Soft Comput (2008) 12:765–775
DOI 10.1007/s00500-007-0235-2

ORIGINAL PAPER

A hybrid neural network classifier combining ordered fuzzy ARTMAP and the dynamic decay adjustment algorithm

Shing Chiang Tan · M. V. C. Rao · Chee Peng Lim

Published online: 14 September 2007

Abstract This paper presents a novel conflict-resolving neural network classifier that combines the ordering algorithm, fuzzy ARTMAP (FAM), and the dynamic decay adjustment (DDA) algorithm, into a unified framework. The hybrid classifier, known as Ordered FAMDDA, applies the DDA algorithm to overcome the limitations of FAM and ordered FAM in achieving a good generalization/performance. Prior to network learning, the ordering algorithm is first used to identify a fixed order of training patterns. The main aim is to reduce and/or avoid the formation of overlapping prototypes of different classes in FAM during learning. However, the effectiveness of the ordering algorithm in resolving overlapping prototypes of different classes is compromised when dealing with complex datasets. Ordered FAMDDA not only is able to determine a fixed order of training patterns for yielding good generalization, but also is able to reduce/resolve overlapping regions of different classes in the feature space for minimizing misclassification during the network learning phase. To illustrate the effectiveness of Ordered FAMDDA, a total of ten benchmark datasets are experimented. The results are analyzed and compared with those from FAM and Ordered FAM. The outcomes demonstrate that Ordered FAMDDA, in general, outperforms FAM and Ordered FAM in tackling pattern classification problems.

Keywords Adaptive resonance theory · Fuzzy ARTMAP · Ordering algorithm · Dynamic decay adjustment algorithm

S. C. Tan (✉)
Faculty of Information Science and Technology,
Multimedia University, Jalan Ayer Keroh Lama,
Bukit Beruang, 75450 Melaka, Malaysia
e-mail: sctan@mmu.edu.my

M. V. C. Rao
Faculty of Engineering and Technology, Multimedia University,
Jalan Ayer Keroh Lama, Bukit Beruang, 75450 Melaka, Malaysia

C. P. Lim
School of Electrical and Electronic Engineering,
University of Science Malaysia, Engineering Campus,
14300 Nibong Tebal, Penang, Malaysia

1 Introduction

Pattern classification, in general, involves partitioning a feature space into several regions and assigning an incoming pattern into one of the classes defined on these regions. An output class is then determined from the mapping between the feature space and the decision space. There are several approaches to solving pattern classification problems, e.g., statistical learning algorithm (Guo and Li 2003; Justino et al. 2005), *k*-nearest neighbor rule (Wu et al. 2002), Bayesian classifiers (Pernkopfa 2005), fuzzy-genetic systems (Ishibuchi et al. 2005), and neural networks (Zhang 2000). Of these approaches, neural networks, which have the advantages of being parallel in nature and adaptive to dynamic environments, have emerged as a promising tool in solving pattern classification problems.

In neural network research, one trend is geared toward enhancing the functionality of neural-network-based classifiers by introducing other soft-computing techniques into their framework. One of the motivations, perhaps the most important one, of forming hybrid classifiers is to improve the classification performance. Nevertheless, according to Simpson (1992), a good classifier, apart from yielding a good performance, should be able (1) to learn a given task quickly; (2) to overcome catastrophic forgetting; (3) to solve nonlinearly separable problems; (4) to provide the capability for soft and hard decisions given the degree of membership of the data within each class; (5) to provide an explanatory facility on how and why the data are classified as such; (6) to perform generalization that is independent of parameter tuning; (7) to operate without prior knowledge about the distribution of data in each class; and (8) to overcome conflicts resulting from overlaps of input space of different classes.

Fuzzy ARTMAP (FAM) (Carpenter et al. 1992), which is a supervised model from the adaptive resonance theory (ART) neural network family, is one of the conspicuous neural-network classifiers that encompasses most of the features above. The FAM network has the property of incremental learning, which accentuates its capability in overcoming the *stability–plasticity* dilemma. Besides, the network does not suffer from catastrophic forgetting. The advantages of an incremental learning system, as explained by Polikar et al. (2001), are the learning system can absorb additional information from new data; new information can be adapted continually without a need for re-training the network with a dataset that constitutes new and old data samples becomes available; the system can preserve previously learned knowledge; and, new categories can be introduced to include new information. The FAM network can also undertake pattern classification tasks without prior knowledge on the distribution of the dataset. It can be trained in a unique fast learning mode. In addition, information that is kept in terms of hyper-rectangles in the network can be extracted and interpreted as *IF–THEN* rules with ease. Research in FAM and its variants is fruitful. Many variations of FAM have been introduced from its initial model (Carpenter et al. 1992); ART-EMAP (Carpenter and Ross 1995), dARTMAP (Carpenter et al. 1998), boosted ARTMAP (Verzi et al. 1998), fuzzy ARTVar (Dagher et al. 1998), Gaussian ARTMAP (Williamson 1996), PROBART (Marriott and Harrison 1995), PFAM (Lim and Harrison 1997), μ ARTMAP (Gomez-Sanchez et al. 2002), and FAMR (Andonie and Sasu 2006). Some FAM-based networks entail a probability estimation or a statistical inference mechanism for tackling especially statistical pattern classification tasks. However, in our work, the development of the FAM network is not geared toward this direction. The details of the proposed network are described in subsequent sections.

Fuzzy ARTMAP (FAM) is an incrementally learning system that can operate either in off-line or on-line modes. The off-line learning mode is the most common learning strategy used in a lot of neural network models, and is described as a "total absence of the concept of an autonomous learning algorithm" by Roy (2000). The difference between off-line and on-line learning modes of FAM is: in the off-line learning mode, the whole set of training data must be available, and all data samples are presented repeatedly to the network; in the on-line learning mode, the network must learn from a new data sample when it is available, and all existing (old) data samples are not re-used in training. In either operation mode, each data sample is presented to the network sequentially and weights are adapted correspondingly. Indeed, in the off-line learning mode, the generalization/performance of FAM is affected by two important factors: (1) network parameters (especially the choice and vigilance parameters); and (2) presentation order of training data. The "default" settings are: a small positive value for the choice parameter, and zero for the vigilance parameter (baseline vigilance). These parameter settings have been adopted in a lot of FAM simulations in the off-line learning mode (Dagher et al. 1999), as well as in our work. To cope with the second problem, one approach is to feed random orders of training data to the network until a network with an acceptable performance is attained. Nevertheless, it is not an easy task to obtain such a well-trained network using this approach. In addition, the approach requires excessive experimentation in searching for a random order of data presentation that could give a good network generalization. As a result, the computational overhead of this approach is high. In view of this problem, an *ordering algorithm* that is used to determine a fixed order of training pattern presentation to FAM for achieving a good generalization is proposed by Dagher et al. (1999). Using the Max–Min clustering approach, the ordering algorithm is able to identify a fixed order of training data presentation for FAM training in an off-line mode. The FAM network trained with the ordering algorithm is called *Ordered FAM*.

