DOI: 10.1109/EAIS.2013.6604105
Abstract—Humans often seek a second or third opinion about an important matter. Then, a final decision is reached after weighing and combining these opinions. This idea is the base of the ensemble based systems. Ensembles of classifiers are well established as a method for obtaining highly accurate classifiers by combining less accurate ones. On the other hand, evolving classifiers are inspired by the idea of evolve their structure in order to adapt to the changes of the environment.

In this paper, we present a proof-of-concept method for constructing an ensemble system based on Evolving Fuzzy Systems. The main contribution of this approach is that the base-classifiers are self-developing (evolving) Fuzzy-rule-based (FRB) classifiers. Thus, we present an ensemble system which is based on evolving classifiers and keeps the properties of the evolving approach classification of streaming data. It is important to clarify that the evolving classifiers are gradually developing but they are not genetic or evolutionary.

I. INTRODUCTION

To make a good decision about an important matter, humans often seek a second opinion, a third opinion, or even any more. Then, these opinions are usually weighted and combined in order to reach a final decision. This process improves our confidence that we are making the right decision. The idea underlying this process is the base of the ensemble based systems. These systems are also known as multiple classifier systems, committee of classifiers or mixture of experts [1].

An ensemble of classifiers is defined by Ditterich [2] as a set of classifiers whose individual decisions are combined in some way (typically by weighted or unweighted voting) to classify new examples. The strategy in ensemble systems is therefore to create a collection of individual accurate classifiers, and combine their outputs such that the combination improves upon the performance of a single classifier. This requires, however, that individual classifiers make errors on different instances, that is, that ensemble members are diverse.

There are many different researches which propose different methods for constructing good ensembles of classifiers. The researches in this field come to the conclusion that ensembles are often much more accurate than the individual classifiers that make them up. According to Ditterich [2], ensembles can improve performance because uncorrelated errors made by individual classifiers can be removed by voting. However, there are still many questions about the best way to construct ensembles as well as issues about how best understand the decisions made by ensembles.

Two of the most well known methods to construct ensembles of classifiers are: Bagging [3] and Boosting [4]. Bagging builds sets of data of the same size of the original data set by applying random sampling with replacement. Boosting also resamples original data set with replacement, but the training set used for each member of the ensemble is chosen based on the performance of the earlier classifier(s) in the set. That is, in Boosting, examples that are incorrectly predicted by the current ensemble are chosen more often than examples that were correctly predicted.

In this paper, we present a proof-of-concept ensemble method in which all the individual classifiers are Evolving Fuzzy-rule-based Classifier (eClass).

An eClass is a fuzzy rule-based classifier with rules that are evolved from streaming data. Thus, the use of an eClass does not need to know beforehand in how many classes the data will be classified as new classes can be introduced during the learning process [5]. An eClass learns new rules from new data gradually preserving/inheriting the rules learned already. Thus, eClass can be defined as a self-developing classifier which has both their parameters but also (more importantly) their structure self-adapting on-line. In addition, an eClass can start learning "from scratch".

It is important to clarify that an eClass is not a genetic or evolutionary classifier. A genetic/evolutionary FRB classifier [6], [7] is a set of input/output rules that are modified by a genetic algorithm. In that case, new rules are generated as a crossover or mutation of previous rules. However, an eClass learns new rules from new data gradually preserving the rules already learned.

eClass has been applied to a wide range of problems, both benchmarks and real. The main contribution of this paper is the use of evolving classifiers in an ensemble system. Although there are several evolutionary classifier ensembles
II. BACKGROUND AND RELATED WORK

A. Ensemble of Classifiers

One of the first works on ensemble systems was proposed in 1979 [10]. In this early work, the authors propose the partition of the feature space using two or more classifiers. In 1990, a new research [11] concluded that the generalization performance of a neural network can be improved using an ensemble of neural networks. Bagging [3] and Boosting [4] are two of the most well-known ensemble learning methods due to their theoretical performance guarantees and strong experimental results.

The literature of ensemble classifiers is truly vast. There are many different approaches which propose different methods for generating individual classifiers, and strategies for combining the outputs of these classifiers. Some of these approaches were defined as combination of multiple classifiers [12][13], dynamic classifier selection [14], classifier fusion [15], classifier ensembles [16] and many others.

There are many different areas in which ensemble of classifiers have been used, and these areas are continuously growing. As it is described by Polikar [1], some of the more promising areas include using ensemble systems in non-stationary environments [16] or in clustering applications [17], [18]. Some of the practical applications in which these ensemble systems are rapidly growing are: biomedical [19], financial [20], remote sensing [21], or chemical [22].

