Learning Optimized MAP Estimates in Continuously-Valued MRF Models

Kegan G. G. Samuel, Marshall F. Tappen
University of Central Florida
School of Electrical Engineering and Computer Science, Orlando, FL
{kegan,mtappen}@eecs.ucf.edu

Abstract

We present a new approach for the discriminative training of continuous-valued Markov Random Field (MRF) model parameters. In our approach we train the MRF model by optimizing the parameters so that the minimum energy solution of the model is as similar as possible to the ground-truth. This leads to parameters which are directly optimized to increase the quality of the MAP estimates during inference. Our proposed technique allows us to develop a framework that is flexible and intuitively easy to understand and implement, which makes it an attractive alternative to learn the parameters of a continuous-valued MRF model. We demonstrate the effectiveness of our technique by applying it to the problems of image denoising and inpainting using the Field of Experts model. In our experiments, the performance of our system compares favourably to the Field of Experts model trained using contrastive divergence when applied to the denoising and inpainting tasks.

1. Introduction

Recent years have seen the introduction of several new approaches for learning the parameters of continuous-valued Markov Random Field models [4, 11, 15, 20, 9, 12]. Models based on these methods have proven to be particularly useful for expresssing image priors in low-level vision systems, such as denoising, and have led to state-of-the-art results for MRF-based systems.

In this paper, we introduce a new approach for discriminatively training MRF parameters. As will be explained in Section 2, we train the MRF model by optimizing its parameters so that the minimum energy solution of the model is as similar as possible to the ground-truth. While previous work has relied on time-consuming iterative approximations [14] or stochastic approximations [9], we show how implicit differentiation, can be used to analytically differentiate the overall training loss with respect to the MRF parameters. This leads to an efficient, flexible learning algorithm that can be applied to a number of different models. Using Roth and Black's Field of Experts model, Section 4 shows how this new learning algorithm leads to improved results over FoE models trained using other methods.

Our approach also has unique benefits for systems based on MAP-inference. While Section 2.1 explores this issue in more detail, our high-level argument is that if one's goal is to use MAP estimates and evaluate the results using some criterion then optimizing MRF parameters using a probabilistic criterion, like maximum-likelihood parameter estimation, will not necessarily lead to the optimal system. Instead, the system should be directly optimized so that the MAP solution scores as well as possible.

2. The Basic Learning Model

Similar to the discriminative, max-margin framework proposed by Taskar et al. [16] and the Energy-Based Models [7], we learn the MRF parameters by defining a loss function that measures the similarity between the ground truth $\mathbf{t}$ and the minimum energy configuration of the MRF model. Throughout this paper, we will denote this minimum energy configuration as $\mathbf{x}^*$.

This discriminative training criterion can be expressed formally as the following minimization:

$$\begin{aligned} \min_{\theta} \quad & L(\mathbf{x}^*(\theta), \mathbf{t}) \\ & \text{where } \mathbf{x}^*(\theta) = \arg\min_{\mathbf{x}} E(\mathbf{x}, \mathbf{y}; w(\theta)) \end{aligned} \tag{1}$$

The MRF model is defined by the energy function $E(\mathbf{x}, \mathbf{y}; w(\theta))$, where $\mathbf{x}$ denotes the current state of the MRF, the observations are denoted $\mathbf{y}$, and parameters θ. The function $w(\cdot)$ serves as a function that transforms θ before it is actually used in the energy function. It could be as simple as $w(\theta) = \theta$ or could aid in enforcing certain conditions. For instance, it is common to use an exponential function to ensure positive weighting coefficients, such as $w(\theta) = e^{\theta}$ for a scalar θ, as employed in work like [11]. Our primary motivation for introducing $w(\cdot)$ here is to aid in the derivation below.

Note that if $E(\mathbf{x}, \mathbf{y}; w(\theta))$ is non-convex, we can only expect $\mathbf{x}^*$ to be a local minimum. Practically, we have found that our method is still able to perform well when learning parameters for non-convex energy functions, as we will show in Section 4.

978-1-4244-3991-1/09/$25.00 ©2009 IEEE

2.1. Advantages of Discriminative Learning

Typically, MRF models have been posed in a probabilistic framework. Likewise, parameter learning has been similarly posed probabilistically – often in a maximum likelihood framework. In many models, computing the partition function and performancing inference are intractable, so approximate methods, such as the tree-based bounds of Wainwright et al [18] or Hinton's contrastive divergence method [5], must be used.

