Adaptive and Neural Methods for Image Segmentation

Joydeep Ghosh

Department of Electrical and Computer Engineering,
The University of Texas, Austin, TX 78712
ghosh@ece.utexas.edu

1 Introduction

The tremendous amount of research in image processing and analysis over the past three decades has been influenced not only by physiological or psychophysical discoveries and psychological observations about perception by living beings, but also by advances in signal processing, computational mathematics, pattern recognition and artificial intelligence. Some researchers in the rejuvenated field of neural networks are also attempting to develop useful models of biological and machine vision. With the human visual system serving as a common source of inspiration, it is not surprising that neural network approaches to image processing/ understanding often have commonalities with more traditional techniques. However, they also bring new elements of non-linear processing with adaptation or learning, some additional insights, and promise breakthroughs through massively parallel and distributed implementations in VLSI [1, 2].

In this chapter, we highlight some *artificial* neural network (ANN) techniques for image segmentation - the process of partitioning an image into regions that are contiguous and relatively homogeneous in image properties. Image segmentation has been studied in great depth ever since satellite images first became available. Segmentation is a key step in any image-based recognition system, and it fundamentally limits the success of all higher level subsystems[3]. The host of sophisticated techniques that have evolved over the past thirty years can be largely grouped into three categories:
(a) *Edge-based* methods, that make use of local and global gradient information to provide boundaries to regions of interest, and thus indirectly segment the image.
(b) *Region-based* methods, that group together local regions with relatively uniform image properties, using methods such as region growing, region splitting and region splitting with merging. Segmentation methods based on *clustering* indirectly use region-based properties, but are often considered as a separate category [4].
(c) *Model-based* methods, that are guided by semantic attributes given to parts of the image, say based on perceived match with (projections of) a set of object models of interest. In this view, segmentation is tightly coupled with image classification or object recognition. This is in contrast with methods in the first two categories, which are primarily based on image attributes rather than on what objects the images may be representing.

Neural network inspired techniques provide additional insights into as well as performance improvements for all three categories. In Section 2, we briefly describe common characteristics of neural network models and introduce some popular models. The next section highlights a prominent edge-based neural technique. Section 4 describes representative region-based methods, specially those that use textural cues. The study of region based methods is continued in Section 5, where we examine optimization based approaches to segmenting textured images. Section 6 describes adaptive clustering techniques for segmentation, while the next section summarizes biological based methods where segmentation is indicated by groups of neurons oscillating in synchrony. Finally, model-based methods are studied in Section 8, where integrated segmentation and recognition techniques are described.

2 Artificial Neural Networks

For our purposes, an Artificial Neural Network (ANN) is a collection of computing cells (artificial neurons) interconnected through weighted links (synapses with varying strengths). The cells perform simple computations using information available locally or from topologically adjacent cells through the weighted links. The knowledge of the system is embodied in the pattern of interconnects and their strengths which vary as the system learns or adapts itself. In this setting we shall see that several ANN models are closely related to established image processing methods such as relaxation labelling, nonlinear filtering and various feature extraction routines.

There is a large variety of ANNs that differ in their topology, cell behavior, weight update mechanisms, amount of supervision or feedback required, etc. Networks with 3 layers of cells: input, "hidden" and output, and with unidirectional or feedforward connections going from one layer to the next, as indicated in Fig. 1, are among the most popular topologies. This class includes the **multi-layered perceptron** (MLP) with a single hidden layer.

In an MLP, given a d-dimensional input $\mathbf{x} = [x_1, x_2, ..., x_d]^T$, the ith network output, y_i is given by

$$y_i(\mathbf{x}) = f(\sum_{j=1}^{M} w_{ij} z_j(\mathbf{x}) + \theta_i), \tag{1}$$

where z_j is the output of the jth hidden unit:

$$z_j = g(\sum_{k=1}^{d} v_{jk} x_k + \theta_j). \tag{2}$$

In the above, v_{jk} denotes the weight of the connection between the jth input and kth hidden unit, and w_{ij} the weight between the jth hidden unit and the ith output. The "activation function" or transfer function $g(\cdot)$ is S-shaped or *sigmoidal*: non-linear, monotonically increasing and bounded. The typical choice is either the logistic map (sometimes called the sigmoid): $g(a) = \frac{1}{1+e^{-a}}$, which is bounded between 0 and 1; or the hyperbolic tangent,

$tanh(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}$, bounded between -1 and 1. The output transfer function $f(\cdot)$ can be linear or sigmoidal as needed.

The MLP realizes a static map between inputs $\mathbf{x}$ and corresponding outputs $\mathbf{y}(\mathbf{x})$. This map depends on the u, w and θ parameters (weights). These weights are trained or adjusted based on training samples $\{ \mathbf{x}(n), \mathbf{t}(n)\}$, where $\mathbf{t}(n)$ is the desired target vector for the nth input vector, $\mathbf{x}(n)$. This adjustment is based on the error $\mathbf{t}(n) - \mathbf{y}(\mathbf{x}(n))$, typically using a stochastic gradient descent on the mean squared value of this error, or via second order methods.

Another popular feedforward network for realizing static maps is the **Radial Basis Function Network** (RBFN). A Radial Basis Function (RBF), ϕ, is one whose output is symmetric around an associated *center* , $\boldsymbol{\mu}_c$. That is, $\phi_c(\mathbf{x}) = \phi(||\mathbf{x} - \boldsymbol{\mu}_c||)$, where $|| \cdot ||$ is a distance norm. For example, selecting the Euclidean norm and letting $\phi(r) = e^{-r^2/\sigma^2}$, one sees that the Gaussian function is an RBF. Note that Gaussian functions are also characterized by a *width* or scale parameter, σ, and this is true for many other popular RBF classes as well.

A set of RBFs can serve as a basis for representing a wide class of functions that are expressible as linear combinations of the chosen RBFs:

$$y_i(\mathbf{x}) = \sum_{j=1}^{M} w_{ij}\phi(||\mathbf{x} - \boldsymbol{\mu}_j||). \tag{3}$$

A Radial Basis Function Network is nothing but an embodiment of (3) as a feedforward network with three layers: the inputs, the hidden layer and the output node(s), i.e. they also have the topology in Fig. 1. Each hidden unit represents a single radial basis function, with associated center position and width. Such hidden units are sometimes referred to as centroids or kernels. Each output unit performs a weighted summation of the hidden units, using the w_js as weights. Powerful *universal* approximation properties of (3) have been demonstrated for various settings.

From (3), one can see that designing an RBFN involves selecting the type of basis functions, ϕ with associated widths, σ, the number of functions, M, the center locations $\boldsymbol{\mu}_j$ and the weights w_j. Typically Gaussians or other *bell-shaped* functions with compact support are used. The choice of M is related to the complexity of the desired map. Given that the number of basis functions and their type have been selected, training an RBFN involves determining the values of three sets of parameters: the centers, the widths and the weights, in order to minimize a suitable cost function. In general, this is a non-convex optimization problem.

One can perform stochastic gradient descent on a mean squared error cost function to iteratively update all three sets of parameters, once per training sample presentation. This may be suitable for non-stationary environments or on-line settings. But for static maps, RBFNs with localized basis functions offer a very attractive alternative, namely that in practice, the estimation of parameters can be decoupled into a *two-stage procedure*:

(i) determine the $\boldsymbol{\mu}_j$s and σ_js, and
(ii) for the centers and widths obtained in step (i), determine the weights to the output units.
Both sub-problems allow for very efficient batch mode solutions. In the first stage only the input values $\{\mathbf{x}(n)\}$ are used for determining the centers $\boldsymbol{\mu}_j$ and the widths σ_j of the basis functions. Thus learning is unsupervised and can even use unlabelled data. Once the basis function parameters are fixed, supervised training (i.e. training using target information) can be employed for determining the second layer weights.

