Learning in dynamic environments: application to the diagnosis of evolving systems

Moamar Sayed Mouchaweh, Omar Ayad, Noureddine Malki
Université de Reims Champagne-Ardenne (URCA-CReSTIC), Reims-France

moamar.sayed-mouchaweh@univ-reims.fr;{omar.ayad;noureddine.malki}@etudiant.univ-reims.fr

Abstract

In dynamic environments, data characteristics may drift over time. This leads to deteriorate dramatically the performance of incremental learning algorithms over time. This is because of the use of data which is no more consistent with the characteristics of new incoming one. In this paper, an approach for learning in dynamic environments is proposed. This approach integrates a mechanism to use only the recent and useful patterns to update the classifier without a "catastrophic forgetting". This approach is used for the acoustic leak detection in the steam generator unit of the nuclear power generator "Prototype Fast React".

Keywords: Classification; incremental learning; dynamic environments; evolving systems.

1. Introduction

Evolving systems assume different functioning modes (normal/faulty) in the course of time. Several changes in system conditions, like a leak, the wear of a tool or a bad setting, can lead the system to a faulty mode. In statistical Pattern Recognition (PR) [15], historical patterns or observations about system functioning modes are divided into groups of similar patterns, called classes, using an unsupervised learning method [7] or human experience. Each class is associated to a functioning mode (normal or faulty). These patterns, with their class assignments, constitute the learning set. They are represented by a set of d features, or attributes, so they can be viewed as vectors or points of d-dimensional features in the feature space. A supervised learning method [2] uses the learning set to build a classifier that best separates the different classes in order to minimize the misclassification error. The model of each class can be represented by a membership function which determines the membership value of a pattern to a class. Then, new incoming patterns are assigned to the class for which they have the maximum membership value. In supervised learning methods, the membership function can be generated by approximating the class conditional Probability Density Function (PDF) [3], [16] [19] or by approximation of classification boundaries and discriminant functions [1], [17], [20].
Classes can be static or dynamic. Static classes are represented by restricted areas formed by similar patterns in the feature space. Hence, the way in which patterns occur is irrelevant to their membership values. Therefore, the classifier parameters remain unchanged with the time. However, data coming from evolving systems is non-stationary. In this case, classes become dynamic and their characteristics change in the course of time. Thus, classes' membership functions must be updated to take into account these temporal changes. Hence, a continuous learning over long period of time with the ability to forget data becoming obsolete and useless is needed. However, it is important that the classifier adjusts its parameters and structure without a "catastrophic forgetting". Therefore, a balance between continuous learning and "forgetting" is necessary to deal with evolving systems.
The general principle of dynamic PR methods [4], [9] [11], [12], [14], is to observe a change in some statistical properties of classes in order to decide in which state (stable, warning or action) the system is. These states correspond respectively to no change, gradual change and abrupt change. Thus, the classifier parameters, i.e. the membership functions, will be respectively unchanged, slightly adapted, or relearned from scratch. The misclassification rate is one of the most used statistical properties to observe a change. In this case, data are divided into batches and their true classes are known in advance; so that the misclassification error is easy to calculate. If this rate decreases significantly after receiving a batch of patterns, then the system is in the action state and the classifier parameters must be completely relearned.
In this paper, an approach to learn a classifier and to update its parameters and structure according to the changes in its environment conditions is proposed. This approach detects the changes in the classifier environments by observing the changes in their conditional PDF. When a significant change is detected, this approach updates the classifier using an incremental learning rule.
The paper is structured as follows. In section 2, we present the proposed approach. In section 3, this approach is evaluated using a real example. The last section concludes the paper.

2. Proposed approach

In this section, we present an approach to learn an adaptive classifier able to adjust its parameters and

structure according to new environment conditions. This approach detects a change and reacts to the detected change using four steps: membership function generation, monitoring, updating and diagnostics ones.

2.1. Membership function generation

This step aims at building a classifier using the learning set. For each class, a membership function is learned in order to be used later to assign new incoming patterns to the known classes. There are several methods to build a classifier. In this approach, the membership function of each class according to each attribute is considered as the marginal conditional PDF. The latter is estimated based on the use of histograms. In [13], an algorithm is proposed to update recursively the conditional PDF after the classification of each point.

In the next, the membership generation and the classification of new patterns are detailed.

