POLLUTANTS TIME-SERIES PREDICTION USING THE GAMMA CLASSIFIER

Itzamá LÓPEZ-YÁÑEZ[*], **Amadeo J. ARGÜELLES-CRUZ,**
Oscar CAMACHO-NIETO, and Cornelio YÁÑEZ-MÁRQUEZ
Alpha-Beta Group, Computational Intelligence Laboratory
Av. Juan de Dios Bátiz s/n Edificio CIC
México, D.F., 07738, México
E-mail: ilopezy@ipn.mx
www.alfabeta.org.mx

Received 20 September 2010
Accepted 2 May 2011

Abstract

In this work we predict time series of air pollution data taken in Mexico City and the Valley of Mexico, by using the Gamma Classifier which is a novel intelligent associative mathematical model, coupled with an emergent coding technique. Historical and current data about the concentration of specific pollutants, in the form of time series, were used. The pollutants of interest are: carbon monoxide (CO), ozone (O_3), sulfur dioxide (SO_2), and nitrogen oxides (NO_x, including both nitrogen monoxide, NO, and nitrogen dioxide, NO_2).

Keywords: Gamma classifier, Time series prediction, Environmental data prediction, Pattern classifier, Associative models.

1. Introduction

Air pollution is a current problem that has been strongly associated to technological advances. Time Series analysis[1–3] is a tool which allows the prediction of particularly bad conditions events, thus enabling better decision making.[4–6]

Atmospheric pollution is of great interest to many Latin American urban centers, since it has become a major problem for megacities such as Mexico City, São Paulo, Santiago, and Buenos Aires.[7] Particularly, the air quality of Mexico City is monitored by sampling and measuring the concentration of several pollutants of interest, through out the Metropolitan Zone of the Valley of Mexico. This is done automatically by several monitoring stations, as well as manually in several fixed and mobile stations, which belong to the Mexico City Atmospheric Monitoring System (SIMAT in Spanish). This data is measured and recorded periodically, as well as processed and published by SIMAT, which is also responsible for keeping the authorities and population informed by means of the Air Quality Metropolitan Index.[8] It is thanks to this periodicity that this data can be analyzed as a time series.

Several methods for analysis and forecasting of air quality data have been proposed around the globe. Some of the most successful among these are: multivariate linear regression (MLR),[9] Bayesian networks and decision trees,[10] artificial neural networks (ANN),[10–13] support vector machines (SVM),[14] and pollution models such as Caline and IITLS.[15] In this paper, air pollution data from Mexico City was used, employing the Gamma classifier (previously introduced as a general, empirical classifier algorithm in Ref. 16, and preliminarily applied to related problems in Refs. 17 and 18).

[*] Corresponding author.

Published by Atlantis Press

The problem studied in the present work is the prediction of time series of air pollution data taken in Mexico City and the Valley of Mexico, through the application of the Gamma classifier, obtaining competitive experimental results. For this, historical and current data about the concentration of specific pollutants, in the form of time series, were used. These data were measured hourly by several monitoring stations. Some pollutants are of particular interest, given their consideration by Mexican (and international) governing bodies as criteria pollutants: carbon monoxide (CO), ozone (O_3), sulfur dioxide (SO_2), and nitrogen oxides (NO_x, including both nitrogen monoxide, NO, and nitrogen dioxide, NO_2).

The rest of the paper is organized as follows. Sec. 2 is dedicated to presenting the Mexico City Atmospheric Monitoring System, thus giving some background on the kind of data used and the purpose for which it is intended, as well as how such data is represented as time series and the corresponding analysis done on them, enabling their prediction. Sec. 3 includes the discussion of the Gamma classifier version 1, while Sec. 4 is dedicated to the experimental design, with Sec. 5 presenting the experiments of the first generation (1G), which uses the Gamma classifier version 1. Based on the analysis of these experiments, the method was improved, giving rise to the Gamma classifier version 2, which is introduced in Sec. 6. The next Sec., the seventh, presents the second generation of experiments (2G), which use the novel Gamma classifier version 2. Later, an analysis and comparison of the methods applied to this problem is presented in Sec. 8, with conclusions and lines of future work drawn in Sec. 9. Finally, the references are included.

2. Time Series of Air Pollution Data in Mexico City

The Mexico City Atmospheric Monitoring System (*Sistema de Monitoreo Atmosférico de la Ciudad de México*, SIMAT in Spanish) is the government body responsible for the surveillance and monitoring of the atmospheric quality in the Metropolitan Zone of the Valley of Mexico. The information presented in this Sec. regarding SIMAT is strongly based on Ref. 8.

SIMAT is committed to operating and maintaining a trustworthy system for the monitoring of air quality in Mexico City, as well as analyzing and publishing this information in order to fulfil current requirements and legislation. The objective of SIMAT is to watch and evaluate the air quality in Mexico City, as a pre-emptive measure for health protection of its inhabitants, in order to promptly inform the populace as well as enable decision making in prevention and air quality improvement programs. Given the risks posed to the population, this is considered a relevant problem by authorities and citizens alike.[19] Thus, SIMAT is made up by four specialized subsystems, one Atmospheric Monitoring Mobile Unit, and a Calibration Standards Transfer Lab.

The subsystem of interest for this work is the Automatic Atmospheric Monitoring Network (RAMA, *Red Automática de Monitoreo Atmosférico* in Spanish), which takes continuous and permanent measurements of several pollutants: O_3, SO_2, NO_x, CO, particulate matter less than 10 microns in diameter (PM_{10}), and particulate matter less than 2.5 microns in diameter ($PM_{2.5}$); each measurement is taken automatically every hour.

The Air Quality Metropolitan Index (*Índice Metropolitano de la Calidad del Aire*, IMECA in Spanish) is a reference value for people to be aware of the pollution levels prevalent in any zone, in a precise and timely manner, in order to take appropriate protection measures. When the IMECA of any pollutant is greater than 100 points, its concentration is dangerous for health and, as the value of IMECA grows, the symptoms worsen, as can be seen in table 1. For simplification purposes (in particular regarding communicating air quality status to the population at large), only one IMECA value is published: that of the pollutant presenting the highest IMECA value. Ref. 20 states the specifications for computing the IMECA for the criteria pollutants: O_3, NO_2, SO_2, and CO.

Given that air quality data recorded by the RAMA subsystem of SIMAT is measured periodically (i.e. one sample each hour), this data can de represented as a time series. It is clear, however, that the phenomena represented by these time series are complex.[2]

With respect to trend, each pollutant of interest shows a small decrease over the years; an example on ozone can be seen in Fig. 1.[8] While this long-term trend is relevant for the air quality itself, it is of no consequence to the proposed method, given that it does not take into account such long-term effects.

On the other hand, the four pollutants exhibit seasonality over 24 hours, presenting higher concentrations at similar hours of each day. In order to

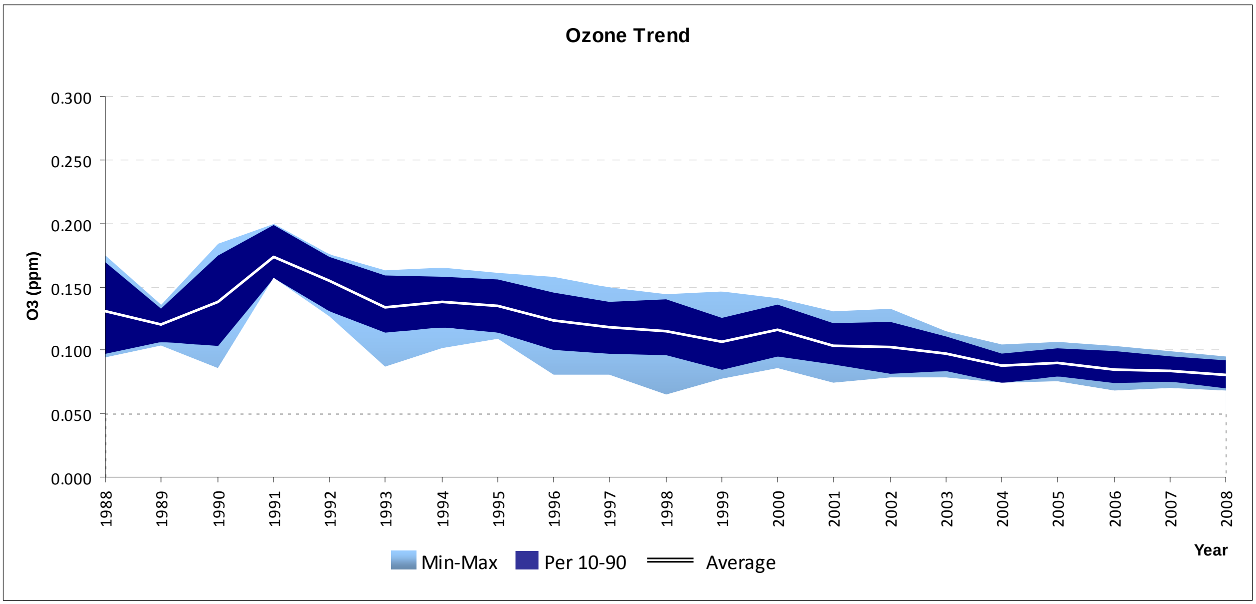

Fig. 1. Ozone long-term trend in Mexico City.

avoid the potential problems imposed by nonstationarity, the differences between consecutive samples were used.[1–3]

Table 1. IMECA and its implications for health.

IMECA	Condition	Effects on Health
0–50: green	Good	Suitable for conducting outdoor activities
51–100: yellow	Regular	Possible discomfort in children, the elderly and people with illnesses
101–150: orange	Bad	Cause of adverse health effects on the population, particularly on children and older adults with cardiovascular and / or respiratory illnesses such as asthma
151–200: red	Very Bad	Cause of greater adverse health effects on the population, particularly on children and older adults with cardiovascular and / or respiratory illnesses such as asthma
>200: purple	Extremely Bad	Cause of adverse health effects in the general population. Serious complications may present in children and older adults with cardiovascular and / or respiratory illnesses such as asthma

The linearity or lack of it by the time series was not explicitly addressed, since the method used does not distinguish either way.

3. Gamma Classifier version 1

This pattern classifier, of recent proposal, has shown some very promising results.[16–18] The algorithm in its initial form was introduced in Ref. 16, as an empirical, general classifier algorithm. Later, it was applied to air quality data prediction, obtaining preliminary results which were presented at an international conference (CIARP 2008).[17] The experiments done in the latter work included only one pollutant (SO_2), whose performance was compared to that of other methods, on other pollutants (mainly O_3). Derived from that work, a book chapter was published,[18] including new experiments on other pollutants. The analysis of all those experimental results enabled the improvement of the method proposed for applying the Gamma classifier to air quality data prediction.

This Sec. is dedicated to discussing the initial Gamma classifier (version 1), as introduced in Ref. 16. The method introduced in Ref. 17 to apply this classifier to predict future concentrations of air pollutants is explained in greater detail in subsections 5.1 and 6.1, while the results obtained with said method (during the 1G of experiments) and their comparison against other methods are discussed in Sec. 6.

On the other hand, the original improvements and modifications to both the classifier and the method of applying it to the current problem (used on the 2G of experiments), which give rise to the Gamma classifier version 2 (presented in this paper) are introduced in Sec. 7 and its performance further discussed in Sec. 8.

The basis of the Gamma classifier in its initial version is the gamma operator, hence its name. In turn, the gamma operator is based on the alpha (α), beta (β), and u_β operators and their properties, in particular when dealing with binary patterns coded with the modified Johnson-Möbius code. Thus, the focus of subsection 3.1 will be on discussing these components of the Gamma classifier: the alpha and beta operators, the u_β operator, the modified Johnson-Möbius code, and the gamma operator; while the algorithm of the classifier (version 1) is discussed in subsection 3.2.

Notice that here we are not working with the classical Johnson-Möbius code, which was previously introduced,[21] but rather with a modification recently proposed by the authors research group and further explained below.

It is worthy of mention that the Gamma classifier was initially designed to work with patterns, represented by vectors composed of real numbers. The RAMA database, however, is not made up by vectors: it contains a series of numeric data points representing the concentration values of a given pollutant, for each station during each hour of a particular year. Then, a relevant contribution of this work is the development of a coding technique which converts those series of values into a set of patterns, which can then be used by the Gamma classifier to forecast the next value. The rest of this Sec. is strongly based on Refs. 16–18.

3.1. *Preliminaries*

The alpha and beta operators were introduced in Refs. 22 and 23 and are the basis of the Alpha-Beta associative memories. The operators are defined in a tabular manner, as shown in table 2, considering that the sets A and B are defined as A = {0, 1} and B = {0, 1, 2}.

Table 2. Definition of the alpha and beta operators.

$\alpha : A\times A\rightarrow B$		
x	y	$\alpha(x, y)$
0	0	1
0	1	0
1	0	2
1	1	1

$\beta : B\times A\rightarrow A$		
x	y	$\beta(x, y)$
0	0	0
0	1	0
1	0	0
1	1	1
2	0	1
2	1	1

These operators have set the foundation for the development of several mathematical models employed in Pattern Recognition, from the original Alpha-Beta associative memories, to the Gamma classifier, passing trough models such as the Alpha-Beta bidirectional associative memories[23] or the Alpha-Beta associative support vector machines.[24] As current research indicates, much of the strength, robustness, and efficacy exhibited by the aforementioned associative models are due to the alpha and beta operators. This particular fact is quite a surprise, since the operators seem to be almost trivially simple. The results obtained with the Alpha-Beta associative models indicate exactly the contrary.

The unary u_β operator receives as input an n-dimensional binary vector **x**, outputs a non-negative integer number, and is calculated as shown below:

$$u_\beta = \sum_{i=1}^{n} \beta(x_i, x_i) \tag{1}$$

Thus, if **x** = [1 0 1 0 1 0] then:

$$\begin{aligned} u_\beta(\mathbf{x}) &= \beta(1,1) + \beta(0,0) + \beta(1,1) + \\ &+ \beta(0,0) + \beta(1,1) + \beta(0,0) \\ &= 1+0+1+0+1+0 = 3 \end{aligned} \tag{2}$$

On the other hand, the modified Johnson-Möbius code —proposed by the authors research group, which is a variation on the classical Johnson-Möbius code— allows us to convert a set of real numbers into binary representations by following these steps:

(i) Subtract the minimum (of the set of numbers) from each number, leaving only non-negative real numbers.
(ii) Scale up the numbers (truncating the remaining decimals if necessary) by multiplying all numbers by an appropriate power of 10, in order to leave only non-negative integer numbers.
(iii) Concatenate $e_m - e_j$ zeros with e_j ones, where e_m is the greatest non-negative integer number to be coded, and e_j is the current non-negative integer number to be coded.

For instance, let $D \subset \mathbb{R}$ be defined as $D = \{1.7, 1.9, 0.2, 0.6, -0.1\}$; now, lets use the modified Johnson-Möbius code (as explained above) to convert the elements of D into binary vectors.

(i) Subtract the minimum:

$$\begin{aligned} &D \longrightarrow T \subset \mathbb{R}, \\ &T = \{1.8, 2.0, 0.3, 0.7, 0.0\} \end{aligned} \tag{3}$$

Since –0.1 is the minimum number in the set, the members of the transformed set T are obtained by subtracting –0.1 to each member of D (which is the same as adding 0.1): $t_i = d_i - (-0.1) = d_i + 0.1$. Notice that this step is particularly useful for handling negative numbers, since now there will be only non-negative numbers, with the minimum being 0.

(ii) Scale up the numbers:

$$T \longrightarrow E \subset \mathbb{Z}^+, \quad E = \{18, 20, 3, 7, 0\} \tag{4}$$

Since there is only one decimal digit, it is enough to multiply each number by 10 to obtain integers, thus the members of the integers set E are calculated as: $e_i = t_i \times 10$.

(iii) Concatenate $e_m - e_j$ zeros with e_j ones, where e_m is the maximum non-negative integer number to be coded, and e_j is the current non-negative integer number to be coded:

$$E \longrightarrow C = \left\{ c_i \middle| c_i \in A^{20} \right\}, \quad C = \begin{Bmatrix} [00111111111111111111] \\ [11111111111111111111] \\ [00000000000000000111] \\ [00000000000001111111] \\ [00000000000000000000] \end{Bmatrix} \tag{5}$$

Given that the maximum number in the set E of non-negative integers is 20, all binary vectors have 20 components, i.e. they all are 20 bits long. In other words, each member of the C set is a binary vector which belongs to the set resulting from applying 20 times the cross product of the set A = {0, 1} to itself.

For example, $e_3 = 18$ is converted into its binary representation c_3 by appending $e_m - e_3 = 20 - 18 = 2$ zeroes "00", followed by $e_3 = 18$ ones "111111111111111111", which gives the final vector: "00111111111111111111".

This is because the maximum number in set E is 20, thus $e_m = 20$; and the number to be converted is $e_3 = 18$.

Finally, the generalized gamma operator γ_g, which takes as input two binary patterns $\mathbf{x} \in A^n$ and $\mathbf{y} \in A^m$ —with $n, m \in \mathbb{Z}^+, n \le m$— and a non-negative integer number θ; and gives a binary number as output; can be computed as follows:

$$\gamma_g(\mathbf{x}, \mathbf{y}, \theta) = \begin{cases} 1 & \text{if } m - u_\beta\left[\alpha(\mathbf{x}, \mathbf{y}) \bmod 2\right] \le \theta \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

where mod2 indicates the usual modulo 2 operator.

In order to better illustrate how the generalized gamma operator works, let us work through some examples of its application.

