Training an Active Random Field for Real-Time Image Denoising

Adrian Barbu

***Abstract*—Many computer vision problems can be formulated in a Bayesian framework based on Markov Random Fields (MRF) or Conditional Random Fields (CRF). Generally, the MRF/CRF model is learned independently of the inference algorithm that is used to obtain the final result. In this paper, we observe considerable gains in speed and accuracy by training the MRF/CRF model together with a fast and suboptimal inference algorithm. An Active Random Field (ARF) is defined as a combination of a MRF/CRF based model and a fast inference algorithm for the MRF/CRF model. This combination is trained through an optimization of a loss function and a training set consisting of pairs of input images and desired outputs. We apply the Active Random Field concept to image denoising, using the Fields of Experts MRF together with a 1-4 iteration gradient descent algorithm for inference. Experimental validation on unseen data shows that the Active Random Field approach obtains an improved benchmark performance as well as a 1000-3000 times speedup compared to the Fields of Experts MRF. Using the ARF approach, image denoising can be performed in real-time, at 8fps on a single CPU for a 256×256 image sequence, with close to state-of-the-art accuracy.**

***Index Terms*—MRF training, CRF training, Fields of Experts, image denoising.**

EDICS: TEC-RST

I. Introduction

Many real-world applications can be regarded as graph-based optimization problems, where the graph nodes are some smaller granularities of the system, such as atoms for material science and pixels for computer vision. In some cases (e.g. material science), a unique energy function that can be described mathematically exists and can accurately represent the relationship between the graph nodes. In computer vision, the natural images exhibit very complex structures for which it is difficult if not impossible to find an exact mathematical model that is computationally feasible.

Many of these computer vision problems are approached by constructing models based on Markov Random Field (MRF) or Conditional Random Field (CRF) energy functions and obtaining the solution through an optimization procedure. The optimization is one of the available MRF/CRF Maximum A Posteriori (MAP) inference algorithms such as gradient descent, Belief Propagation [44], Graph Cuts [5], Iterated Conditional Modes [3], etc. However, such an approach faces two challenges when applied to real-world problems.

First, the energy function must be computationally feasible in the sense that the minimum should be found in polynomial time. This does not usually happen in reality, since finding the global minimum for most energy functions associated to real-world applications is NP hard. For example, finding the global minimum of the Potts model [28], used in Stereo Matching as a prior term [5], [31], [32], is NP hard [5]. In such cases, polynomial-time algorithms are not expected to be found.

Second, it is very hard to find energy functions that always have a global minimum exactly at the desired solution. For example, even though the Potts model has been widely used in Stereo, the energy level of the desired result is higher than the energy obtained by different optimization algorithms [32], or the global minimum [23]. Recent work [1], [19], [20], [34], [35] introduced methods for training the MRF parameters such that the MRF energy minimum is as close as possible to the desired output on a training set.

The goal of this paper is to observe that when an approximate model is sought, it is sometimes not necessary to find the global minimum of the MRF energy. It has been shown in [39] that for applications with limited computational budget, the MAP parameter estimation does not give the best accuracy, and training biased estimators could compensate some of the errors introduced by the fast and approximate inference algorithm. How much can the biased estimators compensate for the suboptimal algorithm? In this paper we attempt to answer this question for image denoising with a target on real-time performance. The energy model and the inference algorithm are no longer independent, so we consider them as parts of an Active Random Field, and their parameters are learned so that they work best together to obtain the desired results. For the image denoising application, we use the Fields of Experts [29] Markov Random Field (MRF) model and a 1-4 iteration gradient descent inference algorithm. The algorithm is restricted to be 1000-3000 times faster than the one previously used for image denoising and the best model-algorithm parameters are trained using a dataset of training pairs consisting of input images corrupted with noise and the desired denoised output (the images without the noise). A comprehensive evaluation on 68 standard benchmark images that were not used for training revealed that the trained model-algorithm combination obtains improved denoising performance compared to the equivalent MRF model while being thousands of times faster.

Section II presents an overview of Markov Random Fields, Energy Based Models and introduces the Active Random Field concept. Section III applies the Active Random Field to image denoising using the Fields of Experts model, presenting a detailed overview of the training procedure and results. Finally, Section IV presents conclusions and future directions.

A shorter version of this paper appeared in CVPR [2].

A. Barbu is with the Department of Statistics, Florida State University, Tallahassee, Florida 32306, USA, Phone: 850-644-6688, Fax: 850-644-5271, Email: abarbu@stat.fsu.edu.

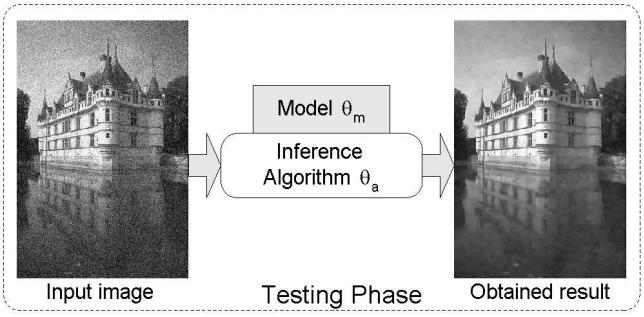

Fig. 1. The Markov Random Field model makes use of an inference algorithm to solve a given task (image denoising in this example).

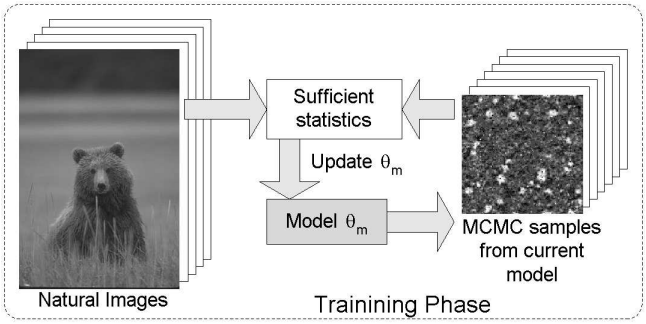

Fig. 2. The Markov Random Field model is usually trained independent of the inference algorithm. The impact of training the model with the inference algorithm will be studied in this paper.

II. Active Random Field: Jointly Training the Model with a Fast and Suboptimal Inference Algorithm

Markov Random Fields (MRF) are used extensively in many areas of computer vision, signal processing and beyond. They are capable of enforcing strong regularization on the desired results.

A. Overview of Markov Random Fields and Conditional Random Fields

Let $G = (V, E)$ be a graph with nodes V and edges E, $\mathbf{x} = (x_v)_{v \in V}$ be a set of random variables representing some hidden attributes (e.g. labels) of the graph nodes $v \in V$, and C be a set of cliques (fully connected subgraphs) of G. In a Bayesian framework, the posterior probability of the hidden variables $\mathbf{x}$ given input data (image, signal) $\mathbf{y}$ is

$$P(\mathbf{x}|\mathbf{y}) \propto P(\mathbf{y}|\mathbf{x})P(\mathbf{x}) \tag{1}$$

The Markov Random Field (C, ϕ) models the prior on the hidden variables $\mathbf{x}$

$$P(\mathbf{x}) = \frac{1}{Z} \exp[\sum_{c \in C} \phi_c(\mathbf{x}_c)] \tag{2}$$

where $\phi_c(\mathbf{x}_c)$ are potential functions that enforce the regularization between the variables $\mathbf{x}_c$ corresponding to the clique c. The cliques can be as small as graph edges (order 2), however larger cliques are preferred, since they are capable of representing more complex relationships.

In our denoising application, the graph G is the pixel lattice and the clique set C contains all the 5×5 pixel patches of the image, thus each clique $c \in C$ contains 25 nodes.

Quite recently, Conditional Random Fields (CRF) [18], [17] were developed as an extension of the MRF so that the clique potentials depend on the observed data $\mathbf{y}$. A CRF is also a pair (C, ϕ) with ϕ depending on $\mathbf{y}$, aimed at directly modeling the posterior $P(\mathbf{x}|\mathbf{y})$ (thus the task that is being solved).

$$P(\mathbf{x}|\mathbf{y}) = \frac{1}{Z(\mathbf{y})} \exp[\sum_{c \in C} \phi_c(\mathbf{x}_c, \mathbf{y})] \tag{3}$$

The MRFs and CRFs have the following advantages and disadvantages:

+ They are capable of encoding complex relationships between the graph attributes $\mathbf{x}$ resulting in flexible yet powerful models
- Inferring the optimal state is computationally demanding. For example, the exact inference is NP hard [5] even for one of the simplest pairwise MRF priors: the Potts model [28]. Hence, approximate solutions are used in practice.
- The MRF is difficult to train, since the normalization constant Z is needed to comparing different MRF models.
- The the MRF/CRF is always used with an inference algorithm, as shown in Figure 1. However, the MRF/CRF is usually trained independently of the inference algorithm, through a procedure illustrated Figure 2. We will observe that by training the MRF/CRF together with the inference algorithm, significant improvements in both speed and accuracy can be obtained.

