16th World Congress of the International Fuzzy Systems Association (IFSA)
9th Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT)

A proposal for regime change/duration classification in chaotic systems

P. A. Lopes[1] I. Y. Sumida[2], H. A. Camargo[1], H. de Campos Velho[2], S. Sandri[2]

[1]Dep. de Computação - UFSCar, São Carlos - SP - Brazil
[2]Lab. Assoc. de Computação e Matemática Aplicada - INPE, São José dos Campos, SP, Brazil

Abstract

In order to to predict regime duration in a given chaotic system, for a set of output prototypes are available, we propose to use a clustering technique for the definition of classes of regime duration, which are then used by a chosen classifier. In this way, the exact boundaries between classes are allowed to emerge from the data, as long as prototypical values fall in distinct classes. We investigate the use of both unsupervised and semi-supervised fuzzy clustering techniques FCM and ssFCM, as well as the traditional k-Means technique. To classify the data, we use neuro-fuzzy system ANFIS and two decision trees (J48 and NBTree). We apply the procedure on the well-known Lorenz strange attractor, having bred vector counts as input variables.

Keywords: Chaotic systems, fuzzy clustering, bred vectors, Lorenz attractor, neuro-fuzzy systems, decision trees

1. Introduction

In chaotic systems, small changes in the initial conditions can lead to very different results from the original system trajectory, what makes the prediction of the future state of such systems a non-trivial problem. Many real-world phenomena present chaotic behaviour and their prediction, however imprecise, can be crucial in the prevention (or minimization of impact) of major natural disasters.

A well-established means for predicting behaviour of chaotic systems is the breeding vector technique, originally developed to evaluate the impact from initial perturbations on ensemble forecasting in numerical weather prediction at the American National Center for Environmental Prediction (NCEP) [31]. Bred vectors are a nonlinear generalization of Lyapunov exponents, that are used to rate the differences of two initially close trajectories of a given chaotic system (see [1] and [15] for studies on the stability properties of evolving flows for Lyapunov and bred vectors, respectively).

The authors of [10] derived rules of thumb for predicting behaviour of the Lorenz strange attractor [23], a prototypical chaotic system with two regimes, using visual observation of bred vector values along time. In a similar manner, the authors of [8] proposed rules for another chaotic system, the coupled three-waves problem from solar physics [7, 22, 24]. In [29], bred vectors were used to automatically predict regime change in the Lorenz attractor and the three-waves problem using both Neural Networks [14] (see also [25]) and Neuro-Fuzzy Systems [21]. In that work, the classes are characterized by intervals of regime duration, that are either arbitrarily chosen or established so as to maximize accuracy.

In classification problems, the definition of classes is usually defined by the end user. However, in certain cases, the exact boundaries between the classes is not so important to the user, as long as some prototypical values fall in distinct classes. The use of prototypes allows one to let the boundaries emerge from the data itself, leading to more accuracy and/or interpretability, while still satisfying the user needs. One of the means to obtain such boundaries is by the use of clustering techniques.

Here we propose a procedure for the prediction of regime change/duration in chaotic systems: a classification algorithm is trained to make a first dichotomous prediction about regime change (yes/no), a clustering algorithm is used to determine the boundaries for the classes using prototypical values and a last classification algorithm is trained to make the final regime duration class prediction. We present the application of the proposed procedure on Lorenz strange attractor, using the breeding vector method to derive the input variables. The main focus of this work is the comparison between unsupervised and semi-supervised fuzzy clustering techniques, here represented by Fuzzy C-Means (FCM) [5] and Semi-Supervised Fuzzy C-Means (ssFCM) [4, 26]. These techniques are compared with the well-known k-Means algorithm [6].

To make the first and the last classification, we have used decision trees induction algorithms C4.5 [28] and NBTree [19], by means of their Weka implementations [33] (C4.5 is then called J48 [34]), and neuro-fuzzy system ANFIS [30].

This paper is organized as follows. In Section 2 we briefly present Lorenz chaotic systems and bred vectors. In Sections 3 and 4, we address the clustering techniques and the classifier systems used in this work. Sections 5 and 6 respectively present the procedure proposal and experiments. Section 7 finally brings the conclusion.

2. Bred vectors and Lorenz strange attractor

A strange attractor is a particular kind of chaotic dynamic system, in which the system trajectory is confined within a bounded region and orbits are never repeated. Lorenz introduced a chaotic dynamic system [23] de-

© 2015. The authors - Published by Atlantis Press

scribed by the following equations:

$$dx/dt = \sigma(y - x) \quad (1)$$
$$dy/dt = \rho x - y - xz \quad (2)$$
$$dz/dt = xy - \beta z \quad (3)$$

With parameters $\sigma = 10, \rho = 28, \beta = 8/3$, the resulting system is a strange attractor with two regimes (see Figure 1). Although described by three simple equations, it is hard to identify exactly when a regime change in such a system will happen and how long it will last.

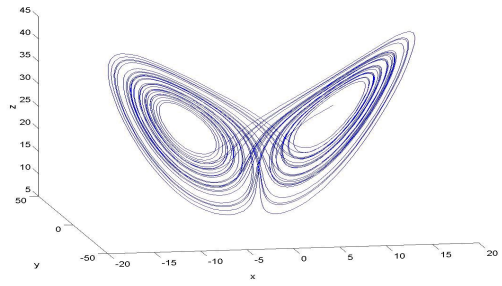

Figure 1: Trajectory of the Lorenz strange attractor (adapted from [10]).