We now briefly describe a third network, one that has a flat topology instead of the layered one of Fig. 1. Each cell is connected to all the cells, including itself, and is also capable of receiving direct input signals. One of the simplest such "fully recurrent" networks is the **binary Hopfield model**.[1] In a network of n cells, the kth cell receives a constant input I_k, and updates its state using:

$$x_k(t+1) = sgn(\sum_{j=1}^{n} w_{kj} x_j(t) + I_k), \tag{4}$$

where "sgn" denotes the signum function, equal to 1 if x is non-negative, and -1 otherwise. The weight matrix should be symmetric, and is fixed, i.e. there is no weight adaptation. Starting from an initial state, given by the values of all cells at $t = 0$, cells update their state asynchronously till a final state is reached where the left and right hand sides of (4) are equal for each of the cells. One can show that such a final state is guaranteed by constructing an energy or cost function:

$$E = -\frac{1}{2} \sum_l \sum_j w_{lj} x_l x_j - \sum_l I_l x_l. \tag{5}$$

By showing that E is bounded from below, and moreover that E is reduced by a finite amount every time a cell changes its value on an update, one concludes that updates have to terminate in finite time.

Clearly this model can serve as an associative memory since it maps the initial state and input to a final state. Also, if one desires to solve an optimization problem where the cost function is quadratic in binary variables, it can be solved on the basis of (5), using one cell per variable, as shown by Hopfield and Tank [6] among others. We shall see such a use in Section 5 where texture segmentation is posed as a suitable optimization problem. Note that there is a generalization of (4) to continuous variables and continuous time. The signum function is now replaced by $tanh(\cdot)$ or the logistic map, so that the cells can have graded responses, and the cell update is now given by a first-order differential equation, which can be readily implemented in VLSI using R-C circuits. Moreover, an analogous energy function exists for this generalization too. Indeed, even for binary optimization problems, it is preferable to use the continuous form, and then consider limiting values of the cell states to obtain the

[1] Though work on this and related models has been carried out previously by many researchers, this name is most popular because of a paper by Hopfield [5] that sparked widespread interest in these networks.

binary solution. Unfortunately, even then this approach to optimization is often viable only for small problems. This is because the cost reduction only leads us to a local minima, and the probability that this minima is a poor or even invalid solution increases rapidly with increase in problem size (number of variables). Fortunately, related but more sophisticated and powerful schemes for optimization have emerged recently, and can be readily applied for texture segmentation [7, 8].

3 Perceptual Grouping and Edge-based Segmentation.

We perceive an image not as an array of pixels but as agglomerations or *groupings* of more abstract entities. A sharp spatial gradient in gray-scale at an image location may not lead to the perception of an edge at that location if similar gradients do not occur in nearby locations. However, if there are sharp gradients with similar orientations in contiguous regions, then one typically perceives a line or contour formed by series of smoothly connected small edges. Such a contour is a common example of a perceptual grouping.

Indeed, there is a wide variety of grouping mechanisms in human perception. Interestingly, some of these groupings are illusory in that they appear to be very real though there may not be evidence at the pixel level. As an example, Fig. 2(a) shows a Moiré pattern which distinctly gives the appearance of a circular flow pattern even though there are no individual dotted contours running through. The grouping is a Gestalt phenomenon, occurring at a high level of abstraction.

Our visual mechanisms tend to perceptually connect edges that seem part of a longer edge, even across small regions with little gradient changes. This kind of grouping is dramatically indicated by Fig. 2(b), which shows a Kanizsa subjective triangle, demonstrating the formation of illusory contours (in this case an upright triangle).

Detecting edges, then eliminating irrelevant ones and connecting (grouping)the others, are key to successful edge-based segmentation. To this end, cooperative processes such as relaxation labeling have been explored by the vision community for over a decade, without explicitly casting them in a neural network framework. The idea behind relaxation labelling is that local intensity edges typically form a part of a global line or boundary rather than occurring in isolation. Thus the presence or absence of a nearby edge of similar angular orientation would tend to reinforce the hypothesis of the existence of an edge at a given point in the intensity field. Detected line segments are assigned line-orientation labels that labels are iteratively updated by a relaxation process, such that they become more compatible with neighboring labels. Thus adjacent "no-line-detected" labels support one another, and so do lines with similar orientation, while two adjacent labels corresponding to orthogonal orientations antagonize each other.

A neural-like scheme called the boundary contour system (BCS) has been proposed by Grossberg and Mingolla [9] to explain how edges are filled in when part of a boundary is missing, and how illusory contours (such as in Fig. 2(b)) can emerge from appropriately positioned line-terminations. This real-time visual processing model explains a variety of perceptual grouping and segmentation phenomena, including the grouping of textured images. The BCS consists of a hierarchy of locally-tuned interactions that controls the emergence of image segmentation and also detects, enhances and completes boundaries. The interaction of BCS with a feature contour system and an object recognition system, attempts to attain a unifying precept for form, color and brightness. The BCS is largely preattentive in that it is primarily driven by image properties. However, the model does allow feedback from the object recognition system to guide the segmentation process.

The BCS consists of several stages arranged in an approximately hierarchical organization. The image to be processed forms the input to the earliest stage. Here, elongated and oriented receptive fields or masks are employed for local contrast detection at each image position and each orientation. Thus there is a family of masks centered at each location, and responding to a prescribed region around that location. These elliptical masks respond to the amount of luminance contrast over their elongated axis of symmetry, regardless of whether image contrasts are due to differences in textural distribution, a step change in luminance or a smoother intensity gradient. The elongated receptive field makes the masks less sensitive to differences in average contrast in a direction orthogonal to the major axis. However, the penalty for making them sensitive to contrasts in the preferred orientation is the increased uncertainty in the exact locations of contrast. This positional uncertainty becomes acute during the processing of image line ends and corners. The authors assert that all line-ends are illusory in the sense that they are not directly extracted from the retinal image, but are created by some process that generates line terminations. One such mechanism that is hypothesized by them is based on two short-range competitive stages followed by long-range cooperation, as described next.

Firstly, each pair of masks at the same location that are sensitive to the same orientation but opposing direction of contrasts, are input to a common cell. The output of such a cell at position (i, j) and orientation k is J_{ijk}, which is related to the two directional mask outputs, U_{ijk} and V_{ijk} , by:

$$J_{ijk} = \frac{[U_{ijk} - \alpha V_{ijk}]^{+}[V_{ijk} - \alpha U_{ijk}]^{+}}{1 + \beta(U_{ijk} + V_{ijk})}, \tag{6}$$

where the notation [p]+ stands for max(p,0). These oriented cells are sensitive to the amount of contrast, but not the direction. They in turn feed two short-range competitive stages. In the first stage, for cells of the same orientation, there is mutual support (via positive or excitatory connections) among very nearby cells, and competition (via negative or inhibitory connections) among cells that are at an intermediate distance. If the strength of a connection is plotted against the distance between the two cells being connected in 2-D, a Mexican-hat or "on-center, off-surround" pattern is observed. Subsequently, in the second competitive stage, there is competition among orthogonally oriented masks at each position.

