Astronomical Object Recognition by means of Neural Networks.

R. Tagliaferri
DMI - Università di Salerno, 84081 Baronissi (Salerno)
and I.N.F.M. Unità di Salerno, 84081 Baronissi (Salerno)
and IIASS "E. R. Caianello", 84019 Vietri s/m (Salerno)
Email: robtag@dia.unisa.it

G. Longo, S. Andreon, S. Zaggia
Osservatorio Astronomico di Capodimonte, via Moiariello 16, 80131 Napoli

N. Capuano, G. Gargiulo
Facoltà di Scienze, Università di Salerno, 84081 Baronissi (Salerno)

Abstract

Astronomical wide field imaging deals with Terabytes data sets and requires new strategies for data reduction and analysis. We discuss here the application of different types of neural nets to the detection and extraction of celestial objects. Preliminary tests show that neural nets are more effective than traditional techniques.

1. Introduction

Wide field imaging and hence sky surveys are used in all fields of observational astronomy and cosmology [1]. Large format photographic plates such as those used for Schmidt telescopes or panoramic (larger than 8kx8k) CCD detectors contain up to 10^6 celestial objects and a typical survey consists of hundreds or thousands of such individual plates or frames. The processing of such an humongous amount of data (in the pixel-space) cannot be handled with the traditional interactive data reduction techniques and calls for A.I. (Artificial Intelligence) based procedures capable to push as far as possible the automatic extraction of useful information from the digitized data. In order to build catalogues of object parameters we need first to detect the objects against the (noisy) background and then to measure some properties of the detected objects such as, for instance, the degree of spatial resolution (point-like or extended objects) geometrical (elongation, position angle of the major axes of the best fitting ellipse, etc.), morphological (type) and photometric (total and isophotal magnitudes, color indexes, etc.). In what follows we shall describe the application of a specifically tailored Neural Net (NN) package to the segmentation of astronomical images, *id est* to the detection of objects against a noisy background. The raw material used for the testing are Schmidt plates obtained at the European Southern Observatory (ESO) Schmidt telescope in the R photometric band. The useful field of each plate covers 5 x 5 sq. deg. of the sky. The plates were digitized using the Perkin & Elmer PDS Microdensitometer at the

Astronomical Observatory of Capodimonte in Naples and calibrated through calibration spots taken simultaneously to the scientific exposition.

2. PCA Neural Nets

Principal Component analysis (PCA) is a widely used technique in data analysis. Mathematically, it is defined as follows: let $\mathbf{C}=E(xx^T)$ be the covariance matrix of L-dimensional zero mean input data vectors x. The i-th principal component of x is defined as $x^T\,\mathbf{c}(i)$, where $\mathbf{c}(i)$ is the normalized eigenvector of $\mathbf{C}$ corresponding to the i-th largest eigenvalue $\lambda(i)$.

The subspace spanned by the principal eigenvectors $\mathbf{c}(1), \ldots, \mathbf{c}(M)$, $(M<L)$ is called the PCA subspace (of dimensionality M) [2], [3]. PCA's can be neurally realized in various ways [4], [5], [6], [2], [7], [8]. The PCA network used by us is a one layer feedforward neural network which is able to extract the principal components of the stream of input vectors. Typically, Hebbian type learning rules are used, based on the one unit learning algorithm originally proposed by Oja [6]. Many different versions and extensions of this basic algorithm have been proposed during the recent years (see [9], [10], [3], [8]). The structure of the PCA NN can be summarized as follows: there is one input layer, and one forward layer of neurons totally connected to the inputs; during the learning phase there are feedback links among neurons, that classify the network structure as either hierarchical or symmetric. After the learning phase the network becomes purely feedforward. The hierarchical case leads to the well known GHA algorithm [8], [10]; in the symmetric case we have the Oja's subspace network [6]. PCA neural algorithms can be derived from optimization problems, such as variance maximization and representation error minimization. We can generalize these problems to nonlinear problems, getting nonlinear algorithms (and relative networks). These have the same structure of the linear ones: either hierarchical or symmetric. These learning algorithms can be further classified in: robust PCA algorithms and nonlinear PCA algorithms [9], [10]. We define robust PCA so that the objective function grows less than quadratically. The non linear learning function appears at selected places only. In nonlinear PCA algorithms all the outputs of the neurons are nonlinear function of the responses. We have seen both in preceding experiments [11] and in this paper that the hierarchical robust NN reaches the best performance. Specifically, in the experiments we compared hierarchical robust NN with learning function $g(t)= \tanh(\alpha x)$ with linear PCA.

