Learning Real-Time MRF Inference for Image Denoising

Adrian Barbu[*]

Abstract

Many computer vision problems can be formulated in a Bayesian framework with Markov Random Field (MRF) or Conditional Random Field (CRF) priors. Usually, the model assumes that a full Maximum A Posteriori (MAP) estimation will be performed for inference, which can be really slow in practice. In this paper, we argue that through appropriate training, a MRF/CRF model can be trained to perform very well on a suboptimal inference algorithm. The model is trained together with a fast inference algorithm through an optimization of a loss function on a training set containing pairs of input images and desired outputs. A validation set can be used in this approach to estimate the generalization performance of the trained system. We apply the proposed method to an image denoising application, training a Fields of Experts MRF together with a 1-4 iteration gradient descent inference algorithm. Experimental validation on unseen data shows that the proposed training approach obtains an improved benchmark performance as well as a 1000-3000 times speedup compared to the Fields of Experts MRF trained with contrastive divergence. Using the new 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.*

1. Introduction

Many computer vision problems are approached by constructing models based on Markov Random Field (MRF) or Conditional Random Field (CRF) energy functions and inferring the solution through an optimization procedure such as gradient descent, Belief Propagation [27], Graph Cuts [3], Iterated Conditional Modes [2], etc. 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 [17], used in Stereo Matching as a prior term [3, 19, 20], is NP hard [3]. In such cases, polynomial algorithms are not expected to be found.

Second, it is hard to find energy functions that always have a global minimum exactly at the desired solution. Recent work [1, 13, 22] 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.

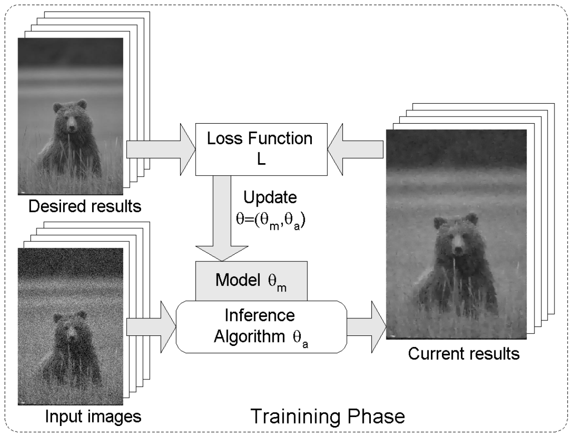

Figure 1. A MRF/CRF model trained together with a fast inference algorithm can obtain surprising speed and accuracy. Training is an optimization of a loss function and a training set of input and desired output images.

Wainwright [24] suggested that in computational limited settings, MAP estimation is not the best choice and a "wrong" model could be beneficial. In this paper, we propose a method to train such a model for solving MRF/CRF based problems where fast inference is desired. In the proposed approach, the MRF/CRF model is trained together with a fast and suboptimal inference algorithm to best collaborate in solving the given task. The MRF/CRF parameters are learned through an optimization procedure illustrated in Figure 1, using a supervised training set and a loss function to monitor progress towards the goal.

This idea is illustrated on an image denoising application, using the Fields of Experts [18] Markov Random Field (MRF) model and a simple gradient descent inference algorithm, previously used together for image denoising. The algorithm is restricted to be 1000-3000 times faster than usual and the best model-algorithm parameters are trained using a dataset of training pairs consisting of input images

[*]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.

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.

The goal of this paper is to demonstrate that training a MRF/CRF model together with a very fast inference algorithm could offer very good results in both speed and accuracy.

2. Markov Random Fields and Conditional Random Fields

Let $G = (V, E)$ be a graph, $\mathbf{x} = (x_v)_{v\in V}$ 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 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 often preferred, being capable of representing more complex relationships.

Quite recently, Conditional Random Fields (CRF) [12, 11] 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 the ϕ depending on $\mathbf{y}$.

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

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
- Inference is computationally demanding. For example, the exact inference even for one of the simplest pairwise MRF priors such as the Potts model [17] is NP hard [3]. Hence, approximate solutions are usually sought and obtained.
- Training the MRF requires the knowledge of the normalization constant Z for comparing different MRF models. The normalization constant Z usually has no closed form solution and is expensive to compute.

