CHAPTER 1.4

MATCH TRACKING STRATEGIES FOR FUZZY ARTMAP NEURAL NETWORKS

Eric Granger[1]*, Philippe Henniges[1], Robert Sabourin[1] and Luiz S. Oliveira[2]

[1] *Laboratoire d'imagerie, de vision et d'intelligence artificielle, École de technologie supérieure (ETS), Montreal, Canada*

[2] *Dept. de Informática Aplicada, Pontifícia Universidade Católica do Paraná (PUCPR), Curitiba, Brazil*

Training fuzzy ARTMAP neural networks for classification using data from complex real-world environments may lead to category proliferation, and yield poor performance. This problem is known to occur whenever the training set contains noisy and overlapping data. Moreover, when the training set contains identical input patterns that belong to different recognition classes, fuzzy ARTMAP will fail to converge. To circumvent these problems, some alternatives to the network's original match tracking (MT) process have been proposed in literature, such as using negative MT, and removing MT altogether. In this chapter, the MT parameter of fuzzy ARTMAP is optimized during training using a new Particle Swarm Optimisation (PSO)-based strategy, denoted `PSO(MT)`. The impact on fuzzy ARTMAP performance of training with different MT strategies is assessed empirically, using different synthetic data sets, and the NIST SD19 handwritten character recognition data set. During computer simulations, fuzzy ARTMAP is trained with the original (positive) match tracking (`MT+`), with negative match tracking (`MT-`), without MT algorithm (`WMT`), and with `PSO(MT)`. Through a comprehensive set of simulations, it has been observed that by training with `MT-`, fuzzy ARTMAP expends fewer resources than with other MT strategies, but can achieve a significantly higher generalization error, especially for data with overlapping class distributions. In particular, degradation of error in fuzzy ARTMAP performance due to overtraining is more pronounced for `MT-` than for `MT+`. Generalization error achieved using `WMT` is significantly higher than other strategies on data with complex non-linear decision bounds. Furthermore, the number of internal categories required to represent decision boundaries increases significantly. Optimizing the value of the match tracking parameter using `PSO(MT)` yields the lowest overall generalization error, and requires fewer internal categories than `WMT`, but generally more categories than `MT+` and `MT-`. However, this strategy requires a large number of training epochs to convergence. Based on this empirical results with `PSO(MT)`, the MT process as such can provide a significant increase to fuzzy ARTMAP performance, assuming that the MT parameter is tuned for the specific application in mind.

*__Corresponding author:__ ETS, 1100 Notre-Dame Ouest, Montreal, Quebec, H3C 1K3, Canada, email: `eric.granger@etsmtl.ca`, phone: 1-514-396-8650, fax: 1-514-396-8595.

1. Introduction

The fuzzy ARTMAP neural network is capable of self-organizing stable recognition categories in response to arbitrary sequences of analog or binary input patterns. It can perform fast, stable, on-line, unsupervised or supervised, incremental learning, classification, and prediction.[6,7] As such, it has been successfully applied in complex real-world pattern recognition tasks such as the recognition of radar signals,[15,32] multi-sensor image fusion, remote sensing and data mining,[9,31,35,40] recognition of handwritten characters,[3,13,23] and signature verification.[29]

A drawback of fuzzy ARTMAP is its ability to learn decision boundaries between class distributions that consistently yield low generalization error for a wide variety of pattern recognition problems. For instance, when trained for automatic classification of handwritten characters, fuzzy ARTMAP cannot achieve a level of performance that is competitive with some other well-known models.[16] Statistical models (*e.g.*, linear and quadratic discriminant function, Gaussian mixture classifier, k-Nearest-Neighbor (kNN) and Support Vector Machines (SVMs)), and neural networks(*e.g.*, Multi-Layer Perceptron (MLP) and Radial Basis Function (RBF) networks) are commonly used for classification due to their learning flexibility and inexpensive computation.[26] Such recognition problems typically exhibit complex decision boundaries, with moderate overlap between character classes.

In the context of batch supervised learning of a finite training set, the main factors affecting fuzzy ARTMAP's capacity to generalize are:

(1) internal dynamics of network: prototype choice and class prediction functions, learning rule, match tracking process, hyper-parameter values, and representation of categories with hyper-rectangles.
(2) learning process: supervised learning strategy (and thus, the number of training epochs), proportion of patterns in the training subset to those in validation and test subsets, user-defined hyper-parameter values, data normalisation technique, sequential gradient-based learning, and data presentation order.
(3) data set structure: overlap and dispersion of patterns, etc., and therefore of the geometry of decision boundaries among patterns belonging to different recognition classes.

Several ARTMAP networks have been proposed to refine the decision boundaries created by fuzzy ARTMAP. For instance, many variants attempt to improve the accuracy of fuzzy ARTMAP predictions by providing for probabilistic (density based) predictions.[10,14,25,36,39,41]

When learning data from complex real-world environments, fuzzy ARTMAP is known to suffer from overtraining, often referred to in literature as the category proliferation problem. It occurs when the training data set contains noisy and overlapping class distributions.[19,22,24] In this case, increasing the amount of training data requires significantly more internal category neurons, and therefore computational complexity, while yielding a higher generalisation error. The cate-

gory proliferation problem is directly connected to the match tracking (MT) process of fuzzy ARTMAP. During fuzzy ARTMAP training, when a mismatch occurs between predicted and desired output responses, MT allows selecting alternate category neurones.

The match tracking process is parameterized by hyper-parameter ϵ, and was originally introduced as a small positive value.[10] In fuzzy ARTMAP literature, this parameter is commonly set to a value ($\epsilon = 0^+$) that allows to minimize network resources. Such a choice may however contribute to overtraining, and significantly degrade the capacity to generalize. As a result, some authors have studied the impact on performance of removing the MT altogether, and conclude that the usefulness of MT is questionable.[2,27] However, training without MT may lead to a network with a greater number of internal categories, and possibly a higher generalization error.

In an extreme case, a well known convergence problem occurs when learning inconsistent cases – identical training subset patterns that belong to different classes.[10] The consequence is a failure to converge, as identical prototypes linked to these inconsistent cases proliferate. This anomalous situation is a result of the original match tracking process. This convergence problem may be circumvented by using the feature of ARTMAP-IC[10] called *negative* match tracking (*i.e.,* setting $\epsilon = 0^-$ after mismatch reset). This allows fuzzy ARTMAP training to converge and find solutions with fewer internal categories, but may however lead to a higher generalization error.

In this chapter, the impact on fuzzy ARTMAP performance of training with different MT strategies – the original positive MT (`MT+`), negative MT (`MT-`) and without MT (`WMT`) - is assessed empirically. As an alternative, a Particle Swarm Optimization (PSO)-based approach called `PSO(MT)` is used to optimize the value of MT hyper-parameter ϵ during fuzzy ARTMAP training, such that the generalization error is minimized. The architecture, weights, and MT parameter are in effect selected to minimize generalisation error by virtue of ARTMAP training, which allows to grow the network architecture (*i.e.,* the number of category neurons) with the problem's complexity. An experimental protocol has been defined such that the generalization error and resource requirements of fuzzy ARTMAP trained with different MT strategies may be compared using different types of pattern recognition problems. The first two types consist of synthetic data with overlapping class distributions, and with complex decision boundaries but no overlap, respectively. The third type consists of real-world data - handwritten numerical characters extracted from the NIST SD19.

In the next section, the MT strategies for fuzzy ARTMAP training are briefly reviewed. Section III presents the experimental methodology, e.g., protocol, data sets and performance measures employed for proof of concept computer simulations. Section IV presents and discuss experimental results obtained with synthetic and NIST SD19 data.

2. Fuzzy ARTMAP Match Tracking

2.1. *The fuzzy ARTMAP neural network:*

ARTMAP refers to a family of neural network architectures based on Adaptive Resonance Theory (ART)[4] that is capable of fast, stable, on-line, unsupervised or supervised, incremental learning, classification, and prediction[6,7]. ARTMAP is often applied using the simplified version shown in Figure 2. It is obtained by combining an unsupervised ART neural network[4] with a map field.

In order to improve the performance or robustness to noise of ARTMAP architectures, several variants have been proposed in literature. Some networks, such as fuzzy ARTMAP, ART-EMAP, ARTMAPT-PI, ARTMAP-IC, Default ARTMAP, Simplified ARTMAP, and Distributed ARTMAP, represent each class using one or more fuzzy set hyper-rectangle, and perform category activation using an L_1 norm. Other networks, such as PROBART, Probabilistic Fuzzy ARTMAP, MLANS, Gaussian ARTMAP, Ellipsoid ARTMAP, boosted ARTMAP, and μARTMAP, represent each class using one or more probability density functions (pdfs). The class predictions of probabilistic ARTMAP variants consist in estimating the posterior probability that each class generated a given input pattern. Then, the Bayes decision procedure may be applied to assign one-of-L possible classes to the input according to the maximum posterior probability decision rule. This rule defines decision boundaries among classes that yield the minimum probability of misclassification.

Fuzzy ARTMAP[7] is one of the earliest and most popular ARTMAP architecture. It can process both analog and binary-valued input patterns by employing fuzzy ART[5] as the ART network. The fuzzy ART neural network consists of two fully connected layers of nodes: an M node input layer, F_1, and an N node competitive layer, F_2. A set of real-valued weights $\mathbf{W} = \{w_{ij} \in [0,1] : i = 1,2,...,M;\ j = 1,2,...,N\}$ is associated with the F_1-to-F_2 layer connections. Each F_2 node j represents a recognition category that learns a prototype vector $\mathbf{w}_j = (w_{1j}, w_{2j}, ..., w_{Mj})$. The F_2 layer of fuzzy ART is connected, through learned associative links, to an L node map field F^{ab}, where L is the number of classes in the output space. A set of binary weights $\mathbf{W}^{ab} = \{w^{ab}_{jk} \in \{0,1\} : j = 1,2,...,N;\ k = 1,2,...,L\}$ is associated with the F_2-to-F^{ab} connections. The vector $\mathbf{w}^{ab}_j = (w^{ab}_{j1}, w^{ab}_{j2}, ..., w^{ab}_{jL})$ links F_2 node j to one of the L output classes.

