolation methods [13,21,23,28]. In practice, many applications use bicubic interpolation [23]. This method is computationally efficient and can handle arbitrary upscaling factors, but lacks accuracy due to the assumption of smoothness over the entire image. SR methods based on machine learning techniques overcome this drawback [3, 8, 11, 12, 35–38]. These approaches exploit prior knowledge and show significantly improved accuracy compared to interpolation methods. However, they have a higher computational complexity and additionally require a training phase.

Many learning approaches build on the concept of neighbor embedding, which assumes that small patches from low-resolution images and their related high-resolution equivalents form manifolds with similar local geometry [8]. As a consequence, a previously unseen patch can be expressed as a com-

bination of known patches in feature space of the low-resolution domain. The same combination is then applied in the high-resolution domain to generate a prediction using the known equivalents of these patches. Therefore, neighbor embedding approaches rely on dictionaries of corresponding low-resolution and high-resolution patch representations. Because these dictionaries tend to grow very large, recent approaches make use of sparse coding to learn more compact dictionary representations and to reduce the computational complexity [35–37, 39].

One alternative to dictionary approaches is to directly map from low-resolution to high-resolution space with Convolutional Neural Networks (CNNs) [11, 12]. Today, the best accuracy is achieved with deep CNNs [24,25]. On the downside, these approaches are computationally expensive and training may take multiple days, even though using powerful GPUs [11]. To overcome this limitation, Schulter *et al*. [31] apply Random Forests (RFs) instead. This approach drastically reduces the computational cost while still achieving high accuracy. As a result, training can be performed within minutes on a CPU. However, this approach has a large memory footprint, since it requires complex models to ensure high accuracy.

In consequence of the above mentioned limitations of existing approaches, we present a new training algorithm which constructs RFs with significantly reduced model size without compromising on accuracy. Our method is inspired by recent works that train RFs under a global training objective [30, 32, 33]. While previous approaches focus on either globally optimizing the structure or globally optimizing the prediction models, we address both tasks jointly. Due to this optimization strategy, we significantly improve the fitting power of shallow RFs. As a consequence, we are able to reduce the depth and node count and thus lower the model complexity and memory demands. Our experiments show that we achieve competitive accuracy compared to existing RF approaches while reducing model size by a factor of 22.

The remainder of this work is structured as follows: In Section 2, we present the preliminaries and discuss related work. Next, we present our novel training algorithm in Section 3. In Section 4, we provide a systematic evaluation of our method and compare it to state-of-the-art approaches.

2. Preliminaries

In the following, we discuss the theoretical foundations our work builds on. First, we briefly review standard RFs [6] and Gradient Boosting (GB) [16]. Then, we discuss two relevant RF extensions: Alternating Decision and Regression Forests (ADRFs) [32, 33] and Global Refinement (GR) [30].

2.1. Random Forests

Decision Trees (DTs) [29] are nonlinear learners which make predictions based on a number of hierarchical decisions organized in the structure of a binary tree. They have proven to be powerful learners that can fit a training data set perfectly when fully grown, however, deep DTs introduce a high risk of overfitting, which results in low generalization [7].

One way to overcome this limitation, is to use an ensemble of DTs instead of a single DT, which reduces variance while maintaining low bias [1, 10, 18, 19]. In fact, a RF [6] is an ensemble that can be interpreted as a nonlinear function

$$F(\mathbf{x}) = \frac{1}{T}\sum_{t=1}^{T} F^t(\mathbf{x}), \tag{1}$$

which makes a prediction for a sample $\mathbf{x}$ by averaging over T individual DT predictions $F^t(\mathbf{x})$. All trees of the ensemble are independent, therefore, training a RF corresponds to constructing T distinct DTs. To create different trees, each DT is trained on a new training data set generated by randomly drawing samples from the original training data set, also known as Bagging [5]. Starting from a single root node, each DT recursively splits the provided training data set into disjoint subsets in a greedy manner [5]. Each node splits the arriving training data by evaluating a binary split function $\sigma(\mathbf{x}, \mathbf{\Theta})$, where the parameter $\mathbf{\Theta}$ is selected from a randomly generated set [9, 19]. This partitioning is greedily continued until a stopping criterion is met [2, 26]. Finally, a prediction model for each leaf node is calculated locally using the training data arriving at each individual leaf [7].