Let u_{ijk} represent the output signal for the cell corresponding to position (i, j) and orientation k, and u_{ijK} be the output for the cell at the same location but with orientation orthogonal to k, at the end of the first stage. The u_{ijk}s are obtained from :

$$\frac{d}{dt}u_{ijk} = -u_{ijk}(1 + B \sum_{(p,q)\in R} BJ_{pqk}w_{pqij}) + I + BJ_{ijk} \tag{7}$$

In (7), I is the external input (pixel value), B is a constant, R a neighborhood of (i,j), and w_{pqij} the strength of the negative (inhibitory) connection between positions (p,q) and (i,j). The activity potentials y_{ijk} of cell outputs in the second stage are governed by:

$$\frac{d}{dt}y_{ijk} = -Ay_{ijk} + (E - y_{ijk})O_{ijk} - y_{ijk}\sum_{m\neq k} O_{ijm} \tag{8}$$

where $O_{ijk} = C[u_{ijk} - u_{ijK}]+$, and A, C and E are constants.

The behavior of the orientation field is shown in Figure 3, in which adjacent lattice points are one unit apart. Each mask has a total exterior dimension of 16 x 8 units. Figure 3(b) shows the y_{ijk} responses at the end of the second competitive stage for the same input stimulus. The two competitive stages together have generated end cuts as can be seen clearly on comparing with Fig. 3(a). Note that the second competitive stage has the property that inhibition of a vertical orientation excites the horizontal orientation at the same position, and vice versa.

The outputs of the second stage are also used for the boundary completion process that involves long-range cooperation between similarly oriented pairs of input groupings. This mechanism is able to complete boundaries across regions that receive no bottom-up inputs from the oriented receptive fields, and thus accounts for illusory line phenomena such as the completion of the edges in a reverse-contrast Kanisza triangle of Fig. 2(b). The process of boundary completion occurs discontinuously across space, using the gating properties of the cooperative cells to successively interpolate boundaries within progressively finer intervals. Unlike a low spatial frequency filter, this process does not sacrifice spatial resolution to achieve a broad spatial range. The cooperative cells used in this stage also provide positive feedback to the cells of the second competitive stage so as to increase the activity of cells of favored orientation and position, thereby providing them with a competitive edge over other orientations and positions. This feedback helps in reducing the fuzziness of boundaries. The detailed architecture, equations and simulation results can be found in [9].

The BCS approach can also form the basis for a hierarchical neural network for texture segmentation and labelling, as shown by Dupaguntla and Vemuri. The underlying premise is that textural segmentation can be achieved by recognizing local differences in texels. The architecture consists of a feature extraction network whose outputs are used by a texture discrimination network. The feature extraction network is a multilayer hierarchical network governed by the BCS theory. The image intensities input is first preprocessed by an array of cells whose receptive fields correspond to a difference of Gaussian filters, and which follow the feed-forward shunting equations of Grossberg. The output of this array of cells form the input to a BCS system and are processed by oriented masks according to (6). These masks then feed into the two competitive stages of the BCS theory, and governed by (7) and (8). However, the long-range cooperative processes described above are not used. Instead, the outputs of the second competitive stage activate region encoding (RE) cells at the next level. Each RE cell gathers its activity from a region of orientation masks of the previous layer, as well as from a neighborhood of adjacent RE nodes of the same orientation. The activity potential of an RE node is given by the following equation, where the y_{lmk}s are obtained from the previous layer according to (8), (l, m) is in the neighborhood of (p, q), and the activation function, f, is sigmoidal.

$$\frac{d}{dt}z_{ijk} = -\mu z_{ijk} + \mu \sum_{(p,q)} (f(z_{pqk}) - f(z_{ijk})) + \sum_{(l,m)} y_{lmk}. \tag{9}$$

The RE cells appear to be functionally analogous to the complex cells in the visual cortex, with the intra-layer connections helping to propagate orientation information across this layer of cells. The outputs (z_{ijk}s) of the feature extraction network are used by a texture discrimination network that is essentially Kohonen's single-layered self-organizing feature map [10]. At each position, there are T outputs, one for each possible texture type, which is assumed to be known a priori. Model (known) textures are passed through the feature extraction network. For a randomly selected position (i, j), the output cell of the texture discrimination network that responds maximally is given the known texture-type label. The weights in the texture discrimination network for that position are adapted according to the feature-map equations. Since these weights are the same for all positions, one can simply replicate the updated weights for all positions. The hierarchical scheme described above has been applied to natural images with good results. However, it is very computationally intensive, since there are cells corresponding to each orientation and position at every hierarchical level.

4 Adaptive Multi-channel Modelling for Texture-based Segmentation

Image texture provides useful information for segmentation of scenes, classification of surface materials and computation of shape, and is exploited by sophisticated biological vision systems for image analysis [11]. In 1980, Marcelja observed that highly-oriented simple cell receptive fields in the cortex can be accurately modeled by 1-D Gabor functions, which are Gaussian modulated sine wave functions. The Gabor functions play an important role in functional analysis and in physics, since they are the unique functions that satisfy the uncertainty principle, which is a measure of the function's simultaneous localization in space and in frequency. Daugman [12] successfully extended Marcelja's neuronal model to 2-D, also extending Gabor's result by showing that the 2-D Gabor functions are the unique minimum-uncertainty 2-D functions. The implication of this for texture analysis purposes, and perhaps for neuronal processing of textured images, is that highly accurate measurements of textured image spectra can be made on a highly localized spatial basis. This simultaneous localization is important, since then it is possible to accurately identify sudden spatial transitions between texture types, which is important for segmenting images based on texture, and for detecting gradual variations within a textured region.

Based on these observations, a multiple channel Gabor filter bank has been used to segment textured images [11]. Each filter's response is localized in the frequency $(u-v)$ plane. A large set of these channel filters is used to sample the frequency plane densely to ensure that a filter exists that will respond strongly to any dominant texture frequency component. Segmentation can be performed by assigning each pixel the label of the maximally responsive filter centered at that pixel. The success of this technique is quite impressive, given that no use use is made of any sophisticated pattern classification superimposed on the basic segmentation structure. More details on multichannel image segmentation can be found in Chapter 4.7 of this handbook.

Some smoothing of the filter outputs before doing the max operation provides better results on texture segmentation. Further improvements can be achieved by using a cooperative-competitive feedback network called Smoothing, Adaptive Winner-Take-All Network (SAWTA) [11]. This network consists of n layers of cells, with each layer corresponding to one Gabor filter, as shown in Fig. 4. On the presentation of an image, a feedforward network using local receptive fields enables each cell plane to reach an activation level corresponding to the amplitude envelope of the Gabor filter that it represents, as outlined in the preceding paragraphs. Let $m_i(x,y), 1 \leq i \leq n$, be the activation of the cell in the ith layer with retinotopic coordinates (x,y). Initially, the n cell activations at each point (x,y) are set proportional to the amplitude responses of n Gabor filters.

To implement the SAWTA mechanism, each cell receives constant inhibition from all other cells in the same column, along with excitatory inputs from neighboring cells in the same row or plane. The synaptic strengths of the excitatory connections exhibit a 2D Gaussian profile centered at (x,y). The network is mathematically characterized by shunting cooperative-competitive dynamics [9] that model on-center off-surround interactions among cells which obey membrane equations. Thus, at each point (x,y), the evolution of the cell in the ith layer is governed by:

$$\tau \frac{d}{dt}(m_i) = -m_i + (A - m_i)J^+ - (B + Cm_i)J^- \qquad (10)$$

where J^+, J^- are the net excitatory and inhibitory inputs respectively, and are given by:

$$J^+ = \alpha \sum_{(x_n,y_n)\in R} m_i(x_n,y_n) e^{\frac{-(x-x_n)^2+(y-y_n)^2}{2\sigma^2}}; \qquad J^- = \sum_{j\neq i} f(m_j(x,y)). \qquad (11)$$

Here, R is the neighboring region of support and f is a sigmoidal transfer function.