2.2. *Algorithm for supervised learning of fuzzy ARTMAP:*

In batch supervised training mode, ARTMAP classifiers learn an arbitrary mapping between training set patterns $\mathbf{a} = (a_1, a_2, ..., a_m)$ and their corresponding binary supervision patterns $\mathbf{t} = (t_1, t_2, ..., t_L)$. These patterns are coded to have unit value $t_K = 1$ if K is the target class label for $\mathbf{a}$, and zero elsewhere. The following steps describe fuzzy ARTMAP learning:

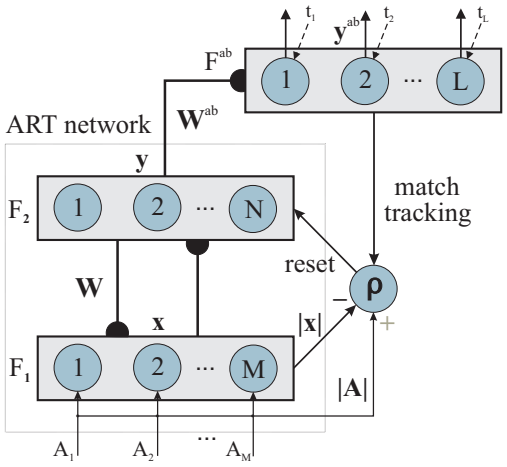


Fig. 1. An ARTMAP neural network architecture for pattern classification.

(1) **Initialisation:** Initially, all the F_2 nodes are uncommitted, all weight values w_{ij} are initialized to 1, and all weight values w_{jk}^{ab} are set to 0. An F_2 node becomes committed when it is selected to code an input vector $\mathbf{a}$, and is then linked to an F^{ab} node. Values of the learning rate $\beta \in [0,1]$, the choice $\alpha > 0$, the match tracking $0 < \epsilon \ll 1$, and the baseline vigilance $\bar{\rho} \in [0,1]$ hyper-parameters are set.

(2) **Input pattern coding:** When a training pair $(\mathbf{a}, \mathbf{t})$ is presented to the network, $\mathbf{a}$ undergoes a transformation called complement coding, which doubles its number of components. The complement-coded input pattern has $M = 2m$ dimensions and is defined by $\mathbf{A} = (\mathbf{a}, \mathbf{a}^c) = (a_1, a_2, ..., a_m; a_1^c, a_2^c, ..., a_m^c)$, where $a_i^c = (1 - a_i)$, and $a_i \in [0,1]$. The vigilance parameter ρ is reset to its baseline value $\bar{\rho}$.

(3) **Prototype selection:** Pattern $\mathbf{A}$ activates layer F_1 and is propagated through weighted connections $\mathbf{W}$ to layer F_2. Activation of each node j in the F_2 layer is determined by the *Weber law choice function*:

$$T_j(\mathbf{A}) = \frac{|\mathbf{A} \wedge \mathbf{w}_j|}{\alpha + |\mathbf{w}_j|} \,, \tag{1}$$

where $|\cdot|$ is the L^1 norm operator defined by $|\mathbf{w}_j| \equiv \sum_{i=1}^{M} |w_{ij}|$, $\wedge$ is the fuzzy AND operator, $(\mathbf{A} \wedge \mathbf{w}_j)_i \equiv \min(A_i, w_{ij})$, and α is the user-defined *choice parameter*. The F_2 layer produces a binary, winner-take-all pattern of activity $\mathbf{y} = (y_1, y_2, ..., y_N)$ such that only the node $j = J$ with the greatest activation value $J = \arg\max\{T_j : j = 1, 2, ..., N\}$ remains active; thus $y_J = 1$ and $y_j = 0, j \neq J$. If more than one T_j is maximal, the node j with the smallest index is chosen. Node J propagates its top-down expectation, or prototype vector $\mathbf{w}_J$, back onto F_1 and the *vigilance test* is performed. This test compares the degree

of match between $\mathbf{w}_J$ and $\mathbf{A}$ against the dimensionless *vigilance parameter* $\rho \in [0, 1]$:

$$\frac{|\mathbf{A} \wedge \mathbf{w}_J|}{|\mathbf{A}|} = \frac{|\mathbf{A} \wedge \mathbf{w}_J|}{M} \geq \rho \ . \tag{2}$$

If the test is passed, then node J remains active and resonance is said to occur. Otherwise, the network inhibits the active F_2 node (*i.e.,* T_J is set to 0 until the network is presented with the next training pair $(\mathbf{a}, \mathbf{t})$) and searches for another node J that passes the vigilance test. If such a node does not exist, an uncommitted F_2 node becomes active and undergoes learning (Step 5). The depth of search attained before an uncommitted node is selected is determined by the choice parameter α.

(4) **Class prediction:** Pattern $\mathbf{t}$ is fed directly to the map field F^{ab}, while the F_2 category $\mathbf{y}$ learns to activate the map field via associative weights $\mathbf{W}^{ab}$. The F^{ab} layer produces a binary pattern of activity $\mathbf{y}^{ab} = (y_1^{ab}, y_2^{ab}, ..., y_L^{ab}) = \mathbf{t} \wedge \mathbf{w}_J^{ab}$ in which the most active F^{ab} node $K = \arg\max\{y_k^{ab} : k = 1, 2, ..., L\}$ yields the class prediction $(K = k(J))$. If node K constitutes an incorrect class prediction, then a *match tracking* (MT) signal raises the vigilance parameter ρ such that:

$$\rho = \frac{|\mathbf{A} \wedge \mathbf{w}_J|}{M} + \epsilon \ , \tag{3}$$

where $\epsilon = 0^+$, to induce another search among F_2 nodes (Step 3). This search continues until either an uncommitted F_2 node becomes active, and learning ensues (Step 5), or a node J that has previously learned the correct class prediction K becomes active.

(5) **Learning:** Learning input $\mathbf{a}$ involves updating prototype vector $\mathbf{w}_J$, and, if J corresponds to a newly-committed node, creating an associative link to F^{ab}. The prototype vector of F_2 node J is updated according to:

$$\mathbf{w}'_J = \beta(\mathbf{A} \wedge \mathbf{w}_J) + (1 - \beta)\mathbf{w}_J \ , \tag{4}$$

where β is a fixed *learning rate parameter*. The algorithm can be set to slow learning with $0 < \beta < 1$, or to fast learning with $\beta = 1$. With complement coding and fast learning, fuzzy ARTMAP represents category j as an m-dimensional hyperrectangle R_j that is just large enough to enclose the cluster of training set patterns $\mathbf{a}$ to which it has been assigned. That is, an M-dimensional prototype vector $\mathbf{w}_j$ records the largest and smallest component values of training subset patterns $\mathbf{a}$ assigned to category j. The vigilance test limits the growth of hyperrectangles – a ρ close to 1 yields small hyperrectangles, while a ρ close to 0 allows large hyperrectangles. A new association between F_2 node J and F^{ab} node K $(k(J) = K)$ is learned by setting $w_{Jk}^{ab} = 1$ for $k = K$, where K is the target class label for $\mathbf{a}$, and 0 otherwise. The next training subset pair $(\mathbf{a}, \mathbf{t})$ is presented to the network in Step 2.

Network training proceeds from one epoch to the next, and is halted for validation after each epoch [a]. Given a finite training data set, batch supervised learning ends after the epoch for which the generalisation error is minimized on an independent validation data set. With the large data sets considered in this chapter, learning through this hold-out validation (`HV`) is an appropriate validation strategy. If data were limited, k-fold cross-validation would be a more suitable strategy, at the expense of some estimation bias due to crossing.[19,34]

Once the weights $\mathbf{W}$ and $\mathbf{W}^{ab}$ have been found through this process, ARTMAP can predict a class label for an input pattern by performing Steps 2, 3 and 4 without any vigilance or match tests. During testing, a pattern $\mathbf{a}$ that activates node J is predicted to belong to class $K = k(J)$. The time complexity required to process one input pattern, during either a training or testing phase, is $O(MN)$.

2.3. *Match tracking strategies:*

During training, when a mismatch occurs between a predicted response $\mathbf{y}^{ab}$ and a desired response $\mathbf{t}$ for an input pattern $\mathbf{a}$, the original positive MT process (`MT+`) of fuzzy ARTMAP raises the internal vigilance parameter to $\rho = (|\mathbf{A} \wedge \mathbf{w}_J|)(M)^{-1} + \epsilon$ in order to induce another search among F_2 category nodes. `MT+` is parameterized by the MT hyper-parameter ϵ, which was introduced as a small positive value, $0 < \epsilon \ll 1$.[7]

It is well documented that training fuzzy ARTMAP with data from noisy and overlapping class distributions may lead to category proliferation, and that this problem is connected to the MT mechanism. Overlapping between classes, which is responsible for misclassifications during training, requires MT to find a more suitable category for the misclassified pattern. The selected $F2$ node requires a larger prototype vector, and thus a smaller size to pass the vigilance test. Such misclassifications are responsible for the formation of a large number of small categories within the overlapping area, many of which contribute little to the classification process.[24] Category proliferation is intensified with the degree of class overlap.

Category proliferation is an indication of overtraining. Increasing the amount of training data requires significantly more resources (*i.e.,* the number of internal category neurons, thus memory space and computational complexity), yet provides a higher generalisation error.[19,22,24] In addition, the MT parameter is commonly set to the value $\epsilon = +0.001$ in fuzzy ARTMAP literature to minimize network resources.[10] Such a choice may play a significant role in category proliferation, and considerably degrade the capacity to generalize.

Although pruning may help reduce category proliferation, some authors have challenged the need for a MT process.[1,27,37] Training without MT (`WMT`) implies creating a new category each time that a predictive response $\mathbf{y}^{ab}$ does not match a desired response $\mathbf{t}$. When an node in the $F2$ layer is chosen to represent an input

[a] An *epoch* is defined as one complete presentation of all the patterns of the training set.

pattern, but this node is mapped to the incorrect label, an uncommitted node is activated to represent this pattern. Note that training fuzzy ARTMAP `WMT` is equivalent to performing MT but setting $\epsilon = 1$. Training `WMT` may create networks with a greater number of internal categories, and possibly a higher generalization error.

In an extreme case, a convergence problem occurs whenever the training set contains identical patterns that belong recognition classes.[10] The effect is a proliferation of identical prototypes associated with the *inconsistent cases*, and a failure to converge. Consider for example that on the first training epoch, fuzzy ARTMAP learns two completely overlapping, minimum-sized prototypes, $\mathbf{w}_{A.1}$ (linked to class A) and $\mathbf{w}_{B.1}$ (linked to class B), for two identical pulse patterns, $\mathbf{a}_1$ and $\mathbf{a}_2$. In a subsequent epoch, $\mathbf{w}_{A.1}$ is initially selected to learn $\mathbf{a}_2$, since $T_{A.1} = T_{B.1} \simeq 1$, and $\mathbf{w}_{A.1}$ was created prior to $\mathbf{w}_{B.1}$ (index $A.1$ is smaller than $B.1$). Since $\mathbf{w}_{A.1}$ is not linked to class B, mismatch reset raises the vigilance parameter ρ to $(|\mathbf{A}_2 \wedge \mathbf{w}_{A.1}|/M) + \epsilon$, where $|\mathbf{A}_2 \wedge \mathbf{w}_{A.1}| = |\mathbf{A}_2 \wedge \mathbf{w}_{B.1}|$. As a result, $\mathbf{w}_{B.1}$ can no longer pass the vigilance test required to become selected for $\mathbf{a}_2$, and fuzzy ARTMAP must create another minimum-sized prototype $\mathbf{w}_{B.2} = \mathbf{w}_{B.1}$. From epoch to epoch, the same phenomenon repeats itself, yielding ever more prototypes $\mathbf{w}_{B.\mathrm{n}} = \mathbf{w}_{B.1}$ for n $= 3, 4, ..., \infty$.

