Learning and Nonlinear Models (L&NLM) – Journal of the Brazilian Neural Network Society, Vol. 8, Iss. 4, pp. 240–252, 2010

AN EXPERIMENTATION WITH IMPROVED TARGET VECTORS FOR MLP IN CLASSIFYING DEGRADED PATTERNS

Shigueo Nomura, José Ricardo Gonçalves Manzan, Keiji Yamanaka
Faculty of Electrical Engineering - FEELT
Federal University of Uberlândia - UFU
shigueonomura@feelt.ufu.br, josericardo@iftm.edu.br, keiji@ufu.br

Abstract – In this paper, we adopt unconventional target vectors to improve the performance of pattern classification systems using neural network techniques based on MLP. Instead of conventional target vectors, the new target vectors are bipolar, orthogonal, and highly dimensional. Since they are orthogonal with bipolar representation, we can take advantage of increasing on the Euclidean distance for these vectors when their number (n in a Euclidean space $\mathbb{R}^n$) of components increases. We define non-orthogonal bipolar vectors considered as conventional target vectors for comparison purposes. Those non-orthogonal bipolar vectors provide a fair reference to ensure the effectiveness of the adopted unconventional target vectors and in justifying the credibility and validity of experimental results. The conventional and unconventional target vectors are used in the experiments for training MLP models by backpropagation algorithm to classify patterns extracted from actual degraded images. Comparison of experimental results lead to conclusions that classification performances of MLP models considerably improved with the adopted unconventional target vectors in classifying degraded patterns.
Keywords – Multilayer perceptron model, conventional target vector, orthogonal bipolar vector, non-orthogonal bipolar vector, degraded image, pattern recognition

1 Introduction

Several artificial neural networks (ANN) have been developed and analyzed over the last few decades [1]. Multilayer perceptron (MLP) as one model of ANN has evolved over the years as a very powerful technique for solving a wide variety of problems. Much progress has been made in improving the MLP performance and in understanding how the MLP operates [1]. For instance, the arising of an effective general method of training MLPs known as backpropagation or the generalized delta rule [2–4] played a great role in the reemergence of ANN as a tool for solving problems [5].

The MLP net as a multilayer, feedforward net trained by backpropagation, can be used to solve problems in several areas [1, 6–11] due to the very general nature of the backpropagation training method [5]. Particularly, the MLP model applied to pattern recognition [12–16] process has lead to important advances and shown potential as a classification system. It has been successfully applied to pattern classification tasks. That success is due to the MLP qualification to imitate the learning capacity of the human brain. Pattern classification tasks involve mapping a given set of inputs to a specified set of target outputs for the nets that are based on supervised training [5].

As a classification system, MLP requires a good approach for analyzing degraded image data, extracting features from these data, generating a set of relevant information, and improving its performance.

A strategy to improve the MLP performance consists of training the net to achieve a balance between the ability to recognize correctly the input patterns from the training set and the ability to provide acceptable responses to input that is similar, but not identical (ability for generalization).

Many efforts have attempted to develop a good method followed by feature extraction systems [17, 18]. However, the need for additional improvements in training MLPs still exists since the training process is very chaotic in nature.

Surprisingly, it is quite difficult to find studies on target vectors (outputs) or expectation values for learning and their importance to the MLP performance improvement in classifying patterns. Conventional target vectors that have been widely adopted in various applications are binary or bipolar, and their sizes are based on the number of classes of input patterns.

In this work, an approach for improving the MLP ability for generalization is experimented by breaking away from such conventional usage of target vectors.

This work experimentally analyzes the performance improvement of MLP using unconventional target vectors with bipolar representation, orthogonality, and high dimensionality [19]. The idea is based on the fact that the MLP performance using bipolar target vectors is already better than performance with binary ones [5]. But no work exists, to our knowledge, adopting orthogonal bipolar vectors as expectation values for learning and same-sized non-orthogonal bipolar vectors as reference target vectors to show a convincing experimental comparison of MLP performances.

Section 2 describes the multiclass learning problem and classification based on the Euclidean distance for orthogonal bipolar vectors. Section 3 presents the theoretical background including the definition of vectors used in the experiments, and analysis of Euclidean distance increase for proposed orthogonal bipolar vectors. The strategy of the proposed approach is presented in Section 4. In Section 5, the experimental procedure for evaluating the proposed approach is presented. Section 6 describes experiments and presents results showing that the orthogonal bipolar vectors as target ones improve MLP performance on degraded pattern classification. Section 7 discusses the experimental results.

2 Multiclass Classification Using Multilayer Perceptron (MLP)

It is known that connectionist algorithms are more difficult to apply to multiclass problems [20]. Multiclass classification problems correspond to tasks of finding an approximate definition for an unknown function $f(x)$ given training examples of the form $\langle x_i, f(x_i)\rangle$. The unknown function f often takes values from a discrete set of "classes" $c_1, c_2, \ldots, c_k$. For example, in digit recognition [21], the function maps each hand-printed digit to one of $k = 10$ classes.

We can distinguish two approaches to handle these multiclass classification tasks as follows:

- We have one-per-class approach when the individual functions $f_1, f_2, \ldots, f_k$ are learned one for each class. To assign a new case p to one of these classes, each of individual function f_i is evaluated on p, and the case p is assigned to the class j corresponding to the function f_j that returns the highest activation [22]. This classification approach is standard for conventional target vectors.

- Distributed output code is an alternative approach pioneered by Sejnowski and Rosenberg [23] in their widely-known NETtalk [24] system. In this approach, each class is assigned to a unique binary string of length n; these strings refer to target vectors in MLP. Then n binary functions are learned, one for each bit position in these binary strings. During training for an example from class i, the desired outputs of these n binary functions are specified by the target vector for class i. With MLP, these n functions can be implemented by the n output units of a single network. A new case p is classified by evaluating each of the n binary functions to generate an n-bit string s. This string is then compared to each of the k target vectors, and p is assigned to the class whose target vector is closest, according to some distance measure, to the generated string s.

3 Theoretical Foundation

The following defined vectors represent expectation values used in this experimental analysis.

