



Personalizable Pen-Based Interface Using Lifelong Learning

Abdullah Almaksour Eric Anquetil
*INSA de Rennes/IRISA, Avenue des Buttes de Coesmes,
F-35043 Rennes, France
abdullah.almaksour@irisa.fr; eric.anquetil@irisa.fr*

Solen Quiniou Mohamed Cheriet
*Synchromedia-ETS, 1100 rue Notre-Dame Ouest,
Montreal (Quebec) H3C 1K3, Canada
solen.quiniou@synchromedia.ca, mohamed.cheriet@etsmtl.ca*

Abstract

In this paper, we present a new method to design customizable self-evolving fuzzy rule-based classifiers. The presented approach combines an incremental clustering algorithm with a fuzzy adaptation method in order to learn and maintain the model. We use this method to build an evolving handwritten gesture recognition system, that can be integrated into an application to provide personalization capabilities. Experiments on an on-line gesture database were performed by considering various user personalization scenarios. The experiments show that the proposed evolving gesture recognition system continuously adapts and evolve according to new data of learned classes, and remains robust when introducing new unseen classes, at any moment during the lifelong learning process.

1. Introduction

Applications with personalization capabilities become more and more of interest, to enable users to have applications that adjust to their needs and not the other way. Indeed, for a given application, various users may not all have the same set of interaction mechanisms (depending on a hierarchy of users, for example) or they may want to use them differently (for example, using a different mechanism to perform the same action). It is even more the case when using pen-based interfaces (like whiteboards or Tablet PCs) that provide handwritten interaction modalities. In such interfaces, each user writes and draws with his own style. Being able to adapt an initial handwriting recognition system to each user could then improve the interface usability. It can be par-

ticularly interesting in the case of handwritten gestures that are associated to functionalities of an application, used to interact with electronic documents, for example. Indeed, gestures can be drawn differently from one user to another, and users may want to add or remove gestures, as long as they use the application (without having to define *a priori* the set of gestures that will be used). To achieve that, the recognition system has to be modified during its use in the application.

In this work, we aim at building a handwriting classifier which can be incrementally adapted and that will be used to recognize gestures, in an application designed for collaborative work. The challenge is to learn new unseen gestures on-the-fly, from scratch and using very few samples. Furthermore, the classification system has to remain robust and has to maintain its knowledge about the existing gestures, when introducing new unseen ones anytime during the lifelong learning process.

An incremental learning algorithm is defined in [10] by the following criteria: it should be able to learn additional information from new data; it should not require access to the original data (*i.e.* data used to train the existing classifier); it should preserve previously acquired knowledge (it should not suffer from *catastrophic forgetting*, *i.e.* significant loss of original learned knowledge); and it should be able to accommodate new classes that may be introduced with new data. We can distinguish two main types of incremental learning algorithms: algorithms for parameter learning and algorithms for structure learning. The incremental learning of parameters can be considered as an "adaptation" algorithm. The structure in such systems is fixed and initialized at the beginning of the learning process, and the system parameters are learned incrementally according to newly available data. Some ex-

amples of these systems are presented in [8, 7]. Most of the structure incremental learning algorithms are based on the principle of the ART clustering algorithm [3], such as [11, 9]. The main problem of these systems is that they are sensitive to the selection of the vigilance parameter, to the noise level in the training data, and to the order in which the training data is presented. A promised incremental clustering approach had been presented in [2], based on the Mountain Clustering algorithm (originally introduced in [13]). The main idea of the proposed approach is that of a *potential* of a given point: it corresponds to a value representing the density in the data space at that point. The potential of a sample can be defined as the inverse of the sum of the distances between that data sample and all the other ones. Samples with high potential are then considered to be candidates to form a cluster. In this paper, we extend the recursive mountain clustering by combining it with a robust fuzzy adaptation method, and we use this hybrid algorithm to incrementally learn an evolving fuzzy rule-based classifier. Our system is used for the recognition of online handwritten gestures, and must be able to learn new classes of gestures and to evolve, sample after sample, without using all the old data. These new gestures, added by the final user, can be assigned to new commands or shortcuts in the pen-based interface, or to replace the default gestures assigned to existing commands. The advantages of our classifier are its simplicity and its adaptation ability that makes it suitable to our context, in which the system must learn rapidly and be operational and ready to be used at any moment during the lifelong learning process.