ARTMAP-IC[10] is an extension of fuzzy ARTMAP that produce a binary winner-take-all pattern $\mathbf{y}$ when training, but use distributed activation of coded $F2$ nodes when testing. ARTMAP-IC is further extended in two ways. First, it biases distributed test set predictions according to the number of times $F2$ nodes are assigned to training set patterns. Second, it uses a *negative MT* process (`MT-`) to address the problem of inconsistent cases, whereby identical training set patterns correspond to different classes labels.

With negative MT (`MT-`), ρ is also initially raised after mismatch reset, but is allowed to decay slightly before a different node J is selected. Then, the MT parameter is set to a small negative value, $\epsilon \leq 0$ (typically a value of $\epsilon = -0.001$), which allows for identical inputs that predict different classes to establish distinct recognition categories. In the example above, mismatch reset raises ρ but $\mathbf{w}_{B.1}$ would still pass the vigilance test. This allows to learn fully overlapping prototypes for training set patterns that belong to different classes.

In some applications, incorporation into fuzzy ARTMAP of the `MT-` feature of ARTMAP-IC may be essential to avoid the convergence problem observed with original `MT+`. Training fuzzy ARTMAP with `MT-` would thereby find solutions with fewer internal categories, but may nonetheless lead to a higher generalization error. In other cases, MT reset may also be buffered based on a category's previous predictive success, thereby improving the compression achieved with minimal loss of accuracy.[18] During supervised learning, match tracking search allocates memory based on the degree of similarity between newly encountered and previously encountered inputs, regardless of their prior predictive success.

An alternate approach consists in optimizing the MT hyper-parameter during batch supervised learning of a fuzzy ARTMAP neural network. In effect, both network (weights and architecture) and ϵ values are co-optimized for a given problem, using the same cost function. The next Subsection presents a Particle Swarm Optimization (PSO)-based approach called `PSO(MT)` that automatically selects a value (magnitude and polarity) of ϵ during fuzzy ARTMAP training such that the generalization error is minimized. This approach is based on the PSO training strategy proposed in,[16] but focused only on a one-dimensional optimization space of $\epsilon \in [-1, 1]$.

2.4. *Particle Swarm Optimisation (PSO) of the match tracking parameter*

PSO is a population-based stochastic optimization technique that was inspired by social behavior of bird flocking or fish schooling.[20] It shares many similarities with evolutionary computation techniques such as genetic algorithms (GAs), yet has no evolution operators such as crossover and mutation. PSO belongs to the class of evolutionary algorithm techniques that does not utilize the "survival of the fittest" concept, nor a direct selection function. A solution with lower fitness values can therefore survive during the optimization and potentially visit any point of the search space.[12] Finally, while GAs were conceived to deal with binary coding, PSO was designed, and proved very effective, in solving real valued global optimization problems, which makes it suitable for this study.

With PSO, each *particle* corresponds to a single solution in the search space, and the population of particles is called a *swarm*. All particles are assigned position values which are evaluated according to the fitness function being optimized, and velocities values which direct their movement. Particles move through the search space by following the particles with the best fitness. Assuming a d-dimensional search space, the position of particle i in an P-particle swarm is represented by a d-dimensional vector $\mathbf{s}_i = (s_{i1}, s_{i2}, \ldots, s_{id})$, for $i = 1, 2, \ldots, P$. The velocity of this particle is denoted by vector $\mathbf{v}_i = (v_{i1}, v_{i2}, \ldots, v_{id})$, while the best previously-visited position of this particle is denoted as $\mathbf{p}_i = (p_{i1}, p_{i2}, \ldots, p_{id})$. For each new iteration $q + 1$, the velocity and position of particle i are updated according to:

$$\mathbf{v}_i^{q+1} = w^q \mathbf{v}_i^q + c_1 r_1 (\mathbf{p}_i^q - \mathbf{s}_i^q) + c_2 r_2 (\mathbf{p}_g^q - \mathbf{s}_i^q) \quad (5)$$

$$\mathbf{s}_i^{q+1} = \mathbf{s}_i^q + \mathbf{v}_i^{q+1} \quad (6)$$

where $\mathbf{p}_g$ represents the global best particle position in the swarm, w^q is the particle inertia weight, c_1 and c_2 are two positive constants called cognitive and social parameters, respectively, and r_1 and r_2 are random numbers uniformly distributed in the range [0,1].

The role of w^q in Equation 5 is to regulate the trade-off between exploration and exploitation. A large inertia weight facilitates global search (exploration), while a small one tends to facilitate fine-tuning the current search area (exploitation).

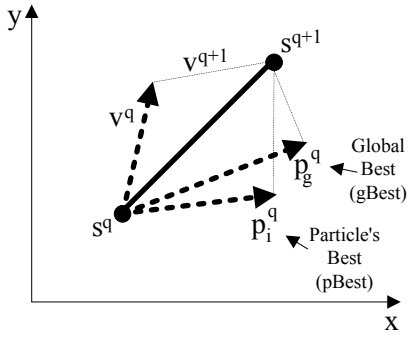


Fig. 2. An ARTMAP neural network architecture specialized for pattern classification.

This is why inertia weight values are defined by some monotonically decreasing function of q. Proper fine-tuning of c_1 and c_2 may result in faster convergence of the algorithm and alleviation of the local minima. Kennedy and Eberhart propose that the cognitive and social scaling parameters be selected such that $c_1 = c_2 = 2$.[21] Finally, the parameters r_1 and r_2 are used to maintain the diversity of the population. Figure **??** depicts the update by PSO of a particle's position from $\mathbf{s}_i^q$ to $\mathbf{s}_i^{q+1}$.

Algorithm 1 shows the pseudo-code of a PSO learning strategy specialized for supervised training of fuzzy ARTMAP neural networks. It essentially seeks to minimize fuzzy ARTMAP generalisation error $E(\mathbf{s}_i^q)$ in the d-dimensional space of hyper-parameter values. For enhanced computational throughput and global search capabilities, Algorithm 1 is inspired by the synchronous parallel version of PSO.[33] It utilizes a basic type of neighborhood called global best or *gbest*, which is based on a sociometric principle that conceptually connects all the members of the swarm to one another. Accordingly, each particle is influenced by the very best performance of any member of the entire swarm. Exchange of information only takes place among the particle's own experience (the location of its personal best $\mathbf{p}_i^q$, lbest), and the experience of the best particle in the swarm (the location of the global best $\mathbf{p}_g^q$, gbest).

The `PSO(MT)` approach is obtained by setting $d = 1$, and particle positions to MT parameter values, $\mathbf{s}_i^q = \epsilon_i^q$. Measurement of any fitness values $E(\mathbf{s}_i^q)$ in this algorithm involves computing the generalisation error on a validation subset for the fuzzy ARTMAP network which has been trained using the MT parameter value at particle position ϵ_i^q. When selecting $\mathbf{p}_i^q$ or $\mathbf{p}_g^q$, if the two fitness values being compared are equal, then the particle/network requiring fewer number of $F2$ category nodes is chosen. The same training and validation sets are used throughout this process. Following the last iteration of Algorithm 1, the overall generalisation error is computed on a test set for the network corresponding to position $\mathbf{p}_g^q$.

Algorithm 1: PSO learning strategy for fuzzy ARTMAP.

A. Initialization:

set the maximum number of iterations $q_{\max}$ *and/or fitness objective* E^*

set PSO parameters P, $\boldsymbol{v}max$, w^0, c_1, c_2, r_1 *and* r_1

initialize particle positions at random such that $\boldsymbol{p}_g^0$, $\boldsymbol{s}_i^0$ *and* $\boldsymbol{p}_i^0 \in [-1,1]^d$, *for* $i = 1, 2, \ldots, P$

initialize particle velocities at random such that $0 \leq \mathbf{v}_i^0 \leq \boldsymbol{v}max$, *for* $i = 1, 2, \ldots, P$

B. Iterative process:

set iteration counter $q = 0$

while $q \leq q_{\max}$ *or* $E(\boldsymbol{p}_g^q) \geq E^*$ **do**

  for $i = 1, 2, \ldots, P$ **do**

    train fuzzy ARTMAP using hold-out validation and $\boldsymbol{s}_i^q$

    compute fitness value $E(\boldsymbol{s}_i^q)$ *of resulting network*

    if $E(\boldsymbol{s}_i^q) < E(\boldsymbol{p}_i^q)$ **then**

      update particle's best personal position: $\boldsymbol{p}_i^q = \boldsymbol{s}_i^q$

    end

  end

  select the particle with best global fitness:

  $g = \arg\min\{E(\boldsymbol{s}_i^q) : i = 1, 2, \ldots, P\}$

  for $i = 1, 2, \ldots, P$ **do**

    update velocity: $\boldsymbol{v}_i^{q+1} = w^q \boldsymbol{v}_i^q + c_1 r_1 (\boldsymbol{p}_i^q - \boldsymbol{s}_i^q) + c_2 r_2 (\boldsymbol{p}_g^q - \boldsymbol{s}_i^q)$

    update position: $\boldsymbol{s}_i^{q+1} = \boldsymbol{s}_i^q + \boldsymbol{v}_i^{q+1}$

  end

  $q = q + 1$

  update particle inertia w^q

end

3. Experimental Methodology

To assess the performance achieved by fuzzy ARTMAP using MT strategies, several data sets were selected for computer simulations. Four synthetic data sets are representative of pattern recognition problems that involve either (1) simple decision boundaries with overlapping class distributions, or (2) complex decision boundaries, were class distributions do not overlap on decision boundaries. A set of handwritten numerical characters from the NIST SD19 database is representative of complex real-world pattern recognition problems. Prior to a simulation trial, these data sets were normalized according to the min-max technique, and partitioned into three parts – training, validation, and test subset.