3. Energy Based Models and Loss Functions

Recent work [1, 13, 22] deals with the challenges related to the normalization constant by training the MRF parameters θ so that the maximum probability MRF points are as similar as possible to the corresponding desired outputs. The differences between the maximum probability MRF points $\mathbf{x}_i$ and the desired outputs $\mathbf{t}_i$ are measured using a loss function $L(\mathbf{x}_i, \mathbf{t}_i)$ and the training procedure 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 solves some of the difficulties related to the MRF training, eliminating 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) and a suboptimal point is usually obtained instead.

4. Proposed Method

Since most fast inference algorithms obtain a suboptimal solution anyway, we propose a different approach in which the model parameters are trained such that the inference algorithm output (and not the "ideal" most likely MRF point) is close to the desired output. This way, we introduce the inference algorithm in the learning loop. This combined approach can be written as:

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

where $\mathbf{x} = A(\mathbf{y}, \theta)$ is the result of applying the algorithm A based on the model and algorithm parameters $\theta = (\theta_m, \theta_a)$ to the input image $\mathbf{y}$. As in Section 3, the training data consists of pairs $(\mathbf{y}_i, \mathbf{t}_i)$ of input images $\mathbf{y}_i$ and the corresponding desired outputs $\mathbf{t}_i$. The loss function L is used to evaluate the performance of the model-algorithm combination on the training set.

The algorithm A is picked from a family $\mathcal{A}$ that can include any type of MRF/CRF inference algorithm: gradient descent, Belief Propagation [27], Graph Cuts [3], etc. However, in contrast to the standard MRF/CRF approaches, the algorithms in the family $\mathcal{A}$ should be very fast, by sacrificing accuracy. For example, the number of gradient descent iterations in our image denoising application is kept very small, on the order of 1 to 4, even though usually 3000-10000 iterations are used in applications. The deficit in accuracy of the algorithm will be compensated by training the model to give best results on this algorithm, resulting in a fast and incredibly accurate combination.

The trained MRF model and its inference algorithm can no longer be separated, because they cannot work well without each other. We call this model-algorithm combination an *Active Random Field* (ARF).

The loss function L is chosen to measure the progress toward solving the given vision task. For image denoising, we used the average PSNR (peak signal-to-noise ratio) over the set of images evaluated (training or testing) and we replaced the minimization in (5) with a maximization. Alternatively, we could have used the Mean Square Error as a loss function.

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

The proposed approach raises two concerns.

1. The main concern is overfitting, where an increased performance on the training set is reflected in a decreased performance on an unseen dataset. It 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 executing the algorithm on all the training examples for each optimization iteration. This is addressed using effective design strategies (e.g. memorization of partial results) and through efficient optimization algorithms such as conjugate gradient or genetic algorithms that can make good use of each function evaluation. Moreover the exponential growth in computational power of a standard PC makes the computation less of an issue.

4.1. Related Work

In the literature, a substantial amount of work combines models with algorithms in different ways. Active Appearance Models [5] are iterative algorithms driven by the data and a PCA-like model to find objects of interest in the image. The solution depends on the initialization inside the image, so they can only be 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 [29], in which an image based regression algorithm is trained to find a vector toward the object of interest from any random location inside the image. By using hundreds of random initialization and a verification step based on Adaboost, a fast and robust object detection system is obtained. 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.

Similar goals in obtaining good results with low computational expense are explored in cost-sensitive learning. In [23], a decision tree was trained to minimize a cost function with terms for accuracy and computational expense for each feature. Also related is [26], 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.

The proposed method bears similarity to the *energy based models* [13], 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 MAP is at the desired location on the training set, independent of the optimization (inference) algorithm that will be used to obtain that minimum. In contrast, the proposed training finds the model parameters that give best results on a training set using a preselected fast inference algorithm.

The same method from Eq. (5) is used in [21] for image denoising, but as an attempt to obtain a stronger MAP optimum than the gradient descent. For that, a more complex inference algorithm, based on variational optimization, is derived. In this paper, we do not aim at obtaining a MAP optimum and focus on preventing and avoiding overfitting instead. Even when using a very fast inference algorithm such as one iteration of gradient descent, through appropriate training, the model will adapt to the simple descent algorithm. Consequently, the image denoising results presented in this paper surpass any previous results based on MRF models.