The remaining parts of this paper are as follows. In Section 2, the evolving classifier used to perform the lifelong gesture personalization is described, whereas the pen-based interface integrating it is presented in Section 3. Then, the results of the experiments of various personalization capabilities are shown in Section 4. Finally, Section 5 draws some conclusions.

2. Evolving fuzzy classifier

In this section, we first present the fuzzy classifier used. Then, we describe the incremental learning algorithm and the extension proposed, using a fuzzy adaptation method.

2.1. System architecture

Our system is based on a fuzzy rule-based classifier. The fuzzy rules make a link between intrinsic models (premises) and system outputs by consequent functions. For a K classes problem, a rule R_i is built for each

fuzzy model P_i :

$$R_i : IF \vec{x} \text{ is } P_i \text{ THEN } y_i^1 \text{ AND } \dots \text{ AND } y_i^k, \quad (1)$$

where \vec{x} is the n -dimensional feature vector, P_i is a fuzzy model (prototype) defined by a center $\vec{\mu}_i$ and a covariance matrix Q_i . The degree of membership of \vec{x} to P_i is given by the Mahalanobis distance:

$$\beta_i(\vec{x}) = 1/(1 + d_{Q_i}(\vec{x}, \vec{\mu}_i)). \quad (2)$$

For the consequent part, we can distinguish different structures [12]: (i) the zero order Takagi-Sugeno (TS) with binary consequents ($y_i^m = 1$ if P_i belongs to class m , 0 otherwise), (ii) the zero order TS with constant consequents ($y_i^m \in [0, 1]$ represents the participation of P_i in the description of class m), (iii) the first order TS, where the consequents are linear functions $y_i^m = \vec{a}_i^m \vec{x}$.

Finally, the sum-product inference is used to compute the system output for each class:

$$y^m(\vec{x}) = \sum_{i=1}^R \beta_i(\vec{x}) y_i^m. \quad (3)$$

2.2. Incremental learning algorithm

In order to incrementally learn a fuzzy rule-based classifier in an on-line manner, we need, on the one hand, to evolve its structure by adding (or deleting) rules, and, on the other hand, to adjust its parameters (prototypes' centers, covariance matrices and the consequent parameters). The emphasis in this section is first placed on the learning of the premise part of the system. We aim at extending the recursive mountain clustering by combining it with a robust adaptation method that can constantly re-center the fuzzy prototypes and re-shape their influence zones, according to each single data sample.

2.2.1 Recursive mountain clustering

As mentioned earlier in Section 1, a recursive (on-line, one-pass, non-iterative) version of the mountain clustering method was introduced in [2]. The recursive formula avoids memorizing the whole previous data but keeps - using few variables - the density distribution in the feature space, based on the previous data:

$$P_k(x(k)) = \frac{k-1}{(k-1)\alpha(k) + \gamma(k) - 2\zeta(k) + k-1}, \quad (4)$$

where $P_k(x(k))$ denotes the potential of the k^{th} data sample and with

$$\alpha(k) = \sum_{j=1}^n x_j^2(k), \quad (5)$$

$$\gamma(k) = \gamma(k-1) + \alpha(k-1), \quad \gamma(1) = 0, \quad (6)$$

$$\zeta(k) = \sum_{j=1}^n x_j(k) \eta_j(k), \quad (7)$$

where $\eta_j(k) = \eta_j(k-1) + x_j(k-1), \eta_j(1) = 0$.

Introducing a new sample affects the potential values of the centers of the existing clusters, which can be recursively updated by:

$$P_k(\mu_i) = \frac{(k-1)P_{k-1}(\mu_i)}{k-2 + P_{k-1}(\mu_i) + P_{k-1}(\mu_i) \sum_{j=1}^n \|\mu_i - x_j\|_2^2}. \quad (8)$$

If the potential of the new sample is higher than the potential of the existing centers, then this sample will be a center of a new cluster (and a new fuzzy rule will be formed in the case of a fuzzy rule-based classifier). If the high potential sample is close to an existing center $\vec{\mu}_i$, then this sample will replace $\vec{\mu}_i$ and no new cluster will be created.

