ELSEVIER

Image and Vision Computing 20 (2002) 483–497

image AND vision COMPUTING

www.elsevier.com/locate/imavis

Image segmentation by relaxation using constraint satisfaction neural network

Fatih Kurugollu[a,b,*], Bülent Sankur[c], A. Emre Harmancı[d]

[a]School of Computer Science, The Queen's University of Belfast, 18 Malone Road, Belfast, N1 BT7 1NN, UK
[b]Marmara Research Center, Information Technologies Institute, Gebze, Kocaeli, Turkey
[c]Department of Electrical-Electronic Engineering, Bogazici University, Bebek, Istanbul, Turkey
[d]Department of Computer Engineering, Istanbul Technical University, Ayazaga, Istanbul, Turkey

Received 14 April 2000; received in revised form 19 December 2001; accepted 10 January 2002

Abstract

The problem of image segmentation using constraint satisfaction neural networks (CSNN) has been considered. Several variations of the CSNN theme have been advanced to improve its performance or to explore new structures. These new segmentation algorithms are based on interplay of additional constraints, of varying the organization of the network or modifying the relaxation scheme. The proposed schemes are tested comparatively on a bank of test images as well as real world images. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Image segmentation; Artificial neural networks; Constraint satisfaction problem; Multiresolution; Markov random fields

1. Introduction

Image segmentation, which aims to divide a given image into homogeneous and meaningful regions, is the crucial step in image processing since it directly affects the performance of subsequent operations such as image analysis, measurement, and classification. Despite the multitude of image segmentation methods proposed in the last three decades [1], the quest for new more effective methods continues. This is partly due to the necessity to handle as broad a category of images as possible, partly to meet the real-time demands in practical applications such as digital video.

We address in the paper constrained relaxation techniques coupled with parallel processing potential of neural networks for segmentation of images. In this context, Artificial neural networks (ANN) forms an effective tool to realize parallelism and real-time implementation [2–5].

There are several applications of ANNs in the image segmentation literature. One of the early applications of the use of ANNs in medical image segmentation was by Ozkan et al., who used the backpropagation learning to segment MR images [6]. Uchiyama and Arbib used competitive learning to cluster colors in images [7]. Littmann and Ritter developed an ANN, named local linear maps, for adaptive color segmentation and compared it with statistical methods [8]. A modular neural network has been used by Verikas et al. for image segmentation [9]. Shen and Ip used Hopfield type ANN by taking into account active surface optimization for segmentation [10].

Bayesian image segmentation has also profited from both the ANN's inherent parallelism and by relaxation techniques. Recall that Bayesian image segmentation constructs a mathematical model of the image and carries out the segmentation by maximizing the posterior probability according to this model. The most popular model is the Markov random field (MRF), which can be cast into an energy minimization formula. The minimization of the energy functional constitutes, in fact, the maximum a posteriori (MAP) segmentation of the image [11,12]. One roadblock in the MAP scheme is its computational complexity. However, the fact that the global energy functional can be decomposed into local interactions using the MRF model calls for the exploitation of the ANN parallelism. In this area Manjunath et al., proposed a Hopfield-type ANN that implements a deterministic relaxation for texture segmentation. In their work, to express the MAP functional, they used image intensity and texture label process based on Gauss Markov random field (GMRF) [13]. Another Hopfield type approach to implement MAP minimization is the deterministic relaxation neural network (DRNN) proposed by Raghu et al.

* Corresponding author. Address: School of Computer Science, The Queen's University of Belfast, 18 Malone Road, Belfast, N1 BT7 1NN, UK. Tel.: +44-28-9027-4628; fax: +44-28-9068-3890.
E-mail address: f.kurugollu@qub.ac.uk (F. Kurugollu).

PII: S0262-8856(02)00023-9

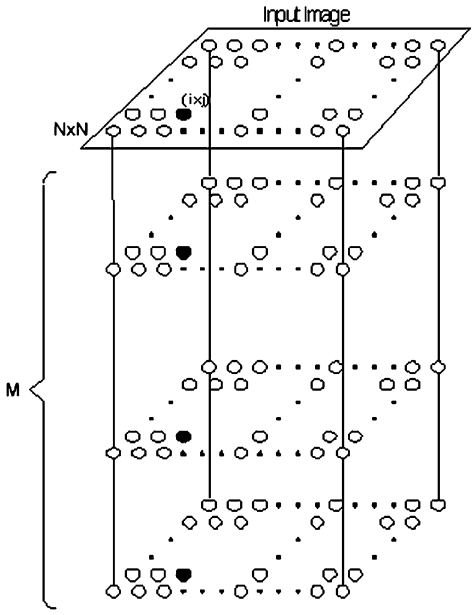

Fig. 1. Topology of the CSNN. Each layer represents a segment. The (i,j)th neuron in each layer holds the probability that (i,j)th pixel belongs to the segment represented by the layer.

[14,15]. The total energy is mapped in their work to a Hopfield model and a deterministic relaxation scheme is applied to obtain the MAP segmentation. Chen and Chen proposed a modified cellular neural network (CNN) architecture for MAP estimation task [16]. They combined the adaptive K-means clustering algorithm of Pappas [17], and the GMRF model. The MAP estimation problem has been carried out by using a combined method based on adaptive K-means algorithm and iterated conditional modes (ICM).

An interesting ANN-based approach to image segmentation is given by constraint satisfaction neural network (CSNN) as proposed by Lin et al. [18]. They cast image segmentation problem as a constraint satisfaction problem (CSP) and develop an ANN to solve the CSP. The method carries out the segmentation by initializing the neurons via some clustering scheme and later by running a relaxation on them, which takes into account the spatial constraints between labels. Although the CSNN is an attractive tool for image segmentation because of its simple network architecture and its flexible constraint scheme, it does have some of its own drawbacks. For example, the CSNN is observed to have a slow convergence rate, and to cause over-smoothing along segment boundaries.

In this paper, we introduce variations on the theme of CSNN-based segmentation. These variations aim to overcome the shortcomings of its segmentation performance and to advance new implementations of the CSNN. The proposed CSNN schemes can be classified into two categories:

- *Improver algorithms*: The goals in these algorithmic varieties are to accelerate the convergence and to improve the segmentation performance, especially on the segment boundaries. For example, while the basic structure of the CSNN is preserved some edge constraints can be imposed resulting in the Boundary CSNN algorithm (B-CSNN in Section 3.1) or a new neuron-scanning scheme is used as in the Multiscan CSNN algorithm (MS-CSNN in Section 3.2).
- *New structure algorithms*: In this category, the structure of the CSNN has been modified to implement the CSP for new applications. In Section 4.1, a pyramidal neural network is discussed, bringing in the capability to provide segmentation results at different scales. A CSNN structure, modified to incorporate data dependency term, is a design to solve the MRF estimation problem. (MRF-CSNN in Section 4.2).

The organization of the paper is as follows. In Section 2, the CSNN algorithm is briefly introduced and shortcomings of the algorithm is presented. In Section 3, two algorithms that improve CSNN in performance and speed attributes are introduced. The algorithms imposing new constraints and structures to CSNN are presented in Section 4. In Section 5, the claimed improvements are shown in terms of better convergence, segments that are more realistic and higher objective scores. Finally, the conclusions of the paper are given in Section 6.

2. The constraint satisfaction neural network

2.1. Review of the CSNN Structure

Image segmentation based on CSNN, as in Lin et al. [18], uses a network topology, as shown in Fig. 1, where links between nodes simply correspond to interactions between pixels. Furthermore, each layer represents one of the possible segments. Assuming for simplicity an $N \times N$ size image, each layer contains that many neurons, and neurons with the same coordinates in each layer hold the probability that the pixel belongs to the segment represented by the layer index. At the end of the relaxation, one would ideally expect the neuron weights to all converge to zero, except for the neuron of the winning layer (segment), which should converge to one. Connections between a pixel and its neighbors are shown in Fig. 2, where 8-neighborhood connectivity is shown, although the neighborhood connectivity order may be varied depending on application. The weights of these connections, which are assigned in a heuristic manner in Ref. [18], represent the constraints for a given topology. These weights are determined so that a neuron excites other neurons in the neighborhood representing the same label, and inhibits the ones that represent significantly distant labels.