While the above training strategy is simple and parallelizable, it does not take the final model structure into account [32, 33]. The final result is computed by averaging locally optimal DT predictions, however, this does not guarantee a globally optimal prediction. As a result, the loss function implied by standard RFs is an average over the losses of the in-

dividual DTs, while ideally, the loss function is evaluated on the final output [30]. Therefore, the training of standard RFs is not directly guidable towards the optimization of a specific loss function. Two approaches which try to overcome this drawback by sharing information between the individual DTs are ADRFs [32, 33] and GR [30].

2.2. Gradient Boosting

GB [16] is an ensemble method, which combines several weak learners $h_d(\mathbf{x})$ to form a strong learner $F(\mathbf{x})$. The prediction of the ensemble is computed as the weighted sum of the different weak learner predictions. In this work, we consider the individual weak learners to be equally weighted:

$$F(\mathbf{x}) = \sum_{d=1}^{D} h_d(\mathbf{x}). \tag{2}$$

GB constructs a strong learner by iteratively adding weak learners one step at a time. More formally, at step d a new weak learner is added to the ensemble to provide a better model $F_d(\mathbf{x}) = F_{d-1}(\mathbf{x}) + h_d(\mathbf{x})$, where $F_{d-1}(\mathbf{x})$ is the current strong learner consisting of $d-1$ weak learners. To improve the model, GB applies gradient descent in function space and fits $h_d(\mathbf{x})$ to approximate the negative gradients

$$-\mathbf{g}_d(\mathbf{x}) = -\left[\frac{\partial \mathcal{L}(\mathbf{y}, F(\mathbf{x}))}{\partial F(\mathbf{x})}\right]_{F(\mathbf{x})=F_{d-1}(\mathbf{x})} \tag{3}$$

of a differentiable loss function given the previous model estimate $F_{d-1}(\mathbf{x})$. In this work, we use the L_2 loss $\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{2}\|\mathbf{y} - \hat{\mathbf{y}}\|_2^2$ as a loss function which reduces the computation of the negative gradients to the residuals $-\mathbf{g}_d(\mathbf{x}) = \mathbf{y} - F_{d-1}(\mathbf{x})$ given the current model estimate. Typically DTs serve as a weak learners [16].

2.3. Alternating Decision and Regression Forests

ADRFs [32, 33] modify the training procedure of standard RFs by incorporating ideas from GB. In contrast to GB, where DTs are added sequentially to the model, ADRFs iteratively increase the depth of all DTs, while their number remains constant.

The main difference of ADRFs compared to standard RFs is the replacement of the greedy DT training strategy by a stage-wise training scheme. Instead of greedily training all DTs independently of each other, the depth of the entire RF is iteratively increased stage by stage, where one stage corresponds to one depth level. Therefore, each iteration extends the RF by one extra depth level, which corresponds to splitting all leaf nodes of the current model. In this case, increasing the depth can be interpreted as adding a new weak learner in the sense of GB.

The stage-wise training scheme produces a fully functional RF in each iteration. At each stage this intermediate RF is used to evaluate the performance of the current model. The obtained results are then used to guide the training of the next stage towards a solution that compensates for the error of the model in its current state. In this way, ADRFs integrate the optimization of a global loss function directly into the tree growing process.

Similar to GB, ADRFs rely on the negative gradients in Eq. (3) of a differentiable loss function $\mathcal{L}(\cdot)$ to influence the training of the next stage. In this work, we evaluate the L_2 loss between the training data ground truth $\mathbf{y}$ and the RF prediction $F(\mathbf{x})$ defined in Eq. (1) given $F_{d-1}(\mathbf{x})$, the current state of the RF trained up to depth $d-1$. As a consequence, the training of ADRFs alternates between updating the global training objective based on $-\mathbf{g}_d(\mathbf{x})$ and increasing the depth of the RF.

2.4. Global Refinement

GR [30] also builds on a global loss optimization for RFs, but follows a completely different strategy compared to ADRFs. Instead of optimizing the tree growing process, GR relearns the prediction models of existing RFs. Starting from a pre-trained RF the leaf node predictions of all DTs are jointly retrained under a global training objective. This is formulated as a convex optimization problem [4]

$$\begin{aligned} \min_{W} \frac{1}{2}\|W\|_F^2 + \frac{C}{N}\sum_{n=1}^{N} \mathcal{L}(\mathbf{y}_n, \hat{\mathbf{y}}_n) \\ \text{s.t. } \hat{\mathbf{y}}_n = W\phi(\mathbf{x}_n)\ , \end{aligned} \tag{4}$$