2.1.1 Membership generation

Let C_1, C_2, .., C_c denote the classes described by d attributes. The membership generation is achieved based on the estimation of the probability histograms for each class according to each attribute. The number of bins h for a histogram is experimentally determined. This number has an important influence on the performances of estimation as well as classification. The histogram upper and lower borders can be determined either as the maximal and minimal learning data coordinates or by experts. The height of each bin b_k^j, $k \in \{1,2,..,h\}$ according to each attribute j is the number of learning patterns $n_{ib_k^j}$ of the class C_i located in this bin. The probability distribution $\{ p_i^j(y_{b_k^j}), k \in \{1,2,..,h\}, j \in \{1,2,..,d\}\}$ of the class C_i, $i \in \{1,2,..,c\}$, according to the attribute j is obtained by dividing the height of each bin by the total number N_i of learning patterns belonging to the same class C_i. These probabilities are assigned to the bins' centres $y_{b_k^j}$, $k \in \{1,2,..,h\}$:

$$p_i^j(y_{b_k^j}) = \frac{n_{ib_k^j}}{N_i} \quad (1)$$

The PDF is obtained by a linear linking between bins heights centers.

Finally, the density of probability P_i^j of the class C_i according to the attribute j is obtained by a linear interpolation of the bins centers of the probability histogram. P_i^j is considered to be the membership function μ_i^j of the class C_i according to the attribute j:

$$\mu_i^j(x^j) = p_i^j(x^j) \quad (2)$$

2.1.2 Classification Phase

The classification of a new pattern x, whose values of the different attributes are x_1, .., x_j, .., x_d, is made in two stages:

- Determination of the probability membership value $p_i^j(x^j)$ of x^j to each class C_i according to the attribute j by a projection on the corresponding probability density P_i^j,
- Merging all the probability membership values $p_i^1(x^1)$, $p_i^2(x^2)$,.., $p_i^d(x^d)$ concerning the class C_i, into a single one by an aggregation operator belonging to the t-norm family such as "minimum": $p_i = T_{j=1}^{d}(p_i^j)$. The result p_i of this fusion corresponds to the global probability membership value that x belongs to the class C_i. The aggregation operator "minimum" was selected because it is simple and gives good results. Finally, x is assigned to the class for which it has the maximum membership value.

2.1.3 Updating phase

The information related to changes in the classifier environment is carried by the new incoming patterns. Thus, this information must be extracted every time a new pattern is available. This extraction is achieved in the third phase. In this phase, the probability histograms will be updated as well as the histograms parameters (number of bins and the borders).

A) Updating of probability histograms

When a new pattern is classified in the class C_i, the number of patterns belonging to this class becomes $N_i + 1$; so the probability of each bin changes. If this pattern is located in the bin b_k^j, the new probability $p_i'^j(b_k^j)$ of this bin can be incrementally calculated as follows [13]:

$$p_i'^j(b_k^j) = p_i^j(b_k^j) \times \frac{N_i}{N_i + 1} + \frac{1}{N_i + 1} \quad (3)$$

For the other bins, the new probability is:

$$p_i'^j(b_z^j) = p_i^j(b_z^j) \times \frac{N_i}{N_i + 1}, z \neq k \quad (4)$$

Using (3) and (4), the probability histograms can be updated after the classification of a new point without the need to calculate them from scratch. We just need to know in which bin this new pattern is located.

B) Updating of histogram parameters

In order to take into account the change in features range, the parameters of each histogram is updated online after the classification of each new pattern. Let Max_i^j and Min_i^j be, respectively, the maximal and

minimal learning patterns' coordinates of the class C_i according to the attribute j. The bin width Δ_i^j of each class C_i according to each attribute j is defined as:

$$\Delta_i^j = \frac{(Max_i^j - Min_i^j)}{h_i^j} \quad (5)$$

where h_i^j is the number of histogram bins according to the class C_i and attribute j. If a new classified pattern involves the creation of a bin for a histogram, then a bin is added to the histogram according to this attribute so that h_i^j is incremented and the histograms' borders are updated as follows:

$$\begin{cases} x_i^j < Min_i^j \Rightarrow Min_i'^j = Min_i^j - \Delta_i^j \\ x_i^j > Max_i^j \Rightarrow Max_i'^j = Max_i^j + \Delta_i^j \\ h'^j_i = h_i^j + 1 \end{cases} \quad (6)$$

2.2. Monitoring step

The purpose of the monitoring step is to detect gradual and abrupt changes in the classifier environments. A gradual change represents a drift of a class of interest. This drift can be a departure of a device from normal function to a failure. During this departure, the device begins to malfunction until the failure takes over completely. An abrupt change can concern the occurrence of a new class. Anomaly in a process function or a human error is an example of abrupt changes.