For the relaxation to start the neurons must be initialized with labels resulting, for example, from some clustering process. In Ref. [18], a Kohonen self-organizing network

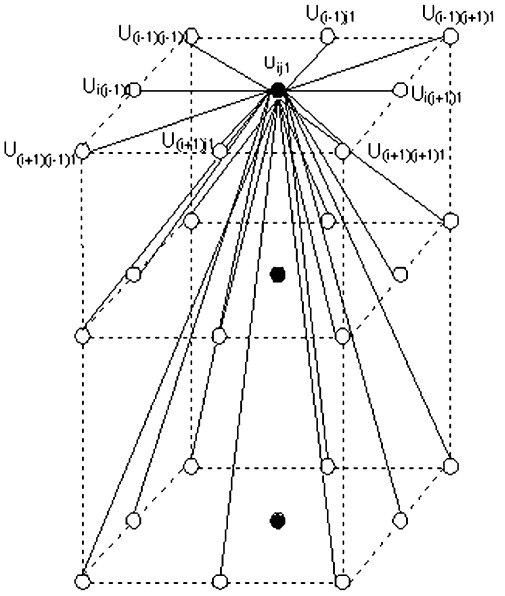

Fig. 2. Connections between a neuron O_{ijk}, $i,j = 1...N$, $k = 1...M$, and its neighbors. The weights of these connections are interpreted as constraints.

was used to produce the initial labels (i.e. probabilities of belonging to a segment). Accordingly, a larger probability value is given to the neuron of the winner class and lower probabilities are assigned to other classes. These probability values at any site should add up at a pixel site to 1.

After the initialization, CSNN is made to converge to a segmentation state, which satisfies all constraints through a parallel and iterative process. The label probabilities grow or diminish in a winner-take-all style because of contention between segments. The total squared difference of neuron outputs, called also the energy, E, between successive iteration steps is taken as the stopping rule as in Eq. (1), where O^t_{ijk} denotes the output value of the neuron in the spatial position i,j and in the kth segmentation layer at the iteration step t. As the CSNN converges, the neuron in the 'correct' layer approaches to 1 while the neurons in the other layers are reduced to 0. Finally, each pixel is assigned the class label of its winner layer.

$$E = \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{M}\left(O^{t+1}_{ijk} - O^{t}_{ijk}\right)^2 \tag{1}$$

The update scheme of a neuron is shown in Fig. 3. The output of the neuron, O_{ijk}, is determined both from its previous state and from the neighbor contributions, H_{ijk}. Neighbor contributions are summed from the outputs of the neighbor neurons through connection weights that represent the spatial constraints. The contribution of the neighborhood for O_{ijk} is given as follows:

$$H^t_{ijk} = \sum_{(q,r,l)\in N_{ij}} W_{ij,qr,k,l} O^t_{qrl} \tag{2}$$

where the state of the (q,r)th neuron in layer l, O_{qrl}, is reflected to the (i,j)th neuron in layer k, O_{ijk}, via a weighting factor, $W_{ij,qr,k,l}$. The non-linear updating 'the winner-take-all' rule is as follows:

$$O^{t+1}_{ijk} = \frac{\mathrm{Pos}\left(O^t_{ijk} + \Delta O^t_{ijk}\right)}{\sum_{l=1}^{M}\mathrm{Pos}\left(O^t_{ijl} + \Delta O^t_{ijl}\right)} \tag{3}$$

where

$$\Delta O^t_{ijk} = \begin{cases} \delta & \text{if } H^t_{ijk} = \max_l \{H^t_{ijl}\} \\ -\delta & \text{for all other } l \end{cases} \quad \text{and}$$

$$\mathrm{Pos}(x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

and δ, the updating step size, is a small positive number, typically chosen as 0.1. The denominator term in Eq. (3) is the normalization factor which keeps the total output of the neuron residing in a specific location less than 1. The

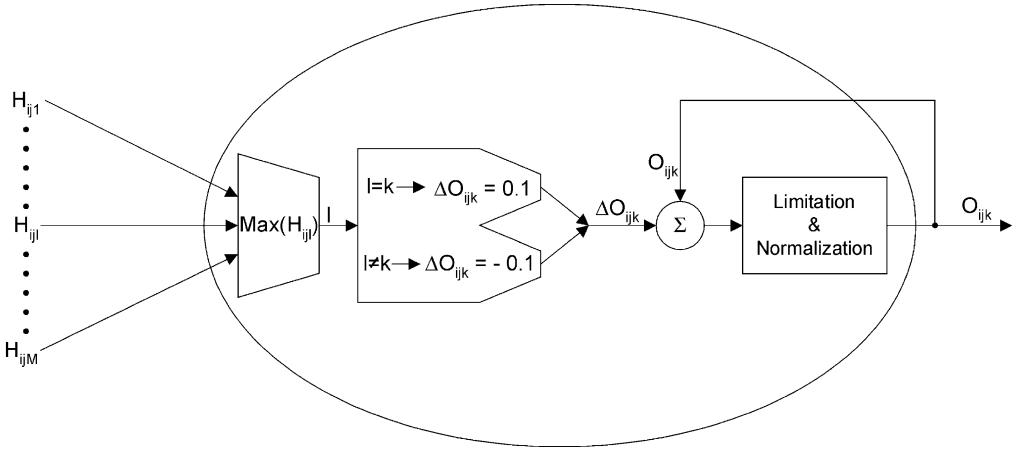

Fig. 3. Structure and update scheme of a neuron in CSNN.

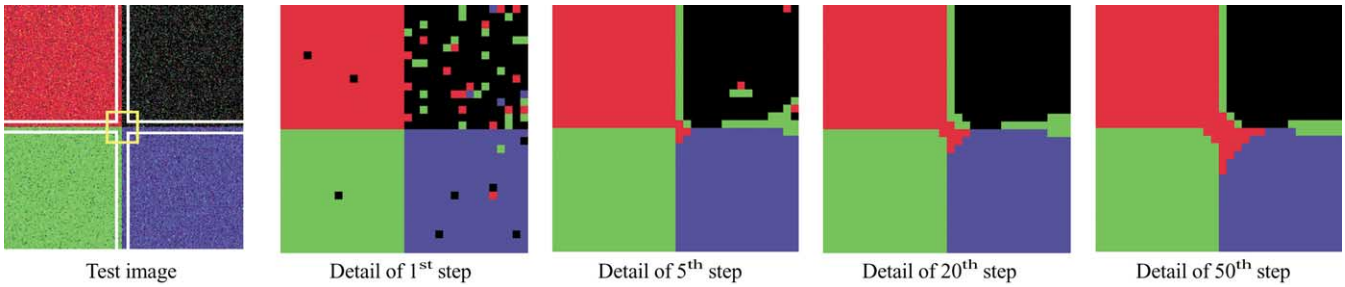

Fig. 4. The results of the CSNN segmentation at specific iteration steps. Test image consists of four quadrants with mean RGB value [0,0,0] [255,0,0] [0,255,0] [0,0,255] and noise with variance 64. To illustrate decision problem in the segment boundaries the region showed as the yellow square in the test image is used. The enlarged regions obtained from the segmentation results for the specific iteration steps are depicted.

updating rule, given in Eq. (3), is based on two principles:

1. If H_{ijk} is the maximum one among all contributions at location (i,j), a positive contribution is added to increase the support for the O_{ijk} neuron. This in turn causes an increase in the output of O_{ijk}.
2. For all other non-winning H_{ijm}, $m \neq k$, the outputs of the O_{ijm}th neuron, $m = 1,\ldots,M$, $m \neq k$, are decremented.

Let us denote the limiting value of the neuron function as $\lim_{t\to\infty} O_{ijk}^{t+1} = O_{ijk}$. Using these likelihood values of the (i,j)th pixel to belong to the kth segment the final assignment is made as pixel label $I_{ij} = \arg\max_k O_{ijk}$. Further details of the method used in determination of the weights and the mathematical construction of the CSNN can be found in Ref. [18]. The demonstration that CSNN actually performs probabilistic relaxation is presented in Appendix A.

2.2. Characteristics and shortcomings of CSNN

CSNN is a tool for image segmentation that combines the clustering operation in measurement space with the relaxation method in spatial domain. The first advantage of CSNN based segmentation is that the neural network structure provides an efficient and simple way to solve the CSP. The relaxation rate can be easily controlled by means of varying the connection weights that correspond to constraints or compatibility coefficients and neighborhood size around each pixel site. The behavior of the network, therefore, can be adjusted according to applications without any changes in the network structure. For example, smoother segments could be obtained by increasing the neighborhood size. Secondly, the CSNN solution to segmentation by relaxation enables parallel and real-time implementation [19].

One observed shortcoming of CSNN is that it has a long convergence time. Furthermore, the energy, given in Eq. (1), does not decrease below a certain value. This problem arises from the pixels on the segment boundaries, as the relaxation algorithm hesitates in the assignment of these pixels to one or the other segment, causing futile iterations. The segmentation results obtained from several iteration steps are shown in Fig. 4. While the CSNN-based relaxation works well for the pixels interior to the segment, it is not as efficient on the segment boundaries, as shown in Fig. 4. This can be seen more clearly in Fig. 5, where the energy of the non-boundary pixels is plotted separately from that of the boundary pixels. One can notice that while the within-segment energy goes rapidly to zero, the boundary energy remains at a standstill.

