A Neural Network Algorithm for Complex Pattern Classification Problems

Allan de Medeiros Martins\textsuperscript{1,2}, Adrião Duarte Dória Neto\textsuperscript{1}, Jorge Dantas de Melo\textsuperscript{1}

\textsuperscript{1}Departamento de Engenharia de Computao e Automao - DCA - UFRN
\textsuperscript{2}Curso de Engenharia de Computao - Universidade Potiguar
E-mails: allan@dca.ufrn.br, adriao@dca.ufrn.br, jdmelo@dca.ufrn.br

Abstract

This work presents an application of neural networks in pattern classification. A new algorithm for automatic classification of data is presented. That algorithm make use of a competitive neural network to aid the classification process. The algorithm gets a data set D and segment it into clusters. The only a priori information given is a number of auxiliary centers and a threshold distance. The algorithm uses the Mahalanobis metrics to cluster the data and find itself the number of classes.

1. Introduction

Many areas like data mining, image segmentation, pattern recognition, statistical data analysis make use of pattern recognition to archive some task that is part of a process. The main objective of a good pattern classification algorithm is to separate classes distributed arbitrarily in the data space, in a non supervised way. Some technics to archive this task have been developed and are based on several heuristics. The most simple way of classification is the use of vector quantization technics like k-means \cite{1} or pure competitive neural networks \cite{2} to find centers that represents the clusters. The similarity measure used in most of this technics is the euclidian distance between the point and the center of its class. More elaborate technics, using Kohonen(SOM) maps \cite{3} or Fuzzy k-means \cite{4} have been developed. Some modifications of the usual SOM algorithm bases in the segmentation of the output map was also been developed \cite{5} and give good results, but need complex computation tasks. In most used technics, the number of classes that exists in the data set must be given \textit{a priori}. In some cases this information is not available, so its important to develop algorithms that perform the automatic classification without this information. Another problem in the pattern classification, is the complexity of the spatial distribution of the data set. The proposed algorithm use a simple competitive neural network and a linking heuristic of auxiliary centers to cluster similar regions using the \textit{Mahalanobis} distance \cite{6} as metric of similarity between points and its centers. The incorporation of the spatial statistics of the data give us a good measure of the distribution of the points, making possible the algorithm to be used to classify very complex data set. The only two \textit{a priori} informations about the data set given to the algorithm is the number of auxiliary centers and a threshold distance.

The article is organized as follows; in the section 2 we will be described the competitive neural networks used to position the auxiliary centers; in the section 3 we will describe the \textit{Mahalanobis} metric. The section 4 shows the proposed algorithm; in the section 5 some results are presented and in the section 6 we present some conclusions.

2. Competitive neural networks

In the proposed algorithm we use auxiliary centers which are distributed in a uniform manner through the data set. This distribution is made using a conventional competitive neural network.

A competitive neural network is a self-organizing neural network that archives vector quantization in a a given data set. The network has an input layer and a output layer as show in the figure 1

![Figure 1: Competitive neural network topology](image)

The input layer of the network takes the training patterns each of then corresponding to a point in the data set. Each neuron in the network will be a center of a certain class that will quantize the total number of points in the data set. The training process of a competitive neural network is based on the update of weights of the winner neuron, that is, the neuron with smallest distance to a given input pattern. In the competitive training, only one neuron is updated for each input pattern, this approach is called \textit{winner takes all}. The similarity criteria is the Euclidian distance between the input pattern and the weights of the neuron. The weights are updated as presented in the equation (1) \cite{2}:

\begin{equation}
\begin{aligned}
W_c(n) &= W_c(n) + \Delta W_c(n) \\
\Delta W_c(n) &= \alpha \eta(n) (X_n - W_c(n))
\end{aligned}
\end{equation}
where $W_c(n)$ are the weights of the winner neuron in the iteration $n$, $\Delta W_c(n)$ is the difference that will be added to the winner neuron, $\alpha$ is the learning parameter and $\eta$ is a convergence parameter. $\eta(n)$ is updated at each iteration having its value decreased to avoid oscillations in the convergence. In general we take $\eta(n + 1) = \eta(n)\tau$, where $\tau$ is a constant. $X_n$ is the input pattern.

After some iteration, the weights of the networks’ neurons converge to the centers that represent the entire data set.

3. Mahalanobis metrics

The Mahalanobis metrics is a similarity measure that consider the spatial statistics of the points where the measure is been made [7]. The distance between two points $p_1$ and $p_2$ inside a space where the covariance matrix of the distribution of the probability that represents the spatial statistics is $C$, is given by:

$$d_m(p_1, p_2, C)^2 = (p_1 - p_2)^T C^{-1}(p_1 - p_2) \quad (2)$$

The figure 2 shows the effect of the spatial probability in the distance of two points. In that figure, even the distance $d_1$ looks greater (in terms of Euclidian distance) it is smaller than the distance $d_2$. That difference exists because the covariance matrix of the spatial distribution gives a greater weight to the distances that doesn’t been in the direction of the distribution.

Figure 2: Distance measurement in a space with non-uniform probability distribution

The Mahalanobis distance is important in classification problems because the information about spatial distribution of the points if incorporated in the metrics. The spatial distribution can be represented by the statistics of the data set where we wish to measure, in some way, the distance between two points.

We can consider the Euclidian distance, a special case of the Mahalanobis distance, where the data would be uniformly distributed. That case corresponds a distribution where the covariance matrix be a diagonal matrix. In that sense, the Mahalanobis metrics become a general case of distance measurements and suitable to be used in classification problems.

4. The algorithm

The figure 3 shows a set of bidimensional points where we have two clusters of data representing two regions of interest. The goal of the pattern classification is to separate this distribution in two regions (two classes) and, given a point out of the data set, to say if that point belongs to a class or another.