2.2.2 Fuzzy vector quantization

As can be noted in section 2.2.1, the condition to have a high potential is a very hard one, and it is inversely proportional to the growing number of data. In this way, we can imagine a cluster center $\vec{\mu}_i$ which is not really in the optimal center position (according to the data history), but that remains the center because it still has the highest potential value. Therefore, the incremental clustering process of the premise part of the fuzzy classifier will not be able to take advantage of the data points that do not have a very high potential, to move (or reshape) the existing clusters. We enhance the incremental clustering process (described in section 2.2.1) by an adaptation algorithm that allows the modification of all the fuzzy prototypes, by re-centering and re-shaping them for each new data point. For this purpose, we use a fuzzy version of the Vector Quantization algorithm [4]. In this method, the farther the normalized activation β_i of the premise of the rule i is away from its objective score β_i^* , the more the prototype has to be moved:

$$\Delta \vec{\mu}_i = \lambda * (\beta_i^* - \beta_i(\vec{x})) * (\vec{x} - \vec{\mu}_i) \quad (9)$$

where the adaptation parameter λ lies between 0 and 1. The objective score β_i^* is 1 if the prototype P_r and \vec{x} belong to the same class and 0 otherwise. In the same way, a fuzzy recursive formula is given in [5], to update the inverse of the covariance matrix as follows:

$$Q_i^{-1} \Leftarrow \frac{Q_i^{-1}}{1 - \alpha \delta_i} - \frac{\alpha \delta_i}{1 - \alpha \delta_i} \cdot \frac{(Q_i^{-1} \vec{d}) \cdot (Q_i^{-1} \vec{d})^T}{1 + \alpha \delta_i (\vec{d}^T Q_i^{-1} \vec{d})} \quad (10)$$

$$\delta_i = \beta_i^* - \beta_i(\vec{x}) \quad (11)$$

where $\vec{d} = \vec{x} - \vec{\mu}_r$ and α lies between 0 and 1.

2.2.3 Learning algorithm

The incremental learning algorithm of the consequent parameters depends on the type of the fuzzy system used. If we considered the three structures mentioned in Section 2.1: (i) no consequent learning is needed for the simple structure with binary consequents; (ii) for the second structure, an online estimation of the constant consequents is presented in [6]; and (iii) the linear consequents learning problem in a first-order TS can be solved by the weighted Recursive Least Square method (wRLS)[2]. The complete learning algorithm can be summarized by Algorithm 1.

Algorithm 1: Online incremental learning algorithm

```

foreach new sample  $\vec{x}$  do
  if  $\vec{x}$  is the first sample of a new class then
    add a new fuzzy prototype centered on  $\vec{x}$ 
    to the system; let its potential be 1;
  else
    calculate the potential of  $\vec{x}$  by [4];
    update the potentials of the existing
    prototypes centers using [8];
    if  $P(\vec{x}) > P_k(\vec{\mu}_i) \quad \forall i \in [1, R]$  then
      if  $\vec{x}$  is close to a center  $\vec{\mu}_i$  then
        let  $\vec{x}$  be the center of the prototype
         $P_i$ ;
      else
        add a new fuzzy prototype
        centered on  $\vec{x}$  to the system; let its
        potential be 1;
      end
    else
      apply premise adaptation according to
       $\vec{x}$  by [9] and [10];
    end
  end
  update the consequent parameters;
end

```

3. Personalizable pen-based interface using gestures

In this section, we present the application integrating the incremental learning classifier as well as the gesture-based personalization capabilities considered.