Then, if $\mathbf{x}$ = [1 1 0 0 0], $\mathbf{y}$ = [1 0 1 0 0], and θ = 2, what is the result of $\gamma_g(\mathbf{x}, \mathbf{y}, \theta)$?

First, we have that $\alpha(\mathbf{x}, \mathbf{y})$ = [1 2 0 1 1]; then [1 2 0 1 1] mod2 = [1 0 0 1 1]; now

$$\begin{aligned} u_\beta[10011] &= \sum \begin{bmatrix} \beta(1,1), \beta(0,0), \beta(0,0), \\ \beta(1,1), \beta(1,1) \end{bmatrix} \\ &= 1 + 0 + 0 + 1 + 1 = 3 \end{aligned} \tag{7}$$

and since m = 5, 5 – 3 = 2; given that $2 \le \theta = 2$, the result is 1.

Meanwhile, if θ = 3, the result would be 1. However, if θ = 1, it is clear that the result would be 0, since 2 > 1, thus it is false that $2 \le \theta$.

On the other hand, if $\mathbf{x}$ = [1 0 0], $\mathbf{y}$ = [1 0 0], and θ = 2, we have:

$$\alpha(\mathbf{x}, \mathbf{y}) = [111] \tag{8}$$

$$[111] \bmod 2 = [111] \tag{9}$$

$$\begin{aligned} u_\beta[111] &= \sum\left[\beta(1,1), \beta(1,1), \beta(1,1)\right] \\ &= 1 + 1 + 1 = 3 \end{aligned} \tag{10}$$

$$\begin{aligned} m - u_\beta[111] &= 3 - 3 = 0 \\ 0 \le \theta = 2 &\therefore \gamma_g(\mathbf{x}, \mathbf{y}, \theta) = 1 \end{aligned} \tag{11}$$

So far, all examples have been for cases in which $m = n$, but what happens if $n < m$? For instance, $\mathbf{x}$ = [1 0 0], $\mathbf{y}$ = [1 0 0 1], and θ = 0.

First, the $m - n$ most significant bits of $\mathbf{y}$ are truncated, leaving $\mathbf{x}$ = [1 0 0], and $\mathbf{y}$ = [0 0 1], which can be operated through α in the usual manner:

$$\alpha(\mathbf{x}, \mathbf{y}) = [210] \tag{12}$$

$$[210] \bmod 2 = [010] \tag{13}$$

$$u_\beta[010] = \sum[\beta(0,0), \beta(1,1), \beta(0,0)] \\ = 0+1+0 = 1 \tag{14}$$

But, since it is m which is considered in the difference and not n, the extra bits which were truncated from **y** are taken into account here:

$$m - u_\beta[010] = 4 - 1 = 3 \\ 3 > \theta = 0 \rightarrow \neg(m - u_\beta[010] \leq \theta) \\ \therefore \gamma_g(\mathbf{x}, \mathbf{y}, \theta) = 0 \tag{15}$$

The main idea of the generalized gamma operator is to indicate (result equal to 1) that two binary vectors are similar, allowing up to θ bits to be different and still consider those vectors similar. If more than θ bits are different, the vectors are said to be not similar (result equal to 0). Thus, if $\theta = 0$ both vectors must be equal for γ_g to output a 1.

3.2. *The Gamma classifier algorithm version 1*

Let $k, m, n, p \in \mathbb{Z}^+$; $\{\mathbf{x}^\mu | \mu = 1, 2, \ldots, p\}$ be the learning (fundamental) pattern set with cardinality p, where $\forall \mu\ \mathbf{x}^\mu \in \mathbb{R}^n$, and let $\mathbf{y} \in \mathbb{R}^n$ be an n-dimensional real-valued pattern to be classified. It is assumed that the fundamental set is partitioned into m different, mutually exclusive classes, each class having a cardinality k_i, $i = 1, 2, \ldots, m$, and thus $\sum k_i = p$. In order to classify **y**, these steps are followed:

(i) Code the fundamental set with the modified Johnson-Möbius code, obtaining a value e_m for each component. This e_m value is calculated as defined in the following equation:

$$e_m(j) = \bigvee_{i=1}^{p} x_j^i \tag{16}$$

where $\vee$ represents the usual maximum operator. That is, $e_m(j)$ is the maximum value across all fundamental patterns, considering only their j-th component.

(ii) Compute the stop parameter, as expressed below:

$$\rho = \bigwedge_{j=1}^{n} e_m(j) \tag{17}$$

where $\wedge$ represents the usual minimum operator.

(iii) Code **y** with the modified Johnson-Möbius code, using the same parameters used with the fundamental set (step i). If any y_j is greater than the corresponding $e_m(j)$, the γ_g operator will use such y_j instead of m (according to Eq. 6).

(iv) Transform the index of all fundamental patterns into two indices, one for the class they belong to, and another for their position in the class (i.e. $\mathbf{x}^\mu$ which belongs to class i becomes $\mathbf{x}^{i\omega}$).

(v) Initialize θ to 0.

(vi) Do $\gamma_g(\mathbf{x}_j^{i\omega}, \mathbf{y}_j, \theta)$ for each component of the fundamental patterns in each class, following Eq. 6.

(vii) Compute a weighted sum c_i for each class, according to this equation:

$$c_i = \frac{\sum_{\omega=1}^{k_i} \sum_{j=1}^{n} \gamma_g(\mathbf{x}_j^{i\omega}, \mathbf{y}_j, \theta)}{k_i} \tag{18}$$

(viii) If there is more than one maximum among the different c_i, increment θ by 1 and repeat steps vi and vii until there is a unique maximum, or the stop condition $\theta \geq \rho$ is fulfilled.

(ix) If there is a unique maximum, assign **y** to the class corresponding to such maximum:

$$C_y = C_j \text{ such that } \bigvee_{i=1}^{m} c_i = c_j \tag{19}$$

(x) Otherwise, assign **y** to the class of the first maximum.

The first five steps in the former algorithm can be seen as the learning phase of the classifier, while steps vi to x (including any iteration if present) can be considered the recalling or classification phase.

Thus, the learning phase consists of coding all patterns (from both the fundamental and test sets) with the modified Johnson-Möbius code, using the same parameters. Also, the index of the fundamental patterns is converted into two indices, in order to identify the class to which each fundamental pattern belongs with the first of these indices. Finally, the stop parameter ρ is computed and θ is initialized to 0.

Meanwhile, the classification phase consists of computing $\gamma_g(\mathbf{x}_j^{i\omega}, \mathbf{y}_j, \theta)$ between each fundamental pattern and the test pattern, for each component. Then, a weighted addition is performed in order to sum all results corresponding to the same class. The weight part corresponds to the division of the sum by the cardinality of the class, thus normalizing unbalanced classes.

If there is a unique maximum among the weighted additions for all classes, that class is assigned to the test pattern. In case of non-unique maximums, θ is incremented and the algorithm is repeated from step vi, as long as θ is less than the stop parameter ρ. If the stop condition $\theta \geq \rho$ is true, the class assigned to the test pattern is any of those corresponding to the non-unique maximums (e.g. the first class which has a maximum weighted addition).

As an example, let us consider the following fundamental patterns:

$$\mathbf{x}^1 = \begin{pmatrix} 2 \\ 8 \end{pmatrix}, \mathbf{x}^2 = \begin{pmatrix} 1 \\ 9 \end{pmatrix}, \mathbf{x}^3 = \begin{pmatrix} 6 \\ 3 \end{pmatrix}, \mathbf{x}^4 = \begin{pmatrix} 7 \\ 4 \end{pmatrix}$$

grouped in two classes: $C_1 = \{\mathbf{x}^1, \mathbf{x}^2\}$ and $C_2 = \{\mathbf{x}^3, \mathbf{x}^4\}$. Then, the patterns to be classified are:

$$\mathbf{y}^1 = \begin{pmatrix} 6 \\ 2 \end{pmatrix}, \mathbf{y}^2 = \begin{pmatrix} 3 \\ 9 \end{pmatrix}$$

As can be seen, the dimensions of all patterns is $n = 2$, and there are 2 classes, both with the same cardinality $k_1 = 2$ and $k_2 = 2$. Now the steps in the algorithm are followed.

(i) Code the fundamental set with the modified Johnson-Möbius code, obtaining a value e_m for each component:

$$\mathbf{x}^1 = \begin{pmatrix} 0000011 \\ 011111111 \end{pmatrix}, \mathbf{x}^2 = \begin{pmatrix} 0000001 \\ 111111111 \end{pmatrix}, \\ \mathbf{x}^3 = \begin{pmatrix} 0111111 \\ 000000111 \end{pmatrix}, \mathbf{x}^4 = \begin{pmatrix} 1111111 \\ 000001111 \end{pmatrix} \qquad (20)$$

Since the maximum value among the first components of all fundamental patterns is 7, then $e_m(1) = 7$, and the maximum value among the second components is $e_m(2) = 9$. Thus, the binary vectors representing the first components are 7 bits long, while those representing the second component are 9 bits long.

(ii) Compute the stop parameter according to Eq. 17:

$$\rho = \bigwedge_{j=1}^{n} e_m(j) = \bigwedge_{j=1}^{2} \left[e_m(1), e_m(2) \right] \\ = \bigwedge(7,9) = 7 \qquad (21)$$

(iii) Code $\mathbf{y}$ with the modified Johnson-Möbius code, using the same parameters used with the fundamental set (step i).

$$\mathbf{y}^1 = \begin{pmatrix} 0111111 \\ 000000011 \end{pmatrix}, \mathbf{y}^2 = \begin{pmatrix} 0000111 \\ 111111111 \end{pmatrix} \qquad (22)$$

Again, the binary vectors representing the first components are 7 bits long, while those representing the second component are 9 bits long.

(iv) Transform the index of all fundamental patterns into two indices, one for the class they belong to, and another for their position in the class.

$$\mathbf{x}^1 = \mathbf{x}^{11}, \mathbf{x}^2 = \mathbf{x}^{12}, \\ \mathbf{x}^3 = \mathbf{x}^{21}, \mathbf{x}^4 = \mathbf{x}^{22} \qquad (23)$$

Given that $C_1 = \{\mathbf{x}^1, \mathbf{x}^2\}$ and $C_2 = \{\mathbf{x}^3, \mathbf{x}^4\}$, we know that both $\mathbf{x}^1$ and $\mathbf{x}^2$ belong to class C_1, so the first index for both patterns becomes 1; the second index is used to differentiate between patterns assigned to the same class, thus $\mathbf{x}^1$ becomes $\mathbf{x}^{11}$, and $\mathbf{x}^2$ is now $\mathbf{x}^{12}$. Something similar happens to $\mathbf{x}^3$ and $\mathbf{x}^4$, but with the first index being 2 since they belong to class C_2: $\mathbf{x}^3$ becomes $\mathbf{x}^{21}$, and $\mathbf{x}^4$ is now $\mathbf{x}^{22}$.

(v) Initialize θ to 0.

$$\theta = 0 \qquad (24)$$

(vi) Do $\gamma_g\left(\mathbf{x}_j^{i\omega}, \mathbf{y}_j, \theta\right)$ for each component of the fundamental patterns in each class, following Eq. 6. Thus, for $\mathbf{y}^1$ we have:

$$\gamma_g\left(\mathbf{x}_1^{11}, \mathbf{y}_1^1, 0\right) = 0, \quad \gamma_g\left(\mathbf{x}_2^{11}, \mathbf{y}_2^1, 0\right) = 0 \\ \gamma_g\left(\mathbf{x}_1^{12}, \mathbf{y}_1^1, 0\right) = 0, \quad \gamma_g\left(\mathbf{x}_2^{12}, \mathbf{y}_2^1, 0\right) = 0 \\ \gamma_g\left(\mathbf{x}_1^{21}, \mathbf{y}_1^1, 0\right) = 1, \quad \gamma_g\left(\mathbf{x}_2^{21}, \mathbf{y}_2^1, 0\right) = 0 \\ \gamma_g\left(\mathbf{x}_1^{22}, \mathbf{y}_1^1, 0\right) = 0, \quad \gamma_g\left(\mathbf{x}_2^{22}, \mathbf{y}_2^1, 0\right) = 0 \qquad (25)$$

Since $\theta = 0$, for the γ_g to give a result of 1 it is necessary that both vectors $\mathbf{x}_j^{i\omega}$ and $\mathbf{y}_j^1$ are equal. But only one fundamental pattern has its first component equal to that of $\mathbf{y}^1$: $\mathbf{x}_1^{21} = [0111111] = \mathbf{y}_1^1$, while no fundamental pattern has a second component equal to that of the test pattern. Given that this is the only case for which a component of a fundamental pattern is equal to the same component of the test pattern, it is also the only instance in which γ_g outputs 1.

Now, for $\mathbf{y}^2$ we have:

$$\gamma_g\left(\mathbf{x}_1^{11}, \mathbf{y}_1^2, 0\right) = 0, \quad \gamma_g\left(\mathbf{x}_2^{11}, \mathbf{y}_2^2, 0\right) = 0 \\ \gamma_g\left(\mathbf{x}_1^{12}, \mathbf{y}_1^2, 0\right) = 0, \quad \gamma_g\left(\mathbf{x}_2^{12}, \mathbf{y}_2^2, 0\right) = 1 \\ \gamma_g\left(\mathbf{x}_1^{21}, \mathbf{y}_1^2, 0\right) = 0, \quad \gamma_g\left(\mathbf{x}_2^{21}, \mathbf{y}_2^2, 0\right) = 0 \\ \gamma_g\left(\mathbf{x}_1^{22}, \mathbf{y}_1^2, 0\right) = 0, \quad \gamma_g\left(\mathbf{x}_2^{22}, \mathbf{y}_2^2, 0\right) = 0 \qquad (26)$$

Again $\theta = 0$, thus forcing both vectors $\mathbf{x}_j^{i\omega}$ and $\mathbf{y}_j^2$ to be equal in order to obtain a 1. Similarly to what happened in the case of $\mathbf{y}^1$, there is but one instance of such occurrence: $\mathbf{x}_2^{12} = [111111111] = \mathbf{y}_2^2$.

(vii) Compute a weighted sum c_i for each class, according to Eq. 18. So, for $\mathbf{y}^1$:

$$c_1 = \frac{\sum_{\omega=1}^{k_1}\sum_{j=1}^{n}\gamma_g\left(\mathbf{x}_j^{1\omega},\mathbf{y}_j^1,\theta\right)}{k_1} = \frac{\sum_{\omega=1}^{2}\sum_{j=1}^{2}\gamma_g\left(\mathbf{x}_j^{1\omega},\mathbf{y}_j^1,0\right)}{2} = \frac{\sum_{\omega=1}^{2}\left[(0+0),(0+0)\right]}{2} = \frac{(0+0)}{2} = \frac{0}{2} = 0 \quad (27)$$

Here we add together the results obtained on all components of all fundamental patterns belonging to class C_1. Since all gave 0 (i.e. none of these were similar to the corresponding component of the test pattern given the value of θ in effect), the result of the weighted addition for class C_1 is $c_1 = 0$.

$$c_2 = \frac{\sum_{\omega=1}^{k_2}\sum_{j=1}^{n}\gamma_g\left(\mathbf{x}_j^{2\omega},\mathbf{y}_j^1,\theta\right)}{k_2} = \frac{\sum_{\omega=1}^{2}\sum_{j=1}^{2}\gamma_g\left(\mathbf{x}_j^{2\omega},\mathbf{y}_j^1,0\right)}{2} = \frac{\sum_{\omega=1}^{2}\left[(1+0),(0+0)\right]}{2} = \frac{(1+0)}{2} = \frac{1}{2} = 0.5 \quad (28)$$

In this case, there was one instance similar to the corresponding component of the test pattern (equal, since $\theta = 0$ during this run). Thus the weighted addition for class C_2 is $c_2 = 0.5$, given that the sum of results is divided by the cardinality of the class, $k_2 = 2$.

And for $\mathbf{y}^2$:

$$c_1 = \frac{\sum_{\omega=1}^{2}\sum_{j=1}^{2}\gamma_g\left(\mathbf{x}_j^{1\omega},\mathbf{y}_j^2,0\right)}{2} = \frac{\sum_{\omega=1}^{2}\left[(0+0),(0+1)\right]}{2} = \frac{(0+1)}{2} = \frac{1}{2} = 0.5 \quad (29)$$

There was one result of γ_g equal to 1, which coupled with the cardinality of class C_2 being 2 makes $c_1 = 0.5$.

$$c_2 = \frac{\sum_{\omega=1}^{2}\sum_{j=1}^{2}\gamma_g\left(\mathbf{x}_j^{2\omega},\mathbf{y}_j^2,0\right)}{2} = \frac{\sum_{\omega=1}^{2}\left[(0+0),(0+0)\right]}{2} = \frac{(0+0)}{2} = \frac{0}{2} = 0 \quad (30)$$

No similarities were found between the fundamental patterns of class C_2 and the test pattern $\mathbf{y}^2$ (for $\theta = 0$), thus $c_2 = 0$.

(viii) If there is more than one maximum among the different c_i, increment θ by 1 and repeat steps vi and vii until there is a unique maximum, or the stop condition $\theta \geq \rho$ is fulfilled.
There is a unique maximum (for each test pattern), so we go directly to step ix.