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.

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

1. In our approach, the normalization constant Z is not important anymore, since different models are compared using the loss function L instead of the likelihood or posterior probability.
2. In a fully supervised way, the proposed approach uses training sets consisting of pairs of input images and desired results. This gives a better idea on when the training is completed or whether overfitting occurs.
3. A complex and well-trained model can complement some of the algorithm's weaknesses, obtaining a fast yet accurate system.

5. Application: Image Denoising

We apply the proposed method to image denoising, where given an image corrupted with noise, the goal is to obtain an image from which the noise was removed. Image Denoising can be viewed as a graph-based optimization

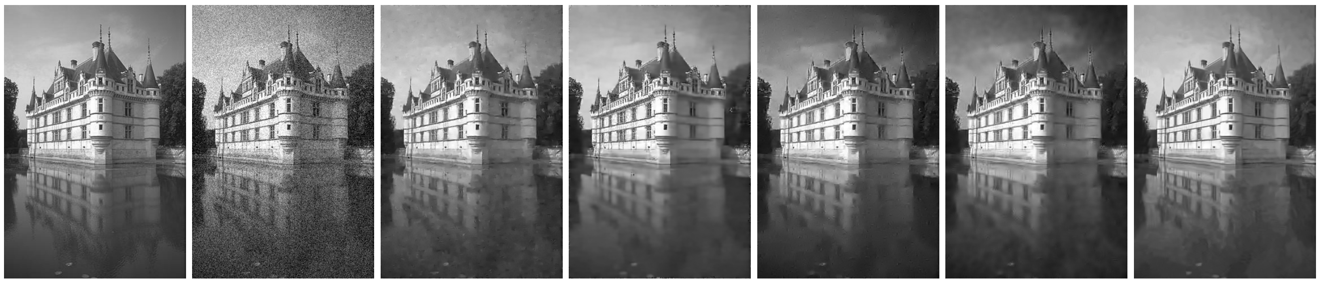
Figure 2. Image denoising example. From left to right: original image, image corrupted with additive Gaussian noise with $\sigma = 25$, PSNR=20.17; our result, PSNR=28.94, 0.6 seconds and Fields of Experts result [18], PSNR=28.67, 2280 seconds; wavelet denoising [16], PSNR=29.05, 16 seconds; overcomplete DCT, PSNR=28.81, 38 seconds, and BM3D [6], PSNR=29.60, 4.3 seconds.

problem, with the graph nodes being the pixels of the image. This problem has been addressed using wavelets in [16, 15] and by learning a MRF prior model named Fields of Experts on 5×5 pixel cliques in [18]. Non-local image denoising methods include [4] and especially 3D collaborative filtering (BM3D) [6], 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 approach proposed in this paper are shown in Figure 2, together with the CPU time required to obtain each result. Another approach [7] uses a sparse representation based on a learned dictionary of primitives, and it is more computationally involved.

We apply the approach proposed in this paper on the Fields of Experts MRF model and the gradient descent algorithm that were presented in [18] and will be briefly mentioned in the next section. The loss function used for training is the average PSNR (Peak Signal to Noise Ratio) over the training set.

5.1. Fields of Experts

The Fields of Experts [18] 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 $\alpha_f, f = 1, ..., N$

$$p_{FOE}(\mathbf{x}, \theta) = \frac{1}{Z(\theta)} \exp(-E_{FOE}(\mathbf{x}, \theta)),$$
$$E_{FOE}(\mathbf{x}, \theta) = \sum_k \sum_{f=1}^{N} \alpha_f \log(1 + \frac{1}{2}(J_f^T \mathbf{x}^{(k)})^2) \tag{6}$$

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 nonlinearity.

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

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

where $\mathbf{x}^j$ is the value of pixel j of image $\mathbf{x}$. (7)

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

$$\nabla_{\mathbf{x}} E_{FOE}(\mathbf{x}, \theta) = \sum_{f=1}^{N} \alpha_f J_f^- * \frac{J_f^T \mathbf{x}}{1 + \frac{1}{2}(J_f^T \mathbf{x})^2} \tag{8}$$

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 $E_{data}(\mathbf{x}) + E_{FOE}(\mathbf{x}, \theta)$. 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 [18].

