PERGAMON

AVAILABLE AT
www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT®

Neural Networks 16 (2003) 297–319

Neural Networks

www.elsevier.com/locate/neunet

2003 Special Issue

Neural networks in astronomy

Roberto Tagliaferri[a,b,*], Giuseppe Longo[c,d], Leopoldo Milano[c,e], Fausto Acernese[c,e], Fabrizio Barone[e,f], Angelo Ciaramella[a,b], Rosario De Rosa[c,e], Ciro Donalek[c,d], Antonio Eleuteri[e,g], Giancarlo Raiconi[a], Salvatore Sessa[h], Antonino Staiano[a,b], Alfredo Volpicelli[a,d]

[a]*Departimento di Matematica e Informatica-DMI, Università di Salerno, Baronissi, Italy*
[b]*INFM, unita de Salerno, via S. Allende, Baronissi, Italy*
[c]*Dipartimento di Scienze Fisiche, Università Federico II, via Cinthia, Napoli, Italy*
[d]*INAF, Osservatorio Astronomico di Capodimonte,via Moiariello 16, Napoli, Italy*
[e]*INFN, Sezione di Napoli, via Cinthia, Napoli, Italy*
[f]*Dipartimento di Scienze Farmaceutiche, Università di Salerno, Fisciano, Italy*
[g]*Dipartimento di Matematica ed Applicazioni, Università Federico II, via Cinthia, Napoli, Italy*
[h]*DICOMMA, Università Federico II,via Monteoliveto, 3, Napoli Italy*

Abstract

In the last decade, the use of neural networks (NN) and of other soft computing methods has begun to spread also in the astronomical community which, due to the required accuracy of the measurements, is usually reluctant to use automatic tools to perform even the most common tasks of data reduction and data mining. The federation of heterogeneous large astronomical databases which is foreseen in the framework of the astrophysical virtual observatory and national virtual observatory projects, is, however, posing unprecedented data mining and visualization problems which will find a rather natural and user friendly answer in artificial intelligence tools based on NNs, fuzzy sets or genetic algorithms. This review is aimed to both astronomers (who often have little knowledge of the methodological background) and computer scientists (who often know little about potentially interesting applications), and therefore will be structured as follows: after giving a short introduction to the subject, we shall summarize the methodological background and focus our attention on some of the most interesting fields of application, namely: object extraction and classification, time series analysis, noise identification, and data mining. Most of the original work described in the paper has been performed in the framework of the AstroNeural collaboration (Napoli-Salerno).

Keywords: Neural networks; Astronomy; Self-organizing maps; Data mining; PCA; MUSIC; Bayesian learning; MLP

1. Overview

The fact that artificial neural networks (NN) were originally introduced as simplified models of the brain functioning (processing nodes instead of neurons, multiple connections instead of dendrites and axons) has often had misleading effects on unexperienced users and it is at the origin of the widespread and obviously wrong belief that such tools may do wonders when set loose on oceans of data (Witten & Frank, 2000). As stressed by Bailer-Jones, Gupta, and Singh (2001), even though the logic of the processing of the signals in the brain and in artificial NNs is very similar, the human brain consists of 10^{11} neurons each connected to many others with a complex topology for a total of 10^{14} synaptic connections. Such a level of complexity is unattainable — even in the wildest speculations — by artificial NNs which, at most, can be used to mimic a few simple capabilities of the mind such as: modeling memory, pattern recognition, short term dynamical predictions, etc. In its most general definition, a NN is a tool which learns about a problem through relationship which are intrinsic to the data rather than through a set of predetermined rules. Before proceeding to describe some astronomical applications, however, we need to do a little systematics. A NN can be schematized as a set of N layers, (see Fig. 1) each layer i being composed by m_i neurons. The first layer ($i = 1$) is usually called 'input layer', the intermediate ones 'hidden layers' and the last one ($i = N$) 'output layer'. Each neuron x_i of the input layer can be connected to one, more or all

* Corresponding author. Present address: Department of Mathematics and Informatics Unita di Salerno, DMI, University of Salerno, INFM, Baronissi, Italy.
E-mail address: robtag@unisa.it (R. Tagliaferri).

0893-6080/03/$ - see front matter © 2003 Published by Elsevier Science Ltd.
doi:10.1016/S0893-6080(03)00028-5

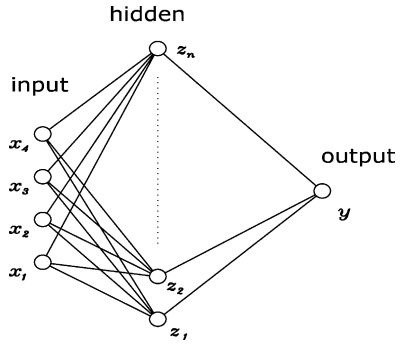

Fig. 1. The architecture of a two layer MLP.

the neurons of the adjacent layer, and this connection is characterized by a weight $w_{j,i}^{(1)}$.

Usually the neuron j in the hidden layer derives a weighted sum of all inputs applied to it (the vector $\mathbf{x}$) and, through either a linear or a non-linear function, produces an output value $\mathbf{z}^{(1)} = f(\mathbf{w}, \mathbf{x})$ which is passed to the neurons in the next layer, and so on up to the last hidden layer ($i = N - 1$) which produces an output vector $\mathbf{y}$. The output layer then performs a simple sum of its inputs and derives the output:

$$y_k = g\left(\sum_{j=0}^{M} w_{k,j}^{(N-1)} z_i^{(N-2)}\right) \tag{1}$$

where z_i is the output of the ith neuron of the $N - 2$ level. The free parameters of the NN are the weight vectors, while the number of layers, the number of neurons in each layer, and the activation functions are chosen from the beginning and specify the so called 'architecture' of the NN.

The whole game of using NNs is in the fact that, in order for the network to yield appropriate outputs for given inputs, the weights must be set to suitable values. The way this is obtained allows a further distinction among modes of operations, namely between 'supervised' and 'unsupervised' methods. In this case, supervised means that the NN learns how to produce the correct output using a rather large subset of data for which there is an a priori and accurate knowledge of the correct answer. Unsupervised means instead that the NN identifies clusters (or patterns) using only some statistical properties of the input data without any need for a priori knowledge. Since this distinction is of fundamental relevance, let us focus on a specific astronomical application, namely the well known star–galaxy classification problem which is of paramount relevance in most astronomical applications and which can be approached in both the pixel and in the catalogue spaces[1] and using both supervised and unsupervised methods.

For simplicity, let us assume that we are working in the catalogue space and that the input is a table where each row is an input vector containing astrometric, morphological and photometric parameters for all objects in a given field.

Supervised methods require that for a rather large subset of data in the input space there must be an a priori and accurate knowledge of the desired property (in this case the membership into either the star or the galaxy classes). This subset defines the 'training set' and, in order for the NN to learn properly, it needs to sample the whole parameter space. The 'a priori knowledge' needed for the objects in the training sets needs therefore to be acquired by means of either visual or automatic inspection of higher S/N and better angular resolution data, and cannot be imported from other data sets, unless they overlap and are homogeneous to the one which is under study. Simulated training data, while sometimes useful for testing, usually fail to reproduce the complexity of true data. Summarizing: supervised methods are usually fast and very accurate, but the construction of a proper training set may be a rather troublesome task (Andreon, Gargiulo, Longo, Tagliaferri, & Capuano, 2000).

Unsupervised methods, instead, do not require a labelled training set and can be used to cluster the input data in classes on the basis of their statistical properties only. Whether these clusters are or are not significant to a specific problem and which meaning has to be attributed to a given class, is not obvious and requires an additional phase, the so called 'labeling'. The labeling can be carried out even if the desired information (label) is available only for a small number of objects representative of the desired classes (in this case, for a few stars and a few galaxies).

A further distinction among different NNs can be based on the way the information propagates across the network: either feedforward (i.e. the information propagates only from the layer K to the layer $K + 1$), or recurrent (i.e. the information may propagate in loops).

As we shall see in what follows, the optimal choices of the architecture of the network and of its operating modes depend strongly on the intrinsic nature of the specific problem to be solved and, since no well defined recipe exists, the user has often to rely on a trial and error procedure. It has to be stressed, in fact that all neural techniques in order to be effective, require a lengthy procedure to be optimized, and an extensive testing is needed to evaluate their robustness against noise and inaccuracy of the input data. More detailed examples and descriptions will be provided in the following sections.

2. Astronomical applications of NNs

In the last few years there has been an increased interest toward the astronomical applications of NNs even though, in spite of the great variety of problems addressed, most applications still make use of an handful of neural models only. This situation is bound to change due to the huge increase in both the quality and the amount of data which are becoming available to the astronomical community

[1] We shall call pixel space the space defined by the raw data, and catalogue space the multi-dimensional one defined by the parameters which are derived from the raw data.

worldwide. Conservative predictions lead in fact to expect that in less than 5 years, much more than 10 TB of data will be acquired worldwide every night and, due to the ongoing efforts for the implementation of the International Virtual Observatory (IVO),[2] most of these data will become available to the community via the network. These huge and heterogeneous data sets will open possibilities which so far are just unthinkable, but it is already clear that their scientific exploitation will require the implementation of automatic tools capable to perform a large fraction of the routine data reduction, data mining and data analysis work.

Some recent reviews of specific aspects of the applications of NNs to astronomical problems can be found in Bailer-Jones et al. (2001) and Gulati and Altamirano (2001). We shall now try, without any pretension of completeness, to summarize some of the most relevant applications which have appeared in the last years. Historically speaking, the first 'astronomical' NNs were applied to star/galaxy recognition, and to spectral and morphological classifications (Odewhan, Stockwell, Penninton, Humpreys, & Zumach, 1992; Klusch & Napiwotzki, 1993). While the problem of star/galaxy classification seems to have been satisfactorily answered Andreon et al. (2000), the other two problems, due to their intrinsic complexity, are still far from being settled (Bazell & Aha, 2001; Ball, 2001; Coryn & Bailer-Jones, 2000; Goderya, Shaukat, & Lolling, 2002; Odewhan & Nielsen, 1994; Snider et al. 2001; Weaver, 2002), for recent applications to galaxy morphology and spectral classification, respectively). NNs have also been applied to planetary studies (Birge & Walberg, 2002; Lepper & Whitley, 2002); to the study and prediction of solar activity and phenomena (Borda et al., 2002; Ganguli, Gavrichtchaka, & Von Steiger, 2002; Lopez Ariste, Rees, Socas-Navarro, & Lites, 2001; Patrick, Gabriel, Rodgers, & Clucas, 2002; Rosa et al., 2000; Steinegger, Veronig, Hanslmeier, Messerotti & Otruba, 2002), to the study of the interplanetary magnetic field (Veselovskii, Dmitriev, Orlov, Persiantsev, & Suvorova, 2000), and to stellar astrophysics (Fuentes, 2001; Torres, García-Berro, & Isern, 2001; Weaver, 2000).

Other fields of application include: time series analysis (Tagliaferri, Ciaramella, Barone, & Milano, 2001; Tagliaferri, Ciaramella, Milano, Barone, & Longo, 1999a), and the identification and characterization of peculiar objects such as QSO's, ultraluminous IR galaxies, and Gamma Ray Bursters (Balastegui, Ruiz-Lapuente, & Canal, 2001); the determination of photometric redshifts (Tagliaferri et al., 2002; Vanzella et al., 2002), the noise removal in pixel lensing data (Funaro, Oja, & Valpola, 2003), the decomposition of multi-frequency data simulated for the Planck mission (Baccigalupi et al., 2000), the search for galaxy clusters (Murtagh, Donalek, Longo, & Tagliaferri, 2003).

Still in its infancy, is the use of NNs for the analysis of the data collected by the new generation of instruments for high energy astrophysics such as, for instance, the neutrino telescopes AUGER (Medina, Tanco, & Sciutto, 2001) and ARGO (Amenomori et al., 2000; Bussino, 1999); the gamma ray Cherenkhov telescope (Razdan, Haungs, Rebel, & Bhat, 2002; Schaefer, Hofmann, Lampeitl, & Hemberger, 2001), the VIRGO gravitational waves interferometer (Acernese et al., 2002; Barone et al., 1999) and even for the search of the Higgs boson (Smirnov, 2002).

In what follows, we shall concentrate on some applications which have been addressed by our group in the framework of the AstroNeural collaboration which was started in 1999 as a joint project of the Department of Mathematics and Informatics at the University of Salerno and of the Department of Physics at the University Federico II of Napoli.[3]

The main goal of this collaboration was the implementation of a set of soft computing tools to be integrated within the data reduction and data analysis pipelines of the new generation of telescopes and focal plane instruments which will soon be producing huge data flows (up to 1 TB per night). These data will often need to be reduced and analyzed on a very short time scale (8–12 h). The tools produced by the AstroNeural collaboration are integrated within the AstroMining package: a set of modules written in MatLab and C++.

3. Neural models most commonly adopted in astronomy

3.1. The multi-layer perceptron and the probabilistic Bayesian learning

A NN is usually structured into an input layer of neurons, one or more hidden layers and one output layer. Neurons belonging to adjacent layers are usually fully connected and the various types and architectures are identified both by the different topologies adopted for the connections and by the choice of the activation function. Such networks are generally called multi-layer perceptron (MLP; Bishop, 1995) when the activation functions are sigmoidal or linear. (Fig. 1).

The output of the jth hidden unit is obtained first by forming a weighted linear combination of the d input values, and then by adding a bias to give:

$$z_j = f\left(\sum_{i=0}^{d} w_{j,i}^{(1)} x_i\right) \tag{2}$$

where d is the number of the input, $w_{j,i}^{(1)}$ denotes a weight in the first layer (from input i to hidden unit j). Note that $w_{j,0}^{(1)}$ denotes the bias for the hidden unit j, and f is an activation

[2] From the fusion of the European AVO and of the American National Astrophysical Observatory (NVO).

[3] Financed through several MURST-COFIN grants, and an ASI (Italian Space Agency).

function such as the continuous sigmoidal function:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{3}$$

The outputs of the network are obtained by transforming the activation of the hidden units using a second layer of processing elements.

$$y_k = g\left(\sum_{j=0}^{M} w_{k,j}^{(2)} z_i\right) \tag{4}$$

where M is the number of hidden unit, $w_{k,j}^{(2)}$ denotes a weight in the second layer (from hidden unit j to output unit k). Note that $w_{k,0}^{(2)}$ denotes the bias for the output unit k, and g is an activation function of the output units which does not need to be the same function as for the hidden units. The learning procedure is the so called 'back propagation' which works as follows: we give to the input neurons the first pattern and then the net produces an output. If this is not equal to the desired output, the difference (error) between these two values is computed and the weights are changed in order to minimize it. These operations are repeated for each input pattern until the mean square error of the system is minimized. Given the pth pattern in input, a classical error function E_p (called sum-of-squares) is:

$$E_p = \frac{1}{2}\sum_j (t_j^p - y_j^p)^2 \tag{5}$$

where t_j^p is the pth desired output value and y_i^p is the output of the corresponding neuron. Due to its interpolation capabilities, the MLP is one of the most widely used neural architectures. The MLP can be trained also using probabilistic techniques.