(ix) If there is a unique maximum, assign $\mathbf{y}$ to the class corresponding to such maximum. For $\mathbf{y}^1$:

$$C_{y^1} = C_2 \text{ since } \bigvee_{i=1}^{2} c_i = \bigvee(0, 0.5) = 0.5 = c_2 \quad (31)$$

while for $\mathbf{y}^2$:

$$C_{y^2} = C_1 \text{ since } \bigvee_{i=1}^{2} c_i = \bigvee(0.5, 0) = 0.5 = c_1 \quad (32)$$

(x) Otherwise, assign $\mathbf{y}$ to the class of the first maximum.
This is unnecessary since both test patterns have already been assigned a class.

We have seen that both patterns were assigned a class by following the algorithm of the Gamma classifier, but does this classification make sense? On one hand, $\mathbf{y}^1$ was assigned to class C_2, but notice that both $\mathbf{x}^3$ and $\mathbf{x}^4$ have relatively large values for the first components and smaller values for the second component; $\mathbf{y}^1$ behaves like this, and even has a value of 6 in the first component, equal to that of $\mathbf{x}^3$. Thus, it makes sense to assign $\mathbf{y}^1$ to class C_2.

On the other hand, $\mathbf{x}^1$ and $\mathbf{x}^2$ have comparatively lower values for the first components and larger ones for their second components. In that sense, $\mathbf{y}^2$ also has a smaller value for the first component and a larger one for the second component, being even equal to the second component of $\mathbf{x}^2$. Then, the fact that this test pattern is assigned the same class of $\mathbf{x}^1$ and $\mathbf{x}^2$, which is C_1, makes a lot of sense.

The Gamma classifier is inspired on the Alpha-Beta associative memories, taking the alpha and beta operators as basis for the gamma operator. As such, the Gamma classifier is a member of the Associative Approach to Pattern Recognition, in which the algorithms and models use concepts and techniques derived from associative memories in order to recognize and classify patterns. As can be seen, this classifier is relatively simple, requiring simple operations: its complexity is polynomial. Also, notice that while being iterative, the classifier will reach a solution in finite time: at best in one iteration, at worst in the same amount of iterations as the stop parameter indicates (see Eq. 17).

Although the initial Gamma classifier is not old, it has already been applied to several different problems: classification of the Iris Plant database, localization of mobile stations, software development effort estimation of small programs, and environmental data prediction, obtaining some promising preliminary results. In these problems, some quite different from each other —and even some for which the basic premises of the classifier do not hold— the Gamma classifier version 1 has shown competitive experimental results.

In this sense, the current work presents a relevant contribution, since a pattern classifier (in this case the Gamma classifier) was successfully applied to the task of predicting a future value, which is not the kind of task envisioned while designing and developing the Gamma model. Thus, the problem of pollutant forecasting has been tackled by using this classifier.

4. Experimental Design

In this Sec., the structure of the experiments is presented and discussed. The main goal of these experiments is the prediction of air pollution levels time series, built from data included in the RAMA database of SIMAT. The pollutant concentration forecasting implied by the former goal is done by applying the Gamma classifier, coupled with a coding technique originally introduced by the authors, which is presented later.

Some relevant considerations are taken into account: for simplicity, only a single time series will be considered. Also, the experimentation is done off-line: data are taken from a known time series, but using only some non-contiguous segments. If results are auspicious, the method will be extended in the future to include more data, and/or to allow for its application to live data, as it is being sampled.

In order to test the proposed method, several experimental runs were made, in two generations. For each experiment of both generations, a set of patterns was built from an already known time series and presented to the corresponding version of the Gamma classifier in order to train it. Then, a different set of patterns was constructed with a time series taken from future dates to those of the set mentioned above, simulating the situation where the first (fundamental) set is built from historical data while the second (test) set is made up by current data. Thus, the Gamma classifier was required to predict the next value, given a small amount of current data.

The 1G of experiments (1 to 12), which is further discussed in Sec. 5, made use of the Gamma classifier version 1. These experiments and their results were analyzed in order to refine that method, which in its improved state (along with the Gamma classifier version 2) was used for the 2G of experiments (13 to 28), which are presented in Sec. 7.

In all instances, the samples measured at a specific station of the RAMA subsystem of SIMAT were used. The fundamental set was made up by building patterns from the time series of a whole year, while the test set was built with patterns made up by the time series of one month of a subsequent year. Then, the Gamma classifier —version 1 on the 1G, version 2 on the 2G— was trained on the fundamental set, and later operated with the test set. For each item of one experimental run, a pattern from the test set was presented to the classifier, asking it what the next sample value would be.

The coding technique which allows the building of patterns from a time series is proposed in the following subsection. Then, the measures of performance are presented. Table 3 (in the following Sec.) presents the experiments done for both generations.

It is noteworthy that, in order to reproduce these experiments with data taken from different periods (whether what changes is the fundamental set, the test set, or both), it is necessary to build new pattern sets according to the pattern coding technique presented before, and to train and operate the model with this new sets.

4.1. *Pattern coding*

Each pattern is made up by n successive samples, concatenated each after the other. As the class for such pattern, the n+1-th sample is used (see Fig. 2). Thus, patterns are built from the samples as mentioned above, and then these patterns are grouped together into the fundamental and the testing sets for each experiment. In this manner, this problem of data prediction is presented as a classification problem, a task for which the Gamma classifier has shown to be quite apt, since its inception with version 1.

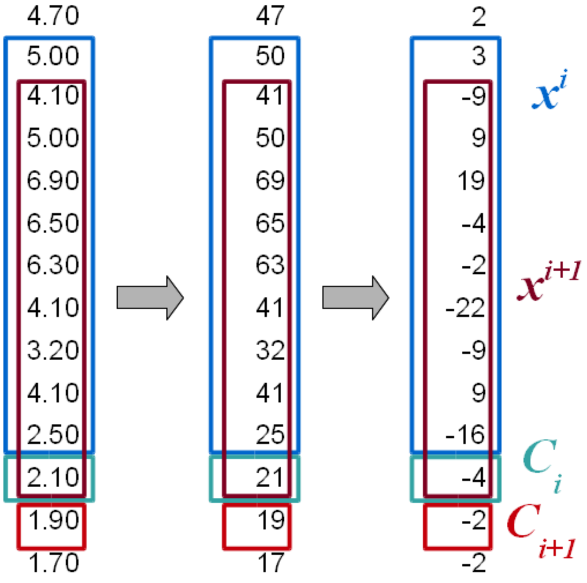

Fig. 2. How patterns are built from the samples, when n = 10.

However, in order to avoid potential problems regarding nonstationarity, the differences between consecutive samples were used.[1–3] Also, these differences are taken into account once all samples have been scaled appropriately, following steps 1 and 2 of the modified Johnson-Möbius code. This process is illustrated in Fig. 2.

For the experiments presented in this work, n was chosen to be equal to 10 ($n = 10$) since this configuration has given good results in preliminary tests.

As can be seen in Fig. 2, the i-th pattern is built by making each value of the series into one component of vector $\mathbf{x}^i$, in order, from the i-th value to the $i + (n - 1)$-th value. The class corresponding to $\mathbf{x}^i$, which is C_i, is the value following the last component of $\mathbf{x}^i$: C_i is equal to the $i + n$-th value. The $i + 1$-th pattern is built by shifting the former arrangement, one value ahead. Thus, $\mathbf{x}^{i+1}$ has as initial component the $i + 1$-th value and as final component the $i + n$-th value, while its class C_{i+1} is equal to the $i + n + 1$-th value.

4.2. *Performance measures*

Three quantitative measures of the performances shown by the Gamma classifier on this application are used. On one side we have the Rooted Mean Square Error (RMSE), which is a widely used measure of performance in data forecasting[4,9,11–15] and is calculated as shown in Eq. 33. On the other side there is the bias, which can be computed by following Eq. 34 and is used to describe how much the system is underestimating or overestimating the results. Finally, we use normalized fit (NFit, see Eq. 35) in order to compare the results shown by different pollutants. For all three equations, P_i represents the i-th predicted (future or unknown) value, while O_i is the i-th original (observed, actual, or known) value, and $mean(O_i)$ indicates the mean of all original values.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(P_i - O_i)^2} \qquad (33)$$

$$Bias = \frac{1}{n}\sum_{i=1}^{n}(P_i - O_i) \qquad (34)$$

$$NFit = \frac{|P_i - O_i|^2}{|O_i - mean(O_i)|^2} \qquad (35)$$

Generally speaking, small values of RMSE are desirable since they indicate a good prediction. On the other hand, bias is a secondary indicator of performance and gives more information about potential lines of improvement for the method, or potential exceptions in the data treated. Usually, small values of bias are also preferred over larger values. Regarding NFit, a lower value also indicates a better performance.

Another difference between these measures of comparison between forecasting methods is that RMSE and NFit give only positive values, while bias can give both positive and negative results (overestimation and underestimation, respectively). If the bias for a method is consistently positive or consistently negative, this can be an indicator of a lower or upper bound to the expected results.

On one hand, the magnitude of results shown on RMSE and bias by different time series will depend on the magnitude of variation among the values on individual data points for each time series. For example, if one time series has values with meaningful decimals up to millimeters (e.g. 2.3, 2.6, 3.1… in cm) while another has values which vary by meters (say 56, 84, 79… in m), then both RMSE and bias will have greater values for the latter time series than for the former one. Thus, these performance measures are not particularly good to compare results on time series used for different variables.

On the other hand, NFit considers the squared error (similarly to RMSE), but normalizes it with respect to the squared difference between the actual value and the mean of the actual values. This has the effect of penalizing errors on data points close to the mean of the original values, assuming that such values should be easier to predict since they are more similar to what can be achieved by simply using the average of the original values as the predictor. Since the focus of this measure of performance is on how different the value to be predicted is regarding the mean of the actual values, the different magnitudes of data points from different time series has a lesser contribution to the difference in the measure. Thus, the NFit is a better mean of performance comparison between results of different variables.

5. First Generation of Experiments

For the 1G of experiments, the structure and assumptions mentioned in the former Sec. are used (including the coding technique), coupled with the Gamma classifier version 1.

First, the experiments run are described. Then, the results obtained are presented and discussed, and later the performance shown by the proposed method on those experiments is compared with that exhibited by other methods, as seen in current literature.

5.1. *First generation: Experiments done*

Table 3 summarizes the experiments done, in particular regarding the pollutant involved, the monitoring station from which data were taken, as well as the time periods from whose data the fundamental and test patterns were built.

Table 3. Configurations of experiments; Gen. means "Generation" and Fund. means "Fundamental".

				Data Set Period	
Gen.	**Experiment**	**Pollutant**	**Station**	**Fund.**	**Test**
1	1	CO	IMP	2000	March 2001
	2	CO	IMP	2006	February 2007
	3	CO	IMP	2006	May 2007
	4	O_3	CES	2006	February 2007
	5	O_3	CES	2006	May 2007
	6	SO_2	TLI	2006	February 2007
	7	SO_2	TLI	2006	May 2007
	8	SO_2	TAC	2001	February 2002
	9	NOx	TAC	2002	June 2003
	10	NO	TLI	2009	May 2010
	11	NO_2	TLI	2009	May 2010
	12	NO_x	TLI	2009	May 2010
2	13	CO	EAC	2010	January 2011
	14	CO	EAC	2010	February 2011
	15	CO	XAL	2010	January 2011
	16	CO	XAL	2010	February 2011
	17	O_3	EAC	2010	January 2011
	18	O_3	EAC	2010	February 2011
	19	O_3	COY	2010	January 2011
	20	O_3	COY	2010	February 2011
	21	SO_2	EAC	2010	January 2011
	22	SO_2	EAC	2010	February 2011
	23	SO_2	TLI	2010	January 2011
	24	SO_2	TLI	2010	February 2011
	25	NO_x	EAC	2010	January 2011
	26	NO_x	EAC	2010	February 2011
	27	NO_x	IZT	2010	January 2011
	28	NO_x	IZT	2010	February 2011

Thus, experiment 1 (for instance) took data from the *Instituto Mexicano del Petróleo* (IMP) station, building the patterns which make up the fundamental set from the samples taken during the whole 2000 year, and using the samples taken during March 2001 to build the patterns which constitute the test set. Then, for experiment 1, the fundamental set is formed by 8750 patterns associated to their corresponding classes, while the test set includes 734 associations.

Throughout all the experiments (both generations), patterns were built using data taken from eight different stations: *Instituto Mexicano del Petróleo* (IMP) for experiments 1–3; *Cerro de la Estrella* (CES) for experiments 4 and 5; *Tultitlán* (TLI) for experiments 6, 7, 10–12, 23, and 24; *Tacuba* (TAC) for experiments 8 and 9, *ENEP Acatlán* (EAC) for experiments 13, 14, 17, 18, 21, 22, 25, and 26; *Xalostoc* (XAL) for experiments 15 and 16; *Coyoacán* (COY) for experiments 19 and 20; and *Iztacalco* (IZT) for experiments 27 and 28.

In the experiments of the 1G, the fundamental and test sets were chosen to be taken from non-consecutive segments of the same time series. This was done mainly to test the robustness of the method, making sure it does not require data corresponding to the last samples taken in order to give a good prediction of the next sample. Of course, the underlying premise of this choice is that the conditions associated to the data from which the classifier learns, are still in effect on the data which is to be used for prediction. If this is not the case, it should be expected that the prediction error would increase.

5.2. *First generation: Results and discussion*

Table 4 presents the performance exhibited by the Gamma classifier version 1 during the 12 experiments run during the first generation, while Figs. 3 and 4 show the results of experiments 3 (CO) and 12 (NO_x), respectively.

As can be seen, the experiments done with data of the same criteria pollutant show similar values for RMSE, while such results are in different ranges for different pollutants. For instance: the mean RMSE for the three experiments done on CO (experiments 1, 2, and 3) is 0.705645 ppm, while the mean RMSE for the SO_2 experiments (7 and 8) is 0.010600 ppm.

Table 4. Experimental results of the first generation (ppm).

Experiment	Pollutant	Bias	RMSE	NFit
1	CO	-0.028513	0.779154	32.809140
2	CO	0.012042	0.726013	777.523037
3	CO	0.062183	0.611769	78.601059
4	O_3	0.000918	0.012302	80.927758
5	O_3	0.000417	0.014443	5.895891
6	SO_2	0.000676	0.012096	54.221637
7	SO_2	0.000795	0.010487	17.041456
8	SO_2	0.000408	0.009218	61.145162
9	NO_x	0.001543	0.026103	75.953068
10	NO	0.000001	0.000037	0.000014
11	NO_2	0	0	0.000000
12	NO_x	0	0	0.000000

This behavior is due to the concentration values commonly shown by each pollutant: CO usually shows larger concentration values than SO_2, which in turn reflect on larger values for errors. Thus, an RMSE of 0.611769 ppm may be too large for a good prediction of NO, while it may indicate a good prediction of CO, for instance.

More specifically, the best performances in RMSE for each pollutant are the following: 0.611769 ppm in CO at experiment 3, 0.012302 ppm of O_3 in experiment 4, 0.009218 ppm for SO_2 by experiment 8, 0.000037 ppm in NO at experiment 10, while both experiments 11 and 12 show the best results for their respective pollutants, NO_2 and NO_x, both with 0.0 ppm.

Regarding NFit, we can see that half the experiments show results between 32 ppm and 81 ppm, which may be considered to be relatively good results.

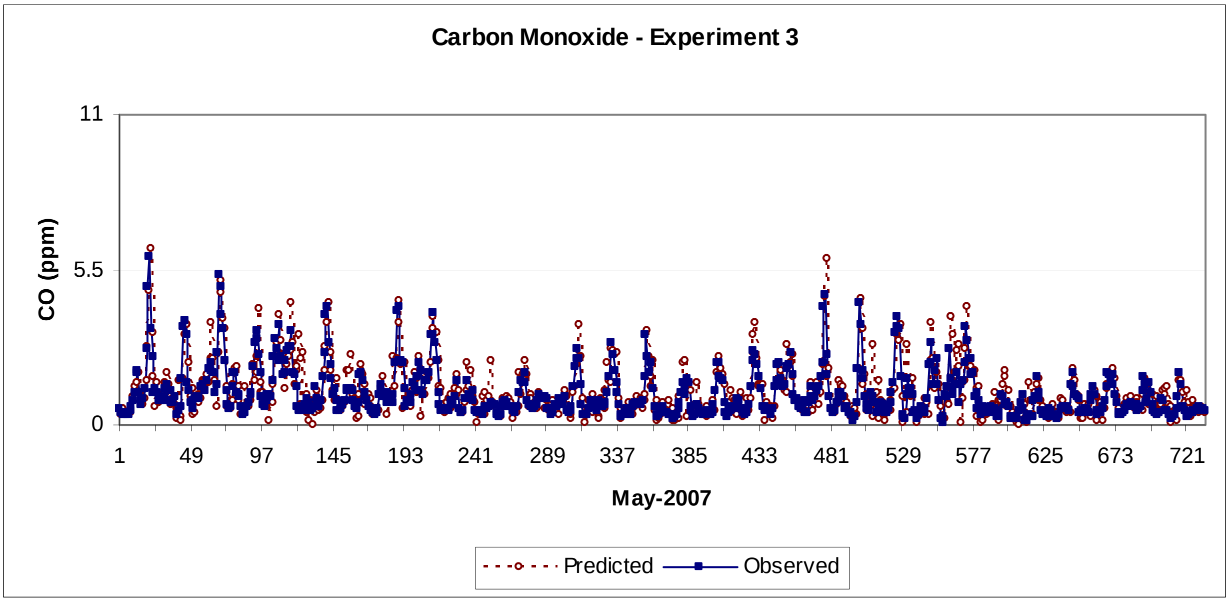

Fig. 3. Experiment 3 results: predicted *vs* observed time series

Yet, the best performance is shown by experiments 7 (17.041456 ppm), 5 (5.895891 ppm), 10 (0.000014 ppm), 11, and 12 (both 0 ppm). However, experiment 2 presents a quite large value for NFit, even though its RMSE value is quite similar to those of experiments 1 and 3.