where the RF prediction is reformulated as the product of a weight matrix W and an indicator vector $\phi(\mathbf{x}_n)$. In this case, W is a matrix holding the prediction models of all leaves and $\phi(\mathbf{x}_n)$ is a binary embedding vector which encodes leaf membership induced from the RF structure given $\mathbf{x}_n$. The dimensionality of $\phi(\mathbf{x})$ equals the total number of leaves of the RF and the i-th element of $\phi(\mathbf{x}_n)$ is 1 if the sample $\mathbf{x}_n$ falls into leaf i. In this work, we choose the convex loss function $\mathcal{L}(\cdot)$ to be the L_2 loss. Additionally, the Euclidean norm of W is minimized

to reduce the risk of overfitting [34] and the parameter C controls the tradeoff between the regularization term and the data term.

The refined leaf prediction models in W are used to overwrite the existing prediction models of the RF. As a result, the predictions of individual DTs are not independent of each other and the leaf nodes of the entire ensemble collaboratively make a prediction. However, the prediction of the RF is still computed as defined in Eq. (1).

3. Additive Global Refinement

In contrast to previous methods, we propose to train RFs with both globally optimized structure and globally optimized prediction models by combining the benefits of ADRFs and GR. In particular, ADRFs globally optimize the structure of RFs, while the leaf prediction models remain untouched, whereas GR relearns the leaf prediction models of pre-trained RFs, while the structure remains untouched. Thus, as a first straight-forward extension we propose to perform GR on top of ADRFs to achieve performance superior to both methods individually. We refer to this combination as ADRF+GR.

In addition to this training approach, we propose a novel refinement strategy which we refer to as Additive Global Refinement (AGR). In this way, we improve upon standard GR by taking advantage of existing prediction models instead of ignoring them. While GR discards the existing prediction models of a RF and replaces them with new ones, our approach refines existing prediction models. In contrast to GR, we propose to use the existing prediction models as a starting point. We perform a global leaf prediction model optimization to calculate prediction models which compensate for the error of existing prediction models. In this way, we improve the existing prediction models of a RF instead of relearning them from scratch.

Thus, AGR can be seen as a variant of GR which performs a gradient descent [4] step on an existing RF. The leaf prediction model optimization is used to apply a step in the steepest descent direction which is computed by the negative gradient of a differentiable loss function given the current state of the model. Therefore, AGR is closely related to a single iteration in the stage-wise training scheme of ADRFs. In contrast, we do not split the leaf nodes in the current model and do not increase the size of the model. Instead, the gradient descent step is implemented by optimizing

$$\min_{W} \frac{1}{2}\|W\|_F^2 + \frac{C}{N}\sum_{n=1}^{N} \mathcal{L}(-\mathbf{g}_d(\mathbf{x}_n), \hat{\mathbf{y}}_n) \quad \text{s.t. } \hat{\mathbf{y}}_n = W\phi(\mathbf{x}_n) \ , \tag{5}$$

a modified version of the optimization problem solved for GR. In contrast to Eq. (4), we evaluate the L_2 loss on the negative gradients $-\mathbf{g}_d(\mathbf{x})$ from Eq. (3), which in this case correspond to the current residuals, and the reformulated RF prediction. All other parameters remain the same. As a result, the prediction models obtained by AGR compensate for the error of existing prediction models instead of making an independent prediction. This means that the new prediction models and the existing prediction models must be summed up to obtain the final refined prediction models. Therefore, the new prediction models in W are added to the existing prediction models, instead of replacing them. Since we only perform a single gradient descent step, we choose the step size of the update to be 1.

The main advantage of AGR compared to GR is that our method retains the strength of individual DTs. In fact, when using GR some leaves have a high contribution to the final result, while others have a low contribution. Therefore, the prediction of a single DT is highly decorrelated to the final result, because there is no constraint which enforces coherence between the individual DT predictions. The deeper the RF, the higher the higher the risk of overfitting. In fact, experiments show that the regularization techniques employed in GR are not sufficient to prevent overfitting [30]. As a consequence, Ren *et al.* [30] propose an exhaustive iterative post-pruning strategy to reduce overfitting.