A second shortcoming, inherent in relaxation algorithms,

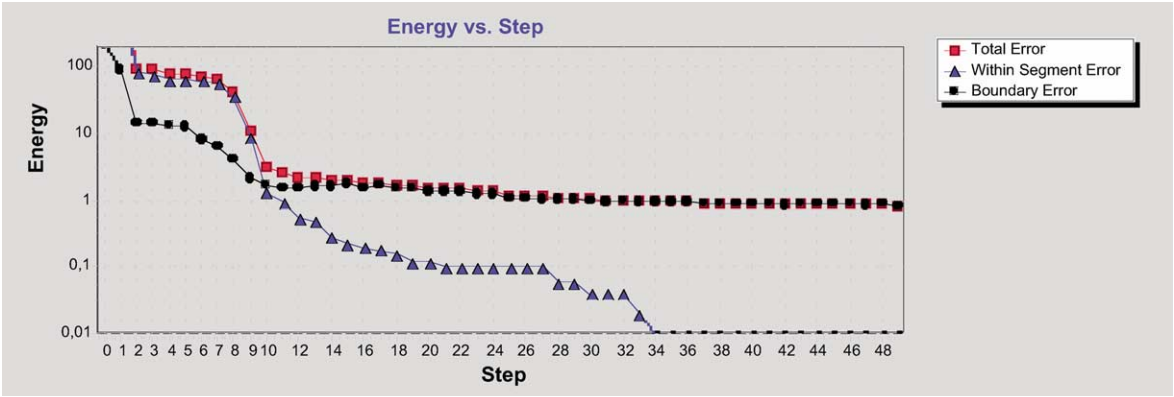

Fig. 5. Energy (E) vs. relaxation steps for a test image. Within-segment energy rapidly converges to zero whereas the boundary energy (the pixels that reside in the area covered by white lines in the test image in Fig. 4) remains constant after 35th step.

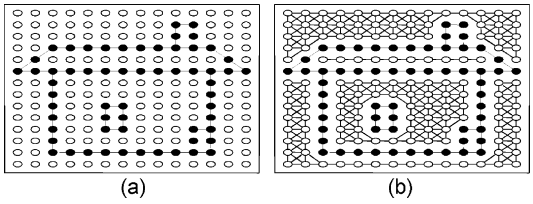

Fig. 6. Alteration of connection weights by using the edge map of the input image. The black dots indicate the edge pixels. Assigning 0 weight to their connection weights precludes the contradictory contributions across a boundary.

is that the CSNN algorithm tends to cause over-smoothing and under-segmentation results, as the big segments tend to capture small segments. This effect is shown in Figs. 14 and 15 where the details of the window segment on the 'House' image are absorbed by the neighborhood segments. Modified CSNN segmentation algorithms presented in the sequel address these shortcomings.

3. Schemes to improve CSNN

3.1. CSNN with boundary constraints: B-CSNN

Many region growing-based segmentation methods can be improved with the aid of edge information [20–23]. For example, Worth and Kennedy have proposed a boundary contour system to impose edge strength to the neuron connections by reducing or shutting off the weights of the connection [24].

It was pointed out that the convergence of CSNN was handicapped by the uncertainties at boundaries. Image edge information, incorporated in the connection weights of neurons, can be instrumental to improve the segmentation performance [25]. As depicted in Fig. 6 the neuron connections across the boundaries are severed by simply setting them to 0. So contributions of the pixels across a boundary, resulting in contradictory information, are precluded. In our work, we used Canny edge detector to guide the CSNN segmentation [26].

With the additional edge constraints imposed onto segmentation, both the numbers of iterations and the convergence error in the B-CSNN are reduced significantly as shown later in Fig. 10. In addition the absorption of small segments by their larger neighbors are prevented in the B-CSNN algorithm as shown in the details of the examples in Figs. 14 and 15.

3.2. Multiscanning segmentation with CSNN: MS-CSNN

In this approach, the structure and dynamics of the neural network is the same as in the CSNN, but the scanning pattern, hence the updating scheme, is changed. One commences with more sparsely sampled pixels but possessing larger neighborhoods, which leads to labels that are more reliable. In each subsequent scanning cycle, the remaining pixels (not yet updated) are visited in an increasingly denser pattern but with smaller neighborhoods as shown in Fig. 7. Previously visited pixels are left intact but they contribute to the updating of the newly visited pixels in their neighborhood. As each scanned pattern converges, one moves on to the next (denser) scanning pattern [27].

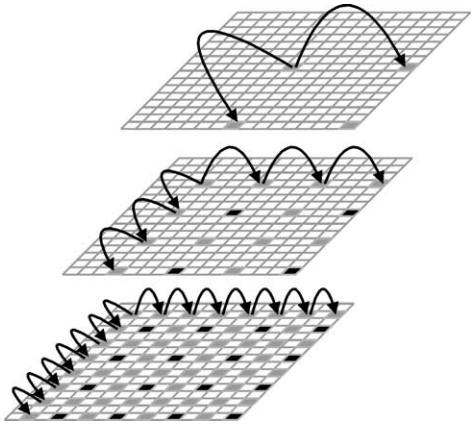

Fig. 7. Scan scheme in MS-CSNN where darker pixels indicate the visited ones. In the first stage, (top figure) the visited pixels are sparser but each is equipped with a larger neighborhood. In the following stages, (middle and bottom figures) pixels closer are visited albeit with smaller neighborhoods.

Since the outputs of the visited pixels in the first stages cannot be changed in the lower stages, the futile iterations are avoided in this algorithm. Therefore, this algorithm speeds up the process as seen in Fig. 10. It slightly improves the results according to the CSNN, as depicted in Figs. 14 and 15.

4. Schemes for new CSNN applications

4.1. Pyramidal CSNN segmentation: P-CSNN

Pyramidal CSNN (P-CSNN) aims to provide segmentation results at different scales. Such a tool is useful in applications requiring segmentation of the scene at different scales, typical examples being astronomical and industrial inspection images.

In the P-CSNN method, each resolution level of the pyramid consists of a label image at a different resolution level. Notice that one obtains not an image pyramid but a label pyramid. In this approach [28], a parent pixel in the upper level is related with its 16 child pixels in the lower level, while a child pixel in the lower level has 4 parent pixels in the upper level (Fig. 8). A voting among the child pixels is performed and the maximum encountered segment label of the child pixels is assigned to the parent pixel. Obviously, the label field, with the dimensions of the given image, constitutes the base of the pyramid (0th level). This label map is used to generate the 1st level map, the 1st level map bears the 2nd level label map, and so on. Each level

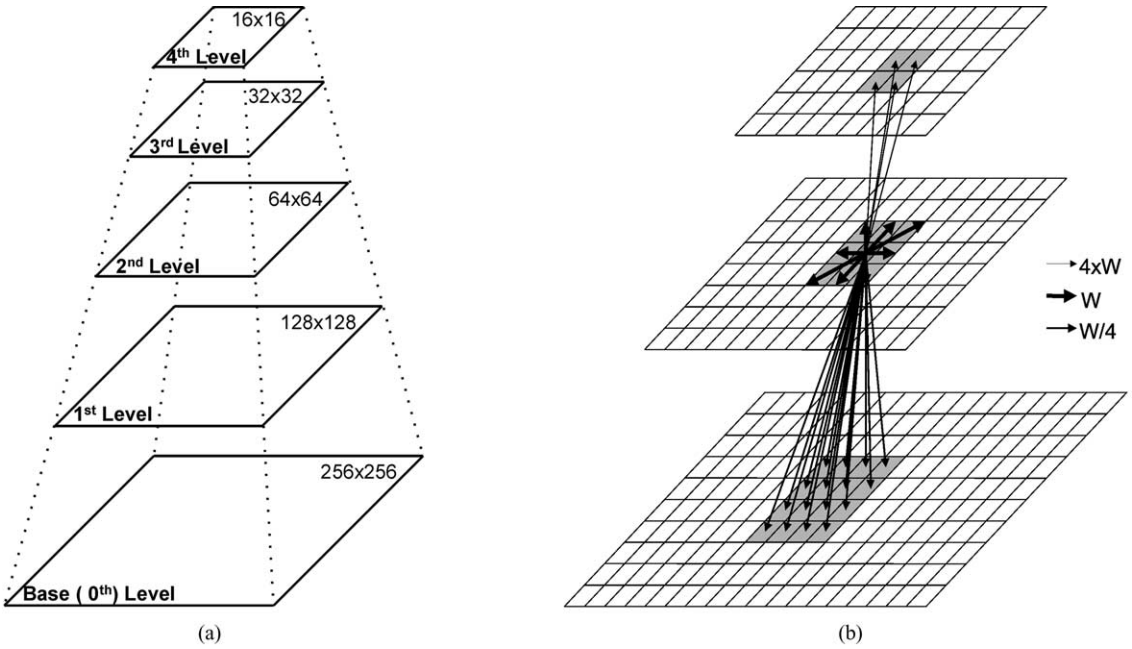

Fig. 8. (a) Structure of P-CSNN. Each level represents a CSNN, which consists of number of segments layers. (b) Connections between levels. These connection weights can be adjusted to obtain more smoothed or more detailed segmentation result. If W denotes the in-layer connection weights, in this figure child–father (from-low-to-high) weights were taken as $W/4$ while father–child (from-high-to-low) weights were taken $4W$. A numerical example can be found in Table 1.

possesses its own CSNN structure, based on interconnections between pixels as well as cross-connections between the one higher and one lower resolution levels [29] as shown in Fig. 8. These CSNNs differ from that described in Figs. 1–3 as they possess additional connections across resolution levels, and each one of them converges separately

After initialization, the neural network searches a segmentation result, which must satisfy all the constraints in all levels of the pyramid. To represent these constraints, each neuron is synapsing with (a) 16 child sites in the lower resolution level, (b) 4 parent sites in the higher resolution level, and (c) 8 neighboring sites in the same resolution level. When the algorithm has converged to a solution, the resulting segmentation is simply the state of the base level of the pyramid.

