AGENZIA NAZIONALE PER LE NUOVE TECNOLOGIE,
L'ENERGIA E LO SVILUPPO ECONOMICO SOSTENIBILE

ISSN/0393-3016

RAMAN SPECTRA MASSIVE CLASSIFICATION USING ARTIFICIAL NEURAL NETWORKS

SABINA BOTTI

ENEA – Unità Sviluppo di Applicazioni delle Radiazioni
Laboratorio Micro e Nano-strutture per la Fotonica
Centro Ricerche Frascati, Roma

G. DIPOPPA, A. PUIU, S. ALMAVIVA

ENEA – Unità Sviluppo di Applicazioni delle Radiazioni
Laboratorio Diagnostiche e Metrologia
Centro Ricerche Frascati, Roma

RT/2014/13/ENEA

AGENZIA NAZIONALE PER LE NUOVE TECNOLOGIE,
L'ENERGIA E LO SVILUPPO ECONOMICO SOSTENIBILE

RAMAN SPECTRA MASSIVE CLASSIFICATION USING ARTIFICIAL NEURAL NETWORKS

SABINA BOTTI

ENEA – Unità Sviluppo di Applicazioni delle Radiazioni
Laboratorio Micro e Nano-strutture per la Fotonica
Centro Ricerche Frascati, Roma

G. DIPOPPA, A. PUIU, S. ALMAVIVA

ENEA – Unità Sviluppo di Applicazioni delle Radiazioni
Laboratorio Diagnostiche e Metrologia
Centro Ricerche Frascati, Roma

RT/2014/13/ENEA

RAMAN SPECTRA MASSIVE CLASSIFICATION USING ARTIFICIAL NEURAL NETWORKS

SABINA BOTTI, GIOVANNI DIPOPPA, ADRIANA PUIU, SALVATORE ALMAVIVA

Sommario
Questo lavoro esplora l'uso del "Machine Learning" per sviluppare metodi di identificazione automatica di sostanze chimiche a partire dagli spettri Raman. La parziale sovrapposizione degli spettri Raman di alcune sostanze di interesse e alcuni problemi di classificazione, hanno suggerito l'uso di una rete neurale (ANN) come algoritmo capace di lavorare, con ottimi risultati, in linea continua, elaborando spettri Raman provenienti dal sistema di rilevazione. I risultati dell'algoritmo ANN sono discussi confrontandoli con quelli ottenuti applicando la Principal Component Analysis (PCA).

Parole chiave: Spettroscopia Raman, Reti Neurali, PCA.

Abstract
This paper explores the use of Machine Learning (ML) methods to develop automated system for the identification of chemical compounds from their Raman spectra. The presence of different substances of interest whose spectra signatures are often very close to each other and the occurrence of specific hard–to–solve classification problems suggested the use of an Artificial Neural Network (ANN) as software paradigm able to deal, with good performances, with on–line processing of Raman data coming from the detection system. The ANN results are discussed by comparing their performances against those of Principal Component Analysis.

Keywords: *Raman Spectroscopy, Artificial Neural Network, Principal Component Analysis.*

Contents

1 Introduction

The specificity of vibrational Raman spectral signatures provides a powerful and effective method for chemical identification of both simple molecular species as well as more complex structures. Due to this unique ability, in spite of its intrinsic weakness, the Raman spectroscopy is gaining a great interest as analytical technique that does not require a preparation of samples, is not destructive and requires acquisition times often less than one minute. However, to use the Raman spectroscopy as on field technique it is necessary to develop systems that may be operated by non - chemists minimally trained personnel for the automated detection and identification of analytes. The difficulty in interpreting spectra is a natural consequence of visually inspecting high dimensional data sets. Unsupervised pattern recognition methods have therefore been developed to automate the classification of data sets into defined clusters by providing model as free approaches [1, 2].

In this work, we report on the development of an artificial neural network that can correctly classify up to 16 different substances: picric acid, nitrocellulose, TNT, tetryl, sodium azide, HMX, RDX, nitroguanidine, HMTD, TATP (synthesised following two preparation steps), PETN, potassium chlorate, urea nitrate, ammonium picrate, styphnic acid. These materials have been choose because their identification can be of interest in the development of a compact device for homeland security applications, but the implemented algorithm, with minor changes, can be exploited for the classification of other kind of materials.

We previously used a cluster-based multivariate data analysis technique to quantify the pattern reproducibility of Raman spectra taken from different region of the same sample. The standard deviation of the Raman signal and background intensity was below the 10%, suitable for the spectral classification within a priori library reference groups. As shown in the following, the application of principal component analysis (PCA) drastically reduced the dimensionality of the spectral arrays, maximizing the spectral variances, 2D PCA plots and hierarchical cluster analysis (HCA) dendrograms give a convenient representations showing the grouping of analytes. Although, PC cluster identification methods based on Raman spectra exhibit enhanced specificity, compared to other spectral fingerprinting schemes, the substantial overlap of spectral regions of different chemical groups with the same reduced mass, naturally leads to a grouping of analysed samples with some miss–assignments. Therefore, these methods that resulted to be successful applied to few chemical species [3–7], do not work well for a massive discrimination.

In this paper, we demonstrate that the use of customized auto–encoder artificial neural network allows the correct classification of 16 samples with an efficiency of 100%. The identification algorithm described here highlights the effectiveness of Raman spectroscopy detection and identification approaches and is of critical interest for the development of a rapid, reagentless and portable diagnostic device.

2 Raman spectroscopy

The Raman spectra were acquired with an integrated table–top Raman system (BwTeK inc, *i*–Raman), in the wavelength range $789 - 1048$ nm, corresponding to Raman shifts of $75 - 3000$ cm^{-1}, with a resolution of 3 cm^{-1}. The near–IR excitation eliminates most of sample fluorescence. Several spectra were acquired from each sample with a laser power ranging from 150 to 300 mW and acquisition times up to 30 s.

Looking at the spectral characteristics of the Raman spectra, the examined substances can be divided in four groups. The first group contains the oxidiser salts as potassium chlorate and urea nitrate (see Fig. 1).

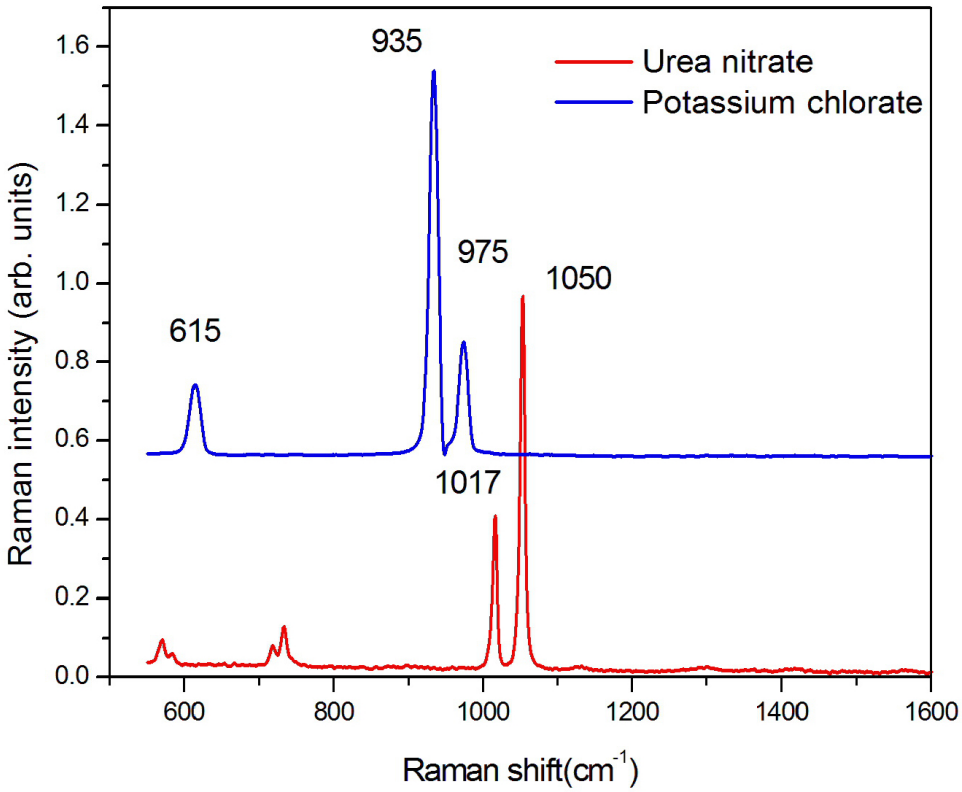

Figure 1. *Raman spectra of potassium chlorate, and urea nitrate excited at 785 nm. The line is the averaged spectrum.*

The Raman spectra of these oxidizer salts are characterized by the vibration of inorganic oxyanions: 620 cm^{-1}, 930 cm^{-1}, 975 cm^{-1} (ν_3, ν_1 and ν_2 vibration of ClO_3^-), $1040 - 1060$ cm^{-1} (symmetric stretching of NO_3^-), $1300 - 1400$ cm^{-1} (asymmetric stretching of NO_3^-).

