Learning Gaussian Conditional Random Fields for Low-Level Vision

Marshall F. Tappen
University of Central Florida
Orlando, FL
mtappen@eecs.ucf.edu

Ce Liu Edward H. Adelson William T. Freeman
MIT CSAIL
Cambridge, MA 02139
{celiu, adelson, billf}@csail.mit.edu

Abstract

Markov Random Field (MRF) models are a popular tool for vision and image processing. Gaussian MRF models are particularly convenient to work with because they can be implemented using matrix and linear algebra routines. However, recent research has focused on on discrete-valued and non-convex MRF models because Gaussian models tend to over-smooth images and blur edges. In this paper, we show how to train a Gaussian Conditional Random Field (GCRF) model that overcomes this weakness and can outperform the non-convex Field of Experts model on the task of denoising images. A key advantage of the GCRF model is that the parameters of the model can be optimized efficiently on relatively large images. The competitive performance of the GCRF model and the ease of optimizing its parameters make the GCRF model an attractive option for vision and image processing applications.

1. Introduction

Markov Random Field (MRF) models are a popular tool for solving low-level vision problems. In the MRF model, the relationships between neighboring nodes, which often correspond to pixels, are modeled by local potential functions. The parameters of these functions can be effectively learned from training samples. The power of learning and inference in an MRF makes it a popular choice for low-level vision problems including image denoising [12], stereo reconstruction [3], and super-resolution [18].

However, learning and inference in MRFs are, in general, nontrivial problems. Because of the nonlinear and non-convex potential functions used in many MRFs, sophisticated sampling-based algorithms are often used for parameter learning [21, 12]. Unfortunately, sampling algorithms can be slow to converge. For certain types of discrete-valued graphs, there are efficient algorithms such as graph cuts [3] and loopy belief propagation [5], but learning still remains a hard problem.

Gaussian Markov random fields, which are MRFs where the variables are jointly Gaussian, have a long history in computer vision research [14]. Gaussian models are particularly convenient to work with because the inference in Gaussian models can be easily accomplished using linear algebra. Algorithms for numerical linear algebra are well understood, and efficient implementations are available. Nevertheless, Gaussian Markov random fields can result in over-smoothed images when the potential functions for neighboring nodes are homogeneous or isotropic, *i.e.* identical everywhere.

Typically, the key to success with Gaussian Markov random fields is to have the neighboring potential function dependent on the input signal, as in [10, 17]. These inhomogeneous or anisotropic Gaussian MRFs can overcome the weakness of the homogeneous ones by reconstructing piecewise smooth image with desirable properties. Because the potential functions now depend on the signal these MRFs are no longer generative models, but are instead conditional models. Therefore, these Gaussian MRFs can also be called Gaussian conditional random field (GCRF). In GCRFs, the parameters that describe how each potential function depends on the input signal is typically designed empirically and hand-tuned to generate the desired results. Although this might be feasible when there are few parameters, this approach does not scale-up when automatically designing and optimizing larger, more general models.

In this paper, we show how the parameters of GCRF can be efficiently for low-level vision tasks. We derive the learning algorithm by minimizing the error in the MAP solution of the model for the training samples. This tractable method of training, described in Section 3, enables the GCRF model to improve on the results produced by the more complex, non-convex Field of Experts model. The fact that GCRF models are relatively easy to train and can perform as well as more complicated models makes them potentially useful for a number of low-level vision and image-processing tasks. To demonstrate the convenience and power of the model, example implementations are available at `http://www.eecs.ucf.edu/~mtappen`.

Section 2 describes the GCRF model. The training algorithm is described in Section 3. Related work and models are discussed in Section 4. The GCRF model is compared to the Field of Experts model in Section 5.

2. Motivating the GCRF Model

The Gaussian Conditional Random Field (GCRF) model can be motivated in two ways: probabilistically as a Conditional Random Field [6], or as an estimator based on min-

imizing a cost function. Section 2.1 shows how the GCRF estimator can be motivated as a probabilistic model, while Section 2.3 shows how it can be motivated as the minimum of a quadratic cost function.

2.1. Basic Gaussian Conditional Random Field Model

Before presenting the actual Gaussian Conditional Random Field (GCRF) model used to denoise images, this section will first describe a GCRF model for images that suffers from the problem of oversmoothing. Section 2.2 will then show how this model can be improved to handle sharp edges and give much better results.

This GCRF model is defined by a set of linear features that are convolution kernels. For a set of features, $f_1 \dots f_{N_f}$, the probability density of an image, $\mathcal{X}$, conditioned on the observed image, $\mathcal{O}$, is defined to be

$$p(\mathcal{X}) = \frac{1}{Z} \exp\left(-\sum_{i=1}^{N_f} \sum_{x,y} \Big((\mathcal{X} * f_i)(x,y) - r_i(x,y;\mathcal{O}) \Big)^2 \right), \tag{1}$$

where $(\mathcal{X} * f_i)(x,y)$ denotes the value at location (x,y) in the image produced by convolving $\mathcal{X}$ with f_i. We assume that this image only contains those pixels where the entire filter fits in $\mathcal{X}$. This corresponds to "valid" convolution in MATLAB. The function $r_i(x,y;\mathcal{O})$ contains the estimated value of $(\mathcal{X} * f_i)(x,y)$. For each feature f_i, the function r_i uses the observed image $\mathcal{O}$ to estimate the value of the filter response at each pixel. In the rest of this paper, $r_i(x,y;\mathcal{O})$ will be shortened to r_i for conciseness.