The Bayesian learning framework offers several advantages over classical ones (Bishop, 1995): (i) it cannot overfit the data; (ii) it is automatically regularized; (iii) the uncertainty in the prediction can be estimated.

In the conventional maximum likelihood approach to training, a single weight vector is found which minimizes the error function; in contrast, the Bayesian scheme considers a probability distribution over the weights. This is described by a prior distribution $p(\mathbf{w})$ which is modified when we observe the data D. This process can be expressed with Bayes' theorem:

$$p(\mathbf{w}|D) = \frac{p(D|\mathbf{w})p(\mathbf{w})}{p(D)} \tag{6}$$

To evaluate the posterior distribution, we need expressions for the prior $p(\mathbf{w})$ and for the likelihood $p(D|\mathbf{w})$. The prior over weights should reflect the knowledge, if any, about the mapping to be built. We can write the prior as an exponential of the form:

$$p(\mathbf{w}) = \frac{1}{Z_W(\alpha)} e^{-\alpha E_W} \tag{7}$$

If the data $s(t)$ are locally smooth, we must have that the network mapping should be smooth, too. This can be easily obtained if we put (Bishop, 1995):

$$E_W = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} = \frac{1}{2}\|\mathbf{w}\|^2 \tag{8}$$

so that the prior is $N(0, 1/\alpha)$. In this way, small weights have higher probability. Note that the prior is parametric, and the regularization parameter α is called a *hyperparameter*, because it controls the distribution of the network parameters. Note also that Eqs. (7) and (8) can be specialized for k different sets of weights by using different regularization parameters α_k for each group. Indeed, to preserve the scaling properties of network mappings, the prior Eq. (7) must be written as (Neal, 1996):

$$p(\mathbf{w}) = \frac{1}{Z_W(\{\alpha_k\}_k)} e^{-\sum_k \alpha_k E_{W_k}} \tag{9}$$

where k runs over the different weight groups. In analogy with the prior, we can write the likelihood as an exponential of the form:

$$p(D|\mathbf{w}) = \frac{1}{Z_D(\beta)} e^{-\beta E_D} \tag{10}$$

where E_D is an error function. Since our problem is a regression one, we want our network to model the distribution $p(q|\mathbf{x})$. If we assume Gaussian noise due to measurement errors, then an appropriate error function is (Bishop, 1995):

$$E_D = \frac{1}{2}\sum_n \{y(\mathbf{x}^n; \mathbf{w}) - q^n\}^2, \tag{11}$$

where y is the network output, q the target and n runs over the patterns. In this way, the likelihood (also termed *noise model*) is $N(y, 1/\beta)$.

Once the expressions for the prior and the noise model is given, we can evaluate the posterior:

$$p(\mathbf{w}|D, \{\alpha_k\}_k, \beta) = \frac{1}{Z(\{\alpha_k\}_k, \beta)} e^{-\beta E_D - \sum_k \alpha_k E_{W_k}} \tag{12}$$

This distribution is usually very complex and multi-modal, and the determination of the normalization factor is very difficult. Also, the hyperparameters must be integrated out, since they are only used to determine the form of the distributions.

The approach followed is the one introduced by MacKay (1993), which integrates the parameters separately from the hyperparameters by means a Gaussian approximation and then finds the mode with respect to the hyperparameters. This procedure gives a good estimation of the *probability mass* attached to the posterior, in particular way for distributions over high-dimensional spaces (MacKay, 1993). Using a Gaussian approximation the following

integral is easily evaluated:

$$p(\{\alpha_k\}_k, \beta|D) = \int p(\{\alpha_k\}_k, \beta, \mathbf{w}|D)\mathrm{d}\mathbf{w} \quad (13)$$

Then, the mode of the resulting distribution can be evaluated:

$$\{\{\hat{\alpha}_k\}_k, \hat{\beta}\} = \mathrm{argmax}_{\{\alpha_k\}_k,\beta}(p(\{\alpha_k\}_k, \beta|D)) \quad (14)$$

and the hyperparameters thus found can be used to evaluate:

$$\hat{\mathbf{w}} = \mathrm{argmax}_{\mathbf{w}}(b(\mathbf{w}|\{\alpha_k\}_k, \beta, D)) \quad (15)$$

The above outlined scheme must be repeated until a self-consistent solution $(\mathbf{w}, \{\alpha_k\}_k, \beta)$ is found.

3.2. The self-organizing maps

The SOM algorithm (Kohonen, 1995) is based on unsupervised competitive learning, which means that the training is entirely data-driven and the neurons of the map compete with each other (Vesanto, 1997). A SOM allows the approximation of the probability density function of the data in the training set (id est prototype vectors best describing the data), and a highly visualized approach to the understanding of the statistical characteristics of the data. In a crude approximation, a SOM is composed by neurons located on a regular, usually one- or two-dimensional grid (see Fig. 2 for the two-dimensional case). Each neuron i of the SOM may be represented as an n-dimensional weight:

$$\mathbf{m}_i = [m_{i_1}, \ldots, m_{i_n}]^{\mathrm{T}} \quad (16)$$

where n is the dimension of the input vectors. Higher dimensional grids are not commonly used since in this case the visualization of the outputs becomes problematic.

In most implementations, SOMs neurons are connected to the adjacent ones by a neighborhood relation which dictates the structure of the map. In the two-dimensional case, the neurons of the map can be arranged either on a rectangular or a hexagonal lattice and the total number of neurons determines the granularity of the resulting mapping, thus affecting the accuracy and the generalization capability of the SOM.

The use of SOM as data mining tools requires several logical steps: the construction and the normalization of the data set (usually to 0 mean and unit variance), the initialization and the training of the map, the visualization and the analysis of the results. In the SOMs, the topological relations and the number of neurons are fixed from the beginning via a trial and error procedure, with the neighborhood size controlling the smoothness and generalization of the mapping. The initialization consists in providing the initial weights to the neurons and, even though the SOM are robust with respect to the initial choice, a proper initialization usually allows faster convergence.

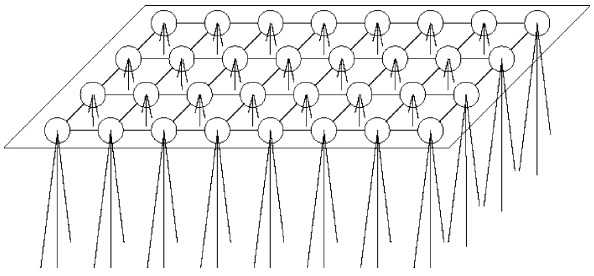

Fig. 2. The architecture of a SOM with a two dimensional grid architecture and three inputs fed to all the neurons.

For instance, the AstroMining package allows three different types of initialization procedures: *random initialization*, where the weight vectors are initialized with small random values; *sample initialization*, where the weight vectors are initialized with random samples drawn from the input data set; *linear initialization*, where the weight vectors are initialized in an orderly fashion along the linear subspace spanned by the two principal eigenvectors of the input data set. The corresponding eigenvectors are then calculated using the Gram-Schmidt procedure detailed in Vesanto (1997). The initialization is followed by the training phase.

In each training step, one sample vector $\mathbf{x}$ from the input data set is randomly chosen and a similarity measure is calculated between it and all the weight vectors of the map. The best-matching unit (BMU), denoted as c, is the unit whose weight vector has the greatest similarity with the input sample $\mathbf{x}$. This similarity is usually defined via a distance (usually Euclidean) and, formally speaking, the BMU can be defined as the neuron for which:

$$\|\mathbf{x} - \mathbf{m}_c\| = \min_i \|\mathbf{x} - \mathbf{m}_i\| \quad (17)$$

where $\|\cdot\|$ is the adopted distance measure. After finding the BMU, the weight vectors of the SOM are updated and the weight vectors of the BMU and of its topological neighbors are moved in the direction of the input vector, in the input space. The SOM updating rule for the weight vector of the unit i can be written as:

$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t) + h_{ci}(t)[\mathbf{x}(t) - \mathbf{m}_i(t)] \quad (18)$$

where t denotes the time, $\mathbf{x}(t)$ the input vector, and $h_{ci}(t)$ the neighborhood kernel around the winner unit, defined as a non-increasing function of the time and of the distance of the unit i from the winner unit c which defines the region of influence that the input sample has on the SOM. This kernel is composed by two parts: the neighborhood function $h(d, t)$ and the learning rate function $\alpha(t)$:

$$h_{ci}(t) = h(\|\mathbf{r}_c - \mathbf{r}_i\|, t)\alpha(t) \quad (19)$$

where r_i is the location of unit i on the map grid. The AstroMining package allows the use of several neighborhood functions, among which the most commonly used is the so called Gaussian neighborhood function:

$$\exp(-\|\mathbf{r}_c - \mathbf{r}_i\|^2/2\sigma^2(t)) \quad (20)$$

The learning rate $\alpha(t)$ is a decreasing function of time which, always in the AstroMining package, is:

$$\alpha(t) = (A/t + B) \tag{21}$$

where A and B are some suitably selected positive constants. Since also the neighbors radius is decreasing in time, then the training of the SOM can be seen as performed in two phases. In the first one, relatively large initial α value and neighborhood radius are used, and decrease in time. In the second phase both α value and neighborhood radius are small constants right from the beginning.

3.3. The generative topographic mapping

Latent variable models (Svensen, 1998) aim to find a representation for the distribution $p(x)$ of data in a D-dimensional space $x = [x_1, \ldots, x_D]$ in terms of a number L of latent variables $z = [z_1, \ldots, z_L]$ (where, in order for the model to be useful, $L \ll D$). This is usually achieved by means of a non-linear function $y(z, \mathbf{W})$, governed by a set of parameters $\mathbf{W}$, which maps the latent space points $\mathbf{W}$ into corresponding points $y(z, \mathbf{W})$ of the input space. In other words, $y(z, \mathbf{W})$ maps the hidden variable space into a L-dimensional non-euclidean manifold embedded within the input space (see Fig. 3 for the three-dimensional input and two-dimensional latent variable space case).

Therefore, a probability distribution $p(z)$ (which is also known as a prior distribution of z) defined in the latent variable space, will induce a corresponding distribution $p(y|z)$ in the input data space. The AstroMining GTM routines are largely based on the Matlab GTM Toolbox (Svensen, 1998) and provide the user with a complete environment for GTM analysis and visualization. In order to render more 'user friendly' the interpretation of the resulting maps, the GTM package defines a probability distribution in the data space conditioned on the latent variables and, using the Bayes Theorem, the posterior distribution in the latent space for any given point x in the input data space, is:

$$p(z_k|x) = \frac{p(x|z_k, \mathbf{W}, \beta)p(z_k)}{\sum_{k'} p(x|z_{k'}, \mathbf{W}, \beta)p(z_{k'})} \tag{22}$$

provided that the latent space has no more than three dimensions ($L = 3$), its visualization becomes trivial.

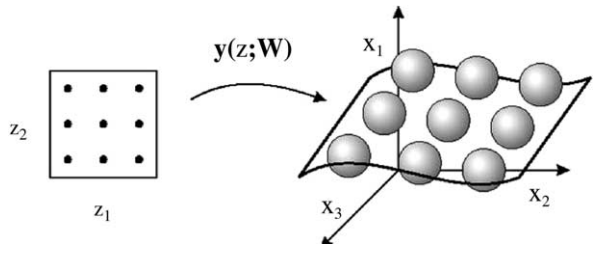

Fig. 3. The architecture of a GTM with a three-dimensional input space and a two-dimensional latent variable space.

3.4. Fuzzy similarity

Fuzzy set theory is a generalization of the classical set theory. Formally introduced by Zadeh (1965), it stresses the idea of partial truth value between the crisp values 0 (completely false) and 1 (completely true), configured in the matter of a degree of belonging of an element to a set X. Let S be an universe of discourse and X be a non-empty subset of S. It is known that in set theory, a subset X can be identified with its characteristic function defined for all $x \in S$, as $\mu_X(x) = 1$ if $x \in X$, and $\mu_X(x) = 0$ otherwise. A fuzzy set μ_X of X is a function from S into the real unit interval $0, 1$, representing just the degree $\mu_X(x)$ with which an element x of S belongs to X. In set theory the operation of conjunction and disjunction between subsets are modeled from the classical operators $\wedge$ (min) and $\vee$ (max), respectively, assuming only the two membership values 0 or 1.

In fuzzy set theory, these operations are generalized to the concept of triangular norm (for short, t-norm) and triangular conorm (for short, t-conorm). Since the concept of conorm is dual of norm, we can restrict ourselves to the case of t-norms. A t-norm $t : [0, 1]^2 \to [0, 1]$ is a binary operation (by setting, as usually, $t(x, y) = xty$ for all $x, y \in [0, 1]$) which is commutative, associative, non-decreasing in each variable coordinate and such that $xt0 = 0$, $xt1 = x$. We know (Hajek, 1998) that if t is assumed to be continuous, then there is a unique operation $x \to_t y$, called *residuum* of t, defined as $(x \to_t y) = \max\{z \in [0, 1] : (xtz) \leq y\}$ such that $(xtz) \leq y$ iff $z \leq (x \to_t y)$. In what follows, we always suppose that t is continuous and use one of the most famous t-norm used in fuzzy set theory (with its correspondent residuum), namely the Łukasiewicz t-norm defined for all $x, y \in [0, 1]$ as:

$$\begin{aligned} xty &= \max\{0, x + y - 1\} \\ (x \to_t y) &= \min\{1, 1 - +y\}; \end{aligned} \tag{23}$$

Now we introduce the concept of *bi-residuum* that offers an elegant way to interpret the concept of similarity (or equivalence) in fuzzy set theory. The bi-residuum is defined as follows:

$$x \leftrightarrow y = (x \to_t y) \wedge (y \to_t x) \tag{24}$$

In the case of Łukasiewicz t-norm, it is easy to see that the bi-residuum is calculated via

$$x \leftrightarrow y = 1 - \max(x, y) + \min(x, y) \tag{25}$$

A crisp equivalence relation R on a subset X has as characteristic function $\mu_R : X \text{ times } X \to \{0, 1\}$ (from now on, for simplicity we put $\mu_R = R$) such that R is reflexive $(R(x, x) = 1 \forall x \in X)$, symmetric $(R(x, y) = R(y, x) \forall x \in X)$ and transitive $(R(x, y) = R(y, z) = 1 \Rightarrow R(x, z) = 1 \forall x, y, z \in X)$. Fuzzifying this concept, a fuzzy similarity relation S on X with respect to a t-norm is a relation $S : X \times X \to [0, 1]$

such that S is reflexive, symmetric and 't-transitive' (i.e. $S(x,y)tS(y,z) \leq S(x,z) \forall x,y,z \in X$). We recall (Hajek, 1998) that any fuzzy set μ_X generates a fuzzy similarity S on X, defined for all $x, y \in X$ as

$$S(x,y) = \mu_X(x) \leftrightarrow \mu_X(y) \tag{26}$$

If we consider n fuzzy similarity S_i, $i = 1,\ldots,n$ on a set X with respect to the Łukasiewicz t-norm, then the following relation defined pointwise as:

$$S\langle x,y\rangle = \frac{1}{n}\sum_{i=1}^{n} S_i(x,y) \tag{27}$$

is the so-called *total fuzzy similarity* (TFS) with respect to the Łukasiewicz t-norm (Turunen, 1999).