What could cause such a large value of NFit for experiment 2? Recalling that NFit penalizes errors on data points whose values are close to the mean of the original data, it is not surprising that even a small error gets magnified when done on such a data point. For instance, on February 2, 2007 at 2:00 hours, the actual concentration of CO was 1.36 ppm, while the predicted value was 3.23 ppm. While this is a relatively large error (1.87 ppm), it is by no means the largest in the month (the maximum error is 3.44 ppm). However, the mean of the original values is 1.357201 ppm, which makes the denominator on the NFit formula equal to 0.000007833 ppm; this in turn gives an NFit value for that prediction of 446419.005 ppm.

This large errors on data points whose values are close to the mean of the original data are probably due to the characteristics of the Gamma classifier version 1, which is more sensitive to the presence of local extremes (minima or maxima) instead of the difference to the mean of actual values.[17–18]

Since the data to be predicted on experiment 2 presents more changes in trend than the test time series of experiment 3, it is to be expected that more data points with errors will be present than at experiment 3. Given that some of this erroneous data points are close to the mean of original values, they are greatly penalized by NFit.

It is also noteworthy that in experiments 10 to 12 the Gamma classifier exhibits a remarkably good performance. In experiment 10 (NO), there is only one pattern predicted incorrectly: the Gamma classifier forecasts a concentration of nitrogen monoxide of 0.006 ppm, while the registered sample was 0.005 ppm. Given that there were 733 associations in the test set, this amount of error out of a whole month is almost

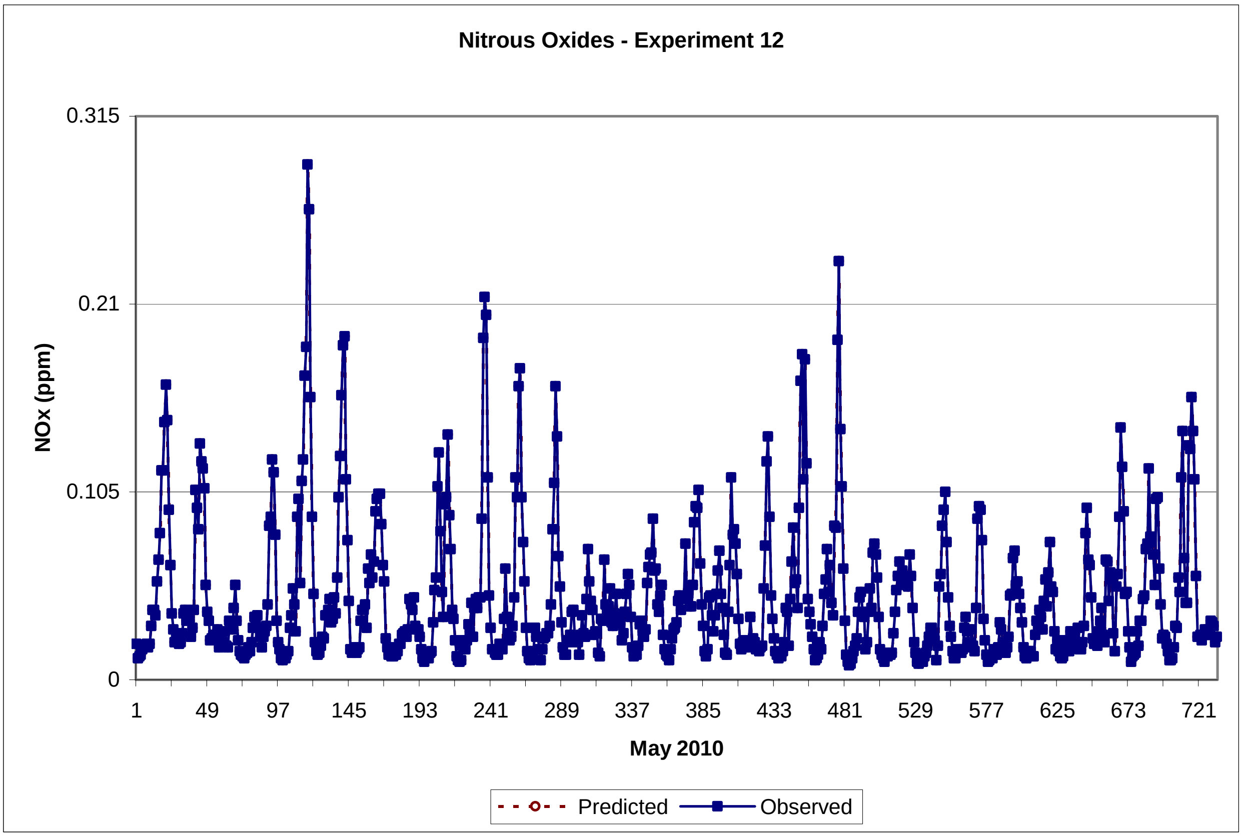

Fig. 4. Experiment 12 results: predicted *vs* observed time series

negligible, if not altogether insignificant. The results for experiments 11 (NO_2) and 12 (NO_x) are even better: there was not even one mistake in the whole month for the prediction of both nitrogen dioxide and nitrogen oxides. This is quite evident in Fig. 4, where the predicted time series is hidden behind that of the observed data.

These results are, of course, rather surprising. The ultimate goal of pattern recognition in general,[25] and its application on time series prediction in particular, is to correctly recognize unknown patterns, thus predicting future data points with as little error as possible. The ideal scenario is, then, to predict the new data points with an error of 0. However, considering that the proposed method uses only one variable when there are more available, and that the phenomenon under study is considered to be a rather complex one,[11,14] it is quite unexpected to obtain a perfect prediction.

The fact that two experiments exhibit a perfect performance (i.e. no error) and that another almost reaches this pinnacle, showing only one data point in error out of 733 considered in the experiment, is far beyond the initial expectations of the authors. Although we intend our method to perform competitively, we did not expected it to do this well.

Then, what causes such a good performance? One possible explanation is that, at least for those three experiments (11, 12, and 10), there is overfitting[25] between the fundamental and test sets. However, since the data points in the corresponding time series used to make up the patterns belonging to the fundamental and test set are taken from different segments of said time series, it can be assumed that those data are different. If there is actually a problem of overfitting, it is most likely due to high similarity between data from both segments.

In the field of Pattern Recognition, such situations where the classification model is overfitted to the learning data are commonly dealt with using cross validation techniques to test different combinations of partitioning the available data between the fundamental and test sets, in order to obtain a performance with data free of this problem.[25] However, such recourse is not adequate to the present situation, since the actual goal is to predict current and future (unknown) data by using historical (known) data, thus making the mixing of these disjoint sets unnecessary and most likely not advisable.

Actually, this situation is strongly related to a couple of considerations mentioned before. One is that the segment of the time series used for learning is non-contiguous to the segment used for testing, in order to avoid as much as possible the problem of overfitting. Also, the fact that data learned from 2009 may be used to perfectly (or almost perfectly) predict the concentration of NO, NO_2 and NO_x during May 2010, indicates that the conditions prevalent during 2009 are still in effect during May 2010, at least for these contaminants at the *Tultitlán* station (see tables 3 and 4).

More importantly, this particular application requires that overfitting is not excluded from the learning and operation data sets, at least when used with live data. The latter is a consequence of the fact that in order to predict, we need to use known data, along with the latest new data, whether these data sets are highly coupled or not. Of course, the results obtained should be compared against other experimental results with great care.

Yet, the fundamental reason for this shockingly good behavior remains to be fully explored, beyond the posed proposition that high similarity between the conditions reflected by the data points used for learning and testing is the cause. If such fundamental cause or conditions are identified, they could be used to greatly improve the performance of prediction methods such as the one proposed in the current paper.

However, if an event should arise in the testing data set which is particularly different to what has been learned, then the prediction results would most likely present large errors. An example of such situation could be a volcanic explosion whose residues drift into the geographic area of interest (which has happened several times in recent decades[26–27]). If such an irregular event happens, the predicted results should be considered to have little trustworthiness.

Both kinds of situation have arisen in other applications of the Gamma classifier version 1. On one hand, when that algorithm was used to classify the position of wireless sensors, the experimental results showed 3 errors out of 36 associations in one experiment, and 4 out of 36 in another, (91.67% and 88.89% of correct classification, respectively). Meanwhile, the well-known k Nearest Neighbor (k-NN) classifier[25] (with $k = 1$) offered 14 mistakes, for a correct classification of 65%. These results are still in the process of being published.

On the other hand, the discussion of experiment 7 in Ref. 18 analyses the behavior of SO_2 during May 2007, which presents an exceptionally high concentration on May 19 (0.174 ppm when the mean for that month was 0.007 ppm). This kind of events causes the Gamma classifier to offer a prediction with large errors, since these exceptional situations are not properly covered in the data from which the method learns.

5.3. *First generation: Comparison with other methods*

Still, one important question remains: just how good are these results when compared to other models? Here arises an important problem of this field, given that there are no common, public, and standardized databases being used by the scientific community working on this application. Thus, most researchers use the database which is closest or more easily available to them, leading to many publications on this field, but with no directly comparable results. This problem remains open.

For the database used in this work (the RAMA subsystem of SIMAT), there are very few published experimental results available (table 5). As can be seen, they are greatly surpassed; in particular for the case of O_3, where the Gamma classifier exhibits a much smaller error: it is several orders of magnitude smaller.

Although no direct comparisons are readily available, some indirect comparisons can be made. These are presented in table 6, where several recent papers are included.

Notice that the target contaminant is quite variable: besides the more common CO, NO_2, SO_2, and O_3, there is also benzo(a)pyrene (BaP), which is one particular component of PM_{10}.[9] Regarding the unit in which each contaminant is measured, there are five alternatives present: parts per million (ppm), parts per billion (ppb), milligrams per cubic meter ($\mu g/m^3$), nanograms per cubic meter (ng/m^3), and ERPI, which is an acronym for "European Regional Pollution Index".[9] On the other hand, the methodologies followed by each group of researchers are quite different too.

Table 5. Comparison of related results (SIMAT database, O_3) in absolute mean error given for pollutant concentration in ppm; exp. means "experiment".

Algorithm Used	Performance (Abs. Mean Error)
Bayesian network[10]	0.221000
C4.5[10]	0.176400
Neural network[10]	0.160000
Gamma classifier exp. 4 (current work & Ref. 18)	0.000918
Gamma classifier exp. 5 (current work & Ref. 18)	0.000417

Even though the results cannot be directly compared, the performance exhibited by the Gamma classifier is also competitive, as shown by its RMSE values —column titled "Performance RMSE (original unit)"— which are quite smaller than those of the other methods.

Again, this comparison cannot be made directly, since different data sets, describing different local

Table 6. Comparison of related results (diverse databases) in RMSE; exp. means "experiment", N/A means "not available".

Algorithm Used	Pollutant Considered (measurement unit)	Performance RMSE (original unit)	Performance RMSE ($\mu g/m^3$)
IITLS[15]	NO_x ($\mu g/m^3$)	19.99	19.99
Neural network[13]	O_3 ($\mu g/m^3$)	15	15
Neural network[12]	O_3 (ppb)	13.79, 9.43	27.028, 18.483
Online SVM[14]	SO_2 ($\mu g/m^3$)	12.96, 10.90	12.96, 10.90
Neural network[11]	CO, NO_2, SO_2, and O_3 (ERPI)	5.852, 1.365	N/A
Gamma classifier exp. 3 (current work & Ref. 18)	CO (ppm)	0.611769	700.476
Multivariate linear regression[9]	BaP (ng/m^3)	0.449	0.000449
Gamma classifier exp. 4 (current work & Ref. 18)	O_3 (ppm)	0.012302	24.112
Gamma classifier exp. 8 (current work & Ref. 18)	SO_2 (ppm)	0.009218	24.151
Gamma classifier exp. 10 (current work)	NO (ppm)	0.000037	0.069560
Gamma classifier exp. 11 (current work)	NO_2 (ppm)	0.000000	0.000000
Gamma classifier exp. 12	NO_x (ppm)	0.000000	0.000000

phenomena, using different measure units, are employed. In particular, a conversion could be tried among the different measure units, in order to use a unique measure unit. This is done in the column titled “Performance RMSE (μg/m^3)”, in which all but one result is converted from its original unit into μg/m^3, according to the definitions published in Refs. 28–32. The performance of the ANN model described in Ref. 11 was not converted, since ERPI presents the value corresponding to the pollutant with the highest concentration, thus it is not known how to properly convert from that unit into μg/m^3.

Using the latter values of RMSE to compare results, it can be seen that experiments 4 and 8 exhibit good performance (considering that only one time series is being used), while experiment 3 presents a very large error. This is clearly a good starting point to look for improvements to the method. The best results are shown by experiments 10, 11, and 12.

6. Gamma Classifier version 2

The experiments of the 1G were analyzed and used as the basis to improve both the classifier and the method of applying it to air quality data prediction, as well as to develop the necessary theoretical foundation. The emergent Gamma classifier version 2 was then tested with the 2G of experiments, comprised of experiments 13 to 28.

There were two clear observations done on the analysis of the experiments of the 1G. The first one is that, sometimes, the predicted value for a pollutant concentration is a negative number. Given that this numerical result represents the concentration of a given substance in atmospheric air, such negative value makes no sense. The second observation is that the Gamma classifier version 1, as presented in Sec. 4, is not guaranteed to correctly classify the whole fundamental set. This means that for some instances, if a test pattern is equal to a fundamental pattern, the classifier will not necessarily yield the correct class: for a percentage of the fundamental patterns larger than 0, the classifier forgets their correct class.

In the case of the first issue, the answer is quite easy: if the predicted value for the pollutant concentration is negative, it is obviously erroneous, so it needs correcting. The simplest manner in which this correction may be done is to assign a value of 0, which is the minimum possible value which is still valid. This is the first improvement to the method, and is included as a post-processing step, once the classifier has given its output and it is being converted back into a concentration value in ppm (see the steps illustrated in Fig. 2, in reverse order).

The second issue is not so simple to address. On one hand, this aspect of Pattern Classification is considered to be of little relevance by most classifiers, since they are willing to exhibit errors on the fundamental set in order to achieve a better generalization, and thus a better performance on the test set. In this sense, there is only one widely known and commonly used classifier which does not misclassify a pattern it has already learnt: the k-NN when $k = 1$. On the other hand, it has been an ongoing effort in the community of researchers which have developed the Associative Approach to Pattern Recognition to ensure that the models and algorithm belonging to this approach recognize and classify correctly the whole fundamental set. In this regard, the Gamma classifier version 1, as originally introduced, was the only Associative model unable to correctly classify the whole fundamental set in a guaranteed manner.

Operatively, this issue means that some test patterns were incorrectly classified, despite the fact that the method already knew them, since these test patterns are equal to some fundamental pattern. The only way to solve this problem is to modify the Gamma classifier, improving its theoretical foundation along the way.

Therefore, a new version of the classifier algorithm is developed: version 2. However, before said algorithm is presented, some theoretical developments are needed.

6.1. *Preliminaries*

First, let us define more specifically the fundamental pattern set to be dealt with by the Gamma classifier, as well as its properties.

Let the *fundamental set* of the Gamma classifier (version 2) be the set of patterns associated to a class, of the form $\left\{\left(\mathbf{x}^{\mu},\mathbf{y}^{\mu}\right)\middle|\mu=1,2,\ldots,p\right\}$ where $\mathbf{x}^{\mu}$ is a pattern and $\mathbf{y}^{\mu}$ is its corresponding class. For such a fundamental set, the following three properties hold:

$$\mathbf{x}^{i} \neq \mathbf{x}^{j} \;\forall i,j \in \{1,2,\ldots,p\} \text{ such that } i \neq j \quad (36)$$

meaning there are no repeated patterns,

$$\mathbf{x}^{i} = \mathbf{x}^{j} \rightarrow \mathbf{y}^{i} = \mathbf{y}^{j} \;\forall i,j \in \{1,2,\ldots,p\} \quad (37)$$

which makes sure no pattern may have associated more than one class, and

$$\mathbf{y}^i \neq \mathbf{y}^j \rightarrow \mathbf{x}^i \neq \mathbf{x}^j \ \forall i, j \in \{1,2,\ldots,p\} \tag{38}$$

that is, different classes have different patterns associated to each other.

Combined, these three properties imply that the fundamental set must induce a relation from the set of patterns to the set of classes, such that said relation is a function.

Now, we need three lemmas which will later be used to prove a theorem related to the classification of fundamental patterns.