On the other hand, learning in FAM incurs either recruitment of a new hyper-rectangular prototype that accommodates novel pattern or generalization of an existing prototype toward the input pattern. This prototype, which represents an arbitrary class under a supervised-learning scheme, settles itself in the feature space with arbitrary boundary. Learning of an existing prototype in the FAM network invariably corresponds to boundary expansion, which in turn, would possibly lead to formation of overlapping boundary among prototypes of different classes in the feature space. However, original FAM (Carpenter et al. 1992) does not impart explicitly a learning scheme that is in settlement with a conflict resulting from overlapping prototypes of different classes. If overlaps among prototypes of different classes occur, original FAM might less likely to make correct prediction/recognition based on undistinguishable (conflicting) boundaries in the feature space. This condition becomes worse with the existence of a ubiquitous number of overlapping prototypes from different classes that reside in the network. If the boundary of such conflicting prototypes remains untouched, it could cause undesirable effects to the generalization of the network. In this regard, the decision boundary between conflicting prototypes in the feature space should be separated so as to reduce the misclassification rate.

Dagher et al. (1999), apart from identifying a fixed order of training data presentation, claimed that the ordering algorithm can assist FAM in learning prototypes of different classes with no overlaps on one another. This is true when the ordering algorithm is applied to process "clean" (i.e., without overlaps of samples from different classes) input patterns that consequently would lead to the formation of prototypes with clear class distribution in the feature space. In real-world problems, however, a dataset is often complex and noisy, and may contain arbitrary number of samples that have scattered distributions in the feature space. While the ordering algorithm is able to determine a fixed order of input patterns based on an Euclidean measure, it may not be able to sort the input patterns in a way that can prevent the formation of overlapping prototypes of different classes. In other words, on presentation of a fixed order of a "noisy" dataset, which is determined by the ordering algorithm beforehand, it is possible for overlapping prototypes of different classes to be formed in Ordered FAM (refer to the example in "Appendix").

In this paper, a novel conflict-resolving adaptive network, which integrates the ordering algorithm, FAM, and the dynamic decay adjustment (DDA) (Huber and Berthold 1995) algorithm into a united framework, is proposed. The benefit of the ordering algorithm is to identify a fixed order of training data presentation that is independent of any permutations of the input training patterns *prior to network learning*. Compared with the ordering algorithm that may help reduce overlapping among prototypes of different classes (yet it is likely to bring an insignificant effect when dealing with a complex, "noisy" dataset), DDA offers an approach that could handle overlapping among prototypes of different classes in an effective way when *network learning is in progress*. Thus, the fusion of these three techniques, known as Ordered FAMDDA, inherits the benefits of its predecessors, i.e., a fast, stable, and incrementally learning classifier (from FAM); a decrease in computational overhead by subscribing to a single fixed order of input pattern presentation (from the ordering algorithm); and a capability of providing an explicit conflict-resolving facility during network learning (from DDA). Hence, in our work, in addition to the two motivations as in Dagher et al. (1999) (i.e., the design of an adaptive network that is independent of parameter tuning and that requires no excessive experimentation by introducing a single fixed order of pattern presentation), another motivation is to encompass a conflict-resolving facility in the learning process of Ordered FAMDDA that can handle overlapping among prototypes of different classes in the feature space. All these motivations are aimed at achieving a good network generalization. The proposed Ordered FAMDDA network is evaluated using ten benchmark datasets, and the results are analyzed and compared with those from FAM and Ordered FAM.

The organization of this paper is as follows. In Sect. 2, the ordering algorithm, FAM, and the DDA are briefly presented. The algorithm of Ordered FAMDDA is described in detail in Sect. 3. In Sect. 4, a series of empirical studies are conducted. First, a synthetic dataset to demonstrate the generalization capability of Ordered FAMDDA is presented. Then, the Ordered FAMDDA network is evaluated using nine datasets from UCI (Hettich et al. 1998). The results are compared with those from FAM and Ordered FAM reported by Dagher et al. (1999). A summary of the work is presented in Sect. 5.

2 The ordering algorithm, FAM, and DDA algorithm

In this section, the operations of the ordering algorithm, FAM, and DDA, are described. For a detailed exposition of these approaches, readers can refer to the relevant references provided.

2.1 The ordering algorithm

The ordering algorithm (Dagher et al. 1999) is a type of Max–Min clustering algorithm, and is used to find the order of presentation of input data for FAM learning. The algorithm, which comprises three stages, requires a pre-defined parameter setting in terms of the number of distinct classes of a classification task (i.e., n_{clust}). In Stage 1, one starts with an M-dimensional input pattern ($\boldsymbol{a}$) and obtains $2M$-dimensional input pattern ($\boldsymbol{A}$) by complement coding (Carpenter et al. 1992), as follows.

$$\boldsymbol{A} = \left(\boldsymbol{a}, \boldsymbol{a}^c\right) \equiv (a_1, \ldots, a_M, 1 - a_1, \ldots, 1 - a_M) \tag{1}$$

Input pattern $\boldsymbol{a}$ that maximizes the sum

$$\sum_{i=1}^{M} |a_{M+i} - a_i| \tag{2}$$

is selected as the first pattern to be presented. This pattern is also treated as the first cluster center of the training patterns. In Stage 2, the next $(n_{\text{clust}} - 1)$ input patterns are identified for presentation during network training. These patterns represent the next cluster centers of the training patterns. They are determined consecutively using the Max–Min clustering algorithm. In this stage, the Euclidean distance between the remaining input patterns and the existing cluster centers $\boldsymbol{a}^k$ $(k \leq n_{\text{clust}})$ are computed. The minimum Euclidean distance between the input pattern and the cluster center is identified:

$$d_{\min}^{a} = \min_{1 \leq j \leq k} \left\{ \text{dist}(\boldsymbol{a}, \boldsymbol{a}^j) \right\} \tag{3}$$

The input pattern, which has the maximum value of these distances, is selected as the next cluster center. In Stage 3, the presentation order of the remaining input patterns are determined by finding the minimum Euclidean distances between these patterns and the n_{clust} cluster centers. The whole procedure of Stage 3 is repeated until the order of all input patterns for the network training phase have been identified.