One can adjust the connection weights in Fig. 8 between the resolution levels to steer CSNN output to smoother or more detailed segmentation maps. For example, if larger connection weights are attributed from-low-to-high synapses smoother segments will be obtained, as lower resolution labels will have more roles. On the contrary, if larger connection weights are chosen from-high-to-low synapses, segments that are more detailed will be generated. A numerical example of this weight-assigning scheme is given in Table 1. In this example, the neighborhood parameter, p, and the number of segment, M, are chosen as 8 and 3, respectively. The weights, as determined by the ad hoc rule of Lin et al. [18], are given in Eq. (4).

$$W_{ij,qr,k,l} = \frac{1}{p}\left(1 - \frac{2|k-l|}{M}\right) \tag{4}$$

where p and M denote neighborhood size and number of segments, respectively.

The proposed P-CSNN method gives the user the flexibility to tune the algorithm for coarser or finer segmentation. A case in point is astronomical images, as shown in Fig. 16, where low-resolution segmentation is used to study the

Table 1
Assignment of connection weights. To obtain a fine segmentation, higher connection weights are assigned to the from-high-to-low synapses as compared to the from-low-to-high ones

From-high-to-low weights $W_{H-L} = 4W$					In-layer weights W					From-low-to-high weights $W_{L-H} = W/4$				
		k					k					k		
		0	1	2			0	1	2			0	1	2
	0	1/2	1/6	−1/6		0	1/8	1/24	−1/24		0	1/32	1/96	−1/96
1	1	1/6	1/2	1/6	1	1	1/24	1/8	1/24	1	1	1/96	1/32	1/96
	2	−1/6	1/6	1/2		2	−1/24	1/24	1/8		2	−1/96	1/96	1/32

morphology of the Nebula, and high-resolution segmentation to study details of its nucleus.

4.2. Markov random field segmentation via CSNN

The principle of MRF based segmentation is the maximization of the posterior estimate (MAP), that is finding the most probable label field satisfying certain spatial constraints given the observation of a degraded image [11,12,30]. This a posterior probability function, $p(I|y)$, is expressed, using Bayes theorem, in terms of a priori quantities, i.e.:

$$p(\text{label field}|\text{noisy image}) = \frac{p(\text{noisy image}|\text{label field})p(\text{label field})}{p(\text{noisy image})} \quad (5)$$

where $p(y|I)$ is the conditional probability of the observed image given a segmentation map with a priori label probability $p(I)$. The literature abounds with algorithms for this MAP estimation. For example, Geman and Geman used stochastic relaxation scheme for MAP estimation [30]. Hansen and Elliott used a suboptimal formulation of MAP and a dynamic programming approach, while Daily solved the problem using regularization theory [31,32]. The other stochastic optimization techniques such as simulated annealing, iterated conditional modes, maximizer of posterior marginals have also been used to estimate the MAP segmentation [12]. Among neural network-based solutions, one can mention the CNN alternative [16], the Hopfield type neural network [14] and the probabilistic neural network [33] solutions. Raghu et al., used a deterministic relaxation neural network (DRNN) on GMRF models [14,15]. Our CSNN-based MRF solution has been inspired from this DRNN, hence we will briefly review the basics of DRNN.

In the DRNN model, the energy to be minimized is derived from three processes: feature formation process, partition process and competition process. First, the feature formation process describes the probability of assigning a feature value to a pixel using the model parameters of each segment for that pixel. This probability is described as $P(G_{ij} = g_{ij}|L_{ij} = k)$ where G_{ij} is the realization of a feature vector at pixel location (i,j), and L_{ij} its corresponding label. Gabor filter-based features are used to model the texture and a Gaussian distribution is assumed for the feature formation process. This process expresses the data dependency term of the energy function. Second, the partition process defines the probability of the label of a pixel given the labels of the pixels in a predefined neighborhood of that pixel. This probability is defined as $P(L_{ij} = k|\ L_{qr},\ qr \in N_{ij})$ where N_{ij} denotes neighborhood of the pixel located in (i,j). This process describes the label continuity term of the energy function. The third process, the competition process, is imposed to prevent assigning multiple labels to any pixel. It is defined by the conditional probability of assigning a new label to an already labeled pixel. It is described as $P(L_{ij} = k|\tilde{L}_{ij})$ where $\tilde{L}_{ij}$ denotes the set of labels that may be assigned to that pixel.

In the light of these processes, they define a posterior probability which describes the labeling (L_{ij}) of the pixel given the feature measurement at the pixel (G_{ij}), the labels of the neighborhood pixels and the possible labels previously assigned to the pixel, as follows:

$P(L_{ij} = k|G_{ij}, L_{qr},\ qr \in N_{ij}, \tilde{L}_{ij})$. Using this posterior probability, the Bayes theorem, MRF energy descriptions for the probabilities and matching resulting Gibbs energy with the energy of the Hopfield ANN, Raghu et al., arrived at a DRNN scheme as given in Eq. (6). The details of the derivation steps can be found in Refs. [14,15].

$$\hat{H}^t_{ijk} = \sum_{(q,r,l)\in N_{ij}} \hat{W}_{ij,qr,k,l}\hat{O}^t_{qrl} - D^t\left(G_{ij}, \mu_{k,t}, \sigma^2_{k,t}\right) \quad (6)$$

where $\hat{D}^t_{ijk} = \|g_{ij} - \mu_{k,t}\|^2/(2\sigma^2_{k,t}) + (1/2)\ln[(2\pi)^M\sigma^2_{k,t}]$. $\mu_{k,t}$ and $\sigma^2_{k,t}$ are the model parameters that are defined as the mean value and the variance of the kth class or segment at the tth iteration step.

The neurons in the DRNN update their outputs according to their inputs in Eq. (6) as follows:

$$\hat{O}^t_{qrl} = \begin{cases} 1, & \text{if } \hat{H}^t_{ijk} > 0 \\ 0, & \text{if } \hat{H}^t_{ijk} < 0 \\ \hat{O}^t_{ijk} & \text{if } \hat{H}^t_{ijk} = 0 \end{cases} \quad (7)$$

while the initialization of the neurons is given in Eq. (8).

$$\hat{O}^0_{ijk} = \begin{cases} 1, & \text{if } k = \max_l P[G_{ij} = g_{ij}|L_{ij} = l] \\ 0, & \text{otherwise} \end{cases} \quad (8)$$

One can notice that the feedback and update scheme of the DRNN is quite similar to that of CSNN. One difference between the two algorithms is that the DRNN has a data dependency term in its energy function in addition to the label consistency term. The second difference is that DRNN can be considered a crisp version of the CSNN. In fact, in the CSNN algorithm presented earlier, the initialization and the updating rules reflect the probability of pixels to the segments, while DRNN has hard decisions. Thus in CSNN, in the iteration steps, the label value of a pixel changes gradually, while in the DRNN scheme, the assignments are hard, in that, the neuron of the winner segment takes value 1, and all the others take 0 value.

