Artificial Neural Networks and Statistical Pattern Recognition: Old and New Connections,
I. Sethi and A. Jain, editors, pp. 155-174, Elsevier Science Press, 1991

Markov Random Fields and Neural Networks with Applications to Early Vision Problems

A. Rangarajan, R. Chellappa and B. S. Manjunath

Signal and Image Processing Institute, Powell Hall of Engineering (PHE) 306, University of Southern California, Los Angeles, CA 90089-0272

Abstract

The current resurgence of interest in Neural Networks has opened up several basic issues. In this chapter, we explore the connections between this area and Markov Random Fields. We are specifically concerned with early vision problems which have already benefited from a parallel and distributed computing perspective. We explore the relationships between the two fields at two different levels of a computational approach. Applications highlighting specific instances where ideas from the two approaches intertwine are discussed.

1 INTRODUCTION

Markov Random Fields (MRFs) and Neural Networks (NNs) have a history dating back over thirty years. While MRFs primarily serve as tools for model building, NNs are much more ambitious; their applications range from computing least square estimates to building parallel computers. However, both fields offer paradigms for the construction of early vision modules and there are numerous connections between them.

We are primarily interested in exploring the relationships between the two fields as they relate to the construction of early vision modules. In this regard, we suggest that the frameworks and principal ideas be compared at different levels of a computational approach. Marr [1] was the first to suggest that there are three levels of a computational approach; computational, algorithmic and hardware. We feel that comparisons at the first two levels are appropriate and illuminating for this audience. In review, the computational level concerns itself with *what* is being computed. The specific scheme of computation is not of interest, only the goal of computation is important. The algorithmic level concerns itself with *how* the goal is computed. An example of the goal of computation is the Traveling Salesman Problem and an example of a method of solution is the Hopfield network [2]. We will not be discussing the relationships at the level of hardware implementation; it is important albeit premature.

MRFs are an important class of stochastic models and have been applied to problems like image estimation and texture segmentation. Once the model variables have been chosen, an MRF is completely specified by the joint distribution over these variables.

The important characteristic of MRFs in early vision is that the conditional densities are dependent only on a small neighborhood surrounding the variable of interest. When this is the case, the conditional densities are called *local characteristics*. In order to perform feature extraction, the joint density is usually specified over a set of variables which are closely related to the data and a further set of attribute variables which categorize the data. The attribute variables are termed *unobservable* since they are not part of the observed data. The joint density of the data variables (computed by integrating out the unobservables) is usually fully connected [3]. This property suggests that fully connected, non trivial models can emerge from an intuitive choice of attributes and local characteristics, thereby allowing us to create hierarchical models which are capable of extracting higher order features. The higher order features constitute an efficient *internal* representation. The unobservables are specified by the designer rendering the internal representation immutable. The goal of computation is the minimum mean square error estimate (MMSE), maximum *a posteriori* (MAP) estimate or in general any estimate based on minimum expected cost. At the algorithmic level, there are several methods, both deterministic and stochastic, of solving the problem. However, in addition to the state estimation problem, we have to solve the parameter estimation problem. The joint distribution is a function of these parameters which have to be specified *apriori* before the cost minimization is done. The long-term goal is rapid feature extraction and classification with continual adaptation to incoming data.

There are a large number of NN related models. The basic idea drawn from biology and neurobiology is to have networks with neuron-like processing elements with simple dynamics and massive interconnections which are capable of parallel, global computation and learning internal representations. In most neural net schemes, the aim is to let the internal representation of the data emerge from a set of *hidden* units which are connected to the data. The connections serve as the long term memory and are modified in order to better approximate the distribution of the incoming data. Higher order statistics are represented by interconnections (which are second-order) and hidden units. In order to take decisions, neuronal dynamics have to be non-linear [4]. The goal of computation is decided by a teacher who then trains the network (with a given number of hidden units) on a set of training samples [4]. There is no need to restrict the interconnections to second-order. Higher-order interconnections can more closely approximate the incoming distribution [5]. In self-supervised schemes, self-organization plays the role of the external teacher [6]. Category formation and knowledge representation are now machine driven with no supervision. The goal of the computation is for the machine to discover the intrinsic complexity (or the information needed for a minimal description) of the data [7]. The algorithmic level problems are the choice of the number of hidden units, the learning algorithm etc. The long term goal once again is rapid feature extraction and classification with continual adaptation to incoming data.

The two paradigms are strongly interrelated. Several ideas are common due to mutually dependent co-origination a few years ago. We wish to re-examine the basic ideas and suggest the different levels at which the two approaches can continue to benefit each other. Section 2 is devoted to this issue. In Section 3, we present two applications highlighting some of the ideas presented in Section 2. Conclusions are presented in Section 4.

2 RELATIONSHIP BETWEEN THE TWO FIELDS

2.1 PRELIMINARIES

In this section, we examine the basic ideas in MRFs and NNs.

2.1.1 MARKOV RANDOM FIELDS

An MRF can be completely specified by the joint distribution over the variables. We denote the data by $\mathbf{Y}$. Associated with the data, we have the process $\mathbf{X}$ and in addition, we have the attribute process $\mathbf{L}$. One of the most interesting aspects of MRFs is that the joint distribution is an MRF if and only if it is also Gibbsian [3]. The Gibbs distribution is written as follows.

$$\mathcal{P}(\mathbf{X} = \mathbf{x}, \mathbf{L} = \mathbf{l} | \mathbf{Y} = \mathbf{y}) = \frac{1}{Z} \exp(-\beta \mathcal{H}(\mathbf{x}, \mathbf{l}, \mathbf{y})) \tag{1}$$

where Z is the *partition function*, $\beta = \frac{1}{T}$ and T is the temperature, and $\mathcal{H}$ is the *energy function* defined over all the processes. The partition function is a function of the observed data $\mathbf{y}$ and the parameters involved in the specification of $\mathcal{H}$. Adopting a Bayesian viewpoint, it is easy to switch between the various distributions. Equation (1) is the *posterior* distribution arising from the *degradation* model $\mathcal{P}(\mathbf{Y} = \mathbf{y} | \mathbf{X} = \mathbf{x}, \mathbf{L} = \mathbf{l})$ and the *prior* distribution $\mathcal{P}(\mathbf{X} = \mathbf{x}, \mathbf{L} = \mathbf{l})$. The prior distribution encodes our beliefs about the processes $\mathbf{X}$ and $\mathbf{L}$ and is Gibbs. The degradation model is specified by the (usually) known transformation from the representation to the observed data and is also Gibbs. In principle, the recipe for inference is clear once these distributions have been specified. The parameters are estimated using maximum likelihood (ML) on the marginal density of the data (which is a function of the parameters) and then the expected cost is minimized with respect to $\mathbf{X}$ and $\mathbf{L}$. In practice, both problems (parameter estimation and cost minimization) are intractable in their pure form. For example, if cost minimization reduces to finding the MAP estimate, then the problem is usually NP-complete and only simulated annealing (SA) is guaranteed to asymptotically reach the optimum solution. In the case of parameter estimation, ML cannot be performed due to the intractability of the distribution of $\mathbf{Y}$.