The exponent:

$$\sum_{i=1}^{N_f} \sum_{x,y} \big((\mathcal{X} * f_i)(x,y) - r_i^2 \big) \tag{2}$$

can be written in matrix form by creating a set of matrices $F_1 \dots F_{N_f}$. Each matrix F_i performs the same set of linear operations as convolving an image with a filter f_i. In other words, if $\hat{\mathcal{X}}$ is a vector created by unwrapping the image $\mathcal{X}$, then $F_i\hat{\mathcal{X}}$ is identical to $\mathcal{X} * f_i$. These matrices can then be stacked and Equation 2 can be rewritten as

$$(F\hat{\mathcal{X}} - r)^T (F\hat{\mathcal{X}} - r), \tag{3}$$

where

$$F = \begin{bmatrix} F_1 \\ F_2 \\ \vdots \\ F_{N_f} \end{bmatrix} \quad \text{and} \quad r = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_{N_f} \end{bmatrix}.$$

Equation 3 can be rewritten in the standard form of the exponent of a multivariate normal distribution by setting the precision matrix, Λ^{-1}, to be F^TF and the mean, μ, to $(F^TF)^{-1}F^Tr$.

Note that the difference between $(\hat{\mathcal{X}} - \mu)^T\Lambda^{-1}(\hat{\mathcal{X}} - \mu)$ and Equation 3 is the constant $r^Tr - \mu^T\mu$. However, this value is constant for all X, so it only affects the normalization constant, Z, and the distributions defined using Equation 1 or Λ and μ are identical.

2.2. Utilizing Weights

One of the weaknesses of the GCRF model in Equation 1 is that it does not properly capture the statistics of natural images. Natural images tend to have smooth regions interrupted by sharp discontinuities. The Gaussian model imposes smoothness, but penalizes sharp edges. In applications like image denoising, optic flow, and range estimation, this violates the actual image statistics and leads to oversmoothing.

One way to avoid this oversmoothing is to use non-convex, robust potential functions, such as those used in [18] and [12]. Unfortunately, the convenience of the quadratic model is lost when using these models.

Alternatively, the quadratic model can be modified by assigning weights to the various terms in the sum. For example, if the filters, $f_1 \dots f_{N_f}$, include derivative filters, the basic GCRF model penalizes the strong derivatives around sharp edges. This causes the images estimated from basic GCRF models to have blurry edges. Using the observed image to lower the weight for those derivatives that cross edges will preserve the sharpness of the estimated image. It should be noted that this example is illustrative. Section 3 will show how use training data to learn the relationship between the observed image and the weights.

Adding weights to the exponent of Equation 1 can be expressed formally by modifying Equation 2:

$$\sum_{i=1}^{N_f} \sum_{x,y} w_i(x,y;\mathcal{O},\theta) \left((\mathcal{X} * f_i)(x,y) - r_i \right)^2, \tag{4}$$

where $w_i(x,y;\mathcal{O},\theta)$ is a positive weighting function that uses the observed image, $\mathcal{O}$, to assign a positive weight to each quadratic term in Equation 4.

In the denoising example that is described in Section 5, the weight function associated with each filter is based on the absolute response of the observed image to a set of multi-scale oriented edge and bar filters. These filters will be shown later in Figure 2. These filters are designed to be sensitive to edges. The underlying idea is that these filter responses will enable the system to guess where edges occur in the image and reduce the smoothness constraints appropriately.

Assigning a weight to each quadratic term improves the model's ability to handle strong edges. For a term based on a derivative filter, the weight could be increased in flat regions of the image and decreased along edges. This would enable the model to smooth out noise, while preserving sharp edges. Section 3 will discuss how to learn the parameters, θ, of this weighting function.

The potential difficulty in using this model is the fact that the weights must be computed from the observed image. This will be problematic if the observed image is too noisy or the necessary information is not easy to compute. Fortunately, as Section 5 will show, the system is able to perform well with high-noise levels when denoising images. This indicates that good weights can still be computed in high-noise problems.

2.3. Cost-Function Perspective

The GCRF model can also be motivated from a cost-function point of view. Let $h(\mathcal{O};\theta)$ be an estimator that uses the observed image, $\mathcal{O}$, to estimate an image. The estimate is the image $\mathcal{X}$ that minimizes the quadratic function:

$$\sum_{i=1}^{N_f}\sum_{x,y} w_i(x,y;\mathcal{O},\theta)\left((\mathcal{X}*f_i)(x,y)-r_i\right)^2. \quad (5)$$

Again, $r_i(x,y;\mathcal{O})$ has been shortened to r_i for conciseness.