So far, in the AstroMining tool, fuzzy similarity has been implemented to perform only object classification (such as, for instance, star–galaxy separation). The idea behind this part of the tool is that a few prototypes for each object class can be used as reference points for the catalog objects.

The prototypes selection is accomplished by means of a self-organizing maps (SOM) or Fuzzy C-means. In this way, using a method to search the prototypes, and the fuzzy similarities, it is possible to compute for each object of the catalog, its degree of similarity with respect to the prototypes.

3.5. Independent component analysis

The independent component analysis (ICA) is used to reveal underlying factors or components in multi-variate multi-dimensional statistical data. To do so, the ICA defines a generative model for the observed multi-variate data. In the model, the data variables are assumed to be linear mixtures of some unknown latent variables, which are assumed to be non-Gaussian and mutually independent. ICA models consist of estimating both the unknown mixing system and the latent variables (called independent components or sources) when we only observe the data variables. Approaches for ICA estimation are based on maximization of non-gaussianity (Hyvärinen, Karhunen, & Oja, 2001), minimization of mutual information (Bell & Sejnowski, 1995) or maximum likelihood (Amari, Cichoki, & Yang, 1996; Pham, Garrat, & Jutten, 1992).

The ICA method is related to the method called blind source separation (BSS) aimed to the extraction of independent sources from their linear mixtures. Formally, the mixing model is written as:

$$\mathbf{x} = \mathbf{A}\mathbf{s} + \mathbf{n} \tag{28}$$

where $\mathbf{x}$ is an observed m-dimensional vector, $\mathbf{s}$ an n-dimensional random vector whose components are assumed to be mutually independent, $\mathbf{A}$ a constant $m \times n$ matrix to be estimated, and $\mathbf{n}$ the additive noise. The additive noise term $\mathbf{n}$ is often omitted in Eq. (28) because it can be incorporated in the sum as one of the source signals. In addition to the independent assumption, we assume that the number of available different mixtures m is at least as large as the number of sources n. Usually, m is assumed to be known in advance, and often $m = n$. Only one of the source signals s_i is allowed to have a Gaussian distribution, because it is impossible to separate two or more Gaussian sources (Hyvärinen et al., 2001). In adaptive source separation an $m \times n$ separating matrix $\mathbf{W}$ is updated so that the vector:

$$\mathbf{y} = \mathbf{W}\mathbf{x} \tag{29}$$

is an estimate $\mathbf{y} \simeq \mathbf{s}$ of the original independent source signals. In what follows we will present the *fixed-point* algorithm, namely FastICA (Hyvärinen et al., 2001). The FastICA learning rule finds a direction, i.e. a unit vector $\mathbf{w}$ such that the projection $\mathbf{w}^{\mathrm{T}}\mathbf{x}$ maximizes independence of the single estimated source y. Independence is here measured by the approximation of the negentropy given by

$$J_G(\mathbf{w}) = [E\{G(\mathbf{w}^{\mathrm{T}}\mathbf{x})\} - E\{G(\nu)\}]^2 \tag{30}$$

where $\mathbf{w}$ is an m-dimensional (weight) vector, $\mathbf{x}$ represents our mixture of signals, and $E\{(\mathbf{w}^{\mathrm{T}}\mathbf{x})^2\} = 1$, ν a standardized Gaussian random variable. Maximizing J_G allows to find *one* independent component, or projection pursuit direction. We remark that the algorithm requires a preliminary whitening of the data: the observed variable $\mathbf{x}$ is linearly transformed to a zero-mean variable $\mathbf{v} = \mathbf{Q}\mathbf{x}$ such that $E\{\mathbf{v}\mathbf{v}^{\mathrm{T}}\} = I$. Whitening can always be accomplished by, e.g. principal component analysis (Hyvärinen et al., 2001).

The one-unit *fixed-point* algorithm for finding a row vector $\mathbf{w}$ is (Hyvärinen et al., 2001)

$$\mathbf{w}^* = E[\mathbf{v}g(\mathbf{w}_i^{\mathrm{T}}\mathbf{v})] - E[g'(\mathbf{w}_i^T\mathbf{v})]\mathbf{w}_i \qquad \mathbf{w}_i = \mathbf{w}_i^*/\|\mathbf{w}_i^*\| \tag{31}$$

where $g(\cdot)$ is a suitable non-linearity, in our case $g(y) = \tanh(y)$, and $g'(y)$ is its derivative with respect to y.

The algorithm of the previous equations estimates just one of the independent components. To estimate several independent components, one needs to run the one-unit FastICA algorithm using several units (e.g. neurons) with weight vectors $\mathbf{w}_1,\ldots,\mathbf{w}_n$. To prevent different vectors from converging to the same maximum, we must decorrelate the outputs $\mathbf{w}_1^{\mathrm{T}}\mathbf{x},\ldots,\mathbf{w}_n^{\mathrm{T}}\mathbf{x}$ after every iteration. In specific applications it may be desirable to use a symmetric decorrelation, in which vectors are not privileged over the others. This can be accomplished by the classical method involving matrix square roots. By assuming that the data are whitened, we have that:

$$\mathbf{W} = \mathbf{W}(\mathbf{W}^{\mathrm{T}}\mathbf{W})^{-1/2} \tag{32}$$

where $\mathbf{W}$ is the matrix of the vectors $(\mathbf{w}_1,\ldots,\mathbf{w}_n)$, and the inverse square root is obtained from the eigenvalue decomposition as:

$$(\mathbf{W}^{\mathrm{T}}\mathbf{W})^{-1/2} = \mathbf{E}\mathbf{D}^{-1/2}\mathbf{E}^{\mathbf{T}}$$

where $\mathbf{E}$ is the eigenvector matrix and $\mathbf{D}$ the diagonal eigenvalue one.

Recently, ICA have been tested for artefacts detection and removal on astronomical images originally taken to study pixel lensing (Funaro et al., 2003). A technique which consists in monitoring the light variation of individual pixels (in the place of individual stars as it is the case in normal light curve analysis). In this case, true pixel variations need to be disentangled from artifacts (induced by variable galactic stars accidentally falling on the same pixel, cosmic ray events, noise peaks, etc.). In this case, the facts that the independence of the artefacts is often theoretically guaranteed, and that the linear model holds exactly, argue in favour of the use of ICA.

3.6. Time series analysis

The search for periodicities in time or spatial dependent signals is a topic of the uttermost relevance in many fields of research, from geology (stratigraphy, seismology, etc., Brescia et al., 1996) to astronomy (Tagliaferri et al., 1999a) where it finds wide application in the study of light curves of variable stars, AGNs, etc. The more sensitive instrumentation and observational techniques become, the more frequently we find variable signals in time domain that previously were believed to be constant. There are two types of problems related either to unknown fundamental period of the data, or to their unknown multiple periodicities; typical cases being the determination of the fundamental period of eclipsing binaries both in the light and/or the radial velocity curves, or the multiple periodicities determination of light curves of pulsating stars. The general problem is how to estimate the frequencies of periodic signals which may be contained in a physical variable x measured at discrete times t_i. In what follows, we shall assume that $x(t_i)$ can be written as the sum of the signal x_s and a random error R :

$$x_i = x(t_i) = x_s(t_i) + R_i \tag{33}$$

Many different tools based on different types of Fourier analysis may be effectively used in the case of even sampling (Deeming, 1975; Horne & Baliunas, 1986; Scargle, 1982), but they lead to ambiguous results when are applied to unevenly sampled data. Possible ways around such as, for instance, the resampling of the data into an evenly sampled sequence via interpolation, are not very useful since they usually introduce a strong amplification of the noise thus preventing the use of Fourier methods which are always very sensitive to the noise level (Horowitz, 1974). A commonly used tool is the so called Periodogram, which is an estimator of the signal energy in the frequency domain (Deeming, 1975; Oppenheim & Schafer, 1989; Kay, 1988; Lomb, 1976) and has been extensively applied to the analysis of unevenly sampled stellar light curves. Its use, however, is undermined by the difficulties encountered in dealing with the aliasing effects. A variation of the classical Periodogram (P) was introduced by Lomb (1976) and is distributed in Numerical Recipes in C: The Art of Scientific Computing (1992). Let us suppose that the signal consists of N data points $x(n)$ and let $\bar{x}$ and σ^2 be the mean and the variance:

$$\bar{x} = \frac{1}{N}\sum_{n=1}^{N} x(n) \qquad \sigma^2 = \frac{1}{N-1}\sum_{n=l}^{N}(x(n)-\bar{x})^2 \tag{34}$$

The normalized Lomb's Periodogram—id est the power spectra as function of an angular frequency $\omega \equiv 2\pi f > 0$—is defined as:

$$P_N(\omega)\frac{1}{2\sigma^2}\left[\frac{\left[\sum_{n=0}^{N-1}(x(n)-\bar{x})\cos\,\omega(t_n-\tau)\right]^2}{\sum_{n=0}^{N-1}\cos^2\,\omega(t_n-\tau)}\right] + \frac{1}{2\sigma^2} \times \left[\frac{\left[\sum_{n=0}^{N-1}(x(n)-\bar{x})\sin\,\omega(t_n-\tau)\right]^2}{\sum_{n=0}^{N-1}\sin^2\,\omega(t_n-\tau)}\right] \tag{35}$$

where τ is an offset defined as:

$$\tan(2\omega\tau) \equiv \frac{\sum_{n=0}^{N-1}\sin 2\omega t_n}{\sum_{n=0}^{N-1}\cos 2\omega t_n} \tag{36}$$

More modern frequency estimators, based on the signal autocorrelation matrix eigenvectors, have been introduced to overcome the biases of more traditional Fourier methods.

Another approach was proposed which makes use of the algorithm STIMA based on the frequency estimator MUlti SIgnal Classificator—MUSIC (Oppenheim & Schafer, 1989) and of a non-linear NN which extracts the principal components of the autocorrelation matrix. Let us assume that the signal is composed by p narrow band sinusoidal components which can be modeled as stationary ergodic signals id est, for which it can be assumed that the phases are independent random variables uniformously distributed in the interval $[0, 2\pi]$ (Kay, 1988). Frequencies may be estimated from the signal autocorrelation matrix (a.m.) which is the sum of the signal and noise matrices; the p principal eigenvectors of the signal matrix are the same of the total matrix.

The eigenvectors of this matrix may then be extracted using a non-linear PCANN (Andreon, Gargiulo, Longo, Tagliaferri, & Capuano, 2000; Karhunen & Joutsensalo, 1994, 1995; Oja, Karhunen, Wang, & Vigario, 1996; Sanger, 1989; Tagliaferri et al., 1999a, 2001). Before being fed to the NN, the data need to be corrected for the average pattern in order to obtain zero mean processes (Karhunen & Joutsensalo, 1994, 1995; Tagliaferri et al., 1999a). In this case, the fundamental learning parameters of the NN are: (i) the initial weight matrix; (ii) the number of

neurons, id est the number of principal eigenvectors which need to be estimated or twice the number of periodicities expected in the signal; (iii) the non-linear learning parameter α; (iv) the learning rate μ.

The weight matrix $\mathbf{W}$ is initialized either with small random values or, only in the case of regular signals and in order to speed up the convergence of the neural estimator (NE), with the first pattern of the signal as columns of the matrix. In order to establish when the NN has converged and to accomplish the periodicities estimation on unevenly sampled data (without interpolation) we used the following modified MUSIC estimator (Tagliaferri et al., 1999a, 2001):

$$P_{\mathrm{MUSIC}} = \frac{1}{M - \sum_{i=1}^{M} |\mathbf{e}_f^H \mathbf{w}(i)|^2} \tag{37}$$

where $\mathbf{w}(i)$ is the ith weight vector after learning, and

$$\mathbf{e}_f^H = [1, \mathbf{e}_f^{\mathrm{j}2\pi f t_0}, \ldots, \mathbf{e}_f^{\mathrm{j}2\pi f t_{(L-1)}}]^H \tag{38}$$

where $\{t_1, \ldots, t_{(L-1)}\}$ are the first L components of the temporal coordinates of the uneven signal. We note that for interpolated data

$$\mathbf{e}_f^H = [1, \mathbf{e}_f^{\mathrm{j}2\pi f}, \ldots, \mathbf{e}_f^{\mathrm{j}2\pi f(L-1)}]^H \tag{39}$$

The final algorithm is the following:

Step 1. Initialize the weight vectors $\mathbf{w}(i)$, $\forall i = 1, \ldots, p$ with small random values, or with orthonormalized signal patterns. Initialize the learning threshold ϵ, the learning rate μ. Reset the pattern counter $k = 0$.
Step 2. Input the kth pattern $\mathbf{x}_k = [x(k), \ldots, x(k+N+1)]$, where N is the number of the input components
Step 3. Calculate the output for each neuron $y(j) = \mathbf{w}^{\mathrm{T}}(j)$ $\mathbf{x}_i$ $\forall i = 1, \ldots, p$
Step 4. Modify the weights $\mathbf{w}_{k+1}(i) = \mathbf{w}_k(i) + \mu_k g(y_k(i))$ $\mathbf{e}_k(i)$ $\forall i = 1, \ldots, p$
Step 5. Apply the convergence test. If:

$$\frac{1}{M - \sum_{i=1}^{M} |\mathbf{e}_f^H \mathbf{w}(i)|^2} > 0 \tag{40}$$

then go to Step 7
Step 6. $k = k + 1$. Go to step 2
Step 7. End

After the learning, the weight matrix columns are fed to the frequency estimator MUSIC and the estimated signal frequencies are obtained as the peak locations of the functions. We note that if f is the frequency of the ith sinusoidal component $f = f_i$, we shall have $\mathbf{e} = \mathbf{e}_i$, and $P_{\mathrm{MUSIC}} \to \infty$. In practice, this means that there will be a peak in correspondence of the component frequency P_{MUSIC}.

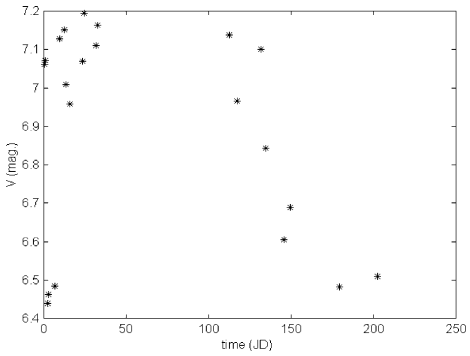

Fig. 4. Light curve of the Cepheid star SU Cyg.