The second group contains picric acid, styphnic acid, nitrocellulose, ammonium picrate, TNT, tetryl and sodium azide (see Fig. 2). All these compounds, with the exception of sodium azide, contain the nitro (NO_2) group with the characteristic vibration of symmetric (1360 cm^{-1}) and a–symmetric stretching (1560 cm^{-1}). It is important to underline that the tetryl contains

a trinitrobenzene and nitramine moiety, but the main vibrational features are those of a nitroaromatic compound. The sodium azide (NaN_3) is, from a chemical point of view, an outlier of this group, but the ν_1 vibration of internal group N_3 also falls at 1360 cm^{-1}.

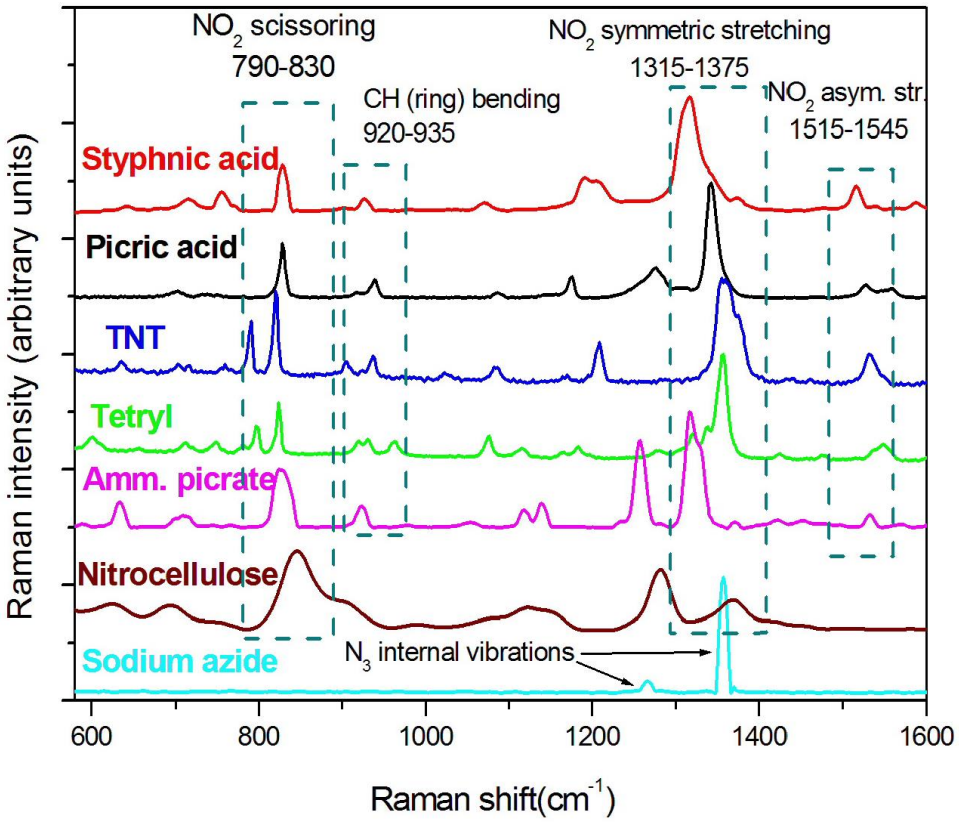

Figure 2. *Raman spectra of styphnic acid, picric acid, TNT, tetryl, ammonium picrate, nitrocellulose and sodium azide, the main vibration features are indicated.*

The HMX, nitroguanidine, HMTD, TATP and PETN belongs to the third group. As shown in (see Fig. 3), their Raman spectra are characterised by the vibration in the $750 - 950$ cm^{-1} region due to the C–O, C–C ring stretching (TATP), O–O stretch (TATP and HMTD), C–N–C stretching (RDX, HMX and nitroguanidine) and O–N stretching (PETN); in the $1200 - 1300$ cm^{-1} region ascribed to the NO_2 symmetric stretching (PETN and nitroguanidine), C–C–C bend (TATP), C–N stretching (HMTD, very weak) and $1400 - 1600$ cm^{-1} characteristic of NO_2 asymmetric stretching (PETN, HMX and nitroguanidine), H–C–H bend (TATP) and CH–C stretching (HMTD).

The observed variations in the Raman spectra across the samples play a critical role in the ability to discriminate and identify between the different compounds. A key component of any Raman spectroscopy based diagnostic platform is a data reduction protocol for accurate identification without any miss–assignment. These methods quantify the reproducible specificity of a

given spectroscopic trace and permit the spectral classification through the comparison with library reference spectra. The PCA is employed to reduce the dimensionality of the spectral arrays, and to identify which are the factors that mainly affect the spectral variation across the different samples. We expect that identification methods based on Raman spectra should exhibit enhanced specificity.

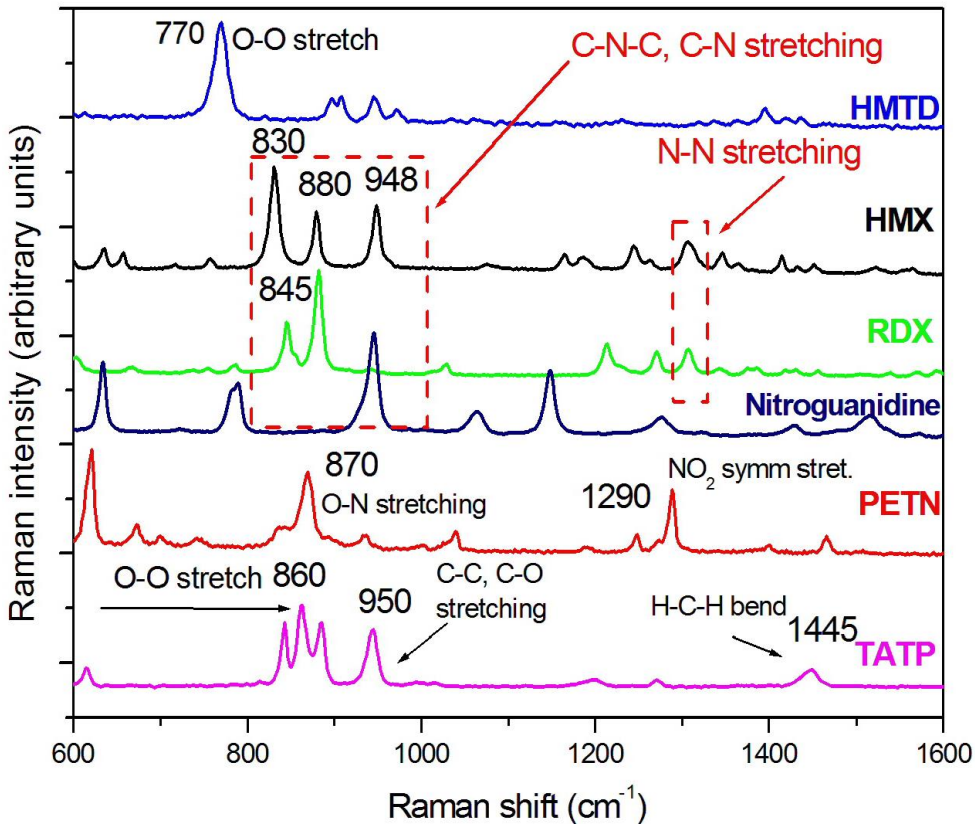

Figure 3. *Raman spectra of HMTD, HMX, RDX, PETN and TATP excited at 785 nm.*

2.1 Clustering with PCA

Principal Component Analysis (PCA) is a statistical procedure that uses orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. The number of principal components is less than or equal to the number of original variables. This transformation is defined in such a way that the first principal component has the largest possible variance (that is, accounts for as much of the variability in the data as possible), and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to (i.e., uncorrelated with) the preceding components. Principal components are guaranteed to be independent if the data set is jointly normally distributed. PCA is sensitive to the relative scaling of the original variables.