3. Neural nets for image segmentation

For the segmentation we used Hierarchical and Hybrid NNs. The former is a Multilayer Unsupervised NN, while the latter is structured in two layers: the first one being an unsupervised neural net and the second a clustering algorithm. Aim of both setups is to attain a number of elements equal to the number of classes in which we want to segment the input image. In the following sections we shall describe first the neural models and the clustering algorithms and hence the hierarchical and hybrid networks.

3.1 Unsupervised neural nets

Kohonen [12],[13] Self Organizing Maps (SOM) are composed by a neuron layer structured in a rectangular grid. When a pattern *x* is presented to the net each neuron *i* receives the components and computes the distance d_i from its weight vector w_i. The unit which has the minimum distance from the input pattern will be the winner. The adaptation step consists in the modification of the weights of the neurons in the following way:

$$w_i^{(t+1)} = w_i^{(t)} + e^{(t)} \cdot h_{s^{(t)}}\left(d(i,k)\right) \cdot \left(x - w_i^{(t)}\right)$$

where $e^{(t)}$ is a *gain term* $(0 \le e^{(t)} \le 1)$ decreasing in time, $h_{s^{(t)}}(x)$ is a unimodal function with variance $s^{(t)}$ decreasing with x and $d(i,k)$ is the distance in the grid between the *i* and the *k* neurons.

The Neural-Gas NNs have a learning algorithm [14] which works better than the preceding one: in fact, it is quicker and it reaches a lower average distortion value[1]. It uses a soft-max adaptation of the weights and it classifies the neurons in an ordered list $(i_1, i_2, \ldots, i_m)$ following their distance form the input pattern. The weight adaptation depends on the position $rank(i)$ of the *i* neuron in the list in the following manner:

$$w_i^{(t+1)} = w_i^{(t)} + e^{(t)} \cdot h_{s^{(t)}}\left(rank(i)\right) \cdot \left(x - w_i^{(t)}\right).$$

The algorithm applies the gradient descent technique to the error function:

$$E_{ng} = \frac{1}{2C(s)} \sum_{j=1}^{m} \int P(x) \cdot h_s\left(rank(x)\right) \cdot \left(x - w_j\right)^2 d^n x \quad \text{with} \quad C(s) = \sum_{k=0}^{m-1} h_s(k).$$

The neural net is composed by a linear layer of neurons.

The Growing Cell Structure (GCS) [15] is a NN which is able to change its structure depending on the data set. Aim of the net is to map the pattern space into a two-dimensional discrete structure *A* in such a way that similar patterns are represented by topological neighbor elements. The structure *A* is a two-dimensional simplex where the vertices are the neurons and the edges attain the topological information. Every modification of the net always maintains the simplex properties. The learning algorithm starts with a simple three node simplex and tries to obtain an optimal network by a controlled growing process: for each *x* pattern of the training set the winner and the neighbors weights are adapted as follows:

$$w_k = w_k + e_b \cdot \left(x - w_k\right); \quad w_i = w_i + e_n \cdot \left(x - w_i\right) \quad \forall i \text{ connected to } k$$

where ε_b and ε_n are constants which determine the adaptation strength for the winner and for the neighbors, respectively.

---

[1] Let $P(x)$ be the pattern probability distribution over the set $V \subseteq \Re^n$ and let $w_{i(x)}$ be the weight vectorof the neuron which classifies the pattern *x*, therefore we define average distortion as:

$$E = \int P(x)\left(x - w_{i(x)}\right)^2 d^n x$$

The insertion of a new node is made after a fixed number λ of adaptation steps. The new neuron is inserted between the unit which has win more times than the others and the more distant of its topological neighbors. The algorithm stops when the network reaches a pre-defined number of elements.