A sigmoidal transfer function is needed to keep the response bounded between 0 and 1 while still maintaining a monotonically increasing response with the argument.

The convergence of a system described by (10) has been shown for the case when the region of support R consists of the single point (x, y). The network is allowed to run for ten iterations before region assignment is performed by selecting the most responsive filter.

Figures 5 and 6 shows comparative experimental result using the SAWTA network for segmentation. The 256x256 gray level images are prefiltered using Laplacian-of-Gaussian filters[2] to remove high dc components, low-frequency illumination effects, and to suppress aliasing. Then, only sixteen circularly-symmetric Gabor filters are used to detect narrow-band components as follows: Sets of three filters with center frequencies increasing in geometric progression (ratio = 2:1) are arranged in a daisy-petal configuration along 5 orientations, while the sixteenth filter is centered at the origin. Figure 5 shows the segmentation achieved for a synthetic texture using three different techniques. 5(a) is the original image; 5(b) is the result of the original multichannel segmentation model [13], 5(c) the results of this model with output smoothing, and 5(d) segmentation after 10 iterations of the SAWTA network. The constants A, B and C in (10) were taken to be 1, 0 and 10 respectively. The activation function used is $\mathrm{f}(x) = \tanh(2x)$. The results are seen to be superior to that obtained by the original multi-channel based segmentation scheme.

Figure 6 shows the effect of varying the number of iteration steps, and the inhibition factor C, on the segmentation obtained. We observe that the SAWTA network achieves a more smooth segmentation in regions where the texture shows small localized variations, while preserving the boundaries between drastically different textures. Usually, 10 iterations suffice to demarcate the segment boundaries, and any changes after that are confined to arbitration among neighboring filters.

The SAWTA network does not require a feature extraction stage as in [14] or computationally expensive masking fields. The incremental and adaptive nature of the SAWTA network enables it to avoid making early decisions about texture boundaries. The dynamics of each cell is affected by the image characteristics in its neighborhood as well by the formation of more global hypotheses. It has been observed that usually four spatial frequencies are dominant at any given time in the human visual system. This suggests the use of a mechanism for post-inhibitory response that suppress cells with activation below a threshold and speeds up the convergence of a SAWTA network. The SAWTA network can be easily extended to allow for multiple "winners". Then, it can cater to multicomponent textures, since a region that contains two predominant frequencies of comparable amplitude will not be segmented but rather viewed as a whole.

Learning and adaptation is also useful in the multi-channel image model for determining the channels (filters) themselves. Indeed, the results of Gabor filtering can be obtained in an iterative fashion, by performing stochastic gradient descent on a suitable cost function. While these filters are useful for a large variety of images, one may wonder whether

[2] for a description of such filters, see Chapter 4.10.

more customized filters may yield better results for a specific class of images, such as images of barcodes, or MRI scans for the brain. This leads to the concept of "texture discrimination masks", which may be learned in order to improve performance in the subsequent classification task [15].

First, note that the multichannel framework does not restrict one to using Gabor filters. Other filters reported include Laplacians of Gaussians, wavelets, and general IIR and FIR filters. Each filter can be considered as a localized feature detector, and after performing spatial smoothing if needed, the filter outputs for each pixel can serve as inputs to a multilayered feedforward network such as the multi-layered perceptron (MLP) that performs the desired classification task. Thus we effectively have an MLP classifier with an additional hidden layer, i.e., the layer whose inputs are a the pixel values in a small image window, and whose afferent weights represent the mask coefficients. While training this network, the filter weights get modified to better perform texture classification. Moreover, by applying node pruning techniques, less important filters can be eliminated. Thus, instead of the usual large set of generic filters, a smaller set of task-specific filters is evolved. Details of this method, along with superior results obtained on page layout segmentation and barcode localization, can be found in [15]. It is speculated that the efficacy of the learned masks stems from their ability to combine different frequency and directionality responses in the same masks, so that high discrimination information can be captured by a smaller number of filters. On the other hand, if the problem domain changes substantially, a new set of filters needs to be learned for the new set of images.

5 An Optimization Framework

The use of Markov Random Field (MRF) models for modeling texture has been investigated by several researchers (see Chapter 4.2). It can be used to model the texture intensity process as well as to describe the texture labelling process. In this framework, segmentation of textured images is posed as an optimization problem. Two optimality criteria considered in [16] are (i) to maximize the posterior distribution of the texture label field given the intensity field, and (ii) to minimize the expected percentage of misclassification per pixel by maximizing the posterior marginal distribution. Corresponding to each criteria, an energy (cost) function can be derived that is a function of $M \times M \times K$ binary labels, one for each of the K possible texture labels that a pixel in an $M \times M$ image can take.

A neural network solution to minimizing this cost function is provided via the discrete Hopfield-Tank formulation described in section 2 [6]. A 3-dimensional lattice of binary (ON/OFF) neurons is used, with one neuron for each of the $M \times M \times K$ labels. The cost function chosen imposes a severe penalty unless exactly one neuron is ON at each of the $M \times M$ positions. The location of this neuron in the third dimension provides the label to be given to the corresponding pixel. The other terms in the cost function encourage solutions where the same label is given to neighboring pixels, and at the same time this class has high probability of occuring given the initial gray levels values of the pixels in the neighborhood [16]. The cost function is quadratic in the neuron output values, and indeed has the form of (5). In Sec. 2 we saw that for such cost functions, a network with simple computing cells and local connections can be specified such that the cost is steadily reduced as the cells update their state, till a local minimum of the cost function is realized. This usually happens in 20-30 iterations, but the quality of the texture labeling thus obtained is quite sensitive to initial conditions as it has a penchant for settling into local optima. Alternatively, a stochastic algorithm such as simulated annealing can be used to minimize the energy function. Indeed, any problem formulated in terms of minimizing an energy function can be given a probabilistic interpretation by use of the Gibbs distribution. The two approaches are related in that a mean field approximation of the stochastic algorithm yields the update equations of the network described above, with the free parameter being proportional to the inverse of the annealing temperature [6].

For the segmentation problem, a constraint on a valid solution is that each image position should have only one of the K labels "on". This constraint is usually incorporated in a soft fashion by adding bias terms to the energy function. Peterson & Soderberg have incorporated the 1-of-K constraint in a Potts glass, and derived a mean field solution for that formulation. The alternative of putting global constraints on the set of allowable states in the corresponding stochastic formulation leads to significantly better solutions. An iterated hill climbing algorithm that combines fast convergence of the deterministic relaxation with the sustained exploration of the stochastic approach has also been proposed in [16] for the segmentation problem. Here, two-stage cycles are used, with the equilibrium state of the relaxation process providing the initial state for a stochastic learning automaton within each cycle.

The relation between neural network techniques and MRFs was explored in detail in [17, 16]. Since the optimization techniques applied were largely rooted in the Hopfield-Tank formulation, they were plagued by large training times, and high possibility of being caught in local minima, leading to poor solutions. Fortunately, more sophisticated and powerful schemes for optimization have emerged recently, and can be readily applied for texture segmentation [7, 8].

The MRF framework can also directly leverage the powerful mapping capabilities of feedforward networks. For example, Hwang & Chen has used an MLP to directly obtain the class distributions conditional on the neighborhood image statistics (needed for the MRF), based on training image samples. This obviates restrictive parametric representation and tedious parameter estimation for the MRF. Of note is the use of the Karhunen-Loeve cost criterion instead of the popular mean squared error, since the former tends to give more accurate estimates of low probability values.