The key problem of the monitoring process is to distinguish between variations due to stochastic perturbations and variations caused by unexpected changes in a system state. If the sequence of observations is noisy, it may contain some inconsistent observations or measurements errors (outliers) that are random and may never appear again. Therefore, it is reasonable to monitor a system and to process observations within time windows in order to average and reduce the noise influence. Moreover, the information about possible structural changes within time windows can be interpreted and processed more easily. As a result, a more reliable classifier update can be achieved by monitoring within time windows.

Changes can lead to a change in: classes regions (boundaries) in the feature space, the number of classes, the feature relevance of a class, the prevalence of a class, its conditional PDF, etc. Thus to monitor these changes, several statistical measures can be used. They are generally divided into two categories. In the first category, the true class label of the new incoming patterns must be known in advance. This information is often unavailable in particular in the case of streaming data. The second category detects a change by analyzing the membership values of new incoming patterns or their distribution in the feature space. Thus, no need to their true class labels to be known a priori. For this reason, we have selected a statistical measure belonging to the second category. This measure is the change in the class conditional PDF.

The conditional PDF of each class represents patterns distribution in this class. Thus, conditional PDF describes a certain structure of the data in a class. If the class distribution, obtained after classification of new patterns in the current time window, differs from the one of previous time windows, this may indicate a change in the underlying class structure.

Let P_{iOld} be the conditional PDF of a class C_i learned using the patterns of the previous time window. Let P_{iNew} be the conditional PDF of the class Ci using only the new assigned patterns in C_i. There are various measures that are used to compare two PDFs [17]. They measure the geometric distance between two PDFs. We have selected the following distance measure, based on the use of SØrensen distance to measure the change in conditional PDF of the class C_i :

$$I = \max_{j\in\{1,..,d\}} \left(\frac{\sum_{k=1}^{k=h_{Old}+h_{New}} \left| P_{iOld} - P_{iNew} \right|}{\sum_{k=1}^{k=h_{Old}+h_{New}} \left| P_{iOld} + P_{iNew} \right| = 2} \right),\ 0 \le I \le 1 \quad (7)$$

The reason of this choice is to take into account with equal weights all the differences between the bins of the tow PDFs. If I is equal to zero, then the new patterns do not carry any change to the class to which they are assigned. While I equal to 1 indicates a complete change (class shift) in the class PDF carried by the new assigned patterns. Indeed in this case, a shift in the feature values, i.e. shift in the feature range, took place because of the new assigned patterns. Thus, their conditional PDF is defined using new histogram bins.

2.3. Updating step

The updating step aims at reacting to the changes detected by the monitoring step. It uses a mechanism to update the existing classifier according to the detected changes in order to preserve its validity and performance over time.

Two types of changes exist: slight and abrupt changes. The slight change may represent a drift. However, in order to distinguish between a drift and a natural variation, because of noises or other stochastic perturbations, the slight changes are accumulated until they reach a predefined threshold. All the patterns contributing to theses accumulated changes are used to update the classifier parameters (membership functions). The case of abrupt change corresponds to the occurrence of new class (new state). In this case, new patterns are not assigned to any of the known classes because they have zero membership values (membership rejection). However, this new class is not yet validated. A sufficient number of new patterns must be absorbed by this new class. The goal is to distinguish between the outliers and the patterns carrying the information about the occurrence of a new class.

The time window length Δt_s is determined by two parameters: th_1 and th_2. th_1 is the smallest membership value used in the previous time window to assign a pat-

tern to the class C_i. If a new pattern is assigned to this class with a membership value less than th_1, then this pattern will be assigned to a temporary set called evolving set. th_2 determines the end of the drift. This drift ends when the new patterns do not carry any new information, i.e., they have the same characteristics as the ones of the evolving set. Thus, when the new incoming patterns do not any more change one of the two monitoring measures, this means that the drift is ended.

When a pattern is rejected, i.e. it has a zero membership value to all known classes, it is considered as the prototype of a new class and not as a drift of one of the known classes.