B. Evolving Fuzzy Classifiers: eClass

eClass (evolving Classifier) family [5] is a set of evolving neuro-fuzzy classifiers which take its roots in evolving Takagi-Sugeno (eTS). The first evolving fuzzy rule-based systems are introduced in [23] and further developed in [24].

During training in these classifiers, a set of fuzzy rules that describes the most important features for the classification of each class is formed. These rules are constantly adjusted to the available training data. One of the advantages of eClass is that it does not require parameter optimization as its only parameter ‘scale’ can be directly inferred from the training data. This technique [25] is based on partitioning the data space into overlapping local regions through Recursive Density Estimation (RDE) and associating clusters (respectively fuzzy sets) to them.

As it is explained in [5], the main differences between eClass family and a conventional Fuzzy Rule-Based (FRB) classifier are:

- the open structure of the rule-base: eClass self-develops on-line starting from scratch, while in a conventional FRB classifier it is determined offline and then fixed.
- the online learning mechanism which takes into account this flexible rule-base structure.

eClass family includes two different architectures and on-line learning methods:

- eClass0 with the classifier consequents representing class label.
- eClass1 for regression over the features using first order eTS fuzzy classifier.

eClass family has been applied in many different areas with a great success. An important area of application of these evolving classifiers is robotics and autonomous systems. Some examples of application include: autonomous landmark recognition [26], self-localization and mapping [27], object detection and tracking [28][29], collision avoidance [30], activity recognition from sensor streams [31][32] and user modeling [33][34]. In this research, eClass0 will be applied as base-classifier in the proposed ensemble.

C. Reasons for using ensemble of evolving classifiers

According to Polikar [1], there are several theoretical reasons why an ensemble system is preferred:

1) Statistical Reasons: A set of classifiers with similar training performances may have different generalization performances.

2) Large Volumes of Data: Training a classifier with vast amount of data is usually not practical. Training different classifiers with different partitions of data and combining their outputs proves to be a more efficient approach.

3) Too Little Data: In the absence of adequate training data, resampling techniques can be used for drawing overlapping random subsets of the available data.

4) Divide and Conquer: The complex decision boundary could be approximated by an appropriate combination of different classifiers.
5) Data Fusion: Ensemble based approaches have successfully been used for applications in which data from different sources are combined.

In the method that we propose in the next section, as we are using evolving classifiers, they can be generated on-line from streaming data. As evolving classifiers can cope with huge amounts of data, the use of large volumes of data or little data is not a reason for using ensembles in our case. However, the other reasons (especially, statistical and “divide an conquer” reasons) proposed by Polikar are quite interesting and we would like to check if using evolving classifiers these reasons are also true.

III. ENSEMBLE METHOD BASED ON EVOLVING CLASSIFIERS

This section introduces the proposed ensemble method based on evolving classifiers, which will be named as eEnsemble. The section is divided in two parts: First, we will detail eClass0, the evolving classifier used for the individual classifiers. Then, the architecture of the proposed ensemble method will be explained in detail.

A. Evolving Classifier: eClass0

eClass0 possesses a zero-order Takagi-Sugeno consequent, so a fuzzy rule in the eClass0 model has the following structure:

\[ \text{Rule}_i = IF (X_1 \text{ is } P_1) \ AND \ldots \ AND (X_n \text{ is } P_n) \ THEN \ Class = \ Class_i \]  

(1)

where \( i \) represents the number of rule; \( n \) is the number of input variables; the vector \( X \) stores the input variables and the vector \( P \) stores the values of the features of one of the prototypes (cluster centre) of the corresponding class. \( \text{Class} \in \{ \text{set of different classes} \} \). 

The eClass0 model is composed of several fuzzy rules per class (the number of rules depends on the heterogeneity of the input data of the same class). During the training process, a set of rules is formed from scratch using an evolving clustering approach to decide when to create new rules. The inference in eClass0 is produced using the “winner takes all” rule and the membership functions that describe the degree of association with a specific prototype are of Gaussian form. The potential (Cauchy function of the sum of distances between a certain data sample and all other data samples in the feature space) is used in the partitioning algorithm. However, in these classifiers, the potential (P) is calculated recursively (which makes the algorithm faster and more efficient). The potential of the \( k^{th} \) data sample \((x_k)\) is calculated [5] by the equation 2. The result of this function represents the density of the data that surrounds a certain data sample.

\[ P(x_k) = \frac{1}{1 + \sum_{i=1}^{k-1} \text{distance}(x_k, x_i)} \]  

(2)

where distance represents the distance between two samples in the data space.