During each simulation trial, the performance of fuzzy ARTMAP is compared from a perspective of different training subset size, and match tracking strategies. In order to assess the effect on performance of training subset size, the number of training subset patterns used for supervised learning was progressively increased, while corresponding validation and test subsets were held fixed. The performance is compared for fuzzy ARTMAP neural networks trained according to four different

MT strategies: `MT+` ($\epsilon = 0.001$), `MT-` ($\epsilon = -0.001$), `WMT` (equivalent to setting $\epsilon = 1$) and `PSO(MT)`. Training is performed by setting the other three hyper-parameters such that the resources (number of categories, training epochs, etc.) are minimized: $\alpha = 0.001$, $\beta = 1$ and $\overline{\rho} = 0$. In all cases, training is performed using the HV strategy[34] described in Subsection 2.2.

The `PSO(MT)` strategy also uses the hold-out validation technique on fuzzy ARTMAP network to calculate the fitness of each particle, and therefore find the network and ϵ value that minimize generalization error. Other fuzzy ARTMAP hyper-parameters are left unchanged. In all simulations involving PSO, the search space of the MT parameter was set to the following range of $\epsilon \in [-1, 1]$. Each simulation trial was performed with $P = 15$ particles, and ended after a maximum of $q_{\max} = 100$ iterations (although none of our simulations have ever attained that limit). A fitness objective E^* was not considered to end training, but a trial was ended if the global best fitness $E(\mathbf{p}_g^q)$ is constant for 10 consecutive iterations. The initial position $\mathbf{s}_1^0$ of one particle was set according to `MT-` ($\epsilon = -0.001$). All the remaining particle vectors were initialized randomly, according to a uniform distribution in the search space. The PSO parameters were set as follows: $c_1 = c_2 = 2$; r_1 and r_2 were random numbers uniformly distributed in [0,1]; w^q was decreased linearly from 0.9 to 0.4 over the $q_{\max}$ iterations; the maximum velocity $\mathbf{v}_{\max}$ was set to 0.2. At the end of a trial, the fuzzy ARTMAP network with the best global fitness value $\mathbf{p}_g^q$ was retained. Independently trials were repeated 4 times[b] with different initializations of particle vectors, and the network with greatest $\mathbf{p}_g^q$ of the four was retained.

Since fuzzy ARTMAP performance is sensitive to the presentation order of the training data, each simulation trial was repeated 10 times with either 10 different randomly generated data sets (synthetic data), or 10 different randomly selected data presentation orders (NIST SD19 data). The average performance of fuzzy ARTMAP was assessed in terms of resources required during training, and its generalisation error on the test sets. The amount of resources required during training is measured by compression and convergence time. *Compression* refers to the average number of training patterns per category prototype created in the F2 layer. *Convergence time* is the number of epochs required to complete learning for a learning strategy. It does not include presentations of the validation subset used to perform hold-out validation. *Generalisation error* is estimated as the ratio of incorrectly classified test subset patterns over all test set patterns. Given that compression indicates the number of $F2$ nodes, the combination of compression and convergence time provides useful insight into the amount of processing required during training to produce its best asymptotic generalisation error. Average results, with corresponding standard error, are always obtained as a result of the 10 independent

[b]From previous study with our data sets, it was determined that performing 4 independent trials of the PSO learning strategy with only 15 particles leads to better optimization results than performing 1 trial with 60 particles.

simulation trials.

The Quadratic Bayes classifier (CQB) and k-Nearest-Neighbour with Euclidean distance (kNN) classifier were included for reference with generalisation error results. These are classic parametric and non-parametric classification techniques from statistical pattern recognition, which are immune to the effects of overtraining. For each computer simulation, the value of k employed with kNN was selected among $k = 1$, 3, 5, 7, and 9, using hold-out validation. The rest of this section gives some additional details on the synthetic and real data sets employed during computer simulations.

3.1. *Synthetic data sets:*

All four synthetic data sets described below are composed of a total of 30,000 randomly-generated patterns, with 10,000 patterns for the training, validation, and test subsets. They correspond to 2 class problems, with a 2 dimensional input feature space. Each data subset is composed of an equal number of 5,000 patterns per class. In addition, the area occupied by each class is equal. During simulation trials, the number of training subset patterns used for supervised learning was progressively increased from 10 to 10,000 patterns according to a logarithmic rule: 5, 6, 8, 10, 12, 16, 20, 26, 33, 42, 54, 68, 87, 110, 140, 178, 226, 286, 363, 461, 586, 743, 943, 1197, 1519, 1928, 2446, 3105, 3940, 5000 patterns per class. This corresponds to 30 different simulation trials over the entire 10,000 pattern training subset.

These data sets have been selected to facilitate the observation of fuzzy ARTMAP behavior on different tractable problems. Of the four sets, two have simple linear decision boundaries with overlapping class distributions, $D_\mu(\xi_{tot})$ and $D_\sigma(\xi_{tot})$, and two have complex non-linear decision boundaries without overlap, D_{CIS} and D_{P2}. The total theoretical probability of error associated with D_μ and D_σ is denoted by ξ_{tot}. Note that with D_{CIS} and D_{P2}, the length of decision boundaries between class distributions is longer, and fewer training patterns are available in the neighborhood of these boundaries than with $D_\mu(\xi_{tot})$ and $D_\sigma(\xi_{tot})$. In addition, note that the total theoretical probability of error with D_{CIS} and D_{P2} is 0, since class distributions do not overlap on decision boundaries. The four synthetic data sets are now described.

$D_\mu(\xi_{tot})$. As represented in Figure 3(a), this data consists of two classes, each one defined by a multivariate normal distribution in a two dimensional input feature space. It is assumed that data is randomly generated by sources with the same Gaussian noise. Both sources are described by variables that are independent and have equal variance σ^2, therefore distributions are hyperspherical. In fact, $D_\mu(\xi_{tot})$ refers to 13 data sets, where the degree of overlap, and thus the total probability of error between classes differs for each set. The degree of overlap is varied from a total probability of error, $\xi_{tot} = 1\%$ to $\xi_{tot} = 25\%$, with 2% increments, by adjusting the mean vector μ_2 of class 2.

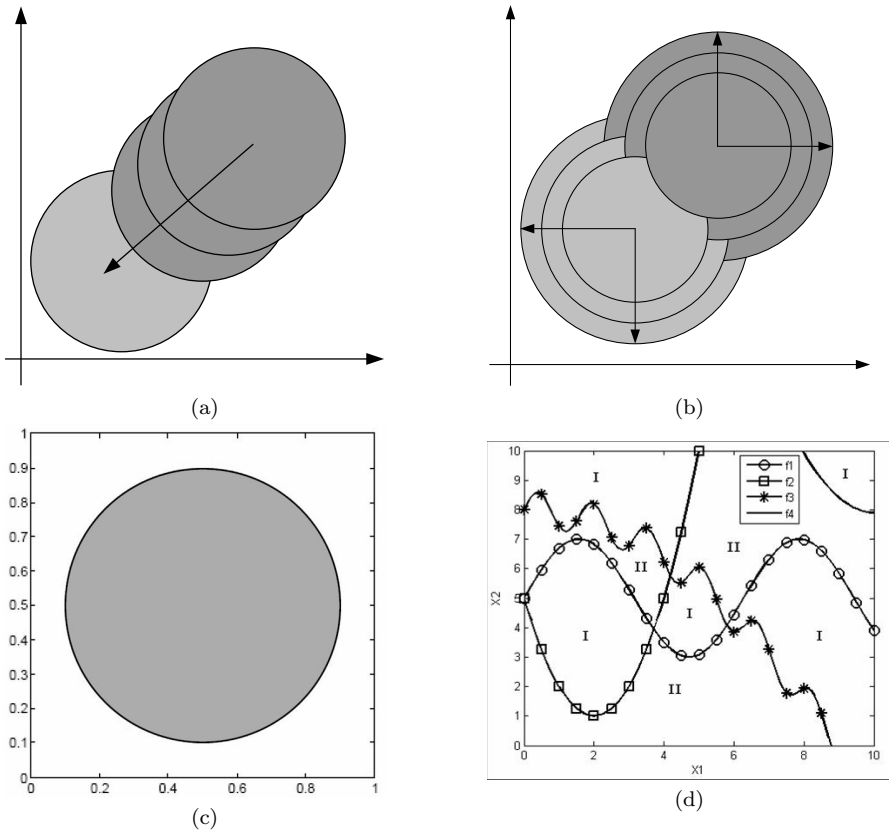


Fig. 3. Representation of the synthetic data sets used for computer simulations: (a) $D_{\mu}(\xi_{tot})$, (b) $D_{\sigma}(\xi_{tot})$, (c) D_{CIS} and (d) D_{P2}.

$D_{\sigma}(\xi_{tot})$. As represented in Figure 3(b), this data is identical to $D_{\mu}(\xi_{tot})$, except that the degree of overlap between classes is varied by adjusting the variance σ_2^2 of both classes. Note that for a same degree of overlap, $D_{\sigma}(\xi_{tot})$ data sets have a larger overlap boundary than $D_{\mu}(\xi_{tot})$ yet they are not as dense.

D_{CIS}. As represented in Figure 3(c), the Circle-in-Square problem[6] requires a classifier to identify the points of a square that lie inside a circle, and those that lie outside a circle. The circle's area equals half of the square. It consists of one non-linear decision boundary where classes do not overlap.

D_{P2}. As represented in Figure 3(d), each decision region of the D_{P2} problem is delimited by one or more of the four following polynomial and trigonometric functions:

$$f_1(x) = 2\sin(x) + 5 \tag{7}$$

$$f_2(x) = (x-2)^2 + 1 \tag{8}$$

$$f_3(x) = -0.1x^2 + 0.6\sin(4x) + 8 \tag{9}$$

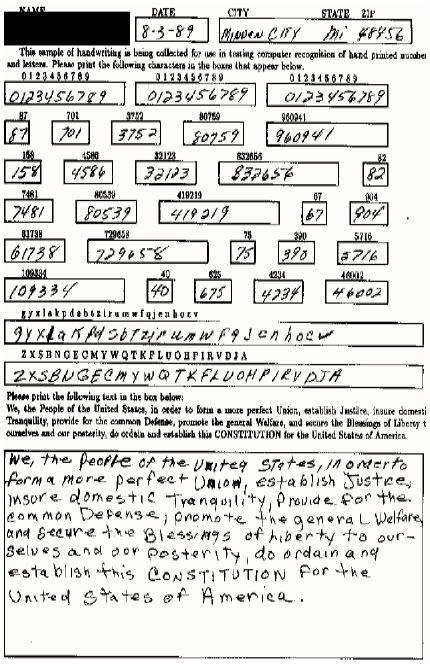
$$f_4(x) = \frac{(x-10)^2}{2} + 7.902 \tag{10}$$

and belongs to one of the two classes, indicated by the Roman numbers I and II.[38] It consists of four non-linear boundaries, and class definitions do not overlap. Note that equation $f_4(x)$ was slightly modified from the original equation such that the area occupied by each class is approximately equal.