At the end of the drift, only the patterns of the evolving set will be used to learn the evolved, or new, class membership function. This learning is achieved online as follows. Let $x = (x^1, x^2, .., x^d) \in \mathbb{R}^d$ be a given pattern vector in a feature space constituted of d parameters or attributes. The time window starts when this pattern is rejected or assigned to a class with a membership value less than th_1. In this case, this pattern is considered as a prototype of a temporary new class C_{temp}. If x is located in the bin $b_k^j, k \in \{1,2,..,h\}$, then the probability histogram of C_{temp} according to the attribute j is: $p^j_{C_{temp}} = \left\{ p^j_{C_{temp}1} = 0, p^j_{C_{temp}2} = 0, .., p^j_{C_{temp}k} = 1, .., p^j_{C_{temp}h} = 0 \right\}$. The possibility histogram will then be computed using (2). Let $C = \{C_1, C_2, .., C_c\}$ be the set of learned classes at present. Let x' be a new incoming pattern which is assigned to C_{temp}. The probability histograms of C_{temp} after the assignment of this new pattern will be updated in an incremental manner using (2). At the end of the drift, the classifier structure will be adjusted as follows: $\left\{ p_c^j \right\} \leftarrow \left\{ p^j_{C_{temp}} : j \in \{1,..,d\} \right\}; \{C_c\} \leftarrow \{C_{temp}\}; c \leftarrow c+1$.

2.4. Diagnostics step

This step aims at interpreting the detected changes in a classifier parameters and structure. This interpretation is then used as a short-term prognosis about the tendency of the future development of the current situation. This prognosis is useful to formulate a control action to modify the dynamics of a system. For instance, let suppose that we have two classes A and B. Let suppose that class A represents the normal function state of a system while class B is a fault state. When the diagnostics step provides the result 'The system state has been moved away from class A and is approaching class B'. This means that the system needs to be repaired, adjusted or reconfigured. The goal is to inverse its tendency, to move towards a fault state, by forcing it to return to the normal function state. In addition, this step may provide the Remaining Useful Life (RUL) of a system before the failure. RUL is used in Condition-based Maintenance (CBM) to schedule required repair and maintenance actions prior to breakdown (failure state). Let us take the example of Fig 1 showing a fault propagation case. If one catches the fault at 4 percent severity, one needs replace only the component. If the fault is not caught until 10 percent severity, the subsystem must be replaced, and at failure, the entire system must be replaced [6]. Thus, predictions about fault severity and impending failures are essential.

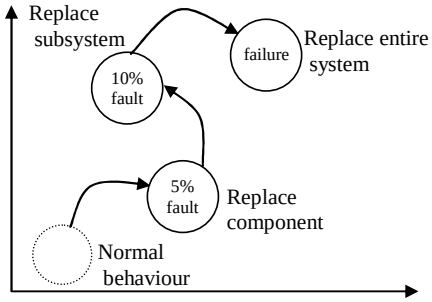

Fig. 1: Propagation of a fault and its required maintenance actions [6].

3. Experimental results

3.1. Presentation of the application

We apply the proposed approach on the acoustic signals recorded by a sensor in response to an injection command (Fig 2). The sensor records background noises resulting of the injection of argon in the steam generator of the nuclear power generator Prototype Fast Reactor (PFR). This injection simulates a fault occurred by a leakage in the steam generator. The latter switches between the two functioning modes (normal: non-injection and faulty: argon-injection) in several time instants as it is shown in Fig. 2. The signal is sampled at the frequency 2048 Hz.

The available acoustic signals were recorded on the steam generator from the end-of-life of PFR at United Kingdom. In these experiments, argon was injected into sodium and acoustic noises were measured. Indeed, experimental results have shown that steam and argon injections give similar acoustic noise output at a given mass flow rate [5].

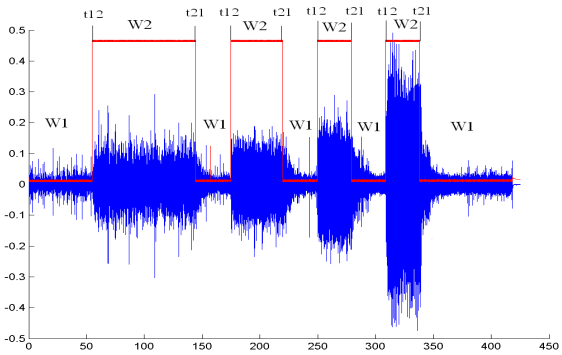

Fig. 2: Acoustic signal in response to Argon command signal. W1: non-injection class, W2: injection class, t12: time of switching from W1 to W2, t21: time of switching from W2 to W1.