B. Energy Based Models and Loss Functions

Recent work on Energy Based Models [1], [19], [25], [35] deals with the normalization constant by training the MRF parameters θ so that the MAP estimates are as similar as possible to the corresponding desired outputs. The differences between the MAP estimates $\mathbf{x}_i$ and the desired outputs $\mathbf{t}_i$ are measured using a loss function $L(\mathbf{x}_i, \mathbf{t}_i)$ and the training procedure for the Energy Based Models can be written as:

$$\min_\theta \sum_i L(\mathbf{x}_i, \mathbf{t}_i), \text{ with } \mathbf{x}_i = \arg\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y}_i; \theta) \tag{4}$$

This approach eliminates the need to compute the normalization constant by comparing models using the loss function. However, these methods still deal with an idealized situation, since in reality the minimum energy MRF point is often too expensive to compute (e.g. NP-hard for the Potts model) obtaining a suboptimal point instead.

C. Active Random Fields

Since most fast inference algorithms obtain a sub-optimal solution anyway, we follow [39] and propose a different approach in which the model parameters are trained such that the inference algorithm output (and not the "ideal" MAP estimate as in the Energy Based Models) is close to the desired output. This way, the suboptimal inference algorithm is involved in the parameter learning phase. This combined approach can be written as:

$$\min_\theta \sum_i L(\mathbf{x}_i, \mathbf{t}_i), \quad \text{with } \mathbf{x}_i = A(\mathbf{y}_i, \theta) \tag{5}$$

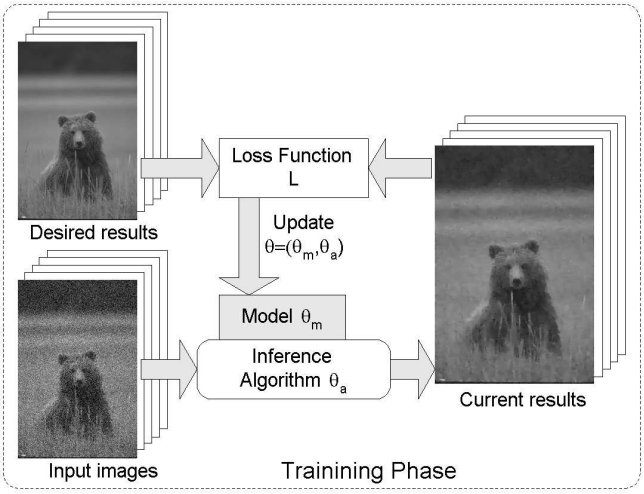

Fig. 3. An Active Random Field is a MRF/CRF model trained together with a fast inference algorithm using pairs of input and desired output as training examples. Training is achieved by optimizing a loss function that evaluates how well the given model+algorithm combination solves the given task (image denoising in this example).

where $\mathbf{x} = A(\mathbf{y}, \theta)$ is the result of the algorithm A with the model and algorithm parameters $\theta = (\theta_m, \theta_a)$ on the input image $\mathbf{y}$. As with the Energy Based Models from Section II-B, the training data consists of pairs $(\mathbf{y}_i, \mathbf{t}_i)$ consisting of input images $\mathbf{y}_i$ and the corresponding desired outputs $\mathbf{t}_i$. This approach is illustrated in Figure 3.

Since the MRF model and the inference algorithm are now inseparable, we define an *Active Random Field* (ARF) as a triplet (C, ϕ, A) consisting of a MRF or CRF (C, ϕ) together with an inference algorithm $A \in \mathcal{A}$. The algorithm A is selected from a family of algorithms $\mathcal{A}$ that provides inference on the input data $\mathbf{y}$ using the model (C, ϕ). The algorithm family $\mathcal{A}$ can include any type of algorithm that can be used for MRF/CRF inference: gradient descent, Belief Propagation [44], Graph Cuts [5], etc. However, in contrast to the standard MRF/CRF approaches, the algorithms in the family $\mathcal{A}$ are restricted to be very fast, by sacrificing accuracy. For example, the number of gradient descent iterations in our image denoising application is kept small, on the order of 1 to 4, as opposed to 3000-10000 iterations used in [29]. The inaccuracy of the algorithm is compensated by training the model to give best results on this algorithm, resulting in a fast and accurate combination.

The performance of the Active Random Field is measured using a loss function L that is a generally accepted benchmark in the community. In image denoising we use the average PSNR (peak signal-to-noise ratio) over the set of images (training or testing) and replace the minimization in Equation (5) with a maximization. Other more appropriate loss functions could be used instead of the PSNR, for example the Structural Similarity Index (SSIM) [40].

The differences from the standard MRF/CRF approaches and the proposed ARF approach are

1) The normalization constant Z is not important in the ARF since different models are compared using the loss function L instead of the likelihood or posterior probability.
2) The training set consists of pairs of input images and desired results. With the loss functions, this avoids the need for sampling from the learned distribution as in the MRF/CRF training. The new training approach gives a better idea on when the training is completed or whether overfitting occurs.
3) The trained model and algorithm complement each other and result in a fast and accurate system.
4) The MRF/CRF are just models that are always used with the help of an inference algorithm. On the other hand, the ARF is a trained model+algorithm combination that given an input, returns a result, thus it is a full computational solution.

D. Related Work

In the literature, a substantial amount of work combines models with algorithms in different ways. Active Appearance Models [8] are iterative algorithms driven by data and a PCA-like model to find objects of interest in the image. The solution depends on the starting location, so they are usually used in cooperation with other algorithms or with user initialization. A more complete solution for object or shape detection is offered by the Shape Regression Machine [46], where an image based regression algorithm is trained to find a vector toward the object of interest from any random location inside the image. Fast and robust object detection is obtained by using hundreds of random initializations and a verification step based on Adaboost. The Shape Regression Machine can thus be seen as a trained model-algorithm combination for object or shape detection. Our work differs from the Regression Machine because it is aimed at training models and algorithms for MRF/CRF inference instead of object/shape detection. Another related work is [12], learning detectors for faces and face parts by exploiting the context between them, but without an explicit MRF formulation.

The ARF resembles the *energy based models* [19], [25], in that only the energy part of the MRF is used without the normalization constant, and a loss function is used for training. The energy based models are models trained in such a way that the minimum energy is at the desired location on the training set, independent of the optimization (inference) algorithm used. In order for that to happen, specific conditions on the energy function are imposed [19]. By contrast, the ARF training finds the model parameters that give best results on a training set using a preselected inference algorithm. As a consequence, no conditions on the energy function or the loss function are imposed on the ARF. The applicability of the ARF is only limited by the existence of a fast inference algorithm that obtains results in a matter of seconds, since it will have to be applied many times during training.

A number of works use the model-algorithm combination for learning the model, but without imposing any computational complexity constraints. In this category is [36], where a CRF based on pairwise potentials is trained for object classification using boosting and a pixelwise loss function. On a similar note, [37] trains a sequence of classifiers for object segmentation. Each classifier is based on features from the data

and on the probability map obtained so far. These two methods train MRF model-algorithm combinations that slowly decrease in speed at each training iteration, because the models become more and more complex. In [30], an approximate posterior was maximized by gradient optimization for learning a pairwise MRF for stereo matching.

There exist a number of works that train model-algorithm combinations with a loss function that is used to report the results. However, these works use inference algorithms that are focused on exact MAP estimation, which is different than what is proposed in this paper. The same quantity from Eq. (5) is minimized in [34] for image denoising, but as an attempt to obtain a stronger MRF optimum than the gradient descent. For that, a more complex inference algorithm, based on variational optimization, is derived. On a similar note, in [33] Gaussian Conditional Random Fields are defined and used for image denoising. They allow exact computation of the MAP solution as well as an analytic gradient of a loss function (the MSE) comparing the solution and the desired result. The analytic computation of the MAP solution and of the gradient are possible by making some compromises in the model (the GCRF). The results presented in [33] are comparable to the two iterations of ARF though obtained at least one hundred times slower. We show in Section III-F that MAP estimation for the Fields Of Experts MRF model can be obtained by a sequence of GCRF estimations. Finally, a model-algorithm combination for optical flow was trained in [20] using stochastic optimization and a loss function based on the average endpoint error. The MRF model was based on 3-cliques and inference was obtained by limited memory BFGS [21]. A common theme in these works is the fact that all use a lot of computation in the inference algorithm for obtaining an strong MRF optimum. This paper differs in this regard by using a fast (close to real-time) and suboptimal inference algorithm which does not try to obtain a strong optimum. We argue in this paper that it is more important to prevent overfitting than to obtain a strong MRF optimum.