The breeding method [31] consists in periodically running a chaotic model twice, one with the original data (control run) and the other with a small perturbation added to it. After a fixed number of time steps, the results obtained in each trajectory are subtracted; the difference is rescaled and then used as the new perturbation. The difference between the two model runs is called a *bred vector*.

Bred vectors can be calculated using a simple procedure:

1. Define a perturbation parameter ϵ, and the number of time-steps n to propagate the perturbation (final time is defined as: $t_n = n\,\Delta t$).
2. Compute the perturbed field:

   $$\delta h(x, t_n) = h(x+\epsilon, t_m \rightarrow t_{m+n})\,.$$

3. Calculate the bred vector:

   $$b(t_n) \equiv \Delta h(x, t_n) \frac{\epsilon}{\|\Delta h(x, t_n)\|}\,,$$

   $$\Delta h(x, t_n) = [h(x, t_n) + \delta h(x, t_n)] - h(x, t_n)$$

   where $\|\cdot\|$ is an suitable norm. The bred vector $b(t_n)$ can be used to evaluate how far a prediction can be from a true value in a dynamical system.

The bred vector technique has been applied for the error analysis in the dynamical systems [9], and this methodology has also been compared with other schemes, such as Lyapunov vector [16]. The amplification of the bred vectors can be used to identify regions of high error growth within an attractor [10]. Figure 2 illustrates bred vectors growth.

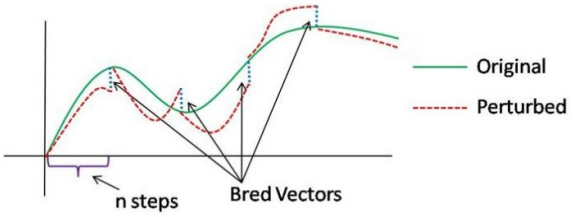

Figure 2: Bred vectors growth illustration (adapted from [29])

In [10], Evans et al derived rules for predicting behaviour of the Lorenz strange attractor upon visual observation of 4 intervals of bred vectors (see Figure 3a). The presence of a red star indicates that the bred vector growth in the previous 8 steps was greater than 0.064, the blue stars indicate a negative growth rate and the green and yellow stars stand for intervals [0, 0.032) and [0.032, 0.064] respectively. Figure 3b) depicts only one axis of the system trajectory: the abscissa (time) separates the two regimes; each inflection point indicates the beginning of a new orbit of the system.

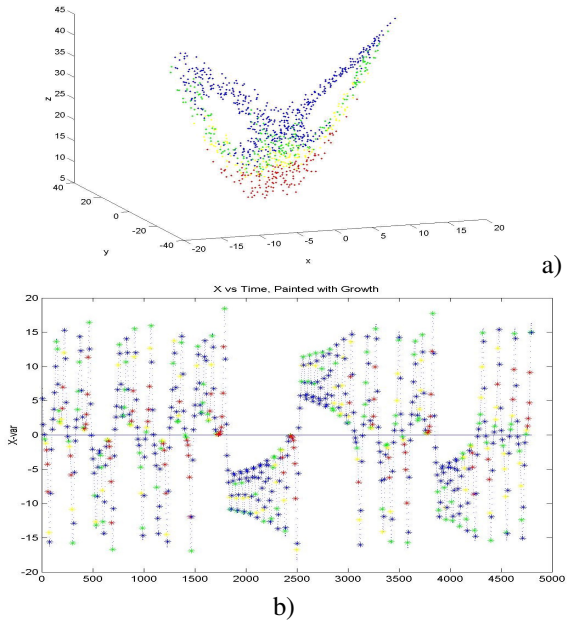

Figure 3: Lorenz attractor painted with 4 bred vector classes: a) in 3-D and b) for $X(t)$ (adapted from [10])

Table 1 presents a database with Lorenz strange attractor trajectory in terms of bred vectors counts, where each row corresponds to a time step. The first column brings the number of bred vectors in classes blue, green, yellow and red (see Figure 3) and the second column indicates whether the regime changes after the current orbit (yes/no). The third column is obtained from the second column: positive (respec. negative) numbers are associated to "yes" (respec. "no") and indicate the duration of the next (respec. current) regime.

In [29], bred vectors were used to automatically pre-

Table 1: Fragment of training database

(b,g,y,r)	change	#orbits
(3,0,0,0)	yes	1
(3,1,2,3)	yes	2
(6,4,2,0)	no	-1
(4,0,1,2)	yes	1
(4,2,2,2)	yes	2
(6,3,2,1)	no	-1
(2,1,0,4)	yes	3
(8,3,1,0)	no	-2
(4,2,3,1)	no	-1
(2,1,0,4)	yes	3
...		

dict regime change in the Lorenz attractor and the three-waves problem using a Multilayer Perceptron Neural Network with 3 layers [14] and Neuro-Fuzzy System ANFIS [21], with a fixed set of data for training and another one for testing. In that work, three kinds of experiments were performed to predict Lorenz strange attractor behaviour. In the first experiment, the goal was to predict only whether the trajectory would remain in a regime after the current orbit was finished. In the other two experiments, it was also tested how long the trajectory would remain in a regime after the current orbit was finished, either when it changed regimes or not, characterized by positive or negative outputs. The number of orbits, for either positive or negative outputs, was partitioned in intervals (classes), arbitrarily chosen.