All the spectra were normalised to their maximum value, the spectral background was removed and the spectra were truncated in the $550 - 1600$ cm^{-1} region which contains all the relevant vibrational features, thus avoiding that small and non relevant bands can acquire the same importance as the large bands related to important functional groups. In this way a data matrix of 66 samples and 536 data points was constructed to be submitted to the PCA algorithm developed in a MATLAB environment.

PCA shows that the first three principal components explains the 62% of the overall spectral variation. The corresponding loading plots, which indicate the specific contribution of each vibrational band in the total variance of spectral data, are reported in Fig. 4.

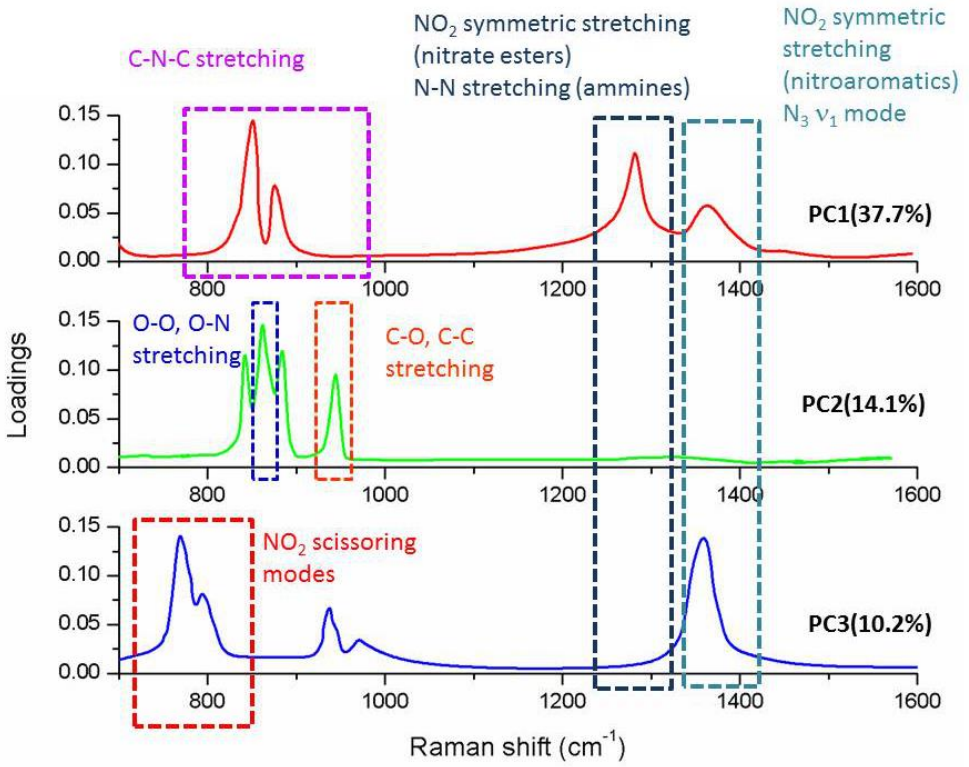

Figure 4. *Loading plots for the first three components. Same scale for all the plots.*

The first principal component, PC1, which accounts for the 37.7% of spectral variance, has four main peaks: 850 cm^{-1}, 880 cm^{-1} that correspond to the C-N-C ring breathing (RDX, HMX and tetryl), 1290 cm^{-1} ascribed to the symmetric stretching in nitrate esters as PETN and N-N stretching in ammines as RDX and HMX, 1360 cm^{-1} due to the NO_2 symmetric stretching vibration in nitroaromatics as styphnic acid, ammonium picrate, picric acid, TNT, nitrocellulose and tetryl which contains a trinitrobenzene and nitramine moiety, but the main vibrational features are those of a nitroaromatic compound. In the second component, PC2, with an explained spectral variance of 14.1%, the peak at 860 cm^{-1} corresponds to the O-O stretching in TATP and HMTD and O-N

stretching in PETN, that at 950 cm^{-1} corresponds to the C-C, C-O stretching in TATP and C-N stretching in HMTD. These peaks are superimposed with the C-N-C stretching peaks of PC1. The loading plot for PC3, which explains the 10.2% of spectral variance shows, besides smaller peaks in position equivalent to those in the PC2 loading plot, exhibits strong bands in the region of NO_2 scissoring modes (770 cm^{-1}) and symmetric stretching (1360 cm^{-1}) in nitro–aromatics. It is worth to note that also the N_3 internal vibration in azide has a strong peak in the same position of NO_2 symmetric stretching. In Tab. 1 the above mentioned vibrational modes assignment are summarized.

Raman shift (cm^{-1})	**Assignment**
$1360 - 1390$	NO_2 symmetric stretching (nitroaromatics compunds) C-N stretching (HMTD) N_3 ν_1 mode of azide
$1250 - 1290$	N-N stretching of ammines NO_2 symmetric stretching (ester nitrates)
$750 - 950$	C-C, C-O, O-O ring stretch in peroxides C-N-C, C-N stretching (HMX, RDX, nitroguanidine) ClO_3 ν_1 vibration NO_2 scissoring (nitroaromatics) C-H (ring) bending (nitroaromatics)

Table 1. *Raman shift vibrational modes assignment.*

Fig. 5 shows the PCA plot for the PC1 and PC2, the two largest principal components of the dataset, which explain 51.8% of the spectral variance and exhibit the greatest cluster separation. In this plot, each sample is represented by a point, and several groups, each corresponding to an explosive compound can be identified. In addition, for each clustered species, an ellipse corresponding to a standard deviation for the PCA values has been drawn. These rings are a representation of the reproducibility of the data, in fact the samples with higher signal–to–noise ratio (SNR) Raman spectra are closely related (this is the case of oxidizer salts) while other compounds, as nitrocellulose, have a very large distribution of points. In principle, such standard deviation rings can be used as one measure of the diagnostic specificity offered by Raman spectroscopy.

As shown in the PC1 vs. PC2 plot reported in Fig. 5, the PCA analysis based on the Raman spectra, shows well- separated clusters, except those of HMTD and nitroguanidine that are substantially overlapped. However, nitro-amines and nitrate esters cannot be separately classified because the clusters of HMX, RDX, MHTD, nitroguadine and PETN are very close. This can be explained considering that the PCs exhibit as spectral striking feature the vibration of C-N-C bond which overlaps the C-N and O-N stretching frequency. Again, azide cluster is very close to those of nitro-aromatics compounds due to the overlapping of N_3 ν_1 stretching mode and NO_2 symmetric stretching. The

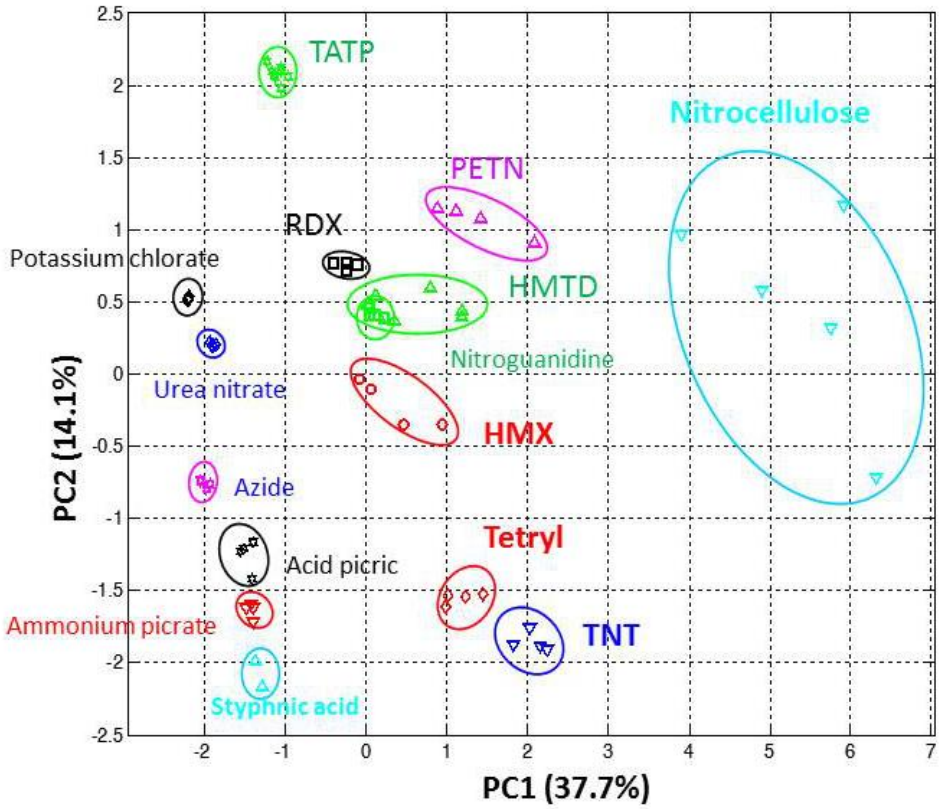

Figure 5. *A PCA 2D score plot of PC1 versus PC2 for the Raman spectra dataset. The percent variance captured by each of these principal components is indicated in parenthesis along the corresponding PCA axis.*

nitrocellulose cluster is judged to be distinct from the other nitro–aromatics compounds.