A key advantage of the discriminative training criterion is that it is not necessary to compute the partition function or expectations over various quantities. Instead, it is only necessary to minimize the energy function. Informally, this can be seen as reducing the most difficult step in training from integrating over the space of all possible images, which would be required to compute the partition function or expectations, to only having to minimize an energy function.

A second advantage of this discriminative training criterion is that it better matches the MAP inference procedure typically used in large, non-convex MRF models. MAP estimates are popular when inference, exact or approximate, is difficult. Unfortunately, parameters that maximize the likelihood may not lead to the highest-quality MAP solution.

The Field of Experts model is a good example of this issue. The parameters in this model were trained in a maximum-likelihood fashion, using the Contrastive Divergence method to compute approximate gradients. Using this model, the best results, with the highest PSNR, are found by terminating the minimization of the negative log-likelihood of the model before reaching a local minimum. As can be seen from Figure 1, if the FoE model trained using contrastive divergence is allowed to reach a local minimum then the performance decreases significantly. This happens because maximum-likelihood criterion that was used to train the system is related, but not directly tied to the PSNR of the MAP estimate.

Our argument is that if the intent is to use MAP estimates and evaluate the estimates using some criterion, like PSNR, a better strategy is to find the parameters such that the quality of MAP estimates are directly optimized. Returning to the example in the previous paragraph, if the goal is to maximize PSNR, maximizing a likelihood and then hoping that it leads to MAP estimates with good PSNR values is not the best strategy. We argue that one should instead choose the parameters such that the MAP estimates have the highest PSNR possible[1]. As Figure 1 shows, when the FoE model is trained using our proposed approach, the PSNR achieved when the minimization terminates is very close to the maximum PSNR achieved over the course of the minimization. This is important because the quality of the model is not tied to a particular minimization scheme. Using the model trained with our method, different types of optimization method could be used, while maintaining high-quality results.

[1]We wish to reiterate that our method is not tied to the PSNR image quality metric – any differentiable image loss function can be used.

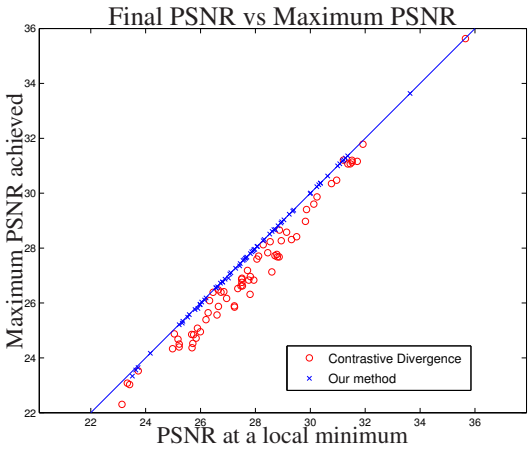

Figure 1. This figure shows the difference between training the FoE model using contrastive divergence and our proposed method. At termination, that is at a local minima, our training method produces results that are very close to the maximum PSNR achieved over the course of the minimization on images with additive Gaussian noise, $\sigma = 15$.

2.2. Related Work

Our approach is most closely related to the Variational Mode Learning approach proposed in [14]. This method works by treating a series of variational optimization steps as a continuous function and differentiating the result with respect to the model parameters. The key advantage of our approach over Variational Mode Learning, is that the result of the variational optimization must be recomputed every time the gradient of the loss function is recomputed. In [14], the variational optimization often required 20-30 steps. This translates into 20 to 30 calls to the matrix solver each time a gradient must be recomputed. On the other hand, our method only requires one call to the matrix solver to compute a gradient.

The approach proposed by Roth and Black [11] to learn the parameters of the Field of Experts model uses a sampling strategy and the idea of contrastive divergence [5] to estimate the expectation over the model distribution. Using this estimate they perform gradient ascent on the log-likelihood to update the parameters. This method, however, only computes approximate gradients and there is no guarantee that the contrastive divergence method converges.