3. Fuzzy Systems for supervised and unsupervised tasks

Fuzzy systems [11] aim at emulating some of the human capacity of reasoning with vague information. Membership to a fuzzy set is measured by a number between 0 and 1 in the real scale, instead of simply 0 or 1 as in its classical counterpart.

Most of the systems created using fuzzy sets theory are based on rules of thumb of the type "If condition then conclusion", where the variables in both the condition and conclusion parts are associated to fuzzy sets. These rules can be either given by the user or learnt from data in a supervised process, using methods such as the so-called neuro-fuzzy systems.

Fuzzy systems have also been used for clustering, an unsupervised task. In this case, the system assigns to each element in a given universe of discourse a membership degree between 0 and 1 to each cluster. A subsequent step usually is undertaken in which the element is assigned to the class with the highest membership degree, as in the classical clustering counterpart.

In the following we describe the fuzzy algorithms used in this work for clustering (FCM and ssFCM) and classification (ANFIS).

3.1. Fuzzy clustering: original Fuzzy C-Means and semi-supervised method ssFCM

Inductive machine learning is traditionally divided in two main categories: supervised, making use of classes information (labels) to perform the learning process, and unsupervised, which makes use of intrinsic information within input data instead of pre-existing labels. Clustering methods are traditionally unsupervised techniques. The objective of clustering is to divide data elements into clusters so that similar data are allocated in the same cluster and dissimilar data are allocated to different clusters. The similarity/dissimilarity between objects can be measured by several different forms, the most common one being based on the Euclidean distance.

Clustering is usually said to be either hard, in which any object belongs to a single cluster, or soft, in which an object may belong to several clusters. One of the most traditional hard clustering methods is k-Means (see [6]), in which a set of objects is partitioned in k clusters and each object is allocated to the cluster with the smallest average distance.

In fuzzy clustering, a type of soft clustering, any object has a membership degree to each cluster, all adding up to 1. One of the most widely used fuzzy clustering algorithms is Fuzzy C-Means (FCM) [5]. Given a set of n objects where each object is described by a vector of attributes, the algorithm returns a set of k cluster centers $C = \{c_1, c_2, \ldots, c_k\}$ and a partition matrix $W = w_{i,j} \in [0,1], 1 \leq i \leq n, 1 \leq j \leq c$, $\sum_j$, where each element w_{ij} gives the degree to which element x_i belongs to cluster c_j. The fact that the number of clusters to be found must be given as an input parameter to the algorithm is very frequently pointed out as its main drawback.

Although supervised and unsupervised methods have been widely and successfully investigated, the idea of adapting methods to share the advantages of labeled and unlabeled data has gained increasing investments in research. These methods may be particularly interesting when dealing with the difficulty to interpret large data sets, where manual interpretation and labeling would be of high cost. Semi-supervised learning refers to the use of labeled and unlabeled data within the learning process [18, 35].

Semi-supervised clustering approaches commonly make use of pre-existing information in two different formats: seeds [28, 27, 2] and pair-wise constraints [32, 3, 12]. Seeds are labeled examples within a mostly unlabeled data set that can be used in the restriction of the clustering process and the definition of the number of clusters and cluster labels. The pairwise restrictions indicate that a pair of examples must belong to the same cluster (must-link) or that they should belong to different clusters (cannot-link). An example of semi-supervised clustering is the Semi-Supervised Fuzzy C-Means algorithm (ssFCM) [4]. This method adapts FCM to consider information in the form of seeds and improve clustering performance.

The modifications of FCM by ssFCM were made aiming to solve three problems from the application of orig-

inal FCM: the difficulty in defining the number of clusters, the difficulty in defining appropriate labels to each cluster after the clustering process and the tendency to equal the number of members in each cluster.

The dataset in ssFCM is a union of the labeled data, Xl, and unlabeled data, Xu. The number of clusters c is set to the number of different labels represented in Xl. The pertinence of each labeled example is defined as 1 to their respective clusters and that value is not altered during the updates of the algorithm. As the pertinence of each labeled example never changes, these examples strongly influence the calculation for new cluster prototypes.

3.2. Neurofuzzy systems

The two most well-known kinds of fuzzy systems are the Mamdani and Sugeno models (the later one is also known as Takagi-Sugeno or Takagi-Sugeno-Kang) [11]. Both frameworks use fuzzy terms to model the conditions on the left-hand side of the rules, but differ on the modeling of the conclusions on the right-hand side of the rules. In Sugeno fuzzy systems, a conclusion of a rule is modeled as a (usually linear) function of the values of the input variables. In Mamdani systems, fuzzy sets of the output variables are used in the conclusions of rules, a characteristic that make these systems interpretable, contrary to what happens with Sugeno systems.