Because Equation 5 contains a quadratic cost function, the minimum can be computed using the pseudo-inverse. Using the matrix notation from Section 2.1, this pseudo-inverse can be expressed as:

$$h(\mathcal{O};\theta) = \left(F^T W(\mathcal{O};\theta) F\right)^{-1} F^T W(\mathcal{O};\theta) r. \quad (6)$$

In this equation, we have introduced a diagonal matrix $W(\mathcal{O};\theta)$ that is a function of the observed image and parameters θ. This diagonal matrix is a block-diagonal matrix constructed from the diagonal sub-matrices $W_1(\mathcal{O};\theta)\ldots W_{N_f}(\mathcal{O};\theta)$. Each entry along the diagonal in the $W_i(\mathcal{O};\theta)$ matrices is equal to the weight of a term at a particular pixel, $w_i(x,y;\mathcal{O},\theta)$.

3. Learning the Parameters of a GCRF

Before the model described in the previous section can be used, the parameters of the weighting functions, $w_i(x,y;\mathcal{O},\theta)$, must be chosen. Traditionally, the parameters of Conditional Random Fields are found by maximizing the likelihood of the training data. Using the GCRF model from 2.2, the conditional log-likelihood of a training sample $\mathcal{T}$, conditioned on an associated observation $\mathcal{O}_\mathcal{T}$, is

$$\begin{aligned} &-\textstyle\sum_{i=1}^{N_f}\sum_{x,y} w_i(x,y;\mathcal{O}_\mathcal{T},\theta)\left((\mathcal{T}*f_i)(x,y)-r_i\right)^2 \\ &-\log\textstyle\int_\mathcal{X}\exp\Big(-\sum_{i=1}^{N_f}\sum_{x,y} w_i(x,y;\mathcal{O}_\mathcal{T},\theta)\times \\ &\qquad\qquad \left((\mathcal{X}*f_i)(x,y)-r_i\right)^2\Big)\,d\mathcal{X} \end{aligned} \quad (7)$$

and its partial derivative with respect to a parameter θ_i is:

$$\begin{aligned} &-\textstyle\sum_{i=1}^{N_f}\sum_{x,y}\frac{\partial w_i(x,y;\mathcal{O}_\mathcal{T},\theta)}{\partial\theta_i}\left((\mathcal{T}*f_i)(x,y)-r_i\right)^2 \\ &+\frac{1}{Z}\int_\mathcal{X} \frac{\partial w_i(x,y;\mathcal{O}_\mathcal{T},\theta)}{\partial\theta_i}\times \\ &\qquad \exp\left(-\textstyle\sum_{i=1}^{N_f}\sum_{x,y} w_i(x,y;\mathcal{O}_\mathcal{T},\theta)\times\right. \\ &\qquad\qquad \left.\left((\mathcal{X}*f_i)(x,y)-r_i\right)^2\right)d\mathcal{X}. \end{aligned} \quad (8)$$

Notice that the integration on the righthand of Equation 8 can be rewritten as an expectation. This implies that computing the gradient of the log-likelihood with respect to the parameters requires computing the expected values of the clique potentials or feature functions. For a Gaussian model, the first step in computing these expectations is inverting the inverse covariance matrix (precision matrix). The key difficulty with using the log-likelihood to fit the GCRF parameters lies in this matrix inversion. While the matrix $\left(F^T W(\mathcal{O};\theta)F\right)$ is sparse, $\left(F^T W(\mathcal{O};\theta)F\right)^{-1}$ will be dense. For a 256×256 image, storing this dense matrix in memory with single-precision values will require 16GB of memory. In addition, inverting a matrix is an $O(N^3)$ operation, making it infeasible for large images.

3.1. Discriminative Learning for GCRFs

The matrix inversion can be avoided by instead penalizing the difference between the mode of the conditional distribution and the ground truth. The penalty is expressed using a loss function $L(\mathcal{Y},\mathcal{T})$ that assigns a loss for an image $\mathcal{Y}$ based on its difference from the ground-truth image $\mathcal{T}$. In the GCRF model, the mode of the conditional distribution is the conditional mean, so the cost, $C(\theta,\mathcal{O},\mathcal{T})$, associated with a particular set of parameters θ is

$$C(\theta,\mathcal{O},\mathcal{T}) = L\Big(\left(F^T W(\mathcal{O};\theta)F\right)^{-1}F^T W(\mathcal{O};\theta)r,\ \mathcal{T}\Big). \quad (9)$$

The parameters θ can be found by minimizing $C(\theta,\mathcal{O},\mathcal{T})$ using gradient descent techniques. For convenience when computing the gradient, we define μ such that

$$\mu = \left(F^T W(\mathcal{O};\theta)F\right)^{-1}F^T W(\mathcal{O};\theta)r. \quad (10)$$