4. Some applications

4.1. Light curve analysis

A good example of the above outlined time series analysis method is the application to the light curve of the Cepheid SU Cygni presented in Tagliaferri et al. (1999a). The sequences were obtained in the Johnson U, B, V, R, I photometric bands and covers the period from June to December 1977. The light curve is composed by 21 samples in the V band, with a period of 3.8^d, as shown in Fig. 4. In this case, the parameters of the n.e. were: $N = 10$, $p = 2$, $\alpha = 20$, $\mu = 0.001$. The estimate frequency interval is [0(1/JD), 0.5(1/JD)]. The estimated frequency is 0.26(1/JD) in agreement with the Lomb's P, but without showing any spurious peak (Figs. 5 and 6).

Similar experiments were also performed with equally good results on the stars U Aql (period $= 7.01^d$), and X Cygni (period $= 16.38^d$), U Peg (two periods of 5.4 and 2.8^d) (Tagliaferri et al., 1999a).

4.2. Image segmentation and object detection

In processing an astronomical image, the final goal is usually the construction of a catalogue containing as many as possible astrometric, geometric, morphological and photometric parameters for each individual object present

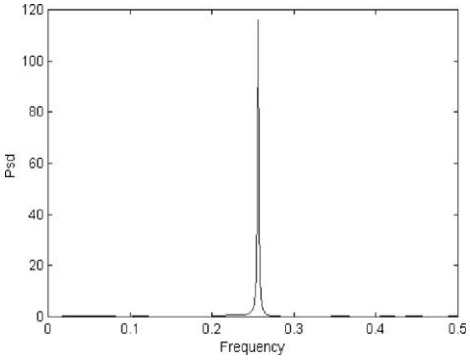

Fig. 5. n.e. estimate of SU Cygni.

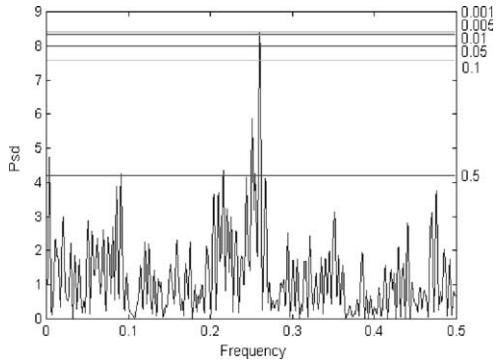

Fig. 6. Lomb's P estimate of SU Cygni.

in the image. The first step is the detection of the objects, a step which, as soon as the quality of the images increases (both in depth and in resolution), becomes much less obvious than it may seem at first glance. The traditional definition of 'object' as a set of connected pixels having brightness higher than a given threshold has in fact several well-known pitfalls. For instance, serious problems are posed by the recognition of low surface brightness galaxies having a central brightness which is often comparable to, or fainter, the detection threshold, and a clumpy shape, thus implying that even though there may be several nearby pixels above the threshold, they can often be unconnected and thus escape the assumed definition. Ferguson (1998) stressed some even stronger limitations of the traditional approach to object detection: (i) the comparison of catalogues obtained by different groups from the same raw material and using the same software shows that, near the detection limits, the results are strongly dependent on the assumed definition of 'object' (Fig. 7); (ii) the object detection performed by the different groups is worse than that attained by even an untrained astronomer by visually inspecting an image or, in other words, many objects that are present in the image are lost by the software, while others that are missing on the image are detected (hereafter spurious objects). Other problems come from the relevant

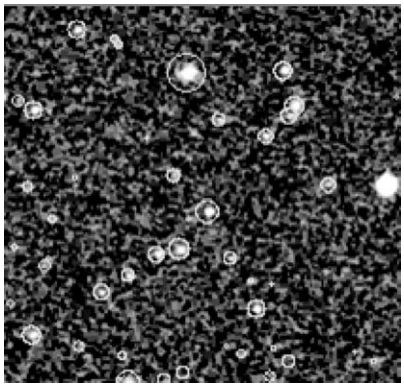

Fig. 7. The spurious objects (created by the software but not really present in the original image) are highlighted.

feature selection which are generally chosen by the astronomer's expertise.

NExt is a NN package implemented by the AstroNeural group, to perform object detection, deblending and star/galaxy classification. In the application of NExt we can identify three main steps: segmentation, feature extraction and classification.

4.2.1. Segmentation

The first step of the procedure consists in an optimal compression of the redundant information contained in the pixels, via a mapping from pixels intensities to a subspace individuated through Principal Component Analysis (= PCA, Fig. 8). From a mathematical point of view, in fact, the segmentation of an image F is equivalent to splitting it into a set of disconnected homogeneous (accordingly to an uniformity predicate P) regions $S_1, S_2, \ldots, S_n$ in such a way that their union is not homogeneous: $\bigcup S_i = F$ with $S_i \bigcap S_j = 0$, $i \neq j$, where $P(S_i) = \text{true} \forall i$ and $P(S_i \bigcup S_j) = \text{false}$ when S_i is adjacent to S_j.

Since the attribution of a pixel to either the 'background' or the object classes depends on both the pixel value and the values of the adjacent pixels, NExt uses a $(n \times n)$ mask (with $n = 3$ or 5) and, in order to lower the dimensionality of the input pattern, an unsupervised PCA NN is used to identify the M (with $M \ll n \times n$) most significant features.

This M-dimensional projected vector becomes the input for a second non-linear NN which classifies pixels into classes. In this respect, we have to stress that non-linear PCA NNs based on a Sigmoidal function outperform linear PCA NNs since they achieve a much better separation of faint objects close to the detection limit of the image (Fig. 9). Linear PCA, in fact, produce distributions with very dense cores (i.e. background and faint objects) and only a few points spread over a wide area (i.e. luminous objects), while non-linear PCAs, on the opposite, produce better sampled distributions and a better contrast between faint and bright objects.

After this step, the principal vectors can be used to project each pixel in the eigenvector space. An unsupervised NN is then used to classify pixels into a few classes (on

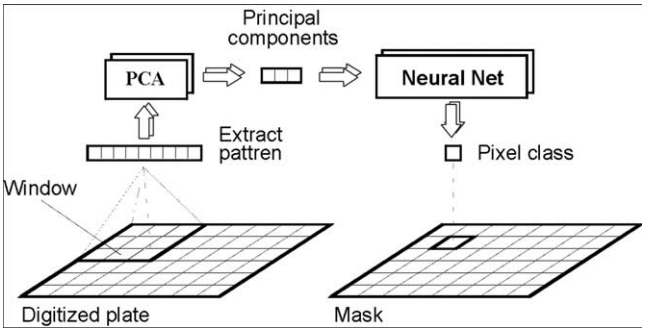

Fig. 8. The overall scheme of the segmentation strategy adopted in NExt.

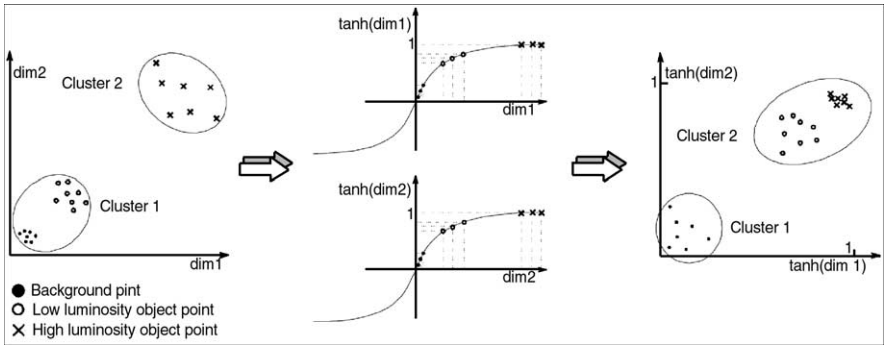

Fig. 9. Simplified scheme of the different performances of linear and non-linear NNs in increasing the contrast between background pixels and faint objects pixels.

average 6, since fewer classes produce poor classifications and more classes produce noisy ones). In all cases, however, only one class represents the background. The classes corresponding to the non-background pixels are then merged together to reproduce the usual object/non-object dichotomy. In order to select the proper NN architecture, we tested various Hierarchical and Hybrid unsupervised NNs and the best performing turned out to be the neural-gas (NG), the multi-layer neural gas (MLNG), the multi-layer self-organizing map (ML-SOM) and the GCS + ML neural gas (Andreon et al., 2000; Tagliaferri, Capuano, & Gargiulo, 1999b).

4.2.2. Object detection performances

Once the objects have been detected, NExt measures a first set of parameters (namely the photometric baricenter, the peak intensity and the flux integrated over the area assigned to the object by the mask). These parameters are needed to recognize partially overlapping objects (from the presence of multiple intensity peaks) and to split them. In NExt, multiple peaks are searched at several position angles after compressing the dynamical range of the data (in order to reduce the spurious peaks produced by noise fluctuations) and, once a double peak has been found, objects are split perpendicularly to the line joining the peaks. A problem which is often overlooked in most packages is that the deblending of multiple (more than two components) objects introduces spurious detections: the search for double peaks and the subsequent splitting produces in fact in each segment of the image a spurious peak which is identified as an additional component in the next iteration (Fig. 10).

In order to correct for this effect, after the decomposition, NExt runs a re-composition loop based on the assumption that true celestial objects present a rapidly decreasing luminosity profile and that, therefore, erroneously split objects will present a strong difference on the two corresponding sides of adjacent masks. After deblending, contour regularization takes place and astrometric and photometric parameters are measured.

One main problem in testing the performances of an extraction algorithm is that the comparison of catalogues obtained by different packages leads often to ambiguous results: unless obvious biases are at work, in the case of conflict it is difficult if not impossible, to decide which algorithm is correct and which is not. We therefore tested several packages (NExt, S-Extractor and FOCAS) on a North Galactic Pole field extracted from the Digitized Palomar Observatory Sky Survey (hereafter DPOSS, Weir, Fayyad, & Djorgovski, 1995). In this case, a deeper and accurate catalogue obtained from high resolution and deeper plates taken at the CFHT telescope was available (Infante & Pritchet, 1992, hereafter IP92) and allowed to define 'True' objects. All packages were run on the same DPOSS region covered by the IP92 catalogue and results were compared. We have to stress that, since in using S-Extractor the choice of the parameters is not critical, we adopted the default values.

The results are presented in Figs. 11 and 12. In both figures, the upper part shows the number of objects in the IP92 catalogue (it is clearly visible that the IP92 catalogue is complete to almost two magnitudes below the DPOSS completeness limit). The lower panel of Fig. 11 gives the fraction (True/Detected) of objects detected by the various NNs and by S-Extractor and shows that all implementations are more or less equivalent in detecting True objects (the best performing being S-Extractor and MLNG5 (where the 5 denotes the 5×5 mask implementation of the MLNG NN). Very different are, instead, the performances in detecting 'false' or spurious objects, id est objects which are not in the IP92 catalogue but are detected on the DPOSS material. In this case, NNs outperform S-Extractor producing in same cases (i.e. MLNG5) up to 80% less spurious detections.

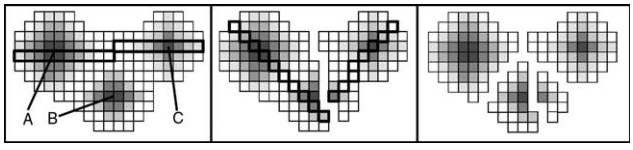

Fig. 10. Example of how some packages erroneously split a triple source into four components.

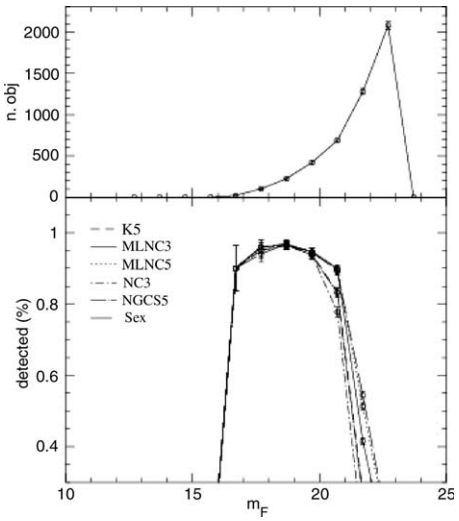

Fig. 11. Comparison of the performances of different NNs architectures plus S-Extractor in detecting True objects.

4.3. Data mining in the catalogue space

As stressed in Section 1, data mining in the catalogue space covers an almost unlimited field of applications, from classification to pattern recognition, to trend analysis, etc. In what follows we shall focus on a few selected cases which appear to be most instructive in order to show the potentialities of NNs.

4.3.1. Star/galaxy classification with NExt

One of the most commonly encountered applications of NN for astronomical data mining is the problem usually called 'star–galaxy' classification. To be more precise, given a specific instrumental setup and therefore the associated Point Spread Function or PSF, the problem consists in recognizing those objects whose light distribution cannot be distinguished from the PSF (unresolved or 'stars') from those whose light distribution appears extended (resolved or 'galaxies').

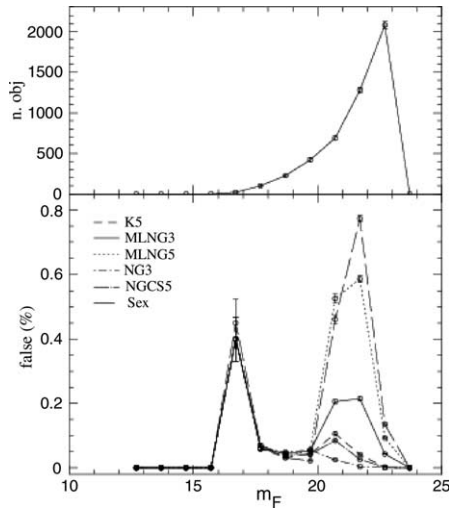

Fig. 12. Comparison of the performances of different NNs architectures plus S-Extractor in detecting False objects.

Human experts can usually classify objects either directly from the appearance of the objects on an image (either photographic or digital) or from the value of some finite set of derived parameters via diagnostic diagrams (such as magnitude versus area). This approach, however, is too much time-consuming and much too dependent on the 'know how' and personal experience of the observer: (i) the choice of the most suitable parameters varies greatly from author to author, making comparisons difficult if not impossible, and (ii) regardless the complexity of the problem, owing to the obvious limitations of representing three or more dimensions in space on a two-dimensional graph, only two features are often considered.