Vectors are expressed in the form shown in Eq. (1):

$$V = (e_1, e_2, \ldots, e_n)^T, \tag{1}$$

where V is a vector, T indicates transposition, e_i represents a component for $i = 1, 2, \ldots, n$, and n is the number of components.

3.1 Orthogonal Bipolar Vectors (OBVs)

Orthogonal bipolar vector (OBV) [19] has bipolar representation, and the norm of U as an OBV in an Euclidean space $\mathbb{R}^n$ is given by Eq. (2):

$$|U| = \sqrt{{x_1}^2 + {x_2}^2 + \cdots + {x_n}^2} = \sqrt{n}, \tag{2}$$

where $U = (x_1, x_2, \ldots, x_n)^T$, x_i represents a component $+1$ or -1 for $i = 1, 2, \ldots, n$, and n is the number of components.

The usual inner product [25] between two vectors U and V in an Euclidean space $\mathbb{R}^n$ is defined by Eq. (3):

$$U.V = x_1 y_1 + x_2 y_2 + \cdots + x_n y_n, \tag{3}$$

where $V = (y_1, y_2, \ldots, y_n)^T$, y_i represents a component $+1$ or -1 for $i = 1, 2, \ldots, n$, and n is the number of components.

Vectors U and V are orthogonal (denoted by $U \perp V$) if and only if $U.V = 0$ in Eq. (4):

$$U \perp V \Leftrightarrow U.V = 0. \tag{4}$$

The similarity is measured by operating the inner product or scalar product (defined by Eq. (4)) on two vectors which produces a scalar. Then, the similarity between two OBVs is null.

3.2 Non-Orthogonal Bipolar Vectors (NOVs)

The following two reasons have led to the definition of non-orthogonal bipolar vector (NOV) [19] used as reference for target vector experimental comparison:

1. An MLP architecture varies with the number of output neurons, which is directly dependent on the size (dimension) of target vectors. Consequently, it is not possible to use the same MLP architecture for training models with different sizes of target vectors. In fact, as shown in Fig. 1, the number "n" of output neurons ($Z_1, \ldots, Z_k, \ldots, Z_n$) will decide the size "$m \times n$" of the weight matrix ($W_{jk}$) where ($j = 1, \ldots, m$; $k = 1, \ldots, n$). Table 1 presents a formal statement of the MLP architecture in Fig. 1. Since NOV can contain the same number of components as OBV, the same architecture of backpropagation MLP can be used for experimenting with OBVs as new target vectors.

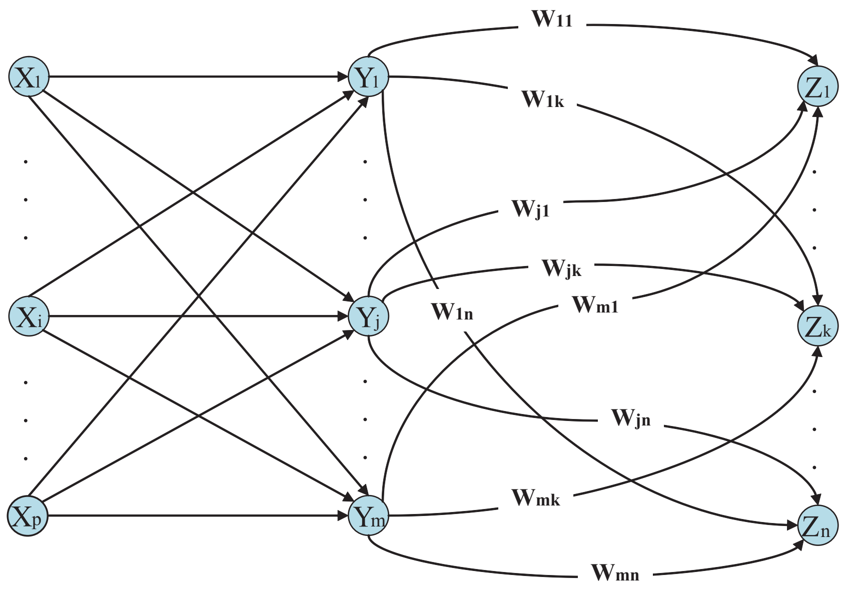


Figure 1: Backpropagation MLP architecture with one hidden layer.

Table 1: A formal statement of the MLP architecture in Fig. 1

Variables	*Description*
p	The number of input units
$X_1, \ldots, X_i, \ldots, X_p$	The input units
m	The number of hidden units
$Y_1, \ldots, Y_j, \ldots, Y_m$	The hidden units
n	The number of output units
$Z_1, \ldots, Z_k, \ldots, Z_n$	The output units
W_{jk}	Connecting weights between the hidden and output layers

Table 2: A formal statement for constructing a set of OBVs

Symbol	*Description*
2^k	The number of orthogonal vectors where $k = 1, 2, 3 \ldots$
s	The number of components in a seed vector where $s = 1, 2, 3 \ldots$
V_s^0	The seed vector with s components as part of OBVs
n	The number of components in an OBV calculated by Eq. (8)
V_n^i	The i^{th} OBV with n components
$fcc(V_s^0, \pm V_s^0)$	The function to concatenate 2 seed vectors and construct a new vector represented by V_{2s}^1 with $2s$ components

2. MLP classification performance by the backpropagation algorithm is strongly dependent on the choice of initial weights. This algorithm is based on the optimization technique known as gradient descent where the choice of these initial weights influences whether the net reaches a global (or only a local) minimum of the error [5]. Unfortunately, we can not algebraically determine exact initial weights to get this global minimum of error, and to provide the best performance of a MLP model trained with each type (NOV or OBV) of target vector. Therefore, NOVs must be constructed to be able to adopt the same initial weights for experimenting OBVs as new target vectors.

For reasons 1 and 2, NOV plays an important role as a fair reference to ensure the effectiveness of OBVs as new target vectors and in justifying the credibility and validity of experimental results presented in the next sections.

Non-orthogonal bipolar vectors (NOVs) are highly similar because they are larger than conventional ones, they are bipolar, and the angle between them is less than 90 degrees. In a general form, Eq. (5) defines NOV with n components for representing p^{th} pattern in q patterns as

$$V_p = (\overbrace{-1, \ldots, -1}^{p-1}, 1, \overbrace{-1, \ldots, -1}^{n-p})^T, \tag{5}$$

where V_p is the NOV for representing the p^{th} pattern, $p = 1, 2, \ldots, q$, q is the number of patterns, and $n > q$ is the number of components.