Using the chain rule, the partial derivative of $C(\cdot)$ with respect to a parameter θ_i is

$$\frac{\partial C(\theta,\mathcal{O},\mathcal{T})}{\partial\theta_i} = \frac{\partial C(\theta,\mathcal{O},\mathcal{T})}{\partial\mu}\frac{\partial\mu}{\partial\theta_i}. \quad (11)$$

Note that $\frac{\partial C(\theta,\mathcal{O},\mathcal{T})}{\partial\mu}$ is a $1\times N_p$ vector, where N_p is the number of pixels in the image, equal to

$$\left[\frac{\partial C(\theta,\mathcal{O},\mathcal{T})}{\partial\mu_1}\cdots\frac{\partial C(\theta,\mathcal{O},\mathcal{T})}{\partial\mu_{N_p}}\right]. \quad (12)$$

The derivative vector $\frac{\partial\mu}{\partial\theta_i}$ is an $N_p\times1$ vector with each element equal to $\frac{\partial\mu_j}{\partial\theta_i}$ $(j=1,\cdots,N_p)$.

If the loss function is the squared-difference between μ and the ground-truth image, $\mathcal{T}$, then

$$C(\theta,\mathcal{O},\mathcal{T}) = (\mu-\mathcal{T})^T(\mu-\mathcal{T}), \quad (13)$$

and

$$\frac{\partial C(\theta,\mathcal{O},\mathcal{T})}{\partial\mu} = 2(\mu-\mathcal{T}).$$

3.2. Differentiating μ

The vector μ, defined in Equation 10, can be differentiated with respect to a parameter θ_i using the matrix derivative identity:

$$\frac{\partial A^{-1}}{\partial x} = -A^{-1}\frac{\partial A}{\partial x}A^{-1}. \quad (14)$$

Using this identity,

$$\begin{aligned} \tfrac{\partial\mu}{\partial\theta_i} = \ & -\left(F^T W(\mathcal{O};\theta)F\right)^{-1}F^T\tfrac{\partial W(\theta,\mathcal{O})}{\partial\theta_i} \\ & \times\left(F^T W(\mathcal{O};\theta)F\right)^{-1}F^T W(\mathcal{O};\theta)r \\ & +\left(F^T W(\mathcal{O};\theta)F\right)^{-1}F^T\tfrac{\partial W(\theta,\mathcal{O})}{\partial\theta_i}r. \end{aligned} \quad (15)$$

Equation 15 can be expressed much more succinctly by replacing $\left(F^T W(\mathcal{O};\theta)F\right)^{-1}F^T W(\mathcal{O};\theta)r$ with μ and factoring:

$$\tfrac{\partial\mu}{\partial\theta_i} = \left(F^T W(\mathcal{O};\theta)F\right)^{-1}F^T\tfrac{\partial W(\theta,\mathcal{O})}{\partial\theta_i}\left(-F\mu+r\right). \quad (16)$$

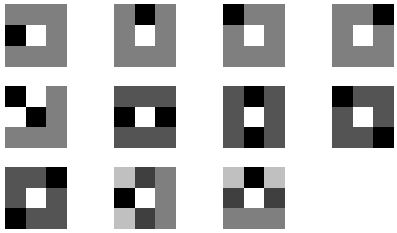


Figure 1. The filters used in the GCRF model for denoising images. These filters are a set of derivatives.

3.3. Computational Implications

Combining Equation 11 with Equation 16, the derivative of the $C(\cdot)$ with respect to a parameter θ_i is

$$\frac{\frac{\partial C(\theta,\mathcal{O},\mathcal{T})}{\partial \theta_i}}{\frac{\partial C(\theta,\mathcal{O},\mathcal{T})}{\partial \mu}} = \left(F^T W(\mathcal{O};\theta) F\right)^{-1} F^T \frac{\partial W(\theta,\mathcal{O})}{\partial \theta_i} \left(-F\mu + r\right). \quad (17)$$

The most expensive step in Equation 17 is computing μ and $\frac{\partial C(\theta,\mathcal{O},\mathcal{T})}{\partial \mu}\left(F^T W(\mathcal{O};\theta) F\right)^{-1}$. Both of these terms involve calculating the product of an inverse matrix and a vector. This can be computed efficiently in $O(N_p^2)$ time using numerical linear algebra techniques. In our implementation, we use the "backslash" operator in MATLAB to compute these vectors.

Also notice that if the matrix multiplications are ordered correctly, each of these vectors must only be computed once to calculate the gradient with respect to all θ_i. This makes the complexity of computing the gradient $O(N_p^2)$. Compared to maximizing the likelihood, which requires $O(N_p^3)$ matrix inversions to compute the covariance matrix of the distribution, this is a significant reduction in computation. Even for a relatively small 128×128 image, there will be an immense savings in computation.

4. Related Work

This GCRF model is similar to that used by Tappen, Adelson, and Freeman to learn parameters for decomposing an image into shading and albedo components [16]. That system was also used to denoise images, though with an error slightly higher than the Field of Experts model. The primary differences between that system and the system described here is that many more filters are used and the filter responses are all set to zero. In [16], a semi-parametric regression system was used to estimate the filter responses for horizontal and vertical first-order derivatives. The GCRF model described here uses higher-order derivatives and constrains them to be zero. This better captures the piecewise continuity in images.