Even when using a fast inference algorithm such as one iteration of gradient descent, through appropriate training and with a complex and flexible enough model, the model will adapt to the simple descent algorithm as predicted by [39]. Consequently, the image denoising results presented in this paper surpass any previous results based on MRF models in both speed and accuracy.

Similar goals in obtaining good results with low computational expense are explored in cost-sensitive learning. In [38], a decision tree was trained to minimize a cost function with terms for accuracy and computational expense for each feature. Also related is [42], where for each instance of the well-known SAT problem, the most efficient algorithm is selected from a pool of SAT solvers using regressors that estimate the algorithm running time. These regressors have been trained beforehand on a dataset of SAT instances.

In general, parameter tuning for a specific application based on a training dataset can be viewed as related work, but we are unaware of any work specifically aimed at studying parameter tuning and ways to prevent overfitting.

E. Training the Active Random Field

Training of the Active Random Field, is achieved using examples in the form of pairs $(\mathbf{y}_i, \mathbf{t}_i)$ of the observed images $\mathbf{y}_i$ and the corresponding desired outputs $\mathbf{t}_i$. Given a training set $T = \{(\mathbf{y}_i, \mathbf{t}_i), i = 1, ..., n\}$ consisting of such pairs, the loss function $L(\mathbf{y}, \mathbf{t})$ is used to evaluate how well the model and algorithm solve the given problem on this training set.

If the model-algorithm combination is parametrized by $\theta = (\theta_m, \theta_a)$, the training is an optimization procedure to find

$$\theta = \arg\min_{\theta} \sum_{i=1}^{n} L(A(\mathbf{y}_i, \theta), \mathbf{t}_i) \tag{6}$$

Depending on the problem, different optimization algorithms (coordinate descent, conjugate gradient, simulated annealing, genetic algorithm, etc) could be appropriate.

There are two main concerns regarding this Active Random Field approach.

1) The main concern is overfitting the training data. This happens when an increased performance on the training data is reflected in a decreased performance on an unseen dataset. Overfitting can be detected using a validation set and appropriate measures can be taken. Possible measure include increasing the number of training examples or changing the type of the training examples (e.g. larger images to avoid boundary effects).
2) Another concern is the computational complexity of the applying the algorithm on all the training examples for each optimization iteration. This concern is addressed in three ways. First, for certain problems, different design strategies (e.g. memorization of partial results) can be used to reduce the computation to a fraction of the full evaluation cost. Second, efficient optimization algorithms such as conjugate gradient or genetic algorithms, can make good use of each function evaluation. Third, the computational demand is less of an issue every day due to the exponential growth in computational power of a standard PC. Even though the CPU frequency has reached a limit recently, the number of CPU cores in a standard PC still increases exponentially. Furthermore, the training can be easily parallelized, resulting in a good utilization of all available computing power.

III. Application: Image Denoising

We apply the ARF idea to image denoising, where given an image corrupted with noise, the goal is to obtain an image from which the noise was removed. This problem has been addressed using wavelets in [27], [26] and by learning a MRF prior model known as Fields of Experts on 5×5 pixel cliques in [29]. Non-local image denoising methods include [6] and especially 3D collaborative filtering (BM3D) [9], the latter obtaining very good results with low computational expense. An example of an image denoising problem and results obtained using the above mentioned methods as well as the ARF approach proposed in this paper are shown in Figure 4, together with the CPU time required to obtain each result. Another approach [10] uses a sparse representation based on a learned dictionary of primitives and is more computationally expensive.

Fig. 4. Image denoising example. Top, from left to right: original image, image corrupted with additive Gaussian noise with $\sigma = 25$, PSNR=20.17; our result, PSNR=28.94, 0.6 seconds and Fields of Experts result [29], PSNR=28.67, 2280 seconds. Bottom results, from left to right: wavelet denoising [27], PSNR=29.05, 16 seconds; overcomplete DCT, PSNR=28.81, 38 seconds; KSVD [10], PSNR=29.02, 250 seconds and BM3D [9], PSNR=29.60, 4.3 seconds.

The ARF approach to image denoising proposed in this paper uses the Fields of Experts MRF model and the gradient descent algorithm that were presented in [29] and will be briefly mentioned in the next section. The loss function used for training the ARF is the average PSNR (Peak Signal to Noise Ratio) over the training set.

A. Fields of Experts

The Fields of Experts [29] is a Markov Random Field prior model with potential functions based on a collection of convolution kernels (filters) $J_f, f = 1, ..., N$ and coefficients $\alpha_f, f = 1, ..., N$

$$\begin{aligned} p_{FOE}(\mathbf{x}, \theta) &= \frac{1}{Z(\theta)} \exp(-E_{FOE}(\mathbf{x}, \theta)), \\ E_{FOE}(\mathbf{x}, \theta) &= \sum_k \sum_{f=1}^{N} \alpha_f \log(1 + \frac{1}{2}(J_f^T \mathbf{x}^{(k)})^2) \end{aligned} \tag{7}$$

The first sum is taken over the cliques k of the denoised image $\mathbf{x}$, and $\mathbf{x}^{(k)}$ are the pixels of $\mathbf{x}$ corresponding to clique k. There is a clique centered at each pixel location inside the image. Basically, each expert is a convolution followed by a robust potential function.

A convolutional approach is also taken in the FRAME model for texture modeling [47]. This is a Maximum Entropy Model with learned potential functions and convolutions of the image with predefined filters such as Laplacian of Gaussian, Gabor, etc. The difference is that in the FRAME model the convolution filters are predefined and the potential functions are learned, while in the FOE the potential functions are fixed and the convolution filters are learned.

For image denoising, this prior is used together with a likelihood that assumes i.i.d. Gaussian noise:

$$p(\mathbf{y}|\mathbf{x}) \propto \exp(-E_{data}(\mathbf{x}|\mathbf{y})),\ E_{data}(\mathbf{x}|\mathbf{y}) = \frac{1}{2\sigma^2} \sum_j (\mathbf{y}^j - \mathbf{x}^j)^2 \tag{8}$$

where $\mathbf{x}^j$ is the value of pixel j of image $\mathbf{x}$.

The beauty of the Fields of Experts formulation consists of an analytical solution for the gradient of the energy with respect to $\mathbf{x}$.

$$\begin{aligned} \nabla_{\mathbf{x}} E_{FOE}(\mathbf{x}, \theta) &= \sum_{f=1}^{N} \alpha_f J_f^- * \frac{J_f^T \mathbf{x}}{1 + \frac{1}{2}(J_f^T \mathbf{x})^2} \\ \nabla_{\mathbf{x}} E_{data}(\mathbf{x}|\mathbf{y}) &= \frac{1}{2\sigma^2}(\mathbf{x} - \mathbf{y}) \end{aligned} \tag{9}$$

where J_f^- is the mirror image of filter J_f around its center pixel.

Given a noisy image and learned parameters θ, the denoising is obtained by gradient descent in the energy $E_{data}(\mathbf{x}|\mathbf{y}) + E_{FOE}(\mathbf{x}, \theta)$. Thus, by taking small steps in the direction of the energy gradient, a denoised image $\hat{\mathbf{x}}$ is obtained in about 3000 iterations. For more details, see [29].

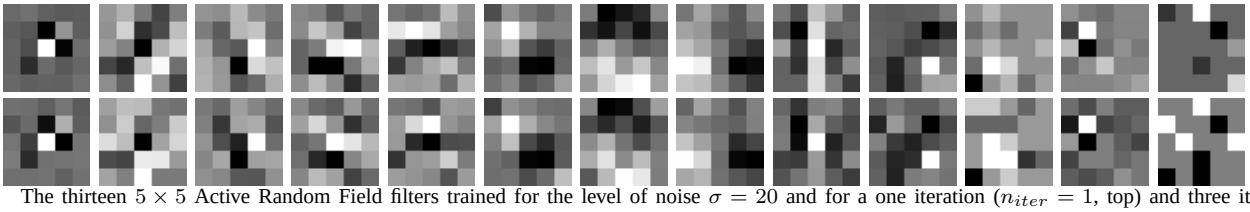

Fig. 5. The thirteen 5×5 Active Random Field filters trained for the level of noise $\sigma = 20$ and for a one iteration ($n_{iter} = 1$, top) and three iteration ($n_{iter} = 3$, bottom) steepest descent inference algorithm.