3.2. Feature space construction

In order to construct the dynamic classifier, the characteristic features, or mode conditions indicators, must be extracted and selected. The goal is to detect as soon as possible a change in the conditions of a func-

tioning mode and to properly estimate its parameters. The acoustic signal is considered as the output of a switched AutoRegressive (AR) system. In statistics and signal processing, an AR (*p*) model is a type of random process which is often used to model and predict various types of systems [8]. In order to capture this change, these features are calculated during a sliding time window (Fig. 3). The sliding window size must include a sufficient number of data points in order to properly estimate the parameters of each mode. We have tested several sizes of time window. We have selected the one which maximizes the discrimination power between the different modes in the feature space, i.e. obtaining compact and separated classes. This experimentation leads to select a sliding window with an initial length = 8192 data points and a shift length equal to 2048 data points. Therefore, to define a pattern in the feature space, a time window containing 8192 data points is required. Consequently, to determine the functioning mode (injection or non-injection), a delay time of 4 seconds is needed since the sampling frequency is equal to 2048 Hz.

It is useful to reduce the feature space defined by the coefficients of the dynamic parametric model AR (*p*). The reduction operation aims at keeping the distinguishing features leading to separate as well as possible the different classes. Thus, the reduction operation improves the modes estimation and the classification accuracy and reduces the time decision (classification) of new patterns. The feature space reduction can be performed using feature extraction and selection methods. We used the Principal Component Analysis (PCA) to extract from the set of features the ones which are uncorrelated. Then, we selected from this set of independent features the ones which have a combination leading to obtain the lowest error of classification. Two independent and discriminative AR model coefficients were conserved to define the feature space.

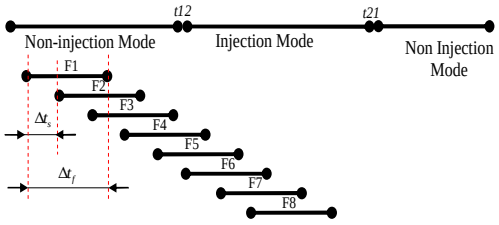

Fig 3: Estimation of the feature space parameters using a sliding time window.

Fig 4 shows the two classes, corresponding to non-injection and injection modes, in the conserved feature space in response to the first argon injection.

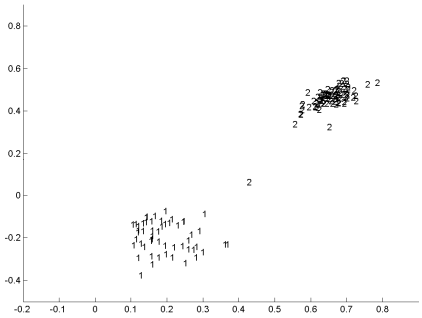

Fig. 4: Classes corresponding to non-injection and injection modes for the first argon injection in the feature space constituted of the two conserved AR model coefficients.

Fig 5 shows the classes corresponding to injection and non-injection modes in response to the last argon injection command of Fig 2. We can observe that the both classes drift. This drift leads to decrease the separability between the both classes and thus to decrease the classifier performance (misclassification error).

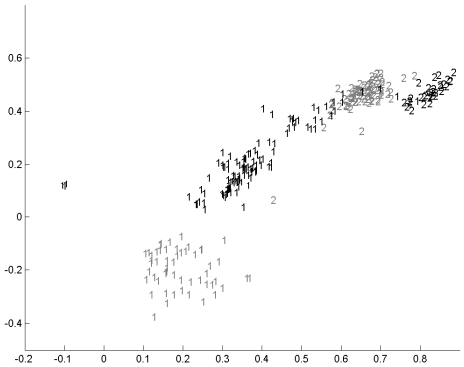

Fig. 4: Classes, corresponding to non-injection and injection modes, for the first argon injection (grey points) and the last argon injection (in black) in the feature space constituted of the two conserved AR model coefficients.

3.3. Obtained results

Table 1 shows the PDF change measure, calculated using (7), for both classes (non-injection and injection) in response to the successive argon injections of Fig. 2. We can observe that this measure justifies the classifier update after each new injection. Table 2 shows the misclassification rate in the case of using an incremental classifier (without forgetting and updating) and the proposed dynamic classifier. We can observe that the performance (quantified by the number of misclassified patterns) of the incremental classifier continue to decrease after each new drift. While the performance of dynamic classifier is improved after each drift thanks to its parameters updating in response to a drift. This classifier updating leads to forget the patterns becoming useless and thus to increase the separability between classes.

Argon injection number	PDF change measure for class 1	PDF change measure for class 2	Updating
2nd	1	0.28	Class 1
3rd	1	0.8	Classes 1 and 2
4th	0.93	0.96	Classes 1 and 2

Table 1: PDF change measure in response to the successive argon injections of Fig 2.