6 Image Segmentation via Adaptive Clustering

The use of clustering for image segmentation dates back to the late 1960s and many of the techniques developed then are still in popular use today. In this approach each pixel is represented by a vector of features based on information derived from image characteristics in its neighborhood, as well as positional information. Similar feature vectors are then grouped together in clusters, and each cluster is given a texture label. Thus pixels that are nearby and have similar local image properties will tend to get grouped together and get the same label.

Clustering can be fruitfully applied to a variety of image domains, including multi-spectral images, range images, textured images and intensity images from dot patterns to gray level or colored images. Key issues in the design of any clustering based segmenter are the choice of the number and type of features used, the distance metric chosen to measure similarity, data reduction techniques used, and the pre- and post-processing routines applied. If the design choices are made suitably, the feature vectors will form more compact and well separated regions in the multidimensional feature space, and one can thus reliably segment the images based on both these regions and on image connectivity. A nice overview of this area can be found in [4].

Neural network research has spawned a variety of adaptive clustering techniques, from competitive learning - an iterative version of K-means clustering, to Learned Vector Quantization (LVQ) [10] - a supervised clustering and classification technique related to classical Vector Quantization. In LVQ, a set of labelled cluster centers (the code-book vectors) are first chosen by random subsampling or by K-means clustering of the data. Then, for every training sample the position of the nearest codebook vector is moved towards or away from that sample depending on whether the two labels match or not. Several variations exist.

Instead of placing each sample in a unique cluster, on can "softly" associate a sample with multiple clusters. The resulting clusters are sometimes called fuzzy clusters as their boundaries are not sharply delineated. Let the association of sample x_i with cluster j be denoted by $a_{i,j}$. We want this association value to decrease as the distance between x_i and the center of the jth cluster increases. Also, it is desired that the associations be non-negative, and that $\sum_j a_{i,j} = 1, \;\; \forall i$. The jth cluster center is simply $\sum_i a_{i,j} x_i$. Depending on how the associations are formed and updated, a variety of powerful fuzzy clustering approaches have been obtained [18].

A critical issue in clustering is the choice of an appropriate scale, which determines the number of clusters obtained and hence the amount of segmentation obtained. For a given image, there are some natural scales for which the clusters are relatively well defined and stable in the sense that the optimum center locations change little with small variations in scale. In fact, just as scale-space theory views salient edges to be those that survive over multiple scales, one can view salient segments in the same way. Statistical mechanics based formulations for clustering provide a nice approach to the issue of scale, which is naturally related to the temperature parameter. At high temperature there are many clusters, and as the temperature is lowered, some of the clusters coalesce. Stable clusters are those that survive over a wide range of temperatures [19]. It turns out that if we adjust only the kernel locations using gradient descent in Gaussian radial basis function network, maintaining fixed and equal output layer weights, fixed widths, and a constant target function, then these locations converge to "optimum" cluster locations for the chosen scale, now indicated by the widths (σs) of the Gaussian units [20]. Moreover, different scales may be indicated as being appropriate for different parts of an image for segmentation purposes. Thus such networks are promising for segmentation with locally adaptive resolution.

7 Oscillation Based Segmentation

It will be remiss not to mention that there are several biologically oriented approaches to segmentation. In such approaches, it becomes clear that segmentation is closely tied with several other mechanisms. For example, when we view an apple, we regard it not as a smear of red amidst a riot of undifferentiated color, but recognize it as distinct desired object. How exactly we do this constitutes the *sensory segmentation* problem. When we thus discern an apple, we naturally separate it from the uninteresting background - this is the *figure-ground separation* problem. When we take a further step, like biting into the apple, then our experience is not merely a jumble of tactile, olfactory and visual sensations, but that these different modes of sensations correspond to a single object - an apple; this is the *binding* problem. The aforementioned problems are intimately interrelated.

A single physical stimulus usually activates several groups of neurons corresponding to different sensory modalities. How does the brain then realize that these groups correspond to the same object? A popular hypothesis is for answering this query is that neurons responding to aspects of a single object fire in synchrony. Applied to visual perception, the hypothesis states that cortical neurons corresponding to a distinct homogeneous area oscillate in phase, and those corresponding to different areas are out of phase.

Supportive of this hypothesis, stimulus-dependent oscillations that are correlated temporally and spatially have been found in the visual cortex of cats and monkeys. These experiments have motivated several oscillating neuron models of sensory segmentation, which largely fall into two broad categories: (i) those in which synchronization is achieved by cooperation/competition among neurons via fixed excitatory and inhibitory couplings, and (ii) those in which synchronization evolves by locally hebbian-like synaptic modification, i.e. synaptic strength increases if both pre-synaptic and post-synaptic activity are simultaneous high, and decreases otherwise. In the former, the visual input is merely transformed into segregated oscillations, whereas in the latter, the input is encoded in modifiable synapses. For example, in [21], oscillating units, each consisting of excitatory and inhibitory cells, are connected by weights, modulated in a "pre-post-synaptic" fashion. Some of these essential ingredients can be seen in several of the subsequent models, e.g., the approach of Konig, wherein synchronization among oscillating units depends on similarity of local features, and segmentation can be achieved by local learning rules.

In [22], a model is proposed in which synchronization of neural oscillations is produced *both* by (i) cooperation and (ii) synaptic modification. It is demonstrated that either of these mechanisms is sufficient to generate coherent oscillations, but the two can be viewed as components of a more integral mechanism for neural synchronization.

In this model, the state of an oscillating unit or a neuron is described by a complex number, z, and each unit is connected to every other unit. The dynamics of the model is a generalization of the Hopfield equations in the complex plane, and is described by:

$$\frac{dz_j}{dt} = \sum_k T_{jk} V_k - (\nu + i(1-\nu)) z_j + I_j \tag{12}$$

$$V_j = tanh(\lambda(\nu + i(1-\nu)) z_j^*) \tag{13}$$

where the quantities, z_j, T_{jk}, and V_k are complex numbers. The real part of the neuron state, Re[z], is analogous to the transmembrane potential of the real neuron; the real part of V is the output firing rate; I_j, is the sum of the external currents entering the jth neuron, and T_{jk}, is the *weight* connecting jth and kth neurons. The mode parameter ν governs the qualitative nature of the above model. For ν close to 0, the model (12, 13) exhibits oscillatory behavior, and for ν near 1, it has fixed-point dynamics.

One can show that oscillations are produced in the model above if the cells are arranged in a 2-D grid, and the weights T_{jk} are real and have a "Mexican-hat" profile. Suppose the 32×32 image of a 'plus (+)' symbol with noise added (Fig. 7) forms the input to a 32×32 grid of cells. The image is presented for an interval of time (210 iterations in this example) and then removed, and the subsequent evolution of the network output is followed. It is seen that cells over the 'plus' region start oscillating more or less in phase, and are about 180 degrees out of phase with the rest of the cells. Fig. 8 depicts the oscillation of a typical neuron. Subsequent to input removal, network response reveals excellent noise removal with precise figure boundaries as indicated in Fig. 9 which was taken after 450 iterations. It is also observed that the amount of noise in the interpreted image decreases right from the first iteration, and Fig. 9 shows almost no noise.

If the weights are random rather than "Mexican-hat", the network exhibits coherent oscillations only while the input is present, and coherency is destroyed subsequent to input removal.