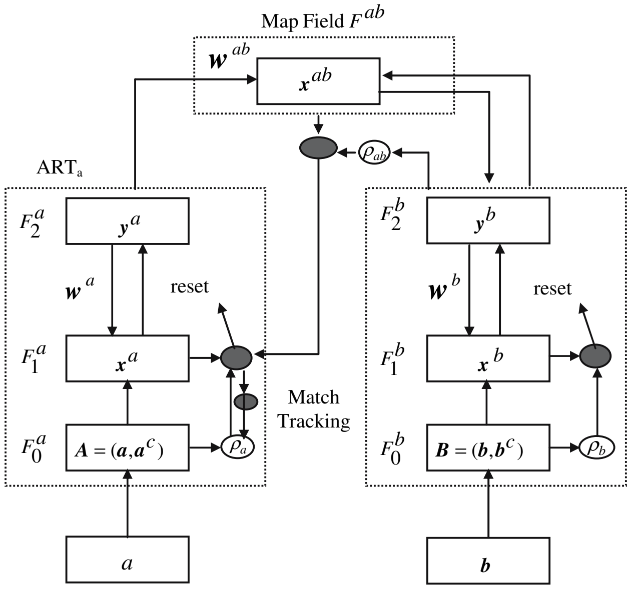

Fig. 1 The network structure of FAM

2.2 Fuzzy ARTMAP

Fuzzy ARTMAP (FAM) (Carpenter et al. 1992) is an incremental learning neural network that is capable of self-organizing and self-stabilizing information and network configuration on presentation of input patterns. Figure 1 shows the FAM network structure. The network is composed of two ART modules (i.e., ART$_a$ and ART$_b$) that are interconnected through a mapping field F^{ab}. Each ART module comprises three layers of nodes; F_0^a (F_0^b) is the normalization layer in which an M-dimensional input vector, $\boldsymbol{a}$, is complement-coded (Carpenter et al. 1992) to a $2M$-dimensional vector $\boldsymbol{A}$; F_1^a (F_1^b) is the input layer which receives the complement-coded input vectors; F_2^a (F_2^b) is the recognition layer which is a dynamic layer that encodes prototypes of input patterns and allows the creation of new nodes when necessary.

During supervised learning, an input pattern is presented to ART$_a$ with its associated target output to ART$_b$. At ART$_a$, input pattern $\boldsymbol{A}$ is propagated from F_0^a to F_2^a through F_1^a. Each node j in F_2^a is activated according to a choice function (Carpenter et al. 1992)

$$T_j = \frac{\left|\boldsymbol{A} \wedge \boldsymbol{w}_j^a\right|}{\alpha_a + \left|\boldsymbol{w}_j^a\right|} \tag{4}$$

where α_a is the choice parameter and is set close to zero; $\boldsymbol{w}_j^a$ the weight of node j. Under a *winner-take-all* competition scheme, the node with the largest activation, denoted as node J, is selected as the winning node. The key feature of FAM is the vigilance test which measures the similarity between the winning prototype patterns, $\boldsymbol{w}_J^a$, and $\boldsymbol{A}$ against a threshold (vigilance parameter, ρ_a) (Carpenter et al. 1992), i.e.

$$\frac{\left|\boldsymbol{A} \wedge \boldsymbol{w}_J^a\right|}{|\boldsymbol{A}|} \geq \rho_a \tag{5}$$

If the winning node fails the vigilance test, then a new search cycle for another winning node is carried out. The process of searching for a winning node is continued until the selected node is able to pass the vigilance test. If no such node exists, a new node is created in F_2^a to code the input pattern. The same pattern-matching cycle occurs simultaneously in ART$_b$ using the target vector to find a winning node that codes the target class.

After the winning nodes in F_2^a and F_2^b have been identified, a prediction is sent from F_2^a to F_2^b via F^{ab}. A map-field vigilance test is used to confirm the prediction.

$$\frac{\left|\boldsymbol{y}^b \wedge \boldsymbol{w}_J^{ab}\right|}{\left|\boldsymbol{y}^b\right|} \geq \rho_{ab} \tag{6}$$

where $\boldsymbol{y}^b$ denote output vector of $\boldsymbol{y}^b$; $\boldsymbol{w}_J^{ab}$ denote the weight vector from F_2^a to F^{ab}; and ρ_{ab} is the map-field vigilance parameter. If the test fails, it implies that the winning node of F_2^a has made a wrong prediction of the target class F_2^b. Under this circumstance, a matching–tracking process (Carpenter et al. 1992) is initiated. The parameter ρ_a which is initially set to a user-defined baseline vigilance parameter $\bar{\rho}_a$, now is raised to

$$\rho_a = \frac{\left|\boldsymbol{A} \wedge \boldsymbol{w}_J^a\right|}{|\boldsymbol{A}|} + \delta \tag{7}$$

where δ is a small positive value. Upon the execution of the matching–tracking process, the current winning node will fail in ART$_a$ vigilance test. A new search cycle in ART$_a$ is initiated with a new level of ρ_a. This process is continued until a correct prediction is made between winning nodes in F_2^a and F_2^b. Then, the system enters a learning phase, where the weight vector of winning node in F_2^a are updated as (Carpenter et al. 1992)

$$\boldsymbol{w}_J^{a(\text{new})} = \beta_a\left(\boldsymbol{A} \wedge \boldsymbol{w}_J^{a(\text{old})}\right) + (1-\beta_a)\boldsymbol{w}_J^{a(\text{old})} \tag{8}$$

where β_a is the learning rate of the ART$_a$ module. The ART$_b$ module also undergoes the same learning process as in ART$_a$. Note that all equations in ART$_a$ are applicable to ART$_b$, but with superscript or subscript a replaced by b.

2.3 The DDA algorithm

The DDA algorithm, which resorts to the constructive nature of the RCE algorithm (Reilly et al. 1982) in providing a growing structure for the radial basis function (RBF) (Moody and Darken 1989) network, is endowed with a capability of adjusting the width of the radial basis prototypes locally. Huber and Berthold (1995) extended the DDA algorithm to

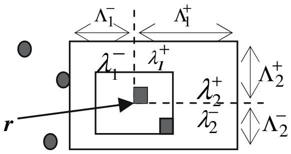

Fig. 2 A 2D prototype that comprises two types of rectangles ($\lambda_*^{+/-}$-regions of inner rectangles; $\Lambda_*^{+/-}$-regions of outer rectangles) and a reference vector r. The prototype includes samples of the same class (indicated by *dark squares*) and excludes samples of different class (indicated by *dark circles*)

construct conflict-free, rectangular basis prototypes. Each dimension of the prototype comprises an inner rectangle and an outer rectangle. A two-dimensional prototype is presented in Fig. 2. Width adjustment of the prototype is class dependent, which distinguishes the prototype from different neighbors. In this work, we use the DDA algorithm (Huber and Berthold 1995) that comprises the following three steps: covered, commit, and shrink. In general, the idea of DDA is similar to FAM in the aspect of recruiting new nodes for accommodating new patterns (i.e., *commit*) and updating existing prototypes with the latest information (i.e., *covered*). When a new pattern is incorrectly classified by an existing prototype of conflicting classes, the width of the outer rectangle of the prototype is reduced through the *shrink* step so as to overcome conflicts. It is noted that the width of all outer rectangles of the prototype is initially infinite. Shrinking of an existing finite dimension is preferred for not losing "infinite volume" of other infinite dimensions. Nevertheless, to avoid the formation of a very thin rectangle, a user-defined minimum width threshold, $\varepsilon_{\min}$ (Huber and Berthold 1995), is enforced on each finite dimension.