The term *neuro-fuzzy* is nowadays used to name any system that somehow involves the joint use of neural networks and fuzzy systems paradigms to solve a problem. A particular class of neuro-fuzzy systems, proposed in [21], derives a fuzzy rule based system, whose defining parameters (fuzzy terms and rules) are learnt through training performed in a neural network-like learning process upon the presentation of a set of pairs (input, desired output). The main representative of this class is ANFIS (Adaptive-Network-Based Fuzzy Inference System) [30], that learns the parameters of a Sugeno fuzzy system. It uses backpropagation to learn the fuzzy terms on the rule left-hand side and the LMS algorithm (Least Square Means) for calculating the parameters on the right-hand side of the rule.

4. Decision Tree Based Classifiers

Decision trees are powerful models of supervised inductive machine learning, which can be applied to classification tasks. Decision trees are widely used as learning algorithms due to several reasons. Besides being built by an understandable and intuitive process, the induced model can be expressed graphically as a tree structure, as well as a set of rules. Despite their simplicity, they are usually competitive with more costly approaches. Decision trees are usually robust, scalable and can be applied to datasets including a large number of examples.

Classic decision trees structures are composed by leaves, representing classes, and intermediary decision nodes, which are associated to attributes and represent tests according to the values of their respective attributes. The automatic construction of a decision tree can be made from a labeled dataset of representative examples by means of a recursive process that partitions the data set as each node is created.

The C4.5 algorithm, proposed by [28], is one of the most popular algorithms for the induction of decision trees. C4.5 uses entropy and information gain measures to decide on the quality of the input attributes and then select one of them to be associated to each node. More specifically, the information gain rate of an attribute is defined by the entropy reduction. C4.5 performs a post pruning method aiming at generalizing the final model and preventing it from becoming too adherent to the specific training examples (overfitting). After the induction of the tree, the model can be used to classify any example whose class is unknown. The inference (classification) process starts from the root node and, depending on the attribute values of the example being classified, follows one of the branches leaving that node, until a leaf node is reached. The class in this leaf is defined as the class of the example.

NBTree [19] is a hybrid algorithm which induces decision trees with intermediary nodes containing attributes (as regular decision trees) but in which leaf nodes contain Naïve-Bayesian classifiers instead of just single classes. Naïve-Bayes classifiers [20] are generally easy to understand and present accurate results in many classification tasks. Besides that, they are very robust to irrelevant attributes and the induction of the model is very fast. On the other hand, Naïve-Bayes classifiers require making strong independence assumptions that, when violated, may not improve accuracy as the database size increases. The NBTree approach attempts to take profit of the segmentation capabilities of classic decision trees and evidence accumulation of Naïve-Bayes classifiers. The J48 implementation of the classic C4.5 algorithm, and the implementation of NBTree, both available at the WEKA framework [34], was used in our experiments.

5. The proposed approach

Here we are interested in guiding the selection of output classes of regime duration in chaotic systems, characterized by intervals. We investigate the case in which the user, instead of furnishing the exact definition of the intervals, only provides output prototypes, around which the intervals are to be automatically determined.

We propose to use clustering as the basis for the automatic determination of classes of regime duration, that are later used in classification methodologies. We investigate the use of both unsupervised and semi-supervised fuzzy clustering techniques on Lorenz strange attractor, using the breeding vector method to derive the input variables.

We propose to divide the construction of the model of classification of regime duration in four phases, as follows:

1. dichotomous classification: a classifier is trained to predict regime change (yes/no) at the end of an orbit;

2. clustering: the database is divided in two parts according to regime change (yes/no) and a clustering algorithm is applied on each one, with the number of clusters furnished by the user;
3. determination of output classes: a decision algorithm is used to obtain the output classes, characterized by intervals on the number of orbits a system will remain in the alternative (respec. present) regime in case it changes (respec. does not change) regimes at the end of an orbit;
4. final classification: a classifier is trained using the classes obtained in the previous phase.

For different types of chaotic systems, the definition above has to be slightly changed. In the coupled three-waves problem [7, 22, 24], for instance, the output are intervals of elapsed time steps before regime change instead of the number of orbits.

The first and second phases of the proposed procedure could be a single one, i.e. the clustering would be performed with both negative and positive outputs, but experiments in [29] indicate that the hierarchical process with the dichotomous classification occurring prior to clustering provides superior overall results.

In the second phase, a clustering algorithm is applied separately on the positive and negative outputs database. At the end of the clustering process, each point in the training data is assigned to a cluster in the associated database. Figure 4 depicts the distribution of elements with positive outputs of a dataset with three clusters. This figure shows that the clustering procedure itself may be unable to partition the number of (either positive or negative) outputs in interval classes in a straightforward manner.

In the third phase, a decision procedure derives the interval classes from the clusters. One can obtain the interval classes from the clusters by simply optimizing accuracy, as done in [29]. Here, however, we are specifically interested in interval classes, which are such that the positive (respec. negative) prototypical outputs P (respec. N) furnished by the user fall in different classes. Here the non-prototypical outputs are then allocated to the clusters so as to maximize global accuracy, but ensuring the three constraints: i) there exists one prototype per class and ii) the classes partition the output space and iii) each class is an interval (the classes are contiguous). For example, let us consider the distribution of elements with positive outputs of a dataset with three clusters depicted in Figure 4. If the user choice of prototypes is $P = \{1, 3, 5\}$ (respec. $P = \{2, 4, 6\}$), the maximal accuracy (so that the restrictions are satisfied) is obtained with classes $\{\{1, 2\}, \{3\}, \{4, ..., 12\}\}$ (respec. $\{\{1, 2\}, \{3, 4\}, \{5, ..., 12\}\}$).