In recent years, several alternatives have been proposed. It is possible to design efficient deterministic algorithms that obtain good sub-optimal MAP or MMSE estimates. For instance, the Iterated Conditional Mode (ICM) [8] algorithm iteratively maximizes the conditional density of each variable with the others held fixed. Similarly the Iterated Conditional Expectations (ICE) algorithm [9] iteratively computes the conditional expectation of each variable with the others held fixed. ICM and ICE yield good, sub-optimal MAP and MMSE estimates respectively. Both methods exploit the Markov structure. In addition, continuation methods [10] have been developed which are akin to some form of deterministic annealing since they involve the gradual change of a control parameter. The problem of parameter estimation has been partially solved by the introduction of pseudo-likelihood (PL) [11] techniques which maximize the product of the conditional densities with respect to the unknown parameters. This avoids the intractable problem of having to compute the partition function. Unfortunately, PL requires prior knowledge of $\mathbf{X}$ and $\mathbf{L}$ which is unavailable. The well known Expectation-Maximization (EM) algorithm [12] has been adapted to jointly solve for the parameters and states.

2.1.2 NEURAL NETWORKS

There are several models of learning and perceptual inference in the Neural Network literature. In this section, we explore the fundamental ideas of the some of these models and of the Boltzmann Machine (BZM) [13] in particular. The BZM has very close ties to the general area of MRFs.

The BZM consists of two phases, a fixed phase and a free phase. As previously mentioned, internal representations are generated by a set of hidden binary units and learning is accomplished by modifying the interconnections between the units. In the fixed phase, the environmental patterns clamp the input/output units and the hidden units are relaxed using SA. In the free phase, all units (sometimes the inputs are clamped) are relaxed using SA. Co-occurrence statistics are collected in both phases and the weights are modified based on the difference between the statistics. It can be shown that this kind of weight modification corresponds to gradient descent on the Kullback information gain [14] which measures the distance between the distributions generated during the fixed and free phases. In essence, the free phase mimics the distribution over the incoming patterns and the weights are modified in order to better approximate the incoming distribution. The order in which the patterns are presented to the system is unimportant, but obviously their frequency is. A pure implementation is infeasible in vision problems due to the excessive computational requirements of the SA algorithm.

The most important alternative that has emerged so far is the Mean Field Theory (MFT) Learning Algorithm [15]. MFT involves tracking the expected value of the units instead of generating their values using a probabilistic (heat bath) algorithm. A very interesting point in relation to all that has already transpired is that MFT exploits an approximation to the partition function (for binary units) that was first formulated in statistical physics. With this approximation in place, a deterministic relaxation algorithm results in place of SA and the equation for the weight change is left intact. MFT has close ties to ICE in the context of MMSE estimation since the expected value of the state variables can be obtained from the partition function. Recall that the MMSE estimate involves computing the conditional expectations of the variables given the data.

The BZM is a supervised pattern classifier. There exist several alternatives to classical unsupervised classifiers in the NN literature. The most notable are Adaptive Resonance Theory (ART) and its variants [6], the Neocognitron [16] and most competitive learning models [4]. The key in most of these models is that learning and category formation are conducted by learning rules which are not obtained by matching the internal representation with an environmentally imposed set of associations. Instead, the learning rules are specified as a dynamical system on the weights. Unsupervised learning results in an organization of the data into categories or features. This process is called self-organization since the system discovers the categories without supervision.

2.2 RELATIONSHIP AT THE COMPUTATIONAL LEVEL

The key differences emerging at the computational level are as follows. MRFs rely heavily on conditional distributions whose local characteristics have small neighborhoods. Also, the models are usually adopted in advance and held fixed. In addition, the models are not restricted to second-order distributions. There is no restriction on the range space of the variables. In NNs, the network is typically fully connected with very simple neuron-like digital processing elements. In the BZM, only second-order connections are allowed and it is hoped that adequate representation emerges from a reasonable number of hidden units. Fixed models are not specified; the network learns from a set of examples. In the case of unsupervised learning, only the learning rules are specified in advance. These learning rules are heuristically specified or are derived from further principles in some cases.

However, there are several common themes. Specifically in BZM, the distributions (incoming and generated) are Gibbsian. There also exists a close analogy between the hierarchical attribute process **L** in the MRF setup and the hidden units. This should come as no surprise since the inspiration for the hidden units did come from the area of Hidden MRFs (HMRF) [17]. The clear difference here is that NNs implicitly build distributed, internal representations using hidden units and interconnections, whereas MRFs build explicit representations using attribute variables and model parameters. The improvement of the model in NNs arises through learning where the long term memory (in the interconnections) is modified taking into account all past and present exchanges between the network and the environment. This is analogous to parameter estimation in MRFs if it is not performed off-line but adaptively as more information is present to the system. The EM algorithm is an archetype of adaptive parameter estimation.

MRFs and NNs answer the question of what is to be computed in different ways. While MRFs exploit the structure of the local characteristics, NNs mostly use fully connected networks with second-order connections. Both paradigms build internal representations on hidden (unobservable) units, MRFs explicitly and NNs implicitly. Adaptation to new data is provided for in both models with NNs incorporating learning and MRFs performing adaptive parameter estimation.

In MRFs, prior distributions typically constrain the number of all possible distributions and are sometimes counter-intuitive. A proper choice of attribute variables and prior distributions is crucial to the success of MRFs. Important areas where NNs can be useful in MRFs are adaptive parameter and state estimation. Similarly, NNs can benefit from the results of MRF based modeling. The choice of the number of hidden units can be made more assuredly with the knowledge of explicit modeling using MRFs. Another important topic is higher-order connections. MRFs utilize local, higher-order connections to obtain better models. Although, in principle, hidden characteristics can be approximated using hidden units (intergrating out the hidden units generates higher-order distributions), it is not clear how well hidden units and second-order connections perform with respect to this point in vision tasks. More work needs to be done with higher-order, partially connected NNs [5, 18].

We now move to the algorithmic level of description.

2.3 RELATIONSHIP AT THE ALGORITHMIC LEVEL

The main problems in MRFs can be divided into two sub-problems; state estimation and parameter estimation. Once the criteria have been imposed on the system, state estimation is usually cast into a MAP or MMSE problem or their variants. The impetus in MRFs dramatically increased with the introduction of SA [3] and it has not waned even after the computational limitations of SA have been pointed out. There are several alternatives to SA especially for MAP estimation. These range from general purpose techniques like ICM (not confined to MAP) to deterministic annealing techniques or continuation methods. Interestingly enough, Hopfield networks and MFT (the two are equivalent for MAP estimation) have been quite successful in early vision problems like image estimation and surface reconstruction. It is important to realize that MAP estimation performed in isolation is not the goal. We also need to perform parameter estimation. Due to the intractability of the partition function, EM like techniques have become popular.

Deterministic networks for MAP estimation have borrowed heavily from neural networks, MFT in particular. MFT as applied to MAP estimation uses approximations to the partition function and ends up minimizing a cost function which in some ways is a smoother version of the original cost function. As a control parameter is varied, the original cost function is better approximated. Also, new methods of solving the *winner-take-all* (WTA) problem (which also suffers from local minima) have emerged [19]. The impetus in NNs began with a novel approach to the TSP problem. There is tremendous potential in further applications of NN related ideas for MAP and MMSE estimation of MRFs. In Section 3, we present different NN related approaches for solving well known problems in early vision.

The parameter estimation problem can be successfully tackled using PL techniques and the EM algorithm. The consistency of these estimators has been proven for very general Gibbs distributions [20]. BZM learning algorithms are very closely related to ML techniques. This is because the BZM minimizes the Kullback information gain [14] which is related to ML. The difference is that ML specifies the parameters by maximizing the distribution of the data whereas the Kullback information gain minimizes the distance between the distributions of the fixed and free phases with respect to the weights.

3 APPLICATIONS

In this section, we concentrate on two applications which illustrate some of the comments made in Section 2.3.