3 The Ordered FAMDDA network

The Ordered FAMDDA network is an integration of the ordering algorithm, FAM, and DDA. The network architecture of Ordered FAMDDA is similar to that of Ordered FAM. Given a set of training patterns, similar to Ordered FAM, Ordered FAMDDA operates in an off-line mode. Hence, in Ordered FAMDDA, all training patterns are first processed by the ordering algorithm prior to network learning. The FAM network is then encapsulated with a conflict-resolving facility provided by the DDA algorithm. Note that, although the ordering algorithm is applied, the presentation of input patterns to the network may inadvertently lead to the formation of conflicting prototypes during the network learning phase. To undertake this problem, the FAM learning procedure, specifically in its ART$_a$ module, is modified for resolving overlaps (conflicts) among prototypes of different classes.

To impart the idea of infinite/finite volume as in Huber and Berthold (1995), each dimension d of the prototype p in the recognition layer (i.e., F_2^a (Carpenter et al. 1992) of the ART$_a$ module is posited to be either in status S_{pd}=0 (i.e., infinite dimension) or S_{pd}=1 (i.e., finite dimension). All dimensions of a newly committed F_2^a prototype are initialized as 0. When the prototype is involved in width shrinking, the status of the selected dimension is updated to 1. In addition, a new set of *reference vector*, $\boldsymbol{w}_j^r$, is introduced to each prototype in F_2^a. Each reference vector is initialized as a zero vector. When learning takes place, besides the weight vector $\boldsymbol{w}_J^a$ (Carpenter et al. 1992), the reference vector of the Jth F_2^a winning node is updated according to a recursive center estimation procedure (Lim and Harrison 1997), as follows.

$$\left(\boldsymbol{w}_J^r\right)^{\text{new}} = \left(\boldsymbol{w}_J^r\right)^{\text{old}} + \frac{1}{N_J}\left(\boldsymbol{A} - \left(\boldsymbol{w}_J^r\right)^{\text{old}}\right) \tag{9}$$

where N_j is the number of input patterns of the Jth node, $N_J = N_J + 1$. Associations between the ART$_a$ and ART$_b$ modules are linked through a mapping field (Carpenter et al. 1992).

If $\boldsymbol{x}$, which represents the weights of the M-dimension (or $2M$-dimension with complement coding) winning prototype falls in the region formed by the prototype of different classes (i.e., λ), a conflict is said to occur, and a width shrinking procedure is executed to avoid the conflict. In this regard, the width of the conflicting prototypical region is shrunk. The shrinking procedure is applied successively between the wining prototype and other conflicting prototypes. Three cases of width shrinking as in Huber and Berthold (1995) are considered. First, if the existing finite dimensions of the conflicting prototype q can be shrunk without falling below a pre-set $\varepsilon_{\min}$, the one with the smallest loss in volume ($\gamma_{\text{best},k}$) is chosen, as follows.

$$\gamma_{\text{best},k} = \min\left\{\left|w_{\boldsymbol{qk}}^{\boldsymbol{r}} - x_k\right| : \forall 1 \le i \le n, i \ne k \right. \\ : \left(\frac{\lambda_{qk} - \left|w_{\boldsymbol{qk}}^{\boldsymbol{r}} - x_k\right|}{\lambda_{qk}} \le \frac{\lambda_{qi} - \left|w_{\boldsymbol{qi}}^{\boldsymbol{r}} - x_i\right|}{\lambda_{qi}}\right) \\ \left. \wedge\left(\lambda_{qk} \ge \varepsilon_{k,\min}\right) : S_{qi}, S_{qk} = 1 \right\} \tag{10}$$

where $\lambda_{qj} = |w_{qj} - w_{qj}^r|$ and w_{qj} represents the weight of the prototype q at dimension j. Equation (10) indicates that shrinking of an existing finite dimension is preferred for not losing "infinite volume" of other infinite dimensions. Such idea also complies with the learning paradigm of FAM, which supports the stability property; i.e., it helps protect previously learned knowledge of Ordered FAMDDA from being washed away. However, if the first option is not satisfied, the following options are considered, i.e., either one of the remaining infinite dimensions is shrunk ($\gamma_{\max,l}$),

$$\gamma_{\max,l} = \max\left\{\left|w_{ql}^r - x_l\right| : S_{ql} = 0\right\} \tag{11}$$

or, the width of the finite dimension of the prototype q is shrunk

$$\gamma_{\min,m} = \min \left\{ \left| w^{\boldsymbol{r}}_{\boldsymbol{qm}} - x_{qm} \right| : \forall 1 \leq i \leq n, i \neq m : \left(\frac{\lambda_{qm} - \left| w^{\boldsymbol{r}}_{\boldsymbol{qm}} - x_{qm} \right|}{\lambda_{qm}} \leq \frac{\lambda_{qi} - \left| w^{\boldsymbol{r}}_{\boldsymbol{qi}} - x_{qi} \right|}{\lambda_{qi}} \right) : S_{qi}, S_{qk} = 1 \right\} \tag{12}$$

Eq. (11) is selected if $\gamma_{\max,l} > \gamma_{\min,m}$. Upon the selection of γ_z from Eqs. (10)–(12), region λ_{qk} is reduced by adjusting w_{qz}, as follows.

$$w^{\text{new}}_{qz} = \begin{cases} w^{\text{old}}_{qz} + \gamma_z & \text{if } w^{\text{old}}_{qz} < w^r_{qz} \\ w^{\text{old}}_{qz} - \gamma_z & \text{if } w^{\text{old}}_{qz} > w^r_{qz} \end{cases} \tag{13}$$

If $w_{qz} > 1$ or $w_{qz} < 0$, it is rounded as $w_{qz} = 1$ or $w_{qz} = 0$, respectively. It should be noted that Ordered FAMDDA undergoes sequential learning which synchronizes the operation of the DDA algorithm with the adaptive learning process of FAM in a fast manner. This indicates that the DDA algorithm is not possible to find an optimum solution which can eliminate all conflicts in reasonable time in this sequential-learning paradigm. Instead, the proposed Ordered FAMDDA network attempts to reduce and/or avoid overlapping among prototypes of conflicting classes during its training session. The conflict-resolved prototypes and their associations obtained in the training phase are used, during the test phase, to recall a prediction when an unseen pattern is presented to ART$_a$.

An example is presented to exemplify the operations of Ordered FAM and Ordered FAMDDA in the "Appendix". In general, the training procedure of Ordered FAMDDA can be summarized as follows.

1. All training patterns are processed by the ordering algorithm according to Eqs. (1)–(3) to determine the sequence of data presentation to Ordered FAMDDA.
2. An M-dimensional ordered input pattern $\boldsymbol{a} \in [0, 1]^M$ in F^a_0 is complement-coded to a $2M$-dimensional vector $\boldsymbol{A}$.
3. The input vector $\boldsymbol{A}$ is propagated from F^a_1 to F^a_2 through the weight vector, $\boldsymbol{w}^a$. The response of each node is computed according to the match function as in Eq. (4).
4. The prototype of node J is propagated back from F^a_2 to F^a_1 to participate in the vigilance test (Eq. (5)).
5. If the vigilance test fails, a new search cycle is initiated by re-visiting step 3. (The above cycle runs simultaneously in ART$_b$).
6. A prediction is sent from F^a_2 (i.e., $\boldsymbol{w}^{ab}_J$) and from F^b_2 (i.e., y^b), respectively, to F^{ab}, where the map-field vigilance test (Eq. (6)) is performed.
7. If the map-field vigilance test fails, a matching–tracking procedure is triggered (Eq. (7)). Similar to Ordered FAM or FAM, the matching–tracking procedure only affects the ART$_a$ module of Ordered FAMDDA, and step 3 is re-visited.
8. The weight vectors $\boldsymbol{w}^a_J$ and $\boldsymbol{w}^r$ are updated according to Eqs. (8)–(9). The weight vector $\boldsymbol{w}^b_J$ of the winning node in ART$_b$ is updated using Eq. (8).
9. The F^a_2 nodes with different class from that of the winning node J are identified. Width shrinking of the existing conflicting nodes is initiated by considering one of the three cases as in Eqs. (10)–(12).
10. The training session is completed when all ordered training patterns have been presented to the network; otherwise, training is continued by presenting the next training pattern to the network, and step 2 is re-visited.