The application of PCA to the Raman spectra expressed by 536 different wavelengths has allowed to reduce the output to only three components, however the score plot indicates that the proposed model is not able to correctly group all the compounds.

Hierarchical cluster analysis (HCA) was performed by using the Unweighted Pair–Group Average (UPGMA) algorithm. Clusters are joined based on the average distance between all members in the two groups. The distance matrix was computed using 'Cosine' indices in which the distances are calculated as the inner product of abundances, each normalized to the cosine of the angle between the vectors. With this method, first the scores from two cases across variables are correlated and in a second step the cosine angle of that correlation is find. In order to group together substances with similar spectral features, we choose to use 'Cosine' index because ignores level differences between samples and looks only at shape differences. The cophenetic correlation coefficient which gives the measure of how faithfully the dendrogram (hierarchical tree) preserves the pairwise distances between the original unmodeled data points was calculated

to be 0.82, which is a good result. The magnitude of this value should be as close as possible to 1 for a high-quality solution.

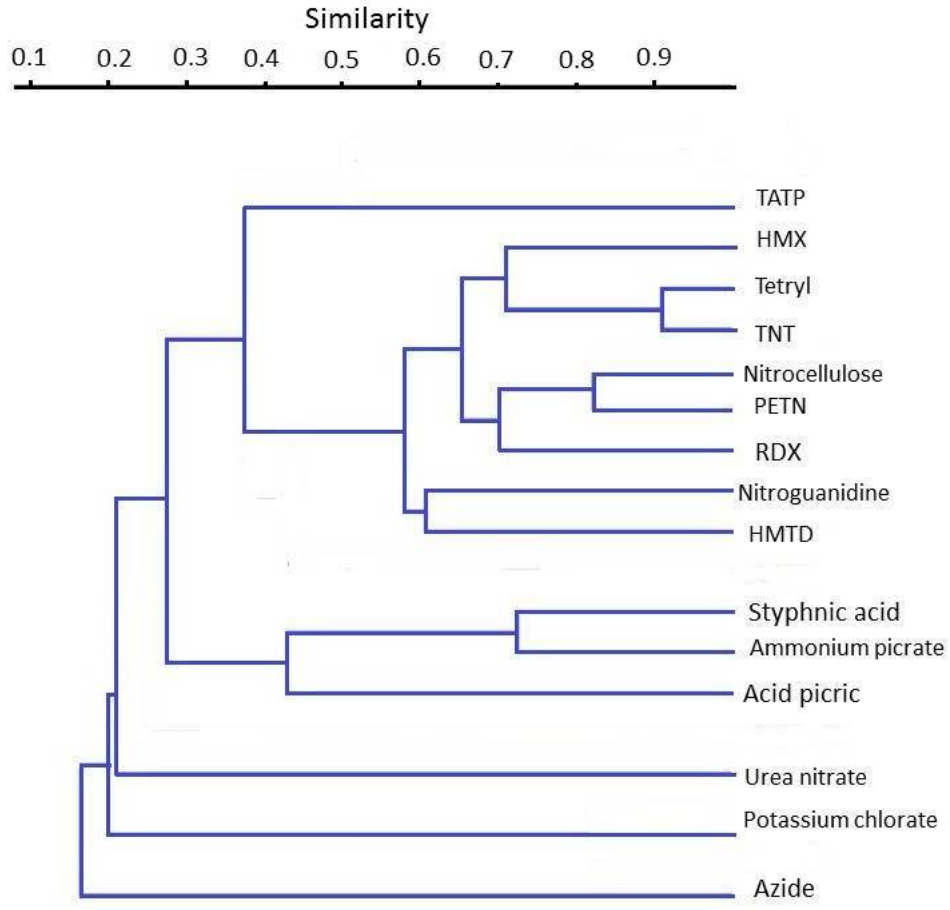

Figure 6. *HCA dendogram corresponding to PCA clustering.*

The corresponding distance-based HCA dendrogram shown in Figure 6 conveys the same assignment difficulty in the grouping of the nitro-amines and nitrate esters. It is also interesting to note that within this data set the nitro cellulose is judged more similar to PETN than to Tetryl or TNT, while the azide is considered apart from the nitro-aromatic group on the contrary of what happens with PCA clustering.
All the reported results indicate that this kind of approaches, that are successful for the correct classification of a small number of samples, are no more suitable for a massive discrimination.

3 Machine Learning Foundation

In nature a neuron consists of 3 main parts (see Fig. 7):

- soma, i.e. the cell body;
- axon, i.e. output line of the neuron only, that can be also separated into thousands of branches;
- dendrite, or dendritic tree since its extensive branching remember the shape of a tree, it is the neuron input line receiving the signals from other axons through the synapses.

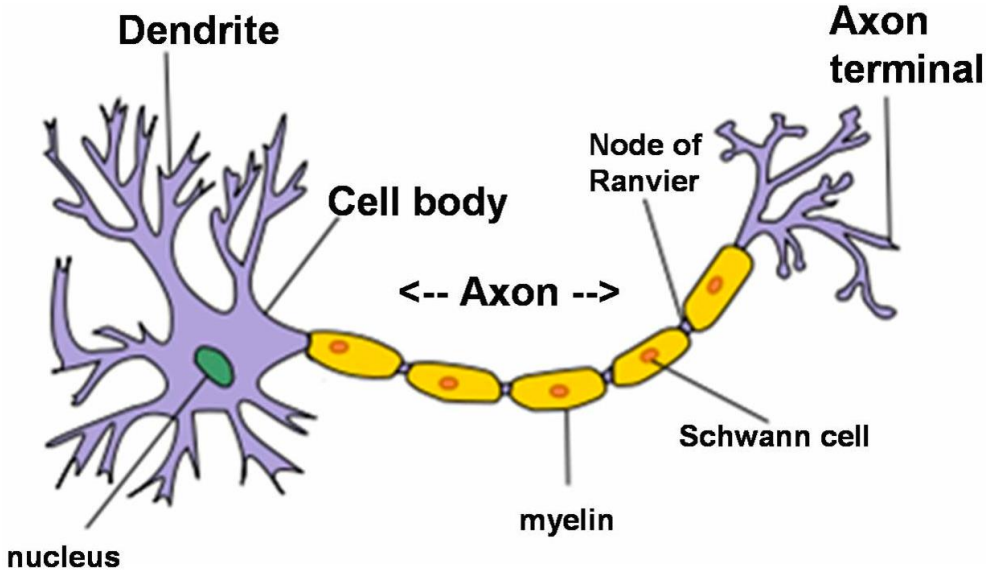

Figure 7. *Structure of a neuron: researchers found in their latest realistic model of neurons that a second electrical potential forms on the membrane of the cell. Unless that potential increases above a fixed threshold the neuron remains in idle state.*

The cell body performs a *weighting algebraic sum* (integration) of the input signals. If the result exceeds a certain threshold value then the neuron becomes active and produces a *potential action* which is sent to the axon. If it does not exceed the threshold value, the neuron remains in idle state.

An artificial neural network receives external signals on one input layer of nodes (processing unit), each of which is connected with a number of internal nodes, organized in several levels. Each node processes the received signals and transmits the result to succeeding nodes.

An artificial neuron [8] is the fundamental calculus unit of the neural network and in the neural model it is formed from three basic elements (see Fig. 8):

- a set of synapses or connections, each of which is characterized by a weight (synaptic efficacy), unlike the human model, the artificial weights can have both positive and negative values;

- an adder that sums the weighted inputs from the respective signals in the synapse, producing in output a linear combination of the inputs;
- an activation function for limiting the amplitude of the output of a neuron. Typically, for convenience, the output amplitude is in the interval $[0, 1]$ or $[-1, 1]$.