Once the classes are established in phase 4, the last phase simply is used to train another classifier to do the final prediction for regime change/duration.

6. Experiments

Following [10], the breeding method was applied on the Lorenz model, integrated with time steps $\Delta t = 0.01$,

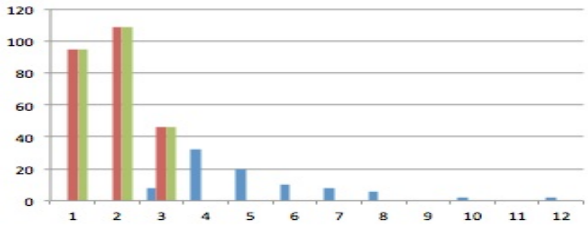

Figure 4: Distribution of positive outputs in 3 clusters

and a second run started from an initial perturbation $(\delta x_0, \delta y_0, \delta z_0)$ added to the control at time t_0. Here, however, as in [29], the same perturbation was used in the subsequent runs.

The data consists in 559 positive (the regime changes) and 715 negative (the regime does not change) data points, with 4 input variables (blue, green, yellow and red) determined by [10], each one accounting for the number of bred vectors in pre-established intervals found in a given orbit (see Section 2). We have used 5-fold cross-validation with stratification to verify the quality of the results.

6.1. Dichotomous classification

In the first phase, we have employed ANFIS and decision trees J48 and NBTree, to classify whether the regime changes at the end of the present orbit. Table 2 brings the average result for both methods.

Table 2: Dichotomous classification accuracy

J48	ANFIS	NBTree
95.44%	94.50%	91.60 %

We see that the results for ANFIS and J48 are similar and that it is natural to expect at least a 5% overall error from the whole procedure. Note that the results from this phase do not interfere with the remaining training process.

6.2. Clusterization and determination of output classes

Three methods have been used for clustering; k-Means, FCM and ssFCM. For ssFCM, for each prototype, 20% of the data points having it as output were randomly chosen as seeds. Tables 3, 4 and 5 respectively bring the clusters found using k-Means, FCM and ssFCM in the 5 folds. The notation for the partitioning in the tables has been simplified; for example, the sequence [1][2 4][5 12] denotes the partition $\{\{1\}, \{2, 3, 4\}, \{5, 6, 7, 8, 9, 10, 11, 12\}\}$.

For ssFCM, the clusters in all folds separated the data already in intervals so the classes were derived straightforwardly. The decision rule was only used in a few folds in which either outputs 12 and -12 were absent. For k-Means and FCM, as expected, due to the absence of seeds, the clusters did not separate the outputs straightforwardly and the decision rule had to be used in

all experiments. Nevertheless, the resulting classes obtained in the folds did not differ very much from those obtained by ssFCM.

Table 3: Classes derived using k-Means

clusters	prototypes	classes
2	{1, 3}	[1] [2 12]
2	{1, 4}	[1] [2 12]
3	{1, 3, 5}	[1] [2 4] [5 12]
		[1 2] [3 4] [5 12]
		[1] [2 3] [4 12]
3	{2, 4, 6}	[1 3] [4 5] [6 12]
2	{-1, -3}	[-1 -2] [-3 -12]
2	{-1, -4}	[-1 -2] [-3 -12]
3	{-1, -3, -5}	[-1] [-2 -4] [-5 -12]
		[-1 -2] [-3 -4] [-5 -12]
3	{-2, -4, -6}	[-1 -2] [-3 -5] [-6 -12]

Table 4: Classes derived using FCM

clusters	prototypes	classes
2	{1, 3}	[1] [2 12]
2	{1, 4}	[1] [2 12]
3	{1, 3, 5}	[1] [2 3] [4 12]
3	{2, 4, 6}	[1 3] [4 5] [6 12]
2	{-1, -3}	[-1 -2] [-3 -12]
2	{-1, -4}	[-1 -2] [-3 -12]
3	{-1, -3, -5}	[-1 -2] [-3] [-4 -12]
3	{-2, -4, -6}	[-1 -2] [-3] [-4 -12] [-1 -2] [-3 -4] [-5 -12] [-1 -3] [-4] [-5 -12]

Table 5: Classes derived using ssFCM

clusters	prototypes	classes
2	{1, 3}	[1] [2 12]
2	{1, 4}	[1 2] [3 12]
3	{1, 3, 5}	[1] [2 4] [5 12]
3	{2, 4, 6}	[1 3] [4 5] [6 12]
2	{-1, -3}	[-1 -2] [-3 -12]
2	{-1, -4}	[-1 -2] [-3 -12]
3	{-1, -3, -5}	[-1] [-2 -3] [-4 -12]
3	{-2, -4, -6}	[-1 -2] [-3 -5] [-6 -12]

6.3. Final classification

Algorithms J48, NBTree and ANFIS were applied using the classes obtained by all of our clustering methods (k-Means, FCM and ssFCM). The average results for this phase are shown in Tables 6, 7 and 8.

In Tables 6, 7 and 8, we see that, for the configurations used, ANFIS has the lowest average accuracy for all sets of prototypes and that J48 fares usually better than NBTree. The highest average accuracy in all experiments is obtained by NBTree for prototypes $\{1, 4\}$ using ssFCM for clustering (95.18 %).