4 Experiments and results

In this section, the classification performance of Ordered FAMDDA is evaluated using ten benchmark datasets, which include the synthetic Ripley dataset (Ripley 1994) and nine datasets available from the UCI machine-learning repository (Hettich et al. 1998). The main objective of the study is to evaluate the generalization capability of Ordered FAMDDA, and compare the performance with those from Ordered FAM and FAM. For each experiment, the dataset was divided into a training set and a test set. Ordered FAMDDA was trained with the "default" parameter settings: n_{clust} was set to one more than the number of classes in the dataset; fast learning, $\beta = 1$; ART$_a$ baseline vigilance, $\bar{\rho}_a = 0.0$; choice parameter, $\alpha \approx 0$; in addition to the minimum width, $\varepsilon_{\min} = 0.1$. The intent of these parameter settings is in line with the design of Ordered FAMDDA, i.e., avoids excessive parameter tuning as well as allows ease of comparison with the performance of other classifiers in the literature. The aspects of performance measure were inspected, i.e., (1) the accuracy percentages of the test set; and (2) the number of nodes created.

Note that the accuracy of FAM might be improved by using a multiple-epoch learning mode, with increased number of nodes (Vigdor and Lerner 2006). However, to avoid excessive experimentation, in this work, Ordered FAMDDA was trained using the one-pass learning approach through the data samples. For each classification task, ten independent runs were conducted with ten orderings of training and test set samples. To ascertain the stability of the performance of Ordered FAMDDA, the bootstrapping method (Efron 1979), a statistical quantization method which does not rely on the assumption that the samples must be drawn from a normal

Table 1 Average accuracy rates and number of nodes for the Ripley problem (*acc.* accuracy)

Classifier	#Nodes	Acc. (%)	Bootstrapped 95% confidence interval of acc.
FAM	14.1	82.42	[81.120, 83.580]
Ordered FAM	14.1	83.26	[82.200, 84.150]
Ordered FAMDDA	16.2	86.11	[85.140, 87.150]

distribution, was deployed to determine the average test accuracy rates and the number of nodes created, as well as the confidence intervals of the average test accuracy rates at 95%.

4.1 The Ripley dataset

The Ripley synthetic dataset (Ripley 1994) consists of data samples characterized by two features in two classes. Each class has a bimodal distribution generated from a mixture of two Gaussian distributions with identical covariance matrices (Ripley 1994). The data samples are available from http://markov.stats.ox.ac.uk/pub/PRNN. The training and test sets, respectively, consist of 250 and 1,000 samples, with equal distribution of samples belonging to each class. The average results are presented in Table 1.

Ordered FAMDDA achieved the best performance, with an average accuracy of 86.11%, which is statistically more significantly from those from Ordered FAM and FAM. This observation is supported by the 95% confidence intervals of the average result estimated from the bootstrap method, i.e., the lower bound of the 95% confidence interval estimate is higher that the upper bound estimates from both Ordered FAM and FAM.

The Ripley problem is essentially a statistical classification task. The optimal Bayes error rate is 8% (or 92% accuracy). Based on the results in Table 1, the three FAM-based networks do not perform well in approaching the Bayes error rate. This is because FAM, in general, does not possess a Bayes strategy or a statistical inference mechanism, and is not suitable for tackling statistical pattern classification problems (Marriott and Harrison 1995; Lim and Harrison 1997). Since Ordered FAM and Ordered FAMDDA inherently apply the learning methodology of FAM, they suffer the same shortcoming in tackling statistical pattern classification tasks.

Figure 3 shows a comparison on the incorrect predictions (symbols in bold) made by Ordered FAMDDA, Ordered FAM, and FAM. In general, the three networks fail to establish a clear delineating boundary to separate the overlapping test samples from two Gaussian sources. However, as can be

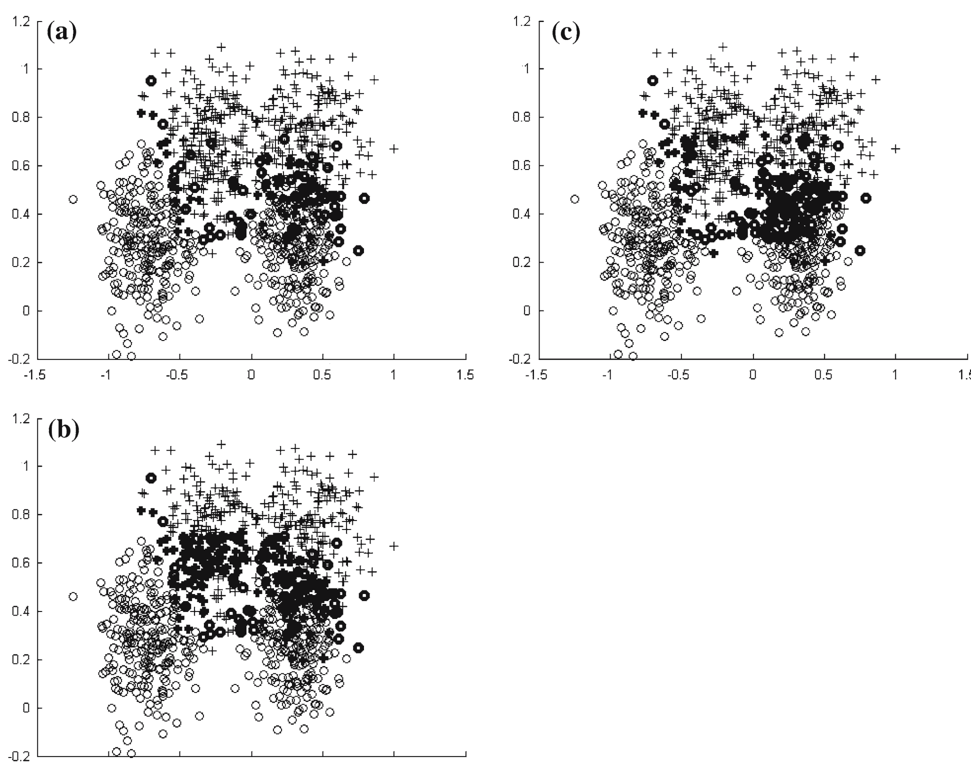

Fig. 3 Classification by **a** Ordered FAMDDA **b** Ordered FAM **c** FAM for the Ripley test data. Symbols, *plus* and *open circle* indicate the test samples of two classes; symbols in *bold* indicate incorrect predictions made by the networks

seen, Ordered FAMDDA, to some extend, made fewer incorrect predictions especially in densely overlapping regions. This might be attributed to the use of DDA that avoids overlapping in the hyper-rectangular prototype structures created in the network, hence improved performance.