In [12], Scharstein and Pal propose another method which uses MAP estimates to compute approximate expectation gradients to learn parameters of a CRF model. However, this approach produces stability issues with models with a large number of parameters. Our method is able to

effectively train a model with a large number of parameters.

Another recently proposed approach to learn the parameters of a continuous-state MRF model is using simultaneous perturbation stochastic approximation (SPSA) to minimize the training loss across a set of ground truth images [9]. When using SPSA it is difficult to determine exact convergence and multiple runs are usually required to reduce the variance of the learnt parameters. Also, SPSA requires various coefficients which have to be 'tweaked' to get optimal performance. While there are guidelines on how those coefficients can be chosen [13], it is still a matter of trial and error in determining the right values to achieve the best performance.

It should be noted that in [3], Do et al. were able to learn hyper-parameters, rather than parameters, of CRF model by using a chain-structured model. Using a chain-structured model makes it possible to compute the Hessian of the CRF's density function, thus enabling the method to learn hyper-parameters. Here, we consider problems where the density makes it impossible to compute this Hessian, as is the case in non-Gaussian models with loops. Because we cannot compute the Hessian, we cannot learn hyper-parameters.

2.3. Using Gradient Minimization

Taskar et al. have shown that for certain types of energy and loss functions, the learning task in Equation 1 can be accomplished with convex optimization algorithms [16, 17]. However, the similarity of Equation 1 to solutions for learning hyper-parameters, such as [3, 1, 6, 2], suggests that it may be possible to optimize the MRF parameters θ using simpler gradient-based algorithms. In the hyper-parameter learning work, the authors were able to use implicit differentiation to compute the gradient of a loss function with respect to hyper-parameters. In this section, we show how the same implicit differentiation technique can be applied to calculate the gradient of $L(\mathbf{x}^*(\theta), \mathbf{t})$ with respect to the parameters θ. This will enable us to use basic steepest-descent techniques to learn the parameters θ.

In the following section, we will begin with a general formulation, then, in later sections, show the derivations for a specific, filter-based MRF image prior.

2.4. Calculating the Gradient with Implicit Differentiation

In this section, we will show how the gradient vector of the loss can be computed with respect to some parameter θ_i using the implicit differentation method used in hyper-parameter learning. We will begin by calculating the derivative vector of $\mathbf{x}^*(\theta)$ with respect to θ_i. Once we can differentiate $\mathbf{x}^*(\theta)$ with respect to θ, a basic application of the chain rule will enable us to compute $\frac{\partial L}{\partial \theta_i}$.

Because $\mathbf{x}^*(\theta) = \arg\min_{\mathbf{x}} E(\mathbf{x}, \mathbf{y}; w(\theta))$, we can express the following condition on $\mathbf{x}^*$:

$$\left.\frac{\partial E(\mathbf{x}; \mathbf{y}, w(\theta))}{\partial \mathbf{x}}\right|_{\mathbf{x}^*(\theta)} = \mathbf{0} \tag{2}$$

This simply states that the gradient at a minimum must be equal to the zero vector. For clarity in the remaining discussion, will replace $\frac{\partial E(\mathbf{x},\mathbf{y};w(\theta))}{\partial \mathbf{x}}$,with the function $g(\mathbf{x}, w(\theta))$, such that

$$g(\mathbf{x}, w(\theta)) \triangleq \frac{\partial E(\mathbf{x}, \mathbf{y}; w(\theta))}{\partial \mathbf{x}}$$

Notice that we have retained θ as a parameter of $g(\cdot)$, though it first passes through the function $w(\cdot)$. We retain θ because it will be eventually be treated as a parameter that can be varied, though in Equation 2 it is treated as a constant.

Using this notation, we can restate Equation 2 as

$$g(\mathbf{x}^*(\theta), w(\theta)) = \mathbf{0} \tag{3}$$

Note that $g(\cdot)$ is a vector function of vector inputs.