It was then conjectured that introducing a data dependency term to CSNN would prevent over-smoothing effect and its futile iterations around boundaries [34]. In addition the probabilistic relaxation of CSNN should allow for better utilization of data and neighborhood information as compared to DRNN where labels change by 1–0 jumps. Therefore, in the new MRF formulation of CSNN the input to a neuron becomes:

$$\bar{H}^t_{ijk} = H^t_{ijk} - D\left(y_{ij}, \mu_{k,t}, \sigma^2_{k,t}\right) \quad (9)$$

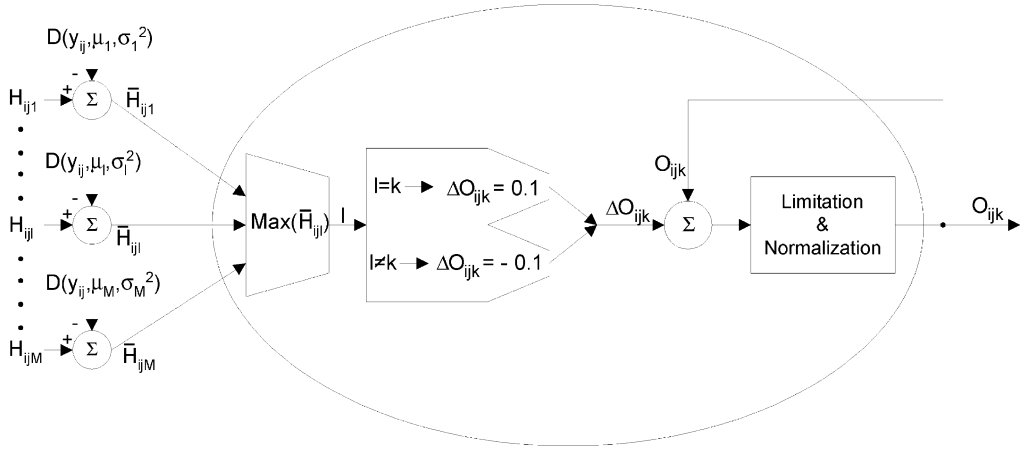

Fig. 9. Update scheme of the MRF-CSNN. Each neuron receives a signal from data dependency process as well as neighborhood contribution process.

where $D(y_{ij}, \mu_{k,t}, \sigma^2_{k,t})$ is as in Eq. (6). In our study the feature vector, y_{ij}, consisted of the RGB components of the ijth pixel, though the formulation allows for any feature. The new neuron update scheme is the modified version of the one in Fig. 3 and it is depicted in Fig. 9.

The new update mechanism of the MRF-CSNN needs to estimate the segment parameters, μ and σ at each iteration. To this effect, first the winner neuron for each pixel, possessing the largest likelihood value, is found. Then using the map of the winning neurons, the parameters are estimated as follows:

$$\mu_k = \frac{1}{|S_k|} \sum_{i,j \in S_k} y_{ij} \tag{10}$$

$$\sigma_k^2 = \frac{1}{|S_k|} \sum_{i,j \in S_k} \|y_{ij} - \mu_k\| \tag{11}$$

where $|S_k|$ denotes cardinality of the kth segment.

The performance of the MRF-CSNN is superior not only in convergence rate manner but also in segmentation result manner. The data dependency term helps to corroborate the correct segmentation decisions and prevent the algorithm from the futile iterations. The convergence rate of the MRF-CSNN is much better than the other algorithms as seen in Fig. 10. The performance on real world images is also better as shown in Fig. 14.

5. Results

In this chapter, the performance of the proposed CSNN varieties will be tested in three aspects: Comparison of their convergence rates, quantitative scores obtained from synthetic test images, and qualitative judgment of some real world images. We have excluded P-CSNN from the comparisons because of its application, hence multiresolution weight settings is domain-dependent and/or subjective.

5.1. Performance in convergence rates

The convergence rates of all the algorithms are shown in Fig. 10. It can be observed that, while the convergence of the CSNN remains at a standstill, for example after 30th iteration step, those of the B-CSNN, MRF-CSNN and MS-CSNN keep on improving. The improvement in B-CSNN is obviously due to the guidelines provided by the edge map; hence, uncertain labels around the edges get more firm decisions. The nature of convergence rate of

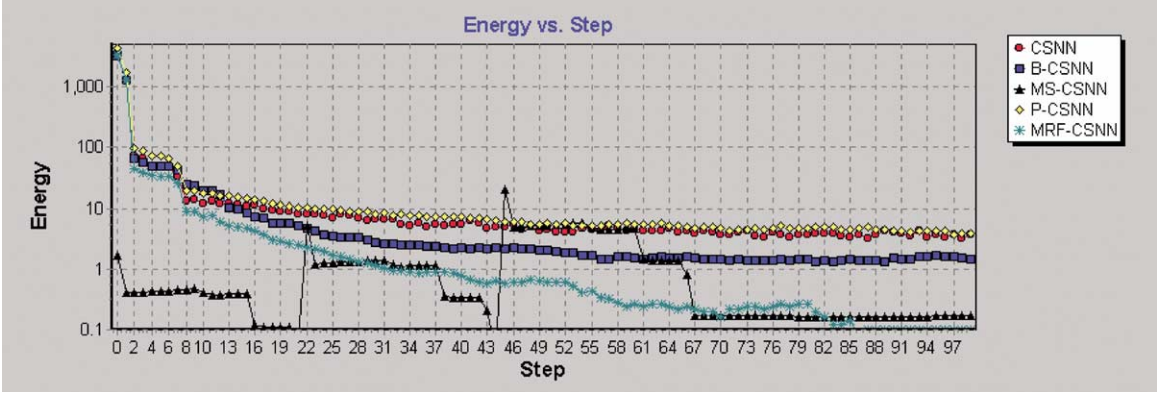

Fig. 10. The convergence error (Energy) with respect to the number of iterations of all the four algorithms for a 256–256 image.

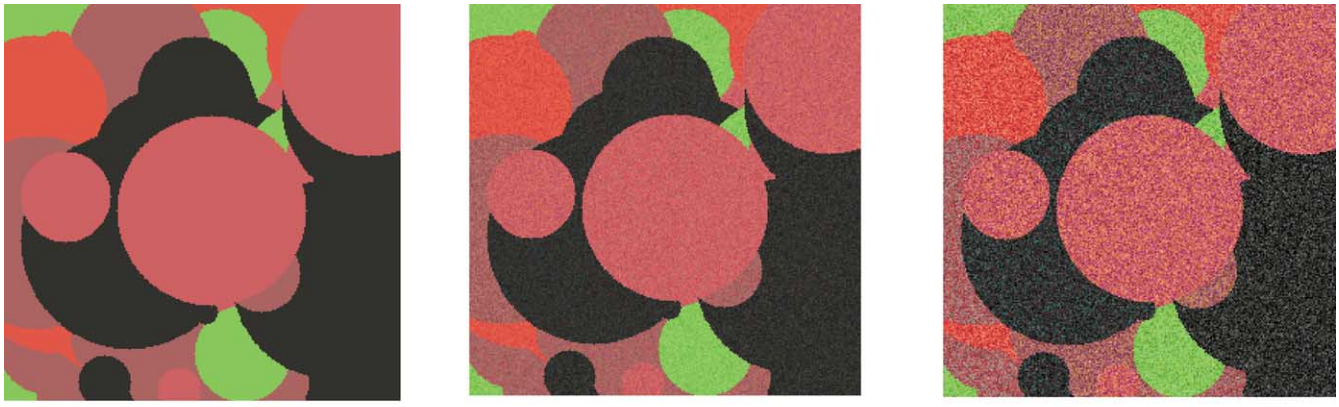

Fig. 11. Illustration of a ground-truthed test image used in the experiments. Gaussian colored synthetic test images with 22 regions and 5 segments. (a) Base image with five randomly selected colors from palette. (b) Image with the same mean colors as in (a) but with colors spread with $\sigma_R = 17$, $\sigma_G = 11$, $\sigma_B = 15$. (c) Image with the same mean colors as in (a) and but with colors spread with $\sigma_R = 31$, $\sigma_G = 34$, $\sigma_B = 37$. The associated classification difficulty parameter, λ, is given in the image captions.

the MS-CSNN is interesting, in that it has saccadic upswings as a resolution level is switched, since the pixels that have not updated yet start then being considered. The MS-CSNN algorithm converges overall more rapidly and to lower final value.

MRF-CSNN yields the best convergence rate. The energy falls off much more quickly. This is due to the inclusion of a data dependency term, so that the update of a pixel's likelihood to belong to a segment is affected not only by the neighboring levels, but also by the degree to which a pixel's feature fits into the feature cluster of that segment. Thus, much futile iteration is avoided in this algorithm, so that it converges more rapidly.

5.2. Quantitative results

To be able to compare segmentation algorithms in general and CSNN-based algorithms in particular, on an objective and quantitative basis, a test image generation procedure is developed [27] based on an idea from Stuller [36]. The procedure generates ground-truthed images with sufficient statistical richness. Briefly, test images consist of the superposition of circles, with random centers and radii, and random occlusions resulting in arbitrarily complex boundary patterns, as shown in Fig. 11. Furthermore each circle was stochastically painted, in other words, for each segment a Gaussian color palette was generated with judiciously selected RGB mean values and randomly selected standard deviations [σ_R, σ_G, σ_B]. By varying the spread of the color clusters around the segment mean (centroidal color) one can control the ratio λ of the between-class variance to the within-class variance. This ratio can be interpreted as a measure of classification difficulty. Two hundred test images were generated with different seed numbers. In these experiments the number of classes varied from two to six, and the standard deviation of the 'Gaussian clusters' in the color cube varied in the 10–40 range, for each of the RGB components [σ_R, σ_G, σ_B]. For example the image shown in Fig. 11 has 22 regions but 5 classes; each class was characterized by one of the 5 mean RGB vectors and color spread with $\sigma_R = 31$, $\sigma_G = 34$, $\sigma_B = 37$.