3.2. *NIST Special Database 19 (SD19):*

Automatic reading of numerical fields has been attempted in several domains of application such as bank cheque processing, postal code recognition, and form processing. Such applications have been very popular in handwriting recognition research, due to the availability of relatively inexpensive CPU power, and to the possibility of considerably reducing the manual effort involved in these tasks.[30]

The NIST SD19[17] data set has been selected due to the great variability and difficulty of such handwriting recognition problems (see Figure 4). It consists of images of *handwritten sample forms* (hsf) organized into eight series, hsf-{0,1,2,3,4,6,7,8}.



(a)

(b)

Fig. 4. Examples in the NIST SD19 data of: (a) a handwriting sample form, and (b) some images of handwritten digits extracted from the forms.

SD19 is divided in 3 sections which contains samples representing isolated handwritten digits ('0', '1', ..., '9') extracted from hsf-{0123}, hsf-7 and hsf-4.

For our simulations, the data in hsf-{0123} has been further divided into training subset (150,000 samples), validation subset 1 (15,000 samples), validation subset 2 (15,000 samples) and validation subset 3 (15,000 samples). The training and validation subsets contain an equal number of samples per class. All 60,089 samples in hsf-7 has been used as a standard test subset. The distribution of samples per class in test sets is approximately equal.

The set features extracted for samples is a mixture of concavity, contour, and surface characteristics.[30] Accordingly, 78 features are used to describe concavity, 48 features are used to describe contour, and 6 features are used to describe surface. Each sample is therefore composed of 132 features that are normalized between 0 and 1 by summing up their respective feature values, and then dividing each one by its summation. With this feature set, the NIST SD19 data base exhibits complex decision boundaries, with moderate overlap between digit classes. Some experimental results obtained with Multi-Layer Perceptron (MLP), Support Vector Machine (SVM), and k-NN classifiers are reported in.[16]

During simulations, the number of training subset patterns used for supervised learning was progressively increased as from 100 to 150,000 patterns, according to a logarithmic rule. The 16 different training subset consist of the first 10, 16, 28, 47, 80, 136, 229, 387, 652, 1100, 1856, 3129, 5276, 8896, and all 15000 patterns per class.

4. Simulation Results

4.1. *Synthetic data with overlapping class distributions:*

Figure 5 presents the average performance obtained when fuzzy ARTMAP is trained with the four MT strategies – `MT-`, `MT+`, `WMT` and `PSO(MT)` – on $D_\mu(13\%)$. The generalisation errors for the Quadratic Bayes classifier (CQB), as well as the theoretical probability of error (ξ_{tot}), are also shown for reference.

As shown in Figure 5(a), `PSO(MT)` generally yields the lowest generalisation error over training set sizes, followed by `WMT`, `MT+`, and then `MT-`. With more than 20 training patterns per class, the error of both `MT-` and `MT+` algorithms tends to increase in a manner that is indicative of fuzzy ARTMAP overtraining.[19] However, with more than about 500 training patterns per class, the generalization error for `MT-` grows more rapidly with the training set size than for `MT+`, `WMT` and `PSO(MT)`. With a training set of 5000 patterns per class, a generalization error of about 21.22% is obtained with `MT+`, 26.17% with `MT-`, 16.22% with `WMT`, and 15.26% with `PSO(MT)`. The degradation in performance of `MT-` is accompanied by a notably higher compression and a lower convergence time than other MT strategies. `MT-` produces networks with fewer but larger categories than other MT strategies because of the

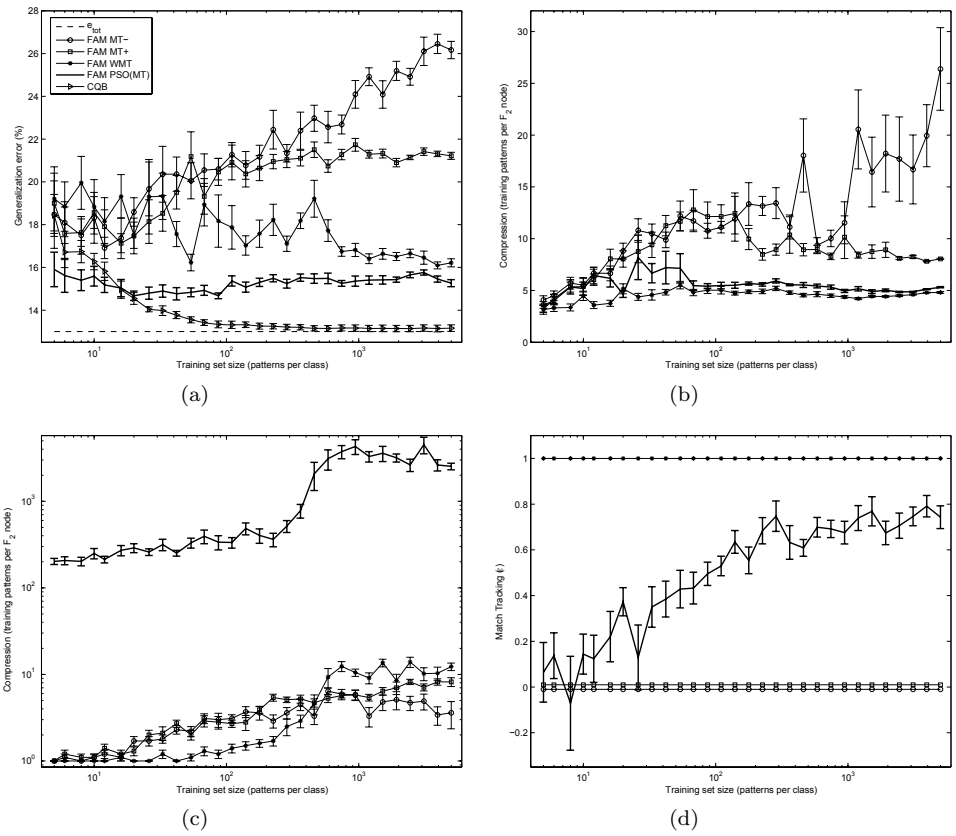


Fig. 5. Average performance of fuzzy ARTMAP (with `MT+`, `MT-`, `WMT` and `PSO(MT)`) versus training subset size for $D_\mu(\xi_{tot} = 13\%)$: (a) generalisation error, (b) compression, (c) convergence time, and (d) MT parameter for `PSO(MT)`. Error bars are standard error of the sample mean.

MT polarity. Those large categories contribute to a lower resolution of the decision boundary, and thus a greater generalization error.

By training with `WMT`, the generalization error is significantly lower than both `MT-` and `MT+` especially with a large amount of training patterns, but the compression is the lowest of all training strategies. Based on the error alone, the effectiveness of the MT algorithm is debateable with overlapping data when compared with `MT-` and `MT+`, especially for application in which resource requirements are not an issue.

By training with `PSO(MT)`, fuzzy ARTMAP yields a significantly lower generalization error than all other strategies, and a compression that falls between that of `WMT` and `MT-` or `MT+`. With a training set of 5000 patterns per class, a compression of about 8.0 is obtained with `MT+`, 26.4 with `MT-`, 4.8 with `WMT`, and 5.3 with `PSO(MT)`. The convergence time is generally comparable with `WMT`, `MT-` and `MT+`. However, `PSO(MT)` requires a considerable number of training epochs to complete the optimization process. With a training set of 5000 patterns per class, a convergence time

of about 8.2 epochs is obtained with `MT+`, 3.6 with `MT-`, 12.3 with `WMT`, and 2534 with `PSO(MT)`.

Empirical results indicate that the MT process of fuzzy ARTMAP has a considerable impact on performance obtained with overlapping data, especially when ϵ is optimized. As shown in Figure 5(d), when $\alpha = 0.001$, $\beta = 1$ and $\overline{\rho} = 0$, and class distributions overlap, the values of ϵ that minimize error tends from about 0 towards 0.8 as the training set size grows. Higher ϵ settings tend to create a growing number of category hyperrectangles close to the bourdary between classes. Generalisation error of `PSO(MT)` tends toward that of to `WMT` on this data set. Furthermore, `PSO(MT)` and `WMT` do not show the performance degradation due to overtraining as with `MT+` and `MT-`.

Very similar tendencies are found in simulation results where fuzzy ARTMAP is trained using the other $D_\mu(\xi_{tot})$ and $D_\sigma(\xi_{tot})$ data sets. However, as ξ_{tot} increases,

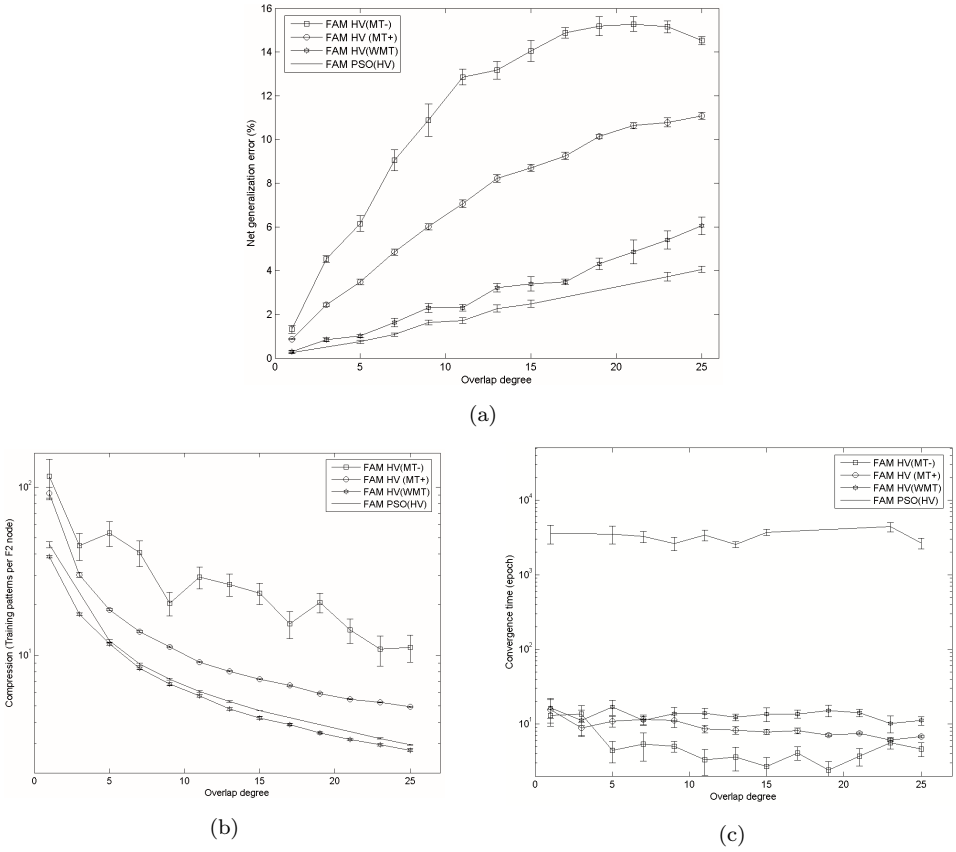


Fig. 6. Average performance of fuzzy ARTMAP (with `MT+`, `MT-`, `WMT` and `PSO(MT)`) as a function of ξ_{tot} for all $D_\mu(\xi_{tot})$ data sets: (a) net generalisation error, (b) compression, and (c) convergence time.

the performance degradation due to training subset size tends to become more pronounced, and occurs for fewer training set patterns. Let us define the *net error* as the difference between the generalization error obtained by using all the training data (5,000 patterns per class) and the theoretical probability of error ξ_{tot} of the database. Figure 6 shows the performance of fuzzy ARTMAP as a function of ξ_{tot} for all $D_\mu(\xi_{tot})$ data sets. As shown, using PSO(HV) always provides the lowest net error over ξ_{tot} values for overlapping data, followed by `WMT`, `MT+` and `MT-`. Again, `MT-` obtains the highest compression, whereas `PSO(MT)` obtain a compression between `WMT` and `MT+`. The convergence time of PSO(HV) is orders of magnitude longer that the other strategies.