A distinguishing characteristic of this method for training is that model parameters are optimized using a loss function that measures the difference between the most likely image and the ground-truth image. This type of training has also been used for discrete-valued graphical models by Collins [4], Taskar et al. [20, 19], and Altun et al. [1]. In [4], Collins shows how a variant of the perceptron algorithm can be used to maximize a margin, similar to the margin in the Support Vector Machine literature, between the correct output from a graphical model and all other possible outputs. In [20], Taskar et al. show how a loss function can be upper-bounded with a hinge function and the parameters can be found using convex optimization. In [1], Altun et al. present a similar formulation.

Unlike [20] and [1], this work does not rely on machinery from convex optimization. Instead, the gradient is calculated analytically and standard gradient-based optimization techniques can be used to optimize the loss function. Relying on convex optimization limits the types of weight-functions and loss-functions that can be used. The only restriction on the weight functions in the GCRF formulation presented in this paper is that they be differentiable.

It should be noted that in our formulation, the learning algorithm takes into account the loops in the graph representation of the CRF, unlike systems trained using pseudo-likelihoods, such as [8].

The GCRF model can also be viewed as a type of Energy-Based Model [7]. The advantage of Energy-Based Models is that the partition function does not need to be computed when training.

In recent work, Lyu and Simoncelli have also proposed a denoising model which uses Gaussian Random Fields to model Fields of Gaussian Scale Mixtures [9]. A key difference between this work and the GCRF model proposed here is that the GCRF model is not used as a part of a non-Gaussian model. Instead, we focus on optimizing the parameters of the Gaussian model to achieve the best possible performance. This enables us to take advantage of the computational benefits of a purely Gaussian model, which are described in Section 1.

5. Comparing GCRFs to Field of Experts Models for Denoising

To demonstrate the power of GCRFs on a real-world problem, we constructed a GCRF model for denoising images. As stated earlier, a model has three major components: the linear features $f_1 \ldots f_{N_f}$, the estimated responses of the features $r_1(\cdot) \ldots r_{N_f}(\cdot)$, and the weighting functions $w_1(\cdot) \ldots w_{N_f}(\cdot)$.

To keep the linear system invertible, the first filter, f_1 is the identity filter with $r_1(x, y; \mathcal{O}) = \mathcal{O}(x, y)$ and $w_1(x, y; \mathcal{O}) = 1$. In other words, the first feature constrains the mean to be somewhat close to the observation.

The remaining filters are a set of eleven 3×3 derivative filters. The filters are shown in Figure 1. For each of these filters, the estimated response, $r_i(\cdot)$, equals zero. Essentially, these filters are imposing smoothness constraints.

The weight function associated with each filter is based on the absolute response of the observed image $\mathcal{O}$ to a set of multi-scale oriented edge and bar filters. These filters are shown in Figure 2. These filters operate at multiple scales in the image. The size of the filters in each of the three rows is 11 pixels, 21 pixels, and 31 pixels. The filters were designed to be elongated in order to be sensitive to extended edges. Due to the large size of the filters, reflected border

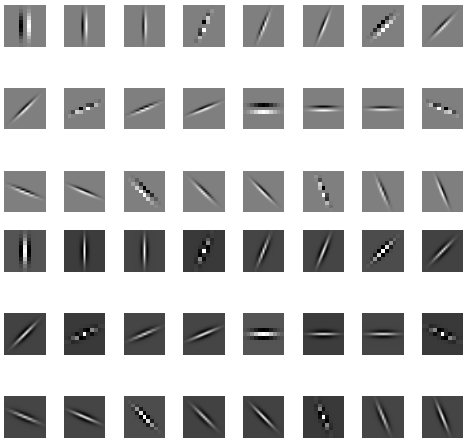


Figure 2. The filters used for generating the features that are used to estimate the weight $w_i(x, y; \mathcal{O})$ of each term in the GCRF model.

handling was used when computing the filter responses. A third set of responses was created by squaring the responses of the corresponding edge and bar filters, then adding these values.

Altogether, there are 72 filter responses at each pixel in the image. Denoting the absolute response to each filter at location (x, y) as $a_1^{(x,y)} \dots a_{72}^{(x,y)}$, where $a_{72}^{(x,y)}$ is a feature containing all ones, the weighting function is

$$w_i(x, y; \mathcal{O}) = \exp\left(\sum_{j=1}^{72} \theta_j^i a_j^{(x,y)}\right), \tag{18}$$

where θ_j^i represents the weight associated with response j for filter i. The complete set of parameters, θ, is the concatenation of the response weights for each filter, $\theta = [\theta^1 \theta^2 \dots \theta^{12}]$. The exponential ensures that the weight is always positive.

The intuition behind this weight function is that the observed image, $\mathcal{O}$, is being used to guess where edges occur in the image. The weight of the smoothing constraints can then be reduced in those areas.