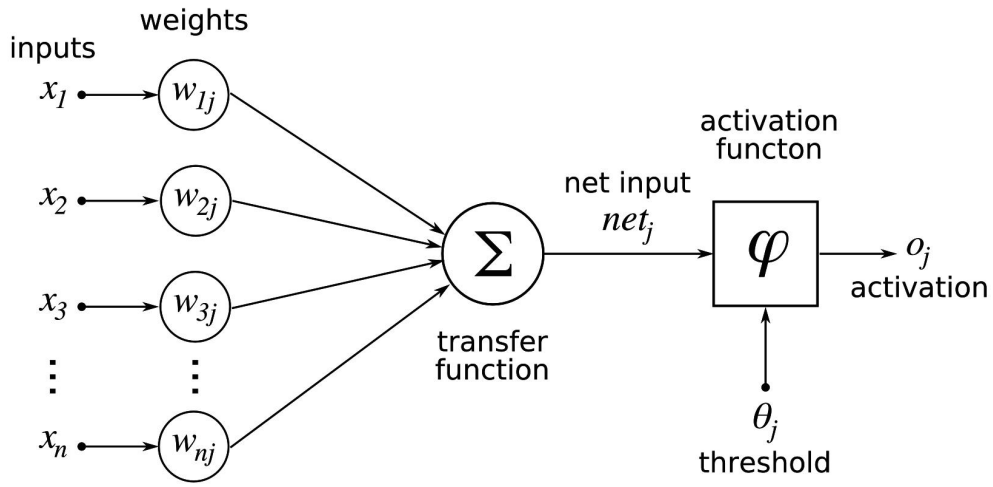

Figure 8. *Artificial neuron: it is a mathematical function conceived as a crude model, or abstraction of biological neurons.*

The neuronal model also includes a threshold value which has the effect, depending on its positivity or negativity, to increase or decrease the net input to the activation function.

3.1 Paradigm: the Artificial Neural Network

A typical Artificial Neural Network (ANN) is built out from simple, non-intelligent units (neurons) which are connected together, becoming able to perform very complex signal processing.

In the learning phase, an ANN is presented with input data set and is trained to fire out the desired values at output layer. The training algorithm iteratively modifies weights on connections through which signals are transmitted, in order to minimize gap between network output and desired one.

The Autoencoder Model, i.e. an Autoassociative Neural Network Encoder, or simply autoencoder (see Fig. 9), has two primary features:

- **Autoassociative Feature**: the network is trained to reproduce at the output layer the same values presented as input. For this reason input and output layer have the same size (i.e. the same number of neural units).
- **Bottleneck Layer**: at least one of the network hidden layers must be smaller than both input and output.

The architecture selected in this work consists of an input layer, five hidden layers and an output layer (see Fig. 10). The first three hidden layers shape

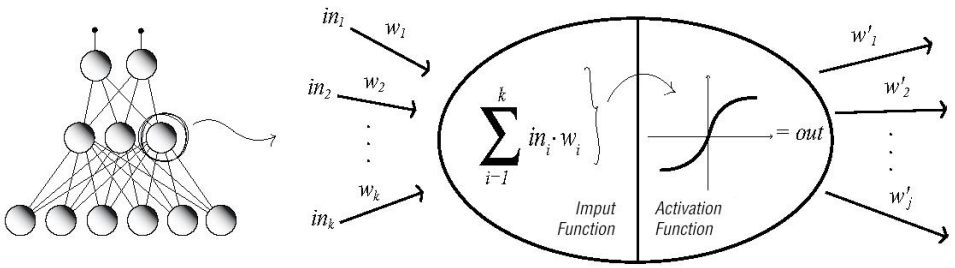

Figure 9. *Structure of ANN and related neurode, a simple processing element of a neural network having several input signals. It is highlighted the dual nature function of the node: linear in input signals processing, nonlinear elaboration of the output through the use of the activation function.*

a feature detection architecture in which bottleneck layer plays the key role in the identity mapping, as it forces the network to develop a compact representation of the training data that better models the underlying system parameters. In fact, they constitute a structure able to capture the characteristics of the domain:

- **mapping** layer, with size greater than the input layer;
- **bottleneck** layer, it must have a lower number of units compared to the other layers of the network. At this level the operation of encoding itself occurs: in particular, the information contained in the domain input is essentially projected within a space of lower dimension;
- **de-mapping** layer, it has the same dimensions of the layer of mapping.

Therefore the network architecture is constituted by an input layer, three decoder-encoder hidden layers, two hidden layers driving to classification and finally the classification output layer with **16** nodes, i.e. the number of compounds to be recognised.

Autoencoder Input

Some data preprocessing is needed in order to feed the autoencoder with Raman measures: normalization inside interval $[-1, 1]$ of the spectra and filtering of data containing anomalies. It is evident that, in this step, the expertise in Raman spectra interpretation is needed. In fact, the training must be conducted only with reliable and suitable data for the classification procedure. In some cases, it can be useful to make an appropriate sampling among the available measures to reduce input and output layer size, thus saving time in training phase.

In mathematical terms we describe a neuron k with the following equations:

$$\begin{aligned} u_k &= \sum_i^m w_{kj} \cdot x_j \\ y_k &= \varphi(u_k + b_k) \end{aligned} \tag{1}$$

where:

- x_i, i.e. synaptic weights of k^{th} neuron;
- u_k, i.e. linear combination of the inputs into neuron;
- b_k, i.e. the threshold value of k^{th} neuron, it can be inserted inside the u_k so as to remove the additional factor inside function;
- $\varphi(x)$, i.e. activation function;
- y_k, it is the output generated by k^{th} neuron.

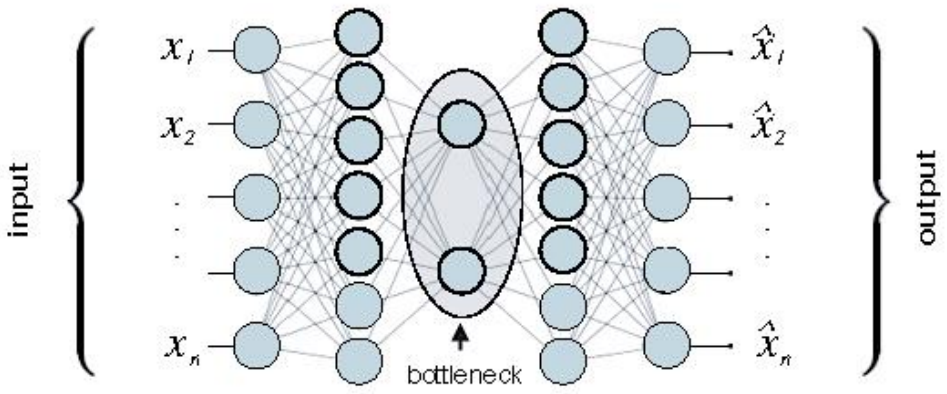

Figure 10. *Example of architecture for autoencoder model. It is a standard architecture Feed Forward fully connected, with three hidden layers. The bottleneck layer always must have a dimension strictly lower than the input layers of the network.*

The way in which the network is structured, depends on the used learning algorithm. In general, three classes of networks are identified:

- in the simplest form of a layer feedforward network, there are the input nodes (input layer) and a layer of neurons (the output layer). The signal in the network propagates forward so acyclic, starting from the input layer and ending in the output. There are connections that go back and even cross-connections in the output layer;

- the class of Multilayer Feedforward networks differs from the previous by the fact that between the input layer and the output we have one or more layers of hidden neurons (hidden layers). Each layer has incoming connections from the previous layer and outgoing links in the next, then the signal propagation takes place along without cycles and without cross-connections.This type of network architecture provides a global perspective as they increase the interactions between neurons;

- a Recurrent Network (or feedback) differs from the previous ones in that it is cyclical. The presence of cycles has a strong impact on the learning capacity of the network and its performance. In particular, it determines the system dynamic.

3.2 Learning Processes

Different types of learning can be applied to neural networks: the learning with error correction, the memory–based learning, the Hebbian [9] Learning and the competitive learning.