Lemma 1. *When two binary vectors* $\mathbf{x}, \mathbf{y} \in A^n$ *are presented to the generalized Gamma operator* γ_g *, with* θ *= 0; if the results is 1, then* **x** *is equal to* **y***:*

$$\theta = 0, \gamma_g(\mathbf{x}, \mathbf{y}, \theta) = 1 \rightarrow \mathbf{x} = \mathbf{y} \tag{39}$$

Proof. By the definition of the generalized Gamma operator γ_g ,

$$\begin{aligned} &\gamma_g(\mathbf{x}, \mathbf{y}, \theta) = 1 \rightarrow \\ &\rightarrow m - u_\beta\left[\alpha(\mathbf{x}, \mathbf{y}) \bmod 2\right] \leq \theta \end{aligned} \tag{40}$$

However, given that θ = 0 and $m = n$ since $\mathbf{y} \in A^n$:

$$\begin{aligned} &\gamma_g(\mathbf{x}, \mathbf{y}, \theta) = 1 \rightarrow \\ &\rightarrow n - u_\beta\left[\alpha(\mathbf{x}, \mathbf{y}) \bmod 2\right] \leq 0 \end{aligned} \tag{41}$$

since $m, n \in \mathbb{Z}^+$:

$$\begin{aligned} &\rightarrow n \leq 0 + u_\beta\left[\alpha(\mathbf{x}, \mathbf{y}) \bmod 2\right] \\ &\rightarrow n = u_\beta\left[\alpha(\mathbf{x}, \mathbf{y}) \bmod 2\right] \end{aligned} \tag{42}$$

this in turn means that:

$$\rightarrow \alpha(\mathbf{x}, \mathbf{y}) \bmod 2 = (1\,1 \ldots 1)_{1\times n} \tag{43}$$

but since $\alpha : A \times A \rightarrow B$, there are three possible outcomes for α:

$$\begin{aligned} &\alpha(\mathbf{x}, \mathbf{y}) = 0, \quad 0 \bmod 2 = 0 \\ &\alpha(\mathbf{x}, \mathbf{y}) = 1, \quad 1 \bmod 2 = 1 \\ &\alpha(\mathbf{x}, \mathbf{y}) = 2, \quad 2 \bmod 2 = 0 \end{aligned} \tag{44}$$

thus:

$$\begin{aligned} &\alpha(\mathbf{x}, \mathbf{y}) \bmod 2 = (1\,1 \ldots 1)_{1\times n} \rightarrow \\ &\rightarrow \alpha(\mathbf{x}, \mathbf{y}) = (1\,1 \ldots 1)_{1\times n} \\ &\rightarrow \alpha(\mathbf{x}_i, \mathbf{y}_i) = 1 \ \forall i \in \{1,2,\ldots,n\} \end{aligned} \tag{45}$$

and given the definition of α, only two possible inputs give the necessary outcome:

$$\begin{aligned} &\alpha(0,0) = 1 \\ &\alpha(1,1) = 1 \end{aligned} \tag{46}$$

thus:

$$\begin{aligned} &\alpha(\mathbf{x}_i, \mathbf{y}_i) = 1 \ \forall i \in \{1,2,\ldots,n\} \rightarrow \\ &\rightarrow \mathbf{x}_i = \mathbf{y}_i \ \forall i \in \{1,2,\ldots,n\} \\ &\rightarrow \mathbf{x} = \mathbf{y} \end{aligned} \tag{47}$$

therefore:

$$\gamma_g(\mathbf{x}, \mathbf{y}, 0) = 1 \rightarrow \mathbf{x} = \mathbf{y} \tag{48}$$

□

Lemma 2. *When the test pattern presented to the Gamma classifier during the classification phase* $\tilde{\mathbf{x}}$ *is equal to a fundamental pattern* $\mathbf{x}^\sigma$*, associated to class* $\mathbf{y}^a$*, this pattern will contribute n to the weighted addition of that class* c_a*, for any value of* $\theta \geq 0$.

Proof. Given the manner in which the weighted addition c_i —corresponding to the i-th class— is computed (according to Eq. 18), the result is the sum of all values $\gamma_g\left(\mathbf{x}_j^{i\omega}, \tilde{\mathbf{x}}_j, \theta\right) = 1$ divided by the cardinality of that class, k_i. Then, by the definition of the generalized Gamma operator γ_g we have:

$$\begin{aligned} &\gamma_g\left(\mathbf{x}_j^{i\omega}, \tilde{\mathbf{x}}_j, \theta\right) = 1 \rightarrow \\ &\rightarrow m - u_\beta\left[\alpha\left(\mathbf{x}_j^{i\omega}, \tilde{\mathbf{x}}_j\right) \bmod 2\right] \leq \theta \end{aligned} \tag{49}$$

On the other hand, it is clear that the most restrictive case is for θ = 0, since the fact that a pattern fulfills the former condition when θ = 0 guarantees that it will do so for any value θ > 0. Also, according to lemma 1:

$$\gamma_g\left(\mathbf{x}_j^{i\omega}, \tilde{\mathbf{x}}_j, 0\right) = 1 \rightarrow \mathbf{x}_j^{i\omega} = \tilde{\mathbf{x}}_j \tag{50}$$

Now, since $\tilde{\mathbf{x}} = \mathbf{x}^\sigma$ which in turn belongs to the a-th class by being associated to $\mathbf{y}^a$, the index transformation of step iv of the algorithm implies:

$$\begin{aligned} &\tilde{\mathbf{x}} = \mathbf{x}^\sigma, \mathbf{x}^\sigma = \mathbf{x}^{ab} \rightarrow \tilde{\mathbf{x}} = \mathbf{x}^{ab} \\ &\rightarrow \tilde{\mathbf{x}}_j = \mathbf{x}_j^{ab} \ \forall j \in \{1,2,\ldots,n\} \end{aligned} \tag{51}$$

thus, when computing $\gamma_g\left(\mathbf{x}_j^{i\omega},\tilde{\mathbf{x}}_j,0\right)$ for indices $i = a$ and $\omega = b$, we need to do $\gamma_g\left(\mathbf{x}_j^{ab},\tilde{\mathbf{x}}_j,0\right)$, then:

$$\begin{aligned}&\tilde{\mathbf{x}} = \mathbf{x}^{ab} \rightarrow\\ &\rightarrow \gamma_g\left(\mathbf{x}_j^{ab},\tilde{\mathbf{x}}_j,0\right) = 1\ \forall j \in \{1,2,\ldots,n\}\end{aligned} \tag{52}$$

Therefore, $\tilde{\mathbf{x}} = \mathbf{x}^{ab}$ contributes with n values of 1 to the weighted addition for the a-th class c_a.

□

Lemma 3. *When the test pattern presented to the Gamma classifier during the classification phase* $\tilde{\mathbf{x}}$ *is equal to a fundamental pattern* $\mathbf{x}^\sigma$*, associated to class* $\mathbf{y}^a$*, all other fundamental patterns* $\mathbf{x}^\delta \neq \mathbf{x}^\sigma$ *will contribute with an amount* $d < n$ *to the weighted addition of their corresponding class* c_δ*, for a value of* $\theta = 0$.

Proof. Given the manner in which the weighted addition c_i —corresponding to the i-th class— is computed (according to Eq. 18), the result is the sum of all values $\gamma_g\left(\mathbf{x}_j^{i\omega},\tilde{\mathbf{x}}_j,\theta\right) = 1$ divided by the cardinality of that class, k_i. Then, by the definition of the generalized Gamma operator γ_g we have:

$$\begin{aligned}&\gamma_g\left(\mathbf{x}_j^{i\omega},\tilde{\mathbf{x}}_j,\theta\right) = 1 \rightarrow\\ &\rightarrow m - u_\beta\left[\alpha\left(\mathbf{x}_j^{i\omega},\tilde{\mathbf{x}}_j\right)\operatorname{mod} 2\right] \leq \theta\end{aligned} \tag{53}$$

Also, according to lemma 1:

$$\gamma_g\left(\mathbf{x}_j^{i\omega},\tilde{\mathbf{x}}_j,0\right) = 1 \rightarrow \mathbf{x}_j^{i\omega} = \tilde{\mathbf{x}}_j \tag{54}$$

Now, since $\tilde{\mathbf{x}} = \mathbf{x}^\sigma$ which in turn belongs to the a-th class by being associated to $\mathbf{y}^a$, the index transformation of step iv of the algorithm implies:

$$\tilde{\mathbf{x}} = \mathbf{x}^\sigma, \mathbf{x}^\sigma = \mathbf{x}^{ab} \rightarrow \tilde{\mathbf{x}} = \mathbf{x}^{ab} \tag{55}$$

On the other hand, since $\mathbf{x}^\delta \neq \mathbf{x}^\sigma$:

$$\mathbf{x}^\delta \neq \mathbf{x}^\sigma \rightarrow \exists j \in \{1,2,\ldots,n\} \middle| \mathbf{x}_j^\delta \neq \mathbf{x}_j^\sigma \tag{56}$$

thus, when computing $\gamma_g\left(\mathbf{x}_j^{i\omega},\tilde{\mathbf{x}}_j,0\right)$ for indices $i \neq a$ and $\omega \neq b$ (i.e. for any fundamental pattern $\mathbf{x}^\delta \neq \mathbf{x}^\sigma$), we have:

$$\begin{aligned}&\mathbf{x}^\delta \neq \mathbf{x}^\sigma \rightarrow \mathbf{x}^\delta \neq \tilde{\mathbf{x}} \rightarrow\\ &\rightarrow \exists j \in \{1,2,\ldots,n\} \text{ such that}\\ &\gamma_g\left(\mathbf{x}_j^{i\omega},\tilde{\mathbf{x}}_j,0\right) = 0\ \forall \mathbf{x}^\delta \neq \mathbf{x}^\sigma\end{aligned} \tag{57}$$

Now, since the amount of components of the fundamental pattern $\mathbf{x}_j^\delta \neq \tilde{\mathbf{x}}_j$ different to the test pattern is unknown (given that it depends on the actual fundamental pattern $\mathbf{x}^\delta$), what is known is that it may be as little as 1 and as large as n; then, the amount d that $\mathbf{x}^\delta$ contributes to its corresponding weighted addition c_δ is:

$$\begin{aligned}&\exists j \in \{1,2,\ldots,n\} \text{ such that}\\ &\gamma_g\left(\mathbf{x}_j^{i\omega},\tilde{\mathbf{x}}_j,0\right) = 0\ \forall \mathbf{x}^\delta \neq \mathbf{x}^\sigma \rightarrow\\ &\rightarrow 0 \leq d < n\ \forall \mathbf{x}^\delta \neq \mathbf{x}^\sigma\end{aligned} \tag{58}$$

Therefore, $\mathbf{x}^\delta \neq \mathbf{x}^\sigma$ contributes with $d < n$ values of 1 to the weighted addition for the δ-th class c_δ.

□

With the former definition of the fundamental set for the Gamma classifier version 2 and the latter three lemmas, we are ready to present and prove a theorem related to the classification of fundamental sets.

Theorem 1. *When the test pattern presented to the Gamma classifier during the classification phase* $\tilde{\mathbf{x}}$ *is equal to a fundamental pattern* $\mathbf{x}^\sigma$*, if each fundamental pattern* $\mathbf{x}^\mu$ *belongs to a class by itself (i.e. each class* $\mathbf{y}^\mu$ *is associated to only one pattern,* $\mathbf{x}^\mu$*) then the whole fundamental set will be correctly classified.*

Proof. Let us consider a test pattern $\tilde{\mathbf{x}}$ equal to an arbitrarily chosen fundamental pattern $\tilde{\mathbf{x}} = \mathbf{x}^\sigma$. Now, according to lemma 2, $\tilde{\mathbf{x}} = \mathbf{x}^\sigma$ will contribute n to its corresponding weighted addition c_σ, regardless of which value of $\theta \geq 0$ is being computed on.

Also, according to lemma 3, all other fundamental patterns $\mathbf{x}^\delta \neq \mathbf{x}^\sigma$ will contribute with $d < n$ to their corresponding weighted addition c_δ.

On the other hand, since each class contains only one pattern, $k_i = 1$ and the weighted addition for the i-th class is computed as:

$$\begin{aligned}c_i &= \frac{\sum_{\omega=1}^{k_i}\sum_{j=1}^{n}\gamma_g\left(\mathbf{x}_j^{i\omega},\tilde{\mathbf{x}}_j,\theta\right)}{1}\\ &= \sum_{j=1}^{n}\gamma_g\left(\mathbf{x}_j^{i},\tilde{\mathbf{x}}_j,\theta\right)\end{aligned} \tag{59}$$

Then, when computing the weighted additions during the first iteration of the algorithm (with $\theta = 0$) we have:

$$c_\sigma = \sum_{j=1}^{n}\gamma_g\left(\mathbf{x}_j^\sigma,\tilde{\mathbf{x}}_j,\theta\right) = n \tag{60}$$

and

$$c_\delta = \sum_{j=1}^{n} \gamma_g\left(\mathbf{x}_j^\delta, \tilde{\mathbf{x}}_j, \theta\right) = d < n \;\forall c_\delta \neq c_\sigma \qquad (61)$$

thus, there will be a unique maximum of such weighted additions:

$$\bigvee c_i = c_\sigma = n \qquad (62)$$

Therefore, $\tilde{\mathbf{x}} = \mathbf{x}^\sigma$ will be assigned class c_σ on the first iteration of the algorithm:

$$\bigvee c_i = c_\sigma \rightarrow \tilde{\mathbf{y}} = \mathbf{y}^\sigma \qquad (63)$$

Given that $\tilde{\mathbf{x}} = \mathbf{x}^\sigma$ was chosen arbitrarily, the latter result holds for all fundamental patterns. Therefore, the whole fundamental set is correctly classified.

□

Taking into account these developments, the Gamma classifier algorithm version 2 is introduced below.

6.2. *The Gamma classifier algorithm version 2*

Let $\left\{\left(\mathbf{x}^\mu, \mathbf{y}^\mu\right) \middle| \mu = 1, 2, \ldots, p\right\}$ be the fundamental pattern set (according to the definition presented in the previous subsection) with cardinality *p*; when a test pattern $\tilde{\mathbf{x}}$ (which is an *n*-dimensional real valued vector $\tilde{\mathbf{x}} \in \mathbb{R}^n$, with $n \in \mathbb{Z}^+$) is presented to the Gamma classifier version 2, these steps are followed:

(i) Code the fundamental set with the modified Johnson-Möbius code, obtaining a value e_m for each component. This e_m value is calculated as defined by Eq. 16.

(ii) Compute the stop parameter, as expressed in Eq. 17.

(iii) Code $\tilde{\mathbf{x}}$ with the modified Johnson-Möbius code, using the same parameters used with the fundamental set (step i). If any $\tilde{\mathbf{x}}_j$ is greater than the corresponding $e_m(j)$, the γ_g operator will use such $\tilde{\mathbf{x}}_j$ instead of *m*.

(iv) Transform the index of all fundamental patterns into two indices, one for the class they belong to, and another for their position in the class (i.e. $\mathbf{x}^\mu$ which belongs to class *i* becomes $\mathbf{x}^{i\omega}$).

(v) Initialize θ to 0.

(vi) If $\theta = 0$, test whether $\tilde{\mathbf{x}}$ is a fundamental pattern, by doing $\gamma_g\left(\mathbf{x}_j^\mu, \tilde{\mathbf{x}}_j, 0\right)$ for $\mu = 1,2,\ldots,p$; and then computing the initial weighted addition c_i^0 for each fundamental pattern, as follows:

$$c_\mu^0 = \sum_{j=1}^{n} \gamma_g\left(\mathbf{x}_j^\mu, \tilde{\mathbf{x}}_j, 0\right) \text{ for } \mu = 1, 2, \ldots, p \qquad (64)$$

If there is a unique maximum, whose value equals *n*, assign the class associated to such maximum to the test pattern.

$$\tilde{\mathbf{y}} = \mathbf{y}^\sigma \text{ such that } \bigvee_{i=1}^{p} c_i^0 = c_\sigma^0 = n \qquad (65)$$

(vii) Do $\gamma_g\left(\mathbf{x}_j^{i\omega}, \mathbf{y}_j, \theta\right)$ for each component of the fundamental patterns in each class, following Eq. 6.

(viii) Compute a weighted sum c_i for each class, according to Eq. 18.

(ix) If there is more than one maximum among the different c_i, increment θ by 1 and repeat steps vii and viii until there is a unique maximum, or the stop condition $\theta \geq \rho$ is fulfilled.

(x) If there is a unique maximum, assign $\tilde{\mathbf{x}}$ to the class corresponding to such maximum:

$$\tilde{\mathbf{y}} = \mathbf{y}^j \text{ such that } \bigvee c_i = c_j \qquad (66)$$

(xi) Otherwise, assign $\tilde{\mathbf{x}}$ to the class of the first maximum.

This new initial step for the classification phase ensures that $\tilde{\mathbf{x}}$ will be correctly classified at the first iteration, if it is equal to a fundamental pattern $\tilde{\mathbf{x}} = \mathbf{x}^\sigma$.

In order to illustrate the operation of the Gamma classifier version 2, let us consider an example where the version 1 would fail (i.e. be unable to correctly classify the whole fundamental set), while the version 2 exhibits a correct classification.

Let the fundamental set be made up by the following patterns:

$$\mathbf{x}^1 = \begin{pmatrix} 12 \\ 3 \end{pmatrix}, \mathbf{x}^2 = \begin{pmatrix} 15 \\ 6 \end{pmatrix}, \mathbf{x}^3 = \begin{pmatrix} 16 \\ 4 \end{pmatrix},$$

$$\mathbf{x}^4 = \begin{pmatrix} 6 \\ 6 \end{pmatrix}, \mathbf{x}^5 = \begin{pmatrix} 3 \\ 6 \end{pmatrix}, \mathbf{x}^6 = \begin{pmatrix} 8 \\ 6 \end{pmatrix}$$

grouped into two classes: $\mathbf{x}^1$, $\mathbf{x}^2$, and $\mathbf{x}^3$ are associated to $\mathbf{y}^a$ (C^1 for version 1), while $\mathbf{x}^4$, $\mathbf{x}^5$, and $\mathbf{x}^6$ are associated to $\mathbf{y}^b$ (C^2 for version 1). Now, in order to classify $\mathbf{y} = \mathbf{x}^2$

according to the Gamma classifier version 1, we do the following:

(i) Code the fundamental set with the modified Johnson-Möbius code, obtaining a value e_m for each component:

$$\mathbf{x}^1 = \begin{pmatrix} 0000111111111111 \\ 000111 \end{pmatrix}, \mathbf{x}^2 = \begin{pmatrix} 0111111111111111 \\ 111111 \end{pmatrix},$$

$$\mathbf{x}^3 = \begin{pmatrix} 1111111111111111 \\ 001111 \end{pmatrix}, \mathbf{x}^4 = \begin{pmatrix} 0000000000111111 \\ 111111 \end{pmatrix},$$

$$\mathbf{x}^5 = \begin{pmatrix} 0000000000000111 \\ 111111 \end{pmatrix}, \mathbf{x}^6 = \begin{pmatrix} 0000000011111111 \\ 111111 \end{pmatrix}$$

The maximum values for each component are 16 and 6, respectively, thus $e_m(1) = 16$ and $e_m(2) = 6$. Then, the binary vectors representing the first components are 16 bits long, while those of the second component are 6 bits long.