Alternatively, synchronization can be produced, without any special predetermined neighborhood, if the weights are not fixed but modified by Hebbian learning. In Hebbian form of learning, the connection between a pair of simultaneously active neurons is strengthened, and is expressed in this model as,

$$\dot{T}_{jk} = -\gamma T_{jk} + g((\nu + i(1-\nu)) z_j^*) g((\nu + i(1-\nu)) z_k) \tag{14}$$

where $g(\cdot)$ is the sigmoid nonlinearity introduced in (13). Eqn. (14) is always simulated together with (12, 13). External input I in (12) produces changes in T_{jk} indirectly via neuron outputs, V_k. As a result, the input pattern is encoded in the weights, and input-dependent synchronization takes place as an emergent effect. Fig. 10 shows the result (for the same input image of Fig. 7) after 530 iterations, using random initial weights that are adapted using the Hebbian learning equations.

The two mechanisms described above for synchronized neural oscillations seem unconnected. However Linsker and others have shown that Mexican-hat like neighborhoods can develop automatically in a multi-layered network with weights adapted by Hebbian learning. Such neighborhoods seem to be canonical for producing stimulus-specific oscillations, obtained as an average effect of learning a large number of patterns. This is supported by experimental results [22].

8 Integrated Segmentation and Recognition

Often segmentation is an intermediate step towards object recognition or classification from 2-D images. For example, segmentation may be used for figure-ground separation or for isolating image regions that indicate objects of interest as differentiated from background or clutter. Even small images have lots of pixels - there are over 64 thousand pixels in a 256x256 image. So it is impractical to consider the raw image as an input to an object recognition or classification system. Instead, a small number of descriptive features are extracted from the image, and then the image is classified or further analyzed based on the values of these features. ANNs provide powerful methods for both the feature extraction and classification steps and have been used with much success in integrated segmentation and recognition applications.

Feature Extraction. The quality of feature selection/extraction limits the performance of the overall pattern recognition system. One desires the number of features to be small but highly representative of the underlying image classes, and highly indicative of the differences among these classes. Once the features are chosen, different methods typically give comparable classification rates when used properly. Thus feature extraction is the most crucial step. In fact, the Bayes error is defined for a given choice of features, and a poor choice can lead to a high Bayes rate.

Perhaps the most popular linear technique for feature extraction is Principal Component Analysis (PCA) (sometimes referred to as the Karhunen-Loeve transform), wherein data is projected in the directions of the principal eigenvectors of the input covariance matrix. There are several iterative "neural" techniques where weight vectors associated with linear cells converge to the principal eigenvectors under certain conditions. The earliest and most well-known of these is Oja's rule, where the weights w_i of a linear cell with single output

$y = \sum_i x_i w_i$, are adapted according to:

$$w_i(n) = w_i(n-1) + \eta(n)y(n)[x_i(n) - y(n)w_i(n-1)]. \tag{15}$$

The learning rate is $\eta(n)$ and should satisfy the Robbins-Munro conditions for convergence, and $x_i(n)$ is the ith component of the input presented at the nth instant. The inputs are presented at random. Then it can be shown that, if $\mathbf{x}$ is a zero-mean random variable, the weight vector converges to unit magnitude with its direction the same as that of the principal eigenvector of the input covariance matrix. In other words, the output y is nothing but the principal component after convergence! Moreover, by feeding the "residual", $x_i(n) - y(n)w_i(n-1)$ into another similar cell, the second principal component is iteratively obtained and so on. Moreover, to make the iterative procedure robust against outliers, one can vary the learning rate so that it has a lower value if the current input is less probable.

The nonlinear discriminant analysis network proposed by Webb and Lowe [23] is a good example of a non-linear feature extraction method. They use a multilayer perceptron with sigmoid hidden units and linear output units. The nonlinear transformation implemented by the subnetwork from the input layer to the final hidden layer of such networks tries to maximize the so-called network discriminant function, $\mathrm{Tr}\{S_B {S_T}^+\}$, where ${S_T}^+$ is the pseudo-inverse of the total scatter matrix of the patterns at the output of the final hidden layer, and S_B is the weighted between-class scatter matrix of the output of the final hidden layer. The role of the hidden layers is to implement a nonlinear transformation which projects input patters from the original space to a space in which patterns are more easily separated by the output layer.

A nice overview of neural based feature techniques is given by Mao and Jain [24], who have also compared the performance of five feature extraction techniques using 8 different data sets. They note that while several such techniques are nothing but on-line versions of some classical methods, they are more suitable for mildly nonstationary environments, and often provide better generalization. Some techniques such as Kohonen's feature map, also provides a nice way of visualizing higher dimensional data in 2 or 3-D space.

Classification. Several feedforward neural networks have properties that make them promising for image classification based on the extracted features. ANN approaches have led to the development of various "neural" classifiers using feed-forward networks. These include the Multi-Layer Perceptron (MLP) as well as kernel-based classifiers such as those employing Radial Basis Functions, both of which are described in Section 2. Such networks serve as adaptive classifiers that learn through examples Thus, they do not require a good *apriori* mathematical model for the underlying physical characteristics. A good review of probabilistic, hyperplane, kernel and exemplar-based classifiers that discusses the relative merit of various schemes within each category, is available in [25]. It is observed that, if trained and used properly, several neural networks show comparable performance over a wide variety of classification problems, while providing a range of trade-offs in training time, coding complexity and memory requirements. Some of these networks, including the multilayered perceptron when augmented with regularization, and the elliptical basis function network, are quite insensitive to noise and to irrelevant inputs. Moreover, a firmer theoretical understanding of the pattern recognition properties of feed-forward neural networks has emerged that can relate their properties to Bayesian decision making and to information theoretic results [26].

Neural networks are not "magical". They do require that the set of examples used for training should come from the same (possibly unknown) distribution as the set used for testing the networks, in order to provide valid generalization and good performance on classifying unknown signals [27]. Also, the number of training examples should be adequate and comparable to the number of *effective* parameters in the neural network, for valid results. Interestingly, the complexity of the network model, as measured by the number of effective parameters, is not fixed, but increases with the amount of training. This provides an important knob: one can start with an adequately powerful network and keep on training till its complexity is of appropriate size. In practice, the latter may be readily arrived at by monitoring the network's performance on a validation set.

Training sufficiently powerful multilayer feedforward networks (e.g. MLP, RBF) by minimizing the expected mean square error (MSE) at the outputs and using a 0/1 teaching function yields network outputs that approximate posterior class probabilities [26]. In particular, the MSE is shown to be equivalent to

$$MSE = K_1 + \sum_{i=1}^{c} \int_{\mathbf{x}} D_i(\mathbf{x})(P(C_i|\mathbf{x}) - f_i(\mathbf{x}))^2 d\mathbf{x} \tag{16}$$

where K_1 and $D_i(\mathbf{x})$ depend on the class distributions only, $f_i(\mathbf{x})$ is the output of the node representing class C_i given an input $\mathbf{x}$, $P(C_i|\mathbf{x})$ denotes the posterior probability, and the summation is over all classes. Thus, minimizing the (expected) MSE corresponds to a weighted least squares fit of the network outputs to the posterior probabilities. Somewhat similar results are obtained by using other cost functions such as cross-entropy.

The above result is exciting because it promises a direct way of obtaining posterior class probabilities and hence attaining the Bayes optimum decision. In practice of course, the exact posterior probabilities may not be obtained, but only an approximation thereof. (If it had, the Bayes error rate could have been attained.) This is because in order to minimize (16), one needs to (i) use an adequately powerful network so that $P(C_i|\mathbf{x})$ can be realized, (ii) have enough number of training samples, and (iii) find the global minima in weight space. If any of the above conditions are violated, different classification techniques will have different inductive biases, and a single method cannot give the best results for all problems. Rather, more accurate and robust classification can obtained by combining the outputs (evidences) of multiple classifiers based on neural network and/or statistical pattern recognition techniques.