The number of nodes of Ordered FAMDDA is greater than those of Ordered FAM and FAM. In FAM and its variants (Ordered FAM, and Ordered FAMDDA), a new node is created if none of the existing nodes can succeed to classify a new input sample. As explained in Sect. 3, Ordered FAMDDA, however, incurs an additional process, i.e., the hyper-rectangular boundary of the network prototypes is revised when necessarily so as to reduce/resolve conflict from overlapping prototypes of different classes in the feature space. Such weight revision affects the prototype structures, hence, the number of nodes created in Ordered FAMDDA. From the results in Table 1, Ordered FAMDDA, on average, created two more nodes than Ordered FAM and FAM. These additional nodes help improve the performance of Ordered FAMDDA in correctly predicting test samples that fall in densely overlapping regions.

Table 2 Datasets used in the study and the number of data samples allocated for the training and test sets

Datasets	Total samples	#Class	#Training	#Test
Diabetes	768	2	513	255
Breast	699	2	467	232
Bupa	345	2	231	114
Iris	150	3	102	48
Balance	625	3	417	208
Glass	214	6	145	69
Cars	1,728	4	1,152	576
Sonar	208	2	139	69
Wine	178	3	120	58

4.2 The UCI benchmark datasets

In this section, Ordered FAMDDA is applied to nine datasets from the UCI machine-learning repository (Hettich et al. 1998). These benchmark datasets are used by Dagher et al. (1999) to access the performance of Ordered FAM. They are *Diabetes*, *Bupa*, *Balance*, *Breast*, *Iris*, *Wine*, *Cars*, *Sonar* and *Glass*. To have a fair performance comparison, the experiments conducted in this work followed closely the procedure and parameter settings as in Dagher et al. (1999). All the data samples were randomly divided into a training set and a test data set with a ratio of 2:1. The percentages of data samples of each class in the training set followed the percentages of each class in the entire dataset. Information on the total number of data samples, the number of output classes, and the number of data samples being allocated for training and test sets are summarized in Table 2.

Note that, in Dagher et al. (1999), 846 samples of the *Cars* dataset were used in the experiments and these samples were actually a subset of the original dataset of 1,728. To avoid bias in selecting a cohort of samples, the whole *Cars* dataset was used in our experiment, and the training and test sets comprised 1,152 and 576 samples, respectively.

In Dagher et al. (1999), FAM and Ordered FAM adopted the multi-epoch training process, i.e., all samples with the same order of presentation were used to train the network repeatedly. Nevertheless, in our work, Ordered FAMDDA was trained in single-epoch, i.e., a single pass of training through an ordered set of training samples. The underlying benefit is that DDA is able to avoid overlapping in hyper-rectangular prototype structures established, hence suffice with single epoch, as evidenced from the results achieved.

In Table 3, a comparison among the results of Ordered FAMDDA, FAM, and Ordered is presented. Note that all the results of Ordered FAM and FAM are extracted from Dagher et al. (1999). Some observations can be made, as follows.

Table 3 Comparison of classification results of FAM, Ordered FAM, and Ordered FAMDDA

	FAM					Ordered FAM	Ordered FAMDDA	
Dataset	Worst acc.	Best acc.	Average acc.	Std. dev.	n_{clust}	Best acc.	Average acc.	Bootstrapped 95% confidence interval of acc.
Diabetes	61.57	70.98	66.63	2.57	3	69.90	71.91	[71.06, 72.90]
Breast	93.10	96.12	94.35	0.95	3	94.39	97.23	[96.64, 97.93]
Bupa	47.37	63.16	56.84	4.22	3	57.01	67.72	[66.49, 69.13]
Iris	89.58	95.83	95.00	1.91	4	97.92	98.74	[97.92, 99.58]
Balance	71.63	78.85	75.91	2.42	4	75.48	80.59	[79.52, 82.02]
Glass	57.97	76.81	63.77	6.18	7	69.56	74.19	[72.17, 76.52]
Cars	86.81	92.36	90.19	2.21	5	90.63	92.48	[92.01, 93.11]
Sonar	63.77	78.26	70.58	4.15	3	79.96	79.37	[78.55, 80.44]
Wine	91.38	98.28	95.69	2.70	4	98.27	97.94	[97.59, 98.28]

Note that the results of FAM and Ordered FAM are those published in Dagher et al. (1999) (*Acc.* accuracy)

First, there is a significant improvement in the generalization of Ordered FAMDDA as compared with those of Ordered FAM in the six (*Bupa*, *Balance*, *Glass*, *Breast*, *Diabetes*, and *Cars*) out of nine datasets. In these tasks, the average accuracy rates of Ordered FAMDDA are higher than the best accuracy rates of Ordered FAM by 10.71, 5.11, 4.63, 2.84, 2.01, and 1.85%, respectively. More importantly, the best accuracy rates of Ordered FAM are outside the 95% confidence interval estimates of the average accuracy of Ordered FAMDDA. In other words, Ordered FAMDDA performs statistically better than Ordered FAM.

Second, when comparing the average results, Ordered FAMDDA outperforms FAM in all nine problems. In addition, in seven (*Bupa*, *Glass*, *Diabetes*, *Balance*, *Iris*, *Breast*, and *Cars*) out of nine datasets, the average accuracy rates of Ordered FAMDDA are better that those of FAM by 10.88, 10.42, 5.28, 4.68, 3.74, 2.88, and 2.29%, respectively.

Third, from the statistical point of view, no significant differences between the performances of Ordered FAMDDA and that of Ordered FAM are observed in *Sonar* and *Wine* problems. This is because, in both classification tasks, the best accuracy rates of Ordered FAM are within the 95% confidence intervals of the average accuracy of Ordered FAMDDA. Indeed, during the training phase of Ordered FAMDDA, no width shrinking among prototypes of different classes was observed. This implied that there was no overlapping in the prototype structures of different classes. Thus, the performance of Ordered FAMDDA was similar to that of Ordered FAM.

In Table 4, a comparison on the network size in terms of the number of nodes of FAM, Ordered FAM and, Ordered FAMDDA is presented. Again, the results of FAM and Ordered FAM are those reported in Dagher et al. (1999). It can be observed that Ordered FAMDDA established fewer number of nodes that those of Ordered FAM and FAM in three out of nine problems. The network sizes of Ordered FAMDDA are actually smaller than those of FAM and Ordered FAM by approximately 50% in *Diabetes* and *Bupa*. In *Balance*, Ordered FAMDDA created 79 nodes, i.e., the same number as that of FAM, but is far fewer than that of Ordered FAM (120 nodes). The only dataset that Ordered FAMDDA established a more complex network size that those of Ordered FAM and FAM is *Iris*, but it is only one node more than that of FAM, and two more than that of Ordered FAM. For the rest of the problems (*Glass, Cars, Sonar*, and *Wine*), the number of nodes in Ordered FAMDDA is, the most, two nodes more than those of Ordered FAM and FAM.