5.2. Supervised Training of the FOE Model

We will use the FOE model presented above, parametrized by the number n of kernels, the convolution kernels $J_f, f = 1, ..., n$, their corresponding coefficients α_f. Together with the FOE model, we also use the family of algorithms $\mathcal{A}$ from above. By ignoring the normalization constant and the Gaussian likelihood, from the energy gradient equation (8) results the gradient descent inference algorithm:

$$\mathbf{x} \leftarrow \mathbf{x} + \delta \sum_{f=1}^{N} \alpha_f J_f^- * \frac{J_f^T \mathbf{x}}{1 + \frac{1}{2}(J_f^T \mathbf{x})^2} \tag{9}$$

These iterative algorithms will form the algorithm family $\mathcal{A}$, parametrized by the number n_{iter} of gradient update iterations (9), and the update parameter δ of equation (9). Therefore

$$\theta = (\theta_m, \theta_a) = (n, J_1, \alpha_1, ..., J_n, \alpha_n, n_{iter}, \delta). \tag{10}$$

The algorithm family $\mathcal{A}$ is restricted to 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 will observe that the denoising performance actually increases compared to FOE, for an appropriately trained system.

In [18], the Fields of Experts model is trained using Contrastive Divergence [9] 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, values obtained using MCMC sampling. The training procedure is computationally intensive and yields a generic prior model for natural images.

In [21], 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 (9).

In what follows, we will show that this is not necessary, since by appropriate training of 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.

Dataset. The same images as [18] are used for training, namely 40 natural images from the Berkeley dataset [14]. 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 about $150,000$ cliques(pixels), the training set contains about $6,000,000$ cliques.

Instead of using the original images as training examples, we experimented with smaller patches (e.g. of size 15×15 as in [18]) and observed on a validation set that overfitting occurs when the patches are smaller than 250×250 pixels. This could be due to the boundary effect, since smaller patches a higher percentage of boundary pixels.

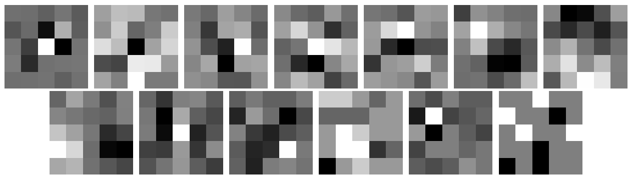
Figure 3. The thirteen 5×5 FOE filters trained using our approach for the level of noise $\sigma = 20$ and for a three iteration ($n_{iter} = 3$) steepest descent inference algorithm.

For testing, we use the same 68 natural images from the Berkeley dataset as [18] as well as some standard image denoising test images. These testing images were not used for training.

Loss Function. The learning is achieved 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 chosen as

$$L(\mathbf{x}, \mathbf{t}) = 20 \log_{10}(255/std(\mathbf{t} - \mathbf{x})) \tag{11}$$

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

Learning is an optimization on the parameters θ to maximize $M(\theta) = \frac{1}{n} \sum_{i=1}^{n} L(A(\mathbf{y}_i, \theta), \mathbf{t}_i)$, the average PSNR obtained after running the denoising algorithm $A(\mathbf{y}_i, \theta)$ with parameters θ on the 40 training examples $\mathbf{y}_i$.

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. We also experimented with gradient ascent, conjugate gradient and the simplex method. For this particular application, we observed that these other methods could not find such a strong optimum as the coordinate ascent. This could be because the path toward the optimum is very narrow and a fast algorithm could not follow it properly or because the gradient cannot be computed analytically and hence was approximated.

Other optimization methods such as genetic algorithms [8] or simulated annealing [10] could be more appropriate for avoiding local optima and are subject to further investigation.

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:

$$\sum_{i=1}^{40} ||\mathbf{t}_i - \mathbf{x}_i - \delta \sum_{f=1}^{N} \alpha_f (J_f^- * \frac{J_f^T \mathbf{x}_i}{1 + \frac{1}{2}(J_f^T \mathbf{x}_i)^2})||^2 \tag{12}$$

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*.