A simpler algorithm is the K-means clustering algorithm [16] in its on-line release which applies the gradient descent directly to the average distortion function above defined as follows:

$$w_i^{(t+1)} = w_i^{(t)} + e^{(t)} \cdot \left(x - w_i^{(t)}\right).$$

The main limitation of this technique is that the error function presents many local minima which stops the learning before reaching the optimal configuration.

The last unsupervised learning algorithm is the Maximum Entropy [17] which applies the gradient descent with soft-max adaptation of the weights to the error function

$$E_{me} = -\frac{1}{b}\int P(x)\ln\sum_{j=1}^{m} e^{-b\left(x-w_j\right)^2} d^n x$$

and adaptation step

$$w_i^{(t+1)} = w_i^{(t)} + e^{(t)} \cdot \frac{\exp\left(-b^{(t)} d_i\right)}{\sum_{k=1}^{m} \exp\left(-b^{(t)} d_k\right)} \cdot \left(x - w_i^{(t)}\right)$$

where *b* is the inverse temperature and takes value increasing in time.

3.2 Hybrid neural nets

Hybrid NNs are composed by a unsupervised single layer NN and a clustering algorithm that uses the information derived by the NN learning algorithm. After the learning of the net, we must apply the clustering algorithm to have a neuron partition in subsets. Their number is equal to the number of the output classes. Furthermore, we want that neurons with similar weight vectors will be in the same class, while neurons with very distant weight vectors will be in different classes. The best strategy is clearly to apply the clustering algorithm directly to the weight vectors of the unsupervised NN after the learning.

A non-neural agglomeration clustering algorithm that divides the pattern set (in this case the weights of the neurons) $W = \{w_1, \ldots, w_m\}$ in l cluster $C_1, \ldots, C_l$ (with $l<m$) can be briefly summarized as follows:

1. we initially divide W in m clusters $C_1, \ldots, C_m$ such that $C_i = \{w_i\}$;
2. we compute the distance matrix D such that $D_{ij} = d(C_i, C_j)$;
3. we find the smallest element D_{ij} of the matrix D and we unify the clusters C_i and C_j in a new one $C_{ij} = C_i \cup C_j$;
4. if the number of clusters is greater than l then go to step 2 else stop.

This is the shared basis of many algorithms appeared in literature [18]. The only difference is the distance function. For example, two different choices can be:

a. $d(C_i, C_j) = \min_{w_{ik}\in C_i \text{ e } w_{jl}\in C_j} \|w_{ik} - w_{jl}\|$ (*nearest neighbor algorithm*);

b. $d(C_i, C_j) = \left\| \frac{1}{|C_i|} \sum_{w_{ik} \in C_i} w_{ik} - \frac{1}{|C_j|} \sum_{w_{jl} \in C_j} w_{jl} \right\|$ *(barycentre method);*

c. $d\left(C_i, C_j\right) = \frac{1}{mn} \sum_{1 \le k \le n,\ 1 \le l \le m} \left\| w_{ik} - w_{jl} \right\|$ (*average between groups*).

The output of the clustering algorithm will be a labeling of the patterns (in this case neurons) in *l* different classes.

3.3 Unsupervised hierarchical neural nets

An alternative approach is to use a new unsupervised single layer NN instead of a clustering algorithm. In this way the second layer NN learns from the weights of the first NN and clusters the neurons on the basis of a similarity measure or a distance. If we apply this process several times then we obtain the unsupervised hierarchical NNs. The number of neurons at each layer decreases form the first to the output layer, and, as a consequence, the net takes a pyramidal aspect as illustrated in Figure 1. The net takes as input a pattern *x* and then the first layer finds the winner neuron. The second layer takes the first layer winner weight vector as input and finds the second layer winner neuron and so on until the top layer. The activation value of the output layer neurons is 1 for the winner unit and 0 for all the others. Briefly, the learning steps of a *s* layer hierarchical NN with training set *X* are the following:

1. The first layer is trained on the patterns of X with one of the previous learning algorithms.
2. The second layer is trained by using the same algorithm or one of the other previous ones on the elements of the set X_2 which is composed by the weigh vectors of the first layer winner units.
3. By using the same algorithm or one of the others, we iterate the process to the *i-th* layer NN (*i>2*) on the training set X_i which is composed by the weight vectors of the winner neurons of the *i-1-th* layer when presenting X to the *1-st* layer NN, X_2 to the *2-nd* layer and so on.