The first application is in image estimation and segmentation. The problem can be succintly stated as follows. We wish to obtain an estimate of a noisy image which at the same time yields a convenient representation in terms of piecewise, homogeneous regions. Adopting the MRF viewpoint, the degradation is assumed to be corruption by additive noise and the prior belief is that the image is composed of piecewise, homogeneous regions. Then the goal of the computation is conveniently expressed as a MAP estimation problem. The noisy image is the **D** process, the prior is the **F** process and the boundaries are the attribute process **L**. The MAP estimation problem is NP-complete and we suggest continuation methods which find good, sub-optimal solutions. We show that our algorithm is equivalent to those of other reseachers when several constraints on the boundary process $\mathbf{L}$ are removed (the uninteresting case). Specifically, our continuation method has close ties to SA, MFT and other non NN related methods.

The second application is in texture segmentation. The problem of interest is to locate the boundaries of the textures present in an image. As in the case of image estimation, we set up the problem as a MAP estimation of the texture labels $\mathbf{L}$ when we know the texture intensities $\mathbf{Y}$. This problem is also NP-complete and we present several algorithms, stochastic and deterministic which solve the problem. Of particular interest is a learning algorithm adapted from the stochastic approximation literature which refines the optimization algorithm by experience. Comparisons with SA and others are provided which facilitate choice of any algorithm in a particular situation.

3.1 IMAGE ESTIMATION AND SEGMENTATION

3.1.1 A GENERALIZED FRAMEWORK FOR IMAGE ESTIMATION AND SEGMENTATION

When Gibbs distributions are used for prior and degradation models, the posterior distribution is still Gibbs. The attractive feature of a Gibbs distribution is in the convenience of its specification; it can be completely specified by an energy function. The energy function is defined over the intensity ($\mathbf{f}$), the vertical ($\mathbf{v}$) and horizontal ($\mathbf{h}$) line processes.

$$\begin{aligned}\mathcal{H}(\mathbf{f},\mathbf{v},\mathbf{h}) &= \sum_{\{i,j\}}(f(i,j)-d(i,j))^2 + \sum_{\{i,j\}}(\lambda^2 f_x^2(i,j)(1-v(i,j)) + \alpha\, v(i,j)) \\ &+ \sum_{\{i,j\}}(\lambda^2 f_y^2(i,j)(1-h(i,j)) + \alpha\, h(i,j)) + \mathcal{H}_c(\mathbf{v},\mathbf{h}) \qquad (2)\end{aligned}$$

where $\mathbf{d}$ is the observed data, $v(i,j), h(i,j) \in \{0,1\}$, $f_x(i,j) \stackrel{\text{def}}{=} f(i+1,j)-f(i,j)$ and $f_y(i,j) \stackrel{\text{def}}{=} f(i,j+1)-f(i,j)$.

When the $\mathcal{H}_c(.)$ term (corresponding to prior knowledge of image contours) is absent, this energy function reduces to the popular *weak membrane* [21]. The term weak membrane arises from the physical nature of this energy function. If discontinuities are absent, the reconstruction would be like a membrane which is continuous everywhere. Our objective is to construct a continuation method which finds good suboptimal MAP estimates. The energy function parameters are λ and α.

The weak membrane energy function can be written in a more compact form.

$$\mathcal{H}(\mathbf{f}) = \sum_{\{i,j\}}(f(i,j)-d(i,j))^2 + \sum_{\{i,j\}} g^*(f_x(i,j)) + \sum_{\{i,j\}} g^*(f_y(i,j)) \qquad (3)$$

where the new function $g^*(f_p)$ arises from the elimination of the line processes and f_p is the generic symbol used for the intensity gradient. The g^* function is written as;

$$g^*(f_p) = \begin{cases} \lambda^2 f_p^2 & \lambda^2 f_p^2 < \alpha \\ \alpha & \lambda^2 f_p^2 \geq \alpha \end{cases} \qquad (4)$$

We also define $g_z^*(f_p, z) \stackrel{def}{=} \lambda^2 f_p^2(1-z) + \alpha\ z,\ z\ \in\ \{0,1\}$ where z is the generic symbol for the line process.

The energy function (3) as it stands is non-convex due to the nature of the g^* function. Several researchers [9, 10, 21] have proposed a variety of continuation methods or convex formulations to deal with this problem. A continuation method essentially tracks minima through the variation of a control parameter; the original energy function is increasingly closely approximated during this variation. All of these approaches can be conceptually synthesized by replacing the g^* function by either a solitary g function or by a sequence of $g^{(k)}$ functions; the integer k is the index of the sequence. Henceforth, we will refer to this sequence simply by the g function since our generalized framework is valid for all members of a given sequence.

We define a new sequence of $g_u^{(k)}$ functions which are related to the old sequence $g^{(k)}$ as follows.

$$g_u(f_p, u) = g(u) + \frac{g'(u)}{2u}(f_p^2 - u^2) \tag{5}$$

where u is a new process.

The new sequence of g_u functions is derived from the $g^{(k)}$ sequence using the basic idea that elimination of the u processes in g_u should yield g.

Examining $g_u(f_p, u)$ as a function of u^2, we get

$$g_s(t, s) = g_2(s) + g_2'(s)(t - s) \tag{6}$$

where $t = f_p^2$, $s = u^2$ and $g_2(u^2) = g(u)$.

Obviously, $g_s(t, s)$ is just a Taylor series expansion in t around s truncated at the first term. The interesting point is that both t and s emerge as full-fledged processes on which relaxation is performed. Considering $g_s(t, s)$ as a function of s, we can find the minimum with respect to s keeping t constant.

$$\frac{\partial g_s(t,s)}{\partial s} = g_2''(s)(t-s) = 0 \tag{7}$$

One of the solutions is $s = t$ which can be reformulated as $u^2 = f_p^2$. When this is substituted back into (6), we obtain

$$g_s(t, t) = g_2(t) = g(f_p) \tag{8}$$

The sufficient condition for the minimum is

$$g_2''(t)\ <\ 0 \tag{9}$$

Interpreting our technique as a *first* order Taylor series expansion giving rise to a new process s has important consequences. The unobservable process s emerges from the Taylor series expansion. The condition for a minimum to exist at $s = t$ requires $g_2(t)$ to be concave. When s is eliminated dynamically by resubstitution, we obtain the original function $g_2(t)$. Constraints can be added in the s space. The consequence of this result is that the sequence $g_u^{(k)}$ and $g^{(k)}$ are equivalent when no further constraints are added in the space of u. Further constraints aid in the organization of the u process which as we show in Section 3.1.2 has the properties of a boundary process.

3.1.2 RECOVERY OF THE LINE PROCESS

The relationship $t = s$ or $u = f_p$ immediately confers the notion of a gradient upon u. We refer to u as the gradient (GRAD) process. The line process can be recovered from the GRAD process keeping in mind that equal, positive and negative values of u must map to the same points in z. We find it convenient to first transform the energy function using $u_{GM} = |u|$, the Gradient-Magnitude (GMAG) process. The criteria for a minimum w.r.t u_{GM} become, $u_{GM} = |f_p|$ and $(g^{'}(u_{GM}) - u_{GM}g^{''}(u_{GM})) \; > \; 0$. The sufficient condition arises from rewriting the concavity condition in terms of u_{GM}. A transformation from the GMAG process to the line process $z = z(u_{GM})$ can be obtained. The transformation should not cause the formation of spurious minima. Every minimum of u_{GM} should be a minimum of z and vice versa. It is easy to show that this condition is satisfied when $z = z(u_{GM})$ is monotonic and $\frac{dz}{du_{GM}}$ is finite.