In contrast, prediction models obtained by AGR are more correlated to the final result, because they build on locally optimal prediction models. AGR performs a refinement which approximates the residuals of the current model. These residuals have low norm as locally optimal prediction models already provide a good estimate. The deeper the RF, the lower the norm of the residuals. AGR might overfit the residuals for deep RFs, but the contribution of the refinement to the existing prediction models is negligible, due to the low norm. Thus, the prediction of a single DT already provides a good estimate of the final prediction, even in the case of overfitting. This leads to DTs with higher individual strength and increased robustness to overfitting.

AGR is applicable to any pre-trained RF for regression and overcomes the drawbacks of standard GR. We propose to combine ADRFs with our novel refinement strategy (ADRF+AGR) to achieve highest accuracy. However, we also evaluate ADRF+GR to highlight the benefits of our proposed refinement method over the straight-forward combination. Additionally, we show that refinement strategies are not only applicable for constant leaf prediction models, but also for linear leaf prediction models which are used in RF approaches for SR [31].

4. Evaluation

In this Section we evaluate our proposed method for single image SR [15, 17, 27]. We provide results on different SR benchmarks, compare against state-of-the-art approaches and provide an analysis of the most important parameters of our training algorithm.

4.1. SR framework

For our evaluation, we build on the SR framework provided by Schulter *et al*. [31], which is based on the code of Timofte *et al*. [35]. The framework uses bicubic interpolation [23] to obtain the desired output resolution followed by a sharpening using machine learning techniques.

First, images are transformed from RGB to YCbCr color space which separates luminance information (Y) and color information (Cb and Cr), because the human visual system is most sensitive to high frequency changes in luminance [14]. In contrast, color information only plays a minor role in the human perception of sharpness. Therefore, bicubic interpolation is applied to all image channels, but sharpening is only performed on the Y channel. The framework works in a patch-based manner, where overlapping patches are extracted from the luminance channel of the upscaled low-resolution image. For each of these patches, a machine learning algorithm is used to estimate a high frequency patch which corrects for the blur in the upscaled patch. Therefore, the bicubic upscaled patch and the estimated high frequency patch are summed up to obtain the final sharpened patch. Finally, the sharpened patches are stitched together to obtain the sharpened output image.

4.2. Experimental setup

We use our proposed approach (ADRF+AGR) to estimate high frequency patches based on features extracted from upscaled low-resolution patches. For the features, we compute the first and second order derivatives of the upscaled low-resolution patches and apply dimensionality reduction using PCA [22]. The dimensionality reduction preserves 99.9% of the average energy [35], which, in this case, results in a 30 dimensional feature vector for each patch.

We use the training data set provided by Yang *et al*. [38], which consists of 91 images with a resolution around 0.1 megapixel. From these images, we sample a total of one million patches for training and report the performance on two test data sets, *Set5* [3] and *Set14* [39]. These data sets have also been used for training and benchmarking different SR methods in [11, 24, 25, 31, 35, 36].

We evaluate the performance by computing the Peak Signal-to-Noise Ratio (PSNR) [20] between the ground truth and the predicted image. For selected experiments, we report the PSNR improvement over bicubic interpolation which we refer to as *gain* in dB [11]. In any case, higher PSNR or *gain* corresponds to better accuracy. Because we train randomized predictors, all figures and tables report the mean of three independent runs [31]. However, the variation in accuracy between these runs was negligibly low in all experiments. We use the same RF parameters and split node optimization as Schulter *et al*. [31]. In particular, we set the number of DTs to 15 and the maximum tree depth to 12.

4.3. Comparison to state-of-the-art

First, we compare the performance of our method against state-of-the-art methods in terms of accuracy. Table 1 presents results for multiple SR factors on *Set5* [3] and *Set14* [39]. A+ [36], ADRF [31] and SRCNN [11] show a similar level of accuracy compared to our method. However, all of these methods are slightly outperformed by recent deep CNN approaches [24, 25]. Depending on the data set DRCN [25] performs 0.6 to 1.3 dB better than our approach. However, one disadvantage of these methods is that powerful GPUs are required to handle the computational cost in a reasonable amount of time. Therefore, we compare the number of Floating-Point Operations (FLOPs) required to super-resolve a 512×512 image for our approach and SRCNN [11], which achieve a similar level of accuracy. FLOPs are an implementation independent metric for the computational workload of an algorithm, while the commonly reported execution time [11, 31, 35, 36]