Table 6: Accuracy for k-Means derived classes (%)

clusters	prototypes	J48	ANFIS	NBTree
2	{1, 3}	**92.12**	90.32	89.80
2	{1, 4}	92.12	88.18	89.80
3	{1, 3, 5}	**88.31**	82.26	83.89
3	{2, 4, 6}	**92.12**	86.57	88.90
2	{-1, -3}	**88.96**	84.77	88.69
2	{-1, -4}	88.96	81.14	88.69
3	{-1, -3, -5}	78.56	69.22	74.54
3	{-2, -4, -6}	82.96	75.97	**83.10**

Table 7: Accuracy for FCM derived classes (%)

clusters	prototypes	J48	ANFIS	NBTree
2	{1, 3}	**92.12**	90.16	89.80
2	{1, 4}	92.12	90.16	89.80
3	{1, 3, 5}	85,81	82.82	84.79
3	{2, 4, 6}	**92.12**	86.58	88.90
2	{-1, -3}	**88.96**	77.62	88.69
2	{-1, -4}	**90.36**	77.62	88.69
3	{-1, -3, -5}	**80.70**	76.78	80.56
3	{-2, -4, -6}	81.11	58.46	80.56

Table 8: Accuracy for ssFCM derived classes (%)

clusters	prototypes	J48	ANFIS	NBTree
2	{1, 3}	**92.12**	90.16	89.67
2	{1, 4}	93.73	87.83	**95.18**
3	{1, 3, 5}	88.01	85.50	82.11
3	{2, 4, 6}	**92.12**	86.58	88.90
2	{-1, -3}	**88.96**	77.62	87.85
2	{-1, -4}	**90.36**	77.62	89.79
3	{-1, -3, -5}	78.35	67.41	76.67
3	{-2, -4, -6}	82.96	79.86	**83.10**

In general, ssFCM fared slightly better than k-Means and FCM, having obtained the highest scores in 6 out of 8 experiments (against 5 out of 8 for both k-Means and FCM).

6.4. Computational issues

FCM and ANFIS were run in an Intel Core I5 2.27 GHz, under Linux operating system, using MATLAB platform. K-Means, J48 and NBTree were run using Weka Plataform, while we used a ssFCM implementation in Java. Executions of NBTree algorithm were performed in an Intel Core I3 3.07 GHz, under Windows 7 operating systems. Executions of K-Means, J48 and ssFCM were run in an AMD Phenom II 3.2Ghz with 4GB, under Windows 7 operation system.

Table 9 brings the clustering time (means and standard deviation) for algorithms k-Means, FCM, and ssFCM. Tables 10, 11 and 12 bring the classification time (means and standard deviation) with classes derived from clusters obtained by k-Means, FCM, and ssFCM, respectively. All values refer to training and test altogether. We see in these tables that the adopted pro-

cedures presented low cost for the Lorenz dataset. In what regards clustering, semi-supervised technique ssFCM is as fast as unsupervised k-Means (FCM cannot be properly compared to these algorithms since another machine was used to get the results). As for classification, J48 is much faster than NBTree (ANFIS cannot be properly compared to these algorithms since another machine was used to get the results).

Table 9: Time statistics (in seconds) for clustering: means (standard deviation)

clst.	proto.	k-Means	FCM	ssFCM
2	{1, 3}	0.02 (0.01)	0.11 (0.03)	0.01 (0.01)
2	{1, 4}	0.02 (0.01)	0.10 (0.00)	0.02 (0.01)
3	{1, 3, 5}	0.01 (0.00)	0.16 (0.05)	0.03 (0.01)
3	{2, 4, 6}	0.01 (0.00)	0.14 (0.03)	0.03 (0.03)
2	{-1, -3}	0.02 (0.02)	0.10 (0.00)	0.02 (0.01)
2	{-1, -4}	0.02 (0.02)	0.11 (0.01)	0.02 (0.01)
3	{-1, -3, -5}	0.02 (0.01)	0.15 (0.02)	0.02 (0.01)
3	{-2, -4, -6}	0.02 (0.01)	0.13 (0.00)	0.02 (0.01)

Table 10: Time statistics (in seconds) for processing classes derived by k-Means: means (standard deviation)

clusters	prototypes	J48	ANFIS	NBTree
2	{1, 3}	0.02 (0.01)	12.48 (0.12)	0.24 (0.09)
2	{1, 4}	0.02 (0.01)	12.72 (0.39)	0.31 (0.19)
3	{1, 3, 5}	0.01 (0.00)	13.05 (0.65)	0.52 (0.10)
3	{2, 4, 6}	0.01 (0.00)	12.49 (0.19)	0.35 (0.13)
2	{-1, -3}	0.02 (0.02)	15.91 (0.10)	0.44 (0.08)
2	{-1, -4}	0.02 (0.02)	16.03 (0.43)	0.45 (0.12)
3	{-1, -3, -5}	0.02 (0.01)	15.64 (0.13)	0.39 (0.16)
3	{-2, -4, -6}	0.02 (0.01)	15.73 (0.07)	0.25 (0.08)