We suggest two transformations; $z_1 = 1 - l_1(u_{GM})$, and $z_2 = l_2(u_{GM})$. The monotonicity condition requires

$$\frac{dz_1}{du_{GM}} = \frac{(g^{'}(u_{GM}) - u_{GM}g^{''}(u_{GM}))}{2\lambda^2 u_{GM}^2} \; > \; 0 \tag{10}$$

for the first transformation, and

$$\frac{dz_2}{du_{GM}} = \frac{1}{\alpha}(g^{'}(u_{GM}) - u_{GM}g^{''}(u_{GM})) \; > \; 0 \tag{11}$$

for the second. Note that the relation $(g^{'}(u_{GM}) - u_{GM}g^{''}(u_{GM}))$ also shows up in the sufficient condition for a minimum. Combining the conditions for monotonicity and for a minimum to exist, we get

$$g^{'}(u_{GM}) - u_{GM}g^{''}(u_{GM}) \; > \; 0 \tag{12}$$

Each of the transformations suggested have corresponding energy functions that are now defined over the intensity and the *analog line* processes.

3.1.3 A CONTINUATION METHOD USING THE GMAG PROCESS

The continuation method used is based on Blake and Zisserman's Graduated Non-Convexity (GNC) algorithm [21]. The sequence of g functions used here is derived from the GNC criterion. The first function in the sequence is constructed to make the energy function convex with respect to the intensities. Successive g functions push the energy function closer to the weak membrane. The functions are created out of piecewise polynomials and are gradually modified by a control parameter c until the function g^* is reached. The g function sequence is shown below.

$$g^{(k)}(u) = \begin{cases} \lambda^2 u^2 & 0 \; \leq \; |u| \; \leq \; q \\ \alpha - \frac{c}{2}(r - |u|)^2 & q \; < \; |u| \; < \; r \\ \alpha & |u| \; > \; r \end{cases} \tag{13}$$

where $c = c_* 2^k$, $k = 0, 1, ..$ and $r^2 = \alpha(\frac{2}{c} + \frac{1}{\lambda^2})$, $q = \frac{\alpha}{\lambda^2 r}$. c_* is the initial value of the control parameter. The other parameters q and r arise as a consequence of creating a convex energy function (corresponding to $c = c_*$).

Proceeding as outlined above we can obtain the two functions $l_1(u_{GM})$ and $l_2(u_{GM})$. The relations can be simplified by modifying the definition of the GMAG process.

The GMAG process u_{GM} is now defined over the interval $[q, r]$. Note that the interval does not stay fixed but keeps shrinking as c is increased. Now the two functions $l_1(u_{GM})$ and $l_2(u_{GM})$ can be written as

$$l_1(u_{GM}) = \frac{c}{2\lambda^2}\frac{(r - u_{GM})}{u_{GM}},\ l_2(u_{GM}) = \frac{c}{2\lambda^2}\frac{(u_{GM} - q)}{q},\ q \ \leq \ u_{GM} \ \leq \ r \tag{14}$$

We check if relation (12) is satisfied.

$$g^{'}(u_{GM}) - u_{GM}g^{''}(u_{GM}) = 2cr \ > \ 0 \tag{15}$$

We drop the subscript on u_{GM} since the GRAD process plays no role in subsequent discussions. The line process can be obtained from either of the two transformations.

3.1.4 THE GENERALIZED GRADUATED NON-CONVEXITY ALGORITHM

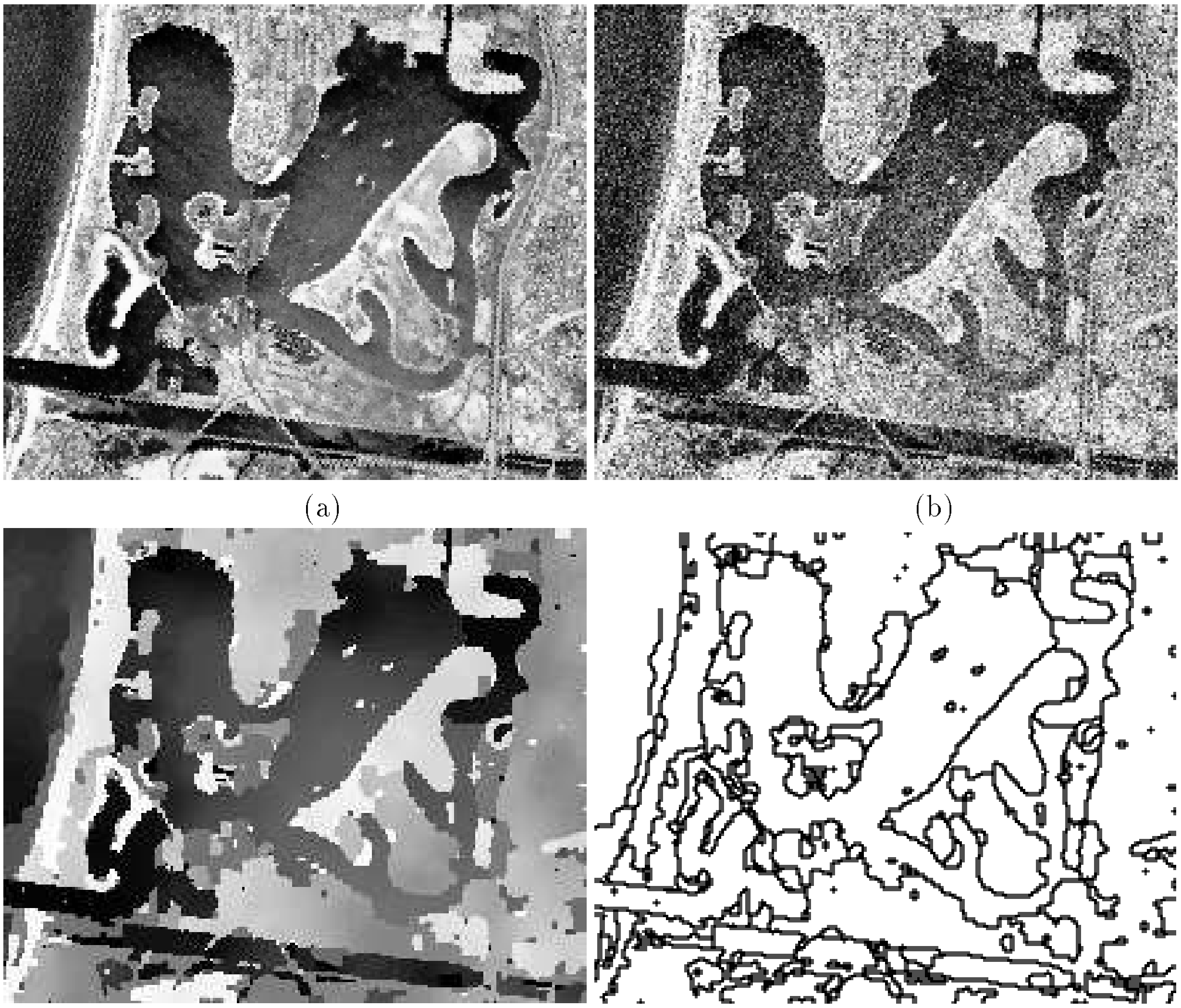
We now suggest introducing interactions on the GMAG process. All line interactions can be transfered to the GMAG domain.