Data Set	SR Factor	Methods							
		Bicubic	A+ [36]	ADRF [31]	SRCNN [11] (ECCV)	SRCNN [12] (PAMI)	VDSR [24]	DRCN [25]	ADRF+AGR (Ours)
Set5 [3]	×2	33.66	36.55	36.70	36.34	36.66	37.53	37.63	36.55
	×3	30.39	32.59	32.58	32.39	32.75	33.66	33.82	32.50
	×4	28.42	30.29	30.21	30.09	30.49	31.35	31.53	30.13
Set14 [39]	×2	30.23	32.28	32.37	32.18	32.45	33.03	33.04	32.27
	×3	27.54	29.13	29.13	29.00	29.30	29.77	29.76	29.07
	×4	26.00	27.33	27.30	27.20	27.50	28.01	28.02	27.25

Table 1: Comparison of state-of-the-art methods for SR on Set5 [3] and Set14 [39]. For each data set and each SR factor, the three best performing methods are highlighted in shades of green. The best accuracy is achieved by recent deep CNN approaches. However, their improved accuracy comes at the price of exhaustive computational workload.

is highly dependent on the specific implementation and the used hardware. Our approach requires 2.75 GFLOPs, whereas SRCNN requires 4.77 GFLOPs. Even though SRCNN only uses a shallow CNN architecture, our approach is much more efficient due to the inexpensive evaluation of RFs. Additionally, deeper CNN architectures [24, 25] require even more FLOPs.

4.4. Parameters

Next, we investigate several parameters which are crucial to the performance of our approach. In addition to our proposed method (ADRF+AGR) we also evaluate a naive combination of ADRF and GR (ADRF+GR) to highlight the benefits of our approach over this simple combination.

4.4.1 Maximum depth

The most important parameter concerning accuracy on previously unseen data is the maximum tree depth D_{max}. Figure 2 shows the accuracy of different RF approaches as a function of D_{max}. In this experiment we report the *gain* in dB and vary D_{max} in the range from 0 to 12.

Non-refinement approaches (RF and ADRF) perform best with deep DTs which correspond to complex models with large model sizes. In contrast, refinement approaches (ADRF+GR and ADRF+AGR) perform best with more shallow DTs and outperform non-refinement approaches for low D_{max}, but show overfitting for high D_{max}. Overall the best accuracy is achieved by deep ADRF. One reason for this is that the global leaf prediction model optimization increases the fitting power of a RF significantly. The weight decay regularization employed in Eqs. (4) and (5) is not sufficient to prevent overfitting for deep RFs.

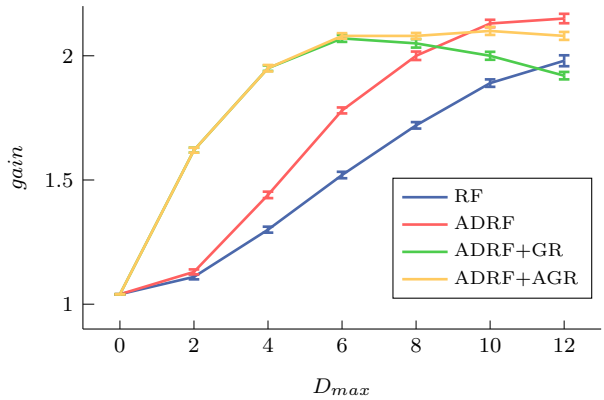


Figure 2: Evaluation of the maximum tree depth D_{max} on Set5 [3]. We report the *gain* which is the PSNR improvement over the bicubic baseline in dB as a function of D_{max}. Our approaches significantly outperform RF and ADRF for low D_{max}.

While the accuracy of ADRF+GR decreases for high D_{max}, the accuracy of ADRF+AGR saturates. One reason for this is that AGR uses the existing prediction models of a RF as a starting point instead of relearning them from scratch. In this way, the contribution of the refinement to the final prediction is reduced, because the residuals which serve as optimization targets have low norm. As a result, AGR is more robust to overfitting than GR. Interestingly, we observe that the accuracy improvement of non-refinement approaches between $D_{max} = 0$ and $D_{max} = 2$ is low. In contrast, the initial D_{max} increments for refinement approaches show the highest accuracy improvement. One reason for this is the large size of the training data set which consists of one million training samples. At $D_{max} = 2$ a balanced DT has 4 leaves and the entire training data set is distributed among these leaves. Therefore, the average number of samples arriving at a node is 250 000. At $D_{max} = 2$ the partitioning of the train-

ing data set is too coarse to fit local prediction models which significantly outperform ordinary regularized linear regression ($D_{max} = 0$). In contrast, refinement approaches can exploit the redundancy of multiple DTs to improve accuracy at low D_{max}.