For performance evaluation, discrepancy-based criteria are used. The first criterion is the misclassification error (CE), originally proposed by Yasnoff et al. [37]. For N objects in the scene the expression becomes

$$\mathrm{CE}^{(k)} = 0.5\left(\frac{\sum_{i=1}^{N} C_{ik} - C_{kk}}{\sum_{i=1}^{N} C_{ik}} + \frac{\sum_{i=1}^{N} C_{ki} - C_{kk}}{\sum_{i=1}^{N}\sum_{j=1}^{N} C_{ij} - \sum_{i=1}^{N} C_{ik}}\right) \quad (12)$$

where C_{ij} is the ijth element of the confusion matrix, that is, the number of pixels that are assigned to object i, while it belonged to the jth object. To better reflect any deformation in the shape of the objects a second criterion is invoked. The boundary function (BF) metric is based on the turning angle function of the objects [35,38]. The rationale was that the CE criterion did not sufficiently reflect the deformation of the objects. The expression for BF for the kth object was:

$$\mathrm{BF}^{k} = \frac{\sum_{s=1}^{P} \left|\Theta_R^k(s) - \Theta_S^k(s)\right|}{P2\pi} \quad (13)$$

where P is the perimeter of an object, and $\Theta_R(s)$, $\Theta_S(s)$ are the turning angle functions of the reference and test images. Thus, these two metrics are complementary to each other. These two segmentation error penalties are each normalized to the range [0,1]. The performance of the whole image, called performance metric (PM), is the average of the CE and BF scores over all objects in the scene, that is, PM = 0.5(CE + BF)

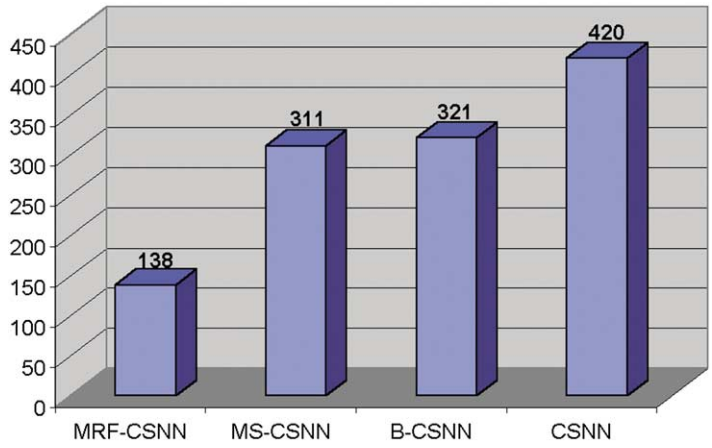

Fig. 12. Rank sums of the four segmentation algorithms tested over 200 images.

The performance of the proposed variations on the CSNN-based segmentation (namely, B-CSNN, MS-CSNN, MRF-CSNN) has been compared using 200 test images. Accordingly, the performance of the B-CSNN was found to be better than that of the CSNN for 137 test images out of 200 images, that is in 68% of the cases. Obviously the higher the edge content of an image, the more B-CSNN algorithm can show its advantage. The performance of the MS-CSNN has been found as 72%, that is, the MS-CSNN is better than the CSNN for 144 images out of 200 test images. Finally, the performance of the MRF-CSNN is the best among the others. It has produced results that are better than the CSNN for 161 images out of 200 test images, hence with a 80% superiority. Another way of showing this comparison is by plotting their rank scores. For each test image, the PM values of the methods are sorted, such that the best method has the highest rank (e.g. rank 0), while the method with the largest PM score has the lowest rank (rank 3). Thus, the method that has the smallest rank sum over all test images is considered the best one over all possible test images. The rank sums of the four methods are shown in Fig. 12. It is found that the MRF-CSNN has the lowest (best) rank score while the CSNN has the worst (highest) one.

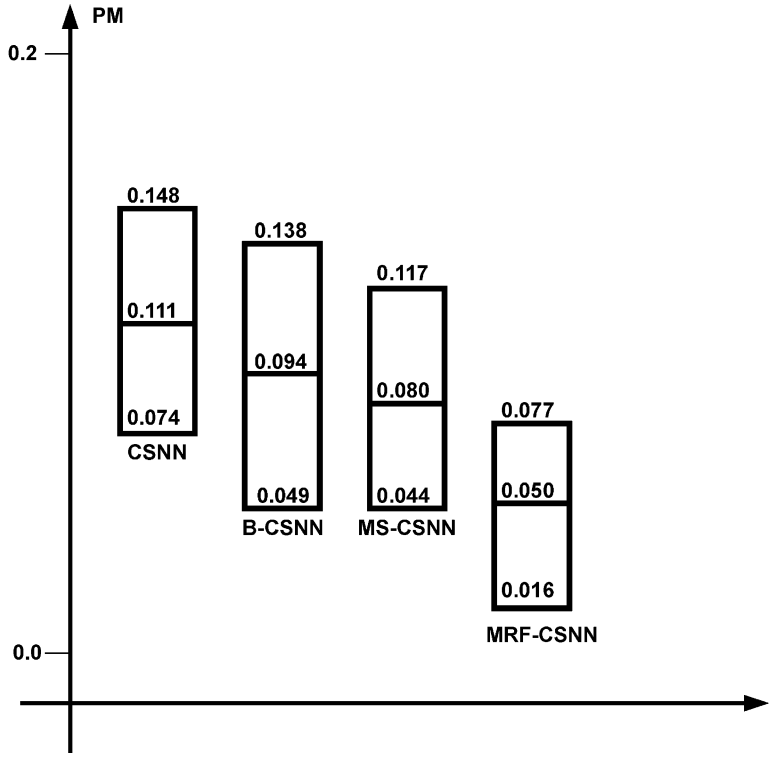

Fig. 13. Box-plots of the PM scores of the segmentation algorithms. Scores obtained from the 200 test images where the number of segments vary from 2 to 6 and where four color-spread categories (σ_R, σ_G, σ_B values) are used.

Fig. 14. Segmentation results of the CSNN based methods. The numbers of segments used in the methods are given under the original image.

Box plots offer an alternative way to compare the location and statistical spread of the performance scores. In Fig. 13, we have plotted the upper and lower 25% percentiles of the PM scores around the median value. The ranking in the box scores confirms the earlier observations. One can note again the distinct superiority of the MRF-CSNN vis-à-vis all the other CSNN algorithms.

5.3. Qualitative results

In this section, the real world image performances of all algorithms are presented. First, the number of segments must be furnished as a priori information to apply the algorithms to the real world images. Determining the number of segments from a given image is, in fact, a requirement in most other segmentation schemes. Often this problem is heuristically solved by means of a cluster validity procedure. Among several cluster validity algorithms, a modified version of the AIC (Akaike Information Criterion) criterion, proposed by Zhang and Modestino [39], which is based on a model fitting approach has been chosen [35].

Results obtained from real world images by using CSNN, B-CSNN, MS-CSNN and MRF-CSNN methods are shown in Fig. 14. It can be noticed that segmentation boundary noise and absorption of the small segments by their larger neighbors are avoided in the B-CSNN and the MRF-CSNN algorithms. The MS-CSNN algorithm, however, improves only slightly the aforementioned problems. This is in contrast to CSNN that produces under-segmentation results as small segments are captured by the big ones. Several test cases in Fig. 15 illustrate the global and the detail performance of the segmentation.

Finally, P-CSNN behavior is illustrated by an application to an astronomical image as shown in Fig. 16. This image, obtained from the NASA web page of Astronomy Picture of the Day,[1] is of M57, Ring Nebula. It is actually a barrel-shaped cloud of gas shrugged off by a dying central star. This image has been produced from Hubble Space Telescope observations using natural appearing colors to indicate the temperature of the stellar gas shroud. Hot blue gas near the energizing central star gives way to progressively cooler green and yellow gas at greater distance with the coolest red gas along the outer boundary. Dark, elongated can also be seen near the nebula's edge.

To extract the whole structure of the nebula a coarse segmentation scheme can be used. If one needs to segment each gas shroud, a fine segmentation scheme should be used. In this example, the fine segmentation result has been obtained by choosing high connection weights that are attributed to low-to-high level synapses whereas a high connection weights for high-to-low level synapses have been assigned to obtain the coarse segmentation result.

[1] http://antwrp.gsfc.nasa.gov/apod/ap010729.html.