We can now differentiate both sides with respect to the parameter θ_i. After applying the chain rule, the derivative of the left side of Equation 3 is

$$\frac{\partial g}{\partial \theta_i} = \frac{\partial g}{\partial \mathbf{x}^*}\frac{\partial \mathbf{x}^*}{\partial \theta_i} + \frac{\partial g}{\partial w}\frac{\partial w}{\partial \theta_i} \tag{4}$$

Note that if $\mathbf{x}$ and $\mathbf{x}^*$ are $N \times 1$ vectors, then $\frac{\partial g}{\partial \mathbf{x}^*}$ is an $N \times N$ matrix. Using Equation 3, we can now solve for $\frac{\partial \mathbf{x}^*}{\partial \theta_i}$:

$$\begin{aligned} \mathbf{0} &= \frac{\partial g}{\partial \mathbf{x}^*}\frac{\partial \mathbf{x}^*}{\partial \theta_i} + \frac{\partial g}{\partial w}\frac{\partial w}{\partial \theta_i} \\ \frac{\partial \mathbf{x}^*}{\partial \theta_i} &= -\left(\frac{\partial g}{\partial \mathbf{x}^*}\right)^{-1}\frac{\partial g}{\partial w}\frac{\partial w}{\partial \theta_i} \end{aligned} \tag{5}$$

Note that because θ_i is a scalar, $\frac{\partial g}{\partial w}\frac{\partial w}{\partial \theta_i}$ is an $N \times 1$ vector.

The matrix $\frac{\partial g}{\partial \mathbf{x}^*}$ is easily computed by noticing that $g(\mathbf{x}^*, w(\theta))$ is the gradient of $E(\cdot)$, with each term from $\mathbf{x}$ replaced with a term from $\mathbf{x}^*$. This makes the $\frac{\partial g}{\partial \mathbf{x}^*}$ term in the above equations just the Hessian matrix of $E(\mathbf{x})$ evaluated at $\mathbf{x}^*$.

Denoting the Hessian of $E(\mathbf{x})$ evaluated at $\mathbf{x}^*$ as $H_E(\mathbf{x}^*)$ and applying the chain rule leads to the derivative of the overall loss function with respect to θ_i:

$$\frac{\partial L(\mathbf{x}^*(\theta), \mathbf{t})}{\partial \theta_i} = -\frac{\partial L(\mathbf{x}^*(\theta), \mathbf{t})}{\partial \mathbf{x}^*}^T H_E(\mathbf{x}^*)^{-1}\frac{\partial g}{\partial w}\frac{\partial w}{\partial \theta_i} \tag{6}$$

Previous authors have pointed out two important points regarding Equation 6 [3, 15]. First, the Hessian does not need to be inverted. Instead, only the value $\frac{\partial L(\mathbf{x}^*(\theta), \mathbf{t})}{\partial \theta_i}^T H_E(\mathbf{x}^*)^{-1}$ needs to be computed. This can

be accomplished efficiently in a number of ways, including iterative methods like conjugate-gradients. Second, by computing $\frac{\partial L(\mathbf{x}^*(\theta),\mathbf{t})}{\partial \theta_i}^T H_E(\mathbf{x}^*)^{-1}$ rather than $H_E(\mathbf{x}^*)^{-1}\frac{\partial g}{\partial w}\frac{\partial w}{\partial \theta_i}$, only one call to the solver is necessary to compute gradient for all parameters in the vector θ.

2.5. Overall Steps for Computing Gradient

This formulation provides us with a convenient framework to learn the parameter θ using basic optimization routines. The steps are:

1. Compute $\mathbf{x}^*(\theta)$ for the current parameter vector θ. In our work, this is accomplished using non-linear conjugate gradient optimization.

2. Compute the Hessian matrix at $\mathbf{x}^*(\theta)$, $H_E(\mathbf{x}^*)$. Also compute the training loss, $L(\mathbf{x}^*(\theta), \mathbf{t})$, and its gradient with respect to $\mathbf{x}^*$.

3. Compute the gradient of the $L(\cdot)$ using Equation 6. As described above, performing the computations in the correct order can lead to significant gains in computational efficiency.

3. Application in Denoising Images

Having outlined our general approach, we now apply this approach to learning image priors. In this section, we will describe how this approach can be used to train a model similar to the Field of Experts model. As we will show in Section 4, training using our method leads to a denoising model that performs quite well.