Table 11: Time statistics (in seconds) for processing classes derived by FCM: means (standard deviation)

clusters	prototypes	J48	ANFIS	NBTree
2	{1, 3}	0.00 (0.00)	13.27 (0.08)	0.52 (0.34)
2	{1, 4}	0.00 (0.00)	13.27 (0.08)	0.24 (0.11)
3	{1, 3, 5}	0.01 (0.00)	12.09 (0.29)	0.54 (0.18)
3	{2, 4, 6}	0.00 (0.00)	12.34 (0.03)	0.28 (0.06)
2	{-1, -3}	0.00 (0.00)	16.21 (0.20)	0.43 (0.11)
2	{-1, -4}	0.00 (0.00)	16.21 (0.20)	0.49 (0.19)
3	{-1, -3, -5}	0.00 (0.01)	16.46 (0.32)	0.15 (0.08)
3	{-2, -4, -6}	0.01 (0.00)	12.33 (0.02)	0.39 (0.26)

Table 12: Time statistics (in seconds) for processing classes derived by ssFCM: means (standard deviation)

clusters	prototypes	J48	ANFIS	NBTree
2	{1, 3}	0.01 (0.01)	13.27 (0.08)	0.22 (0.08)
2	{1, 4}	0.00 (0.00)	13.48 (0.13)	0.12 (0.04)
3	{1, 3, 5}	0.01 (0.00)	12.00 (0.03)	0.21 (0.05)
3	{2, 4, 6}	0.00 (0.01)	12.34 (0.03)	0.16 (0.03)
2	{-1, -3}	0.01 (0.01)	16.21 (0.20)	0.20 (0.03)
2	{-1, -4}	0.01 (0.00)	16.21 (0.20)	0.18 (0.04)
3	{-1, -3, -5}	0.01 (0.01)	16.37 (0.04)	0.24 (0.06)
3	{-2, -4, -6}	0.01 (0.01)	16.37 (0.08)	0.13 (0.01)

7. Conclusions and future work

We have proposed a procedure for the construction of a classification model for prediction of regime change/duration in chaotic systems, built around a set of output prototypes furnished by the user. In this procedure, first a classification algorithm is trained to make a first dichotomous prediction about regime change (yes/no), then a clustering algorithm is used to determine the boundaries for the classes, using the prototypes furnished by the user, and a last classification algorithm is trained to make the final prediction of regime duration. The goal of this approach is to allow the exact boundaries between classes to emerge from the data, as long as prototypical values fall in distinct classes, aiming at satisfying the user needs while obtaining gains in accuracy and/or interpretability.

We have applied the proposed procedure on Lorenz strange attractor [23], using the breeding vector method to derive the input variables. The main focus of this work is the comparison between unsupervised and semi-supervised fuzzy clustering techniques. For that, we have used algorithms k-Means [6] and Fuzzy C-Means (FCM) [5] as well as Semi-Supervised Fuzzy C-Means (ssFCM) [4, 26], along with a decision rule to guarantee that the resulting output classes are intervals. To make the first and the last classification, we have used decision tree C4.5 [28] by means of its Weka implementation J48 [34], decision tree NBTree [19] and neuro-fuzzy system ANFIS [30].

The results have shown that the proposed approach (dichotomous classification followed by clustering followed by final classification) is overall very promising, having produced good results with all techniques used, with semi-supervised clustering yielding slightly better accuracy in general than k-Means and FCM.

As future work, we intend to test the proposed procedure in other problems, such as the coupled three-waves problem [7, 22, 24]. We also intend to study the impact of the proposed in terms of rule interpretability, both in terms of the methods used here but also in other, such as those employing Mamdani-based neuro-fuzzy systems [11].

Acknowledgments

The authors thank P.L.B. dos Santos and R.S. Cintra, for help in the preparation of this work, and acknowledge financial support from FAPESP, CAPES and CNPq.

References

[1] K.T. Alligood, T.D. Sauer and J.A. Yorke, *Chaos: an introduction to dynamical systems*, Springer, 1996.

[2] S. Basu, A. Banerjee and R. J. Mooney, Semi-supervised clustering by seeding, In *Proc. of the 19th Int. Conf. on Machine Learning (CML'02)*, pp 27–34, 2002.

[3] S. Basu, A. Banerjee and R. J. Mooney, Active semisupervision for pairwise constrained clustering, In *Proc. of the 2004 SIAM Int. Conf. on Data Mining*, pp 333–344, 2004.

[4] A. M. Bensaid, L. O. Hall, J. C. Bezdek, and L. P. Clarke, Partially supervised clustering for image segmentation, *Pattern Recognition*, 29(5):859–871, May 1996.

[5] J.C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Kluwer Academic Publishers Norwell, MA, USA, 1981.

[6] C. M. Bishop, *Pattern recognition and machine learning*, Springer, 2006.

[7] A.C.-L. Chian, S.R. Lopes and M.V. Alves, Nonlinear excitation of Langmuir and Alfvén waves by auroral whistler waves in the planetary magnetosphere, *Astron. Astrophys.*, 288:981–984, 1994.

[8] R. S. Cintra and H. F. Campos Velho, Predictability for a Chaotic Solar Plasma System, In *Iberian and Latin-Am. Cong. on Comp. Methods for Engin. (XXIX CILAMCE)*, Maceió (Br), pp 1–8, 2008.