Figure 7 presents an example of decision boundaries obtained for $D_\mu(\xi_{tot} = 13\%)$ when fuzzy ARTMAP is trained with 5,000 patterns per class and different MT strategies. For overlapping class distribution, `MT-` tends to create much fewer $F2$

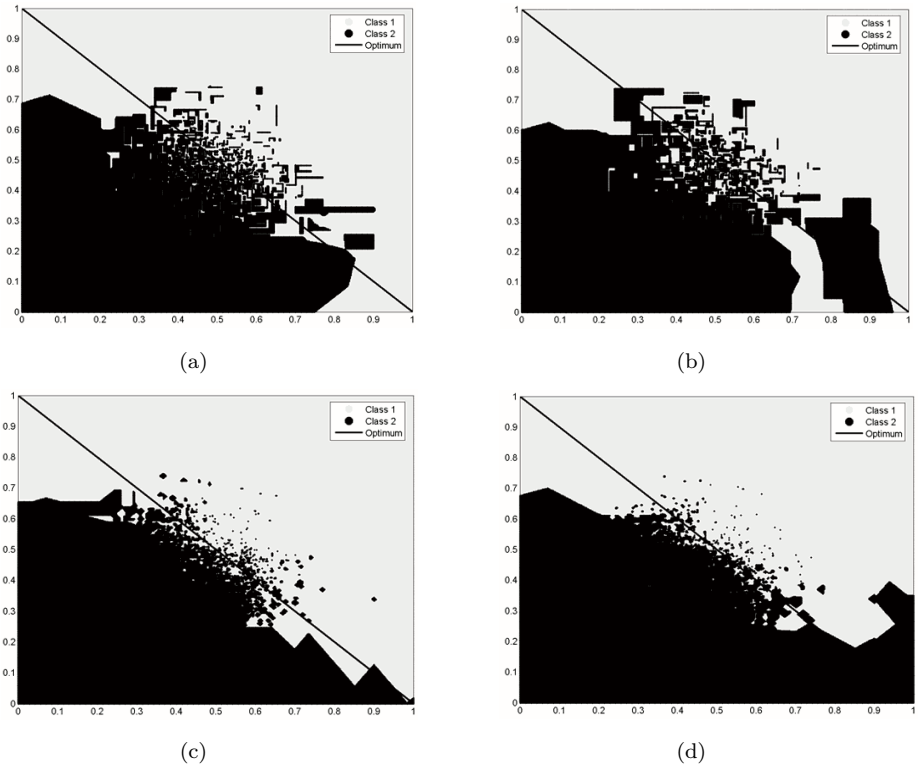


Fig. 7. An Example of decision boundaries formed by fuzzy ARTMAP in the input space for $D_\mu(\xi_{tot} = 13\%)$. Training is performed (a) with `MT+`, (b) with `MT-`, (c) `WMT`, and (d) `PSO(MT)` on 5,000 training patterns per class. The optimal decision boundary for $D_\mu(\xi_{tot} = 13\%)$ is also shown for reference. Note that virtually no training, validation or test subset patterns are located in the upper-left and lower-right corners of these figures.

nodes (908 categories with 5000 patterns per class) than the other MT strategies because of the polarity of ϵ. Although it leads to a higher compression, and can resolve inconsistent cases, the larger categories produce coarse granulation of the decision boundary, and thus a higher generalization error. With `PSO(MT)` and `WMT`, the lower error is a consequence of the finer resolution on overlap regions of the decision boundary between classes.

4.2. *Synthetic data with complex decision boundaries:*

Figure 8 presents the average performance obtained when fuzzy ARTMAP is trained on D_{CIS} using the four MT strategies – `MT-`, `MT+`, `WMT` and `PSO(MT)`. The generalisation error for the k-NN classifier, as well as the theoretical probability of error, ξ_{tot}, are also shown for reference.

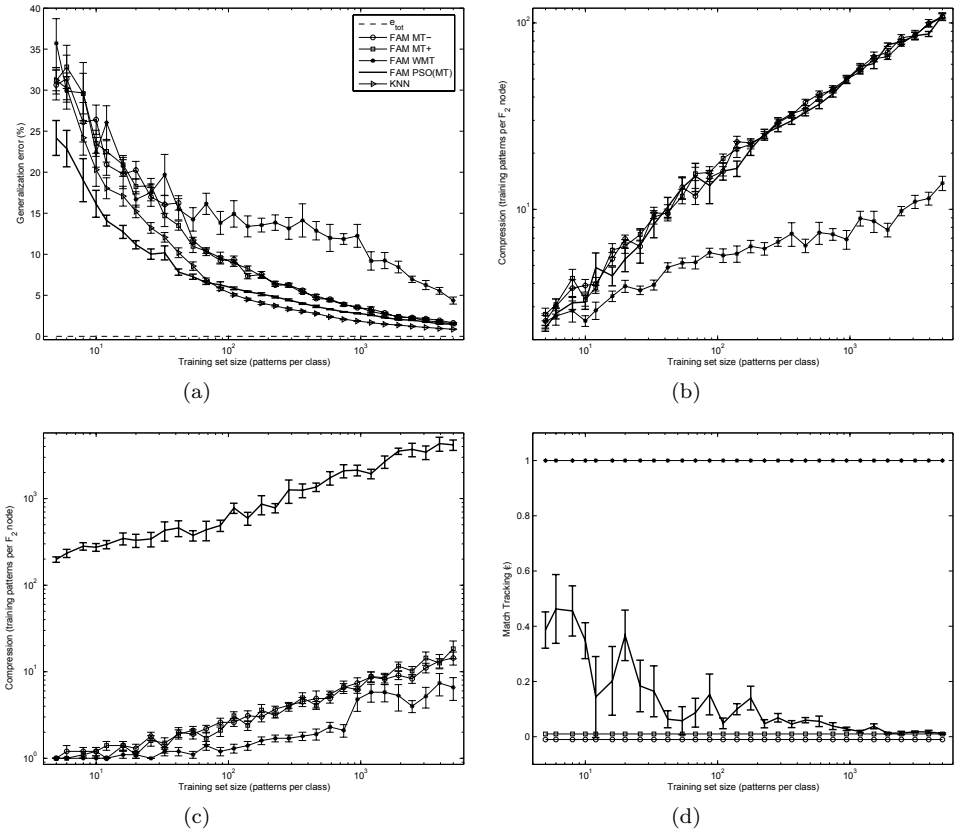


Fig. 8. Average performance of fuzzy ARTMAP (with `MT+`, `MT-`, `WMT` and `PSO(MT)`) versus training subset size for D_{CIS}: (a) generalisation error, (b) compression, (c) convergence time, and (d) MT parameter for `PSO(MT)`. Error bars are standard error of the sample mean.

In this case, `MT+`, `MT-` and `PSO(MT)` obtain a similar generalization error across training set sizes, while `WMT` yields an error that is significantly higher than the others strategies for larger training set sizes. For example, with a training set of 5000 patterns per class, a generalization error of about 1.51% is obtain with `MT+`, 1.64% with `MT-`, 4.36% with `WMT`, and 1.47% with `PSO(MT)`. Compression of fuzzy ARTMAP as a functions of training set size in a grows in a similar way for `MT-`, `MT+` and `PSO(MT)`. With a training set of 5000 patterns per class, a compression of 107 is obtained with `MT+`, 108 with `MT-`, 14 with `WMT`, and 109 with `PSO(MT)`. `WMT` does not allow to create a network with higher compression because data structure leads to the creation many small categories that overlap on the decision boundary between classes. However, `WMT` requires the fewest number of training epochs to converge, while `PSO(MT)` requires a considerable number of epochs. With a training set of 5000 patterns per class, a convergence time of about 18.4 epochs is required with `MT+`, 14.4 with `MT-`, 6.6 with `WMT`, and 4186 with `PSO(MT)`.

Empirical results indicate that the MT process of fuzzy ARTMAP also has a considerable impact on performance obtained on data with complex decision boundaries, especially when ϵ is optimized. As shown in Figure 8(d), when $\alpha = 0.001$, $\beta = 1$ and $\overline{\rho} = 0$, and decision boundaries are complex, the values of ϵ that minimize error tends from about 0.4 towards 0 as the training set size grows. Lower ϵ settings tend to create fewer category hyperrectangles close to the bourdary between classes. Generalisation error of `PSO(MT)` tends toward that of to `MT+` and `MT-` on this data.

Similar tendencies are found in simulation results where fuzzy ARTMAP is trained using the D_{P2} data set. However, since the decision boundaries are more complex with D_{P2}, a greater number of training patterns are required for fuzzy ARTMAP to asymptotically start reaching its minimum generalisation error. Moreover, all MT strategies tested on data with non linear decision boundaries generate no overtraining.[19]

Figure 9 presents an example of decision boundaries obtained for D_{CIS} when fuzzy ARTMAP is trained with 5,000 patterns per class and different MT strategies. For data with complex decision boundaries, training fuzzy ARTMAP `WMT` yields higher generalization error since it initially tends to create some large categories, and then compensates by creating many small categories. This leads to coarse granulation of the decision boundary, and thus a higher generalization error.