(ii) Compute the stop parameter according to Eq. 17:

$$\rho = \bigwedge_{j=1}^{n} e_m(j) = \bigwedge_{j=1}^{2} [16, 6] = 6 \qquad (67)$$

(iii) Code the test pattern **y** with the modified Johnson-Möbius code, using the same parameters used with the fundamental set.
Since the test pattern is also a fundamental pattern, this has already been done:

$$\mathbf{y} = \mathbf{x}^2 = \begin{pmatrix} 0111111111111111 \\ 111111 \end{pmatrix} \qquad (68)$$

(iv) Transform the index of all fundamental patterns into two indices, one for the class they belong to, and another for their position in the class.

$$\begin{aligned} &\mathbf{x}^1 = \mathbf{x}^{11}, \mathbf{x}^2 = \mathbf{x}^{12}, \mathbf{x}^3 = \mathbf{x}^{13}, \\ &\mathbf{x}^4 = \mathbf{x}^{21}, \mathbf{x}^5 = \mathbf{x}^{22}, \mathbf{x}^6 = \mathbf{x}^{23} \end{aligned} \qquad (69)$$

(v) Initialize θ to 0.

$$\theta = 0 \qquad (70)$$

(vi) Do $\gamma_g\left(\mathbf{x}_j^{i\omega}, \mathbf{y}_j, \theta\right)$ for each component of the fundamental patterns in each class, following Eq. 6:

$$\begin{aligned} &\gamma_g\left(\mathbf{x}_1^{11}, \mathbf{y}_1, 0\right) = 0, \quad \gamma_g\left(\mathbf{x}_2^{11}, \mathbf{y}_2, 0\right) = 0 \\ &\gamma_g\left(\mathbf{x}_1^{12}, \mathbf{y}_1, 0\right) = 1, \quad \gamma_g\left(\mathbf{x}_2^{12}, \mathbf{y}_2, 0\right) = 1 \\ &\gamma_g\left(\mathbf{x}_1^{13}, \mathbf{y}_1, 0\right) = 0, \quad \gamma_g\left(\mathbf{x}_2^{13}, \mathbf{y}_2, 0\right) = 0 \\ &\gamma_g\left(\mathbf{x}_1^{21}, \mathbf{y}_1^1, 0\right) = 0, \quad \gamma_g\left(\mathbf{x}_2^{21}, \mathbf{y}_2, 0\right) = 1 \\ &\gamma_g\left(\mathbf{x}_1^{22}, \mathbf{y}_1, 0\right) = 0, \quad \gamma_g\left(\mathbf{x}_2^{22}, \mathbf{y}_2, 0\right) = 1 \\ &\gamma_g\left(\mathbf{x}_1^{23}, \mathbf{y}_1, 0\right) = 0, \quad \gamma_g\left(\mathbf{x}_2^{23}, \mathbf{y}_2, 0\right) = 1 \end{aligned} \qquad (71)$$

Since $\theta = 0$, for the γ_g to give a result of 1 it is necessary that both vectors $\mathbf{x}_j^{i\omega}$ and $\mathbf{y}_j$ are equal. Given that $\mathbf{x}^2 = \mathbf{y}$, γ_g outputs 1 for both components of $\mathbf{x}^2$, but it also outputs 1 for the second component of $\mathbf{x}^4$, $\mathbf{x}^5$, and $\mathbf{x}^6$, since the second components of those fundamental patterns are equal to that of the test pattern.

(vii) Compute a weighted sum c_i for each class, according to Eq. 18:

$$\begin{aligned} c_1 &= \frac{\sum_{\omega=1}^{3} \sum_{j=1}^{2} \gamma_g\left(\mathbf{x}_j^{1\omega}, \mathbf{y}_j, 0\right)}{3} \\ &= \frac{\sum_{\omega=1}^{3} \left[(0+0), (1+1), (0+0)\right]}{3} \\ &= \frac{(0+2+0)}{3} = \frac{2}{3} \cong 0.666 \end{aligned} \qquad (72)$$

$$\begin{aligned} c_2 &= \frac{\sum_{\omega=1}^{3} \sum_{j=1}^{2} \gamma_g\left(\mathbf{x}_j^{2\omega}, \mathbf{y}_j, 0\right)}{3} \\ &= \frac{\sum_{\omega=1}^{3} \left[(0+1), (0+1), (0+1)\right]}{3} \\ &= \frac{(1+1+1)}{3} = \frac{3}{3} = 1 \end{aligned} \qquad (73)$$

(viii) If there is more than one maximum among the different c_i, increment θ by 1 and repeat steps vi and vii until there is a unique maximum, or the stop condition $\theta \geq \rho$ is fulfilled.
There is a unique maximum, so we go directly to step ix.

(ix) If there is a unique maximum, assign **y** to the class corresponding to such maximum:

$$C_y = C_2 \text{ since } \bigvee_{i=1}^{2} c_i = \bigvee (0.666, 1) = 1 = c_2 \qquad (74)$$

(x) Otherwise, assign **y** to the class of the first maximum.

This is unnecessary since the test pattern has already been assigned a class.

Notice, however, that the result is incorrect: we already know that $\mathbf{y} = \mathbf{x}^2$, thus the test pattern should receive the same class that $\mathbf{x}^2$, which is C^1; yet, $\mathbf{y} = \mathbf{x}^2$ was classified as belonging to C^2. Let us now try using the second version to classify $\tilde{\mathbf{x}} = \mathbf{x}^2$.

(i) Code the fundamental set with the modified Johnson-Möbius code, obtaining a value e_m for each component. This e_m value is calculated as defined by Eq. 16.

$$\mathbf{x}^1 = \begin{pmatrix} 0000111111111111 \\ 000111 \end{pmatrix}, \mathbf{x}^2 = \begin{pmatrix} 0111111111111111 \\ 111111 \end{pmatrix},$$
$$\mathbf{x}^3 = \begin{pmatrix} 1111111111111111 \\ 001111 \end{pmatrix}, \mathbf{x}^4 = \begin{pmatrix} 0000000000111111 \\ 111111 \end{pmatrix},$$
$$\mathbf{x}^5 = \begin{pmatrix} 0000000000000111 \\ 111111 \end{pmatrix}, \mathbf{x}^6 = \begin{pmatrix} 0000000011111111 \\ 111111 \end{pmatrix}$$

This is equal to what was done on version 1, thus $e_m(1) = 16$ and $e_m(2) = 6$.

(ii) Compute the stop parameter (Eq. 17):

$$\rho = \bigwedge_{j=1}^{n} e_m(j) = \bigwedge_{j=1}^{2} [16, 6] = 6 \tag{75}$$

(iii) Code $\tilde{\mathbf{x}}$ with the modified Johnson-Möbius code, using the same parameters as in step i. If any $\tilde{\mathbf{x}}_j$ is greater than the corresponding $e_m(j)$, the γ_g operator will use such $\tilde{\mathbf{x}}_j$ instead of m.
Since the test pattern is also a fundamental pattern, this has already been done (like in version 1):

$$\tilde{\mathbf{x}} = \mathbf{x}^2 = \begin{pmatrix} 0111111111111111 \\ 111111 \end{pmatrix} \tag{76}$$

(iv) Transform the index of all fundamental patterns into two indices, one for the class they belong to, and another for their position in the class.

$$\begin{gathered} \mathbf{x}^1 = \mathbf{x}^{11}, \mathbf{x}^2 = \mathbf{x}^{12}, \mathbf{x}^3 = \mathbf{x}^{13}, \\ \mathbf{x}^4 = \mathbf{x}^{21}, \mathbf{x}^5 = \mathbf{x}^{22}, \mathbf{x}^6 = \mathbf{x}^{23} \end{gathered} \tag{77}$$

(v) Initialize θ to 0.

$$\theta = 0 \tag{78}$$

(vi) If $\theta = 0$, test whether $\tilde{\mathbf{x}}$ is a fundamental pattern, by doing $\gamma_g\left(\mathbf{x}_j^{\mu}, \tilde{\mathbf{x}}_j, 0\right)$ for $\mu = 1,2,\ldots,p$; and then computing the initial weighted addition c_i^0 for each fundamental pattern, as indicated by Eq. 64. If there is a unique maximum, whose value equals n, assign the class associated to such maximum to the test pattern:

$$\begin{aligned} &\gamma_g\left(\mathbf{x}_1^1, \tilde{\mathbf{x}}_1, 0\right) = 0, \quad \gamma_g\left(\mathbf{x}_1^1, \tilde{\mathbf{x}}_2, 0\right) = 0 \\ &\gamma_g\left(\mathbf{x}_1^2, \tilde{\mathbf{x}}_1, 0\right) = 1, \quad \gamma_g\left(\mathbf{x}_2^2, \tilde{\mathbf{x}}_2, 0\right) = 1 \\ &\gamma_g\left(\mathbf{x}_1^3, \tilde{\mathbf{x}}_1, 0\right) = 0, \quad \gamma_g\left(\mathbf{x}_2^3, \tilde{\mathbf{x}}_2, 0\right) = 0 \\ &\gamma_g\left(\mathbf{x}_1^4, \tilde{\mathbf{x}}_1^1, 0\right) = 0, \quad \gamma_g\left(\mathbf{x}_2^4, \tilde{\mathbf{x}}_2, 0\right) = 1 \\ &\gamma_g\left(\mathbf{x}_1^5, \tilde{\mathbf{x}}_1, 0\right) = 0, \quad \gamma_g\left(\mathbf{x}_2^5, \tilde{\mathbf{x}}_2, 0\right) = 1 \\ &\gamma_g\left(\mathbf{x}_1^6, \tilde{\mathbf{x}}_1, 0\right) = 0, \quad \gamma_g\left(\mathbf{x}_2^6, \tilde{\mathbf{x}}_2, 0\right) = 1 \end{aligned} \tag{79}$$

$$\begin{aligned} c_1^0 &= \sum_{j=1}^{2} \gamma_g\left(\mathbf{x}_j^1, \mathbf{y}_j, 0\right) = 0 + 0 = 0 \\ c_2^0 &= \sum_{j=1}^{2} \gamma_g\left(\mathbf{x}_j^2, \mathbf{y}_j, 0\right) = 1 + 1 = 2 \\ c_3^0 &= \sum_{j=1}^{2} \gamma_g\left(\mathbf{x}_j^3, \mathbf{y}_j, 0\right) = 0 + 0 = 0 \\ c_4^0 &= \sum_{j=1}^{2} \gamma_g\left(\mathbf{x}_j^4, \mathbf{y}_j, 0\right) = 0 + 1 = 1 \\ c_5^0 &= \sum_{j=1}^{2} \gamma_g\left(\mathbf{x}_j^5, \mathbf{y}_j, 0\right) = 0 + 1 = 1 \\ c_6^0 &= \sum_{j=1}^{2} \gamma_g\left(\mathbf{x}_j^6, \mathbf{y}_j, 0\right) = 0 + 1 = 1 \end{aligned} \tag{80}$$

In this step, the weighted additions are computed but considering each fundamental pattern on a class by itself. As we can see, only $\tilde{\mathbf{x}} = \mathbf{x}^2$ gives a sum of 2. Since this is the maximum value, it is unique, and also equals n, that class is assigned to the test pattern:

$$\tilde{\mathbf{y}} = \mathbf{y}^2 = \mathbf{y}^a \text{ since } \bigvee_{i=1}^{6} c_i^0 = c_2^0 = 2 = n \tag{81}$$

(vii) Do $\gamma_g\left(\mathbf{x}_j^{i\omega}, \mathbf{y}_j, \theta\right)$ for each component of the fundamental patterns in each class, following Eq. 6.
This is unnecessary since the test pattern has already been assigned a class.

(viii) Compute a weighted sum c_i for each class, according to Eq. 18.
This is unnecessary since the test pattern has already been assigned a class.

(ix) If there is more than one maximum among the different c_i, increment θ by 1 and repeat steps vii and viii until there is a unique maximum, or the stop condition $\theta \geq \rho$ is fulfilled.

This is unnecessary since the test pattern has already been assigned a class.

(x) If there is a unique maximum, assign $\tilde{\mathbf{x}}$ to the class corresponding to such maximum:
This is unnecessary since the test pattern has already been assigned a class.

(xi) Otherwise, assign $\tilde{\mathbf{x}}$ to the class of the first maximum.
This is unnecessary since the test pattern has already been assigned a class.

As can be seen, the new version of the classification algorithm offers an improved performance, classifying correctly all known patterns.

7. Second Generation of Experiments

Taking into account the improvements introduced with the second version of the Gamma classifier, as well as the modifications to the method of applying it to the current problem explained in Sec. 6, a new set of experiments was run. This is the 2G, which also follows the experimental design described in Sec. 4.

In the following subsection, the experiments run are described. Then, the results obtained are presented and discussed, and later the performance shown is compared with that exhibited by other methods.

7.1. *Second generation: Experiments done*

For this new generation of experimental runs, the fundamental set was built using data taken from two stations for each one of four pollutants, for the year 2010. On the other hand, the test set was built with data taken from one month of another year: one experiment instance with January 2011 and another with data from February 2011, for each station and each pollutant. The details of each experimental run can be seen in Table 3. For the classification step, the Gamma classifier version 2 was used.

7.2. *Second generation: Results and discussion*

The results obtained on the 2G (experiments 13 to 28) are shown in Table 7. Again, each pollutant exhibits similar performance through most experiments, in different ranges for each pollutant. The experiments which show better results on each pollutant, for bias, are: 16 for CO (–3.0 ppm), 20 for O_3 (0.289 ppm), 22 for SO_2 (0.183 ppm), and 28 for NO_x (0.843 ppm). Regarding RMSE, the best performers are: 13 on CO (0.6042 ppm), 19 on O_3 (0.0095 ppm), 21 on SO_2 (0.0076 ppm), and 28 on NO_x (0.0343 ppm). On the other hand, NFit presents the best results for each pollutant in the following experiments: 16 for CO (8.9029 ppm), 18 for O_3 (11.2557 ppm), 24 for SO_2 (10.6560 ppm), and 26 for NO_x (27.4858 ppm). The former is indicated in table 7 by the results in bold.

Table 7. Experimental results of the second generation (ppm).

Experiment	Pollutant	Bias	RMSE	NFit
13	CO	19.800	**0.604214**	17.414193
14	CO	-26.100	0.653224	16.960662
15	CO	-15.500	0.870378	18.825927
16	CO	**-3.000**	0.637459	**8.902862**
17	O_3	1.233	0.010294	19.000468
18	O_3	0.546	0.009803	**11.255713**
19	O_3	0.678	**0.009504**	41.206207
20	O_3	**0.289**	0.011565	18.720893
21	SO_2	0.234	**0.007617**	14.356505
22	SO_2	**0.183**	0.008495	18.383810
23	SO_2	0.361	0.009618	73.880135
24	SO_2	0.317	0.010164	**10.655986**
25	NO_x	1.391	0.044623	53.588005
26	NO_x	1.588	0.045032	**27.485791**
27	NO_x	1.075	0.038326	164.569051
28	NO_x	**0.843**	**0.034269**	1033.516665

It bears mention that there is not one instance for which all three measures of performance agree on the best performer: although experiment 16 shows the minimum bias and NFit among CO, it does worse than experiment 13 on RMSE. The case of experiment 28 is even more dramatic: it is the best result for NO_x when evaluated with bias or RMSE, but it is the worst on NFit… and by a very long shot! These differences are due to what each measure indicates, as well as their focus and limitations, as explained in subsections 4.2 and 5.2.

One other aspect of interest is that none of these new experiments shows such good performance as experiments 10, 11, and 12 from the 1G, which lead us to suspect that the cause for such good results lies in the data used for said experiments.

On the other hand, by comparing the results obtained during this 2G against those of the 1G, a slight improvement can be seen. Although the bias for all experiments 13 to 28 are larger than those of experiments 1 to 12, both the mean RMSE and mean NFit for CO, O_3, and SO_2 is better on experiments 13 to 28 than on 1 to 12. The data on NO_x is suspicious, but comparing the performance of experiment 9 to those of experiments 25 to 28, the NFit is better on experiments 25 and 26, while experiment 9 shows better RMSE.

Regarding this slight improvement, let us recall that improvements introduced by the Gamma classifier version 2 and the modified method affect the result of some of the test pattern, not all of them. The effects introduced by the method of the 2G, compared against the result that would have been obtained if the original method (used on the 1G) was applied, can be seen in table 8.

Table 8. Effects of modified method of the 2G; Exp. means "Experiment".