Pattern Recognition Techniques Specific to Segmentation. While the above discussion applies to generic pattern recognition systems, image segmentation has specific characteristics that may call for custom approaches. Firstly, some invariance to (small) changes in rotation, scale or translation is often desired. For example, in OCR, tolerances to such minor distortions is a must. Invariance can be achieved by (i) extracting invariant features such as Zernicke moments, (ii) by providing additional examples with different types of distortion, e.g. for each character in OCR also present various rotated, scaled or shifted versions, or (iii) making the mapping robust to invariances via weight replications or symmetries. The last alternative is the most popular, and has led to specialized feedforward networks with *two* or more hidden layers. Typically, the early hidden layers have cells with local receptive fields, and their weights are shared among all cells with similar purpose (i.e. extracting the same features) but acting on different portions of the image. A good example is the *convolutional net*, [28] where the first hidden layer may be viewed as a 3-D block of cells. Each column of cells extract different features from the corresponding *localized* portion of the image. These feature extractors essentially perform convolution using nonlinear FIR filters. They are replicated at other localized portions by having identical weights among all cells in the same layer of the 3-D block. Multiple layers with subsampling is proposed to form an image processing pyramid. Higher layers are fully connected to extract more global information. For online handwriting recognition, a Hidden Markov Model post-processor can be used. Remarkable results on document recognition are given in [28].

In an integrated segmentation and recognition scheme, it may even be possible to avoid the segmentation step altogether. For example, it is well known that presegmented characters are relatively easy to classify, but isolating such individual characters from handwriting is difficult. On can however develop a network that avoids this segmentation by making decisions only if the current window is centered on a valid character, and otherwise give a "non-centered" verdict. Such networks can also be trained with handwriting that is not pre-segmented, thus saving substantial labor.

9 Concluding Remarks

Neural network based methods can be fruitfully applied in several approaches to image segmentation. While many of these methods are closely related to classical techniques involving distributed iterative computation, new elements of learning and adaptation are added. Such elements are particularly useful when the relevant properties of images are nonstationary, so continuous adaptation can yield better results and robustness than a fixed solution. On the other hand, most of the methods have not been fully developed as products with friendly GUI that a non-expert end user can obtain off-the-shelf and readily use. Moreover, detailed comparative analysis is desired for several of the techniques described in this chapter, to further understand when they are most applicable. Thus, further analysis, benchmarking, product development and system integration is necessary if these methods are to gain widespread acceptance.

An exciting aspect of neural network based image processing is the prospect of parallel hardware realization in analog VLSI chips such as the silicon retina [1]. Such analog chips use networks of resistive grids and operational amplifiers to perform edge detection, smoothing, segmentation, compute optic flow, etc., and can be readily embedded in a variety of smart platforms, from toy autonomous vehicles that can track edges, movements, etc., to security systems to retinal replacements [2]. Further progress towards the development of low-power, real-time vision hardware requires an integrated approach encompassing image modeling, parallel algorithms and the underlying implementation technology.

Acknowledgments: I would like to thank Al Bovik for his friendship and inspiration over the years, and for numerous perceptive and humorous comments on this chapter that greatly helped in improving its quality. This work was funded in part by ARO contracts DAAH04-95-10494 and DAAG55-98-1-0230.

References

[1] C.A. Mead. *Analog VLSI and Neural Systems.* Addison-Wesley, Reading, Mass, 1989.

[2] C. Koch and B. Mathur. Neuromorphic vision chips. *IEEE Spectrum*, pages 38–46, May 1996.

[3] N.R. Pal and S.K. Pal. A review on image segmentation techniques. *Pattern Recognition*, 26(9):1277–94, 1993.

[4] A. K. Jain and P. J. Flynn. Image segmentation using clustering. In K. Boyer and N. Ahuja, editors, *Advances In Image Understanding: A Festschrift for Azriel Rosenfeld*, pages 65–83. IEEE Computer Society Press, 1996.

[5] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings National Academy of Science*, 79:2554–2558, Apr. 1982.

[6] J. J. Hopfield and David W. Tank. Neural computation of decisions in optimization problems. *Biological Cybernetics*, 52:141–152, 1985.

[7] A. Rangarajan, S. Gold, and E. Mjolsness. A novel optimizing network architecture with applications. *Neural Computation*, 8:1041–60, 1996.

[8] K. Smith, M. Paliniswami, and M. Krishnamoorthy. Neural techniques for combinatorial optimization with applications. 9(6):1301–09, 1998.

[9] S. Grossberg and E. Mingolla. Neural dynamics of perceptual grouping: Textures, boundaries and emergent segmentations. *Perception and Psychophysics*, 38:141–171, 1985.

[10] T. Kohonen. *Self-Organizing Maps.* Springer-Verlag, Berlin, 2nd Ed. 1997.

[11] J. Ghosh and A. C. Bovik. Processing of textured images using neural networks. In I.K. Sethi and A. Jain, editors, *Artificial Neural Networks and Statistical Pattern Recognition*, pages 133–154. Elsevier Science, Amsterdam, 1991.

[12] J. G. Daugman. Complete discrete 2-D Gabor transforms by neural networks for image analysis and compression. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 36:1169–1179, July 1988.

[13] A. C. Bovik, M. Clark, and W. S. Geisler. Multichannel texture analysis using localized spatial filters I: Segmentation by channel demodulation. *IEEE Trans. PAMI*, 12(1):55–73, 1990.

[14] N. R. Dupaguntla and V. Vemuri. A neural network architecture for texture segmentation and labeling. In *International Joint Conference on Neural Networks*, pages 127–144 (I), June 1989.

[15] A.K. Jain and K. Karu. Learning texture discrimination masks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(2):195–205, 1996.

[16] B.S. Manjunath, T. Simchony, and R. Chellappa. Stochastic and deterministic networks for texture segmentation. *IEEE Trans. on Acoustics, Speech and Signal Processing*, 38:1039–1049, June 1990.

[17] A. Rangarajan, R. Chellappa, and B.S. Manjunath. Markov random fields and neural networks with applications to early vision problems. In I.K. Sethi and A. Jain, editors, *Artificial Neural Networks and Statistical Pattern Recognition*, pages 155–174. Elsevier Science, Amsterdam, 1991.

[18] J. C. Bezdek and S.K. Pal. *Fuzzy Models for Pattern Recognition.* IEEE Press, Piscataway, NJ, 1992.

[19] K. Rose. Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. *Proc. IEEE*, 86(11):2210–39, 1998.

[20] S. V. Chakaravathy and J. Ghosh. Scale based clustering using a radial basis function network. *IEEE Transactions on Neural Networks*, 2(5):1250–61, Sept 1996.

[21] C. von der Malsburg and W. Schneider. A neural cocktail-party processor. *Biological Cybernetics1*, 54:29–40, 1986.

[22] S. V. Chakravarthy, V. Ramamurti, and J. Ghosh. A network of oscillating neurons for image segmentation. In P. Chen B. Fernandez C. Dagli, M. Akay and J. Ghosh, editors, *Intelligent Engineering Systems Through Artificial Neural Networks, Volume 5.* ASME Press, 1995.

[23] D. Lowe and A. R. Webb. Optimized feature extraction and the bayes decision in feed-forward classifier networks. *IEEE Trans. PAMI*, 13:355–364, April 1991.

[24] J. Mao and A.K. Jain. Artificial neural networks for feature extraction and multivariate data projection. *IEEE Transactions on Neural Networks*, 6:2:296–317, March 1995.