3.1. Background: Field of Experts Model

Recent work has shown that image priors created from the combination of image filters and robust penalty functions are very effective for low-level vision tasks like denoising, in-painting, and de-blurring [8, 11, 14]. The Field of Experts model is defined by a set of linear filters, $f_1, ...f_{N_f}$ and their associated weights $\alpha_1, ..., \alpha_{N_f}$. The Lorentzian penalty function, $\rho(x) = \log(1 + x^2)$ is used to define the clique potentials. This leads to a probability density function over an image, $\mathbf{x}$, to be defined as:

$$p(\mathbf{x}) = \frac{1}{Z}\exp\left(-\sum_{i=1}^{N_f}\alpha_i\sum_{p=1}^{N_p}\rho(\mathbf{x}_p * f_i)\right) \tag{7}$$

where N_p is the number of pixels, N_f is the number of filters and $(\mathbf{x}_p * f_i)$ denotes the result of convolving the patch at pixel p in image $\mathbf{x}$ with the filter f_i.

When applied to denoising images, the probability density in Eq. 7 is used as a prior and combined with a Gaussian likelihood function to form a posterior probability distribution of an image given by:

$$p(\mathbf{x}|\mathbf{y}) = \frac{1}{Z}\exp\left(-\sum_{i=1}^{N_f}\alpha_i\sum_{p=1}^{N_p}\rho(\mathbf{x}_p * f_i) - \frac{1}{2\sigma^2}\sum_{p=1}^{N_p}(\mathbf{x}_p - \mathbf{y}_p)^2\right) \tag{8}$$

where σ is the standard deviation of the Gaussian noise and $\mathbf{y}$ is a noisy observation of $\mathbf{x}$.

3.2. Energy and Loss Function Formulation

As stated in Section 2 we learn the MRF parameters by defining a loss function that measures the similarity between the ground truth $\mathbf{t}$ and the minimum energy configuration of the MRF model. We use the negative log-posterior given in Equation 8 to form our energy function as:

$$E(\mathbf{x}, \mathbf{y}; \alpha, \beta) = \sum_{i=1}^{N_f} e^{\alpha_i}\sum_{p=1}^{N_p}\log(1+(F_i\mathbf{x}_p)^2)+\sum_{p=1}^{N_p}(\mathbf{x}_p-\mathbf{y}_p)^2 \tag{9}$$

Here we have used multiplication with a doubly-Toeplitz matrix F_i to denote convolution with a filter f_i. Each filter, F_i, is formed from a linear combination of a set of basis filters $B_1, , .., B_{N_B}$. The parameters β determine the coefficients of the basis filters, that is, F_i is defined as

$$F_i = \sum_{j=1}^{N_B}\beta_{ij}B_j \tag{10}$$

This formulation allows us to learn the filters, $F_i, ..F_{N_f}$, via the parameters β as well as their respective weights via α.

In the following section we assume a loss function, $L(\mathbf{x}^*(\theta), \mathbf{t})$ to be the pixelwise-squared error between the $\mathbf{x}$ and $\mathbf{t}$. That is,

$$L(\mathbf{x}^*(\theta), \mathbf{t}) = \sum_{p=1}^{N_p}(\mathbf{x}_p - \mathbf{t}_p)^2 \tag{11}$$

where we have grouped the parameters α and β into a single vector θ. However, our proposed formulation is flexible enough to allow the user to choose a loss function that matches their notion of image quality.

3.3. Calculating Gradients in the FoE Model

For clarity, we will describe how to compute the required derivatives in the denoising formulation by considering a model with one filter. In this case the gradient of the energy function $E(\mathbf{x}, \mathbf{y}; \alpha, \beta)$ can then be written as

$$\frac{\partial E(\mathbf{x}, \mathbf{y}; \alpha, \beta)}{\partial \mathbf{x}} = F^T\rho'(\mathbf{u}) + 2(\mathbf{x} - \mathbf{y}) \tag{12}$$

where $\rho'(\mathbf{u})$ is a function that is applied elementwise to the vector $\mathbf{u} = F\mathbf{x}$ and defined as

$$\rho'(z) = \exp(\alpha)\frac{2z}{1 + z^2} \tag{13}$$

Following the steps in Section 2.4 we can write the derivative of **x*** w.r.t. the parameter β as the vector given by:

$$\frac{\partial \mathbf{x}^*}{\partial \beta_j} = -(F^T W F + I)^{-1}(B_j^T \rho'(\mathbf{u}^*) + F^T \rho''(\mathbf{u}^*) B_j \mathbf{x}) \tag{14}$$

where W is a N_pxN_p diagonal matrix and a $[W]_{i,i}$ entry is determined by applying the function

$$\rho''(z) = \exp(\alpha)\frac{2 - 2z^2}{(1 + z^2)^2} \tag{15}$$

elementwise to the vector $\mathbf{u}^* = F\mathbf{x}^*$.