Argon injection number	Misclassification error	
	Incremental classifier	Dynamic classifier
2nd	19	19
3rd	33	8
4th	58	4

Table 2: Misclassification rate calculated for incremental classifier (without forgetting and updating) and dynamic classifier according to the successive argon injections of Fig 2.

4. Conclusion

In this paper, an approach to learn a classifier and to update its parameters and structure according to the changes in its environment conditions is proposed. This approach detects the changes in the classifier environment by observing the changes in its conditional probability density function during a sliding time window. When a significant change is detected, this approach updates the classifier using an incremental learning rule.

We are looking to develop an ensemble classifier approach for the learning in dynamic environments. This approach uses different classification methods to build an ensemble of classifiers. Each classifier is an expert for the classification of patterns in a particular region of the feature space. In addition, each classifier will be adapted for the monitoring of a particular type of changes (gradual, abrupt and recurring).

Acknowledgment

This work is supported by the Scientific Interest Group surveillance, safety and security of the big systems (GIS3SGS).

References

[1] B. D. Ripley, Pattern Recognition and Neural Networks, *Cambridge University Press*, Cambridge. 1996.

[2] C.W. Therrien, Decision Estimation and Classification: An Introduction to Pattern Recognition and Related Topics, *John Wiley & Sons*, New York, 1989.

[3] E. Parzen, On the estimation of a probability density function and mode, *Annals of Mathematical. Statistics*, 33, 1065-1076, 1962.

[4] G. Nakhaeizadeh, C. Taylor and G. Kunisch, Dynamic Supervised Learning. Some Basic Issues and Application Aspects. *Classification and Knowledge Organization (R. Klar et O. Opitz (Eds.))*, Springer Verlag, Berlin, Heidelberg, 123-135, 1997.

[5] G.S. Srinivasan, Om Pal Singh and Prabhakar R, Leak noise detection and characterization using statistical features, Annals of Nuclear Energy, Vol. 27, n° 4, 2000, pp 329-343.

[6] G. Vachtsevanos, F. L. Lewis, M. Roemer, A. Hess and B. Wu, Intelligent Fault Diagnosis and Prognosis for Engineering Systems, *John Wiley & Sons*, New York, 2006.

[7] H. Frigui and R. Krishnapuram, Clustering by competitive agglomeration, *Pattern Recognition*, 30 (7), 1109-1119, 1997.

[8] H. Akaike, A new look at the statistical model identification, *IEEE Trans Autom Cont.*, 19, 716-723, 1974.

[9] L. Angstenberger, Dynamic Fuzzy Pattern Recognition. *Dissertation, Fakultät für Wirtschaftswissenschaften der Rheinisch-Westfälischen Technischen Hochschule*, Aachen, Germany, 2000.

[10] L.I. Smith, A Tutorial on Principal Components Analysis. *Cornell University*, USA, 2002.

[11] L. Cohen, M. Last and G. Avrahami, Incremental Info-Fuzzy Algorithm for Real Time Data Mining of Non-Stationary Data Streams, *TDM Workshop*, Brighton UK, 2004.

[12] M. Last, Online classification on non stationary data streams, *Intelligent Data Analysis*, 6 (2), 129-147, 2002.

[13] M. Sayed Mouchaweh, A. Devillez, G. Villermain Lecolier and P. Billaudel, Incremental learning in fuzzy pattern matching, *Fuzzy Sets and Systems*, 132 (1), 49-62, 2002.

[14] P.P Angelov, A fuzzy controller with evolving structure, *Information Sciences*, 161 (1-2), 21-35, 2004.

[15] R.O. Duda, P. E. Hart and D. G. Stork, Pattern Classification 2nd edition. *Wiley-Interscience*, 2001.

[16] R.R. Yager, An extension of the naive Bayesian classifier, *Information Sciences*, 176 (5), 577-588, 2006.

[17] S.H. Cha, Comprehensive survey on distance/similarity measures between probability density functions, *International Journal of Mathematical Models and Methods in Applied Sciences*, 4, 300–307, 2007.

[18] S. Medasani, K. Jaeseok and R. Krishnapuram, An overview of membership function generation techniques for pattern recognition, *International Journal of Approximate Reasoning*, 19, 391-417, 1998.

[19] T.M. Cover and P.E. Hart, Nearest neighbor pattern classification, *IEEE Transaction Information Theory*, 13, 21-27, 1967.

[20] V. Vapnik, Statistical Learning Theory, *John Wiley & Sons Inc.*, New York, 1998.