The first technique consists of constantly correcting the error reporting, as each neuron k receives as input a signal stimulus $x(n)$ and generates a response $y_k(n)$, where n is a discrete time. We denote also by $d_k(n)$ the desired response. Consequently, it generates an error signal $e_k(n)$:

$$e_k(n) = d_k(n) - y_k(n). \tag{2}$$

The error signal $E_k(n)$ implements a control mechanism with the aim of applying a value adjustments sequence to the synaptic weights of the k^{th} neuron in order to approximate the response to the desired one. This process takes one step at a time by minimizing a cost function:

$$E(n) = \frac{1}{2} \cdot e_k^2(n) \tag{3}$$

To do this, we resort to the gradient method (Fig. 11) or the Widrow-Hoff method [10]. Let $w_{kj}(n)$ the synaptic weights of the k^{th} neuron excited by the element $x_j(n)$ (i.e. the stimulus signal), then the adjustment applied to $w_{kj}(n)$ is:

$$\Delta w_{kj} = \eta \cdot e_k(n) \cdot x_j(n) \tag{4}$$

where η is a positive constant known as learning rate. The new weights will be then:

$$w_{kj}(n+1) = w_{kj}(n) + \Delta w_{kj}(n) \tag{5}$$

The learning with error correction is an example of a system whose stability depends on the parameters that constitute the cycle and in particular η. The choice of this parameter is very important because it affects the convergence and stability of the whole learning process.

On memory–based learning, all (or many) of past experiences are stored in a large memory input–output pairs that are correctly classified, $\{(x_i, d_i)\}_{i=1}^{n}$, where x_i is the input vector and d_i is the desired response that, without loss

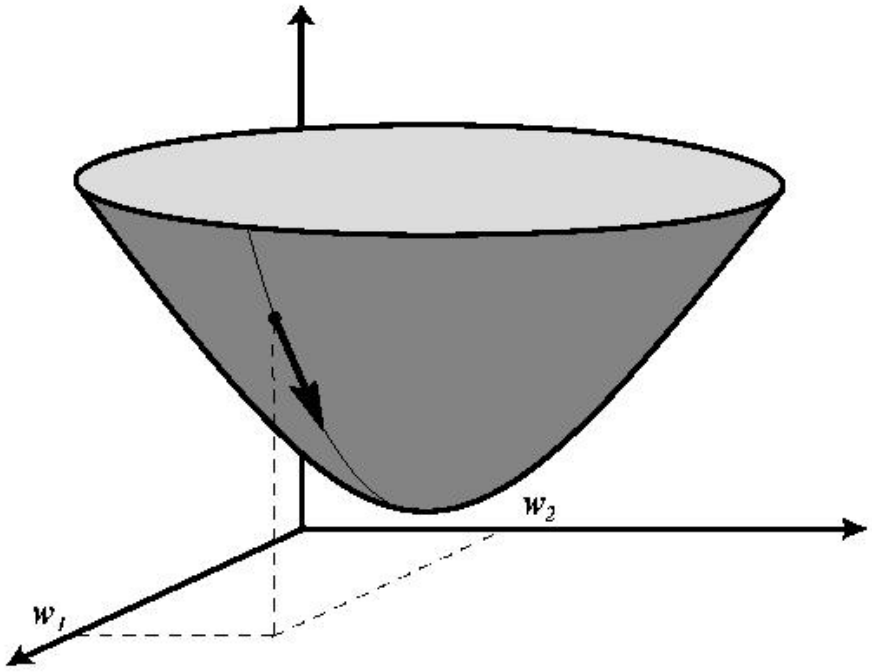

Figure 11. *Representation of surface error for a perceptron with three units, connected by two links, whose weights are indicated in the figure by* w_1 *and* w_2*: the hypotheses space is therefore identified by the plan* $w_1 * w_2$*. The training algorithm modifies the weights by calculating the gradient descent down the error surface, with the objective to reach its global minimum. For the case of multilayer ANN multiple minimum local exist.*

of generality, has been reduced to a scalar. When it will be requested, the classification of an instance never met before x_{test}, the system will answer by finding and analyzing samples stored in a neighborhood of x_{test}. All learning methods based on memory comprise two basic ingredients:

- the criterion used to define the neighborhood of x_{test};
- the learning method applied to examples in x_{test} neighborhood.

An example of this method is the **nearest neighbour** [11] where the closest example to the example test $x_N^{'}$ is the one that has the minimum Euclidean distance.

$$\begin{aligned} x_N^{'} &\in \{x_1, x_2, \ldots, x_N\} \\ \min_i[d(x_i, x_{test})] &= d(x_N^{'}, x_{test}) \end{aligned} \tag{6}$$

where d is the Euclidean distance. The class that is assigned to x_{test} is the same as $x_N^{'}$. A variant of this method is the k-nearest neighbour [12] in which:

- around the test example is not only one, but the set of k closest examples stored;

- the assigned class is the one with the higher frequency in the neighborhood of the example test.

The third ranking technique is called Hebbian, from the Hebb postulate [9] on learning: when an axon of a neuron A is near enough to excite a neuron B, and this, in a repetitive and persistent way, sends a potential action, in one or both of the neurons the A efficiency starts growing.

From this postulate it is possible to derive two basic rules:

- if two neurons connected by a synapse are activated simultaneously, then the weight of the synapse is progressively increased;
- if two connected neurons are activated asynchronously, then the weight of the synapse is progressively reduced or eliminated.

Synapses of this type are called *Hebbian*.

In the fourth learning type, i.e. the competitive learning, the output neurons compete with each other to become active, but only one neuron can be active at a given time n. There are three basic elements for a method of competitive learning:

1. a set of neurons less than the randomly generated synaptic weights, which respond differently to a given set of inputs;
2. a limit to the *strength* of each neuron;
3. a mechanism that allows neurons to compete for responding to a given subset of inputs, so that only one neuron, or one for group, is active at a given time. The winning neuron is called *winner-takes-all*.

In this way, the neurons tend to specialize on a set of similar input becoming recognizers characteristics for different classes of inputs. In its simplest form the neural network has a single layer of output neurons, fully connected to the input nodes (forward excitatory connections). The network can include connections between neurons that lead to lateral inhibition (inhibitory feedback connections). The k^{th} neuron that wins the competition is the one with input net v_k higher for a given input x. The output signal of the neuron y_k (victorious) is set to 1, while the other is set to 0.

$$y_k = \begin{cases} 1 & \text{se } v_k > v_j \quad \forall j,\, j \neq k \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

where v_k is the linear combination of all input and feedback forward. Let $w_{kj} \geq 0$ the synaptic weights between j and k input neuron. Let suppose that each neuron has a fixed amount of synaptic weight distributed between the input nodes:

$$\sum_j w_{kj} = 1 \quad \forall k \tag{8}$$

The neuron that wins the competition learns by shifting synaptic weights from the inactive inputs to active. To do this, it uses the standard rule of competitive learning:

$$\Delta w_{kj} = \begin{cases} \eta \cdot (x_j - w_{kj}) & \text{if } k \text{ neuron wins} \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

3.3 The ANN architecture

A network consists of a set of inputs (input layer), one or more hidden layers of neurons and a set of output neurons. The input signal propagates through the network forward from layer to layer. Each neuron includes a differentiable nonlinear activation function (e.g., **sigmoid function**), the network contains one or more hidden layers that are not part of either the input or the output of the network, which finally has a high connectivity.

An efficient method for the network training is the **back-propagation algorithm**, where we can identify two different types of signals:

1. **function signals**: input signal or stimulus that enters the input layer and propagates forward through the hidden layer to the output layer, emerging as an output signal;

2. **error signal**: its origin is in the output layer and propagates back through the network.

Each hidden neuron or the output of a pattern **multilayer perceptron** performs only two computations: that of function signal, expressed as a nonlinear function of a continuous input signal and of synaptic weights associated to the neuron, and that of a gradient vector required for updating synaptic weights.

The error signal of the output neuron j iteration n (presentation of the n^{th} instance of the training set) is defined by:

$$e_j(n) = d_j(n) - y_j(n) \tag{10}$$

where:

- $d_j(n)$ is the expected output of the neuron j;
- $y_j(n)$ is the output of neuron j.

The total error of the output layer presenting the n^{th} instance of the training set is defined as:

$$E(n) = \frac{1}{2} \cdot \sum_{j \in C} e_j^2(n) \tag{11}$$

where C is the set of neurons that form the output layer.

With N total number of patterns contained in the training set, the mean square error is the cost function:

$$E_{av} = \frac{1}{N} \cdot \sum_{i=1}^{N} E(i) \tag{12}$$

Then the goal of the training is to minimize E_{av} adjusting the free parameters of the neural network.