The one iteration filters for $\sigma = 25$ are trained using *Marginal Space Learning* [28]. Marginal Space Learning is an effective optimization procedure that finds the modes of a high-dimensional distribution by propagating a set of particles through a sequence of marginal probabilities in subspaces of increasing dimensions. This procedure was chosen to deal with the difficulty of finding strong optima in the large parameter space of the filters and their coefficients (325 parameters for 13 filters of size 5×5).

For training the one iteration filters, the subspaces parametrize models with different numbers of filters and filter sizes, as shown in Figure 4. Only one particle (the maximum PSNR configuration) is propagated between the subspaces. The first subspace parametrizes models with one filter of size 3×3. The particle is obtained by PSNR optimization with coordinate ascent, starting at the location where the filter parameters are all 0 except $F_1(1,1) = 0.1, F_1(1,2) = -0.1$. As shown in Figure 4, the second subspace parametrizes models with two-filters of size 3×3.

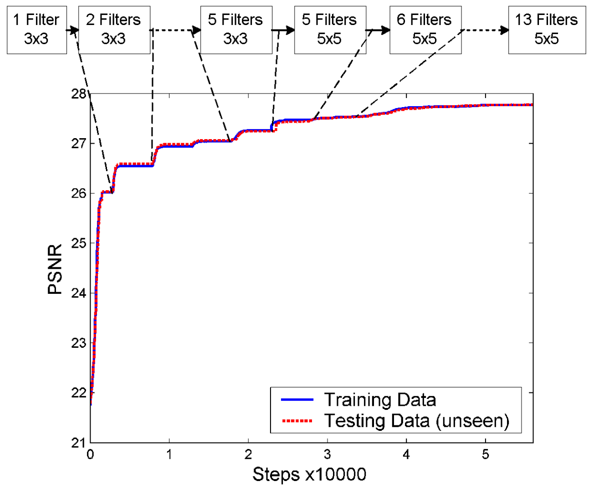

Figure 4. Diagram for training the MRF parameters for the level of noise $\sigma = 25$ and the one iteration ($n_{iter} = 1$) steepest descent inference algorithm. Also displayed is the PSNR evolution on the training and test set observed during the training.

The search for the max PSNR particle in the second subspace is initialized by setting the first filter from the first subspace particle and second filter with 0 and the PSNR optimization was run again. The process of increasing the marginal space (either by adding one more filter initialized with zeros or by increasing the filter sizes by padding zeros) and seeking the particle of maximum PSNR in that subspace was repeated until there were a total of $N = 13$ filters of size 5×5, as shown in Figure 4. This number was chosen by observing on the training set (or the validation set since the curves are almost identical) that no further improvement in PSNR was obtained. The evolution of the PSNR over all this training, starting with one 3×3 filter and ending with 13 filters of size 5×5 is plotted in Figure 4. As one could see from the plot, the training seems very robust to overfitting.

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 5. The 3-iteration filters also work very well for 4-iterations, so the 4-iteration filters were just copied from the 3-iteration filters.

Training each of the arrows in Figure 5 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 3.

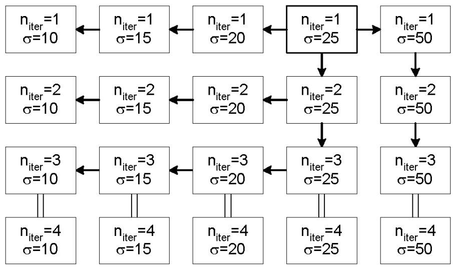

Figure 5. Diagram of the training of the model parameters for different levels of noise and numbers of iterations of the steepest descent inference algorithm.

5.3. Results

The performance of the proposed method is evaluated on the same datasets as [18]. First, results on some standard images - Lena, Barbara, Boats, House and Peppers - at the noise level $\sigma = 25$ are shown in Table 1. Note that these images were not used for training. Our results are obtained between 7 and 3000 times faster than the other methods.

Table 1. Performance evaluation and comparison of our ARF method (1-4 iterations) with other methods on some standard benchmark images, $\sigma = 25$. Our method is 7-3000 times faster.