Equation (6) defines a form of NOV used in the experimental analysis. In the experiments, the MLP model classifies patterns of digits into ten classes.

$$V_i = (\overbrace{-1, \ldots, -1}^{i-1}, 1, \overbrace{-1, \ldots, -1}^{n-i})^T, \tag{6}$$

where V_i is the NOV for representing the i^{th} digit, $i = 1, 2, \ldots, 9$, and $n > 10$ is the number of components.

The digit “0” is defined as a 10^{th} digit by Eq. (7):

$$V_0 = (\overbrace{-1, \ldots, -1}^{9}, 1, \overbrace{-1, \ldots, -1}^{n-10})^T. \tag{7}$$

The similarity value (defined in Section 3.1) between two NOVs is greater than the corresponding value for OBVs. Also, if the NOVs expand then the corresponding similarity value increases.

Defined vectors (OBV, NOV) are used as target vectors to experimentally analyze the MLP performance improvement in classifying degraded patterns (digits).

3.3 Algorithm for Constructing Orthogonal Bipolar Vectors (OBVs)

As defined in Section 3.1, an OBV should satisfy two properties:

- Orthogonality. The inner product of any pair of OBVs must be zero.
- Bipolarity. A component in an OBV is $+1$ or -1.

The number of components in an OBV can be calculated by Eq. (8):

$$n = 2^k s. \tag{8}$$

A set of 2^k mutual OBVs is constructed with $2^k s$ components [5]. Table 2 presents the formal statement to construct a set of OBVs.

The algorithm for constructing the set of vectors is as follows:

- 1^{st} step: Initialize s and k, where $s = 1, 2, 3 \ldots$ and $k = 1, 2, 3 \ldots$.

Table 3: An example of OBV with 32 components

Digit	OBV^T
"1"	(1,-1,-1,1,-1,1,1,-1,-1,1,1,-1,1,-1,-1,1,-1,1,1,-1,1,-1,-1,1,1,-1,-1,1,-1,1,1,-1)
"2"	(1,-1,-1,1,-1,1,1,-1,-1,1,1,-1,1,-1,-1,1,1,-1,-1,1,-1,1,1,-1,-1,1,1,-1,1,-1,-1,1)
"3"	(1,-1,-1,1,-1,1,1,-1,1,-1,-1,1,-1,1,1,-1,-1,1,1,-1,1,-1,-1,1,-1,1,1,-1,1,-1,-1,1)
"4"	(1,-1,-1,1,-1,1,1,-1,1,-1,-1,1,-1,1,1,-1,1,-1,-1,1,-1,1,1,-1,1,-1,-1,1,-1,1,1,-1)
"5"	(1,-1,-1,1,1,-1,-1,1,-1,1,1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,1,-1,1,-1,-1,1,1,-1,-1,1)
"6"	(1,-1,-1,1,1,-1,-1,1,-1,1,1,-1,-1,1,1,-1,1,-1,-1,1,1,-1,-1,1,-1,1,1,-1,-1,1,1,-1)
"7"	(1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,-1,1,1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,1,-1)
"8"	(1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,1,-1,-1,1,1,-1,-1,1)
"9"	(1,-1,1,-1,-1,1,-1,1,-1,1,-1,1,1,-1,1,-1,-1,1,-1,1,1,-1,1,-1,1,-1,1,-1,-1,1,-1,1)
"0"	(1,-1,1,-1,-1,1,-1,1,-1,1,-1,1,1,-1,1,-1,1,-1,1,-1,-1,1,-1,1,-1,1,-1,1,1,-1,1,-1)

- 2^{nd} step: Initialize the seed vector with s components $\Rightarrow V_s^0 = (\overbrace{1, 1, \ldots, 1}^{s})^T$. For example, if $s = 2$ then $V_2^0 = (1, 1)^T$.
- 3^{rd} step: Calculate the number (n) of components in an OBV using Eq. (8). For example, if $(s = 2)$ and $(k = 2)$ then $n = 2^2 \times 2 = 8$.
- 4^{th} step: Construct
  $V_{2s}^1 = fcc(V_s^0, V_s^0)$,
  $V_{2s}^2 = fcc(V_s^0, -V_s^0)$. For example, $V_{2\times2}^1 = fcc(V_2^0, V_2^0) = (1, 1, 1, 1)^T$ and $V_{2\times2}^2 = fcc(V_2^0, -V_2^0) = (1, 1, -1, -1)^T$
- 5^{th} step: Construct
  $V_{4s}^1 = fcc(V_{2s}^1, V_{2s}^1)$,
  $V_{4s}^2 = fcc(V_{2s}^1, -V_{2s}^1)$,
  $V_{4s}^3 = fcc(V_{2s}^2, V_{2s}^2)$,
  $V_{4s}^4 = fcc(V_{2s}^2, -V_{2s}^2)$.
  For example,
  $V_{4\times2}^1 = fcc(V_{2\times2}^1, V_{2\times2}^1) = (1, 1, 1, 1, 1, 1, 1, 1)^T$
  $V_{4\times2}^2 = fcc(V_{2\times2}^1, -V_{2\times2}^1) = (1, 1, 1, 1, -1, -1, -1, -1)^T$
  $V_{4\times2}^3 = fcc(V_{2\times2}^2, V_{2\times2}^2) = (1, 1, -1, -1, 1, 1, -1, -1)^T$
  $V_{4\times2}^4 = fcc(V_{2\times2}^2, -V_{2\times2}^2) = (1, 1, -1, -1, -1, -1, 1, 1)^T$.
- 6^{th} step: Continue until 2^k orthogonal vectors with n components have been constructed $\Rightarrow V_n^1, \cdots, V_n^{2^k}$. For example, if $(s = 2)$ and $(k = 2)$ then $2^k = 4$ OBVs with 8 components already have been constructed until 5^{th} step. However, if $k = 3$ then the 6^{th} step will be executed to construct $2^k = 8$ OBVs with 16 components in each vector.