By varying the learning algorithms of the layers we obtain different NNs with different properties and abilities. For instance, by using only SOMs we have a Multi-layer SOM (*ML-SOM*) [19] where every layer is a two-dimensional grid. We can easily obtain *ML-Neural-Gas*, *ML-Maximum-Entropy* or *ML-K-mean*s organized on a hierarchy of linear layers. The ML_GCS has a more complex architecture and has at least 3 units for layer.

We can think to have hierarchical NNs where different layers have different learning algorithms so that we can take advantage from the properties of each model (for example since we cannot have a ML-GCS with 2 output units, then we can use another NN in the output layer).

To solve our basic problem, we need to have a hierarchical NN with a number of output layer neurons equal to the number of the output classes. In this way the labeling becomes a simple problem without reducing the generalization capacity of the net. In fact, the first layer neurons are enough to correctly accomplish the distribution probability density of the input patterns. On the other hand, the number of neurons of a layer cannot rapidly decrease with respect to the number of units of the preceding layer, a hierarchical NN is slower than a single layer NN, and the

computing time depends on the layer number. After the learning phase, it is simple to label in a unique way the input neurons depending on the corresponding output units. In this way we use single layer NNs in the computation on the test set.

4. Experimental results

The experiments were performed on a 1° by 1° degree extracted from one of the plates. Each pixel is a 16 bit integer and each plate is a 8 kbytes × 8 kbytes. In order to identify the principal components of the system we used a 5 x 5 running window to feed both a non linear PCA neural network and a traditional linear PCA. In both cases it turned out that 90% of the information is contained in three components only. As it is clear from Figure 2, the non linear PCA neural network outperforms the traditional PCA tool allowing a much better discrimination of the objects against the background at faint light levels. In Figure 3 we also show the advantage of adopting a hyperbolic tangent as activation function of the non linear PCA net. With respect to the linear case, the distance between faint and luminous object is reduced and the contrast between background and objects is greatly increased.

We therefore used both Hierarchical and Hybrid neural nets to segment the image in six classes. The results of the application of the NNs are summarized in Table 1 and Table 2 and Figure 4, which - for the sake of clarity - refers only to a small fraction of the field. The NN dimensions are illustrated in the second column of Table 1. The performance of the NNs has been computed as distortion rate and as percent of correct detected object after the deblending which is applied to all the segmented images in the same way. This third and final step consists in running a simple algorithm capable to resolve partially overlapping objects. The best performing networks are the Hierarchical (3 layers) Neural gas and the GCS + Neural gas (first layer GCS plus two layers of neural gas). In fact, for what concern the average distortion rate, ML_SOM , ML_Neural Gas and GCS + Neural gas reach the same value of the hybrid Neural Gas systems as shown in Table 1. For what concern the performance in the object identification tasks, ML_Neural Gas and GCS + Neural gas reach a similar correctness percent (more than 93%) and the identification of about 720 corret objects, while the only comparable model is the ML_SOM with a 92.33% of correctness but only 699 correct objects. The comparison of GCS + Neural gas (chosen because is faster than ML_Neural Gas with a comparable correctness) with FOCAS is shown in Table 3 and Figure 5, where our system outperforms the standard system.

References

[1] Jarvis J.F., Tyson J.A., FOCAS: Faint Object Classification And Analysis System in The Astronomical Journal, vol. 86, no. 3, pp. 476-495, 1981.

[2] Oja E., Ogawa H., Wangviwattana J., Learning in nonlinear constrained Hebbian network, In T. Kohonen et al. (Eds.), Artificial neural networks, 385-390, Amsterdam: North-Holland, 1991.