As stressed above, the star/galaxy problem can be addressed by a wide variety of neural methods. In what follows, we shall focus our attention on those tested within the AstroNeural collaboration. The first step in the NExt approach consists in identifying among the measured parameters those which are most significant for the classification task. In order to select the relevant ones, we adopted the sequential backward elimination strategy (Bishop, 1995) which consists in a series of iterations eliminating at each step the feature which is less significant for the classification. Extensive testing showed that the best performances in star/galaxy classification are obtained by using six features only (two radii, two gradients, the second total moment and a Miller and Coe ratio). Star/galaxy classification was performed by means of a MLP. In order to teach the MLP how to classify galaxies, we divided the data set into three subsets, namely the training, validation and test sets. Learning was performed on the training set and the early stopping technique is used to avoid overfitting (Bishop, 1995). As a comparison classifier we used S-Extractor (Bertin & Arnouts, 1996) which also uses a NN (a MLP) trained on a set of 10^6 simulated images to attach to each object a 'stellarity index' ranging from 0 (galaxies) to 1 (stars). We wish to stress here that NExt is (to our knowledge) the only package trained on real, noisy data. The training was then validated on the validation set and tested on the test set. Results are shown in Fig. 13 and confirm that NExt misclassifies less galaxies than S-Extractor, whose performance were optimized by the use of the validation set (for a fair comparison).

4.3.2. Star/galaxy classification with fuzzy similarities

The catalogue used in our preliminary experiments consists of 231.000 labeled Stars/Galaxies (S/G) objects and each object has 16 features. From the experience we know that for each object is enough to select the seven most significant features. In our experiment we choose randomly only 10.000 objects. It is interesting to compare in Fig. 14(a) and (b), the distribution of the feature values of the catalog objects after the preprocessing step described below. Our

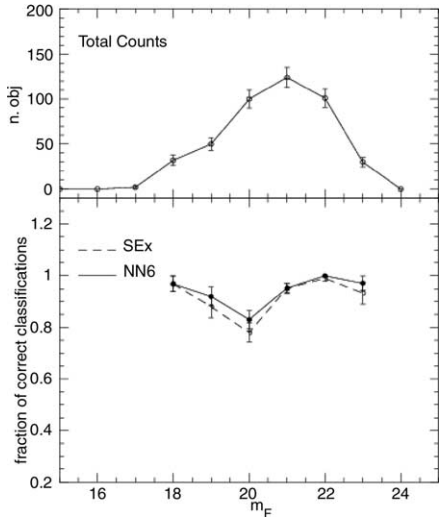

Fig. 13. Comparison of the performances of MLNG5 and S-Extractor in classifying stars and galaxies.

method is composed of the following steps: (1) normalization of object features (fuzzy sets constructions); (2) unsupervised prototypes selection from the object catalog (one for each stars/galaxies class); (3) calculation of the TFS between the prototypes and each object; (4) S/G classification by using the computed TFS. In the first step we apply a preprocessing to the selected features of the data in order to obtain a fuzzy set for each feature. In detail, the normalization of the xth object with respect to ith feature is accomplished by

$$\mu_{\text{feature}_i}(x) = \frac{(\text{feature}_i(x) - (\text{lowest feature}_i))}{(\text{highest feature}_i) - (\text{lowest feature}_i))} \tag{41}$$

In the second step the selection of two prototypes is realized by means of the SOM (Kohonen, 1995). In the first level of the SOM we set the number of the nodes to be the number of object features. The second level is composed by two nodes, one for each class (S/G). In this way we obtain two prototypes corresponding to the SOM weights. In the next step, we calculate the similarity between the objects of the catalog and the two prototypes using the TFS described above. Finally, the object labeling is obtained by assigning an object to the star or to the galaxy classes on the basis of the maximum similarity with respect to each prototype. In order to test our methodology we applied the described algorithms to the labeled catalogue previously described. In this preliminary experiment we use the chosen objects for

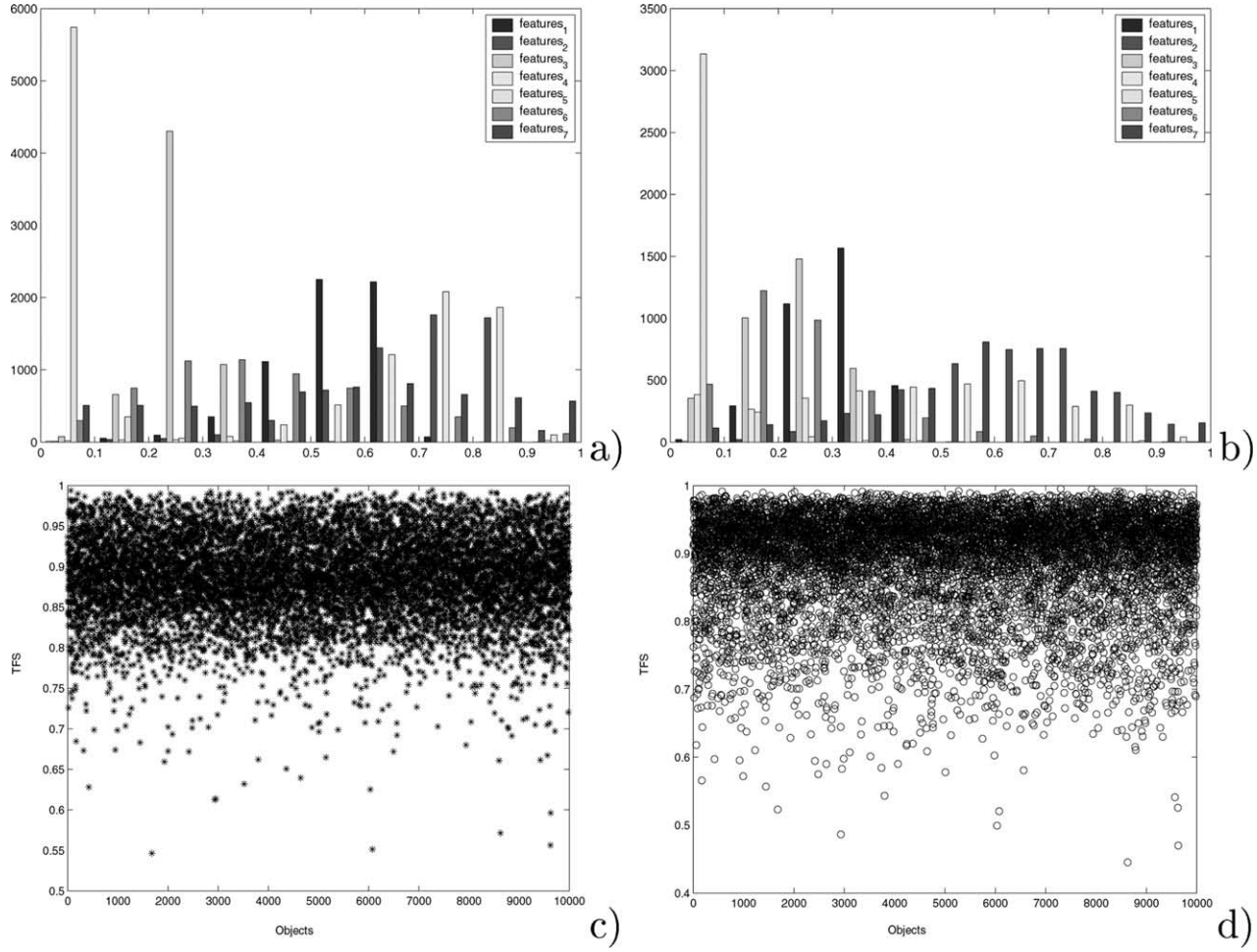

Fig. 14. Up down and left right: (a) Stars histogram; (b) Galaxies histogram; (c) Stars TFS; (d) Galaxies TFS.

Table 1
Confusion matrix of the experimental results of *S*/*G* separation

	G	*S*
G	2867	590
S	1672	4871

both the prototypes selection and the test phases, obtaining a rate of correct classification of 77.38% which corresponds to the correct classification of 7738 objects of the catalog. This results are shown in Table 1 as a confusion matrix. More in detail, we are able to guess the correct classification of 2867 galaxies and 4871 stars. In Fig. 14(c) and (d) we show the TFS computed for the Stars prototype and Galaxies prototype, respectively.

4.3.3. Photometric redshifts of galaxies

The redshift of a galaxy (namely its recession velocity from the observer) is of paramount relevance in observational cosmology since it often provides the only way to estimate its distance. The redshift may be measured either spectroscopically (but this procedure, even though accurate, is very time consuming) or photometrically (less accurate and subject to many systematic errors but easier to apply to large samples of objects; (Baum, 1962; Pushell, Owen, & Laing, 1982). The recent availability of large datasets containing homogeneous and multi-band photometric data for large samples of objects together with spectroscopic redshifts for a non-negligible subsample, has opened the way to the use of supervised NN for redshift evaluation.

In a recent paper (Tagliaferri et al., 2002), we used this type of approach to study the Sloan Dital Sky Survey Early Data Release (SDSS-EDR) (Stoughton et al., 2002): a data set containing photometric data for some 16 million galaxies and spectroscopic redshift for a subsample of about 50.000 objects distributed over a relatively large redshift range. The method can be summarized as follows: (i) an unsupervised SOM is used to cluster the data in the training, validation and test set in order to ensure a complete coverage of the input parameter space; (ii) a MLP in Bayesian framework is used to estimate the photometric redshifts; (iii) a labelled SOM is used to derive the completeness and the contamination of the final catalogues.

In order to build the training, validation and test sets, a set of parameters (magnitudes in the bands u, g, r, i, z, both total and petrosian magnitudes, petrosian radii, 50% and 90 petrosian flux levels, surface brightness and extinction coefficients, (Stoughton et al., 2002) was extracted from the SDSS-EDR. Due to the highly non-homogeneous distribution of the objects in the redshift space (with a marked cutoff at $z \sim 0.6$), the density of the training points dramatically decreases for increasing redshifts, and therefore, unless special care is paid to the construction of the training set, all networks will tend to perform much better in the range where the density of the training points is higher. Therefore, in order to achieve an optimal training of the NNs, two different approaches to the construction of the training, validation and test sets were implemented: the uniform sampling and the clustered sampling (via K-means and/or SOM). In both cases the training set data are first ordered by increasing redshift, then, in the case of uniform sampling, after fixing the number of training objects (which needs in any case to be smaller than 1/3 of the total sample) objects are extracted following a decimation procedure. This method, however, is undermined by the fact that the input parameter space is not necessarily uniformously sampled, thus causing a loss in the generalization capabilities of the network. In the clustered sampling method, objects in each redshift bin are first passed to a SOM or a K-means algorithm which performs an unsupervised clustering in the parameter space looking for the most significant statistical similarities in the data. Then, in each bin and for each cluster, objects are extracted in order to have a good sampling of the parameter space. The experiments were performed using the NNs in the Matlab and Netlab Toolboxes, with and without the Bayesian framework. All NNs had only one hidden layer and the experiments were performed varying the number of the input parameters and of the hidden units. Extensive experiments lead us to conclude that the Bayesian framework provides better generalization capabilities with a lower risk of overfitting, and that an optimal compromise between speed and accuracy is achieved with a maximum of 22 hidden neurons and 10 Bayesian cycles. In Table 2, we summarize some of the results obtained from the experiments and, in Fig. 15, we compare the spectroscopic redshifts versus the photometric redshifts derived for the test set objects in the best experiment.

In practical applications, one of the most important problems to solve is the evaluation of the contamination of the final photometric redshift catalogues or, in other words, the evaluation of the number of objects which are erroneously attributed a z_{phot} significantly (accordingly to some arbitrarily defined treshold) different from the unknown z_{spec}. This problem is usually approached by means of extensive simulations. The problem of contamination is even more relevant in the case of NNs based methods, since NNs are necessarily trained only in a limited range of redshifts and, when applied to the real data, they will produce misleading results for most (if not all) objects which ‘in the real word’ have redshifts falling outside the training range. This behavior of the NNs is once more due to the fact that while being good interpolation tools, they have very little, if any, extrapolation capabilities. Furthermore, in mixed surveys, the selection criteria for the spectroscopic sample tend to favour the brightest (and, on average, the closer) galaxies with respect to the fainter and more distant ones and, therefore, the amount of contamination encountered, for instance, in the test set sets only a lower limit to the percentage of spurious redshifts in the final catalogue.

Table 2
Column 1: higher accepted spectroscopic redshift for objects in the training set; column 2: input (hence number of input neurons) parameters used in the experiment; column 3: number of neurons in the hidden layer; column 4: interquartile errors evaluated on the test set; column 5: number of objects used in each of the training, validation and test set

Range	Parameters	Neu.	Error	Objects
$z < 0.3$	r, u-g, g-r, r-i, i-z	18	0.029	12000
$z < 0.5$	r, u-g, g-r, r-i, i-z	18	0.031	12430
$z < 0.7$	r, u-g, g-r, r-i, i-z	18	0.033	12687
$z < 0.3$	r, u-g, g-r, r-i, i-z, radius	18	0.025	12022
$z < 0.5$	r, u-g, g-r, r-i, i-z, radius	18	0.026	12581
$z < 0.7$	r, u-g, g-r, r-i, i-z, radius	18	0.031	12689
$z < 0.3$	r, u-g, g-r, r-i, i-z, radius, petrosian fluxes, surf. br.	22	0.020	12015
$z < 0.5$	r, u-g, g-r, r-i, i-z, radius, petrosian fluxes, surf. br.	22	0.022	12536
$z < 0.7$	r, u-g, g-r, r-i, i-z, radius, petrosian fluxes, surf. br.	22	0.025	12680

To be more specific: in the SDSS-EDR spectroscopic sample, over a total of 54,008 objects having $z > 0$, only 88, 91 and 93% had redshift z lower than, respectively, than 0.3, 0.5 and 0.7. To train the network on objects falling in the above ranges implies, respectively, a minimum fraction of 12, 9 and 7% of objects in the photometric data set having wrong estimates of the photometric redshift. On the other hand, as we have shown, the higher is the cut in redshifts,