Table 3 presents an example of OBV with 32 components constructed by the above algorithm.

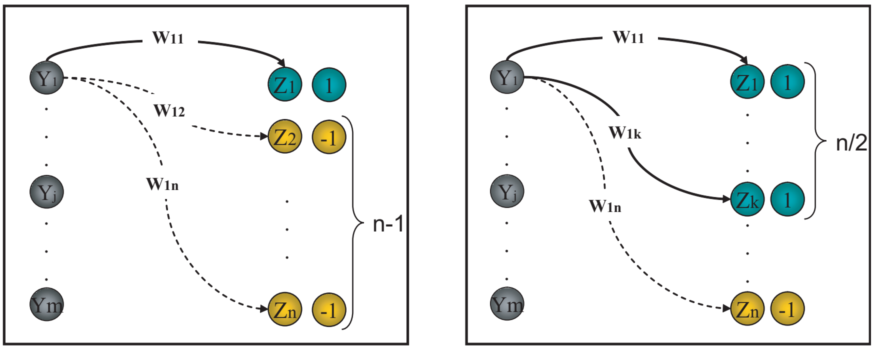
3.4 Number of Active Neurons for NOV and OBV

In the case of NOV defined in Section 3.2 and adopted as reference in our experimentation, its number of active neurons is only one as shown in Fig. 2(a). In this case, it means that only the neuron $Z1$ is active (represented by "1") whereas the others are not active (represented by "-1").

On the other hand, the minimum number of active neurons for OBVs is half of total number (n) of output neurons as shown in Fig. 2(b). Also, this number of active neurons increases according to the increase in the OBV size. This property leads to the increase in Euclidean distance between two OBVs as analyzed next. Then, this evidence explains our experimentation hypothesis that OBVs as target vectors can improve the performance of MLP in classifying degraded patterns.

3.5 Euclidean Distance Analysis for NOVs

Different sizes of NOVs (V_n^1 and V_n^2) defined in Section 3.2 can be constructed to represent two different digits "1" and "2", and calculate the Euclidean distance (d_n) for these vectors in a vector space $\mathbb{R}^n$, where n is the number of components as follows:



(a) NOV (b) OBV

Y1, ..., Yj, ..., Ym	Hidden units
Z1, ..., Zk, ..., Zn	Output units
Wjk (j = 1, ..., m; k = 1, ..., n)	Weights

Figure 2: Number of active neurons for NOV and OBV.

- NOV with 2 components:
  $V_2^1 = (1, -1)^T$
  $V_2^2 = (-1, 1)^T$
  $d_2 = \sqrt{(1+1)^2 + (-1-1)^2} = 2\sqrt{2}$
- NOV with 3 components:
  $V_3^1 = (1, -1, -1)^T$
  $V_3^2 = (-1, 1, -1)^T$
  $d_3 = \sqrt{(1+1)^2 + (-1-1)^2} = 2\sqrt{2}$
- NOV with 4 components:
  $V_4^1 = (1, -1, -1, -1)^T$
  $V_4^2 = (-1, 1, -1, -1)^T$
  $d_4 = \sqrt{(1+1)^2 + (-1-1)^2} = 2\sqrt{2}$
- NOV with n components:
  $V_n^1 = (1, -1, \overbrace{-1, \ldots, -1}^{n-2})^T$
  $V_n^2 = (-1, 1, \overbrace{-1, \ldots, -1}^{n-2})^T$
  $d_n = \sqrt{(1+1)^2 + (-1-1)^2} = 2\sqrt{2}$

This analysis shows that the Euclidean distance (d_n) is invariable and equal to $2\sqrt{2}$, that is, the distance for non-orthogonal bipolar vectors does not depend on their size.

3.6 Euclidean Distance Analysis for OBVs

Different numbers (n) of components for OBVs are calculated by Eq. (8) defined in Section 3.3. For instance, OBVs (V_n^1 and V_n^2) can be constructed to represent the digits "1" and "2" and calculate the Euclidean distance (d_n) for these vectors in a vector space $\mathbb{R}^n$, where n is the number of components as follows:

- OBV with 2 components:
  $V_2^1 = (1, -1)^T$
  $V_2^2 = (1, 1)^T$
  $d_2 = \sqrt{(1+1)^2} = \sqrt{2 \times 2}$

- OBV with 4 components:
  $V_4^1 = (1, 1, 1, 1)^T$
  $V_4^2 = (1, 1, -1, -1)^T$
  $d_4 = \sqrt{(1+1)^2 + (1+1)^2} = \sqrt{2 \times 4}$
- OBV with 6 components:
  $V_6^1 = (1, 1, 1, 1, 1, 1)^T$
  $V_6^2 = (1, 1, 1, -1, -1, -1)^T$
  $d_6 = \sqrt{(1+1)^2 + (1+1)^2 + (1+1)^2} = \sqrt{2 \times 6}$
- OBV with $n = 2s$ components:
  $V_n^1 = (\overbrace{1, 1, \ldots, 1}^{n})^T$
  $V_n^2 = (\overbrace{1, 1, \ldots, 1}^{m}, \overbrace{-1, -1, \ldots, -1}^{m})^T$
  $d_n = \sqrt{\underbrace{(1+1)^2 + (1+1)^2 + \ldots (1+1)^2}_{m}} = \sqrt{2n}$

In using OBVs as target vectors, the Euclidean distance (d_n) increases according to the increase in the number (n) of components. The distance between two OBVs with n components is then calculated by the equation $d_n = \sqrt{2n}$.

3.7 Euclidean Distance Increase for NOVs and OBVs

For the distance increase analysis, the number of components in each vector was varied from 2 to 64 incremented by 2. We can verify that the distance for 64 component-NOV ($dNOV_{64}$) compared to the distance for 64 component-OBV ($dOBV_{64}$) is

$$\left.\begin{array}{r} dOBV_{64} = 8\sqrt{2} \\ dNOV_{64} = 2\sqrt{2} \end{array}\right\} dOBV_{64} = 4 \times dNOV_{64}. \tag{9}$$

This increase of Euclidean distance for NOVs and OBVs is an important aspect to explain the MLP performance improvement when larger sizes of OBVs are used as target vectors in the next experiments.