We choose two basic kinds of interaction terms. This has been inspired by the work of Geiger and Girosi [9] and by Geiger and Yuille [19]. Consider the following modification of (2).

$$\begin{aligned}\mathcal{H}_{G^2NC} = & \sum_{\{i,j\}}\Big((f(i,j) - d(i,j))^2 + (g_u(f_x(i,j), u_v(i,j)) + g_u(f_y(i,j), u_h(i,j))\Big) \\ & -\frac{c}{2}\epsilon_1\sum_{\{i,j\}}(u_v(i,j) - q)(u_v(i,j+1) - q) + \frac{c}{2}\epsilon_2\sum_{\{i,j\}}(u_v(i,j) - q)(u_v(i+1,j) - q) \\ & -\frac{c}{2}\epsilon_1\sum_{\{i,j\}}(u_h(i,j) - q)(u_h(i+1,j) - q) + \frac{c}{2}\epsilon_2\sum_{\{i,j\}}(u_h(i,j) - q)(u_h(i,j+1) - q)\end{aligned} \tag{16}$$

where all the indices are analogous to the earlier line process case. Physically, we are trying to decrease the penalty on a vertical (horizontal) line if its vertical (horizontal) neighbors are "on" and increase the penalty on a vertical (horizontal) line if its adjacent vertical (horizontal) neighbors are "on". This corresponds to encouraging hysteresis which leads to the formation of unbroken contours and non-maximum suppression which prevents the formation of multiple edges and adjacent parallel lines. The penalty is controlled by the parameters ϵ_1 and ϵ_2.

We now proceed with the algorithm itself. The improved line process in the GNC formulation prompted us to call this algorithm the Generalized GNC or G^2NC algorithm. We first solve for u keeping the intensities fixed. Now, u becomes a function of the neighbors as well as the intensity gradient. We choose to run ICM on the u process. This requires a black and white updating scheme since ICM is guaranteed to converge only when the updating is asynchronous.



(a) (b)

(c) (d)

Figure 1: (a) Original image of Mission Bay (San Diego), (b) noisy image, (c) restored image using the G^2NC algorithm, and (d) line process image using the G^2NC algorithm

In order to run ICM on the GMAG process, we have to solve for u when interactions are present. The criterion for a minimum for the vertical GMAG process is:

$$\frac{\partial \mathcal{H}_{G^2NC}}{\partial u_v(i,j)} = 0 \tag{17}$$

From this we get

$$u_v(i,j) = \frac{\mid f_x(i,j) \mid}{\sqrt{1 - \frac{2\left(\epsilon_1 \overline{(u_v^v(i,j)-q)} - \epsilon_2 \overline{(u_v^h(i,j)-q)}\right)}{r}}} \tag{18}$$

and $\overline{(u_v^v(i,j) - q)} = \frac{(u_v(i,j+1)-q)+(u_v(i,j-1)-q)}{2}$ and $\overline{(u_v^h(i,j) - q)} = \frac{(u_v(i+1,j)-q)+(u_v(i-1,j)-q)}{2}$ The formula for the horizontal GMAG process can be similarly derived. This solution for u is not guaranteed to be within $[q, r]$. If the solution lies outside, the end point energies are again compared and the solution with the lower energy is chosen. The solution depends heavily on ϵ_1 and ϵ_2. We have used the generic symbol ϵ for both ϵ_1 and ϵ_2 when they assume the same value.

We have chosen to run the Conjugate Gradient (CG) algorithm with an optimal step on the intensities. This is because the CG algorithm converges much faster than steepest descent (SD). The course we pursue is to apply ICM on the GMAG processes until they converge. We alternate between ICM on the GMAG processes and CG on the intensities. We have noticed that ICM takes very few (one to five) iterations to converge (for our case studies). The algorithm is as follows.

1. Set $c = c_*$ (usually $c_* = 0.25$)
2. Run the CG algorithm on the intensities.
3. Update GMAG processes using ICM until convergence
4. Return to Step 2 till convergence
5. Increase c, usually $c = 2^k c_*, k = 0, 1, ..$
6. Return to Step 2 until convergence of the GMAG processes

More details on our approach can be found in [22].

We show results for our energy function and compare it to the GNC algorithm. Figures 1(a) and 1(b) contain an aerial view of Mission Bay, San Diego and the corresponding noisy image. Figures 1(c) and 1(d) show the results of applying our algorithm with the interaction terms. The parameter α was set to 676, the parameter λ to 8 and ϵ to 0.5.

The technique we have used to incorporate interaction terms is a general one and not restricted to the GNC algorithm.

3.2 TEXTURE SEGMENTATION

3.2.1 IMAGE MODEL

We use a fourth order Gauss-Markov Random Field (GMRF) to model the conditional probability density of the image intensity array given its texture labels. The texture labels are assumed to obey a first or second order discrete Markov model with a single parameter β, which measures the amount of clustering between adjacent pixels.

Let Ω denote the set of grid points in the $M \times M$ lattice, i.e., $\Omega = \{(i,j)\ ,\ 1 \leq i,j \leq M\}$. Following Geman and Graffigne [20] we construct a composite model which accounts for texture labels and gray levels. Let $\{L_s\ ,\ s \in \Omega\}$ and $\{Y_s\ ,\ s \in \Omega\ \}$ denote the labels and gray level arrays respectively. Let N_s denote the symmetric fourth order neighborhood of a site s. Then assuming that all the neighbors of s also have the same label as that of s, we can write the following expression for the conditional density of the intensity at the pixel site s:

$$\mathcal{P}(Y_s = y_s \mid Y_r = y_r, r \in N_s, L_s = l) = \frac{e^{-U(Y_s=y_s\ \mid\ Y_r=y_r, r\in N_s, L_s=l)}}{Z(l|y_r, r \in N_s)} \tag{19}$$

where $Z(l|y_r, r \in N_s)$ is the partition function of the conditional Gibbs distribution and

$$U(Y_s = y_s \mid Y_r = y_r, r \in N_s, L_s = l) = \frac{1}{2\sigma_l^2}(y_s^2 - 2\sum_{r\in N_s} \Theta^l_{s,r} y_s y_r) \tag{20}$$

In (20), σ_l and Θ^l are the GMRF model parameters of the l-th texture class. The model parameters satisfy $\Theta^l_{r,s} = \Theta^l_{r-s} = \Theta^l_{s-r} = \Theta^l_\tau$.

We view the image intensity array as composed of a set of overlapping $k \times k$ windows W_s, centered at each pixel $s \in \Omega$. In each of these windows we assume that the texture label L_s is homogeneous (all the pixels in the window belong to the same texture) and compute the joint distribution of the intensity in the window conditioned on L_s. The corresponding Gibbs energy is used in the relaxation process for segmentation. Let $\mathbf{Y}^*_s$ denote the 2-D vector representing the intensity array in the window W_s. Using the Gibbs formulation and assuming a free boundary model, the joint probability density in the window W_s can be written as

$$\mathcal{P}(\mathbf{Y}^*_s = \mathbf{y}^*_s | L_s = l) = \frac{e^{-U_1(\mathbf{y}^*_s|L_s=l)}}{Z_1(l)}$$

where $Z_1(l)$ is the partition function and

$$U_1(\mathbf{Y}^*_s|L_s = l) = \frac{1}{2\ \sigma_l^2}\sum_{r\in W_s}\left\{ y_r^2 - \sum_{\tau\in N^*|r+\tau\in W_s} \Theta^l_\tau y_r(y_{r+\tau} + y_{r-\tau})\right\} \tag{21}$$

N^* is the set of shift vectors corresponding to a fourth order neighborhood system:

$$\begin{aligned} N^* &= \{\tau_1, \tau_2, \tau_3, \cdots, \tau_{10}\} \\ &= \{(0,1),(1,0),(1,1),(-1,1),(0,2),(2,0),(1,2),(2,1),(-1,2),(-2,1)\} \end{aligned}$$