[9] M. Corazza, E. Kalnay, D. J. Patil, S.-C. Yang, R. Morss, M. Cai, I. Szunyogh, B. R. Hunt, J. A. Yorke (2003): Use of the breeding technique to estimate the structure of the analysis 'errors of the day", *Nonlinear Processes in Geophysics*, 10:1-11.

[10] E. Evans, E., N.K. Bathi, J. Kinney, L. Pann. M. Peña, S-C Yang, E. Kalnay and J. Hansen, Rise Undergraduates Find That Regime Changes in Lorenz's Model are Predictable. *Bull. Amer. Meteor. Soc.*, 85:521–524, 2004.

[11] F. Gomide, W. Pedrycz, *An introduction to Fuzzy Sets*, MIT Press, 1998.

[12] N. Grira, M. Crucianu and N. Boujemaa, Active semi-supervised fuzzy clustering, *Pattern Recognition*, 41(5):1851–1861, 2008.

[13] D. Guégan and J. Leroux, Predicting Chaos with Lyapunov Exponents: Zero Plays no Role in Forecasting Chaotic Systems, Chaotic Systems, ed. Esteban Tlelo-Cuautle, In http://www.intechopen.com/articles/show/title/predicting-chaos-with-lyapunov-exponents-zero-plays-no-role-in-forecasting-chaotic-systems, Access: 04/Mar/2011.

[14] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Prentice Hall, 1999.

[15] E. Kalnay, *Atmospheric modeling, data assimilation and predictability*, Cambridge Uni. Press, 2002.

[16] E. Kalnay, M. Corazza, M. Cai (2002): Are bred vectors the same as Lyapunov vectors?, EGS XXVII General Assembly, Nice (Fr), 21-26 April (2002), abstract #6820.

[17] E. Kalnay, M. Peña, S.-C. Yang and M. Cai, Breeding and predictability in coupled Lorenz models, In *Proc. of the ECMWF'02*, 2002.

[18] A. Klose and R. Kruse, Semi-supervised learning in knowledge discovery, Fuzzy Sets and Systems, 149:209–233, 2005.

[19] R. Kohavi, Scaling Up the Accuracy of Naive-Bayes Classifiers: a Decision-Tree Hybrid, In *Proc. 2nd Int. Conf. Know. Discovery and Data Mining*, 202-207, 1996.

[20] P. Langley, W. Iba, K. Thompson, An analysis of Bayesian classifiers. In *Proc. 10th National Conf. on Artificial Intelligence*, pp 223-228, AAAI Press and MIT Press, 1992.

[21] C.-T. Lin and C. S. G. Lee, *Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems*, Prentice Hall, 1996.

[22] S.R. Lopes and A.-L. Chian, A coherent nonlinear theory of auroral Langmuir-Alfvén-whistler (LAW) events in the planetary magnetosphere, *Astron. Astrophys*, 365:669–676, 1996.

[23] E. N. Lorenz, Deterministic non-periodic flow, *J. Atmos. Sci.*, 20: 130–141, 1963.

[24] C. Meunier, M.N. Bussac and G. Laval, Intermittency at the onset of stochasticity in nonlinear resonant coupling processes. *Physica*, D4:236–243, 1982.

[25] A. Pasini and V. Pelino, Can We Estimate Atmospheric Predictability by Performance of Neural Network Forecasting? The Toy Case Studies of Unforced and Forced Lorenz Models, In *Proc. CIMSA'2005*, 2005.

[26] W. Pedrycz, Algorithms of fuzzy clustering with partial supervision. *Pattern Recognition Letters*, 3(1):13–20, 1985.

[27] W. Pedrycz and J. Waletzky, Fuzzy clustering with partial supervision, *IEEE Trans. on Systems, Man, and Cybernetics*, Part B, 27(5):787–795, 1997.

[28] J.R. Quinlan, *C4.5: Programs for Machine Learning* 1st ed., Morgan Kaufmann, 1993.

[29] P.L.B. dos Santos, H.F. de Campos Velho, R. Cintra and S. Sandri, Chaotic Systems Predictability Using Neuro-Fuzzy Systems and Neural Networks with Bred Vectors, In *Studies in Fuzziness and Soft Computing*. Springer, 317:295-313, 2014.

[30] J. Shing and R. Jang, ANFIS: Adaptive-network-based fuzzy inference systems, *IEEE Trans. on Systems. Man, and Cybernetics*, 23:714–723, 1993.

[31] Z. Toth and E. Kalnay, Ensemble forecasting at NCEP and the breeding method, *Monthly Weather Review*, 126:3292–3302, 1997.

[32] K. Wagstaff, C. Cardie, S. Rogers and S. Schrödl, Constrained k-means clustering with background knowledge, In *Proc. of the 18th Int. Conf. on Machine Learning (ICML'01)*, pp 577–584, 2001.

[33] WEKA - Waikato Environment for Knowledge Analysis, http://www.cs.waikato.ac.nz/ ml/weka/ Access: 25 jul. 2012.

[34] I.H. Witten, E. Frank and M.A. Hall, *Data Mining: Practical Machine Learning Tools and Techniques*, 3rd ed., Morgan Kaufmann, 2011.

[35] X. Zhu, Semi-supervised learning literature survey, Tech. Rep. 1530, Computer Sciences, Uni. of Wisconsin-Madison, 2005.