[3] Oja E., Karhunen J., Wang L., Vigario R., Principal and independent components in neural networks - recent developments, Seventh Italian

Workshop on Neural Networks, Vietri 1995, M.Marinaro & R.Tagliaferri Ed.s, World Scientific Pu. Singapore, 16-35, 1996.

[4] Baldi P., Hornik K., Neural networks for principal component analysis: learning from examples without local minima, Neural Networks, 2, (7), 53-58, 1989.

[5] Jutten C., Herault J., Blind separation of sources, part I: an adaptive algorithm based on neuromimetic architecture, Signal Processing, 24, (1), 1-10, 1991.

[6] Oja E., A simplified neuron model as a principal component analyzer, Journal of Mathematical Biology, 15, 267-273, 1982.

[7] Plumbley M., A Hebbian/anti Hebbian network which optimizes information capacity by orthonormalizing the principal subspace, in Proc. IEE Conf. on Artificial Neural Networks, Brighton, UK, 86-90, 1993.

[8] Sanger T. D., Optimal unsupervised learning in a single-layer linear feedforward network, Neural Networks, 2, 459-473, 1989.

[9] Karhunen J., Joutsensalo J., Representation and separation of signals using nonlinear PCA type learning, Neural Networks, 7, 113-127, 1994.

[10] Karhunen J., Joutsensalo J., Generalization of principal component analysis, optimization problems, and neural networks, Neural Networks, 8, 549-562, 1995.

[11] Rasile M., Milano L., Tagliaferri R., Longo G., Periodicity Analysis of Unevenly Spaced Data by means of Neural Networks, Proceedings of the 9th Italian Workshop on Neural Nets WIRN Vietri '97, M. Marinaro & R. Tagliaferri Ed.s, Springer-Verlag, London (pp. 201-212) (1997).

[12] Kohonen T., Self-organized formation of topologically correct feature maps in Biological Cybernetics, vol. 43, pp. 59-69, 1982.

[13] Kohonen T., Self-organization and associative memory (2nd edition) Springer-Verlag Berlin, 1988.

[14] Martinetz T., Berkovich S., Schulten K., Neural-Gas Network for Vector Quantization and its Application to Time-Series Prediction in IEEE Transactions on Neural Networks, vol. 4, no. 4, pp. 558-568, 1993.

[15] Fritzke B., Growing Cell Structures - A Self-Organizing Network for Unsupervised and Supervised Learning in Neural Networks, vol. 7, no. 9, pp. 1441-1460, 1994.

[16] Lloyd S., Least squares quantization in PCM in IEEE Transaction on Information Theory, vol. IT-28, p. 2, 1982.

[17] Rose, Gurewitz F., Fox G., Statistical mechanics and phase transition in clustering in Physical Review Letters, vol. 65, no. 8, pp.945-948, 1990.

[18] Everitt B., Cluster Analysis Social Science Research Council. Heinemann Educational Books. London, 1977.

[19] Koh J., Suk M., Bhandarkar S., A Multilayer Self-Organizing Feature Map for Range Image Segmentation in Neural Networks, vol. 8, no. 1, pp. 67-86, 1995.

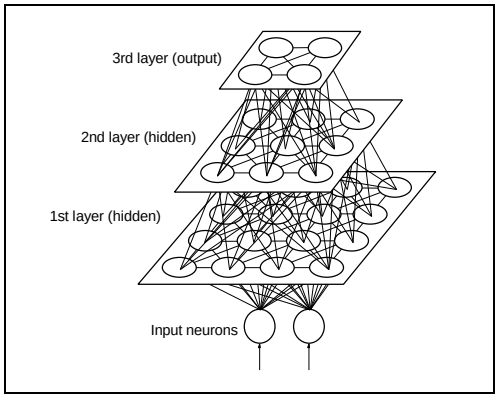

Fig. 1 - A hierarchical NN.