The label field is modeled as a first or second order discrete MRF. If $\hat{N}_s$ denotes the appropriate neighborhood for the label field, then we can write the distribution function for the texture label at site s conditioned on the labels of the neighboring sites as:

$$\mathcal{P}(L_s | L_r\ ,\ r \in \hat{N}_s) = \frac{e^{-U_2(L_s\ |\ L_r)}}{Z_2}$$

where Z_2 is a normalizing constant and

$$U_2(L_s \mid L_r,\ r \in \hat{N}_s) = -\beta \sum_{r \in \hat{N}_s} \delta(L_s - L_r)\ ,\ \beta > 0 \tag{22}$$

In (22), β determines the degree of clustering, and $\delta(i-j)$ is the Kronecker delta. Using the Bayes rule, we can write

$$\mathcal{P}(L_s \mid \mathbf{Y}^*_s,\ L_r,\ r \in \hat{N}_s) = \frac{\mathcal{P}(\mathbf{Y}^*_s \mid L_s)\ \mathcal{P}(L_s \mid L_r,\ r \in \hat{N}_s)}{\mathcal{P}(\mathbf{Y}^*_s)} \tag{23}$$

Since $\mathbf{Y}^*_s$ is known, the denominator in (23) is just a constant. The numerator is a product of two exponential functions and can be expressed as,

$$\mathcal{P}(L_s \mid \mathbf{Y}^*_s,\ L_r,\ r \in \hat{N}_s) = \frac{1}{Z_p}\ e^{-U_p(L_s\ |\ \mathbf{Y}^*_s,\ L_r,\ r \in \hat{N}_s)} \tag{24}$$

where Z_p is the partition function and $U_p(.)$ is the posterior energy corresponding to (23). From (21) and (22) we write

$$U_p(L_s \mid \mathbf{Y}^*_s,\ L_r,\ r \in \hat{N}_s) = w(L_s) + U_1(\mathbf{Y}^*_s \mid L_s) + U_2(L_s \mid L_r,\ r \in \hat{N}_s) \tag{25}$$

Note that the second term in (25) relates the observed pixel intensities to the texture labels and the last term specifies the label distribution. The bias term $w(L_s) = \log Z_1(L_s)$ is dependent on the texture class and it can be explicitly evaluated for the GMRF model considered here using the toroidal assumption (the computations become very cumbersome if toroidal assumptions are not made). An alternate approach is to estimate the bias from the histogram of the data as suggested by Geman and Graffigne [20]. Finally, the posterior distribution of the texture labels for the entire image given the intensity array is

$$\mathcal{P}(\mathbf{L} \mid \mathbf{Y}^*) = \frac{\mathcal{P}(\mathbf{Y}^* \mid \mathbf{L})\ \mathcal{P}(\mathbf{L})}{\mathcal{P}(\mathbf{Y}^*)} \tag{26}$$

Maximizing (26) gives the optimal Bayesian estimate. We note that a stochastic relaxation algorithm requires only the computation of (24) to obtain the optimal solution. The deterministic relaxation algorithm given in the next section also uses these values and performs descent on a related energy function.

3.2.2 A NEURAL NETWORK FOR TEXTURE CLASSIFICATION

We describe the network architecture used for segmentation and the implementation of deterministic relaxation algorithms. Let $U_1(i,j,l) = U_1(\mathbf{Y}_s^*, L_s = l) + w(l)$ where $s = (i,j)$ denotes a pixel site and $U_1(\ .\ )$ and $w(l)$ are as defined in (25). The network consists of K layers, each layer arranged as an $M \times M$ array, where K is the number of texture classes in the image and M is the dimension of the image. The elements (neurons) in the network are assumed to be binary and are indexed by (i,j,l) where $(i,j) = s$ refers to their position in the image and l refers to the layer. The (i,j,l)-th neuron is said to be ON if its output V_{ijl} is 1, indicating that the corresponding site $s = (i,j)$ in the image has the texture label l.

A solution for the MAP estimate can be obtained by minimizing (26). We approximate the posterior energy by

$$U(\mathbf{L}|\mathbf{Y}^*) = \sum_s \{U(\mathbf{Y}_s^*|L_s) + w_{L_s} + U_2(L_s)\} \tag{27}$$

and the corresponding Gibbs energy to be minimized can be written as

$$E = \frac{1}{2}\sum_{i=1}^{M}\sum_{j=1}^{M}\sum_{l=1}^{K} U_1(i,j,l)V_{ijl} - \frac{\beta}{2}\sum_{l=1}^{K}\sum_{i=1}^{M}\sum_{j=1}^{M}\sum_{(i',j')\in\hat{N}_{ij}} V_{i'j'l}V_{ijl} \tag{28}$$

where $\hat{N}_{ij}$ is the neighborhood of site (i,j). In (28), it is implicitly assumed that each pixel site has a unique label, i.e. only one neuron is active in each column of the network. For the deterministic relaxation algorithm, a simple method is to use a WTA circuit for each column so that the neuron receiving the maximum input is turned on and the others are turned off.

The network model is a version of the ICM algorithm of Besag [8]. We observe that in general any algorithm based on MRF models can be easily mapped on to neural networks with local interconnections.

3.2.3 STOCHASTIC ALGORITHMS FOR TEXTURE SEGMENTATION

The MAP rule searches for the configuration L that maximizes the posterior probability distribution. This is equivalent to maximizing $\mathcal{P}(\mathbf{Y}^* \mid \mathbf{L})\ \mathcal{P}(\mathbf{L})$ as $\mathcal{P}(\mathbf{Y}^*)$ is independent of the labels and $\mathbf{Y}^*$ is known. The right hand side of (26) is a Gibbs distribution. To maximize (26) we use simulated annealing [3]. It samples from the conditional distribution

$$\frac{e^{-\frac{1}{T_k}U_p(L_s \ \mid\ \mathbf{Y}_s^*,\ L_r, r\in\hat{N}_s)}}{Z_{T_k}}$$

in order to maximize

$$\frac{e^{-U_p(\mathbf{L}\ \mid\ \mathbf{Y}^*)}}{Z}$$

T_k being the time varying parameter, referred to as the temperature. The process is guaranteed to converge to a uniform distribution over the label configuration that corresponds to the MAP solution.

The choice of the objective function for optimal segmentation can significantly affect its result. In many implementations, the most reasonable objective function is the one that minimizes the expected percentage misclassification per pixel. The solution to the above objective function can be obtained by maximizing the marginal posterior distribution (MPM) of L_s given the observation $\mathbf{Y}^*$, for each pixel s.

$$\mathcal{P}\{L_s = l_s \mid \mathbf{Y}^* = \mathbf{y}^*\} \;\propto\; \sum_{\mathbf{l}|L_s=l_s} \mathcal{P}(\mathbf{Y}^* = \mathbf{y}^* \mid \mathbf{L} = \mathbf{l})\; \mathcal{P}(\mathbf{L} = \mathbf{l})$$

The summation above extends over all possible label configurations keeping the label at site s constant. To find the optimal solution we use the stochastic algorithm suggested in [23]. The algorithm samples out of the posterior distribution of the texture labels given the intensity at $T = 1$. The Markov chain associated with the sampling algorithm converges with probability one to the posterior distribution.