	Test Patterns Affected		Performance Difference (ppm)		
Exp.	**Amount**	**Percent**	**Bias**	**RMSE**	**NFit**
13	6	0.81	-1.800	0.004140	0.022021
14	3	0.45	-1.600	0.006820	0.410543
15	6	0.81	-2.700	0.005695	0.059208
16	2	0.30	-0.800	0.002345	0.016599
17	48	6.44	-0.254	0.000323	0.034240
18	28	4.16	-0.092	0.000081	0.003244
19	41	5.50	-0.292	0.000569	-30.287696
20	31	4.61	-0.125	0.000175	4.977420
21	34	4.56	-0.031	0.000053	-1.986110
22	45	6.69	-0.038	0.000228	-10.205255
23	90	12.08	-0.054	0.000554	-65.539397
24	124	18.42	-0.020	0.001077	289.424768
25	18	2.42	-0.546	0.000887	1.979086
26	22	3.27	-0.210	0.000409	53.801941
27	7	0.94	-0.126	0.000208	0.967188
28	12	1.78	-0.246	0.000482	0.318623

As can be seen, the effect of the modified method is not very large, yet not negligible. Experiment 16 has the least amount of patterns affected, only 2 out of 673 (0.3%), but the difference in performance when compared to the results given by the original method on the same data is not the minimum, for either bias, RMSE or NFit. On the other hand, the experiment with most patterns affected by the variations to the method is 24, with 124 out of 673 patterns (18.42%). Again, this is not the experiment with the greatest difference in bias or RMSE, but it has the largest difference on NFit: 289.42.

Again, the differences tend to group together in different ranges for different pollutants, although this tendency is not as clear-cut as the actual values of performance.

Regarding bias, it is always larger (both in magnitude and sign) for the modified method, indicating that this method tends to underestimate less and overestimate more than the method used on the 1G. Notice also that all values of RMSE are better for the improved method, again in different ranges for each pollutant. In particular, the mean difference in RMSE for CO is 0.004750 ppm, while that of O_3, SO_2, and NO_x are, respectively: 0.000287 ppm, 0.000478 ppm, and 0.000496 ppm.

The behavior of NFit is different. While it improves for 12 out of 16 experiments, the magnitude of most of those 12 results are quite small: the average NFit difference for those experiments where it improved is 28.25 ppm, while the average for the experiments where the NFit was better with the original method is -27.00 ppm; their difference is barely 1.25, even when the NFit for experiment 24 was an improvement of 289.4248.

Looking at the considerable variations on the impact that the modified method has on performance, it is clear that new improvements and extension of the method are called for.

7.3. *Second generation: Comparison with other methods*

Again, a comparison against other methods is in order. Thus, table 9 shows the experimental results obtained with data taken from the same database (RAMA subsystem of SIMAT) for O_3, while table 10 includes results obtained with data taken from diverse databases.

Table 9. Comparison of related results (SIMAT database, O_3) in absolute mean error given for pollutant concentration in ppm; exp. means "experiment".

Algorithm Used	Performance (Abs. Mean Error)
Bayesian network[10]	0.221000
C4.5[10]	0.176400
Neural network[10]	0.160000
Gamma classifier exp. 4 (current work & Ref. 18)	0.000918
Gamma classifier exp. 5 (current work & Ref. 18)	0.000417
Gamma classifier exp. 17	0.001682
Gamma classifier exp. 18	0.000826
Gamma classifier exp. 19	0.000925
Gamma classifier exp. 20	0.000437

Table 10. Comparison of related results (diverse databases) in RMSE; exp. means "experiment".

Algorithm Used	**Pollutant Considered (measurement unit)**	**Performance RMSE (original unit)**	**Performance RMSE (μg/m^3)**
IITLS[15]	NO_x (μg/m^3)	19.99	19.99
Neural network[13]	O_3 (μg/m^3)	15	15
Neural network[12]	O_3 (ppb)	13.79, 9.43	27.028, 18.483
Online SVM[14]	SO_2 (μg/m^3)	12.96, 10.90	12.96, 10.90
Neural network[11]	CO, NO_2, SO_2, and O_3 (ERPI)	5.852, 1.365	N/A
Gamma classifier exp. 3 (current work & Ref. 18)	CO (ppm)	0.611769	700.476
Multivariate linear regression[9]	BaP (ng/m^3)	0.449	0.000449
Gamma classifier exp. 4 (current work & Ref. 18)	O_3 (ppm)	0.012302	24.112
Gamma classifier exp. 8 (current work & Ref. 18)	SO_2 (ppm)	0.009218	24.151
Gamma classifier exp. 10	NO (ppm)	0.000037	0.069560
Gamma classifier exp. 11	NO_2 (ppm)	0.000000	0.000000
Gamma classifier exp. 12	NO_x (ppm)	0.000000	0.000000
Gamma classifier exp. 13	CO (ppm)	0.604214	691.825481
Gamma classifier exp. 19	O_3 (ppm)	0.009504	18.627704
Gamma classifier exp. 21	SO_2 (ppm)	0.007617	19.957555
Gamma classifier exp. 28	NO_x (ppm)	0.034269	64.426059

When compared to the results exhibited by other methods when working with data taken from the SIMAT database, the performance shown by the Gamma classifier is better on experiments of both 1G and 2G.

In relation to the results obtained on data from other databases, the performance of the proposed method is mixed. For O_3, the results are competitive: experiment 19 has an RMSE of 18.6277 μg/m^3, which is better than the worst performance (ANN from Ref. 12 with 27.028 μg/m^3), but not as good as the best one (ANN from Ref. 12 with 18.483 μg/m^3).

On SO_2, the results are not so good: experiment 21 exhibits an RMSE of 19.9576 μg/m^3, which trails behind the Online SVM of Ref. 14 with 10.90 μg/m^3 and 12.96 μg/m^3. The situation of NO_x is similar: experiment 28 presents an RMSE of 64.4261 μg/m^3, while the IITLS of Ref. 15 has 19.99 μg/m^3 of RMSE.

CO is harder to compare, since the ANN of Ref. 11 takes as input data regarding CO, NO_2, SO_2, and O_3, but it is not possible to separate them at the output, which is also in a different unit: ERPI. Thus, it is not clear just how good the results obtained by the proposed method for CO are (experiment 3 from 1G with 70.4760 μg/m^3, and experiment 13 from 2G with 691.8255 μg/m^3). The former claim is especially true when considering that concentration values for this pollutant are usually several orders of magnitude greater than those for the other pollutants, which may make such apparently large values of RMSE actually quite competitive.

These comparisons exhibit some emerging traits of the proposed method. First, it seems to show a better performance on O_3 than on the other pollutants so far considered, when only past values for the concentration of the target pollutant are used. On the other hand, the results on other pollutants are not as good, but still competitive.

On the topic of the extremely (and surprisingly) good results of experiments 10 to 12, the fact that other experiments done with the same method (versions 1 and 2) on different data offer more conventional results, leads to the suspicion that the data used for those three experiments is abnormally good. Two interesting questions left for future work, which arise from the latter observation, are the following: on one hand, would other methods perform as well? And what makes these data so appropriate to work with the proposed method?

8. Analysis and Comparison

In order to compare the performance of the Gamma classifier against other methods of pollutant prediction, the results of seven research papers (Refs. 9–15) have been used. In those works, the authors have applied the following methods to predict the concentration of several contaminants: MLR,[9] Bayesian networks and decision trees,[10] ANN,[10-13] SVM,[14] and the pollution models Caline and IITLS.[15] Regarding these comparisons, it is necessary to emphasize two considerations.

- Consideration 1. There is no universally best learning algorithm, which implies that there is no classification or prediction algorithm which is *the best* for all and every situation. Such a strong claim is based on a series of theorems, known as the No-Free-Lunch Theorems, which establish that for any algorithm, any elevated performance over one class of problems is offset by performance over another class.[33] In this sense, Duda and Hart refer to the No-Free-Lunch Theorems for Pattern Recognition in the following terms: "If an algorithm seems to surpass another on a particular situation, this is a consequence of its adaptation to the particular pattern recognition problem, and not to a general superiority of that algorithm".[25]
- Consideration 2. When comparing the performance of two prediction or classification models, the ideal situation is to do such comparison under the same conditions and using the same measure criteria. In other words, we would like to have both algorithms trained with the same data set, as well as being tested on the same test pattern set, and using the same criteria of performance evaluation.

However, in the topic of environmental data prediction, it is quite difficult —if not altogether impossible— to obtain scientific publications on different models where the same data sets have been used for training, on one hand, and the same data sets were used for testing, on the other hand, while also sharing the same measure criteria for performance. This is due to the lack of standardized, publicly accessible data sets.

When these considerations are taken into account, it is possible to state that there are no founded reasons to claim that any one method is inherently better than the rest for all cases. Thus, if for some data sets the Gamma classifier exhibits better performance than the other methods, this can be explained by a better adaptation of that method to the specific data sets used. However, it is also important to notice that, for some experiments, the Gamma classifier shows large errors: such is the case of experiment 3, where the RMSE is quite high.

Then, we may claim that it is not only the method by itself or the data sets by themselves, which explain a good performance: it is the combination of a good method and an appropriate data set which explains a better performance.

On the other hand, we can see that the conditions presented in the second consideration are very difficult to meet, which leads us to look for publications whose experiments use the same measure criteria (e.g. RMSE, NFit, or Absolute Mean Error), even though the prediction is done on different pollutants for different places around the world; that is, even when the data sets used for learning and training are different from paper to paper.

This is exactly what happened in the present work: fixed unit and measure criteria were chosen, all data was converted into those units of measure (where possible), and the results were compared using the measure criteria picked at the beginning.

On a different but related train of thought, it is a common practice for the authors of scientific works on pollutants prediction to not describe in detail the methods they use for their predictions, referring the reader to other publications where those details are described.

Even though the papers from Refs. 9–15 are no exception to that practice, below an analysis of the prediction mechanisms used in those papers is done, as far as possible. That is, the text of those seven papers is taken as the basis to try and explain the nature of each consulted method for pollution prediction, showing their advantages and drawbacks. This is done in order to sort out the reasons behind the robustness and good performance shown experimentally by the Gamma classifier, when compared to these methods.

- **Ref. 9**: In this work the authors use MLR models to predict benzo(a)pyrene (BaP) air concentrations in two sampling places, taking PM_{10} and meteorological variables as possible predictors. It is worth to note that BaP is a complex chemical, belonging to the family of Polycyclic Aromatic Hydrocarbons (PAH), which is released widespread into the air, and that is emitted as product of fuel thermal processes and is released from anthropogenic activities involving the devolatilization of coal, oil, wood, diesel, and gasoline.
  The authors explain that the meteorological variables recorded for each sampling place are: temperature (◦C), relative humidity (%), solar radiation (W/m^2), UV radiation (W/m^2), pressure (mbar), rainfall (mm), wind speed (m/s) and season. However, they do not give details about the method employed, declaring that they used the SPSS version 15.0 statistical package MLR models to run MLR models in order to find the best-fit model between the estimated and the experimental BaP

concentrations according to PM_{10} concentrations and meteorological variables.

- Advantages: the model is as simple as a linear equation with one dependent variable and several independent variables. Also, in order to find the model which fits best a specific problem, it is enough to feed a commercial statistical package such as SPSS version 15.0 and pick the best model.
- Drawbacks: the main disadvantage is that the authors pretend to model a non-linear phenomenon such as pollution with linear dependencies, which in the simplest case equals to representing a straight line on a plane.

- **Ref. 10**: In this work Bayesian networks, decision trees (namely C4.5) and ANN are used for ozone prediction in Mexico City, where the ozone level is used as a global indicator for the air quality in different parts of the city. According to the authors, it is important to predict the ozone level a day, or at least several hours in advance, to reduce the health hazards and industrial losses that occur when the ozone reaches emergency levels.

  The main contribution of this paper is an algorithm for structure learning in predictive expert systems based on a Bayesian network representation. The goal is to find the "simplest" structure (minimum number of links) with acceptable predictive capability. The algorithm starts by building a tree structure based on measuring mutual information between pairs of variables, and then it adds links as necessary to obtain a certain predictive performance.

  - Advantages: the proposed model is a Bayesian network with the "simplest" structure (minimum number of links) with acceptable predictive capability. Also, a well-known model is used for comparison purposes: the C4.5 decision tree, which may be easily run on the WEKA environment.[34] The Bayesian network surpasses the C4.5 tree.
  - Drawbacks: by looking for the "simplest" structure for the Bayesian network, the authors are not interested on improved predictive results, but rather to have an acceptable predictive capability. On the other hand, although the proposed model beats the ANN results, no description of the ANN model is given, instead referring to another paper. Independently from the latter, it is well-known by the scientific community that ANN have (at least) two disadvantages: (A) the network may converge to local minima, which are not the correct output values; and (B) the network topology design is artisanal, and given that usually the Backpropagation algorithm is used for training, which in turn is an iterative process, it becomes quite difficult, almost impossible, to know beforehand whether the network will converge. If it does not converge, after investing considerable amounts of time and effort, one needs to start over a new topology, which also may not converge… and so on.

- **Ref. 11**: After acknowledging the difficulty in forecasting concentration trends with a reasonable error and claiming it is still an open problem, the authors applied ANN in order to forecast the maximum daily value of the European Regional Pollution Index (ERPI) as well as the number of consecutive hours, during the day, with at least one of the pollutants above a threshold concentration, 24 to 72 hours ahead. The concentrations of four pollutants (NO_2, SO_2, CO, and O_3) play a part in the ERPI computing process, thus making more difficult a comparison of other indices against ERPI.

  Although experimental results with data taken from seven monitoring stations are presented, for comparison purposes we have used only the results of the station which exhibits the best RMSE performance, as reported in Ref. 11.

  - Advantages: even though there is no mention of how the ANN was implemented, it is well-known that there are several applications (both commercial and open source) for designing, programming, training, and testing feedforward MultiLayer Perceptron (MLP) ANN, which is the model used on this work.
  - Drawbacks: the ANN presents the same disadvantages of Ref. 10 regarding convergence and local minima. As discussed earlier, the main disadvantage for comparison purposes is that the unit used, ERPI, combines input data regarding CO, NO_2, SO_2, and O_3, taking into account only the pollutant with the highest concentration for the output. Thus, it is not possible to convert ERPI into $\mu g/m^3$, since it is not known which pollutant gave a particular value, and thus which particular equivalence use for the conversion. Therefore, it is not clear just how good the results obtained by the proposed method for CO are

(experiment 3 from the 1G with 70.4760 μg/m^3, and experiment 13 from the 2G with 691.8255 μg/m^3).

- **Ref. 12**: The authors apply the same kind of ANN that was used in Ref. 11 —feedforward MLP ANN— to the prediction of maximum tropospheric ozone concentrations for the next day. For this, the main ozone precursors and meteorological parameters were used.
  - Advantages: these are similar to those of Ref. 11, there are several open source and commercial applications for designing, programming, training, and testing feedforward MLP ANN. The authors also omit details on the implementation of the neural networks. However, they mention during their discussion that this kind of training (of MLP) is relatively easy.
  - Drawbacks: the ANN has the same disadvantages of Refs. 10 and 11 regarding convergence and local minima.
- **Ref. 13**: This work presents a model called NEUROZONE, which is based on the MLP. The Neurozone is used in a real time forecasting of hourly maximum ozone, for which it is fed with the values of 34 meteorological variables.
  - Advantages: one advantage is similar to that of Refs. 11 and 12, regarding the availability of commercial and open source applications for designing, programming, training, and testing feedforward MLP ANN; another one is that Neurozone is used for real time prediction.
  - Drawbacks: the ANN presents the same disadvantages of Refs. 10–12 on convergence and local minima.
- **Ref. 14**: The authors of this paper claim that the development of an accurate model for the time series forecasting problems is very difficult, because of high non-linear and non-stable relations between input and output data. Given the importance of online prediction for air quality parameters forecasting, the authors propose an online SVM to predict air pollutant levels in an advancing time series based on the monitored air pollutant database.
  - Advantages: according to the authors, the SVM have a solid theoretical basis, which ensures its possessing more salient advantages than other machine learning methods like ANN in generalization and convergence. Also, the SVM model they propose operates online.
  - Drawbacks: as the authors themselves admit, the greatest disadvantage of this model is that the computational problem with the numerical optimization in a high-dimensional space may suffer from the curse of dimensionality. In other words, their proposal may be computing-intensive and of high complexity, depending on the data used.
- **Ref. 15**: In this paper, two line source models are uses: the California line source version 4 (Caline) and the Indian Institute of Technology Line Source (IITLS), to assess the impact of diesel vehicles on NO_x and PM emissions. The methods used are not described, only their names are mentioned and two references are given. As happened with Ref. 11, Ref. 15 reports results for six monitoring stations, but in this work only data from the station with the best RMSE was used for comparison.
  - Advantages: the paper includes no useful data regarding the advantages of Caline and IITLS.
  - Drawbacks: the paper includes no useful data regarding the disadvantages of Caline and IITLS.

The latter point concludes the explanations on the nature of each consulted method for pollution prediction, showing their advantages and drawbacks. This information shall be of great help to express the possible reasons for the outstanding robustness and performance shown by the Gamma classifier on some experiments presented in this paper, while comparing such results against those of other methods.

It is important to realize that the Gamma classifier is not the best performer on all experiments: in some cases, its performance was actually not good. For instance, in experiment 19 (O_3) the Gamma Classifier has an RMSE of 18.6277 μg/m^3, which is not as good as the best one (Neural Network from Ref. 12 with 18.4830 μg/m^3). On SO_2, the results are even worse: experiment 21 exhibits an RMSE of 19.9576 μg/m^3, which trails behind the Online SVM of Ref. 14 with 10.90 μg/m^3 and 12.96 μg/m^3. The situation of NO_x is similar: experiment 28 presents an RMSE of 64.4261 μg/m^3, while the IITLS of Ref. 15 has 19.99 μg/m^3 of RMSE. These results agree with the No-Free-Lunch Theorems.