At first, the algorithm finds $N_a$ auxiliary centers that separate the data set in $N_a$ regions according with euclidian distance. For this task, we use a competitive neural network that localize the auxiliary centers. The number of auxiliary centers must be chosen in order to divide the region so much as possible without slowing the algorithm. In general, one choice based in a percentage of the total number of points is suitable.

After the choice of the auxiliary centers, the data set is divided in $N_a$ regions that still not correspond to the real clusters of the data set, as shows in the figure 4. To cluster the regions, for each center, the Mahalanobis distance is measured between that center and all others. Each distance is compared with a threshold distance. If that distance is less than the threshold, the two centers are linked.

The covariance matrix used for the Mahalanobis distance between the auxiliary centers is an estimate of the covariance matrix of the distribution formed by the regions classified for both auxiliary centers. This is made in the following way; be $\{x_1, x_2, \ldots, x_n\}$ points that belong
to the region of one of two centers where we want to calculate the Mahalanobis distance and \( m \) the mean point of the data set. The covariance matrix \( C \) can be estimated as shown in the equation (3) [8].

\[
C = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - m)(x_i - m)^T
\]  

(3)

Using the equation (3) is possible to estimate the covariance matrix of the data where the auxiliary centers are in. With this, it is possible to calculate the Mahalanobis distance between each pair of auxiliary centers. The figure 5 shows the calculation of the distance between two centers whose regions where used to estimate the covariance matrix.

![Figure 5: Mahalanobis distance between 2 centers](image)

In the figure 5, only the sets marked with crosses where used to estimate the covariance matrix used in the calculation of the Mahalanobis between \( c_1 \) and \( c_2 \). In this way, points whose data are "aligned" will have a distance less than points where the data aren’t in the variation of the whole set. According with the measured distance between each center each others it is established the concept of linking between centers. If the measured Mahalanobis distance where less than a threshold \( d_i \), the centers are linked and, from this moment, the two regions became one. If a link it will be made to a center that already was linked to another center, so the data of these 3 centers become a single region and so one. At the end of the process, the set will be divided in \( N \) regions, each one defined by one or more linked centers. The figure 6 shows two regions at the end of linking process.

The number os classes found by the algorithm will be equal to the number os set of linked auxiliary centers. In the worst case, the number os classes will be equal to the number of auxiliary centers. To classify a point out of the data set, we find to which auxiliary center it belongs (using the euclidian metrics) and so, which cluster that center belongs. That cluster will be the cluster at which that point belongs to.

The outline of the algorithm is shown follow

![Figure 6: Result of the link of centers based on the a threshold distance \( d_i \)](image)

1. Choice \( N_a \) and \( d_i \) for the data set that we want to classify
2. Calculate the \( N_a \) centers using a competitive neural network.
3. For all centers \( c_i \), compute the distance \( d_{ij} = d_m(c_i, c_j, C) \) to all others \( (i \neq j) \), where \( C \) is the estimate of the covariance matrix of the data of the class where the center is \( c_i \).
4. If for a center \( c_i \) the distance \( d_{ij} \) where less than \( d_i \) link \( c_i \) with \( c_j \).

As we can see see in the algorithm above, the final result depends on the parameters \( N_a \) and \( d_i \). The choice of the number of auxiliary centers \( N_a \) is less critical because it is sufficient to put a large number of centers until its represents the spatial tendency of the data set. The implication of choice of a very large number os auxiliary centers is the computational cost spent because, for each center, we have to measure the Mahalanobis distance to each others. The threshold distance \( d_i \) is an empirical parameter and it depends on the spatial distribution of the data set. A good way to choice that threshold is to normalize the data set before its classification.

5. Results

The proposed algorithm was tested in many data sets that have a complex spatial distribution. Even the tests where made with bi and tree-dimensional data sets, the algorithm can be used in data sets with higher dimensions. For comparison purpose, the sets where also classified with pure competitive neural networks. The figures 7 to 12 shows the data sets and the results obtained with the neural network and with the proposed algorithm. The figure 7 shows the classification of a non-linear separable data set with the neural network. One can see the wrong classification of some points because the euclidian metrics. In the figure 8 we have the classification with the proposed for \( N_a = 10 \) and \( d_i = 100 \). The figures 9 and 10
shows the same procedure however with a data set more difficult to separate. In the figures 11 and 12 the algorithm is compared in a mostly interlaced data set.

![Figure 7: Data set classified with neural network](image)

![Figure 8: Data set classified with the proposed algorithm \(N_a = 10 e d_t = 100\)](image)

![Figure 9: Data set classified with neural network](image)

![Figure 10: Data set classified with the proposed algorithm \(N_a = 15 e d_t = 100\)](image)

![Figure 11: Data set classified with neural network](image)

![Figure 12: Data set classified with the proposed algorithm \(N_a = 60 e d_t = 100\)](image)

6. Conclusions

The proposed algorithm has separated data in a automatic way without the need to inform \textit{a priori} the number of classes. This kind of problem is still less explored and doesn’t has a completely efficient solution. The algorithm showed itself efficient in the classification of very complex data generated artificially like presented in the results section. The good classification still depends on the choice of \(d_t\) and \(N_a\) that well represents the statistics separation of the data, making possible the correct link
of the auxiliary centers. In general the algorithm is an alternative efficient tool to be applied in problems where we have a spatially complex distributed data and we can perform several tests to $d_i$ and $N_d$ values.

The algorithm can be applied to several problems at several areas like data mining, pattern recognition, signal and image processing and any problem that involves cluster analysis and data classification.

**Acknowledgment**

We want to thanks to Dr. Pablo Javier Alsina and Dr. Francisco das Chagas Motta for the comments and suggestions.

**References**