3.2.4 STOCHASTIC LEARNING AND NEURAL NETWORKS

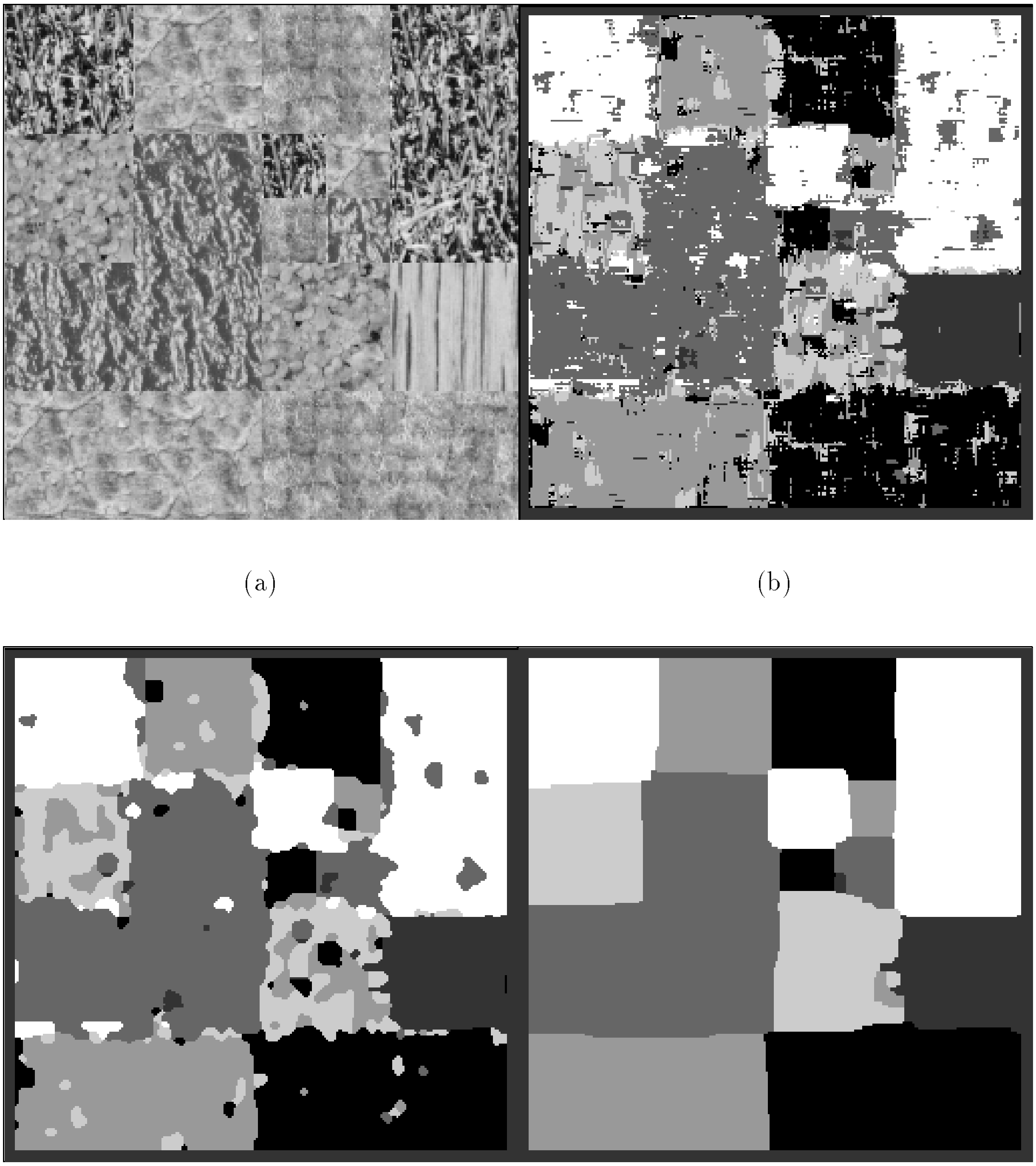
The texture classification discussed in the previous sections can be treated as a relaxation labelling problem and stochastic automata can be used to learn the texture labels. A stochastic automaton is a decision maker operating in a random environment [24] and is assigned to each of the pixel sites in the image. The actions of the automata correspond to selecting a label for the pixel site to which it is assigned. Thus for a K class problem each automaton has K actions and a probability distribution over this action set. Initially the labels are assigned randomly with equal probability. Since the number of automata involved is very large, it is not practical to update the action probability vector at each iteration. Instead we combine the iterations of the neural network described in the previous section with the stochastic learning algorithm. After each convergence of the deterministic relaxation algorithm, the action probabilities ($p_{s,l}$) are updated as follows.

$$\begin{aligned} p_{s,l_s}(t+1) &= p_{s,l_s}(t) + a\ \lambda(t)\ [1 - p_{s,l_s}(t)] \\ p_{s,l}(t+1) &= p_{s,l}(t)[1 - a\ \lambda(t)],\ \forall l \neq l_s \text{ and } \forall\ s \end{aligned} \tag{29}$$

where l_s denotes the label of site s at equilibrium and $p_{s,l}(t)$ is the probability of choosing label l for site s at time t. The response $\lambda(t)$ is derived as follows: Suppose the present label configuration resulted in a lower energy state compared to the previous one, then it results in a $\lambda(t) = \lambda_1$ and if the energy increases we have $\lambda(t) = \lambda_2$ with $\lambda_1 > \lambda_2$. In our simulations we used $\lambda_1 = 1$ and $\lambda_2 = 0.25$. A new configuration for the relaxation network is then generated from the updated action probabilities.

Thus the system consists of a relaxation network and a learning network. The relaxation network is similar to the one in Section 3.2.2, except that the initial state is decided by the learning network. This results in an iterative hill climbing type algorithm which combines the fast convergence of deterministic relaxation with the sustained exploration of the stochastic algorithm. The stochastic part prevents the algorithm from getting trapped in local minima and at the same time "learns" from the search by updating the state probabilities. Each cycle now has two phases: the first consists of the deterministic relaxation network converging to a solution; the second consists of the learning network updating its state. For further details, the reader is referred to [25].

3.2.5 EXPERIMENTAL RESULTS



(a) (b)

(c) (d)

Figure 2: (a) Original image consisting of six textures, (b) maximum likelihood solution, (c) deterministic relaxation with (b) as initial condition, (d) MAP estimate using simulated annealing

The segmentation results using the above algorithms are given on one example. The least-square estimates (LSE) of the parameters σ_l and Θ_l corresponding to the fourth