Image	Lena	Barbara	Boats	House	Peppers	Average
FOE [18]	30.82	27.04	28.72	31.11	29.20	29.38
Ours, 1 iter	30.15	27.10	28.66	30.14	28.90	28.99
Ours, 2 iter	30.66	27.49	28.99	30.80	29.31	29.45
Ours, 3 iter	30.76	27.57	29.08	31.04	29.45	29.58
Ours, 4 iter	30.86	27.59	29.14	31.18	29.51	29.66
Wavelet [16]	31.69	29.13	29.37	31.40	29.21	30.16
DCT [7]	30.89	28.65	28.78	31.03	29.01	29.67
Dictionary[7]	31.20	27.57	29.17	31.82	29.84	29.92
KSVD [7]	31.32	29.60	29.28	32.15	29.73	30.42
BM3D [6]	32.08	30.72	29.91	32.86	30.16	31.15

For further comparison, Table 2 and Figure 6 present results on the same 68 test images from the Berkeley dataset as [18]. Note that these images were also not used for training. We present results for 1: Wiener filter, 2: nonlinear diffusion [25], 3: Non-local means [4], 4: Fields of Experts (FOE) [18] with 3000 iterations, 5,6,7,8: our algorithm with 1,2,3,4 iterations, 9: wavelet based denoising [16], 10: Overcomplete DCT [7], 11: KSVD [7] and 12: BM3D [6]. Since this evaluation is on 68 images, it should be regarded as a more thorough evaluation than the results on 5 specific images.

From the evaluation, it is clear that the one iteration ARF trained as proposed is on par with the FOE, while being 3000 times faster. Therefore, training the MRF model together with the inference algorithm offers significant advan-

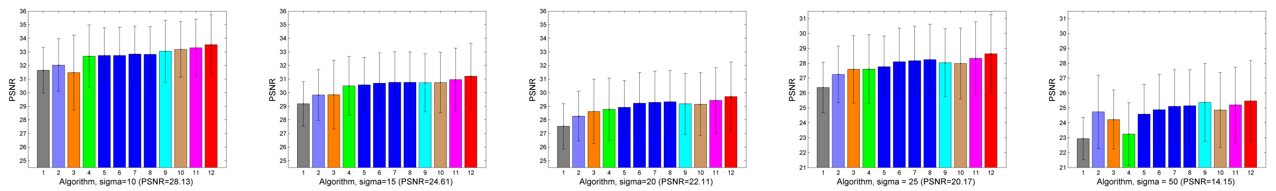

Figure 6. Average PSNR in dB for different image denoising algorithms at different noise levels on 68 images from the Berkeley dataset. 1: Wiener Filter, 2: nonlinear diffusion, 3: Non-local means [4] 4: Fields of Experts [18], 5,6,7,8: Our ARF algorithm with 1,2,3 and 4 iterations, 9: wavelet based denoising [16], 10: Overcomplete DCT [7], 11: KSVD [7] and 12: BM3D [6]. The results are also shown in Table 2.

tages in speed and accuracy. One could also observe that the ARF trained using our method is within 0.5dB from the best method, and it is outperformed by two methods: KSVD [7], BM3D [6] and for some noise levels by wavelet denoising [16] and overcomplete DCT [7].

We also evaluated the FOE model with one iteration on the same 68 images from the Berkeley dataset, choosing the step size δ to maximize the PSNR. For the level of noise $\sigma = 15$, the obtained average PSNR was 25.44, which is inferior to all the approaches presented in Table 2.

For different applications, different trade-offs between speed and accuracy might be important. Figure 7 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 proposed approach is very competitive when high processing speeds are required, such as in real-time medical applications.

Table 2. 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.NL Diffusion[25]	32.03	29.83	28.28	27.25	24.73
3.Non-local [4]	31.48	29.86	28.62	27.59	24.22
4.FOE [18]	32.68	30.50	28.78	27.60	23.25
5.Ours, 1 iter	32.74	30.57	28.92	27.77	24.58
6.Ours, 2 iter	32.74	30.70	29.23	28.10	24.88
7.Ours, 3 iter	32.84	30.76	29.29	28.17	25.11
8.Ours, 4 iter	32.82	30.76	29.33	28.24	25.14
9.Wavelet [16]	33.05	30.73	29.18	28.03	25.37
10.DCT [7]	33.19	30.75	29.15	27.98	24.86
11.KSVD [7]	33.30	30.96	29.43	28.33	25.20
12.BM3D [6]	33.53	31.21	29.71	28.63	25.47

The computation complexity of the FOE image denoising algorithm trained with the proposed approach 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) brings the computation time down to 0.12s per iteration for the same image size. In addition, a parallel implementation on multiple CPUs or a GPU implementation could further bring this algorithm to real-time performance.