5.1. Training the GCRF Model

We trained the GCRF model by optimizing the response weights, θ, to minimize the absolute value of the difference between between the predicted image and the true noiseless image, $\mathcal{T}$. To make the loss function differentiable, we used $\sqrt{(x^2 + 0.1)}$ to approximate $|x|$. Using the derivations from Section 2.2, the derivative with respect to a particular θ_j^i is

$$\begin{aligned} \frac{\partial C}{\partial \theta_j^i} = \; & \frac{(\mu - \mathcal{T})}{((\mu-\mathcal{T})^2+0.1)^{\frac{1}{2}}} \times \\ & \left(F^T W F\right)^{-1} F^T \frac{\partial W(\theta, \mathcal{O})}{\partial \theta_j^i} \left(-F\mu + r\right). \end{aligned} \tag{19}$$

For conciseness, we have dropped the arguments from C and W. Note that μ and $\mathcal{T}$ are vectors, so the division and square-root operations in $\frac{(\mu-\mathcal{T})}{((\mu-\mathcal{T})^2+0.1)^{\frac{1}{2}}}$ are element-wise operations.

Model	Average RMS Error (Lower is Better)	Average PSNR (Higher is Better)
Field of Experts	11.04	27.59
GCRF	10.47	28.04

Table 1. This table compares the average error in denoised estimates produced by the GCRF model described in this paper and the Field of Experts model from [12]. The results are averaged over the same 68 test images used in [12] The standard deviation of the noise, σ, in the observations was 25.

We found that optimizing the parameters of the model with this smoothed absolute error criterion led to an estimator that generalized better than the estimator trained using a squared-error criterion. Optimizing under a squared-error criterion causes the algorithm to focus on the images with larger errors. These images are treated equally under the absolute error criterion. This helps generalization performance because outlier images do not receive undue weight during training.

The θ vector was optimized using a gradient-descent algorithm with the "bold-driver" heuristic [2]. Under this heuristic, the step size was decreased each time the error in the denoised estimates increased.

The system was trained on forty 129×129 patches taken from the same training images used by Roth and Black. White Gaussian noise, with a standard deviation of 25, was added to each observed image. Computing the gradient for two images required approximately two minutes on a 2.4 GHz Intel Xeon processor. Using a computing cluster, we were able to train the model in a few hours. The training images used in training the GCRF model were much larger than the 15×15 patches used to train the Field of Experts model, which was also trained using a cluster. It is useful to be able to train on large images because they better match the test images.

5.2. Results

For comparison, we ran the Field of Experts (FoE) implementation provided by Roth and Black on the same 68 test images from the Berkeley Segmentation Database that Roth and Black identified as test images. The observations were created by adding white Gaussian noise with a standard deviation of 25 to each image. The FoE model denoises images by using gradient-descent to minimize a robust cost function. The results vary depending on the step-size parameter used. We found that we obtained images with the highest log-posteriors by optimizing for 5000 iterations with a step-size of 0.6. This enabled us to compare the two models fairly because the results from the GCRF model are optimal under that model.

As Table 1 shows, the GCRF model produces denoised images with a lower error than the Field of Experts Model. In addition, as Table 2 shows, the GCRF model requires nearly half the time to produce a denoised 481×321 image. Note that in this table, we have shown the time required for 2500 iterations of the steepest-descent algorithm, which is the number of iterations suggested by Roth and Black.

Model	Time to Produce Estimates (sec)
Field of Experts (2500 iterations)	376.9
GCRF (MATLAB linear solver)	180.7
GCRF (300 conjugate gradient iterations)	97.82

Table 2. A comparison of the amount of time needed to denoise an image using either the Field of Experts model, which uses steepest-descent, or the GCRF model, which uses the MATLAB linear solver. The Conjugate Gradients method is described in Section 5.3. Times were computed on a 3.2Ghz Intel Xeon Processor.

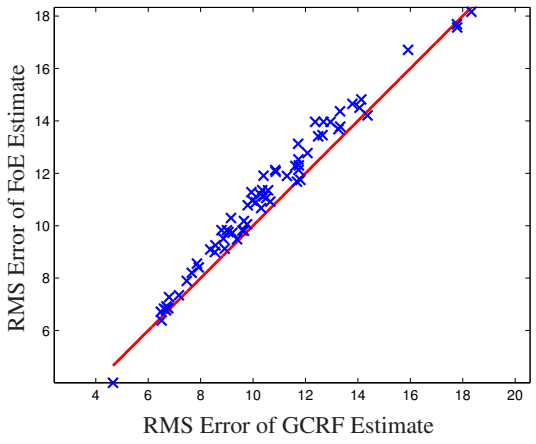


Figure 3. This scatterplot shows the RMS error in each of the 68 test images when denoised with either the Field of Experts (FoE) model or the GCRF model. The GCRF model outperforms the FoE model for all images where the point is above the red line.