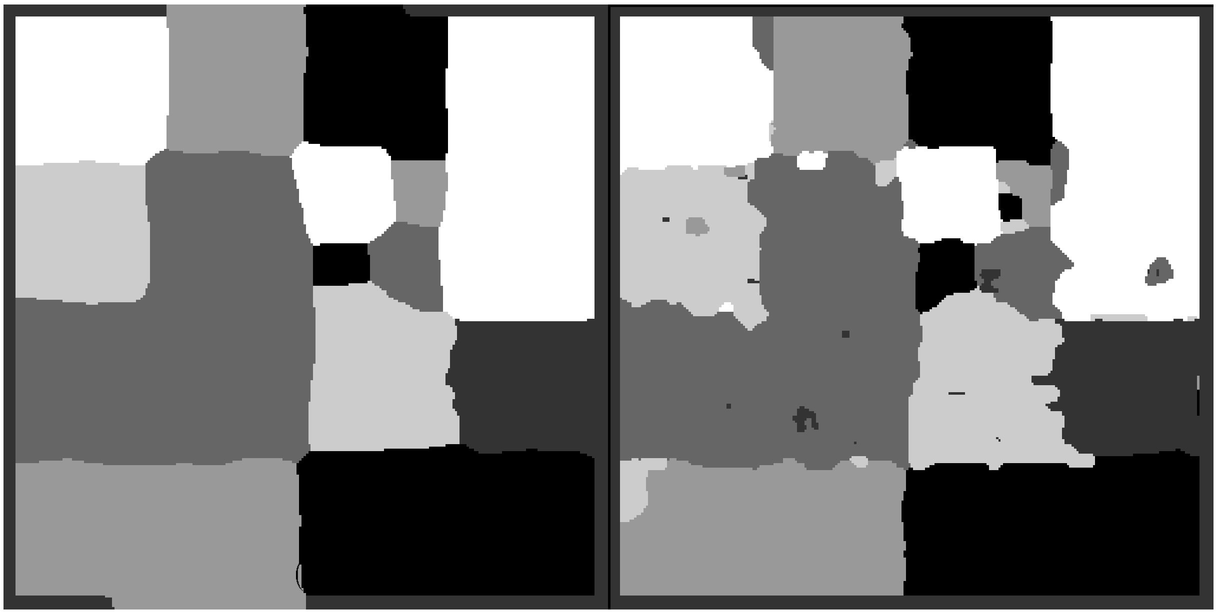


Figure 2: (contd.) (e) MPM solution and (f) network with stochastic learning

order GMRF for each texture class were pre-computed from 64×64 images of the textures.

Figure 2(a) shows a 256×256 image having six textures: leather, grass, wool, wood, pig skin and sand. The maximum likelihood solution is shown in Figure 2(b) and Figure 2(c) is the solution obtained by the deterministic relaxation network with the result in Figure 2(b) as the initial condition. The MAP solution using simulated annealing is shown in Figure 2(d). Figure 2(e) shows the MPM result. As indicated in Table 1, simulated annealing has the lowest percentage error in classification. Introducing learning into deterministic relaxation considerably improves the performance (Figure 2(f)).

Table 1
Percentage misclassification for the six class problem

Algorithm	Percentage Error
Maximum Likelihood Estimate	22.17
Neural network (MLE as initial state)	16.25
Simulated annealing (MAP)	6.72
MPM algorithm	7.05
Neural network with learning	8.7

4 CONCLUSIONS

We have presented two applications which highlight the close links between MRFs and NNs. There are several issues which have not been addressed and are beyond the scope of this chapter. We briefly mention a few of these issues. The use of MRFs in integrating visual modules [26] has been an important area of investigation in recent years. Density function estimation [27] is an important area where NNs and MRFs can be related. Finally, the important problem of binding syntactical structures is addressed by the Dynamic Link Architecture [28] which uses a fast synaptic plasticity term in addition to the more conventional slow weight modification term. This is an alternative to using either higher-order connections or additional hidden units to model hidden structural relationships in scenes.

References

[1] D. Marr, *"Vision"*, W. H. Freeman and Co., San Francisco, CA, 1982.

[2] J. J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons", *Proc. Natl. Acad. Sci., U.S.A.*, 81, pp. 3088–3092, 1984.

[3] S. Geman and D. Geman, "Stochastic relaxation, Gibbs Distributions and the Bayesian restoration of Images", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, pp. 721–741, November 1984.

[4] D. E. Rumelhart, J. L. McClelland, and the PDP Research Group, editors, *"Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1: Foundations"*, MIT Press, Cambridge, MA, 1986.

[5] T. Maxwell, C. L. Giles, and Y. C. Lee, "Nonlinear Dynamics of Artificial Neural Systems", In *Proceedings of the Neural Networks for Computing Conference*, Snowbird, Utah, April 1986.

[6] G. A. Carpenter and S. Grossberg, "A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine", *Computer Vision, Graphics and Image Processing*, 37, pp. 54–115, 1987.

[7] S. Lloyd and H. R. Pagels, "Complexity as Thermodynamic Depth", *Annals of Physics*, vol. 188, pp. 186–213, 1988.

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

[9] D. Geiger and F. Girosi, "Parallel and deterministic algorithms for MRFs: surface reconstruction and integration", Technical Report A. I. Memo, No. 1114, Artificial Intelligence Lab, M.I.T, June 1989.

[10] Y. G. Leclerc, "Constructing Simple Stable Descriptions for Image Partitioning", *International Journal of Computer Vision*, vol. 3, pp. 73–102, 1989.

[11] J. Besag, "Efficiency of Pseudo-likelihood Estimation for Simple Gaussian Fields", *Biometrika*, 64, pp. 616–618, 1977.

[12] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood from incomplete data via the EM algorithm", *Journal of the Royal Statistical Society B*, 39, pp. 1–38, 1977.

[13] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, "A learning algorithm for Boltzmann machines", *Cognitive Science*, 9, pp. 147–169, 1985.

[14] S. Kullback, *"Information Theory and Statistics"*, Dover Publications, New York, NY, 1968.

[15] C. Peterson and J. R. Anderson, "A mean field theory learning algorithm for neural networks", *Complex Systems*, 1, pp. 995–1019, 1987.

[16] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position", *Biological Cybernetics*, 36(4), pp. 193–202, 1980.

[17] L. R. Rabiner and B.-H. Juang, "An Introduction to Hidden Markov Models", *IEEE ASSP Magazine*, vol. 3(1), pp. 4–16, January 1986.

[18] H. Haken, *"Information and Self-Organization: a macroscopic approach to complex systems"*, volume 40 of *Springer Series in Synergetics*, Springer-Verlag, 1989.

[19] D. Geiger and A. Yuille, "A Common Framework for Image Segmentation", In *Proceedings of the International Conf. on Pattern Recognition, ICPR-90*, Atlantic City, NJ, June 1990.

[20] S. Geman and C. Graffigne, "Markov random fields image models and their application to computer vision", In *Proc. of the Int. Congress of Mathematicians 1986*, Ed. A.M. Gleason, American Mathematical Society, Providence, 1987.

[21] A. Blake and A. Zisserman, *"Visual Reconstruction"*, MIT Press, Cambridge, 1987.

[22] A. Rangarajan and R. Chellappa, "Generalized Graduated Non-Convexity Algorithm for Maximum A Posteriori image estimation", In *Proceedings of the International Conf. on Pattern Recognition, ICPR-90*, Atlantic City, NJ, June 1990.

[23] J. L. Marroquin, S. Mitter, and T. Poggio, "Probabilistic solution of ill-posed problems in computational vision", *J. Am. Stat. Assoc.*, 82, pp. 76–89, 1987.

[24] K. S. Narendra and M. A. L. Thathachar, *"Learning Automata"*, Prentice-Hall, New York, NY, 1989.

[25] B. S. Manjunath, T. Simchony, and R. Chellappa, "Stochastic and Deterministic Networks for Texture Segmentation", *IEEE Trans. on Acoust., Speech and Sig. Proc.*, vol. ASSP-38(6), pp. 1039–1049, June 1990.

[26] T. Poggio, E. B. Gamble, and J. J. Little, "Parallel Integration of Vision Modules", *Science*, vol. 242, pp. 436–440, October 1988.

[27] T. Kohonen, *"Self-Organization and Associative Memory"*, volume 8 of *Springer series in Information Sciences*, Springer-Verlag, New York, NY, 3rd edition, 1989.

[28] C. von der Malsburg and E. Bienenstock, "Statistical coding and short-term synaptic plasticity: A scheme for knowledge representation in the brain", In E. Bienenstock, F. Fajelman Soulie, and G. Weisbuch, editors, *Disordered Systems and Biological Organization*, Springer-Verlag, Berlin, 1986.