Table 4 Comparison of the network size (number of nodes) of FAM, Ordered FAM, and Ordered FAMDDA

Datasets	FAM		Ordered FAM	Ordered FAMDDA
	Average size	n_{clust}	Net size	Average size
Diabetes	43	3	44	19
Breast	8	3	9	7
Bupa	31	3	31	15
Iris	5	4	4	6
Balance	79	4	120	79
Glass	27	7	30	29
Cars	53	5	57	54
Sonar	6	3	5	6
Wine	4	4	6	5

The results of FAM and Ordered FAM are those published in Dagher et al. (1999)

From the results in Tables 3 and 4, one can see that Ordered FAMDDA outperforms FAM and Ordered FAM in the *Diabetes*, *Breast*, and *Bupa* datasets with a smaller network size. The results of Ordered FAMDDA in the *Iris* and *Cars* datasets are higher than those of FAM and Ordered FAM, and the numbers of nodes are also similar. In the *Balance* dataset, Ordered FAMDDA performs better than FAM and Ordered FAM, with the same number of nodes as compared with the former and with fewer number of nodes as compared with the latter. In the *Glass* dataset, Ordered FAMDDA performs better than Ordered FAM and FAM, and its network size is in between those of FAM and Ordered FAM. In summary, it is reasonable to conclude that Ordered FAMDDA, in general, performs better that Ordered FAM and FAM in terms of test accuracy and network size in the experiments with nine UCI datasets as reported in Dagher et al. (1999).

5 Summary

In this paper, a novel adaptive conflict-resolving network, which is based on the integration of FAM, the ordering and DDA algorithms, has been described. Similar to Ordered FAM, Ordered FAMDDA operates in the off-line mode. Ordered FAMDDA is able to identify a fixed order of training data presentation that is independent of any permutations of the training samples. In addition, the network is encapsulated with a conflict-resolving facility which helps reduce and/or avoid overlapping among prototypes of different classes during the learning phase. The effectiveness of Ordered FAMDDA has been demonstrated empirically using ten benchmark datasets, which include a synthetic dataset, and nine datasets from the UCI machine-learning repository. The results in terms of test accuracy and network size among Ordered FAMDDA, Ordered FAM, and FAM have been analyzed and compared. The bootstrap method has been used to quantity the test accuracy rates of Ordered FAM statistically. In general, Ordered FAMDDA has exhibited encouraging and promising performance in terms of network generalization and network size in the benchmark classification problems tested.

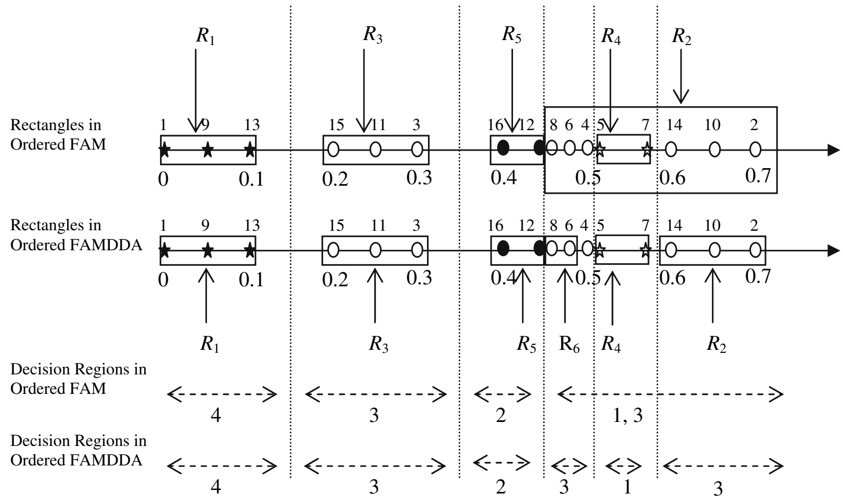

Fig. 4 Rectangles and decision regions of Ordered FAM and Ordered FAMDDA

As future work, we intend to exploit the incremental learning property of FAM, and to embed it into Ordered FAMDDA. In this regard, the performance of Ordered FAMDDA can be accessed by deploying a strategy that combines both off-line and on-line learning modes. The learning process of the network is first conducted in an off-line mode. On-line learning of the trained network is then initiated when the network receives new incoming data samples that are available at arbitrary time. With DDA, the network should correspondingly resolve/reduce conflicts, if any, among overlapping prototypes of different classes on-line. The deployment of this *dual-mode* learning is a direction of further work.

Acknowledgments The authors would like to thank the reviewers, for their valuable remarks and constructive comments, which contributed to the improvement of this paper.

Appendix

A. A numerical example of Ordered FAMDDA in width shrinking

Suppose a total of 16 one-dimensional training data that belong to four classes are given, i.e., two data, 0.51, 0.58 come from Class 1; two data 0.40, 0.45 come from Class 2; nine data 0.20, 0.25, 0.30, 0.46, 0.48, 0.50, 0.60, 0.65, 0.70 come from Class 3; and three data 0, 0.05, 0.10 come from Class 4. By setting $n_{\text{clust}} = 4$, the ordering algorithm computes the order of training pattern for Ordered FAM (and Ordered FAMDDA) as follows: 0, 0.70, 0.30, 0.50, 0.51, 0.48, 0.46, 0.05, 0.65, 0.25, 0.45, 0.58, 0.10, 0.60, 0.20, 0.40.

In Fig. 4, the numbers above the circle and star indicate the order of training pattern presentation. Upon completion of the training phase of Ordered FAMDDA, six rectangles (R_1[0, 0.10]; R_2[0.59,0.70]; R_3[0.20,0.30]; R_4[0.51, 0.58]; R_5[0.46, 0.48]; and R_6[0.40, 0.45]) are formed. Note that the boundary of each rectangle is made clear. The boundary of the existing rectangle (R_2) that previously includes the data of number 4 is adjusted to avoid conflict. As for Ordered FAM, five rectangles (R_1[0, 0.1]; R_2[0.46, 0.70]; R_3[0.20, 0.30]; R_4[0.51, 0.58]; and R_5[0.40, 0.45]) are formed. However, two rectangles (i.e., R_2 and R_4) of Ordered FAM are committed with a conflict; R_4 of Class 1 resides in the region of R_2 of Class 3. The implication from this example is that Ordered FAM may not be able to resolve conflicts among prototypes of different classes when the distribution of training pattern is scatter in the feature space. In particular, the ordering algorithm which is essentially an unsupervised clustering algorithm, is inefficient to sort training patterns in a way that can prevent the formation of overlapping prototypes of different classes when the (supervised) learning phase of Ordered FAM is in operation. Such limitation can be overcome by Ordered FAMDDA—which has a capability of resolving overlap among prototypes of different classes in situ during its learning phase while receiving a fixed order of training patterns that is secured from excessive experimentation—for achieving good generalization performance.