To compare the performance of the two models across the test set, Figure 3 shows of a scatterplot of the RMS errors of the images estimated with the two models. Those images where the GCRF model produces a better estimate appear above the red line. As Figure 3 shows, the GCRF model produces the better estimate for the majority of the images. It is important to note that the RMS error is just one type of error metric. Fortunately, if good performance is desired under a specific error metric, the GCRF model can be easily adapted to different error metrics. It should be noted that the GSM denoising algorithm [11] still outperforms both of these models in terms of PSNR.

Figures 4, 5, and 6 show three examples of images from the test set where the GCRF model produces better denoised images in terms of PSNR. Figure 7 shows an example where the Field of Experts model outperforms the GCRF model in terms of PSNR.

Qualitatively, the GCRF model does a much better job at preserving the fine-scale texture. This comes at the expense eliminating low-frequency noise in smooth portions of the image, which can be seen in Figure 7.

5.3. Iterative Estimation in GCRF models

For large images, the overhead of constructing the matrices and solving the linear system may be prohibitive. In these situations, the denoised image can be obtained by minimizing Equation 5 iteratively. A significant advantage of the GCRF model is that denoising is accomplished by solving a linear system. This linear system can be solved iteratively using the conjugate gradient method. This method converges more quickly than standard steepest-descent [13].

Much of the computation in the conjugate gradient method involves matrix-vector products. When using this method for the GCRF model, each of the matrix-vector products correspond to computing a convolution. This makes each iteration of the conjugate gradient slightly more expensive than an iteration of steepest-descent.

One advantage of the conjugate-gradient method is that there are no step-size parameters. We also found that it converged quickly. After 300 iterations of the conjugate gradient method, the average absolute difference, per pixel, between the iterative estimate and the estimate found with the direct matrix solver was 5.3851e-05. As Table 2 shows, using the iterative solver significantly reduces the amount of time needed to produce an estimate. Further improvements in speed may be possible by using a preconditioner suited for images, such as the preconditioner described in [15].

While the conjugate gradients method could be used to find estimates in the Field of Experts model, the non-linear conjugate gradient algorithm requires line-searches, which would affect its overall performance.

6. Conclusion – Advantages and Limitations of the GCRF Model

We have shown how a Gaussian Conditional Random Field (GCRF) model can outperform the non-convex Field of Experts model on the task of denoising images. The GCRF model produced more accurate estimates, in terms of PSNR, and required less computation to produce estimates. This is significant because of the advantages of the GCRF model:

1. It is convenient to work with. The GCRF model can be implemented using standard numerical linear algebra routines.

2. As Sections 3 and 5 showed, the model can be trained on relatively large images.

3. Any differentiable loss function on the estimated images can be used during training. This is important because the behavior of a system can be significantly influenced by the loss function used to train it. Allowing for different loss functions during training enables a system's behavior to be controlled more precisely.

One potential limitation of this system is that that the system is trained for a specific noise level. This would seem to indicate that a generative model, such as the FoE model, would be more useful for applications involving variable amounts of noise. However, this potential limitation of the GCRF model must be taken in context. First, as we have demonstrated, training the GCRF model is relatively simple, thus making it feasible to train the model for different noise levels. Variable noise could be accomodated by choosing the appropriate weighting coefficients. Second, to

(a) Original (b) Noisy Observation (c) Results using FoE Model, PSNR = 25.23 (d) Results using GCRF Model, PSNR= 26.28

Figure 4. In this denoising task, the GCRF model produces a more accurate estimate of the true image than the Field of Experts (FoE) model in terms of PSNR. The GCRF model also does a better job of preserving the texture in the image.

(a) Original (b) Noisy Observation (c) Results using FoE model, PSNR = 25.77 (d) Results using GCRF model, PSNR = 26.75

Figure 5. A second example where the GCRF model produces superior results in terms of PSNR. Again, the GCRF better preserves texture.

achieve optimizal performance in the FoE model, the current state-of-the-art generative model, the parameters predicted by the generative model cannot be used. Instead, the weight of the likelihood term in the log-posterior must be set to a value that is determined conditionally by the noise level in the observation. Thus, the FoE model can be viewed as a conditional model, somewhat akin to the GCRF model.

Another potential limitation of the GCRF is its reliance on the input for setting the weights. Data may be missing or the observations may be of very poor quality. In difficult cases, it may be useful to perform simple pre-processing, such as bilinear interpolation to fill in missing data, before estimating the weights.

Overall though, the GCRF's convenience, combined with its ability to perform as well as non-convex model, make this model an attractive model for low-level vision and image processing applications.

Acknowledgments

Funding for this research was provided by NGA NEGI-1582-04-0004 and Shell Research.