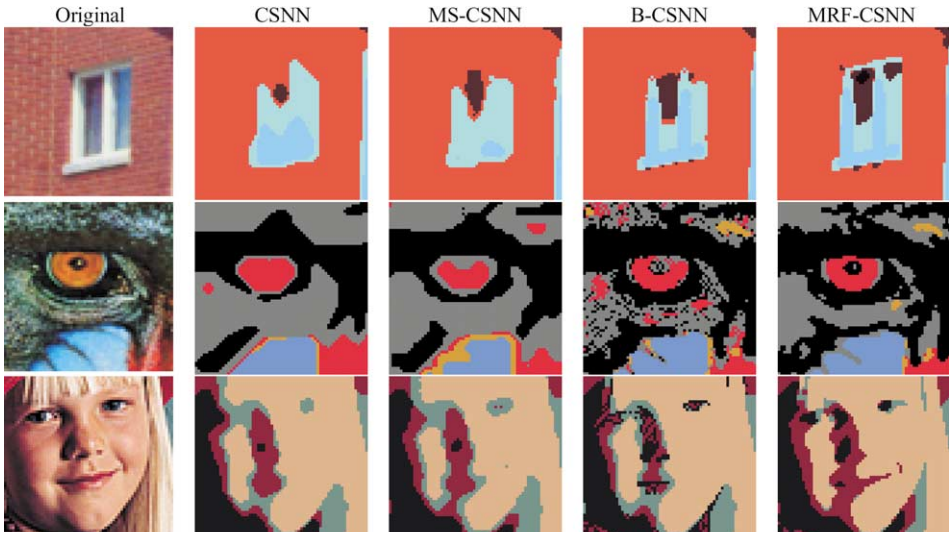

Fig. 15. Details of the segmentation results.

6. Conclusions

In this study, various innovations of the CSNN segmentation algorithm have been described and tested. The new algorithms are based on the introduction of new constraints and/or modification of the relaxation scheme.

The pyramidal version of the CSNN relaxation gives a handle to the user to adjust the desired local detail or global morphology of the segmentation. If edge information is available or if the segmentation accuracy along the boundaries is paramount, then the edge-guided B-CSNN algorithm can be used. Finally, rather than relying purely on the initial label assignment, but having a chance to consult the original data features, as in the MRF-CSNN scheme, gives the overall best performance among the CSNN varieties.

Acknowledgements

The authors would like to thank the anonymous reviewer for constructive criticisms that helped much to improve the paper.

Appendix A. Relationship between the CSNN and the probabilistic relaxation

We want to show that CSNN-based segmentation is in fact a realization of probabilistic relaxation. It will be useful to review comparatively the steps of the CSNN-based algorithm and those of a probabilistic relaxation from the literature, as detailed, for example, in Ref. [40]. Recall that in the probabilistic relaxation method, a probability or confidence vector $p_{ij} = [p_{ij}(1), \ldots, p_{ij}(N)]^{\mathrm{T}}$, is assigned to each pixel, each of its elements denoting the probability of that pixel belonging to the corresponding segment. Obviously these probabilities sum to one for each pixel, that is $\sum_{i=1}^{N} p_{ij}(i) = 1$.

1. *The initialization step*: In both the CSNN scheme and the probabilistic relaxation the initial weights are assigned by some clustering technique, such as histogram thresholding, fuzzy c-means, Kohonen's SOM etc.
2. *Compatibility functions vs. connection weights*: The idea behind probabilistic relaxation is to reveal compatibility between neighboring pixels, hence try to give them the same segment label. This is carried out by using compatibility functions and the update rules of the confidence vectors. The compatibility between pixels l and k is measured as:

$$r(k,l) = \begin{cases} < 0 & \text{Segments } S_k \text{ and } S_l \text{ are incompatible} \\ = 0 & \text{Segments } S_k \text{ and } S_l \text{ are independent} \\ > 0 & \text{Segments } S_k \text{ and } S_l \text{ are compatible} \end{cases} \tag{A1}$$

   The specific form of the compatibility function either is known a priori or can be estimated from the initialization process. On the other hand, in CSNN, this compatibility control is carried out by using connection weights, which impose constraints on the label map. The layers of the CSNN, which represent the segments, are ordered to reflect these constraints. Thus, CSNN use compatibility scheme of the probabilistic relaxation thorough connection weights. Thus, formally they both aim for energy and incompatibility relaxation, although they may differ in the specific form of the functions.
3. *Update rules*: In both the CSNN and the probabilistic relaxation a competition and cooperation process runs until a steady state is reached. In the probabilistic

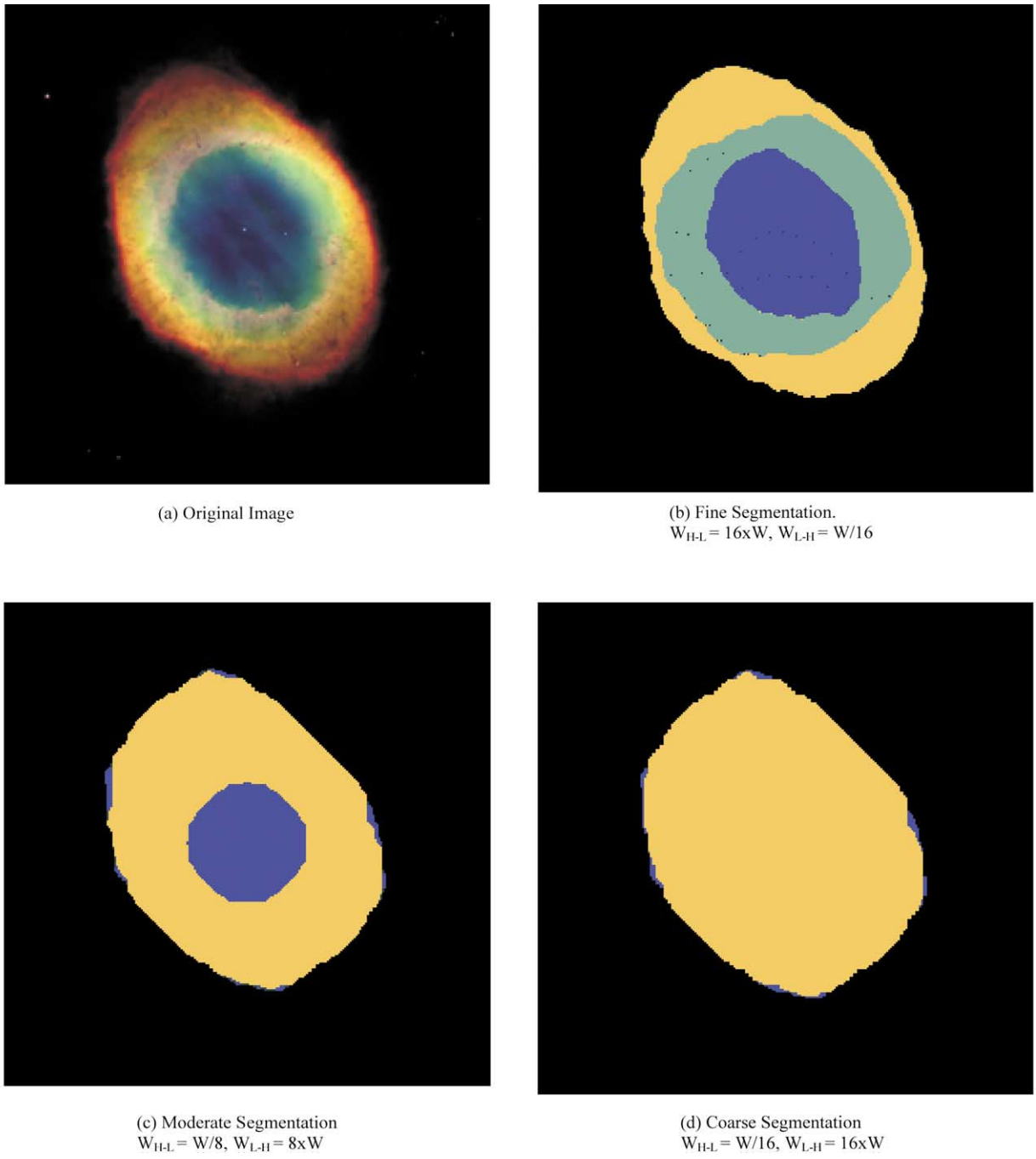

Fig. 16. P-CSNN segmentation in astronomy. W denotes the between-layer connection weights. W_{H-L} and W_{L-H} represent form-high-to-low and from-low-to-high connection weights, respectively.

relaxation the probability updating at step n is as follows:

$$\Delta p_{ij}^{(t)}(k) = \sum_{m,n,\epsilon Nij} d_{ij,m}\left[\sum_{l=1}^{N} r_{ij,mn}(k,l)p_{mn}^{(t)}(l)\right] \tag{A2}$$

where the weight $d_{ij,mn}$ represent the relevance of the contribution of a neighborhood pixel at position (m,n) to the pixel at position (i,j). The total for this parameter over all neighborhoods must be equal to one. In Eq. (A2) one observes that the amount of change in the probability for a specific segment is calculated by using the current probabilities of the neighbor pixels and the compatibility between the segments. The neighborhood pixel contributions is integrated into the pixel probability via the formula:

$$p_{ij}^{(t+1)}(k) = \frac{p_{ij}^{(t)}(k)\left(1 + \Delta p_{ij}^{(t)}(k)\right)}{\sum_{l=1}^{N}\left(p_{ij}^{(t)}(l)\left[1 + \Delta p_{ij}^{(t)}(l)\right]\right)} \tag{A3}$$

where the denominator is the normalizing factor. The CSNN update rule in Eqs. (2) and (3) in the text is reminiscent of the formula in Eq. (A3). The main difference between the CSNN and the probabilistic update schemes is that in the probabilistic relaxation, this change is allowed to be positive for more than one segment, while in the CSNN there is only one positive change, all the others having to be negative. This yields the speed up in the convergence of the CSNN.