4.5. Model size

The above analysis shows that all evaluated RF approaches show similar performance for the best performing D_{max}. Figure 3 presents qualitative results for different RF approaches for a test image from Set5 [3], which confirm this observation. Non-refinement approaches perform best with higher D_{max}, which corresponds to complex models with large model sizes. In contrast, refinement approaches perform best for lower D_{max}, which corresponds to simpler models with smaller model sizes. Table 2 reports the accuracy and model size of different RF approaches for their individual best performing D_{max}. Non-refinement approaches show high memory demands, due to the exponential growth of the model size with D_{max}. In contrast, the model size obtained by refinement approaches is more than 22 times smaller than the model size obtained by non-refinement approaches. Interestingly, Schulter *et al*. [31] train even deeper ADRFs ($D_{max} = 15$), but only increase the mean accuracy across different SR factors by negligible 0.02 dB compared to our ADRF experiments with $D_{max} = 12$.

Method	D_{max}	PSNR	Model Size
RF	12	32.38	447.54
ADRF	12	32.54	446.71
ADRF+GR	6	32.46	19.81
ADRF+AGR	6	32.48	19.81

Table 2: Evaluation of the model size for different methods. The reported results show the mean PSNR on Set5 [3] in dB and the model size in MB. Due to the improved fitting power at low D_{max}, our approaches show significantly reduced model size while achieving competitive accuracy.

4.6. Strength and correlation

In this experiment we analyze the strength and correlation of individual DTs for different methods. Breiman [6] shows that the generalization error of RFs depends on the strength and correlation of the individual DTs. The higher the strength and the lower the correlation, the lower the generalization error. For regression, the residuals of a DT indicate the strength of a DT. The lower the residuals, the higher the strength. Table 3 reports the strength and correlation of individual DTs for $D_{max} = 5$.

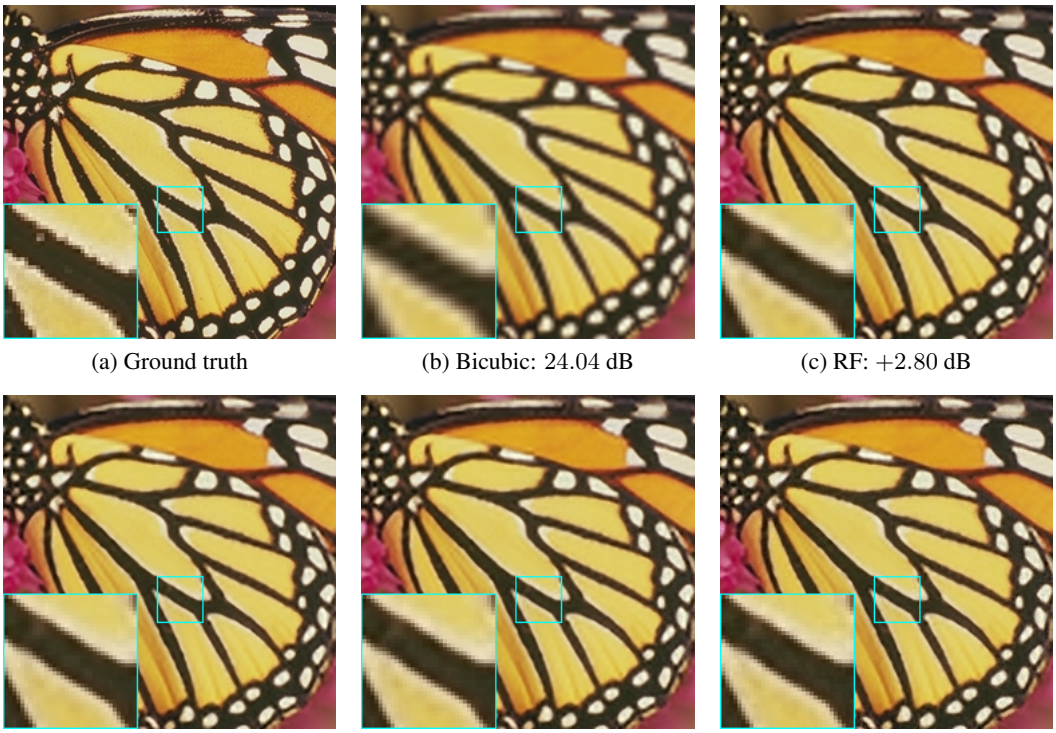
Method	PSNR (RF)	PSNR (DTs)	Correlation
RF	3.24 ± 0.02	3.92 ± 0.04	0.67 ± 0.02
ADRF	3.14 ± 0.02	5.54 ± 0.08	0.33 ± 0.02
ADRF+GR	3.36 ± 0.13	15.65 ± 0.68	0.10 ± 0.01
ADRF+AGR	3.12 ± 0.07	7.06 ± 0.08	0.20 ± 0.01