B. Operation of Ordered FAMDDA on the presentation of the first five training patterns as in Fig. 4

The parameters of both ART modules are set to their "default" values: $\alpha \approx 0$ (conservative mode), $\beta = 1$ (fast learning), and $\bar{\rho} = 0.0$ (force choice), and minimum width $\varepsilon_{\text{min}} = 0.1$. By presenting the first four data samples 0, 0.70, 0.30, 0.50, 0.51, 0.48, 0.46, 0.05, 0.65, 0.25, 0.45, 0.58, 0.10, 0.60, 0.20, 0.40 of class 4, 3, 3, 3, 1, 3, 3, 4, 3, 3, 2, 1, 4, 3, 3, 2, respectively, to the network, three rectangles (prototypes) are formed. These rectangles, with complement coding, are $\boldsymbol{w}_1 = [0, 1]$ of Class 4, $\boldsymbol{w}_2 = [0.50, 0.30]$ of Class 3, $\boldsymbol{w}_3 = [0.30, 0.70]$ of Class 3. The reference vectors of these rectangles, according to Eq. (9), are $\boldsymbol{w}_1^r = [0, 1]$, $\boldsymbol{w}_2^r = [0.60,$

0.40], and $\boldsymbol{w}_3^r = [0.30, 0.70]$, respectively. On presentation of the fifth complement-coded data samples (i.e., $\boldsymbol{A} = [0.51, 0.49]$ of class 1), an additional rectangle $\boldsymbol{w}_4 = [0.51, 0.49]$ is formed. Note that since the weight of the first component of $\boldsymbol{w}_4 = [0.51, 0.49]$ is between the weights of the first component of $\boldsymbol{w}_2 = [0.50, 0.30]$ and $\boldsymbol{w}_2^r = [0.60, 0.40]$; a conflict is said to occur. The width of one of the dimensions of the conflicting rectangle $\boldsymbol{w}_2$ is shrunk so as to reduce the overlapping region. Since the status S_{21} of $\boldsymbol{w}_2$ is initially 0, therefore, the width shrinking option as in Eq. (11) is chosen. In this case, $\boldsymbol{w}_{21} = x_1 = 0.51$, $\gamma_{\max,1} = \max\left\{\left|w_{21}^r - x_1\right|\right\} = \max\{0.60 - 0.51\} = 0.09$. The first component of $\boldsymbol{w}_2$ is adjusted to 0.59 (0.50+0.09). This adjustment is oriented to the weight of the rectangle according to the idea of establishing a dynamic weight that can reduce the impact of conflict among different rectangles for achieving a good generalization. The weights of $\boldsymbol{w}_2$ are [0.59, 0.30] (with complement coding) or [0.59, 0.70] (without complement coding).

References

Andonie R, Sasu L (2006) Fuzzy ARTMAP with input relevances. IEEE Trans Neural Netw 17:929–941

Carpenter GA, Ross W (1995) ART-EMAP: a neural network architecture for learning and prediction by evidence accumulation. IEEE Trans Neural Netw 6:805–818

Carpenter GA, Grossberg S, Markuzon N, Reynolds J, Rosen D (1992) Fuzzy ARTMAP: a neural network architecture for incremental learning of analog multidimensional maps. IEEE Trans Neural Netw 3:698–713

Carpenter GA, Milenova B, Noeske B (1998) Distributed ARTMAP: a neural network for fast distributed supervised learning. Neural Netw 11:793–813

Dagher I, Georgiopoulos M, Heileman G, Bebis G (1998) Fuzzy ART-Var: an improved fuzzy ARTMAP algorithm. In: Proceedings of IEEE world congress computational intelligence WCCI'98, pp 1688–1693

Dagher I, Georgiopoulos M, Heileman GL, Bebis G (1999) An ordering algorithm for pattern presentation in fuzzy ARTMAP that tends to improve generalization performance. IEEE Trans Neural Netw 10:768–778

Efron B (1979) Bootstrap methods: another look at the jackknife. Ann Stat 7:1–26

Gomez-Sanchez E, Dimitriadis Y, Cano-Izquierdo J, Lopez-Coronado J (2002) ARTMAP: use of mutual information for category reduction in fuzzy ARTMAP. IEEE Trans Neural Netw 13:58–69

Guo GD, Li SZ (2003) Content-based audio classification and retrieval by support vector machines. IEEE Trans Neural Netw 14:209–214

Hettich S, Blake CL, Merz CJ (1998) UCI repository of machine learning databases [http://www.ics.uci.edu/∼mlearn/MLRepository.html]. Department of Information and Computer Science, University of California, Irvine, CA

Ishibuchi H, Yamamoto T, Nakashima T (2005) Hybridization of fuzzy GBML approaches for pattern classification problems. IEEE Trans Syst Man Cybern B Cybern 35:359–365

Justino EJR, Bortolozzi F, Sabourin R (2005) A comparison of SVM and HMM classifiers in the off-line signature verification. Pattern Recognit Lett 26:1377–1385

Lim CP, Harrison RF (1997) An incremental adaptive network for on-line supervised learning and probability estimation. Neural Netw 10:925–939

Marriott S, Harrison RF (1995) A modified fuzzy ARTMAP architecture for the approximation of noisy mappings. Neural Netw 8:619–641

Moody MJ, Darken CJ (1989) Fast learning in networks of locally-tuned processing units. Neural Comput 1:281–294

Pernkopfa F (2005) Bayesian network classifiers versus selective k-NN classifier. Pattern Recognit 38:1–10

Polikar R, Udpa L, Udpa SS, Honovar V (2001) Learn++: an incremental learning algorithm for supervised neural networks. IEEE Trans Syst Man Cybern C 31:497–508

Reilly DL, Cooper LN, Elbaum C (1982) A neural model for category learning. Biol Cybern 45:35–41

Ripley BD (1994) Neural networks and related methods for classification. J R Stat Soc B 56:409–456

Roy A (2000) Artificial neural networks—a science in trouble. ACM SIGKDD Explor 1:33–38

Simpson PK (1992) Fuzzy min–max neural networks—part 1: classification. IEEE Trans Neural Netw 3:776–786

Verzi S, Heileman G, Georgiopoulos M, Healy M (1998) Boosted ARTMAP. In: Proceedings of IEEE world congress computational intelligence WCCI'98, pp 396–400

Vigdor B, Lerner B (2006) Accurate and fast off and online fuzzy ARTMAP-based image classification with application to genetic abnormality diagnosis. IEEE Trans Neural Netw 17:1288–1300

Williamson J (1996) Gaussian ARTMAP: a neural network for fast incremental learning of noisy multidimensional maps. Neural Netw 9:881–897

Wu Y, Ianakiev K, Govindaraju V (2002) Improved k-nearest neighbor classification. Pattern Recognit 35:2311–2318

Zhang GP (2000) Neural networks for classification: a survey. IEEE Trans Syst Man Cybern C Appl Rev 30:451–462