Table 1 shows the average generalisation error obtained with the reference classifiers and the fuzzy ARTMAP neural network using different MT strategies on $D_{\mu}(\xi_{tot})$, D_{CIS} and D_{P2}. Training was performed on 5,000 patterns per class. When using `PSO(MT)`, the generalisation error of fuzzy ARTMAP is always lower than when using `MT+`, `MT-` and `WMT`, but is always significantly higher than that of the Quadratic Bayes and k-NN classifiers. When data contains overlapping class distributions, the values of ϵ that minimize error tends towards $+1$. In contrast, when decision boundaries are complex, these ϵ values tend towards 0.

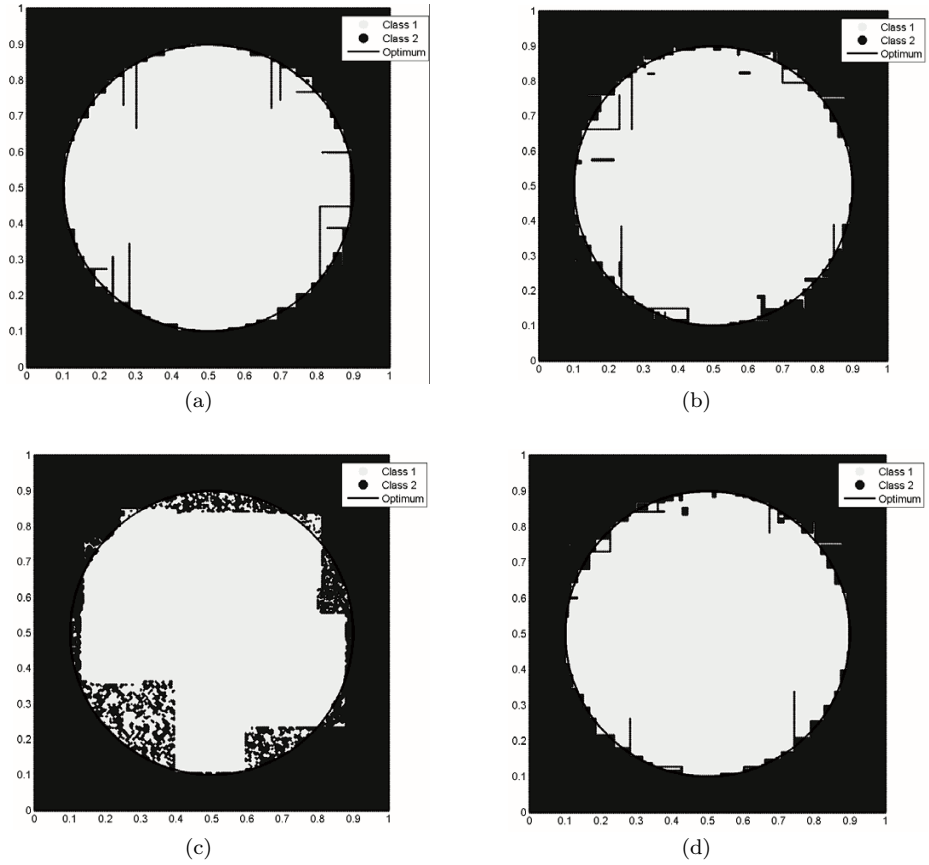


Fig. 9. An Example of decision boundaries formed by fuzzy ARTMAP in the input space for D_{CIS}. Training is performed (a) with `MT+`, (b) with `MT-`, (c) `WMT`, and (d) `PSO(MT)` on 5,000 training patterns per class. The optimal decision boundary for D_{CIS} is also shown for reference.

4.3. *NIST SD19 data:*

Figure 10 presents the average performance obtained when fuzzy ARTMAP is trained on the NIST SD19 data using the four MT strategies – `MT-`, `MT+`, `WMT` and `PSO(MT)`. The generalisation error for the k-NN classifier are also shown for reference.

As shown in this figure, `MT-` and `MT+` obtain similar average generalization error across training set sizes. Using a training set of 52760 patterns, a generalization error of about 5.81% is obtained with `MT+`, 6.02% with `MT-`, 32.84% with `WMT`, and 5.57% with `PSO(MT)`. When optimizing the MT parameter with `PSO(MT)`, generalization error is lower then other MT strategies with a small number of training pattern, and similar to `MT-` and `MT+` with greater number of training pattern. `WMT` is unable to create fuzzy ARTMAP network with low generalization error on NIST SD19.

Table 1. Average generalisation error of reference and fuzzy ARTMAP classifiers using different MT strategies on synthetic data sets. Values in parenthesis are standard error of the sample mean.

Data set	**Average generalisation error (%)**					
	CQB	***k*-NN**	**FAM w/ `MT+`**	**FAM w/ `MT-`**	**FAM w/ `WMT`**	**FAM w/ `PSO(MT)`** $\rightarrow \epsilon$
$D_\mu(1\%)$	1,00(0,04)	1,08(0,03)	1,87(0,04)	2,31(0,19)	1,30(0,03)	1,24(0,04) $\rightarrow$ 0,61(0,06)
$D_\mu(3\%)$	3,08(0,05)	3,31(0,06)	5,44(0,09)	7,52(0,16)	3,84(0,09)	3,66(0,06) $\rightarrow$ 0,75(0,05)
$D_\mu(5\%)$	4,87(0,07)	5,26(0,08)	8,48(0,13)	11,15(0,36)	6,01(0,07)	5,75(0,08) $\rightarrow$ 0,79(0,04)
$D_\mu(7\%)$	7,00(0,10)	7,48(0,11)	11,85(0,15)	16,05(0,47)	8,63(0,20)	8,07(0,08) $\rightarrow$ 0,73(0,04)
$D_\mu(9\%)$	9,12(0,08)	9,88(0,08)	15,01(0,14)	19,88(0,74)	11,30(0,21)	10,62(0,11) $\rightarrow$ 0,72(0,02)
$D_\mu(11\%)$	11,00(0,08)	11,81(0,13)	18,06(0,18)	23,85(0,37)	13,29(0,15)	12,72(0,12) $\rightarrow$ 0,77(0,05)
$D_\mu(13\%)$	13,16(0,15)	14,27(0,18)	21,22(0,17)	26,17(0,41)	16,22(0,19)	15,26(0,16) $\rightarrow$ 0,74(0,05)
$D_\mu(15\%)$	15,11(0,15)	16,13(0,13)	23,69(0,16)	29,05(0,48)	18,40(0,32)	17,42(0,15) $\rightarrow$ 0,74(0,04)
$D_\mu(17\%)$	16,96(0,10)	18,39(0,09)	26,25(0,16)	31,87(0,25)	20,49(0,13)	19,79(0,33) $\rightarrow$ 0,71(0,08)
$D_\mu(19\%)$	19,25(0,16)	20,71(0,16)	29,13(0,09)	34,19(0,44)	23,30(0,26)	22,24(0,11) $\rightarrow$ 0,79(0,05)
$D_\mu(21\%)$	20,97(0,13)	22,70(0,16)	31,63(0,14)	36,28(0,34)	25,86(0,54)	24,35(0,12) $\rightarrow$ 0,79(0,05)
$D_\mu(23\%)$	22,99(0,12)	25,04(0,13)	33,77(0,21)	38,15(0,28)	28,40(0,41)	26,72(0,19) $\rightarrow$ 0,71(0,03)
$D_\mu(25\%)$	25,11(0,10)	27,23(0,12)	36,08(0,14)	39,52(0,18)	31,05(0,40)	29,05(0,14) $\rightarrow$ 0,72(0,04)
D_{CIS}	N/A	0,86(0,03)	1,51(0,04)	1,64(0,04)	4,36(0,43)	1,47(0,04) $\rightarrow$ 0,01(0,00)
D_{P2}	N/A	1,65(0,04)	3,45(0,19)	4,33(0,22)	7,13 0,48)	3,44(0,06) $\rightarrow$ 0,01(0,00)

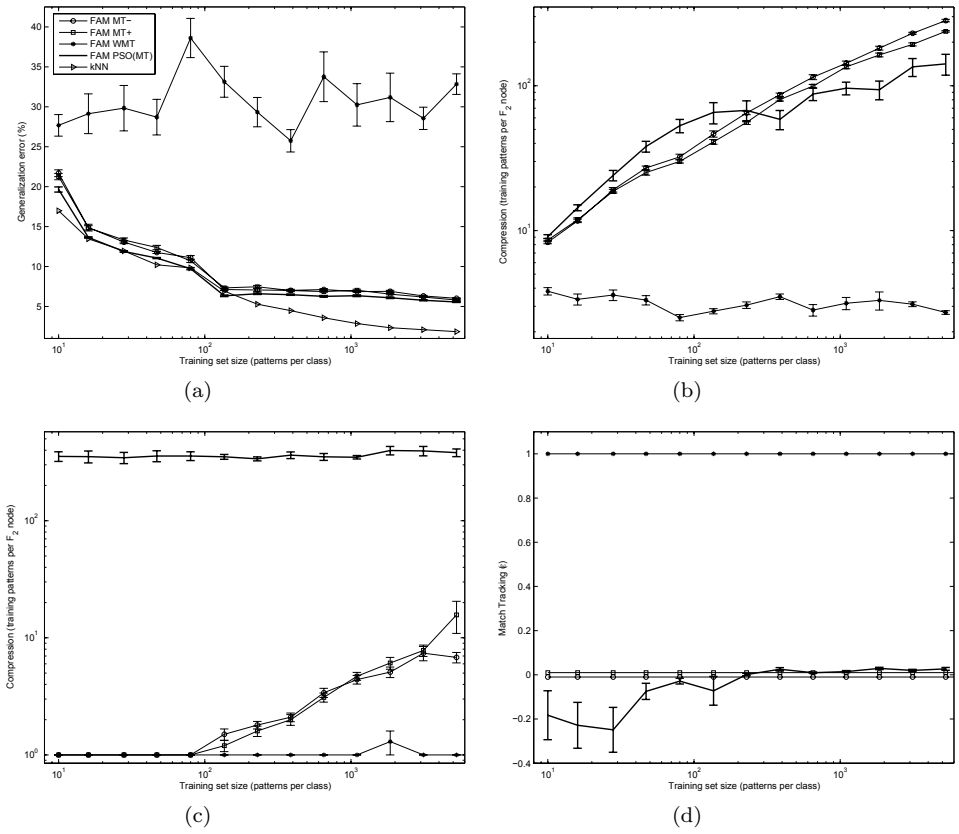


Fig. 10. Average performance of fuzzy ARTMAP (with `MT+`, `MT-`, `WMT` and `PSO(MT)`) versus training subset size for NIST SD19 data set: (a) generalisation error, (b) compression, (c) convergence time, and (d) MT parameter for `PSO(MT)`. Error bars are standard error of the sample mean.

Since NIST database possesses complex decision boundaries with a small degree of overlap, `WMT` cannot generate a good representation of the decision boundaries because it generates too many categories that overlap between classes.