3.1. Context

This work is part of the ICIOS project, which is a project between the Sychromedia laboratory of the Ecole de Technologie Supérieure (ETS) in Montreal, Canada, and the Imadoc team project of the Institut de Recherche en Informatique et Systèmes Aléatoires (IRISA) in Rennes, France. The aim of this project is to integrate handwritten interaction capabilities into the Sychromedia platform, an infrastructure that already integrates other modalities (audio, video, text) to support collaborative work in telepresence, so as to improve the usability of the platform by the various users that take part to a collaborative work session (collaborative applications may include virtual classrooms, e-health, or meeting using videoconference). The handwritten interaction is based on digital ink, acquired on whiteboards or Tablet PCs. The digital ink can correspond to annotations on electronic documents shared by the various users, or to gestures associated to commands and used to manipulate the documents (zoom, rotation, or flip, for example) or the annotations (copy, cut, paste, or delete, for example). In the latter case of digital ink corresponding to gestures, the set of gestures considered may differ according to the collaborative task (e-health or virtual classrooms, for example), the nature of the underlying documents (an X-ray image or a course pdf document, for example), or even each user. Here, we focus on offering each user gesture-based personalization capabilities. The personalization process can be invoked independently by each user, anytime during a collaborative session (using the evolving classifier presented in Section 2), and will persist from one session to another. The personalization capabilities considered here are described in the following sub-section.

3.2. Personalization capabilities for gestures

There are 2 options to personalize the gestures of a user that are considered here:

- adaptation to the user style;
- addition of a new gesture, associated to a functionality of the application.

The first personalization option enables the recognition system to adapt to the way the user draws the gestures that this system has already learned. The second personalization option allow users to add a gesture to a functionality that did not have an associated gesture, or even to add another gesture to a functionality which already possesses a gesture. In the first case, a user may want to add a gesture to a functionality he uses a lot

and that did not have an associated gesture (it may also be due to the fact that the functionality may have been added afterwards). In the latter case, a user may add a new gesture that he considered to be better than the previous one (better to remember, to draw, or to recognize, for example). It may also be useful if two users are sharing the same whiteboard and they want to use different gestures for the same functionality.

4. Experiments and results

In this section, we present the handwritten gesture database and the experimental setup used in the experiments. Then, the results of the experiments conducted on the various personalization capabilities, using the presented evolving classifier, are described.

4.1. SIGN: on-line gesture database

The following experiments were performed on the *SIGN* database, which contains on-line handwritten gestures. The data collection sessions were performed at the Sychromedia laboratory, and at the Imadoc team project. It is composed of samples for 17 different gestures, drawn by 20 writers on Tablet PCs. Each writer has performed 4 acquisition sessions: each session starts with drawing each gesture 5 times, and then gestures to draw are presented in a random order to the writer, to simulate the use of an application. So each writer draws each gesture 25 times during each session. Thus, each writer has drawn each gesture 100 times, which leads to having 1,700 gestures in each writer-specific gesture dataset, and totally 34,000 gestures. The database (with additional information on the data) can be downloaded freely [1].

4.2. Gestures and functionalities considered

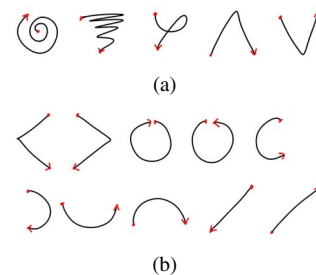


Figure 1. (a) Initial gesture set, and (b) secondary gesture set

In the following experiments, the gesture set is decomposed into 2 sets: an initial set (corresponding to