References

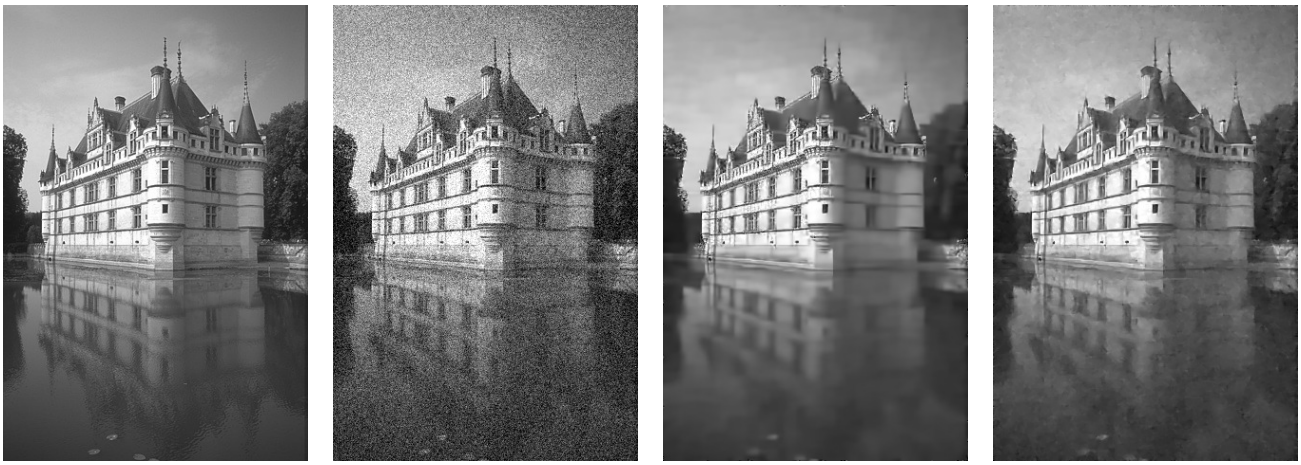
[1] Y. Altun, I. Tsochantaridis, and T. Hofmann. Hidden markov support vector machines. In *ICML*, 2003.

[2] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

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

[4] M. Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *EMNLP 2002*, 2002.

[5] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning low-level vision. *IJCV*, 40(1):25–47, 2000.



(a) Original (b) Noisy Observation (c) Result from FoE, PSNR = 28.59 (d) Result from GCRF, PSNR = 28.68

Figure 6. In this comparison, the PSNR for the Field of Experts result is slightly lower than the PSNR reported in [12]. We found that as the Field of Experts code optimized the denoised images, the PSNR began to decrease as the log-posterior of the estimate continued to increase. However, the result in (c) is visually quite similar to that in [12].

(a) Original (b) Noisy Observation (c) Result using FoE model, PSNR = 36.07 (d) Result using GCRF model, PSNR = 34.77

Figure 7. These images are an example of how the Field of Experts model can produce superior estimates in terms of PSNR. The FoE model is better at removing low-frequency noise from flat regions.

[6] J. Lafferty, F. Pereira, and A. McCallum. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.

[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] C. Lee, S. Wang, F. Jiao, D. Schuurmans, and R. Greiner. Statistical modeling of images with fields of gaussian scale mixtures. In *NIPS*, volume 19, May 2007.

[9] S. Lyu and E. P. Simoncelli. Statistical modeling of images with fields of gaussian scale mixtures. In *NIPS*, volume 19, May 2007.

[10] H. Nagel and W. Enkelmann. An investigation of smoothness constraint for the estimation of displacement vector fiels from images sequences. *IEEE TPAMI*, 8:565–593, 1986.

[11] J. Portilla, V. Strela, M. Wainwright, and E. P. Simoncelli. Image denoising using scale mixtures of gaussians in the wavelet domain. *IEEE Trans. Image Processing*, 12(11):1338–1351, November 2003.

[12] S. Roth and M. Black. Field of experts: A framework for learning image priors. In *Proc. CVPR*, volume 2, pages 860–867, 2005.

[13] J. R. Shewchuk. An introduction to the conjugate gradient method without the agonizing pain. Available at `http://www.cs.cmu.edu/~jrs/jrspapers.html`.

[14] R. Szeliski. Bayesian modeling of uncertainty in low-level vision. *IJCV*, 5(3):271–301, 1990.

[15] R. Szeliski. Locally adapted hierarchical basis preconditioning. *ACM SIGGRAPH*, 25(3):1135–1143, August 2006.

[16] M. F. Tappen, E. H. Adelson, and W. T. Freeman. Estimating intrinsic component images using non-linear regression. In *Proc. CVPR*, volume 2, pages 1992–1999, 2006.

[17] M. F. Tappen, W. T. Freeman, and E. H. Adelson. Recovering intrinsic images from a single image. In *NIPS*, pages 1343–1350, 2003.

[18] M. F. Tappen, B. C. Russell, and W. T. Freeman. Efficient graphical models for processing images. In *Proc. CVPR*, volume 2, pages 673–680, 2004.

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

[20] B. Taskar, S. Lacoste-Julien, and M. Jordan. Structured prediction via the extragradient method. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *NIPS*. MIT Press, Cambridge, MA, 2006.

[21] S. C. Zhu, Y. Wu, and D. Mumford. Filters, random fields and maximum entropy (frame): Towards a unified theory for texture modeling. *IJCV*, 27(2):107–126, 1998.