3.4 Neural Network Parameters

The learning algorithm for Neural Networks requires certain parameters such as the *learning rate*, *momentum*, type of *activation function* of each neuron, *error calculation function* and the *network topology* to train and test. In this work, the first goal to achieve is to find a structure of adequate network while the optimization of the parameters remains a secondary aspect. In fact, considering the type of data and physical principles that characterise them, it is seen that spectra values are ranging in the intervals (normalized or not) centered at 0, i.e. absence of scattering. Therefore, the activation function is a sigmoid that admits symmetric values in the range $[-1, 1]$.

The output consists of 16 neurons that emit continuous values in the interval $[0, 1]$ and in the studied cases, more precise values, as close to the classifications (0 and 1) as one can wish.

By fixing the number of input and output nodes, we tried to vary the number of intermediate levels and relative quantity of neurons per layer. Several configurations were tested on a training for a maximum of 4000 epochs, or by following the empirical law:

- H_1 s.t. $|H_1| \simeq O(\frac{n}{2})$;
- H_2 s.t. $|H_1| \simeq O(\frac{n}{2})|H_2| \simeq O(\frac{n}{4})$;
- H_3 s.t. $|H_1| \simeq O(\frac{n}{2})|H_2| \simeq O(n), |H_3| \simeq O(\frac{n}{4})$,

where H stands for Hidden Layer, n is the number of input nodes and O is the output layer; or by following another technique, having excellent performance in image recognition, which provides a compression followed by an expansion:

- H_1 s.t. $|H_1| \simeq O(n)$;
- H_2 s.t. $|H_1| \simeq O(n), |H_2| \simeq O(\frac{n}{4})$.

3.5 Creating Datasets

In order to train *Machine Learning* algorithms a knowledge base needs from which to extract a function assumed would be impossible to describe. The approach taken in this area, as mentioned, is definitely to obtain a classification panel function. The acquired data should be grouped into two sets T and S, respectively, the set for the *training* and *test*. Typically $T \subseteq S$, but it is good practice to use subsets such that $T \subset S$ so as to avoid a *overfitting* on the training data [13].

In these sets each vector is associated with the classification arbitrarily a–priori determined (supervised learning). This decision must be, however, mostly consisting of all S. Clearly classify the opposite $\forall v \in A \subset S$ and $\forall w \in B \subseteq S$ such that $A \cap B \neq \emptyset$ make all the more inaccurate the resulting function is greater than $|A \cap B|$.

3.6 Results

Learning from Labeled Data

The learning task has been traditionally studied as **Supervised Learning** framework [14]. This paradigm assumes that there is a training set consisting of data points x (from some set X) and their labels y (from some set Y), and the goal is to learn a function $f : X \to Y$ that will accurately predict the labels of data points arising in the future. In practice, labels must be explicitly associated to the patterns. This work could become time expensive, but it is an important preparatory phase to be accurately performed by the Raman spectra experts. So, our *Teaching Output* is a Matrix $m \times n$, where:

Y_n columns are the memberships (or not) for each sample established for the training. The membership is identified with a value equal to 1;

X_m rows are $\vec{x}$ vectors to be read as the labels of samples. Therefore, each row of Teaching Matrix, originally filled with 0, can carry only one box with a value equal to 1.

In this way, the *Weight Matrix* is trained to learn the *Target Function*, in our case a correct classification of the Raman spectra. So, the expected result, in the subsequent test, is a new *Output Matrix*, approximating with asyntothic trend 0.0, except for the diagonal, which data accuracy are near to 1.0 for every single box. For example, all the matrix elements $a_{i,j}$ (with $i = j$), in the case of exactly the same Inputs for training and testing. Having acquired this result with the desired threshold of the permitted error (i.e. the difference between Teaching Output and Output Matrix), ANN can be applied to any other input that meets the requirements of the reference domain (result that has been reached).

Procedure and Testing Results

The used tool has been Scilab version 5.4.1 [15] with third party ANN ToolBox version 0.4.2.5 [16]. The training uses a standard on-line (i.e. it applies incremental methods since the parameters are updated at each individual pattern of the training set) back-propagation algorithm with Bias, containing a Sum Squared Error (SSE) performance function.

All analyzed samples were combined into a single matrix, cutting off the initial ramp and the final part of the Raman spectra which do not contain significant informations.

The second step consisted to normalize the samples matrix inside [-1,1] interval: it is a requirement for the used activation function, the symmetric sigmoid (3D graphic representation of this matrix is shown in Fig. 12).

In this work, we made the choice to not report both the intuition which leaded to design several hidden layers logically divided into two functional blocks as well as the numerous and tedious attempts to finally get a robust and general neural network, since this process explicits in its approach to ANN architecture

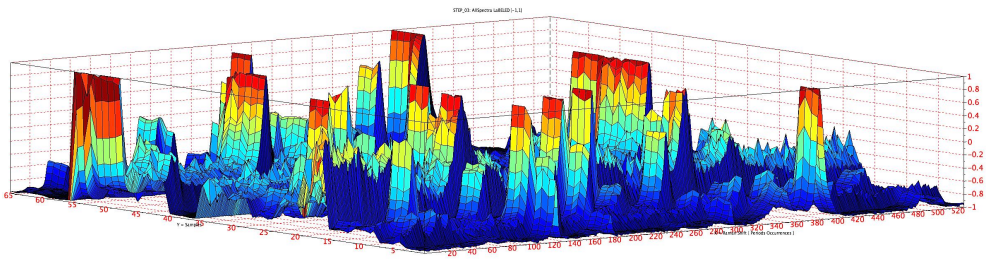

Figure 12. *All the spectra, covering* 16 *types of different substances, are included in this matrix normalized in the range* $[-1, 1]$. *There are* 66 *samples. After the cutting operation, on top and tail, spectra still maintain more than* 536 *significant data, that are sufficient to have appropriate patterns.*

(all that started from previous experiments based on ANN *autoencoder* architecture [17] applied in the trend prediction of certain physical phenomena). In fact, the topology of this ANN, full connected, has been so organized (the round brackets enclose nodes cardinality):

- $m = 536$, i.e. the number of input nodes
- $k = 16$, i.e. the number of membership classes

Network allotment for an acceptable Input replica

$$\boxed{\mathsf{I}(m)} \gg \boxed{H_1(\simeq (1.5 \times m))} \gg \boxed{H_2(\simeq (0.5 \times m))} \gg \boxed{H_3(\simeq (1.5 \times m))}$$

Hypothetical Output (suppressed layer)

$$\boxed{\mathsf{O}(m)}$$

Network allotment for classifier

$$\boxed{H_4(\simeq (0.22 \times m))} \gg \boxed{H_5(\simeq (0.08 \times m))} \gg \boxed{\mathsf{O}(k)}$$

Consequently, to connect the two functional blocks, links were added among the nodes of H_3 and H_4 hidden layers. Here it is worth underlining that the real aim of ANN first functional block is to emphasize the slopes and peaks of Raman spectra to generate new patterns that are more suitable to increase the probability of correctly discriminate the spectra (this is the mission of ANN second functional block).

Other parameters were:

- Weights matrix initialited to 0.0
- BIAS (threshold activation) initialited to [-1,1]
- Epochs number: fixed to 1500
- Learning Rate: 2.5

- Momentum: 0.0

The epochs number was the best balance among the desired result and computational available resources and this number can be determined only after the running of several tests.

The testing, conducted with the same samples used for the training, but entered to the network in a random order, has provided the results reported in Tab. 2.

ANN Performance (best values nearest to 1)			
Substances	**mean**	**high**	**low**
Picric Acid	0.990046	0.990148	0.990003
HMX	0.990525	0.991083	0.990003
N–Guanidine	0.990030	0.990099	0.990002
Urea Nitrate	0.990146	0.990300	0.990001
Nitrocellulose	0.990298	0.991191	0.989998
PETN	0.990299	0.990434	0.990003
RDX	0.990308	0.990485	0.990000
Tetryl	0.990219	0.990439	0.990003
TNT	0.853106	0.990094	0.443398
Azide	0.990643	0.990893	0.989999
HMDT	0.990597	0.991189	0.989999
TATP	0.990643	0.991688	0.989998
TATP Art	0.989101	0.989152	0.989050
Potassium chlorate	0.990275	0.990451	0.990007
Styphnic Acid	0.990024	0.990046	0.990003
Ammonium Picrate	0.990341	0.990823	0.990007

Table 2. *Results of ANN identification testing procedure.*

The worst result of Tab. 2 was obtained for the TNT Raman spectrum reported in Fig. 13, curve (b), which has SNR = 15 much lower than that of Raman spectra used for the training procedure, i.e. SNR = 200 as curve (a) of Fig. 13. The noisy spectrum has been intentionally introduced in the testing data set to verify the real potentiality of the identification algorithm.