Since both methods use the same formalism of label initialization, pixel interaction and label updating it is concluded that the CSNN performs a variety of the probabilistic relaxation.

References

[1] N.R. Pal, S.K. Pal, A review on image segmentation techniques, Pattern Recognition 26 (1993) 1277–1294.

[2] G. Minchin, A. Zaknich, A design for FPGA implementation of the probabilistic neural network, Sixth International Conference on Neural Information Processing, Perth, WA, IEEE, 2, 1999 pp. 556–559.

[3] J. Kowalski, M. Strzelecki, A VLSI Kohonen neural network chip and its basic measurement results. International Symposium Nonlinear Theory and its Applications, (NOLTA. 2000), Dresden, Germany, 2000, September, 17–21.

[4] L. Yang, L.O. Chua, K.R. Krieg, VLSI implementation of cellular neural networks, Proceedings of IEEE International Symposium on Circuits and Systems, (ISCAS'90), 1990, pp. 2425–2427.

[5] K.R. Perko, I. Fajfar, T. Tuma, J. Puhan, Low-cost, high-performance CNN simulator implemented in FPGA, Proceedings of IEEE International Workshop on Cellular Neural Networks and Their Applications, (CNNA'2000), 2000, Catania, pp. 277–282.

[6] M. Ozkan, B.M. Dawant, R.J. Maciunas, Neural-network-based segmentation of multi-modal medical images: a comparative and prospective study, IEEE Transactions on Medical Imaging 12 (1993) 534–544.

[7] T. Uchiyama, M.A. Arbib, Color image segmentation using competitive learning, IEEE Transactions on PAMI 16 (1994) 1197–1206.

[8] E. Littmann, H. Ritter, Adaptive colour segmentation—a comparison of neural and statistical methods, IEEE Transactions on Neural Network 8 (1997) 175–185.

[9] A. Verikas, K. Malmquist, L. Bergman, Colour image segmentation by modular neural network, Pattern Recognition Letters 18 (1997) 173–185.

[10] D. Shen, H.H.S. Ip, A Hopfield neural network for adaptive image segmentation: an active surface paradigm, Pattern Recognition Letters 18 (1997) 37–48.

[11] R.C. Dubes, A.K. Jain, Random field models in image analysis, Journal of Applied Statistics 16 (1989) 131–162.

[12] S.Z. Li, Markov Random Field Modeling in Computer Vision, Springer, USA, 1995.

[13] B.S. Manjunath, T. Simchony, R. Chellappa, Stochastic and deterministic networks for texture segmentation, IEEE Transaction on ASSP 38 (1990) 1039–1049.

[14] P.P. Raghu, R. Poongodi, B. Yegnanarayana, Unsupervised texture classification using vector quantization and deterministic neural network, IEEE Transaction on Image Processing 6 (1997) 1376–1387.

[15] P.P. Raghu, B. Yegnanarayana, Segmentation of Gabor-filtered textures using deterministic relaxation, IEEE Transactions on Image Processing 5 (1996) 1625–1635.

[16] C.W. Chen, L. Chen, Cellular neural network architecture for Gibbs random-field-based image segmentation, Journal of Electronic Imaging 7 (1998) 45–51.

[17] T. Pappas, An adaptive clustering algorithm for image segmentation, IEEE Transaction on Signal Processing 40 (1992) 901–914.

[18] W.C. Lin, E.C. Tsao, C.T. Chen, Constraint satisfaction neural networks for image segmentation, Pattern Recognition 25 (1992) 679–693.

[19] D. Abramson, K. Smith, P. Logethetis, D. Duke, FPGA based implementation of a Hopfield neural network for solving constraint satisfaction problems, Proceedings EuroMicro Workshop on Computational Intelligence, 1998, pp. 688–693.

[20] J. Fan, D.K.Y. Yau, A.K. Elmagarmid, W.G. Aref, Automatic image segmentation by integrating color-edge extraction and seeded region growing, IEEE Transaction on Image Processing 10 (2001) 1454–1466.

[21] C.C. Chu, J.K. Aggarwal, The integration of image segmentation maps using region and edge information, IEEE Transaction on PAMI 15 (1993) 1241–1252.

[22] T. Pavlidis, Y. Liow, Integration region growing and edge detection, IEEE Transaction on PAMI 12 (1990) 225–233.

[23] D. Zugaj, V. Lattuati, A new approach of color images segmentation based on fusion region and edge segmentation outputs, Pattern Recognition 31 (1998) 105–113.

[24] A.J. Worth, D.N. Kennedy, Segmentation of magnetic resonance brain images using analogue constraint satisfaction neural networks, Image and Vision Computing 12 (1994) 345–354.

[25] F. Kurugöllü, S. Birecik, M. Sezgin, B. Sankur, Image segmentation based on boundary constraint neural network, Proceedings of Third International Workshop on Image and Signal Processing on the Theme of Advances in Computational Intelligence, Manchester, UK, November 4–7, 1996, pp. 353–356.

[26] J.F. Canny, A computational approach to edge detection, IEEE Transaction on PAMI 8 (1986) 679–698.

[27] F. Kurugöllü, B. Sankur, Image segmentation based on multi-scan constraint satisfaction neural network, Pattern Recognition Letter 20 (1999) 1553–1563.

[28] P.J. Burt, T.H. Hong, A. Rosenfeld, Segmentation and estimation of image region properties through cooperative hierarchical computation, IEEE Transactions On SMC 11 (1981) 802–809.

[29] F. Kurugöllü, B. Sankur, Color cell image segmentation using pyramidal constraint satisfaction neural network, Proceedings of IAPR Workshop on Machine Vision Applications, Makuhari-Chiba, Japan, November 17–19, 1998, pp. 85–88.

[30] S. Geman, D. Geman, Stochastic relaxation Gibbs distributions, and the Bayesian restoration of images, IEEE Transaction on PAMI 6 (1984) 721–741.

[31] F.R. Hansen, H. Elliot, Image segmentation using simple Markov field models, Computer Graphics and Image Processing 20 (1982) 101–132.

[32] M.J. Daily, in: Y. Mandavieh, R.C. Gonzales (Eds.), Color Image Segmentation, in Advances in Image Analysis, Society of Photo Optical, USA, 1992.

[33] P.P. Raghu, B. Yegnanarayana, Supervised texture classification using a probabilistic neural network and constraint satisfaction model, IEEE Transaction on Neural Networks 9 (1997) 516–522.

[34] F. Kurugöllü, B. Sankur, MAP segmentation of color images using constraint satisfaction neural network, IEEE International Conference on Image Processing, Kobe, Japan, October 24–28 1999.

[35] F. Kurugöllü, Color image segmentation: multithresholding and constraint satisfaction methods, PhD Dissertation, Istanbul Technical University, Control and Computer Engineering Department, Istanbul, Turkey, 1999.

[36] J. Stuller, R. Shah, An image model based on occluding object images

and maximum entropy, IEEE Transaction on Image Processing 7 (1998) 1300–1310.
[37] W.A. Yasnoff, J.K. Mui, J.W. Bacus, Error measures for scene segmentation, Pattern Recognition 9 (1977) 217–231.
[38] E.M. Arkin, L.P. Chew, D.P. Huttenlocker, K. Kedem, J.S.B. Mitchell, An efficient computable metric for comparing polygonal shapes, IEEE Transaction on PAMI 13 (1991) 209–215.
[39] J. Zhang, J.W. Modestino, Model-fitting approach to cluster validation with application to stochastic model-based image segmentation, IEEE Transaction on PAMI 12 (1990) 1009–1017.
[40] I. Pitas, Digital Image Processing Algorithms and Applications, Wiley, USA, 2000.