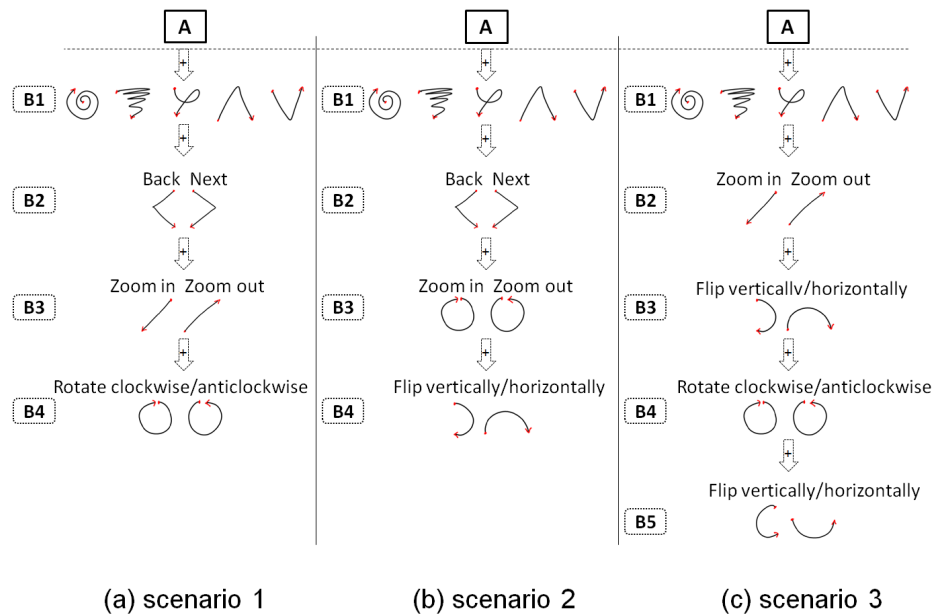


Figure 2. Various user personalization behaviors

gestures initially assigned to the basic functionalities of the application) and a secondary set (corresponding to gestures that could be incrementally associated to the functionalities of the application). The initial gesture set contains 5 gestures, as shown in Figure 1(a). The first gesture (called *snail*) is used to call a contextual menu that will allow the user to change the properties of the digital ink, or to switch between the ink and gesture modes. The other gestures are associated to basic operations on digital ink (drawn after selecting the ink to consider), *i.e.* deleting, cutting, copying, and pasting the selected ink strokes (gestures are depicted in this order, in Figure 1(a)). The secondary gesture set contains 10 gestures, as shown in Figure 1(b). These gestures can be assigned to the previous ink editing functionalities, or to other functionalities on ink strokes or on the underlying document, like rotation, flip, or zoom.

4.3. Experimental setup

In order to emulate a real context, we suppose that the classification system is pre-trained on “batch” mode (phase A), on the initial gesture set, using a writer-independent dataset; these gestures are supposed to be assigned to the main functionalities. Then, after the interface has been delivered to its specific user, the personalization process (phase B) starts using the incremental lifelong learning. This incremental learning includes, on one hand, the adjustment of the learning of the initial gesture set, using the newly available writer-dependent data (*i.e.* adaptation) (phase B1), and, on the

other hand, the learning of new gestures according to the user needs (phases B2 to B5). Figure 2 shows three use cases that correspond to various user personalization behaviors that we consider in the experiments.

Each gesture is described by a set of 21 features. The presented results are the average of results of 20 different tests for the 20 writers. For each experimental run, we use the datasets of 19 writers for the writer-independent learning phase and one writer-specific dataset for the personalization phase. In order to get the results unbiased by the data order effect, we repeat the experiment for each writer 40 times with different random data orders and the mean results are considered. We used about half of the database for the incremental learning process and the rest is used to estimate the evolution of the performance during the learning process. Two fuzzy incremental learning models are compared in these experiments: (a) Model I (section 2.2.1), *i.e.* evolving zero-order TS classifier with binary consequents and recursive mountain clustering learning, and (b) Model II (section 2.2.3) *i.e.* our extended version in which we integrate the fuzzy vector quantization algorithm (section 2.2.2) in the incremental learning process. The parameters λ and α are set to 0.005 and 0.001 respectively.

4.4. Results

Figure 3 shows the experimental results for the three use cases. We first note that the adaptation phase (B1) enhances the system performance for the default ges-

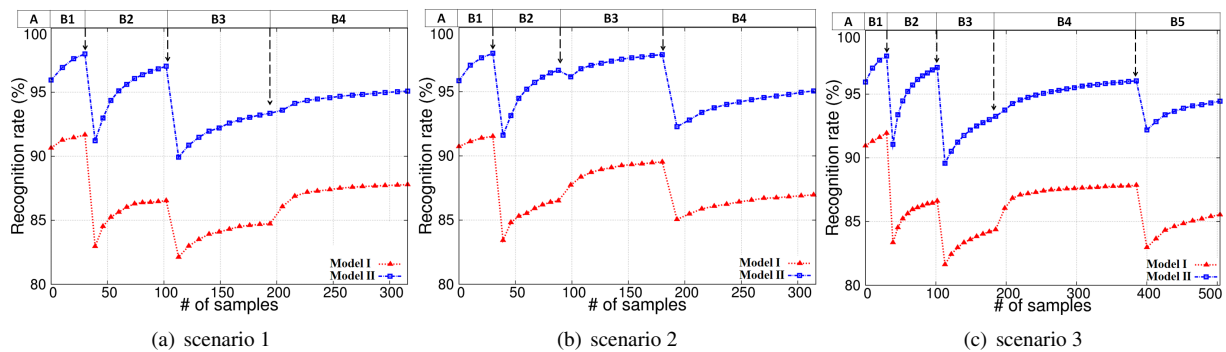


Figure 3. Results for the three different scenarios.