However, it is still true that the Gamma classifier performance was quite good on most of the experiments. Then, how can such good performance with respect to other methods be explained?

Below some potential answers to such question are discussed. These possible reasons for the behavior of the Gamma classifier are taken mainly from its inherent nature and properties, as well as from the information posed by the previous qualitative analysis.

(i) The Gamma classifier algorithm models pollution phenomena as a non-linear process (in Ref. 14 it is even claimed that such phenomena are highly non-linear), instead of forcing them to a false linearity, such as what happens in Ref. 9. For instance, see Fig. 4, which illustrates the results of experiment 12: the values predicted by the Gamma classifier hide behind the originally observed values.

(ii) In each experiment, the Gamma classifier treats pollution phenomena as real, actual time series. That is, only one independent variable (time) and one dependent variable (the pollutant concentration at that time) are considered. On the other hand, most methods try to do multivariate regression, leading to situations such as contemplating 34 independent variables.[13]

(iii) Derived from the latter item, given that the Gamma classifier only takes into account one independent variable, it has no problems regarding correlation among variables, or any of the problems introduced by the probability distribution characteristics of multiple independent variables.

(iv) The Gamma classifier algorithm suffers no convergence or local minima problems, shown by the ANN models,[10–13] which leads them to reduced performance (both in predictive accuracy and time).

(v) The Gamma classifier algorithm lacks the drawbacks of Bayesian networks or decision trees.[10] On one hand, the correlation of independent variables is one source of diminished performance for Bayesian networks, but it has no effect on the Gamma classifier since it considers only one independent variable. On the other hand, the C4.5 decision tree exhibits a poor ability to solve the multi-class problems.[35] There are two drawbacks of Quinlan's C4.5 when dealing with multi-class problems: the training speed will become slow and the classification accuracy will decrease acutely. Meanwhile, the Gamma classifier was designed to tackle the multi-class problem from the beginning, presenting no such drawbacks.

(vi) The Gamma classifier algorithm is not affected by the curse of dimensionality as the SVM does.[14] This latter model exhibits the computational problem with the numerical optimization in a high-dimensional space.

(vii) Regarding Ref. 15, we have no founded elements to put forward an informed opinion. We have simply shown through experimental results that the Gamma classifier defeats Caline and IITLS for pollutants prediction, at least for some cases on NO_x.

We believe that these seven reasons show the good performance of our method related to the other existing time series prediction methods; that is, we have clearly specified why the Gamma classifier is better than the other time series prediction methods.

However, is perhaps the Gamma classifier free of drawbacks or limitations? Of course not, this model also has disadvantages, some of which appear when it is compared to those methods present in Refs. 9–15, as shown below.

(i) The Gamma classifier does not operate in real time, like the method of Ref. 13.

(ii) The Gamma classifier does not operate online, something that the method of Ref. 14 does.

(iii) Given the nature of the Gamma classifier algorithm, it is unable to predict pollutant concentration values which are absent from the fundamental set. This means that, when a target pollutant takes a concentration value unknown by the Gamma classifier, there will be a prediction error, even if the unknown value is quite similar to a known value. On the other hand, most of the other models can do this: for instance the MLR model[9] is able to predict values which are not present in the learning set.

(iv) Like any other pollutant prediction model, the Gamma classifier cannot predict dramatic changes in contaminant concentration due to unforeseen events, such as a volcanic eruption, an uncontrolled fire, or an explosion. This point is also discussed in subsection 5.2.

Yet, an issue remains: how is it that in experiments 11 and 12, the RMSE is zero? In order to answer this question let us consider the concept of robustness presented by Rosenhead, which is one of the most important experts in this topic: "A simple statement of the robustness criterion is that, other things being equal, an initial commitment should be preferred if the proportion of desirable future situations that can still be reached once that decision has been implemented is high".[36] The same author tells us which are the elements that must be specified in a problem situation for robustness analysis:

- a set of alternative initial commitments to be considered,
- a set of 'futures' representative of possible environments of the system, and
- a set of relevant possible configurations of the system which the decisions will modify.

The three elements above need to be complemented by information of the following types:

- assessments of the compatibility of each commitment-configuration pair,
- evaluation of the performance of each configuration in each future.

Following Rosenhead, we will start from the results on one experiment described in our paper, which are quite good but not as good as those of experiments 11 and 12: experiment 10 has a very low RMSE, yet it is not cero. Then, why experiment 10 has so small RMSE? What happened is that the Gamma classifier only misclassifies *one* data point when predicting each concentration value. The prediction is done by finding in the fundamental set, the nearest pattern to the test pattern (made by the previous ten data points to the data point of May 4th at 4:00 AM), according to the Gamma classifier criteria, and taking the next concentration value as the prediction. This error consists in that the concentration value given by the fundamental pattern (e.g. the corresponding class), which is the prediction, is 0.006 ppm, while the actual concentration for that data point was 0.005 ppm. On one hand, the test pattern is built with the following sequence of differences (see the pattern coding technique at subsection 4.1 from the present paper), with its corresponding class being:

$$\tilde{\mathbf{x}} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ -1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \tilde{\mathbf{y}} = 0.005 \qquad (82)$$

Nevertheless, the fundamental pattern responsible for the classification of this test pattern and the corresponding class are:

$$\mathbf{x}^{7689} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 2 \end{pmatrix}, \mathbf{y}^{7689} = 0.006 \qquad (83)$$

Based on the previous explanation, it becomes clearer why the RMSE for experiments 11 and 12 is zero. Such miss-classification as the one presented above does not happen for any data point in the test data sets of experiments 11 and 12.

This discussion illustrates the robustness of the Gamma classifier algorithm, according to the criteria proposed by Rosenhead.

9. Conclusions and Future Work

Data related to atmospheric pollution in Mexico City can be worked as time series, thanks to its periodicity. In the present work, the air quality data measured at Mexico City was used to build several time series. The Gamma classifier —versions 1 and 2— combined with an emerging coding technique has been used to predict future values of the time series, thus forecasting the concentration levels of four criteria pollutants: carbon monoxide (CO), ozone (O_3), sulfur dioxide (SO_2), and nitrogen oxides (NO_x). The experimental results show a competitive performance by the Gamma classifier as a predictor of pollutant concentrations.

In order to apply the proposed method to different data, such as other pollutants or time intervals, new fundamental and test pattern sets should be built and presented to the method.

Notice that it is not possible to make a direct comparison of published results, given that different data sets, describing different local phenomena with different measure units, are used. This is mainly due to the lack of a standardized, publicly accessible database regarding this particular problem.

A new version of the Gamma classifier algorithm is introduced in this paper, along with its theoretical foundation.

From these results, several lines of future work can be drawn:

- Implement other methods found in related scientific literature and test them on the same data used by the Gamma classifier, in order to compare all methods under the same methodological conditions.
- Application of the Gamma classifier and the proposed coding technique to the analysis and prediction of time series built with data taken from other air quality databases, and comparison of results with those present in scientific literature.
- Development of variations to the coding technique which may yield improved performance.
- Experimentation with other pollutants, and combinations of pollutants.
- Extension of the proposed method to incorporate more data besides past concentration values of the target pollutant, such as meteorological data, concentration values of other pollutants, or concentration values at other stations, either of the target pollutant or other pollutants.
- Application of the Gamma classifier and the proposed coding technique to other areas, such as water quality assessment.[37]

Acknowledgements

The authors would like to thank the Instituto Politécnico Nacional (Secretaría Académica, COFAA, SIP, and CIC), the CONACyT, SNI, and the ICyTDF (grants PIUTE10-77 and PICSO10-85) for their economical support to develop this work.

References

1. B. Schelter, M. Winterhalder and J. Timmer (eds.), *Handbook of Time Series Analysis*, (Wiley, Weinheim, Germany, 2006).
2. A.K. Palit and D. Popovic, *Computational Intelligence in Time Series Forecasting* (Springer-Verlag, London, U. K, 2005).
3. D.S.G. Pollock, *A Handbook of Time-Series Analysis, Signal Processing and Dynamics* (Academic Press, London, U. K., 1999).
4. D. Moreira, and M. Vilhena, *Air Pollution and Turbulence: Modeling and Applications* (CRC Press, Boca Raton, FL, USA, 2010) ix–x.
5. Y. Shi, X. Liu, T. Li, X. Peng, W. Chen, R. Zhang and Y. Fu, Chaotic Time Series Prediction Using Immune Optimization Theory, *International Journal of Computational Intelligence Systems*, **3**(Supplement 1) (2010) 43–60.
6. A. Bouchachia, Radial Basis Function Nets for Time Series Prediction, *International Journal of Computational Intelligence Systems*, **2**(2) (2009) 147–157.
7. O. Bekir and G. Surhid, *Vehicular Air Pollution: Experiences From Seven Latin American Urban Centers*. (World Bank, Washington, D.C, USA, 1997) xiii–xiv.
8. Sistema de Monitoreo Atmosférico de la Ciudad de México, *SIMAT* (in Spanish) (2010). Available at http://www.sma.df.gob.mx/simat2 (accesed September 12, 2010).
9. M.S. Callén, J.M. López and A.M. Mastral, Seasonal variation of benzo(a)pyrene in the Spanish airborne PM10. Multivariate linear regression model applied to estimate BaP concentrations, *Journal of Hazardous Materials*, **180**(1–3) (2010) 648–655. ISSN 0304-3894. DOI: 10.1016/j.jhazmat.2010.04.085.
10. L.E. Sucar, J. Pérez-Brito, J.C. Ruiz-Suárez and E. Morales, Learning Structure from Data and Its Application to Ozone Prediction, *Applied Intelligence*, **7**(4) (1997) 327–338. ISSN: 0924-669X.
11. K.P. Moustris, I.C. Ziomas and A.G. Paliatsos, 3-Day-Ahead Forecasting of Regional Pollution Index for the Pollutants NO2, CO, SO2, and O3 Using Artificial Neural Networks in Athens, Greece, *Water, Air, & Soil Pollution*, **209**(1) (2010). Springer. 29–43. ISSN: 0049-6979 (Print), 1573-2932 (Online). DOI: 10.1007/s11270-009-0179-5
12. E. Salazar-Ruiz, J.B. Ordieres, E.P. Vergara and S.F. Capuz-Rizo, Development and comparative analysis of tropospheric ozone prediction models using linear and artificial intelligence-based models in Mexicali, Baja California (Mexico) and Calexico, California (US), *Environmental Modelling & Software*, **23**(8) (2008) 1056–1069. ISSN: 1364-8152. DOI: 10.1016/j.envsoft.2007.11.009.
13. A.-L. Dutot, J., Rynkiewicz, F.E. Steiner and J. Rude, A 24-h forecast of ozone peaks and exceedance levels using neural classifiers and weather predictions, *Environmental Modelling and Software*, **22**(9) (2007) 1261–1269. ISSN: 1364-8152. DOI: 10.1016/j.envsoft.2006.08.002.
14. W. Wang, C. Men and W. Lu, Online prediction model based on support vector machine, *Neurocomputing*, **71**(4–6) (2008) 550–558. ISSN: 0925-2312.
15. P. Goyal, N. Jaiswal, A. Kumar, J.K. Dadoo and M. Dwarakanath, Air quality impact assessment of NOx and PM due to diesel vehicles in Delhi, *Transportation Research Part D: Transport and Environment*, **15**(5) (2010) 298–303. ISSN 1361-9209. DOI: 10.1016/j.trd.2010.03.002.
16. I. López-Yáñez, *Clasificador Automático de Alto Desempeño* (In Spanish). M.Sc. Thesis. (National Polytechnics Institute, Computers Research Center, Mexico, Mexico, 2007).
17. C. Yáñez-Márquez, I. López-Yáñez and G.delaL. Sáenz-Morales, Analysis and Prediction of Air Quality Data with the Gamma Classifier, *Lecture Notes in Computer Science*, LNCS 5197 (2008), Springer-Verlag Berlin Heidelberg, 651–658. ISBN: 978-3-540-72394-3.
18. I. López-Yáñez, C. Yáñez-Márquez and V.M. Silva-García, Forecasting Air Quality Data with the Gamma Classifier, in *Pattern Recognition*, ed. Y. Peng-Yeng (INTECH, Croatia, 2009) 499–512. ISBN: 978-953-307-014-8. Available from: http://sciyo.com/articles/show/title/forecasting-air-quality-data-with-the-gamma-classifier
19. M. Jablonowski, Implications of Fuzziness for the Practical Management of High-Stakes Risks, *International Journal of Computational Intelligence Systems*, **3**(1) (2010) 1–7.
20. Gobierno del Distrito Federal. Norma Ambiental para el Distrito Federal (in Spanish), *Gaceta Oficial del Distrito Federal*, XVI Epoch (2006).
21. K.S. Papadomanolakis, A.P. Kakarountas, N. Sklavos and C.E. Goutis, A Fast Johnson-Mobius Encoding Scheme for Fault Secure Binary Counters, in *Proc. Design, Automation and Test in Europe 2002*, eds. C.D. Kloos and J. da Franca (Paris, France, 2002).
22. C. Yáñez-Márquez, *Memorias Asociativas Basadas en Relaciones de Orden y Operadores Binarios* (In Spanish). Ph.D. Thesis. (National Polytechnics Institute, Computers Research Center, Mexico, Mexico, 2002).
23. M.E. Acevedo-Mosqueda, C. Yáñez-Márquez and I. López-Yáñez, Alpha-Beta Bidirectional Associative Memories: Theory and Applications, *Neural Processing Letters*, **26**(1) (2007) Springer-Verlag Berlin Heidelberg, 1–40. ISSN: 1370-4621. DOI: 10.1007/s11063-007-9040-2.
24. L.O. López-Leyva, C. Yáñez-Márquez, R. Flores-Carapia and O. Camacho-Nieto, Handwritten Digit Classification Based on Alpha-Beta Associative Model, *Lecture Notes in Computer Science*, LNCS 5197 (2008) Springer-

Verlag Berlin Heidelberg, 437–444. ISBN: 978-3-540-72394-3.

25. R.O. Duda, P.E. Hart and D.G. Stork, *Pattern Classification* (John Wiley & Sons, New York, USA, 2001, ISBN: 0-471-05669-3).
26. A.L. Martin-Del Pozzo, T. González-Morán, R. Espinasa-Pereña, M.A. Butron and M. Reyes, Characterization of the recent ash emissions at Popocatepetl Volcano, Mexico, *Journal of Volcanology and Geothermal Research*, **170**(1–2) (2008) Elsevier, 61–75. ISSN: 03770273. DOI: 10.1016/j.jvolgeores.2007.09.004.
27. A.C.L. Larocque, J.A. Stimac, C. Siebe, K. Greengrass, R. Chapman and S.R. Mejia, Deposition of a high-sulfidation Au assemblage from a magmatic volatile phase, Volcán Popocatépetl, Mexico, *Journal of Volcanology and Geothermal Research*, **170**(1–2) (2008) Elsevier, 51–60. ISSN: 03770273. DOI: 10.1016/j.jvolgeores.2007.09.009.
28. F. Pérez-Duarte, *Criterio para evaluar la calidad del aire ambiente con respecto al ozono (O_3)* (Modificación a la Norma Oficial Mexicana NOM-020-SSA1-1993, Secretaría de Salud, México, D. F., México, 2002).
29. E. Enríquez-Rubio, *Criterios para evaluar la calidad del aire ambiente con respecto al monóxido de carbono (CO)* (Norma Oficial Mexicana NOM-021-SSA1-1993, Secretaría de Salud, México, D. F., México, 1994).
30. F. Pérez-Duarte, *Criterio para evaluar la calidad del aire ambiente con respecto al bióxido de azufre (SO_2)* (Norma Oficial Mexicana NOM-022-SSA1-1993, Secretaría de Salud, México, D. F., México, 1994).
31. F. Pérez-Duarte, *Criterio para evaluar la calidad del aire ambiente con respecto al bióxido de nitrógeno (NO_2)* (Norma Oficial Mexicana NOM-023-SSA1-1993, Secretaría de Salud, México, D. F., México, 1994).
32. E. Enríquez-Rubio, *Criterios para evaluar la calidad del aire ambiente con respecto a material particulado* (Modificación a la Norma Oficial Mexicana NOM-025-SSA1-1993, Secretaría de Salud, México, D. F., México, 2005).
33. D.H. Wolpert and W.G. Macready, No free lunch theorems for optimization, *IEEE Transactions on Evolutionary Computation*, **1**(1) (1997).
34. M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I.H. Witten, The WEKA Data Mining Software: An Update, *SIGKDD Explorations*, **11** (2009) 10–18.
35. M. Lu, C.L.P. Chen, J. Huo and X. Wang, Multi-stage decision tree based on inter-class and inner-class margin of SVM, in *Proc. 2009 IEEE International Conference on Systems, Man, and Cybernetics* (San Antonio, TX, USA, 2009).
36. J. Rosenhead, Robustness analysis: keeping your options open, in *Rational Analysis for a Problematic World Revisited: problem structuring methods for complexity, uncertainty and conflict*, eds. J. Rosenhead and J. Mingers (Wiley, Chichester) (2001) 181-207.
37. P.Quevauviller, O. Thomas and A. van der Beken, *Wastewater Quality Monitoring and Treatment* (John Wiley & Sons, West Sussex, England, 2006).