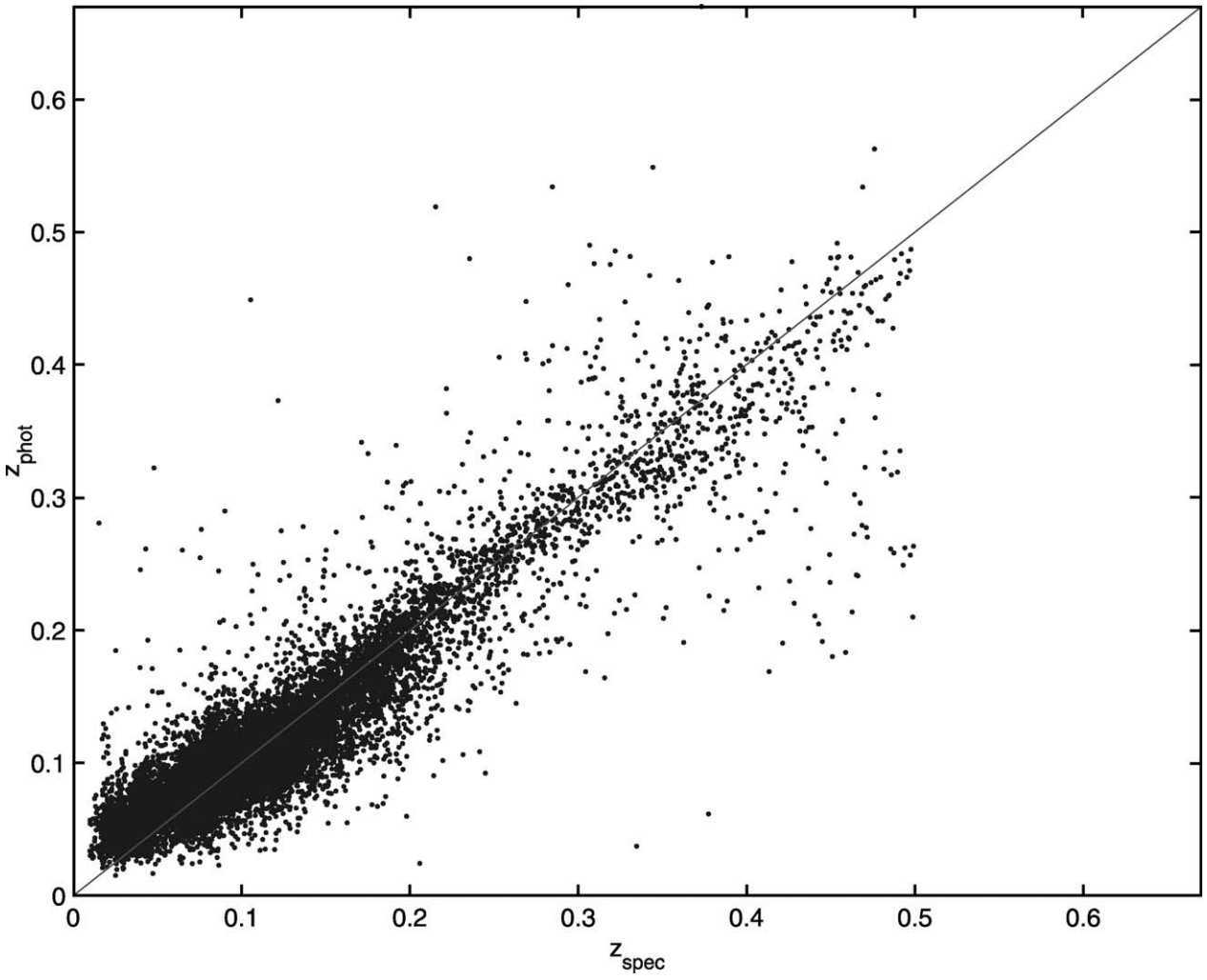

Fig. 15. Photometric versus spectroscopic redshifts obtained with a Bayesian MLP with 2 optimization cycles, 50 learning epochs of quasi-Newton algorithm and 5 inner epochs for hyperparameter optimization. Hyperparameters were initialized at $\alpha = 0.001$ and $\beta = 50$.

the lower is the accuracy and a compromise between these two factors needs to be found on objective grounds.

An accurate estimate of the contamination may be obtained using unsupervised SOM clustering techniques over the training set.

Fig. 16 show the position of the BMU as a function of the redshift bin. Each exagon represents a neuron and it is clearly visible that low redshift ($z < 0.5$) tend to activate neurons in the lower right part of the map, intermediate redshift ones ($0.5 < z < 0.7$) neurons in the lower left part and, finally, objects with redshift higher than 0.7 activate only the neurons in the upper left corner. The labeling of the neurons (shown in the upper left map) was done using the training and validation data sets in order to avoid overfitting, while the confidence regions were evaluated on the test set.

Therefore, test set data may be used to map the neurons in the equivalent of confidence regions and to evaluate the degree of contamination to be expected in any given redshift bin. Conversely, when the network is applied to real data, the same confidence regions may be used to evaluate whether a photometric redshift correspondent to a given input vector may be trusted upon or not.

The above derived topology of the network is also crucial since it allows to derive the amount of contamination. In order to understand how this may be achieved, consider the case of objects which are attributed a redshift $z_{\text{phot}} < 0.5$. This prediction has a high degree of reliability only if the input vector activates a node in the central or right portions of the map. Vector producing a redshift $z_{\text{phot}} < 0.5$ but activating a node falling in the upper left corner of the map are likely to be misclassified. In Fig. 17, we plot the photometric versus spectroscopic redhift for all test set objects having $z_{\text{phot}} < 0.5$ and activating nodes in the correct region of the map.

As it can be seen, out of 9270 objects with $z_{\text{phot}} < 0.5$, only 39 (id est, 0.4% of the sample) have discordant spectroscopic redshift. A confusion matrix helps in better quantifying the quality of the results. In Table 3, we give the confusion (or, in this case, 'contamination') matrix obtained dividing the data in three classes accordingly to their spectroscopic redshifts, namely class I: $0 < z < 0.3$, class II: $0.3 < z < 0.5$, class III: $z > 0.5$. The elements on the diagonal are the correct classification rates, while the other elements give the fraction of objects belonging to a given class which have been erroneously classified into an other class.

As it can be seen, in the redshift range (0,0.3), 95.4% of the objects are correctly identified and only 4.6% is attributed a wrong redshift estimate. In total, 94.2% are correctly classified. By taking into account only the redshift range $0 < z < 0.5$, this percentage becomes 97.3%. From the confusion matrix, we can therefore derive a completeness of 97.8% and a contamination of about 0.5%.

A simple step filter applied to the accepted BMU's allows therefore to optimise the filter performances. For instance, it allows to choose whether to minimize the number of misclassified objects (thus reducing the completeness) or to minimize the classification error in a given redshift bin more than in another one.

Summarizing: on the SDSS-EDR data set, the best result (robust error $= 0.21$) was obtained by a MLP with 1 hidden layer of 22 neurons, after five Bayesian cycles. Once they are trained, NNs are extremely effective in terms of computational costs.

4.4. Noise identification and removal in gravitational waves detectors

One of the most common problems in applied science is the extraction of faint signals from noisy backgrounds. This is especially true in analyzing the data which will be produced by the planned gravitational waves (GW) detectors such as VIRGO (Bradaschia et al., (1989, 1992, 1995) and LIGO (Vogt, Drever, Raab, & Thorne, 1989) which have noise broadband colored with the contribution of some spectral peaks, but can also be not stationary in time and not Gaussian. Furthermore, if we take into account the extremely high sensitivities required (about $10^{-19}\ Hz^{-1/2}$ in a band of about 1 kHz) and the fact that the signal is limited at very low frequency by seismic noise and at high frequencies by shot noise, we have a rather complete picture of the complexity of the problem posed by the analysis of GW data. NNs have recently been applied to the whitening of the main known sources of noise present at the interferometer output, such as violin resonances of the mirror suspensions, main power harmonics, ring down noises from servo control systems, electronic noise and environmental noise (such as electromagnetic, acoustic and seismic noise). The test data were collected in 1994 with the IFO-DMRO (interferometer differential mode readout) at the 40 m prototype Caltech interferometer.

First, by using a two sided surrogates time series and a reversal symmetry third order statistics (based on the TISEAN software package[4] (Hegger, Kantz, & Schreiber, 1999) we found that the null hypothesis test of a rescaled linear Gaussian random process had to be rejected at the 95% level of significance (in conflict with the usual assumption that GW data may be dealt with standard linear modeling techniques).

The system identification consists in finding the input-output mapping of a dynamic system. A discrete-time multi-input multi-output (MIMO) non-linear system (or a continuous-time sampled-data system) can be described by a set of non-linear difference equations of the form (*input–output* representation):

$$y(n) = \mathscr{F}[\mathbf{y}(n-1), \mathbf{y}(n-2), \ldots, \mathbf{y}(n-n_y), \mathbf{u}(n - n_d), \ldots, \mathbf{u}(n-n_u-n_d+1)] \quad (42)$$

[4] Available free of charge at the site: http://www.physik3.gwdg.de/tstool

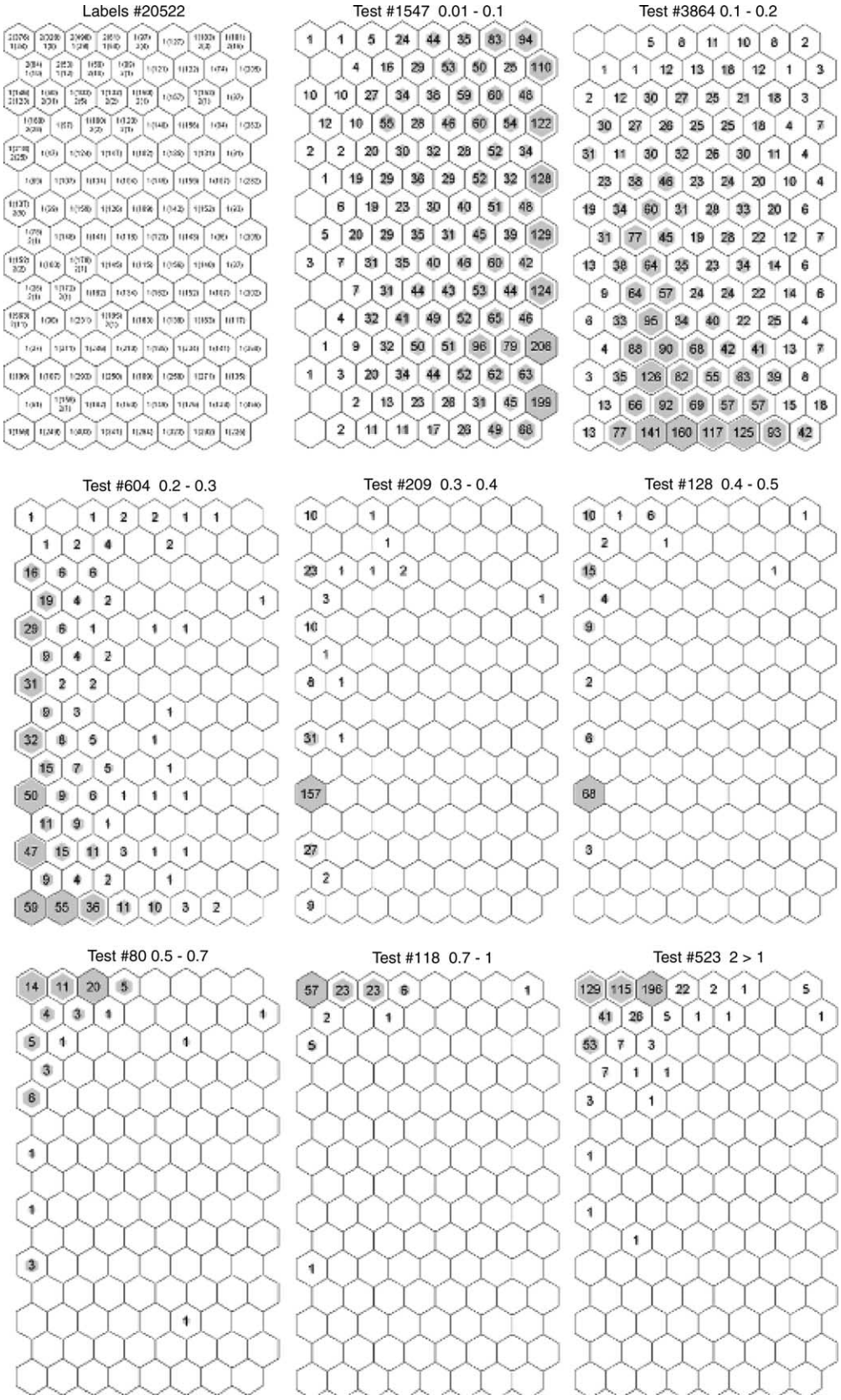

Fig. 16. Maps of the neurons activated by the input data set. Exagons represent the NNnodes. In the map in the upper left corner, for a given node, the notation $n(m)$ must be read as follows: n is the class ($n = 1$ meaning $z < 0.5$ and $n = 2$ meaning $z > 0.5$) and m is the number of input vectors of the correspondent class which have activated that node. This map was produced using the training and validation data sets. The other maps, produced each in a different redshift bin, indicate how many input vectors from the test data set activated a given node.

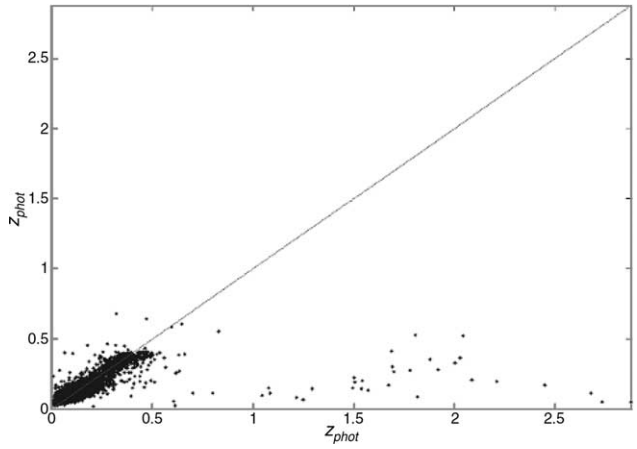

Fig. 17. Spectroscopic versus photometric redshifts for objects belonging to Class I and Class II. Outliers are misclassified galaxies.

where $\mathbf{u} \in R^M$ is the input vector and $\mathbf{y} \in R^N$ is the output vector; n_u and n_y the maximum lags of the input and output vectors, respectively, and n_d the pure time-delay (or dead-time) in the system. This is called a NARX model (non-linear autoregressive with exogenous input) and it can be shown that a wide class of discrete-time systems can be put in this form (Chen & Billings, 1989). From this starting point, we further analyzed the problem by searching for a structure allowing the creation of a non-linear model. Every discrete-time (or continuous sampled-data, as in our case) dynamical system can be described by a set of difference equations which describe the evolution of the system state:

$$\mathbf{x}(t+1) = \mathscr{F}[\mathbf{x}(t)] \tag{43}$$

In general, the form of the functional $\mathscr{F}(\cdot)$ is not known but given some measurements of the state variables $\mathbf{x}(t)$, it is possible to build a fair *approximation*.

The process which generates the observed data $s(t)$ can be described by its state vector $\mathbf{x}(t)$, which usually cannot be observed. It turns out, however (Takens, 1981) that a space formally equivalent to the original state space can be reconstructed using the data sequence $s(t)$. This *phase space reconstruction* provides components of the original space starting from elements of $s(t)$:

$$x(t) = [s(t), s(t-\tau_k), s(t-2\tau_k), \ldots] \tag{44}$$

where the delay τ_k can be computed as an integer multiple of the sampling time τ_s of the system:

$$\tau_k = k\tau_s. \tag{45}$$

Table 3
Confusion matrix for the three classes described in the text

	Objects	Class I (%)	Class II (%)	Class II (%)
Class I	9017	95.4	2.96	1.6
Class II	419	6.4	76.6	16.9
Class III	823	3.8	2.1	94.2

k can, at least in principle, be derived from the autocorrelation function, as it is done when building linear models. A theoretical information measure of correlation, which does not depend on the properties of the (unknown) equations describing the dynamics of the system, is the *average mutual information*:

$$I(k) = \sum_{t=1}^{N} P(s(t), s(t-k\tau_s)) \log\left[\frac{P(s(t), s(t-k\tau_s))}{P(s(t))P(s(t-k\tau_s))}\right] \tag{46}$$

which gives the amount of information in bits learned about $s(t-k\tau_s)$ from the knowledge of $s(t)$. P is the empirical distribution of the data. What value of k should be chosen? In Abarbanel (1996) it is suggested, as a prescription, the value k_m (the 'time lag' os system) where the first minimum of $I(k)$ occurs.