This leads to the overall expression for computing the derivative of the loss function with respect to β to be defined as

$$\frac{L(\mathbf{x}^*(\theta), \mathbf{t})}{\partial \beta} = -(\mathbf{x}^* - \mathbf{t})^T (F^T W F + I)^{-1} C_\beta \tag{16}$$

where $C_\beta = (B_j^T \rho'(\mathbf{u}^*) + F^T \rho''(\mathbf{u}^*) B_j \mathbf{x})$

In a similar fashion, we get the expression for computing the derivative of the loss function with respect to α to be defined as

$$\frac{L(\mathbf{x}^*(\theta), \mathbf{t})}{\partial \alpha} = -(\mathbf{x}^* - \mathbf{t})^T (F^T W F + I)^{-1} C_\alpha \tag{17}$$

where $C_\alpha = F^T \rho'(\mathbf{u}^*)$.

The above equations can be easily extended for multiple filters $f_1, ...f_{N_f}$ and their corresponding doubly-Toeplitz matrices $F_1, ...F_{N_f}$.

Now that we have the necessary information to compute the required gradients (Equations 16 and 17) we can follow the steps outlined in Section 2.5 to learn the parameters of the FoE model.

3.4. Computational Issues

An important practical difference between this approach for learning MRF parameters and previous approaches for learning hyper-parameters is that the matrices $F_1 \dots F_{N_f}$ are sparse. We found that we were able to train on relatively large 51×51 image patches without having to use iterative solvers.

Another feature we can exploit is the fact that Equations 17 and 16 are identical except for the last C_θ term. This means that we only have to compute the Hessian once per gradient calculation.

3.5. Potential Problems

Because the energy function has multiple local minima, it is likely that re-optimizing $E(\cdot)$ from an arbitrary initial condition will find a different minimum than the values of **x*** used during training. Despite this, the model trained with this method still performs very well, as we will show in the following section. Training on many images prevents the system from trying to exploit specific minima that may be unique to one image. Instead, the system is attempting to find parameters that leads to a good solution across the training set. Utilizing a large enough training set will prevent the system from overfitting to accommodate a certain minimum energy configuration for a particular image.

4. Experiments and Results

We conducted our experiments using the images from the Berkeley segmentation database[10] identified by Roth and Black in their experiments. We used the same 40 images for training and 68 images for testing. Since our approach allowed us to train on larger patches than those used by Roth and Black, the results reported in this paper are all from training using four randomly selected 51 x 51 patches from each training image, giving us a total of 160 training patches. We learnt 24 filters with dimension 5 x 5 pixels using the same basis as Roth and Black. Training was also done using 2000 randomly selected 15x15 patches with slightly lower results.

In order to compare our results, we denoised the test images using the Field of Experts(FoE) implementation provided by Roth and Black. The parameters in this implementation were learnt using the contrastive divergence method mentioned in Section 2.2 and uses gradient ascent to denoise images. For convenience, we will refer to this system as FoE-GA (for gradient ascent). We ran those experiments for 2500 iterations per image as suggested by Roth and Black using a step size $\eta = 0.6$. We also implemented a denoising system using the same parameters from the FoE-GA system but using conjugate gradient descent instead of gradient ascent. This system we will refer to as FoE-CG (for conjugate gradient) and results are reported when the system converges at a local minimum.

As shown in Figure 2, we achieve similar and at times better performance at convergence when compared to FoE-GA across the test images when our MAP inference is allowed to terminate at a local minimum. We wish to reiterate that FoE-GA terminates after a fixed number of iterations which is chosen to maximize performance and not when the system reaches a local minimum. When compared to FoE-CG our performance is noticeably much better. This shows a significant and important difference using our proposed training method: if the minimization is allowed to terminate at a local minimum then our performance is much better. Convergence is also achieved much faster using our training method as shown in Table 1. We used the same stopping criteria for our testing results and FoE-CG.