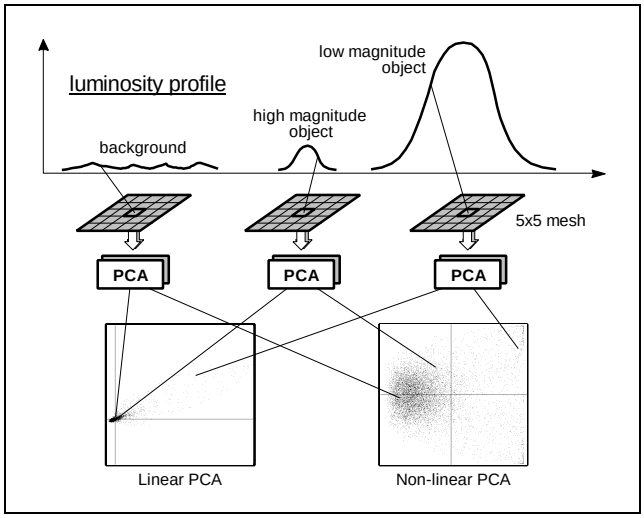

Fig. 2 - Comparison of a Linear PCA and a Non-linear PCA.

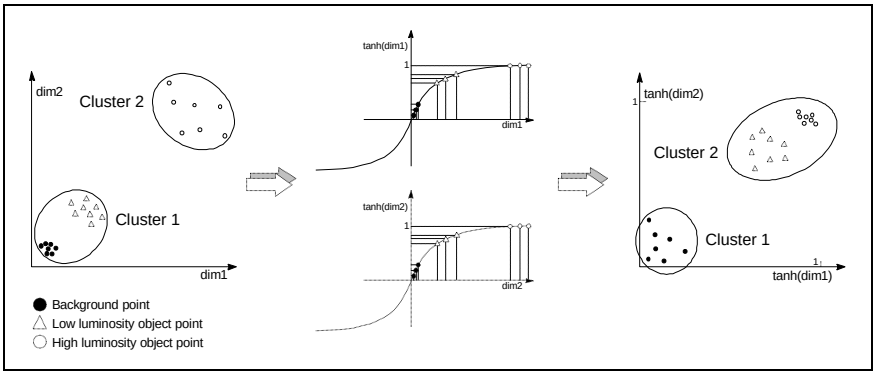

Fig. 3 - Effect of the *tanh* activation function.