How many delays are necessary to completely describe (*unfold*) the phase space of the system? We must determine a dimension for the space in which neighbor states $\mathbf{x}(t)$ are near because of dynamics and not because we observe them in a low dimensional space. The observed data can be seen as a *projection* of the state space onto a lower dimensional space (in our case, the output of the interferometer is a scalar sequence); the effect is that there is an ambiguity because the orbits of the system will intersect. The dimension in which all the orbits of the system unfold is the *embedding dimension*. To evaluate the embedding dimension we use the Cao's (1997) method implemented in the TSTOOL Toolbox for Matlab©. Knowing the time lag k_m and the embedding dimension d, we can describe the state of the system, as:

$$x(t) = [s(t), s(t-\tau_m), s(t-2\tau_m), \ldots, s(t-(d-1)\tau_m)]. \tag{47}$$

Note that strictly speaking these results assume a noiseless and infinite-length sequence. Nonetheless, they can be used in practice since the effects of noise and finiteness are mainly an overestimate of the lag and the embedding dimension, leading to an unnecessarily complex description of the dynamics. The last step is then the setup of a non-linear, NN-based model.

We used several criteria to evaluate the goodness of noise identification and the whitening procedure according to the following performance index:

- The spectral flatness measure of a power spectral density (PSD) can be defined as (Kay, 1988):

$$\xi = \frac{\exp\left(\frac{1}{f_s}\int \ln(S(f))\mathrm{d}f\right)}{\frac{1}{f_s}\int S(f)\mathrm{d}f} \tag{48}$$

The integrals extend over the Nyquist frequency bandwidth i.e. $-f_s/2 \div f_s/2$, where f_s is the sampling frequency. The ξ parameter varies between 0 and 1 according to the flatness of the spectrum. If $S(f)$ is very

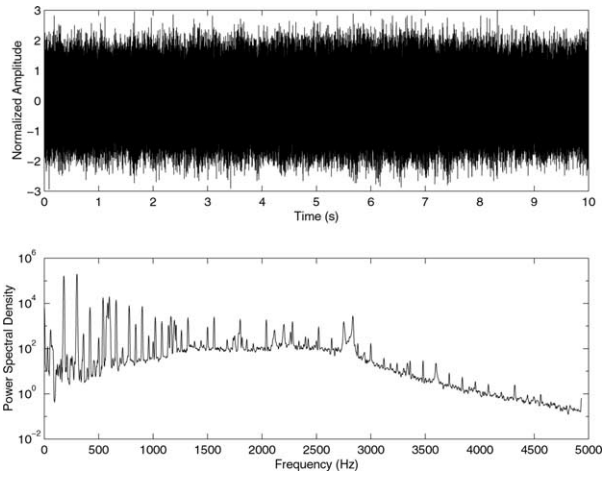

Fig. 18. The time plot (top fig.) and the Power Spectrum (bottom fig.) of a 10 s sample of the data frame used in the training stage before the preprocessing.

peaked ξ is nearly 0 whilst if the spectrum is flat $\xi \simeq 1$. It is possible to show that the ratio of ξ indexes before and after the whitening process equals the ratio of the respective autocorrelation functions for zero lags.
- The χ^2 test at 95% of significativity.
- Prediction residual analysis.
- The Kolmogorov–Smirnov test (KStest, Chakravart, 1997) to evaluate if a sample of data comes from a specific distribution or if two data set are sampled from the same distribution.

In order to whiten the Caltech data, we implemented a MLP which defines a *model* of the dynamics of the interferometer by using only its output. In Fig. 18 is shown one of the data frame used in the training step.

The procedure was the following: first, the data were pre-processed using statistical analysis tools to have an insight on their gaussianity and stationarity; then

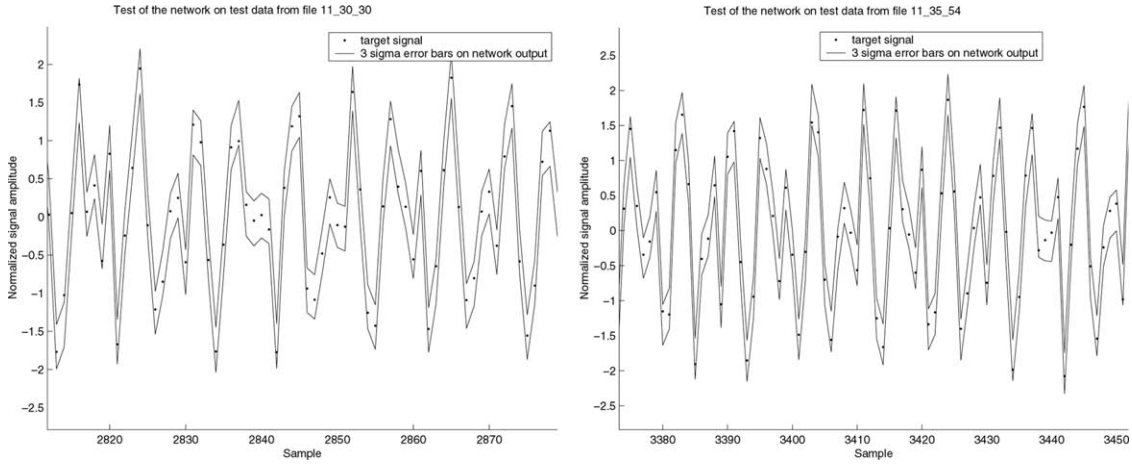

Fig. 19. Segments of LIGO test sequences with 3σ error bars.

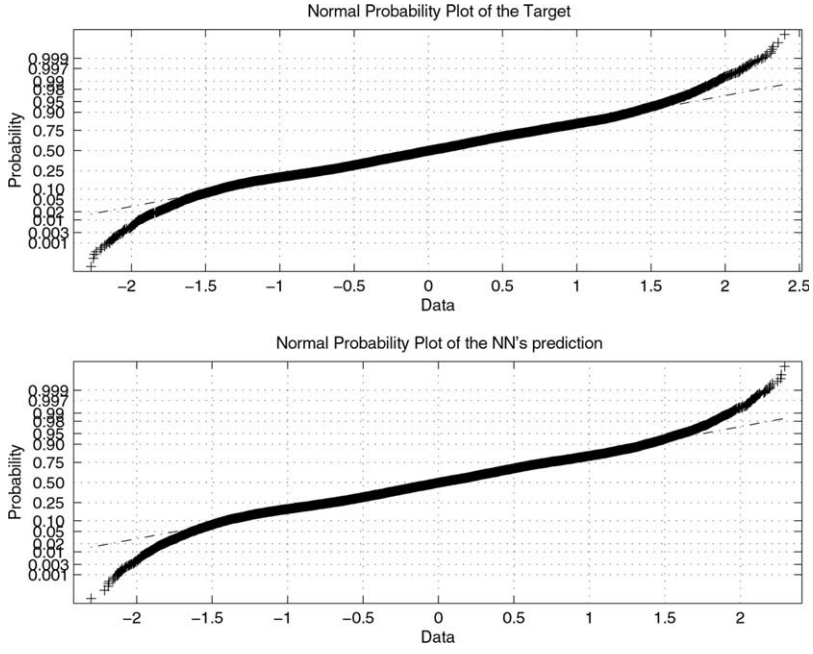

Fig. 20. Normality plot of input data frame used for the training step (top) and normality plot of NNs prediction (bottom).

Table 4
Results of the K–S test on some chunks of data

Data file	Test output
11_22_24	0.998
11_27_48	0.932
11_3-_3-	0.997
11_33_12	0.997
11_35_54	0.963

a pre-processing using non-linear analysis techniques to extract information on the underlying data-generating process was performed; then a probabilistic Bayesian approach was used to create the model thus allowing the exploration of a *model space*, rather than the use of a single 'optimal' model as it usually happens with linear modeling techniques.

The data used to train the model were a sequence of six seconds and the original sampling rate was downsampled to 2500 Hz and the model was tested on a different sequence of about 1 min. Before feeding the data to the network, they were pre-processed to find a suitable time lag and the embedding dimension. The results of the analysis were the following: the first minimum of AMI is $2\tau_s$ and the embedding dimension is chosen as the first point in which the Cao's (1997) function reaches the plateau at $d = 20$.

The adopted trained MLP consisted in: 20 input units, one output unit with linear activation function, 20 hidden units with hyperbolic tangent activation function. Furthermore, 25 regularization parameters were used: one for every group of connections from each input unit (20), one for each bias connection for input and hidden units (2), one for the group of connections to the output unit.

As it can be seen from Fig. 18, even in presence of a system with a rather complex evolution, our simple network was rather successful in predicting the system dynamics. In Fig. 19 two chunk of data used in the testing step are shown. As it can be seen the prediction is bracketed between the 3σ error bars and so the network exhibits good generalization properties.

Furthermore, as it can be seen in Fig. 20 the normality plot of the data frame used in the training step (up) and the normality plot of the NNs prediction (bottom) turned out to be very similar. In other words, with our approach we do not introduce higher order cumulants in the output statistic of the data as it is possible to see from residuals and from the KStest we performed.

Finally, the KS test shows that there is not a statistically significative difference (at a 95% level) between them (see Table 4) even if we use several chunks of data. Only in one case (file 11-27-48) the hypothesis can be accepted at 93% of significant level. We performed also a χ^2 test and we obtained that the null hypothesis on the two probability distributions can be accepted at 95% level of significance. Note that the model could be made more reliable by using more neurons and/or by training the NNs for more epochs (without the risk of overfitting owing to the Bayesian learning procedure we adopted).

The whitening spectrum is nearly flat (Fig. 21), in fact the spectral flatness measure is $\xi = 0.7$. We performed a χ^2 test at 95% of significance to verify the hypothesis that the whitening sequence is a Gaussian distribution. In our case the $\chi^2_{\text{crit.}} = 9.49$ with four degrees of freedom and we obtained $\chi^2_{\text{rid}} = 1.47 < \chi^2_{\text{crit.}}$ and so we can accept the null hypothesis. Furthermore we performed a robust non-parametric KS test using the Kuiper statistic at 95% level of significance and obtained that the null hypothesis on the whitening sequence extracted from Gaussian distribution can be accepted.

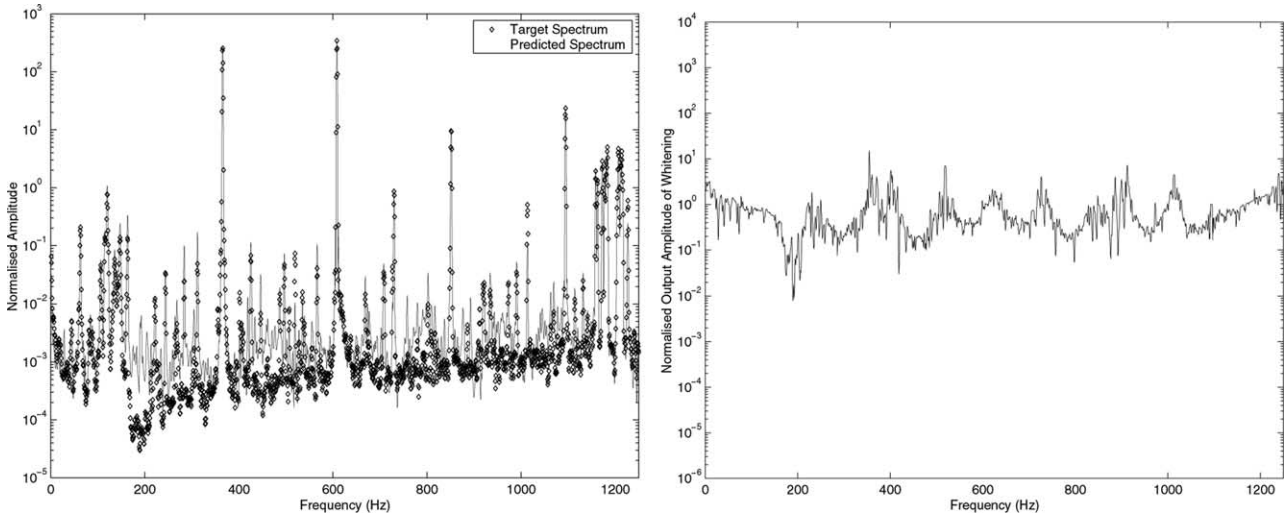

Fig. 21. The spectra of output and target sequences (file 11-30-30) are shown for DMRO Caltech 40 m interferometer(sx). The whitening spectrum (dx) exhibits $\xi = 0.7$.

5. Conclusions

In this paper we summarized the most relevant properties of the main types of NN used for astronomical applications focusing our attention to those models which are most commonly encountered. We also outlined some applications studied by the AstroNeural collaboration in the context of various problems. The above paragraphs should have convinced the general reader of the enormous potentialities which NN have in the most disparate fields of astronomical data reduction analysis and mining. It should also have become evident that neural tools, in order to be truly effective, need to be specifically tailored to any specific application and that, therefore, they need extensive testing aimed to define the most apt architecture and the optimal choice of the training, validation and test sets (for the supervised methods).

It is therefore highly realistic to conclude that the use of NN and other soft computing tools will become much more common once the Astrophysical Virtual Observatory (AVO) will become fully operational, thus bringing to the community an unprecedented wealth of high quality and heterogeneous data.

Acknowledgements

The AstroNeural collaboration is financed through several grants: two grants from the Italian 'Ministero della Ricerca Scientifica e Tecnologica' (MURST), a grant from the Italian Space Agency (ASI), and some minor grants from Regione Campania. The authors also acknowledge the LIGO collaboration for the permission to use the data taken from Caltech 40-meter prototype interferometer in November 1994 for our work and the CRoNaRio (CaltechRomeNapoliRio) collaboration for making available their astronomical catalogues.

References

Abarbanel, H. D. I. (1996). *Analysis of observed chaotic data*. Berlin: Springer.

Acernese, F., Barone, F., Ciaramella, A., De Rosa, M., De Rosa, R., Eleuteri, A., Milano, L., & Tagliaferri, R. (2002). A neural network-based approach to noise identification of interferometric GW antenna's: the case of 40 m Caltech laser interferometer. *Classical and Quantum Gravity*, *19*, 3293–3307.

Amari, S., Cichoki, A., & Yang, H. H. (1996). *A new learning algorithm for blind source separation. Neural information processing systems*, Cambridge: MIT Press, pp. 757–763.