We also computed the perceptually based SSIM index[19] to measure the denoising results. Table 2 gives the average PSNR and SSIM index computed from the denoised images. Figures 3 and 4 show examples in which our results are better in terms of PSNR and SSIM. Visually, the texture is preserved better in our denoised images. How-

	FoE-CG	Ours
$\sigma = 25$	1874	227
$\sigma = 15$	1009	94

Table 1. This table shows the average number of conjugate gradient iterations taken before convergence when performing denoising on the 68 test images. All iterations have the same complexity; so in this case fewer iterations means faster performance.

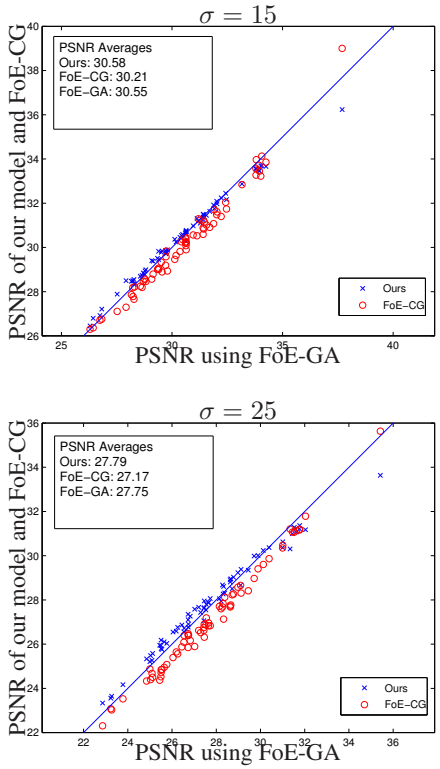

Figure 2. Scatterplots comparing the PSNR between our results and Field of Experts' using contrastive divergence on the test images. A point above the line means performance is better than FoE-GA. As can be seen, our performance is generally better than FoE-GA. On the other hand,FoE-CG does worse than FoE-GA

System	Average PSNR	Average SSIM
Ours	27.86	0.7760
FoE-GA	27.75	0.7632
FoE-CG	27.17	0.7266

Table 2. This table compares the average PSNR and SSIM index on the test images between our results and FoE-GA/CG for images with Gaussian noise of standard deviation, $\sigma = 25$.

ever, in images which are composed of relatively smooth regions then FoE-GA and FoE-CG produce better results as can be seen in Figure 5.

For completeness, we also report the performance our implementations of other methods, which were mentioned in Section 2.2, applied to learning the parameters of the FoE model for denoising. As shown in Table 3 our proposed training approach produced the best testing results. The

Training System	Average PSNR
Scharstein and Pal [12]	29.56
SPSA[9]	29.87
Foe-CG	30.21
Variational Mode Learning[14]	30.25
FoE-GA	30.55
Ours	30.58

Table 3. This table compares the results among the various methods used to learn the parameters of the FoE denoising model. The test images all had gaussian noise of standard deviation, $\sigma = 15$. Note that the FoE-GA results are obtained by terminating the optimization early, rather than finding the MAP solution in the FoE model. As argued in Section 2.1, this ties the model to a specific optimization procedure.

Figure 6. This figure shows the results when our trained model was applied to inpainting. Top: Original image. Bottom: Restored image

performance differences between our results and the other training methods are statistically significant at a 95% level using a signed rank test.

4.1. Inpainting

We also use our trained model to perform inpainting in a similar manner to what is done by Roth and Black [11]. The goal here is to remove unwanted parts of an image such as scratches and occluding objects while keeping the reconstructed parts as realistic as possible. Only the FoE prior is used and rather than updating all the pixels in the image only pixels specified by a user supplied mask are updated. In our case we used gradient descent with the 'bold driver' method to perform the inpainting.

Figure 6 shows our inpainting results which are qualitatively similar to those in [11]. This shows that even though our model is trained differently it is still able to perform the same tasks as well as or even better than the FoE model trained using contrastive divergence.

(a) Original image

(b) Noisy image, $\sigma = 25$

(c) Our denoised image, PSNR=29.33, SSIM=0.7341

(d) FoE-GA denoised image, PSNR=28.98, SSIM=0.7169

(e) FoE-CG denoised image, PSNR=28.58, SSIM=0.6988

Figure 3. An example from the Berkeley dataset comparing denoising results between the different systems.