This specific ANN architecture, independently on the used platform is very time consuming in the training phase, but it has a wide range of applications. It has been also successfully tested on spectra of biological samples (Spores and Bacillus), that have a very different behaviours when scanned with Raman device. Finally, we can confirm that featuring the network topology with two functional hidden blocks, we reached the desired results.

In all case, the experiment was conducted on a MacBook Pro with Mac OSX Lion as operative system, so configured:

CPU 2, 6 GHz IntelCore i7;

RAM 16 GB 1600 MHz DDR3.

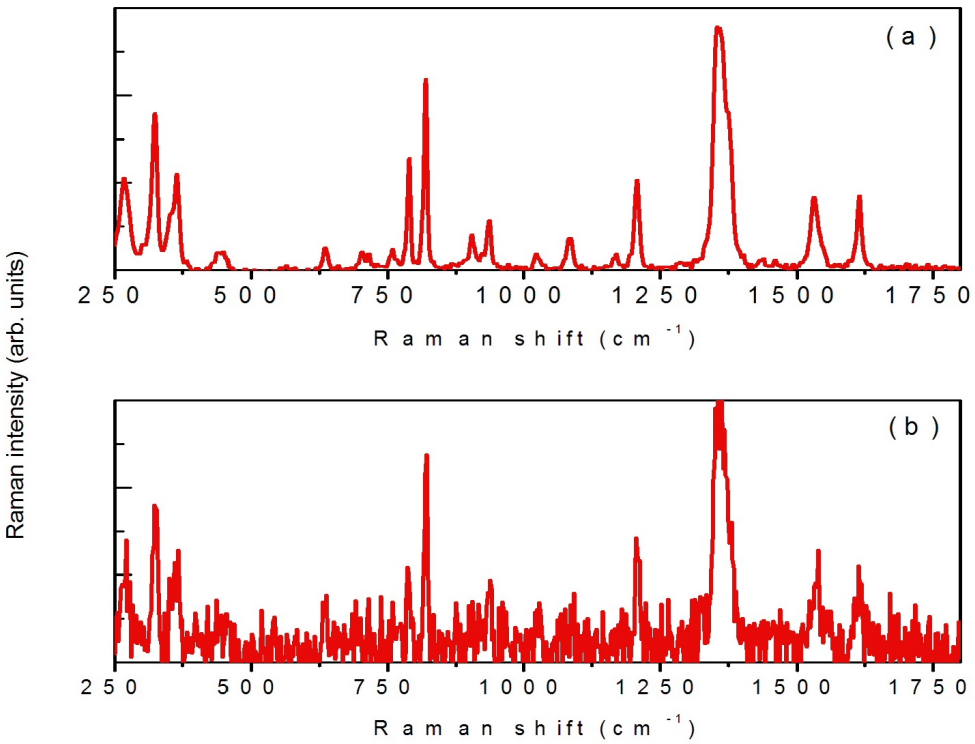

Figure 13. *TNT Raman spectra for* 10 *s acquisition time: (a)* 150 *mW laser power, SNR* = 200*; (b)* 30 *mW laser power, SNR* = 15.

The input matrix of patterns consisted of 66 samples spread over 536 intervals. The global time of training required about 24 hours for 1500 epochs, where the total of groups of elementary floating point computations, excluding the intervention of BIAS nodes, were estimated to be slightly more 1440 millions runnings.

4 Conclusions

In this work a customised auto–encoder ANN was developed to identify up to 16 chemical compounds from their Raman spectra. Due to the partial overlapping of spectral signature of some compounds in the studied set, the PCA and HCA approaches produced clusters of spectra that not always resulted in good agreement with the chemical classification.

On the contrary, the proposed procedure has demonstrated a full capability to discriminate Raman spectra, also in the case of noisy real curves, leading to identification of the corresponding chemical compounds without any miss–assignment. In addition, it is worth to underline that the developed algorithm is perfectly translable in **C** language, to be compiled on a processor connected to a Raman spectrometer for an alarm system to be used in the field. This operation, however, requires the introduction of FANN Library instead of ANN ToolBox.

Acknowledgments The authors gratefully acknowledge Maj. Luigi Cassioli, Cap. Silvana Grossi and Lt. Col. Leonardo Mariani (Aeronautica Militare Italiana, Comando Logistico) that, in the frame of the ENEA–Italian Ministry of Defence agreement, supplied and handled the chemical compounds used in the Raman experiments performed at ENEA UTAPRAD-MNF Laboratory.

Bibliography

[1] P. R. Griffiths I. R. Lewis, D. N. W. Daniel Jr. Interpretation of Raman spectra of nitro–containing explosive materials. Part I: group frequency and structural class membership. *Appl. Spectroscopy*, 51(12):1854–1857, 1997.

[2] P. R. Griffiths D. N. W. Daniel Jr, I. R. Lewis. Interpretation of Raman spectra of nitro–containing explosive materials. Part II: the implementation of neural, fuzzy and statistical models for unsupervised pattern recognition. *Appl. Spectroscopy*, 51(12):1868–1879, 1997.

[3] T. J. Glynn A. G. Ryder, G. M. O'Connor. Quantitative analysis of cocaine in solid mixtures using Raman spectroscopy and chemometric methods. *J. Raman Spectrosc.*, 31:221–227, 2000.

[4] P. Donfack R. M. El-Abassy and A. Materny. Visible Raman spectroscopy for the discrimination of olive oils from different vegetable oils and the detection of adulteration. *J. Raman Spectrosc.*, 40(9):1284–1289, 2009.

[5] E. Manzano C. Cardell N. Navas, J. R. Pastor. Raman spectroscopic discrimination of pigments and tempera paint model samples by principal component analysis on first-derivative spectra. *J. Raman Spectrosc.*, 41:1486–1493, 2010.

[6] L. Cantarini A. Palucci A. Puiu A. Rufoloni S. Botti, S. Almaviva. Trace level detection and identification of nitro–based explosives by surface–enhanced Raman spectroscopy. *J. Raman Spectrosc.*, 44:463–468, 2013.

[7] T. Munshi J. E. Cummingham E. H. Linfield A. G. Davies H. G. M. Edwards M. D. Hargreaves, A. D. Burnett. Comparison of near infrared laser excitation wavelengths and its influence on the interrogation of seized drugs–of–abuse by Raman spectroscopy. *J. Raman Spectrosc.*, 40:1974–1983, 2009.

[8] W. Pitts W. S. McCullock. A logical calculus of the ideas imminent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:115–133, 1943.

[9] D. O. Hebb. *The organization of behaviour, a neuropsychological theory.* Wiley, New York, 1949.

[10] M. E. Hoff B. Widrow. Adaptive switching circuits. *1960 IRE Western Electrique Show and Convention Record*, 23, 1960.

[11] M. O. Rabin. Automata on innite objects and church's problem. *American Mathematical Society*, MA, USA, 1972.

[12] S. A. Dudani. The distance-weighted k-nearest-neighbour rule. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-6, 4:325327, 1976.

[13] Y. Liu X. Yao. Machine learning. *Search Methodologies*, Springer US:341373, 2005.

[14] R. J. Marks R. D. Reed. *Supervised learning in feedforward artificial neural network*. MIT Press, Cambridge, USA, 1998 (ISBN:0262181908).

[15] http://www.scilab.org/download/5.4.1.

[16] R. M. Hristev. The ANN book (Edition 1), 1998.

[17] G. Dipoppa, M. Martinelli, E. Tronci and C. Balducelli. Electric power system anomaly detection using neural networks. *Proceedings of Knowledge-Based Intelligent Information and Engineering Systems, 8th Conference, Wellington, New Zealand*, 20–25 September, 2004.

Edito dall' ENEA
Servizio Comunicazione

Lungotevere Thaon di Revel, 76 - 00196 Roma

www.enea.it

Stampa: Tecnografico ENEA - CR Frascati
Pervenuto il 8.7.2014
Finito di stampare nel mese di luglio 2014