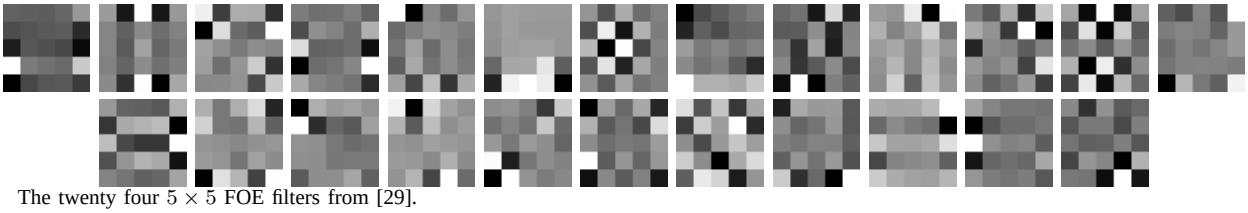

Fig. 6. The twenty four 5×5 FOE filters from [29].

B. Active Random Field Approach

In the Fields of Experts formulation, the model (MRF prior +likelihood) is trained independently from the MAP inference algorithm (gradient descent). In what follows, they will be trained together in a joint optimization.

For image denoising with Active Random Fields, we use the model and family of algorithms $\mathcal{A}$ from the Fields of Experts formulation presented above. By ignoring the normalization constant, from the gradient equation (9) we obtain the iterative gradient descent inference algorithm that is used for MAP estimation.

$$\mathbf{x} \leftarrow \mathbf{x} + \delta\big[\frac{\beta}{2\sigma^2}(\mathbf{x}-\mathbf{y}) + \sum_{f=1}^{N} \alpha_f J_f^- * \frac{J_f^T\mathbf{x}}{1+\frac{1}{2}(J_f^T\mathbf{x})^2}\big] \quad (10)$$

These iterative algorithms from equation (10) form an algorithm family $\mathcal{A}$, parametrized by N convolution kernels $J_f, f = 1, ..., N$ with corresponding coefficients α_f, the data coefficient β, the number n_{iter} of gradient update iterations (10), and the update parameter δ. Therefore

$$\theta = (\theta_m, \theta_a) = (N, J_1, \alpha_1, ..., J_N, \alpha_N, \beta, n_{iter}, \delta). \quad (11)$$

When training for a particular noise level σ, we observed a very modest contribution of at most 0.01dB of the data term $\frac{\beta}{2\sigma^2}(\mathbf{x}-\mathbf{y})$ to the final result. Hence we keep $\beta = 0$ until section III-E.

In our approach, instead of taking $n_{iter} = 3000$ iterations with small steps ($\delta = 0.2$) as in the FOE model, the algorithms in the family $\mathcal{A}$ have a small number of iterations $n_{iter} \in \{1, 2, 3, 4\}$ with $\delta = 400/n_{iter}$. Since the number of iteration is small, the result is obtained between 800 and 3000 times faster than the FOE. At the same time we observe that the denoising performance actually increases compared to FOE for an appropriately trained system.

C. Training the Active Fields of Experts

In [29], the Fields of Experts model is trained using Contrastive Divergence [15] and Markov Chain Monte Carlo sampling. The procedure involves gradient descent in the parameter space to minimize the KL divergence between the model probability and the empirical prior probability obtained from the training examples. The parameters are updated based on expected values with respect to the current probability distribution, obtained using MCMC sampling. The training procedure is computationally intensive and yields a generic prior model for natural images.

In [34], the same FOE model is used and trained using a loss function and stochastic gradient descent. With the help of a family of upper bounds of the nonlinear function $\log(1+x^2)$, another inference algorithm is obtained, with the hope that it can obtain a stronger optimum than the gradient descent (10).

In what follows, we will show that this is not necessary, since by appropriately training the ARF (i.e. the FOE model together with the steepest descent algorithm), the model will adapt to make the simple gradient descent work very well, making it unnecessary to use a more powerful inference algorithm. This was predicted by Wainwright in [39] but the extent to which this statement is true is quite surprising.

1) Dataset: The same images as [29] are used for training, namely 40 natural images from the Berkeley dataset [22]. The training examples consist of the 40 pairs $(\mathbf{y}_i, \mathbf{t}_i)$ of input images $\mathbf{y}_i$ and desired results $\mathbf{t}_i, i = 1, ..., 40$. The desired results $\mathbf{t}_i$ are the original noise-free training images. The input images $\mathbf{y}_i$ are the original training images $\mathbf{t}_i$ corrupted with Gaussian noise of similar variance as expected at testing time. Since each training example contains $150,000$ cliques, the training set contains $6,000,000$ cliques. We experimented with smaller patches (e.g. of size 15×15 as in [29]) and observed that overfitting occurs when the patches are smaller than 250×250 pixels. This could be due to the boundary effect since the graph nodes close to the patch boundary don't have all the neighbors to communicate with and behave differently than the interior nodes.

For testing, we use the same 68 natural images from the Berkeley dataset as [29] as well as some standard image denoising test images. These testing images were not used for training.

2) Loss Function: The ARF is trained by optimizing the same criterion that is used for evaluating the denoising system performance, namely the average PSNR over the images in the set. Thus the loss function is

$$L(\mathbf{x}, \mathbf{t}) = 20 \log_{10}(255/std(\mathbf{t}-\mathbf{x})) \quad (12)$$

where $std(\mathbf{t}-\mathbf{x})$ is the standard deviation of the difference between the original image $\mathbf{t}$ and the denoised image $\mathbf{x}$. More

appropriate loss functions could be used instead of the PSNR, for example the Structural Similarity Index (SSIM) [40].

Learning is an optimization on the parameters θ to maximize

$$M(\theta) = \frac{1}{n}\sum_{i=1}^{n} L(A(\mathbf{y}_i, \theta), \mathbf{t}_i), \tag{13}$$

the average PSNR obtained after running the denoising algorithm $A(\mathbf{y}_i, \theta)$ with parameters θ on the 40 training examples $\mathbf{y}_i$.

3) Optimization: In this work, coordinate ascent was used for maximizing the loss function. Coordinate ascent is a greedy iterative optimization algorithm in which at each step, one of the variables θ_i of the current state θ is chosen at random and its value is modified by a small amount (0.0001 to 0.001 in our experiments) if $M(\theta)$ does not decrease. If the $M(\theta)$ decreases, the variable θ_i is rolled back to its old value. For our problem, each filter is constrained to have a zero-sum so we modified the coordinate ascent so that when a filter is selected to be modified, two locations inside the filter are chosen randomly and modified by the same small amount, but with opposite signs. This way the filters always remain zero-sum.

We also experimented with gradient ascent, conjugate gradient and the simplex method [24]. For this particular application, we observed that these other methods could not find such a strong optimum as the coordinate ascent. This is probably because the optimum path is very narrow and a fast algorithm could not follow it properly. Other optimization methods such as genetic algorithms [14] or simulated annealing [16] could be more appropriate for avoiding local optima and are subject to further investigation.

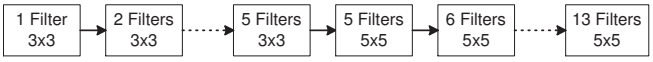

Fig. 7. Training diagram for Active Random Field parameters for the level of noise $\sigma = 25$ and the one iteration ($n_{iter} = 1$) steepest descent inference algorithm.

The one iteration parameters were trained first, for the level of noise $\sigma = 25$. For the one iteration parameters, the coefficients α_f can be well approximated analytically as the solution of the least squares problem:

$$\sum_{i=1}^{40} ||\mathbf{t}_i - \mathbf{x}_i - \delta \sum_{f=1}^{N} \alpha_f \mathbf{g}_f^i||^2, \quad \text{where}$$
$$(\mathbf{g}_f^i)^j = J_f^- * \frac{J_f^T \mathbf{x}_i^{(j)}}{1 + \frac{1}{2}(J_f^T \mathbf{x}_i^{(j)})^2} \tag{14}$$

This leaves only the value of the filters $F_f, f = 1, ..., N$ for optimization. At each step of the optimization, the coefficients α_f are obtained by solving the above least squares problem and then $M(\theta)$ is evaluated. This technique is know as *Rao-Blackwellization* [4], [7].

Since the function $M(\theta)$ is not convex, the optimization is prone to be stuck in local maxima. To alleviate this problem, the one iteration filters for $\sigma = 25$ are trained using a simplified version of *Marginal Space Learning* [45]. Marginal Space Learning is an optimization procedure aimed at finding optima in high dimensional spaces by propagating a set of particles in a sequence of spaces of increasing dimensions until the full parameter space is reached. In our case, a single particle (maximum) is propagated starting from the small dimensional space of parameters of only one filter and gradually increasing the dimensionality by adding filters or by increasing the filter size. Each time the dimensionality is increased, the position of the particle in the larger Marginal Space is searched by Coordinate Ascent.