Amenomori, M., Ayabe, S., Caidong, et al (Tibet ASGamma Collaboration) (2000). *Primary proton spectrum between 200 TeV and 1000 TeV observed with the Tibet burst detector and air shower array*. Astro-ph/0010292.

Andreon, S., Gargiulo, G., Longo, G., Tagliaferri, R., & Capuano, N. (2000). Wide field imaging. I. Applications of neural networks to object detection and star/galaxy classification. *Monthly Notices Royal Astronomical Society*, *319*, 700–716.

Baccigalupi, C., Bedini, L., Burigana, C., De Zotti, G., Farusi, A., Maino, D., Maris, M., Perrotta, F., Salerno, E., Toffolatti, L., & Tonazzini, A. (2000). Neural networks and the separation of cosmic microwave background and astrophysical signals in sky maps. *Monthly Notices Royal Astronomical Society*, *318*, 769–780.

Bailer-Jones, C. A. L., Gupta, R., & Singh, H. P. (2001). In Gupta (Ed.), *Automated data analysis in astronomy*, Astro-ph/0102224.

Balastegui, A., Ruiz-Lapuente, P., Canal, R (2001). *Reclassification of gamma-ray bursts*. Astro-ph/0108272.

Ball, N. M (2001). *Morphological classification of galaxies using artificial neural networks*, Master Thesis, Astro-ph/0110492.

Barone, F., De Rosa, R., Eleuteri, A., Garufi, F., et al. (1999). A neural network-based ARX model of Virgo noise. In M. Marinaro, R. Tagliaferri, et al. (Eds.), (pp. 171–183). *11th Italian workshop on neural networks WIRN*, Berlin: Springer.

Baum, W. A. (1962). Problems of extragalactic research. *IAU Symposium no. 15*, 390–397.

Bazell, D., & Aha, D. W. (2001). Ensembles of classifiers for morphological galaxy classification. *Astrophysical Journal*, *548*, 219–223.

Bell, A., & Sejnowski, T. (1995). An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, *7*, 1129–1159.

Bertin, E., & Arnouts, S. (1996). SExtractor: software for source extraction. *Astronomy and Astrophysics*, *117*(Suppl.), 393–404.

Birge, B., Walberg, G (2002). A computational intelligence (CI) approach to the precision mars lander problem, Technical Report, North Carolina State Univ.

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford: Clarendon Press.

Borda, F., Roberto, A., Mininni, P. D., Mandrini, C. H., Gómez, D. O., Bauer, O. H., & Rovira, M. G. (2002). Automatic solar flare detection using neural network techniques. *Solar Physics*, *206*, 347–357.

Bradaschia, C. et al (1989, 1992, 1995). VIRGO: proposal for the construction of the very large baseline interferometric antenna for gravitational wave detection, Proposal to the INFN (Italy) and CNRS (France).

Brescia, M., D'Argenio, B., Ferreri, V., Longo, G., Pelosi, N., Rampone, S., & Tagliaferri, R. (1996). Neural net aided detection of astronomical periodicities in geologic records. *Earth and Planetary Science Letters*, *139*, 33–45.

Bussino, S (1999). Gamma-Hadron *discrimination with a neural network in the ARGO-YBJ Experiment*. Astro-ph/9906201.

Cao, L. (1997). Practical method for determining the minimum embedding dimension of a scalar time series. *Physica D*, *110*, 43–50.

Chakravart, L. R. (1997) (*Vol. 1*) (pp. 392). *Handbook of methods of applied statistics*, New York: Wiley.

Chen, S., & Billings, S. A. (1989). *International Journal of Control*, *50*(6), 2151.

Coryn, A.L., Bailer-Jones, C (2000). Stellar parameters from very low resolution spectra and medium band filters: Teff, log *g* and [M/H] using neural networks, Astro-ph/0003071.

Deeming, T. J. (1975). Fourier analysis with unequally-spaced data. *Astrophysics and Space Science*, *36*, 137–158.

Ferguson, H. C. (1998). The hubble deep field. In M. Livio, S. M. Fall, & P. Madau (Eds.), (*Vol. 11*) (pp. 181–192). *STScI Symposium Series*, New York: Cambridge University Press.

Fuentes, O. (2001). Automatic determination of stellar atmospheric parameters using neural networks and instance-based machine learning. *Experimental Astronomy*, *12*, 21–31.

Funaro, M., Oja, E., Valpola, H (2003). Independent component analysis for artefact separation in astrophysical images, Neural Networks (this volume).

Ganguli, S.B., Gavrichtchaka, V. V., Von Steiger, R (2002). Adaptive Systems for Detection and Forecasting of Coronal Mass Ejections From

Solar Mean Magnetic Field, American Geophysical Union Spring Meeting, SH52A-01.

Goderya,, Shaukat, N., & Lolling, L. M. (2002). Morphological classification of galaxies using computer vision and artificial neural networks: a computational scheme. *Astrophysics and Space Science*, *279*, 377–387.

Gulati, R. K., & Altamirano, L. (2001). Artificial neural networks in stellar astronomy. In Aguilar (Ed.), *Focal points in Latin American astronomy* (*Vol. 85*) (pp. 85*Revista Mexicana de Astronomia y Astrofisica Serie de Conferencias*.

Hajek, P. (1998). *Metamathematics of Fuzzy Logic*. Dordrecht, The Netherlands: Kluwer.

Hegger, R., Kantz, H., & Schreiber, T. (1999). *Practical implementation of nonlinear time series methods: the TISEAN package, Chaos*, *9*, 413.

Horne, J. H., & Baliunas, S. L. (1986). A prescription for period analysis of unevenly sampled time series. *Astrophysical Journal*, *302*, 757–769.

Horowitz, L. L. (1974). *IEEE Transactions*, *ASSP-22*, 22.

Hyvärinen, A., Karhunen, J., & Oja, E. (2001). *Independent Component Analysis*. New York: Wiley.

Infante, L., & Pritchet, C. J. (1992). CFHT north galactic pole faint galaxy survey. *Astrophysical Journal Supplement Series*, *83*, 237–259.

Karhunen, J., & Joutsensalo, J. (1994). Representation and separation of signals using nonlinear PCA type learning. *Neural Networks*, *7*, 113–127.

Karhunen, J., & Joutsensalo, J. (1995). Generalization of principal component analysis. Optimization problems and neural networks. *Neural Networks*, *8*, 549–562.

Kay, S. M. (1988). *Modern spectral estimation: Theory and application*. Englewood Cliffs-NJ: Prentice-Hall.

Klusch, M., & Napiwotzki, R. (1993). HNS—a hybrid neural system and its use for the classification of stars. *Astronomy and Astrophysics*, *276*, 309–319.

Kohonen, T. (1995). *Self-organizing maps*. Berlin: Springer.

Lepper, K., Whitley, V. H (2002). Distinguishing sedimentary depositional environments on mars using in-situ luminescence measurements and neural network analysis. *33rd Annual Lunar and Planetary Science Conference*, Abs. 1400.

Lomb, N. R. (1976). Least-squares frequency analysis of unequally spaced data. *Astrophysics and Space Science*, *39*, 447–462.

Lopez Ariste, A., Rees, D. E., Socas-Navarro, H., & Lites, B. W. (2001). Pattern recognition techniques and the measurement of solar magnetic fields. In J. L. Starck, & F. D. Murtagh (Eds.), *Proceedings of Astronomical Data Analysis* (*Vol. 4477*) (pp. 96–106). *SPIE*.

MacKay, D. J. C. (1993). *Hyperparameters: optimise or integrate out? Maximum entropy and Bayesian methods, Dordrecht*.

Medina Tanco, G. A., Sciutto, S. J (2001). *A multivariate study of mass composition for simulated showers at the Auger South observatory*. Astro-ph/0109140.

Murtagh, F., Donalek, C., Longo, G., & Tagliaferri, R. (2003). Bayesian Model Selection for Spatial Clustering in 3D Surveys. *Proceedings of the SPIE*, *4847* in press.

Neal, R. M. (1996). *Bayesian learning for neural networks*. Berlin: Springer.

(1992). *Numerical recipes in C: The art of scientific computing*. Cambridge: Cambridge University Press, p. 525.

Odewhan, S. C., & Nielsen, M. L. (1994). Star–galaxy separation using neural networks. *Vistas in Astronomy*, *38*, 281–301.

Odewhan, S. C., Stockwell, E. B., Penninton, R. L., Humpreys, R. M., & Zumach, W. A. (1992). Automated star/galaxy discrimination with neural networks. *Astronomical Journal*, *103*, 318–331.

Oja, E., Karhunen, J., Wang, L., & Vigario, R. (1996). VII Italian Workshop on Neural Networks. In M. Marinaro, & R. Tagliaferri (Eds.), (pp. 16–20). *IAU Symposium No. 161*, Singapore: World Scientific.

Oppenheim, A. V., & Schafer, R. W. (1989). *Discrete-time signal processing*. Englewood Cliffs-NJ: Prentice-Hall.

Patrick, G., Gabriel, S., Rodgers, D., & Clucas, S. (2002). Neural network prediction of solar energetic proton events with long lead times. In H. Sawaya-Lacoste (Ed.), *Second solar cycle and space weather Euroconference* (*SP-477*) (pp. 517–520). *ESA*.

Pham, D. T., Garrat, P., & Jutten, C. (1992). *Separation of a mixture of independent sources through a maximum likelihood approach*. *Proceedings of the EUSPICO*, pp. 771-774.

Pushell, J. J., Owen, F. N., & Laing, R. A. (1982). *Astrophysical Journal*, 275–280.

Razdan, A., Haungs, A., Rebel, H., & Bhat, C. L. (2002). Image and non-image parameters of atmospheric cherenkov events: a comparative study of their gamma ray/hadron classification potential in ultrahigh energy regime. *Astroparticle Physics*, *17*, 497–508.

Rosa, R. R., Sawantí, H. S., Cecatto, J. R., Neto, C., Rodrigues Lopes, V. C. A., Subramanian, K. R., Fernandes, F. C. R., Saito, J. H., Moron, C. E., Mucheroni, M. L., Furuya, N., & Mascarenhas, N. (2000). Phenomenological dynamics of coronal loops using a neural network approach. *Advances in Space Research*, *25*, 1917–1921.

Sanger, T. D. (1989). Optimal unsupervised learning in a single layer linear feedforward network. *Neural Networks*, *2*, 459–473.

Scargle, J. D. (1982). Studies in astronomical time series analysis: statistical aspects of spectral analysis of unevenly spaced data. *Astrophysical Journal*, *263*, 835–853.

Schaefer, B.M., Hofmann, W., Lampeitl, H., Hemberger, M (2001). *Particle identification by multifractal parameters in gamma-astronomy with the HEGRA-Cherenkov-Telescopes*. Astro-ph/0101318.

Smirnov, D. (2002). A neural net technique for the Higgs search at the Tevatron. *Proceedings of the Meeting of the American Physical Society*.

Snider, S., Allende-Prieto, C., von Hippel, T., Beers, T. C., Sneden, C., Qu, Y., & Rossi, S. (2001). Three-dimensional spectral classification of low-metallicity stars using artificial neural networks. *Astrophysical Journal*, *562*, 528–548.

Steinegger, M., Veronig, A., Hanslmeier, A., Messerotti, M., & Otruba, W. (2002). A neural network approach to solar flare alerting. In R. J. Garcia, L. R. Rebolo, & M. R. Zapaterio Osorio (Eds.), *11th Cambridge Workshop on Cool Stars, Stellar Systems and the Sun* (*Vol. 223*) (pp. 1165–1169). *ASP Conference Series*.

Stoughton, C., Lupton, R. H., Bernardi, M., Blanton, M. R., Burles, S., Castander, F. J., Connolly, A. J., Eisenstein, D. J., Frieman, J. A., Hennessy, G. S., et al. (2002). Sloan digital sky survey: early data release. *Astronomical Journal*, *123*, 485.

Svensen, M (1998). *GTM: the generative topographic mapping*. PhD Thesis, Aston University.

Tagliaferri, R., Ciaramella, A., Milano, L., Barone, F., & Longo, G. (1999a). Spectral analysis of stellar light curves by means of neural networks. *Astronomy and Astrophysics Supplement Series*, *137*, 391–405.

Tagliaferri, R., Capuano, N., & Gargiulo, G. (1999b). Automated labeling for unsupervised neural networks a hierarchical approach. *IEEE Transactions of Neural Networks*, *10*, 199–203.

Tagliaferri, R., Ciaramella, A., Barone, F., & Milano, L. (2001). Neural networks for spectral analysis of unevenly sampled data. In A. J. Banday, S. Zaroubi, & M. Bartelmann (Eds.), *Mining the sky* (pp. 386–390). Heidelberg: Springer.

Tagliaferri, R., Longo, G., Andreon, S., Capozziello, S., Donalek, C., Giordano, G (2002). *Neural networks and photometric redshifts*. Astro-ph/0203445.

Takens, F. (1981). In D. Warwick, & L.-S. Young (Eds.), *Dynamical systems and turbulence* (*898*). *Lecture notes in mathematics*, Berlin: Springer.

Torres, S., García-Berro, E., & Isern, J. (2001). Classification of the white dwarf populations using neural networks. In A. J. Banday, S. Zaroubi, & M. Bartelmann (Eds.), *Mining the sky* (pp. 391–393). Heidelberg: Springer.

Turunen, E. (1999). *Mathematics behind fuzzy logic*. *Advance in soft computing*, Berlin: Verlag.

Vanzella, E, Cristiani, S., Arnouts, S., Dennefeld, M., et al (2002). *A VLT/FORS2 spectroscopic survey in the HDF-S*. Astro-ph/0210146.

Vesanto, J (1997). *Data mining techniques based on the self-organizing map*. PhD Thesis, Helsinki University of Technology.

Veselovskii, I. S., Dmitriev, A. V., Orlov, Yu. V., Persiantsev, I. G., & Suvorova, A. V. (2000). Simulation of statistical distributions in the space of parameters of the solar wind and interplanetary magnetic field using artificial neural networks. *Astronomicheskii Vestnik*, *34*, 116–125.

Vogt, R. E., Drever, R. W., Raab, F. J., Thorne, K. S (1989). *Proposal for the construction of a large interferometric detector of gravitational waves*. Proposal to the National Science Foundation, California Institute of Technology, Pasadena, California, USA.

Weaver, B. (2002). Artificial neural networks and spectral classification. *The Observatory*, *122*, 166.

Weaver, Wm. B. (2000). Spectral classification of unresolved binary stars with artificial neural networks. *Astrophysical Journal*, *541*, 298–305.

Weir, N., c, U. M., & Djorgovski, S. (1995). Automated star/galaxy classification for digitized poss-II. *Astronomical Journal*, *109*, 2401–2414.

Witten, I. H., & Frank, E. (2000). *Data Mining*. Los Altos: Morgan Kaufmann Publishers.

Zadeh, L. A. (1965). Fuzzy sets. *Information and Control*, *8*, 338–353.