4 Proposed Approach

This work aims to experimentally analyze the improvement of pattern classification rate using MLP based on the proposed new target vectors.

To justify the credibility and validity of experimental analysis presented in this work, the strategy consists of the following steps:

1. Construct non-orthogonal bipolar vectors to be used as conventional target vectors.
2. Construct orthogonal bipolar vectors to be used as new target vectors for the classification rate improvement.
3. Define a topology of MLP net to be learned with non-orthogonal vectors and orthogonal vectors used as target ones.
4. Compare experimental results using non-orthogonal bipolar vectors with those results using orthogonal bipolar vectors.

In summary, this approach takes advantage of non-orthogonal bipolar vectors as a fair reference to ensure the effectiveness of proposing orthogonal bipolar vectors as target ones.

5 Experimental Procedure

An MLP model is set to experimentally evaluate the proposed OBV-based approach.

Experimental data originated from license plate photos automatically taken by traffic control radars installed in Uberlândia City, Brasil. They are very degraded images with such problems as luminosity, contrast, focalization, resolution, and size, all of which require preprocessing able to extract relevant features for pattern recognition process. The original preprocessing methods proposed in such previous works as adaptive contrast enhancement [26], adaptive thresholding [27], automatic segmentation and extraction of feature vectors [28] were used.

Figure 3: The training set of 120 images, digitized on a 20×15 grid.

5.1 Data Representation

Segmented patterns (20×15) are represented by feature vectors with 300 components, and each component in the vector should represent one pixel of the pattern (bipolar value $+1$) or one pixel of the image background (bipolar value -1).

Figure 3 shows the 120-image training set representing digits as input data.

In this work, the adopted data representation is bipolar since the learning may be improved if the input is represented in bipolar form and the bipolar sigmoid is used for the activation function [5]. The reason is if one factor in the weight connection expression is the activation of the lower unit then units whose activations are null will not learn [5].

5.2 MLP Topology

The adopted multilayer neural network in the experiments consisted of the architecture with one layer of hidden neurons.

Such usual strategies in MLP [29] as one parameter keeping and the variation of remaining parameters defined the appropriate topology. Conventional experiments get adequate topology for classifying input digits (20×15) represented by the 300-dimensional feature vectors. An experimental MLP model consists of 300 neurons in the input layer. The adequate number of neurons in the hidden layer is set according to each experiment. The number of neurons in the output layer is defined by the target vector type or its size selected for each experiment.

5.3 Training Stage

The standard backpropagation algorithm [5] was adopted as the learning algorithm of each MLP model. Since all experimental target vectors are bipolar, the adopted activation function is the typical bipolar sigmoid [5], which has a range of $(-1, 1)$. Initial weights are generated as random values between -0.25 and 0.25. The learning rate parameter is set as 0.02. The criterion for stopping the learning algorithm is to require that the maximum value of the average squared error be equal to or less than the tolerance.

Training data set is constituted by 120 pattern-images not belonging to the testing data set. It contains input patterns for training the MLP model to classify digits extracted from license plates into categories. Each category is represented by 12 input patterns.

5.4 Testing Stage

The classification rate is calculated by Eq. (10):

$$cr = \frac{\sum_i^N c_i}{N}, \tag{10}$$

where cr is classification rate, N is number of testing patterns, and c_i is defined as

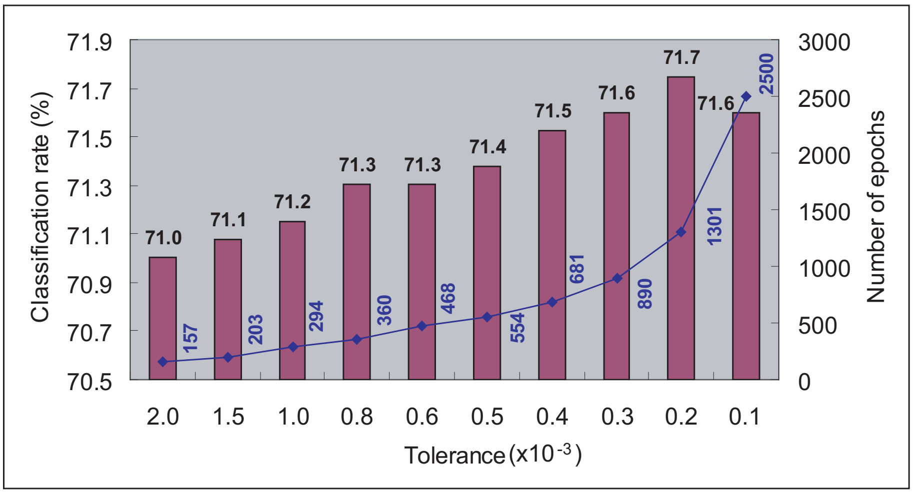


Figure 4: MLP performance using NOVs with 16 components as target vectors.

$$c_i = \begin{cases} 1 & : \quad p_i = r_i \\ 0 & : \quad p_i \neq r_i \end{cases} , \tag{11}$$

where p_i is a classified pattern (output of the MLP model), and r_i is the corresponding category (correct response).

Testing data set that contains 1352 images of extracted digits (“0”,“1”,. . . ,“9”) from license plate images after image preprocessing [26–28, 30]. A trained MLP model task consists of classifying extracted digits of the testing data set into ten classes.

6 Experiments and Results

The experimentation was proceeded using NOV and OBV as target vectors defined in Sections 3.1and 3.2. We evaluated the degraded pattern classification performance of MLP adopting the proposed OBVs as target vectors and the NOVs as other target ones for comparison of classification results. Figures 4–9 show the graphs with the number of epochs and the corresponding tolerance value for training convergence of the MLP model to provide the degraded pattern classification rate during the testing stage.

6.1 Influence of OBV on number of epochs and classification rate

To evaluate the influence of OBVs as target vectors on the number of epochs for training MLP models and their classification performance, the experiment consisted of the following strategy:

1. Construct NOVs of the same size as OBVs.
2. Adopt the same set of initial weights for training the MLP models defined by NOV and OBV as target vectors.
3. Compare the highest classification results of a pair of MLP models trained with different types of target vectors but with same sizes.