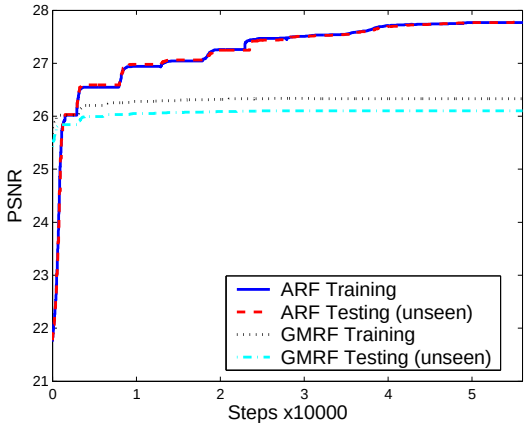

Fig. 8. PSNR evolution on the training and test set observed while training the one iteration ($n_{iter} = 1$) ARF parameters for the level of noise $\sigma = 25$. Also displayed in dotted lines are the training and testing PSNR of a GMRF with the same model complexity, described in section III-G.

More specifically, the Marginal Space Learning procedure is started with one filter of size 3×3 with all entries 0 except on the first row, $F_1(1,1) = 0.1, F_1(1,2) = -0.1$. Starting from this initial setting, the PSNR optimization was run until not much improvement in $M(\theta)$ was observed. This is the location of the particle in the first Marginal Space. Then the parameter space was enlarged by adding another filter with all entries 0 and optimizing for 3000 steps, obtaining the particle position in the second space. The process of increasing the Marginal Space by adding one filter and retraining was repeated until there were a total of five 3×3 filters. Then the Marginal Space was enlarged by increasing the filter size to 5×5 by padding zeros on the border of each filter. The new position of the particle (maximum) was searched through 3000 steps of optimization. The process of enlarging the Marginal Space by adding filters (now of size 5×5) and retraining was repeated until the number of filters reached $N = 13$. This

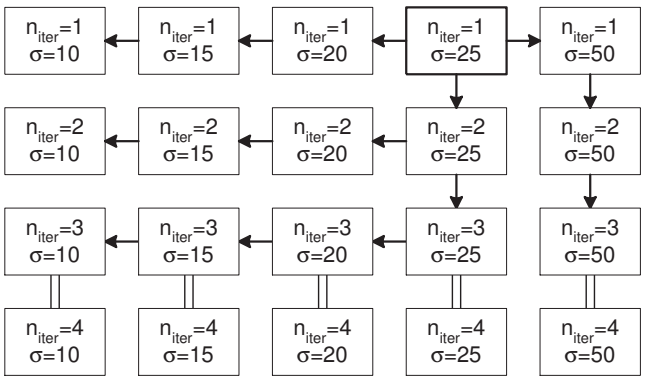

Fig. 9. Diagram of the training of the ARF parameters for different levels of noise and numbers of iterations of the steepest descent inference algorithm. The double lines mean that the filters are the same.

TABLE I
PERFORMANCE EVALUATION AND COMPARISON OF OUR METHOD (1-4 ITERATIONS) WITH OTHER METHODS ON SOME STANDARD BENCHMARK IMAGES, $\sigma = 25$. THE ARF RESULTS ARE OBTAINED 7-3000 TIMES FASTER.

Image	Lena	Barbara	Boats	House	Peppers	Average
Fields of Experts [29]	30.82	27.04	28.72	31.11	29.20	29.38
1-iteration ARF	30.15	27.10	28.66	30.14	28.90	28.99
2-iteration ARF	30.66	27.49	28.99	30.80	29.31	29.45
3-iteration ARF	30.76	27.57	29.08	31.04	29.45	29.58
4-iteration ARF	30.86	27.59	29.14	31.18	29.51	29.66
Wavelet Denoising [27]	31.69	29.13	29.37	31.40	29.21	30.16
Overcomplete DCT [10]	30.89	28.65	28.78	31.03	29.01	29.67
Globally Trained Dictionary [10]	31.20	27.57	29.17	31.82	29.84	29.92
KSVD [10]	31.32	29.60	29.28	32.15	29.73	30.42
BM3D [9]	32.08	30.72	29.91	32.86	30.16	31.15

number was chosen by observing on the validation set that no further improvement in PSNR could be obtained. The whole procedure is illustrated in Figure 7.

The evolution of the PSNR over all this training, starting with one 3×3 filter and ending with thirteen 5×5 filters is plotted in Figure 8. Training the 5 filters of size 3×3 takes about 7 hours on a dual-core 2.4Ghz PC while the whole training for the one iteration $\sigma = 25$ filters takes about 3 days.

Since the optimization is prone to be stuck in local optima, the other filters are initialized from already trained filters in the order presented in Figure 9. The 3-iteration filters are well trained to perform iterative denoising and can also be used for 4-iterations without any modifications.

Training each of the arrows in Figure 9 takes about one day on a 8-core 2Ghz PC. We believe that by using better optimization algorithms, the training time can be further improved. The trained 5×5 filters for $\sigma = 20$ and $n_{iter} = 3$ are shown in Figure 5.

4) Overfitting: As already mentioned, we initially used images of size 15×15 as in [29] and observed that the training PSNR was increasing significantly while the PSNR on the validation set was actually decreasing, a sign of overfitting. We experimented with increasing the size of the training images and observed that this alleviated the overfitting problem. Finally we observed that when the training images were at least 250×250, there were no signs of overfitting.

D. Results

The performance of the Active Random Field system is evaluated on the same datasets as [29]. First, results on some standard images - Lena, Barbara, Boats, House and Peppers - at the noise level $\sigma = 25$ are shown in Table I. Note that these images were not used for training. The ARF results are obtained between 7 and 3000 times faster than the other methods.

For further comparison, Table II and Figure 10 present results on the same 68 test images from the Berkeley dataset as [29]. Note that these images were also not used for training. We present results for 1: Wiener filter, 2: Nonlinear Diffusion [41] using the nonlinear diffusion Matlab toolbox provided by Frederico D'Almeida, with parameters $\lambda = 8$, $\sigma = 1$, diffusivity $= 3$, step size $= 8$, steps $= 2$, and with the `aos` option, 3: Non-local means [6] with search window 17×17, similarity window 9×9 and h tuned for best results for each σ, 4: Fields of Experts (FOE) [29] with 3000 iterations, 5,6,7,8: our algorithm with 1,2,3,4 iterations, 9: wavelet based denoising [27], 10: Overcomplete DCT [10], 11: KSVD [10] and 12: BM3D [9]. Since this evaluation is on 68 images, it should be regarded as a more thorough evaluation than the results on 5 specific images. We should mention that all other algorithms were run on Matlab code provided by their authors and were not implemented by us.

From the evaluation, it is clear that the one iteration ARF is on par with the FOE while being 3000 times faster. Therefore, training the MRF model together with a suboptimal inference algorithm offers significant advantages in speed and accuracy. One could also observe that the ARF is within 0.5dB from the best method and it is outperformed by two methods: KSVD [10], BM3D [9] and for some noise levels by wavelet denoising [27] and overcomplete DCT [10].

Depending on application, trade-offs between speed and accuracy might be important. Figure 11 shows a plot of the PSNR performance in dB of the algorithms compared above as a function of the processing speed in fps. From the figure, one can see that the Active Random Fields are very competitive candidates when high processing speeds are required such as in real-time medical applications.

The computation complexity of the ARF image denoising algorithm is due to the necessity of performing $2N$ convolutions (where N is the number of filters) for each iteration. A standard Matlab implementation takes about 0.8s for each iteration on a 256×256 image and a 2.4GHz PC. A better C++ implementation using IPL (Intel Image Processing Library)

TABLE II
PERFORMANCE EVALUATION OF DIFFERENT DENOISING METHODS ON 68 IMAGES FROM THE BERKELEY DATASET. AVERAGE PSNR OF THE DENOISING RESULTS OBTAINED BY THE METHODS AT DIFFERENT NOISE LEVELS.