The potential can be calculated using the euclidean or the cosine distance. In this case, cosine distance (\( \text{cosDist} \)) is used to measure the similarity between two samples; as it is described in equation 3.

\[ \text{cosDist}(x_k, x_p) = 1 - \frac{\sum_{j=1}^{n} x_{kj} x_{pj}}{\sqrt{\sum_{j=1}^{n} x_{kj}^2} \sum_{j=1}^{n} x_{pj}^2} \]  

(3)

where \( x_k \) and \( x_p \) represent the two samples to measure its distance and \( n \) represents the number of different attributes in both samples.

Note that the expression in the equation 2 requires all the accumulated data sample available to be calculated, which contradicts to the requirement for real-time and on-line application needed in the proposed problem. For this reason, in [5] it is developed a recursive expression cosine distance.

All details about the eClass0 model and the learning algorithm can be found in [25].

B. Architecture of the proposed method: eEnsemble

The ensemble that we propose in this paper is composed of a number of more simple classifiers (individual or base-classifiers). The number of these individual classifiers (\( k \)) must be defined at the beginning by hand. All of these \( k \) classifiers have the same eClass0 structure, but they will be trained with different (and disjoint) sets of data.

The architecture of the proposed method is shown in Figure 1 and it can be divided in three steps: Receiving the data streams, creating the individual classifiers and combining individual classifiers. These parts are explained in the following subsections.
1) Receiving the data streams: Unlike most of the ensembles of classifiers, the method that we propose has the following characteristic:

- The classification is online,
- The inputs of the method are streaming data.

As streaming data arrive continuously, there is not an initial data set. In this case, when a new sample arrives, it is sent randomly to one of the $k$ individual classifiers. This on-line partitioning is done without taking into account any variable or technique; it is done entirely randomly. Thus, the examples are randomly distributed across the set of individual classifiers.

2) Creating the Individual Classifiers: Once an example is received by an individual classifier, it is analyzed using eClass0. As we already explained before, this classifier is designed with an evolving (self-developing) FRB structure. Also, this classifier can start “from scratch” and it is no need to define any variable or parameter for the different classifiers.

It is important to highlight that the (flexible) rule-base structure of the individual classifiers will be different since the data streams that they receive are also different. Thus, the number of rules and the focal points of the fuzzy rules will be different in the individual classifiers. This is the main characteristic of the proposed method.

3) Combining Individual Classifiers (Combination Module): Once the individual classifiers have been trained with example data streams, we can use the proposed eEnsemble in order to classify a new example/instance. In this case, the new example will be tested by all the classifiers and their output will be combined in order to obtain a final classification. This task can be achieved by using different methods: Linear functions like average function [35]; non-linear combination methods, like majority voting (Bagging, Boosting) or meta-learning methods [36] [37].

In the ensemble that we propose in this paper, we will take the simplest approach: unweighted vote. The outputs of the individual classifiers will be combined by taking a majority vote of their classification. For any given instance, the class chosen by most individual classifiers is the ensemble classification. This method is also used in Bagging [3].

Although this is a very simple method, the experience in the forecasting literature has been that simple, unweighted voting is robust [38]. However, this module could be changed without changing the base idea of the proposed ensemble.

C. eEnsemble variation: eEnsembleR

One of the characteristic of Bagging is that it creates the ensembles by repeatedly randomly resampling the training data. However, in eEnsemble, an instance will be only the input of a specific base-classifier (randomly chosen). For this reason, in order to take into account that an example could be the input of several base-classifiers, we propose a variation named eEnsembleR.

The only difference between eEnsemble and eEnsembleR is that in this last method, when an instance arrives, it will go (randomly) to one of the base-classifiers or to all of them. In this case, each instance has the probability of 1/2 to be repeated in all of the base-classifiers. This version can give us an idea about how the addition of the same instance in different individual classifiers can vary the result of the proposed method.

IV. EXPERIMENTAL SETUP AND RESULTS

The two proposed methods (eEnsemble and eEnsembleR) have been tested on some benchmark problems from the UCI machine learning repository [39]. To apply eEnsemble to these data sets, we consider them as pseudo-online streams. Thus, although we are using data sets in an offline mode, the proposed method is designed to be used with data streams. The reason for this experiments is to achieve comparable results and obtain further validation of the proposed ensembles.

A. Benchmark Problems from UCI Repository

The different datasets used in this experimentation are detailed as follows:

1) Wine Data Set: These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the 3 types of wines.

2) Cardiotocography (CGT) Data Set: This dataset contains 2126 fetal cardiotocograms (CTGs) which were automatically processed and the respective diagnostic features measured.

3) Waveform Database Generator (V1) Data Set: This dataset describes 3 classes of waves which are described by 21 attributes and all of the instances include noise.

4) Page Blocks Data Set: This is a set of 5473 examples which come from 54 distinct documents. Each observation concerns one block.