(a) Original image

(b) Noisy image, $\sigma = 15$

(c) Our denoised image, PSNR=28.59, SSIM=0.7990

(d) FoE-GA denoised image, PSNR=28.28, SSIM=0.7845

(e) FoE-CG denoised image, PSNR=27.66, SSIM=0.7203

Figure 4. A second example from the Berkeley dataset comparing denoising results between the different systems. The over smoothing by FoE-GA/CG can be seen on the flat rock surface on the left of the image.

5. Conclusion

We have introduced a new approach for discriminatively training MRF parameters. In our approach, the MRF parameters are directly optimized so that the MAP solution scores as well as possible. This new approach allows us to develop a framework that is flexible, intuitively easy to understand and relatively simple to implement.

We demonstrate the effectiveness of our proposed approach by using it to learn the parameters for the Field of Experts model for denoising images. The performance of the model trained using our proposed method compares well with the results of the Field of Experts model trained using contrastive divergence when applied to the same tasks.

Acknowledgements

This work was supported by grant HM-15820810021 through the NGA NURI program.

References

[1] Y. Bengio. Gradient-based optimization of hyperparameters. *Neural Comput.*, 12(8):1889–1900, 2000.

[2] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Mach. Learn.*, 46(1-3):131–159, 2002.

(a) Original image (b) Noisy image, $\sigma = 15$ (c) Our denoised image, PSNR=36.24, SSIM=0.9087

(d) FoE-GA denoised image, PSNR=37.69, SSIM=0.9424 (e) FoE-CG denoised image, PSNR=39.00, SSIM=0.9707

Figure 5. An example from the Berkeley dataset comparing denoising results between the different systems on an image with flat regions.

[3] C. Do, C.-S. Foo, and A. Ng. Efficient multiple hyperparameter learning for log-linear models. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 377–384. MIT Press, Cambridge, MA, 2007.

[4] G. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(7):1771–1800, 2002.

[5] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002.

[6] S. S. Keerthi, V. Sindhwani, and O. Chapelle. An efficient method for gradient-based adaptation of hyperparameters in svm models. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 673–680. MIT Press, Cambridge, MA, 2007.

[7] Y. LeCun and F. Huang. Loss functions for discriminative training of energy-based models. In *Proc. of the 10-th International Workshop on Artificial Intelligence and Statistics (AIStats'05)*, 2005.

[8] A. Levin, R. Fergus, F. Durand, and W. T. Freeman. Image and depth from a conventional camera with a coded aperture. In *ACM SIGGRAPH 2007*, page 70, New York, NY, USA, 2007. ACM.

[9] Y. Li and D. P. Huttenlocher. Learning for optical flow using stochastic optimization. In *ECCV (2)*, pages 379–391, 2008.

[10] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, July 2001.

[11] S. Roth and M. Black. Field of experts: A framework for learning image priors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 860–867, 2005.

[12] D. Scharstein and C. Pal. Learning conditional random fields for stereo. *Computer Vision and Pattern Recognition, 2007.*, pages 1–8, June 2007.

[13] J. C. Spall. Implementation of the simultaneous perturbation algorithm for stochastic optimization. *American Statistician*, 1995. Submitted.

[14] M. F. Tappen. Utilizing variational optimization to learn markov random fields. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR07)*, 2007.

[15] M. F. Tappen, C. Liu, E. H. Adelson, and W. T. Freeman. Learning gaussian conditional random fields for low-level vision. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR07)*, 2007.

[16] B. Taskar, V. Chatalbashev, D. Koller, and C. Guestrin. Learning structured prediction models: A large margin approach. In *ICML*, 2005.

[17] B. Taskar, S. Lacoste-Julien, and M. Jordan. Structured prediction via the extragradient method. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 1345–1352. MIT Press, Cambridge, MA, 2006.

[18] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Map estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. *IEEE Transactions on Information Theory*, 51(11):3697–3717, November 2005.

[19] Z. Wang, A. C. Bovik, H. R. Sheikh, S. Member, E. P. Simoncelli, and S. Member. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13:600–612, 2004.

[20] Y. Weiss and W. T. Freeman. What makes a good model of natural images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.