Figures 4–6 show testing stage results (pattern classification rate) using NOV with 16, 32, and 64 components, respectively. The similarity values calculated by Eq. (3) are 12, 28, and 60, respectively.

Figures 7–9 show testing stage results (pattern classification rate) using OBV with 16, 32, and 64 components, respectively. In this case, the similarity values are null because all the vectors are orthogonal.

Table 4 presents the highest classification rate and the corresponding number of epochs selected from each graph of Figs. 4–9. The values (classification rate and number of epochs) are listed in Table 4 for comparison purposes after adopting NOVs and OBVs as target vectors on the MLP model. Also, the last row of Table 4 shows the corresponding tolerance value to achieve the number of epochs during the training stage. We adopted the early stopping criterion by considering the balance between memorization and generalization. Based on this criterion is not necessarily advantageous to continue training until the error actually reaches a minimum [5].

6.2 Improvement on MLP performance due to OBV size variation

In this section, the influence of OBV size on MLP pattern classification performance improvement is presented. The experimental analysis consisted of adopting 64, 128, 256, or 448 components as OBV sizes [19]. Consequently, each experimental MLP topology consisted of 64, 128, 256, or 448 neurons in the output layer accordingly to the adopted OBV size.

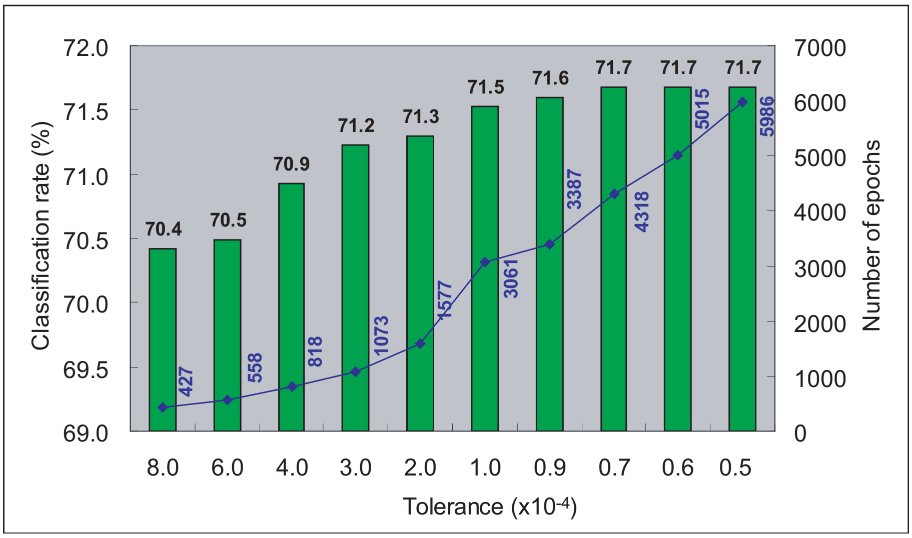


Figure 5: MLP performance using NOVs with 32 components as target vectors.

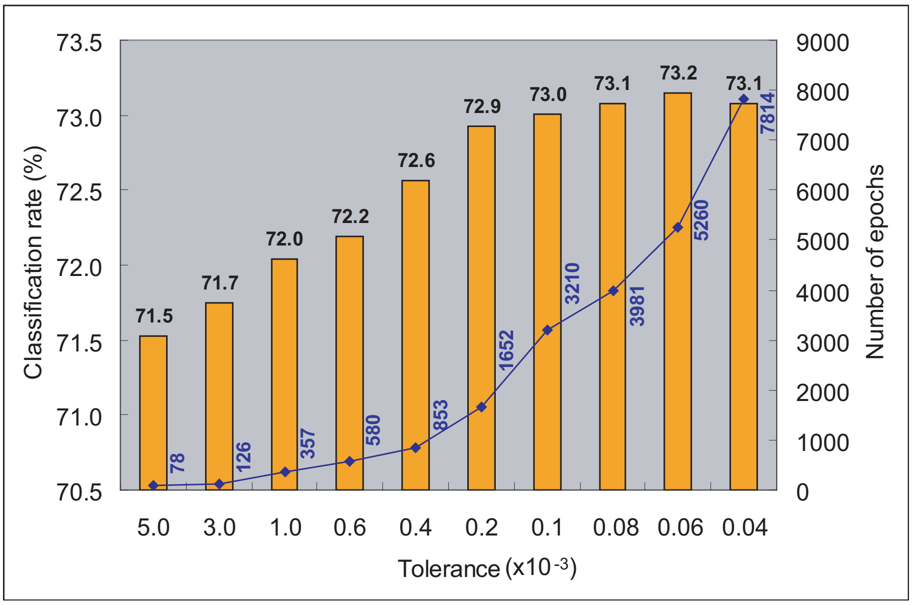


Figure 6: MLP performance using NOVs with 64 components as target vectors.

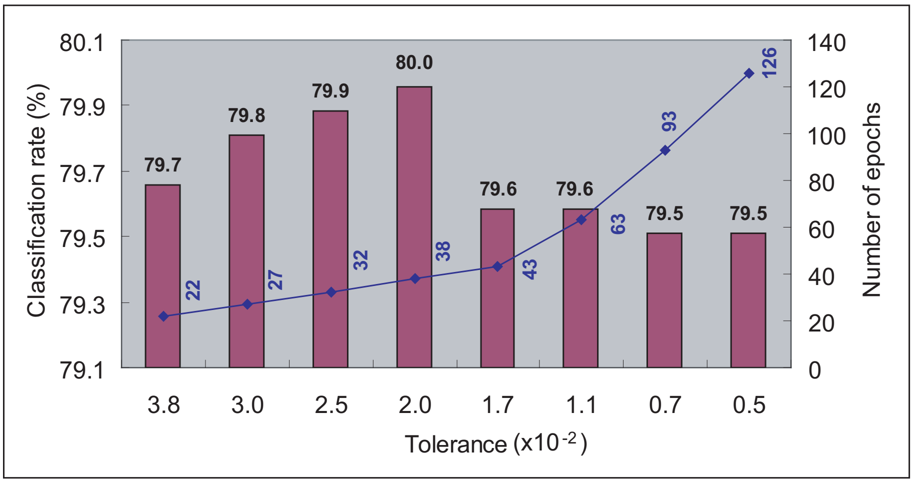


Figure 7: MLP performance using OBVs with 16 components as target vectors.