Table 3: Evaluation of strength and correlation for different methods. The reported results show the PSNR in dB. Refinement techniques decrease the strength, but increase the decorrelation of individual DTs.

Non-refinement approaches (RF and ADRF) show higher strength, but also higher correlation. In contrast, refinement approaches (ADRF+GR and ADRF+AGR) shower lower strength, but also lower correlation. Maximum performance is achieved by ARF+AGR which finds the best tradeoff between strength and correlation. This experiment proves that AGR helps in retaining the strength of individual DTs compared to GR.

5. Conclusion

In this work we presented a method for constructing RFs with reduced model size under a global training objective. Our method combines the benefits of ADRFs [32, 33] and GR [30] by constructing RFs with both globally optimized structure and globally optimized prediction models. Experiments confirm that we achieve competitive performance compared to state-of-the-art RF approaches with significantly simpler models which correspond to more shallow DTs. For single image SR, we reduce the memory requirement by a factor of 22 without loss of accuracy. Moreover, our training strategy improves robustness to overfitting. In contrast to GR, our AGR builds on the existing prediction models of RFs instead of relearning them from scratch. In this way, the contribution of the refinement to the final prediction is reduced. As a result, refined RFs are less prone to overfitting. Additionally, we show that our SR approach is more efficient than CNN-based methods. From an algorithmic point of view we require only half the operations compared to SRCNN [11] while achieving competitive accuracy.



(a) Ground truth (b) Bicubic: 24.04 dB (c) RF: +2.80 dB

(d) ADRF: +3.27 dB (e) ADRF+GR: +3.12 dB (f) ADRF+AGR: +3.23 dB

Figure 3: Qualitative SR results for *butterfly* from Set5 [3] using an upscaling factor of 3. We report the PSNR for the bicubic upscaled image and the *gain* for different RF approaches. While all of these methods achieve a similar level of visual quality, the size of the used models differs significantly (see Table 2).

References

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

[2] R. Barros, A. de Carvalho, and A. A. Freitas. *Automatic Design of Decision-Tree Induction Algorithms*. Springer, 2015. 2

[3] M. Bevilacqua, A. Roumy, C. Guillemot, and M.-L. Morel. Low-Complexity Single-Image Super-Resolution based on Nonnegative Neighbor Embedding. In *British Machine Vision Conference*, 2012. 1, 5, 6, 7, 8

[4] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. 3, 4

[5] L. Breiman. Bagging Predictors. *Machine Learning*, 24(2):123–140, 1996. 2

[6] L. Breiman. Random Forests. *Machine Learning*, 45(1):5–32, 2001. 2, 7

[7] L. Breiman, J. Friedman, C. Stone, and R. Olshen. *Classification and Regression Trees*. CRC Press, 1984. 2

[8] H. Chang, D.-Y. Yeung, and Y. Xiong. Super-Resolution Through Neighbor Embedding. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2004. 1

[9] T. Dietterich. An Experimental Comparison of three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization. *Machine Learning*, 40(2):139–157, 2000. 2

[10] T. Dietterich. Ensemble Methods in Machine Learning. In *Multiple Classifier Systems*, pages 1–15. Springer, 2000. 2

[11] C. Dong, C. C. Loy, K. He, and X. Tang. Learning A Deep Convolutional Network for Image Super-Resolution. In *European Conference on Computer Vision*, 2014. 1, 2, 5, 6, 7

[12] C. Dong, C. C. Loy, K. He, and X. Tang. Image Super-Resolution Using Deep Convolutional Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2):295–307, 2016. 1, 2, 6

[13] C. Duchon. Lanczos Filtering in One and Two Dimensions. *Journal of Applied Meteorology*, 18(8):1016–1022, 1979. 1

[14] C. Feichtenhofer, H. Fassold, and P. Schallauer. A Perceptual Image Sharpness Metric based on Local Edge Gradient Analysis. *IEEE Signal Processing Letters*, 20(4):379–382, 2013. 5