Level of Noise σ	10	15	20	25	50
1. Wiener Filter	31.65	29.18	27.53	26.37	22.94
2. Nonlinear Diffusion [41]	32.03	29.83	28.28	27.25	24.73
3. Non-local [6]	31.48	29.86	28.62	27.59	24.22
4. Fields of Experts [29]	32.68	30.50	28.78	27.60	23.25
5. 1-iteration ARF	32.74	30.57	28.92	27.77	24.58
6. 2-iteration ARF	32.74	30.70	29.23	28.10	24.88
7. 3-iteration ARF	32.84	30.76	29.29	28.17	25.11
8. 4-iteration ARF	32.82	30.76	29.33	28.24	25.14
9. Wavelet Denoising [27]	33.05	30.73	29.18	28.03	25.37
10. Overcomplete DCT [10]	33.19	30.75	29.15	27.98	24.86
11. KSVD [10]	33.30	30.96	29.43	28.33	25.20
12. BM3D [9]	33.53	31.21	29.71	28.63	25.47

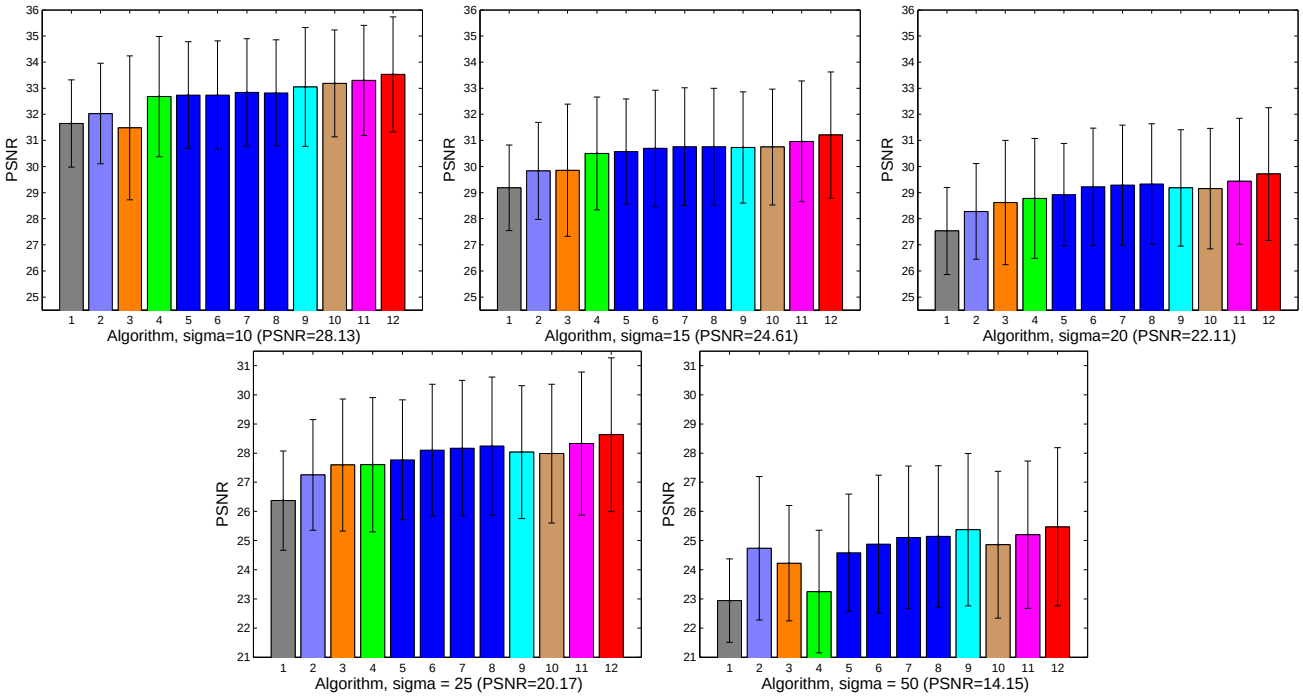

Fig. 10. Average PSNR in dB for different image denoising algorithms at different noise levels on 68 images from the Berkeley dataset. 1: Wiener Filter, 2: nonlinear diffusion, 3: Non-local means [6] 4: Fields of Experts [29], 5,6,7,8: our algorithm with 1,2,3 and 4 iterations, 9: wavelet based denoising [27], 10: Overcomplete DCT [10], 11: KSVD [10] and 12: BM3D [9]. The results are also shown in Table II.

cuts computation time to 0.12s per iteration for the same image size. Furthermore, a parallel implementation on multiple CPUs and/or a GPU implementation could bring this algorithm to real-time performance.

E. A Study of the One Iteration Algorithms

It is intriguing that such results could be obtained in a single gradient descent iteration. Could this be due to the specially trained filters F_j or to the filter coefficients α_j? In this section we perform more experiments on different one iteration algorithms to determine the cause of this performance.

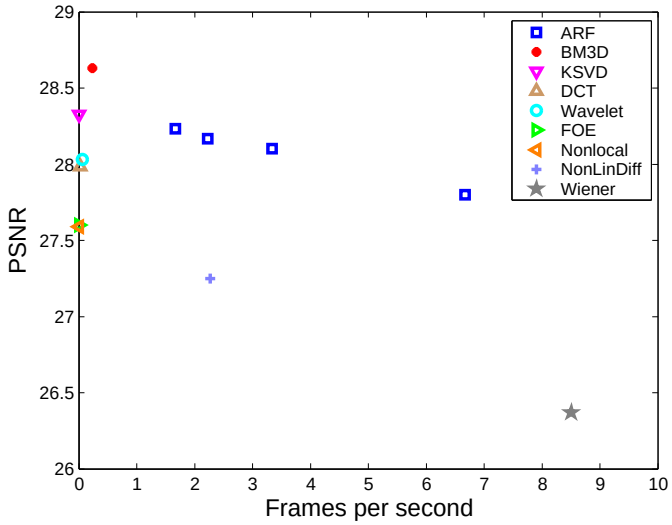

Fig. 11. PSNR (dB) vs speed (fps) of different denoising algorithms on the 68 Berkeley test images for $\sigma = 25$. The colors are the same as in Figure 10.

First, we plot in Figure 12 the performance of the 1-iteration ARF in the range $\sigma \in [10, 50]$. For comparison, the FOE performance is also displayed. It is clear that the ARF algorithms are specialized at the levels of noise they were trained for.

Then we evaluated the FOE model with a single iteration gradient descent algorithm on the same 68 images from the Berkeley dataset, choosing the step size δ to maximize the PSNR. The results are presented in the second row in Table III and in Figure 13.

Another 1-iteration FOE model was obtained by retraining the coefficients α_j for each σ to maximize the PSNR on the training data, while keeping the original filters F_j. These results are posted in the third row in Table III and in Figure 13. For each algorithm are also displayed in Table III the number of parameters that are trained for each noise level σ.

The fourth row of the table displays an evaluation of the FOE filters when the filter norms ν_j (i.e. scalar multipliers $F_j = \nu_j F_j^{FOE}$) and their coefficients α_j were trained for each value of σ by maximizing the PSNR on the training set.

The fifth row in Table III presents the performance of a 1-iteration ARF that was trained on images corrupted with noise levels $\sigma = 15$ and 25 and no data term. Observe that the performance decreased slightly while the noise range was extended.

In the 1-iteration ARF, the data term has theoretically no influence, since before the iteration $\mathbf{x} - \mathbf{y} = 0$. We slightly modified the 1-iteration ARF to a 2-step version that bears the same computational expense but can take into account the

TABLE III
TESTING ERRORS ON 68 UNSEEN BERKELEY IMAGES WHEN TRAINING DIFFERENT 1-ITERATION ARF ALGORITHMS.

Algorithm	Number of σ-dependent params	10	15	20	25	50
FOE [29]	1	32.68	30.50	28.78	27.60	23.25
1-iteration FOE	1	29.57	25.96	23.35	21.31	14.92
1-iter. FOE, retrained coeffs	24	30.23	26.63	24.00	21.92	15.39
1-iter. FOE, retrained coeffs & norms	48	32.29	29.73	27.99	26.69	22.39
ARF, no data term	0	30.13	29.89	28.99	27.40	18.69
ARF w/data term, trained with $\sigma \in [15, 25]$	1	31.99	30.37	28.99	27.63	20.26
ARF w/data term, train with $\sigma \in [15, 40]$	1	31.13	29.55	28.56	27.72	23.38

data term:

$$\begin{aligned} &1.\ \mathbf{x} \leftarrow \mathbf{y} + \delta \sum_{f=1}^{N} \alpha_f J_f^- * \frac{J_f^T \mathbf{y}}{1 + \frac{1}{2}(J_f^T \mathbf{y})^2} \\ &2.\ \mathbf{x} \leftarrow \mathbf{x} + \delta \frac{\beta_\sigma}{2\sigma^2}(\mathbf{x} - \mathbf{y}) \end{aligned} \tag{15}$$

where the data term has a coefficient β_σ that depends on σ, as in [29]. This can be also written in a single iteration as

$$\mathbf{x} \leftarrow \mathbf{y} + \delta \sum_{f=1}^{N} \alpha_f (1 + \frac{\beta_\sigma}{2\sigma^2}) J_f^- * \frac{J_f^T \mathbf{y}}{1 + \frac{1}{2}(J_f^T \mathbf{y})^2} \tag{16}$$

The last two rows of Table III and the red and green solid lines in Figure 13 show results obtained with this modified 1-iteration ARF. The first one is trained with images corrupted with noise levels $\sigma = 15$ and 25, while the second one with images corrupted with noise levels $\sigma = 15$ and 40. One can see that the band-pass behavior disappeared after the introduction of the data term.