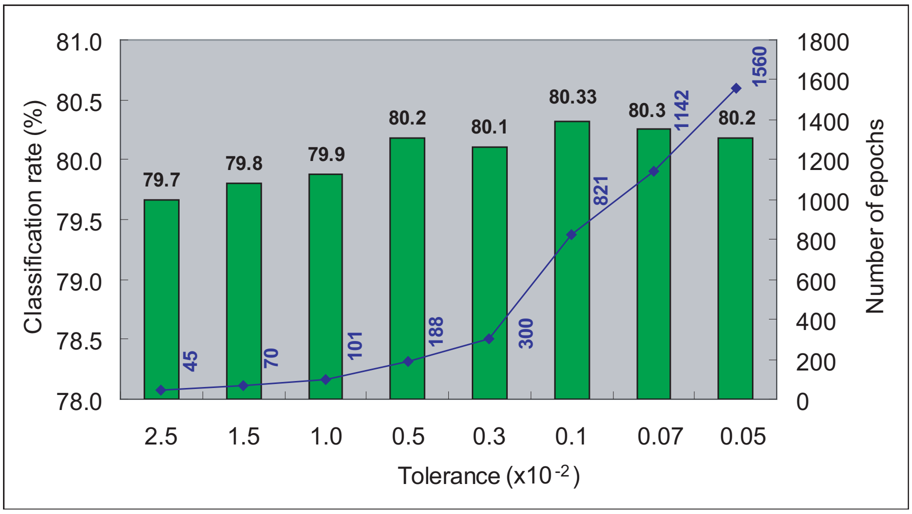


Figure 8: MLP performance using OBVs with 32 components as target vectors.

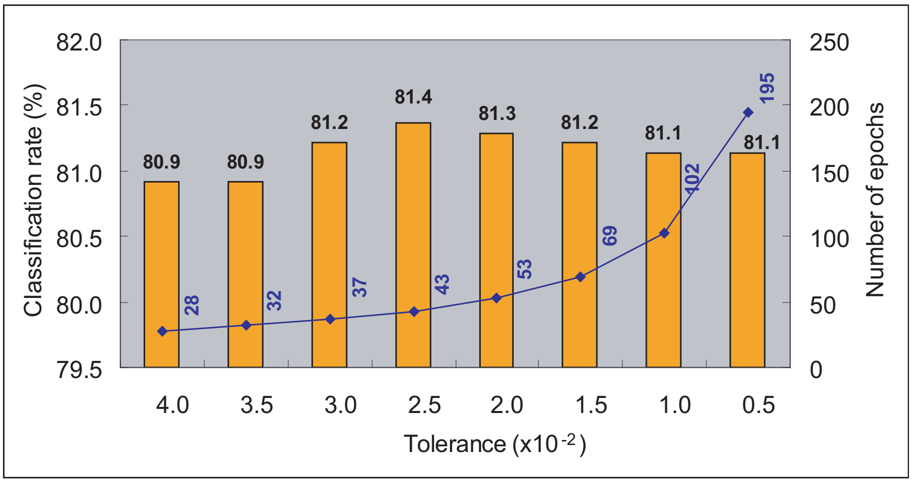


Figure 9: MLP performance using OBVs with 64 components as target vectors.

Table 4: Relevant results from the graphs of Figs. 4–9

	NOV16	*NOV32*	*NOV64*	*OBV16*	*OBV32*	*OBV64*
Classification %	71.70	71.70	73.20	$\boxed{80.00}$	$\boxed{80.33}$	$\boxed{81.40}$
Epochs	1301	4318	5260	$\boxed{38}$	$\boxed{821}$	$\boxed{43}$
Tolerance $\times 10^{-4}$	2	0.7	0.6	200	11	250

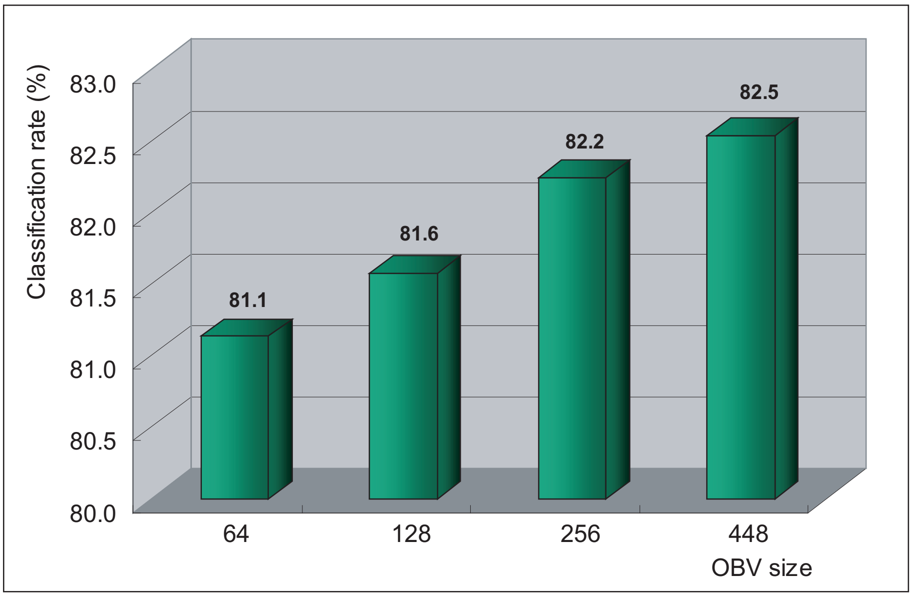


Figure 10: Performance evolution of MLP based on the OBV size increase.

The graph in Fig. 10 shows that the performance of MLP improves when trained with a larger OBV size following the increase of Euclidean distance for OBVs [19]. A significant 1.4% increase in the classification rate is verified from 81.1% (64 component-OBV) to 82.5% (448 component-OBV).