[15] W. Freeman, E. Pasztor, and O. Carmichael. Learning Low-Level Vision. *International Journal of Computer Vision*, 40(1):25–47, 2000. 1, 5

[16] J. Friedman. Greedy Function Approximation: A Gradient Boosting Machine. *Annals of Statistics*, 29(5):1189–1232, 2001. 2, 3

[17] D. Glasner, S. Bagon, and M. Irani. Super-Resolution from a Single Image. In *IEEE International Conference on Computer Vision*, 2009. 1, 5

[18] T. K. Ho. Random Decision Forests. In *IEEE International Conference on Document Analysis and Recognition*, 1995. 2

[19] T. K. Ho. The Random Subspace Method for Constructing Decision Forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998. 2

[20] Q. Huynh-Thu and M. Ghanbari. Scope of Validity of PSNR in Image/Video Quality Assessment. *Electronics Letters*, 44(13):800–801, 2008. 5

[21] L. Jing, G. Zongliang, and Z. Xiuchang. Directional Bicubic Interpolation - A New Method of Image Super-Resolution. In *International Congress on Image and Signal Processing*, 2013. 1

[22] I. Jolliffe. *Principal Component Analysis*. Wiley & Sons, 2002. 5

[23] R. Keys. Cubic Convolution Interpolation for Digital Image Processing. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 29(6):1153–1160, 1981. 1, 5

[24] J. Kim, J. K. Lee, and K. M. Lee. Accurate Image Super-Resolution Using Very Deep Convolutional Networks. *The Computing Research Repository*, arXiv:1511(04587):1–8, 2015. 2, 5, 6

[25] J. Kim, J. K. Lee, and K. M. Lee. Deeply-Recursive Convolutional Network for Image Super-Resolution. *The Computing Research Repository*, arXiv:1511(04491):1–8, 2015. 2, 5, 6

[26] J. K. Martin. An Exact Probability Metric for Decision Tree Splitting and Stopping. *Machine Learning*, 28(2):257–291, 1997. 2

[27] S. C. Park, M. K. Park, and M. G. Kang. Super-Resolution Image Reconstruction: A Technical Overview. *IEEE Signal Processing Magazine*, 20(3):21–36, 2003. 1, 5

[28] A. Parker, R. Kenyon, and D. Troxel. Comparison of Interpolating Methods for Image Resampling. *IEEE Transactions on Medical Imaging*, 2(1):31–39, 1983. 1

[29] J. R. Quinlan. Induction of Decision Trees. *Machine Learning*, 1(1):81–106, 1986. 2

[30] S. Ren, X. Cao, Y. Wei, and J. Sun. Global Refinement of Random Forest. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 2, 3, 4, 7

[31] S. Schulter, C. Leistner, and H. Bischof. Fast and Accurate Image Upscaling with Super-Resolution Forests. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 2, 5, 6, 7

[32] S. Schulter, C. Leistner, P. Wohlhart, P. M. Roth, and H. Bischof. Alternating Regression Forests for Object Detection and Pose Estimation. In *IEEE International Conference on Computer Vision*, 2013. 2, 3, 7

[33] S. Schulter, P. Wohlhart, C. Leistner, A. Saffari, P. M. Roth, and H. Bischof. Alternating Decision Forests. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013. 2, 3, 7

[34] A. Tikhonov and V. Arsenin. *Solutions of Ill-Posed Problems*. VH Winston, 1977. 3

[35] R. Timofte, V. De, and L. Van Gool. Anchored Neighborhood Regression for Fast Example-Based Super-Resolution. In *IEEE International Conference on Computer Vision*, 2013. 1, 2, 5

[36] R. Timofte, V. De Smet, and L. Van Gool. A+: Adjusted Anchored Neighborhood Regression for Fast Super-Resolution. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014. 1, 2, 5, 6

[37] C.-Y. Yang and M.-H. Yang. Fast Direct Super-Resolution by Simple Functions. In *IEEE International Conference on Computer Vision*, 2013. 1, 2

[38] J. Yang, J. Wright, T. Huang, and Y. Ma. Image Super-Resolution as Sparse Representation of Raw Image Patches. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008. 1, 5

[39] R. Zeyde, M. Elad, and M. Protter. On Single Image Scale-Up using Sparse-Representations. In *Curves and Surfaces*, pages 711–730. Springer, 2012. 2, 5, 6