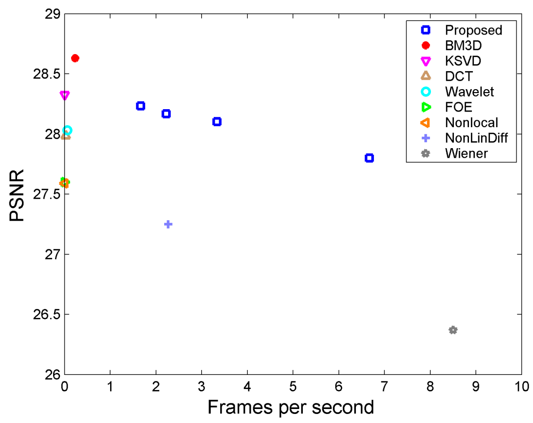

Figure 7. PSNR (dB) vs speed (fps) of different denoising algorithms on the 68 Berkeley test images. The colors are the same as in Figure 6.

6. Conclusions

In this paper we observed surprising advantages of a supervised training approach for a MRF/CRF using a fast and suboptimal inference algorithm. The proposed approach is an optimization of a loss function on a training set consisting of pairs of input and desired outputs. This method does not need the MRF normalization constant Z and can use a validation set to detect when the training is completed and whether overfitting occurs.

Applied to image denoising with a Fields of Experts MRF and a 1-4 iteration steepest descent, experiments showed that the supervised training obtains a model that is more accurate and thousand of times faster than its counterpart trained in an unsupervised way. Moreover, the obtained results are competitive in terms of accuracy with the state of the art while being much faster.

A direct practical application of this method is the denoising of 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 FOE based real-time image denoising system.

This type of supervised training can be applied to other application where fast MRF inference is desired on models with a large number of parameters.

References

[1] Y. Altun, I. Tsochantaridis, and T. Hofmann. Hidden Markov SVM. *ML Workshop*, 20(1):3, 2003.

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

[3] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE TPAMI*, 23(11):1222–1239, 2001.

[4] A. Buades, B. Coll, and J.M. Morel. A Non-Local Algorithm for Image Denoising. *CVPR*, 2005.

[5] T.F. Cootes, G.J. Edwards, and C.J. Taylor. Active appearance models. *TPAMI*, 23(6):681–685, 2001.

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

[7] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE T Img. Proc.*, 15(12), 2006.

[8] D.E. Goldberg et al. *Genetic algorithms in search, optimization, and machine learning*. 1989.

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

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

[11] S. Kumar and M. Hebert. Discriminative random fields: a discriminative framework for contextual interaction in classification. *ICCV*, 2003.

[12] J.D. Lafferty, A. McCallum, and F.C.N. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. *ICML 2001*.

[13] Y. LeCun and F.J. Huang. Loss functions for discriminative training of energy-based models. *AIStats*, 3, 2005.

[14] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms. *ICCV 2001*, 2:416–425.

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

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

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

[18] S. Roth and MJ Black. Fields of Experts: a framework for learning image priors. *CVPR 2005*.

[19] J. Sun, N.N. Zheng, and H.Y. Shum. Stereo matching using belief propagation. *TPAMI*, 25:787–800, 2003.

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

[21] M.F. Tappen and FL Orlando. Utilizing Variational Optimization to Learn Markov Random Fields. *CVPR*, pages 1–8, 2007.

[22] B. Taskar, S. Lacoste-Julien, and M. Jordan. Structured Prediction via the Extragradient Method. *NIPS*, 18:1345, 2006.

[23] P.D. Turney. Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. *J. of AI Research*, 2:369–409, 1995.

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

[25] J. Weickert. A Review of Nonlinear Diffusion Filtering. *Int Conf Scale-Space Th. in Comp. Vis.*, 1997.

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

[27] J.S. Yedidia, W.T. Freeman, and Y. Weiss. Generalized belief propagation. *NIPS*, 13, 2001.

[28] 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.

[29] S.K. Zhou and D. Comaniciu. Shape Regression Machine. *IPMI*, pages 13–25, 2007.