ture set thanks to the writer-specific learning data. Secondly, we note that the average recognition rate during the incremental learning process (B2–B5) is about 95% when using Model II, while it is about 86% using Model I. These results show that integrating the premise adaptation in our model enhances significantly the performance in the incremental learning process, and decreases the error rate by about 65%. We note also the good stability of the model when introducing new unseen classes. Thus, for the three considered scenarios, the recognition rate does not go below 90% even when adding a new gesture.

5. Conclusion and future work

In this paper, we have presented an incremental learning algorithm for fuzzy-based classifiers. The proposed classifier was used as a gesture recognition system that enables user personalization capabilities, when integrating into a pen-based application. The dynamic nature of these classifiers allow them to adapt to each user style as well as to the addition of new unseen classes, without destroying the already learned ones. Experiments considering gesture-based personalization scenarios showed the robustness of this approach.

Future work will focus on human-computer interaction purposes. First, we will investigate the integration of the gesture recognition system into a real pen-based application (as mentioned in section 3), especially on how to provide the correct label of each drawn gesture to the lifelong learning process. Indeed, when there is a recognition error made by the system, the user will have to provide the correct label to the gesture that was drawn. So, we will focus on designing approaches that will enable the user to correct the recognition result so that they would be as user friendly as possible. Helping the user in its choice of new gestures would also be of interest. In that case, we could give to the user informa-

tion on possible confusions between the new gestures he wants to add and the existing ones, so as to optimize the usability of the gesture-based application.

References

- [1] <http://www.synchromedia.ca/web/ets/gesturedataset>.
- [2] P. Angelov and D. Filev. An approach to online identification of takagi-sugeno fuzzy models. *IEEE Tr. Systems, Man, and Cybernetics*, 34(1):484–498, 2004.
- [3] G. A. Carpenter and S. Grossberg. The art of adaptive pattern recognition by a self-organizing neural network. *Computer*, 21(3):77–88, 1988.
- [4] F. Chung and T. Lee. Fuzzy competitive learning. *Neural Networks*, 7(3):539–551, 1994.
- [5] S. De Backer et al. Texture segmentation by frequency-sensitive elliptical competitive learning. *Image and Vision Computing*, 19(9–10):639–648, 2001.
- [6] J.-C. de Barros and A. L. Dexter. On-line identification of computationally undemanding evolving fuzzy models. *Fuzzy Sets and Systems*, 158(18):1997–2012, 2007.
- [7] J.-S. Jang. Anfis: adaptive-network-based fuzzy inference system. *IEEE Tr. on Systems, Man and Cybernetics (Part B)*, 23(3):665–685, 1993.
- [8] J. LaViola et al. A practical approach for writer-dependent symbol recognition using a writer-independent symbol recognizer. *PAMI*, 29(11):1917–1926, 2007.
- [9] E. Lughofer. Flexfis: A robust incremental learning approach for evolving takagi-sugeno fuzzy models. *IEEE Tr. on Fuzzy Systems*, 16(6):1393–1410, 2008.
- [10] R. Polikar et al. Learn++: An incremental learning algorithm for supervised neural networks. *IEEE Tr. on Systems, Man, and Cybernetics*, 31(4):497–508, 2001.
- [11] J. Sadri et al. A new clustering method for improving plasticity and stability in handwritten character recognition systems. In *ICPR*, pages 1130–1133, 2006.
- [12] T. Takagi and M. Sugeno. Fuzzy identification of systems and its applications to modeling and control. *IEEE TSMC*, 15(1):116–132, 1985.
- [13] R. R. Yager and D. P. Filev. Learning of fuzzy rules by mountain clustering. In B. Bosacchi and J. C. Bezdek, editors, *SPIE*, volume 2061, pages 246–254, 1993.