5) Statlog (Landsat Satellite) Data Set: This dataset consists of the multi-spectral values of pixels in 3x3 neighborhoods in a satellite image, and the classification associated with the central pixel in each neighbourhood.

6) Pen-Based Recognition of Handwritten Digits Data Set: This data set contains the handwriting information from a pressure sensitive tablet with an integrated LCD display. This information is classified in ten digits (from 0 to 9).

7) Statlog (Shuttle) Data Set: This data set contains 58000 instances (with 9 attributes) which are classified in 8 different classes.

The data set size and other characteristic of the data sets are listed in Table I. Note that the number of instances is very different in each dataset.
<table>
<thead>
<tr>
<th>DataSet</th>
<th>Number of Instances</th>
<th>Number of Attributes</th>
<th>Number of Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wine</td>
<td>178</td>
<td>13</td>
<td>3</td>
</tr>
<tr>
<td>Cardiotocography</td>
<td>2126</td>
<td>23</td>
<td>11</td>
</tr>
<tr>
<td>Waveform</td>
<td>5000</td>
<td>21</td>
<td>3</td>
</tr>
<tr>
<td>Page Blocks</td>
<td>5473</td>
<td>10</td>
<td>6</td>
</tr>
<tr>
<td>Landsat</td>
<td>6435</td>
<td>36</td>
<td>8</td>
</tr>
<tr>
<td>Pen Digit</td>
<td>10992</td>
<td>16</td>
<td>10</td>
</tr>
<tr>
<td>Shuttle</td>
<td>58000</td>
<td>9</td>
<td>8</td>
</tr>
</tbody>
</table>

**B. Experimental Setup**

As we already mentioned, before applying eEnsemble, it is necessary to parameterize the number of individual classifiers (represented as $k$) that will make up the ensemble. As the result of the eEnsemble varies according to this parameter, we have used three different values for $k$: 3, 5 and 10.

The performance of the ensemble that we propose depends on how the data stream is distributed among the individual classifiers. As this distribution is randomly done, the result of the eEnsemble can change because of this aspect. For this reason, the results shown in this section are the average of 100 different executions of eEnsemble.

Finally, in order to evaluate the performance of eEnsemble, it is compared with the eClass0. Also, to measure the performance of the different methods, for each data set, 80 percent of the data are kept as training instances while the rest are used as the pool of test examples.

The ensemble eEnsembleR has been evaluated with the same experimental setup described previously.

**C. Experimental Results**

Table II shows the rate for eClass0 and the average rate for eEnsemble and eEnsembleR in any of the 7 different datasets. Major rows correspond to the 7 different datasets and columns show the results of the experiment for each model (and also for the number of individual classifiers ($k$) in the case of the proposed ensembles). According to the results, we can conclude that, in general, the results of the proposed ensembles are comparable with those obtained from eClass0. However, in some cases, the standard deviation is high what represents that the ensemble created is not very stable.

Finally, it is very important to remark the importance of the standard deviation in this case. As we can observe, in some ensembles, this value is very variable. However, in many cases this value is very small what indicates that the proposed ensembles are quite stables in those cases.

**V. CONCLUSIONS AND FUTURE WORK**

In this paper, we have presented and evaluated a proof-of-concept method for constructing ensembles based on individual evolving classifiers. These ensembles (named eEnsemble and eEnsembleR) are based on evolving classifiers and they keep the properties of the evolving approach classification of streaming data. According to the experimental results, we can conclude that, in general, the results of the proposed ensembles are comparable with those obtained from eClass0. However, in some cases, the standard deviation is high what represents that the ensemble created is not very stable.

In order to solve that problem, we could apply some technique (for example, Genetic Algorithms) to decide the best configuration (especially, the value of $k$) of the ensemble taking into account the nature of the data. Furthermore, evolving systems could be used to evolve the configuration of the ensemble according to the received data. Our main future work is the use of Evolving Systems for creating a stable ensemble which can adapts/evolves its structure on-line and in real time.

**ACKNOWLEDGMENT**

This work has been supported by the Spanish Government under i-Support (Intelligent Agent Based Driver Decision Support) Project (TRA2011-29454-C03-03).

**REFERENCES**

TABLE II: Experimental results: eClass0 vs. eEnsemble and eEnsembleR

<table>
<thead>
<tr>
<th>Dataset</th>
<th>eClass0 (rate %)</th>
<th>eEnsemble</th>
<th>eEnsembleR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our approaches - rate % (average)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSD</td>
<td>93.58 ± 2.8</td>
<td>73.54 ± 7.1</td>
<td>93.58 ± 2.8</td>
</tr>
<tr>
<td>Shuttle</td>
<td>92.98 ± 3.2</td>
<td>68.20 ± 4.0</td>
<td>92.98 ± 3.2</td>
</tr>
</tbody>
</table>