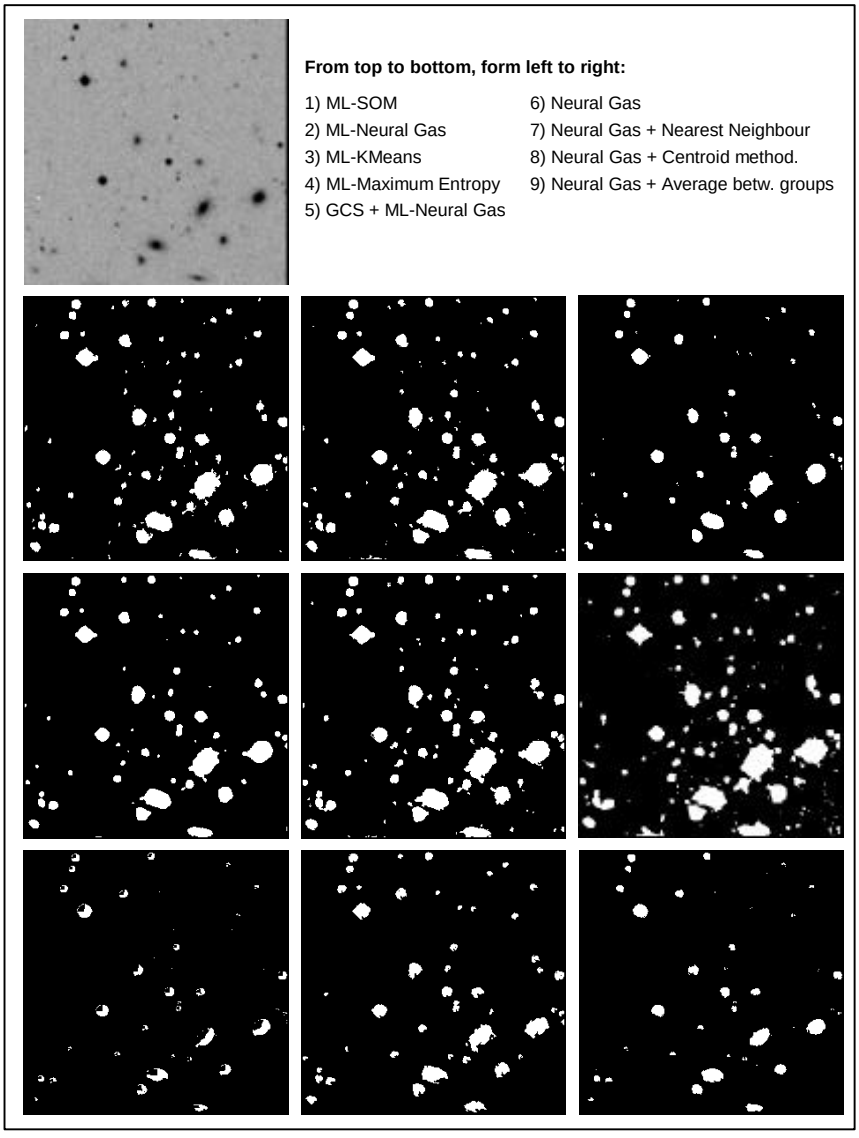

Fig. 4 - Comparison between hierarchical NNs and Hybrid NNs on a real image.

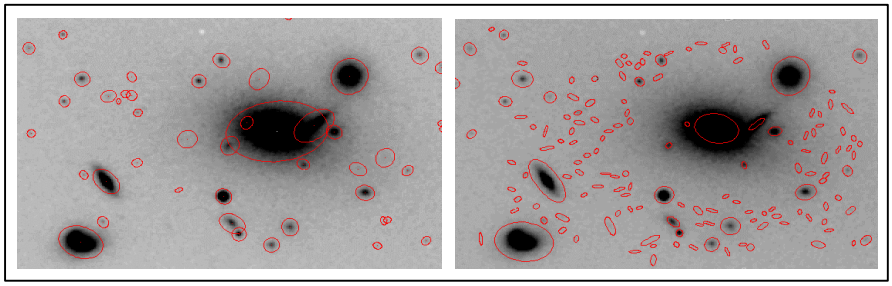

Fig. 5 - Comparison of our method (left) with SKICAT (right) after deblending.

	NN type	neurons	avr. distort.	time (sec.)
1	ML-SOM	50 20 6	0.1188	340
2	ML-Neural Gas	50 20 6	0.1163	302
3	ML-K Means	50 20 6	0.1300	112
4	ML-Maximum Entropy	50 20 6	0.3478	217
5	GCS + ML-Neural Gas	50 20 6	0.1166	235
6	Neural Gas	6	0.2831	103
7	" " + Nearest Neighbour	50	0.1163	256
8	" " + Barycentre method.	50	0.1163	249
9	" " + ave. Between groups	50	0.1163	243

Tab 1 – Hierarchical unsupervised NNs performance in image segmentation tasks: average distortion and computing time.

	NN type	detected obj.	Correct obj.	Corr. Perc.
1	ML-SOM	757	699	92.34
2	ML-Neural Gas	771	720	93.34
3	ML-K Means	426	394	92.49
4	ML-Maximum Entropy	637	607	95.29
5	GCS + ML-Neural Gas	768	719	93.62
6	Neural Gas	1113	720	64.70
7	" " + Nearest Neighbor	384	354	92.19
8	" " + Barycentre Method	438	381	86.99
9	" " + Ave. between Groups	308	297	96.43

Tab 2 – Hierarchical unsupervised NNs performance in object identification tasks.

	FOCAS	Our Method
Total Detected Obiects	2443	1923
Spurius Artifacts	635	120
Right Objects	1808	1803
Correctness percent	74.01	93.76

Tab 3 – Comparison of our method with FOCAS on Coma cluster core.