The FOE with the coefficients and norms retrained at each noise level (fourth row in Table III) has a very good overall performance. However, compared to the ARF algorithms displayed in Table III, it has 48 times more parameters (24 norms and 24 coefficients) that are trained at each noise level.

F. Relation to M-estimation and Gaussian CRF

In this section we will use a standard statistical method named M-estimation [11], p. 99, which maps robust regression to an iterative weighted regression.

The FOE energy with a data term can be written as:

$$\begin{aligned} E_{FOE}(\mathbf{x}) &= \sum_{f=1}^{N} \sum_{j} \alpha_f \rho(J_f^T \mathbf{x}^{(j)}) + \beta(\mathbf{x} - \mathbf{y})^T (\mathbf{x} - \mathbf{y}) \\ &= \sum_{f=1}^{N} \sum_{j} \alpha_f \rho(J_{f,j}^T \mathbf{x}) + \beta(\mathbf{x} - \mathbf{y})^T (\mathbf{x} - \mathbf{y}) \end{aligned} \tag{17}$$

where $\rho(x) = \log(1 + x^2/2)$ is the Lorentzian and $J_{f,j}$ is the j-th column of J_f.

Thus minimizing $E_{FOE}(\mathbf{x})$ means solving a robust regression problem. One commonly used algorithm for robust regression is the iterative weighted regression using M-estimation, [11], p. 99.

Taking the derivative with respect to $\mathbf{x}^{(k)}$ and setting it to zero, gives:

$$\frac{\partial}{\partial \mathbf{x}^{(k)}} E_{FOE}(\mathbf{x}) = \sum_{f=1}^{N} \sum_{j} \alpha_f \rho'(J_{f,j}^T \mathbf{x}) J_{f,j,k} + 2\beta(\mathbf{x} - \mathbf{y}) = 0, \tag{18}$$

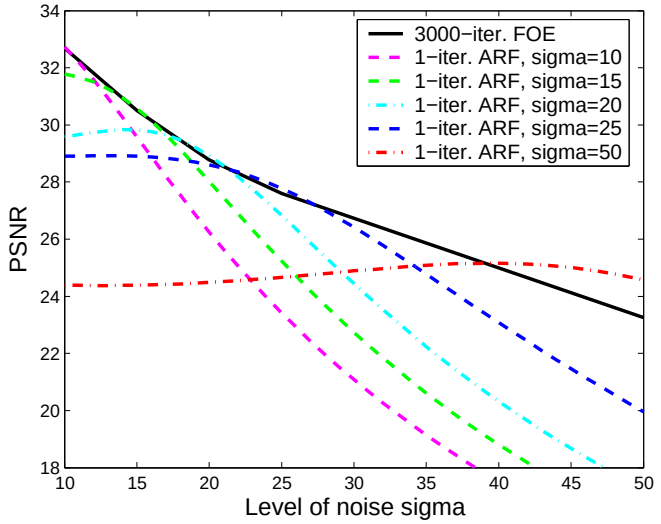

Fig. 12. PSNR (dB) curve of the 1-iteration denoising algorithms from Table II and Figure 10 for all noise levels in the range $\sigma \in [10, 50]$ on the 68 Berkeley test images. The 3000 iteration FOE algorithm was also included for comparison.

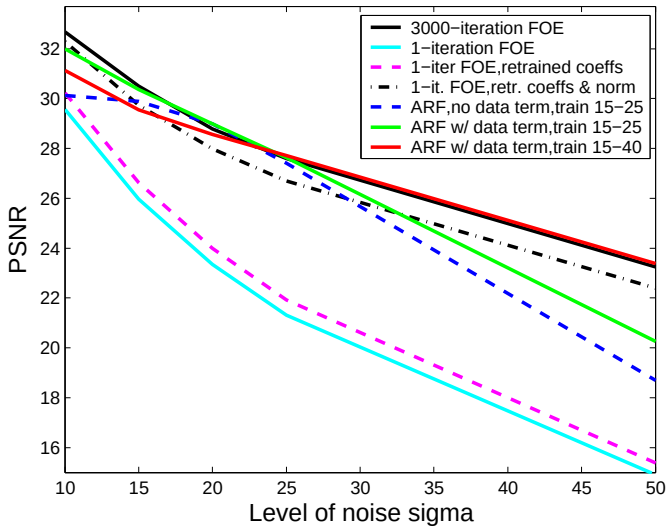

Fig. 13. PSNR (dB) curve of different 1-iteration denoising algorithms at different noise levels on the 68 Berkeley test images. The 3000 iteration FOE algorithm was also included for comparison.

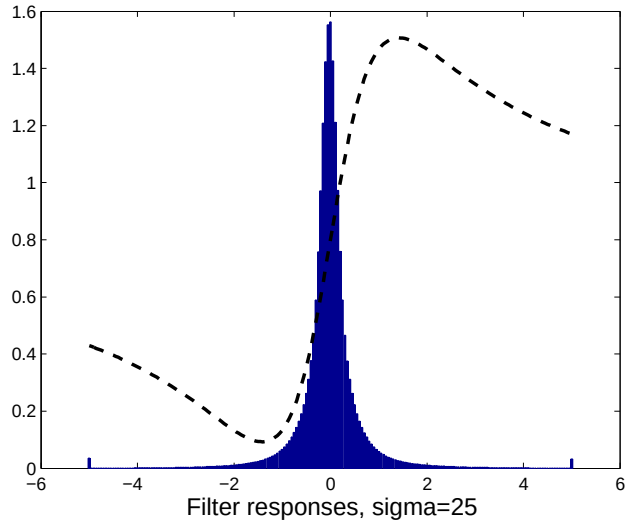

Fig. 14. Histogram of the filter responses over the training set for the one iteration algorithm for $\sigma = 25$. Approximately 8% of the responses fall outside the interval where the robust function $\rho(x) = \log(1+x^2/2)$ behaves like a gaussian, i.e. where the derivative $\rho'(x)$ (displayed as a dashed curve) is almost linear.

which can be written as

$$\sum_{f=1}^{N} \sum_{j} \alpha_f \frac{1}{1 + (J_{f,j}^T \mathbf{x})^2/2} J_{f,j,k} J_{f,j}^T \mathbf{x} + 2\beta(\mathbf{x} - \mathbf{y}) = 0 \quad (19)$$

By fixing $w_{j,f} = \alpha_f/(1 + (J_{f,j}^T \mathbf{x})^2/2)$, observe that (19) can be regarded as solving (minimizing) the weighted least squares

$$E_w(\mathbf{x}) = \sum_{f=1}^{N} \sum_{j} w_{jf} (J_{f,j}^T \mathbf{x})^2 + \beta(\mathbf{x} - \mathbf{y})^T(\mathbf{x} - \mathbf{y}) \quad (20)$$

This leads to a fixed point solution to the M-estimation problem by starting with an initial $\mathbf{x} = \mathbf{x}_0$ and iteratively minimizing (20) and updating the weights $w_{j,f} = \alpha_f/(1 + (J_{f,j}^T \mathbf{x})^2/2)$ based on the newly obtained $\mathbf{x}$.

Since the weighted sum of squares from eq. (20) is exactly the energy of a Gaussian Conditional Random Field [33]

$$E_{GCRF}(\mathbf{x}) = \sum_{f=1}^{N} \sum_{j} w_f(\mathbf{y})^{(j)} (J_f * \mathbf{x}^{(j)} - r_f(\mathbf{y})^{(j)})^2 \quad (21)$$

we obtain that the FOE energy can be minimized in an iterative way by alternatively minimizing a GCRF energy (20) and updating the weights as described above.

In the ARF approach presented in this paper, a simpler approach to minimizing the FOE energy is taken as a few iterations of gradient descent. This eliminates the need to solve costly least squares problems and it works very well.

G. Comparison to Gaussian MRF

Finally, to see how close the one iteration ARF is to exact inference on a Gaussian MRF with energy

$$E_{GRF}(\mathbf{x}) = \mathbf{x}^T \Sigma^{-1} \mathbf{x} + (\mathbf{x} - \mathbf{y})^T(\mathbf{x} - \mathbf{y}), \quad (22)$$

we get the exact minimum at

$$\hat{\mathbf{x}} = (2\Sigma^{-1} + I)^{-1} \mathbf{y} = A\mathbf{y}. \quad (23)$$

By limiting to only short-range interactions, (23) can be seen as a filtering operation

$$\hat{\mathbf{x}} = F * \mathbf{y}. \quad (24)$$

Comparing this to the one-iteration ARF equation e.g. (16), the Gaussian MRF solution above (24) is linear whereas the one iteration ARF is non-linear.

To see how close the one iteration ARF is to being a linear operator, we obtained a histogram of the filter responses over the test set, displayed in Figure 14. Approximately 8% of the filter responses fall outside the interval $[-1, 1]$ where the Lorentzian $\rho(x) = \log(1 + x^2/2)$ can be well approximated with a Gaussian, i.e. where the derivative $\rho'(x)$ (overlaid as a dashed curve in Figure 14) is approximately linear.