[25] K. Ng and R.P. Lippmann. Practical characteristics of neural network and conventional pattern classifiers. In J.E. Moody R.P. Lippmann and D.S. Touretzky, editors, *Advances in Neural Information Processing Systems-3*, pages 970–976, 1991.

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

[27] J. Ghosh and K. Tumer. Structural adaptation and generalization in supervised feed-forward networks. *Jl. of Artificial Neural Networks*, 1(4):431–458, 1994.

[28] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324, 1998.

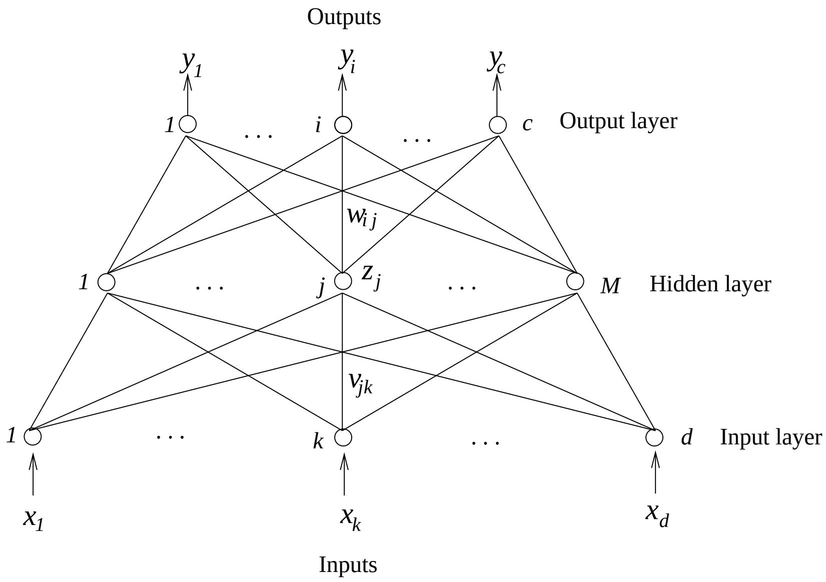

Figure 1: Topology of a feedforward ANN with a single hidden layer.

Figure 2: Examples of perceptual grouping. (a) A random-dot Moiré pattern created by taking a pattern of 1000 dots and superimposing it with a copy rotated about 2 degrees. (b) The Kanizsa subjective triangle.

Figure 3: (a) Shows the output of the oriented masks superimposed on the input pattern (shaded area). Lengths and orientations of lines encode the relative sizes of the activations and orientations of the masks at the corresponding positions. 3(b) shows the output of the second competitive stage for the same input as in 3(a) (Grossberg & Mingolla 85).

Figure 4: The SAWTA network for segmentation of textured images.

Figure 5: Segmentation of a synthetic texture using the SAWTA network (clockwise from top left): (a) Original Image; b) Result of the original multichannel segmentation model [13];(c) the results of this model with output smoothing;(d) segmentation after 10 iterations of the SAWTA network.

Figure 6: Effect of iteration steps and inhibition factor on segmentation: (a) same as Fig. 3(d); (b) segmentation with C = 3; (c) after 100 iterations; (d) after 100 iterations.

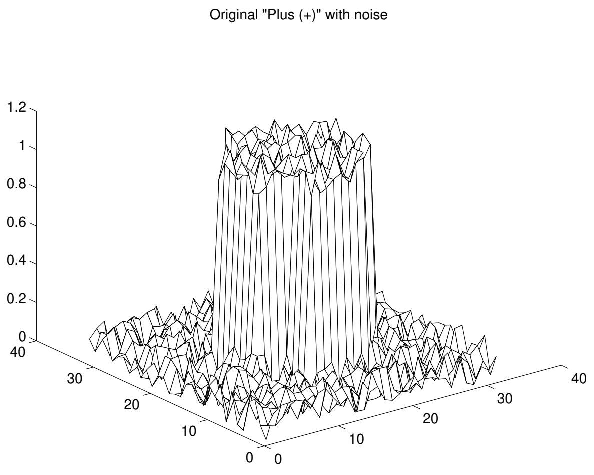

Figure 7: Original Plus(+) image corrupted with uniformly distributed noise.

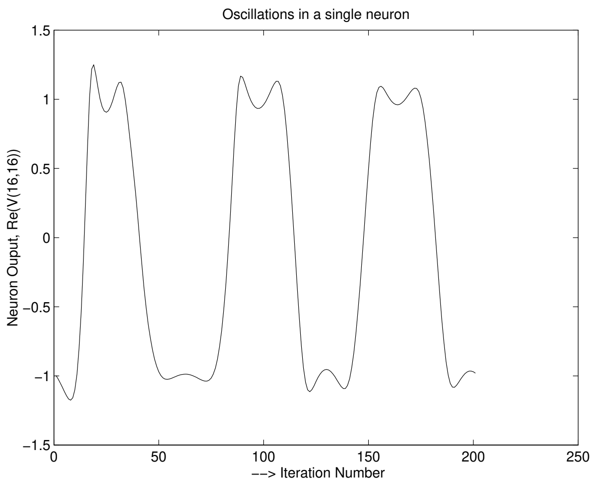

Figure 8: Output of neuron(16,16) for a sequence of 2.5 iterations.

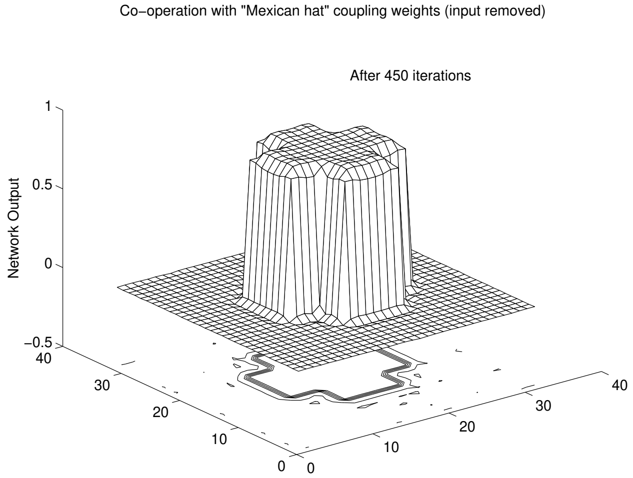

Figure 9: Network output (after 450 iterations) when the input image (noisy) is removed. Fixed "Mexican Hat" neighborhood connections are used.

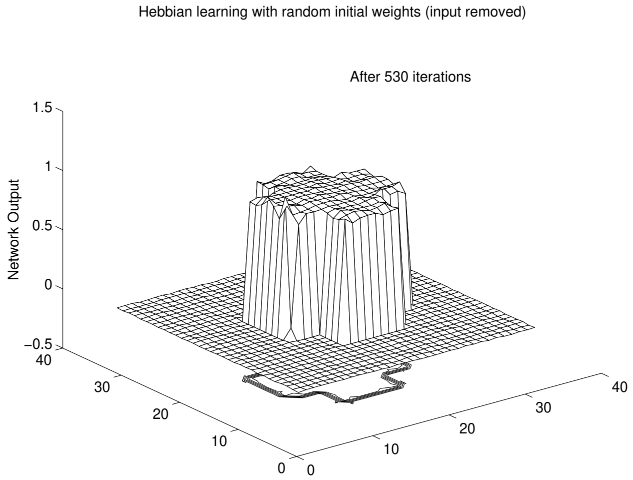

Figure 10: Network output (after 530 iterations) when the input image (noise corrupted) is removed and weight adaptation is continued.