7 Discussion

Essentially, the same parameters such as initial weights, tolerance, and learning rate were adopted for training models with different types of target vectors to provide a fair comparison of results. The main comparison focuses on analyzing MLP performances by adopting NOVs and OBVs as target vectors. Since NOV and OBV can contain the same number of components, it was possible to use the same net topology for experimenting with both vectors to provide a fair comparison of MLP performances.

Relevant results from the graphs of Figs. 4–9 show that the classification rate results using MLP models trained with OBVs as target vectors are better than the results with NOVs for all different sizes. Also, Table 4 reveals that the corresponding number of epochs necessary to train each MLP model decreased when OBVs are adopted as target vectors. In the case of vectors with 64 components, OBVs increase the classification rate 8.2% from 73.2% to 81.4% over NOVs and significantly decrease the number of epochs from 5260 to 43 in Table 4.

8 Conclusion

The orthogonality of OBVs leads to the enlargement of output space created during the supervised learning (input-output mapping) of an MLP model. In other words, an MLP model based on OBVs as target vectors can be learned with higher ability to generalize by higher tolerance than a similar model based on NOVs as target vectors. Also, the model has a faster convergence speed to reach the training stop condition. Then, a suitable avoidance of overfitting or overtraining is promising too. Consequently, the topology design using OBVs provided a better MLP performance to classify degraded patterns.

9 Acknowledgment

This work is supported by the PROPP-UFU grant from Federal University of Uberlândia under Agreement No. 72/2010.

References

[1] W. H. Delashmit and M. T. Manry. Recent developments in multilayer perceptron neural networks. *Proceedings of the* 7th *Annual Memphis Area Engineering and Science Conference*, MAESC 2005.

[2] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and J. L. McClelland, eds., *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol. 1, pp. 318–362, MIT Press, Cambridge, MA, 1986.

[3] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[4] J. L. McClelland and D. E. Rumelhart. *Explorations in Parallel Distributed Processing: A Handbook of Models, Programs, and Exercises*. MIT Press, Cambridge, MA, 1988.

[5] L. V. Fausett. *Fundamentals of Neural Networks: Architectures, Algorithms, and Applications*. Prentice Hall, Englewood Cliffs, NJ, 1994.

[6] M. Collins and B. Roark. Incremental parsing with the perceptron algorithm. *Proceedings of the* 42^{nd} *Annual Meeting on Association for Computational Linguistics*, pages 111–118,2004.

[7] M. V. Ribeiro, J. G. A. Barbedo, J. M. T. Romano, and A. Lopes. Fourier-lapped multilayer perceptron method for speech quality assessment. *EURASIP Journal on Advances in Signal Processing*, 9:1425–1434, June 2005.

[8] M. T. Manry, H. Chandrasekaran, and C. Hsieh. Signal processing using the multilayer perceptron. In Y. H. Hu and J. Hwang, eds., *Handbook of Neural Network Signal Processing*, pp. 2.1–2.29, CRC Press, 2001.

[9] J. M. Nazzal, I. M. El-Emary, and S. A. Najim. Multilayer perceptron neural network (MLPs) for analyzing the properties of Jordan Oil Shale. *World Applied Sciences Journal*, 5(5):546–552, 2008.

[10] W. H. Delashmit and M. T. Manry. Enhanced robustness of multilayer perceptron training. *Thirty-Sixth Annual Asilomar Conference on Signals, Systems and Computers*, pages 1029–1033, Pacific Grove, CA, 2002.

[11] W. H. Delashmit. Multilayer perceptron structured initialization and separating mean processing. *Ph.D. Dissertation*, University of Texas at Arlington, May 2003.

[12] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, New York, 1973.

[13] K. S. Fu. *Syntactic Methods in Pattern Recognition*. Academic Press, New York, 1974.

[14] L. Kanal. Patterns in pattern recognition: 1968–1974. *IEEE Transactions on Information Theory*, 20(6):697–722, Nov. 1974.

[15] J. R. Ullman. *Pattern Recognition Techniques*. Butterworth, London, 1973.

[16] H. Andrews. *Introduction to Mathematical Techniques in Pattern Recognition*. John Wiley & Sons, New York, 1972.

[17] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, New York, 2001.

[18] A. Browne. *Neural Network Analysis, Architectures, and Applications*. Institute of Physics Pub., Bristol, 1997.

[19] S. Nomura, K. Yamanaka, O. Katai, H. Kawakami, and T. Shiose. Improved MLP learning via orthogonal bipolar target vectors. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 9(6):580–589, 2005.

[20] T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

[21] Y. LeCun, B. Boser, J. S. Denker, B. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.

[22] N. J. Nilsson. *Learning Machines*. McGraw-Hill, New York, 1965.

[23] T. J. Sejnowski and C. R. Rosenberg. Parallel networks that learn to pronounce english text. *Complex Systems*, 1:145–168, 1987.

[24] T. J. Sejnowski and C. R. Rosenberg. NETtalk: A parallel network that learns to read aloud. John Hopkins University Department of Electrical Engineering and Computer Science. *Technical Report*, 86/01, 1986.

[25] B. Noble and J. W. Daniel. *Applied Linear Algebra*. Prentice Hall, Englewood Cliffs, New Jersey, second edition, 1977.

[26] S. Nomura, K. Yamanaka, O. Katai, and H. Kawakami. A new method for degraded color image binarization based on adaptive lightning on grayscale versions. *IEICE Trans. on Information and Systems*, E87-D(4):1012–1020, 2004.

[27] S. Nomura and K. Yamanaka. New adaptive methods applied to binarization of printed word images. In N. Younan, editor, *Proceedings of the Fourth IASTED International Conference Signal and Image Processing*, pages 288–293, Kauai, USA, 2002.

[28] S. Nomura, K. Yamanaka, O. Katai, H. Kawakami, and T. Shiose. A novel adaptive morphological approach for segmenting characters in degraded images. *Pattern Recognition*, 38:1961–1975, Nov. 2005.

[29] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, New Jersey, second edition, 1999.

[30] S. Nomura and K. Yamanaka. New adaptive approach based on mathematical morphology applied to character segmentation and code extraction from number plate images. In *Proc. of* 6^{th} *World Multi-Conference on Systemics, Cybernetics and Informatics*, volume IX, Florida, USA, Jul. 2002.