To compare how well the GMRF can denoise with the same model complexity, we trained it on the same 40 images from the Berkeley dataset. We used the same Marginal Space Learning approach as for training the ARF, but now there is only one filter that is started as the 1×1 identity filter and is enlarged by placing zeros on the border followed by 3000 iterations of coordinate ascent with the same PSNR loss function as the ARF. The enlarging and 3000 iteration optimization were alternated until the filter had size 19×19, which has slightly more parameters as 13 FOE filters of size 5×5. The PSNR on the training and test set are plotted in Figure 8. One could see that the Fields of Experts has better modeling power and it is better suited for image denoising, achieving more than 1.5dB improvement when compared to the GMRF.

IV. Discussion and Conclusions

Wainwright [39] predicted that in computationally restricted applications, models trained using MAP estimation might not be the best choice and biased models might give better results. In this paper, we studied what biased models can give us for real-time image denoising. We defined Active Random Field as the combination of a MRF/CRF with a fast and suboptimal inference algorithm and trained this combination using pairs of input and desired output as well as a benchmark error measure (loss function). This training approach does not need to evaluate the MRF normalization constant and can use a validation set to detect when the training is completed and whether overfitting occurs.

Applied to image denoising, experiments show that considerable gains in speed and accuracy are obtained when compared to the standard MRF formulation. Moreover, the obtained results are comparable in terms of accuracy with the state of the art while being faster.

A direct practical application of this method is denoising fluoroscopy (X-ray) sequences, where one could use pairs of low-dose (noisy) and high-dose (good quality) X-rays obtained from cadavers or phantoms to train a similar Active Random Field based real-time image denoising system.

This type of training can be used in other applications where fast MRF inference is desired on models with a large number of parameters, for example super-resolution [13], [43]. If the number of model parameters is small, the model might not be flexible enough to adapt to the fast inference algorithm.

REFERENCES

[1] Y. Altun, I. Tsochantaridis, and T. Hofmann. Hidden Markov Support Vector Machines. *International Conference in Machine Learning*, 20(1):3, 2003.

[2] A. Barbu. Learning Real-Time MRF Inference for Image Denoising. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[3] J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society B*, 48(3):259–302, 1986.

[4] PJ Bickel and KA Doksum. Mathematical Statistics: Basic Ideas and Selected Topics, vol. II, 2008.

[5] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(11):1222–1239, 2001.

[6] A. Buades, B. Coll, and J.M. Morel. A Non-Local Algorithm for Image Denoising. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 2, 2005.

[7] G. Casella and C.P. Robert. Rao-Blackwellisation of sampling schemes. *Biometrika*, 83(1):81–94, 1996.

[8] T.F. Cootes, G.J. Edwards, and C.J. Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681–685, 2001.

[9] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image Denoising by Sparse 3-D Transform-Domain Collaborative Filtering. *Image Processing, IEEE Transactions on*, 16(8):2080–2095, 2007.

[10] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Trans. Image Process*, 15(12):3736–3745, 2006.

[11] J.J. Faraway. *Linear Models with R*. Chapman & Hall/CRC, 2005.

[12] M. Fink and P. Perona. Mutual boosting for contextual inference. 2004.

[13] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning low-level vision. *Int. J. Comput. Vision*, 40:25–47, 2000.

[14] D.E. Goldberg et al. *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley Pub. Co Reading, Mass, 1989.

[15] G.E. Hinton. Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation*, 14(8):1771–1800, 2002.

[16] S. Kirkpatrick, CD Gelati Jr, and MP Vecchi. Optimization by Simulated Annealing. *Biology and Computation: A Physicist's Choice*, 1994.

[17] S. Kumar and M. Hebert. Discriminative random fields: a discriminative framework for contextual interaction in classification. *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1150–1157, 2003.

[18] J.D. Lafferty, A. McCallum, and F.C.N. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. *Proceedings of the Eighteenth International Conference on Machine Learning table of contents*, pages 282–289, 2001.

[19] Y. LeCun and F.J. Huang. Loss functions for discriminative training of energy-based models. *Proc. of the 10-thInternational Workshop on Artificial Intelligence and Statistics (AIStats 05)*, 3, 2005.

[20] Y. Li and D. P. Huttenlocher. Learning for optical flow using stochastic optimization.

[21] D.C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1):503–528, 1989.

[22] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms. *Proc. of ICCV01*, 2:416–425.

[23] T. Meltzer, C. Yanover, and Y. Weiss. Globally optimal solutions for energy minimization in stereo vision using reweighted belief propagation. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 428–435 Vol. 1, 2005.

[24] JA Nelder and R. Mead. A simplex method for function minimization. *The computer journal*, 7(4):308, 1965.

[25] F. Ning, D. Delhomme, Y. LeCun, F. Piano, L. Bottou, and P.E. Barbano. Toward automatic phenotyping of developing embryos from videos. *Image Processing, IEEE Transactions on*, 14:1360–1371, 2005.

[26] A. Pizurica and W. Philips. Estimating the Probability of the Presence of a Signal of Interest in Multiresolution Single-and Multiband Image Denoising. *Image Processing, IEEE Transactions on*, 15(3):654–665, 2006.

[27] J. Portilla, V. Strela, MJ Wainwright, and EP Simoncelli. Image denoising using scale mixtures of Gaussians in the wavelet domain. *Image Processing, IEEE Transactions on*, 12(11):1338–1351, 2003.

[28] RB Potts. Some generalized order-disorder transitions. *Proc. Camb. Phil. Soc*, 48:106–109, 1952.

[29] S. Roth and M.J. Black. Fields of Experts. *International Journal of Computer Vision*, 82(2):205–229, 2009.

[30] D. Scharstein and C. Pal. Learning conditional random fields for stereo. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–8, 2007.

[31] J. Sun, N.N. Zheng, and H.Y. Shum. Stereo matching using belief propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(7):787–800, 2003.

[32] MF Tappen and WT Freeman. Comparison of graph cuts with belief propagation for stereo, using identical MRF parameters. *ICCV*, pages 900–906, 2003.

[33] M.F. Tappen, Ce Liu, E.H. Adelson, and W.T. Freeman. Learning gaussian conditional random fields for low-level vision. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–8, 2007.

[34] M.F. Tappen and FL Orlando. Utilizing Variational Optimization to Learn Markov Random Fields. *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8, 2007.

[35] B. Taskar, S. Lacoste-Julien, and M. Jordan. Structured Prediction via the Extragradient Method. *Advances in Neural Information Processing Systems (NIPS)*, 18:1345, 2006.

[36] A. Torralba, K. P. Murphy, and W. T. Freeman. Contextual models for object detection using boosted random fields. *Advances in Neural Information Processing Systems (NIPS)*, 2004.

[37] Z. Tu. Auto-context and its application to high-level vision tasks. In *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008*, pages 1–8, 2008.

[38] P.D. Turney. Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. *Journal of Artificial Intelligence Research*, 2(1995):369–409, 1995.

[39] M. J. Wainwright. Estimating the "wrong" graphical model: Benefits in the computation-limited setting. *J. Mach. Learn. Res.*, 7:1829–1859, 2006.

[40] Z. Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[41] J. Weickert. A Review of Nonlinear Diffusion Filtering. *Proceedings of the First International Conference on Scale-Space Theory in Computer Vision*, pages 3–28, 1997.

[42] L. Xu, F. Hutter, H.H. Hoos, and K. Leyton-Brown. SATzilla: Portfolio-based Algorithm Selection for SAT. *Journal of Artificial Intelligence Research*, 32:565–606, 2008.

[43] J. Yang, J. Wright, T. Huang, and Y. Ma. Image super-resolution as sparse representation of raw image patches. In *CVPR*, 2008.

[44] J.S. Yedidia, W.T. Freeman, and Y. Weiss. Generalized belief propagation. *Advances in Neural Information Processing Systems*, 13:689–695, 2001.

[45] Y. Zheng, A. Barbu, B. Georgescu, M. Scheuering, and D. Comaniciu. Four-Chamber heart modeling and automatic segmentation for 3-D cardiac CT volumes using marginal space learning and steerable features. *Medical Imaging, IEEE Transactions on*, 27(11):1668–1681, 2008.

[46] S.K. Zhou and D. Comaniciu. Shape Regression Machine. *Proceedings of Information Processing in Medical Imaging*, pages 13–25, 2007.

[47] S.C. Zhu, Y. Wu, and D. Mumford. Filters, random fields and maximum entropy (FRAME): Towards a unified theory for texture modeling. *International Journal of Computer Vision*, 27(2):107–126, 1998.