Using all the training data, `MT-` acheives the highest compression, followed by `MT+`, `PSO(MT)` and `WMT`. However, with small amount of training patterns, `PSO(MT)` generates the highest compression. For example, with a training set of 52760 patterns, a compression rate of about 237.4 is obtained with `MT+`, 281.9 with `MT-`, 2.7 with `WMT`, and 141.6 with `PSO(MT)`. `WMT` obtains the lowest compression rate because if creates many very small categories to define the decision boundaries. With a training set of 52760 patterns, a convergence time of about 15.7 epochs is obtain with `MT+`, 6.8 with `MT-`, 1 with `WMT`, and 381 with `PSO(MT)`. `WMT` still possesses the fastest convergence time. The low generalization error of `PSO(MT)` requires a high convergence time (about 24.3 time higher than `MT+` with all training pattern).

As shown in Figure 10(d), when $\alpha = 0.001$, $\beta = 1$ and $\overline{\rho} = 0$, and decision boundaries are complex, the values of ϵ that minimize error tends from about -0.2 towards 0 as the training set size grows. As with D_{CIS} and D_{P2}, Generalisation error of `PSO(MT)` tends toward that of to `MT+` and `MT-` on this data set. Despite promising results training fuzzy ARTMAP with `PSO(MT)`, other pattern classifiers (such as SVM) have achieved significantly lower generalization error.[28,30]

5. Conclusions

A fuzzy ARTMAP neural network applied to complex real-world problems such as handwritten character recognition may achieve poor performance, and encounter a convergence problem, whenever the training set contains noisy and overlapping patterns that belong to different classes. In this chapter, a PSO-based strategy called `PSO(MT)` is used to optimize the MT parameter during training. The impact on fuzzy ARTMAP performance of adopting different MT strategies – the original positive MT (`MT+`), negative MT (`MT-`), without MT (`WMT`), and `PSO(MT)` – is assessed. An experimental protocol has been defined such that the generalization error and resource requirements of fuzzy ARTMAP trained with different MT strategies may be assessed on different types of synthetic problems and on the NIST SD19 handwritten character recognition data sets.

Overall, empirical results indicate that using the MT process for batch supervised learning has a significant impact on fuzzy ARTMAP performance. When data is defined by overlapping class distributions, training with `MT-` tends to produce fewer categories than the other MT strategies, although this advantage coincides with a higher generalization error. The need for `MT+` or `MT-` is debateable as `WMT` yields a significantly lower generalization error. However, `PSO(MT)` has been shown to create fuzzy ARTMAP networks with a finer resolution on decision bounds, and an even lower error than `WMT`. In addition, it has been shown to eliminate the degradation of error due to overtraining. To represent overlapping class distributions with `PSO(MT)`, the lowest errors are obtained for MT parameter values that tend toward the maximum value ($\epsilon = 1$) as the training set size grows. `PSO(MT)` thereby favors the creation of new internal categories to define decision boundaries.

When data is defined by complex decision boundaries, training with `PSO(MT)` creates the decision boundaries that yield the lowest generalization error, followed most closely by `MT-` and then `MT+`. Training with `WMT` yields a considerably higher generalization error and lower compression than the other MT strategies, specially when for larger training set sizes. To represent complex decision boundaries with `PSO(MT)`, the lowest errors are obtained for MT parameter values that tend toward 0 as the training set size grows.

Finally, with the NIST SD19 data set, when using all training pattern the generalization error obtain with `PSO(MT)` is about 0.84% lower than `MT-`, but comes at the expense of lower compression and a convergence time that can be two order

of magnitude greater than other strategies. Training with a Multi-Objective PSO (MOPSO) based strategy, where the cost function accounts for both generalization error and compression would provide solutions that require fewer internal categories. In addition light weight versions of PSO may reduce the convergence time.

In this chapter, training fuzzy ARTMAP with `PSO(MT)` has been shown to produce a significantly lower generalization error than with other MT strategies. These results are always produced at the expense of a significantly higher number of training epochs. Nonetheless, results obtained with `PSO(MT)` underline the importance of optimizing the MT parameter during training, for different problems. The MT parameter values found using this strategy vary significantly according to training set size and data set structure, and differ considerably from the popular choice ($\epsilon = 0+$), specially when data has overlapping class distributions.

Acknowledgements

This research was supported in part by the Natural Sciences and Engineering Research Council of Canada, and le Fonds québécois de la recherche sur la nature et les technologies.

References

1. Anagnostopoulos, G. C., Georgiopoulos, M., Verzi, S. J., and Heileman, G. L., "Boosted Ellipsoid ARTMAP," *Proc. SPIE – Applications and Science of Computational Intelligence V*, **4739**, 74-85, 2002.
2. Anagnostopoulos, G. C., and Georgiopoulos, "Putting the Utility of Match Tracking in Fuzzy ARTMAP Training to the Test," *Lecture Notes in Computer Science*, **2774**, 1-6, 2003.
3. Bote-Lorenzo, M. L., Dimitriadis, Y., Gómez-Sánchez, E., "Automatic extraction of human-recognizable shape and execution prototypes of handwritten characters," *Pattern Recognition*, **36:7**, 1605-1617, 2003.
4. Carpenter, G. A., and Grossberg, S., "A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine," *Computer, Vision, Graphics and Image Processing*, **37**, 54-115, 1987.
5. Carpenter, G. A., Grossberg, S., and Rosen, D. B., "Fuzzy ART: Fast Stable Learning and Categorisation of Analog Patterns by an Adaptive Resonance System," *Neural Networks*, **4:6**, 759-771, 1991.
6. Carpenter, G. A., Grossberg, S., and Reynolds, J. H., "ARTMAP: Supervised Real-Time Learning and Classification of Nonstationary Data by a Self-Organizing Neural Network," *Neural Networks*, **4**, 565-588, 1991.
7. Carpenter, G. A., Grossberg, S., Markuzon, N., Reynolds, J. H., and Rosen, D. B., "Fuzzy ARTMAP: A Neural Network Architecture for Incremental Supervised Learning of Analog Multidimensional Maps," *IEEE Trans. on Neural Nets*, **3:5**, 698-713, 1992.
8. Carpenter, G. A., and Ross, W. D, "ART-EMAP: A Neural Network Architecture for Object Recognition by Evidence Accumulation," *IEEE Trans. on Neural Networks*, 6:4, 805-818, 1995

9. Carpenter, G.A., Gjaja, M.N., Gopal, S., and Woodcock, C.E., "ART Neural Networks for Remote Sesing: Vegetation Classification from Landsat TM and Terrain Data," *IEEE Trans. on Geosciences and Remote Sensing*, **35:2**, 1997.
10. Carpenter, G. A., and Markuzon, N., "ARTMAP-IC and Medical Diagnosis: Instance Counting and Inconsistent Cases," *Neural Networks*, **11:2**, 323-336, 1998.
11. Carpenter, G. A., Milenova, B. L., and Noeskeand, B. W., "Distributed ARTMAP: a neural network for fast distributed supervised learning," *Neural Nets*, **11**, 793813, 1998.
12. Eberhart, R. C. and Shi, Y., "Comparison Between Genetic Algorihms and Particle Swarm Intelligence," in *Evolutionary Programming VII*, V. W. Porto et al, eds., Springer, 611-616, 1998
13. Gómez-Sánchez, E., Gago-Gonzalez, J. A., Dimitriadis, Y. A., Cano-Izquierdo, J. M., Lopez Coronado, J., "Experimental Study of a Novel Neuro-Fuzzy System for On-Line Handwritten UNIPEN Digit Recognition," *Pattern Recognition Let.*, **19**, 357-364, 1998.
14. Gómez-Sánchez, E., Dimitriadis, Y. A., Cano-Izquierdo, J. M., Lopez-Coronado, J., "μARTMAP: Use of Mutual Information for Category Reduction in Fuzzy ARTMAP," *IEEE Trans. on Neural Networks*, **13:1**, 58-69, 2002.
15. Granger, E., Rubin, M., Grossberg, S., and Lavoie, P., "A What-and-Where Fusion Neural Network for Recognition and Tracking of Multiple Radar Emitters," *Neural Networks*, **14**, 325-344, 2001.
16. Granger, E., Henniges, P., Sabourin, R., and Oliveira, L. S., "Supervised Learning of Fuzzy ARTMAP Neural Networks Through Particle Swarm Optimization," *Journal of Pattern Recognition Research*, **2:1**, 27-60, 2007.
17. Grother, P. J., "NIST Special Database 19 - Handprinted forms and characters database," *National Institute of Standards and Technology (NIST)*, 1995.
18. Grossberg, S., Rubin, M. A., and Streilein, W. W., "Buffered reset leads to improved compression in fuzzy ARTMAP classification of radar range profiles," *Intelligent Engineering Systems Through Artificial Neural Networks*, **6**, 419-424, 1996.
19. Henniges, P., Granger, E., and Sabourin, R., "Factors of Overtraining with Fuzzy ARTMAP Neural Networks," *International Joint Conference on Neural Networks 2005*, 1075-1080, Montreal, Canada, August 1-4, 2005.
20. Kennedy, J., and Eberhart, R. C., "Particle Swarm Intelligence," *Proc. Int'l Conference on Neural Network*, 1942-1948, 1995.
21. Kennedy, J., and Eberhart, R. C., *Swarm Intelligence*, Morgan Kaufmann, 2001.
22. Koufakou, A., Georgiopoulos, M., Anagnostopoulos, G., and Kasparis, T., "Cross-Validation in Fuzzy ARTMAP for Large Databases," *Neural Nets*, **14**, 1279-1291, 2001.
23. Lee, S.-J., and Tsai, H.-L., "Pattern Fusion in Feature Recognition Neural Networks for Handwritten Character Recognition", *IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics*, **28:4**, 612-617, 1998.
24. Lerner B., and Guterman, H., "Advanced Developments and Applications of the Fuzzy ARTMAP Neural Network in Pattern Classification," In *Advanced Computational Intelligence Techniques in Data Analysis and Applications*, Springer-Verlag, **SCI 137**. 77-107, 2008.
25. Lim C. P., and Harrison, R. F., "Modified Fuzzy ARTMAP Approaches for Bayes Optimal Classification Rates: An Empirical Demonstration," *Neural Network*, **10:4**, 755-774, 1997.
26. Liu, C.-L., Sako, H., and Fujisawa, H., "Performance Evaluation of Pattern Classifiers for Handwritten Character Recognition," *Int'l J. on Document Analysis and*

Recognition, **4**, 191-204, 2002.
27. Marriott, S., and Harrison, R. F., "A modified fuzzy ARTMAP architecture for the approximation of noisy mappings", *Neural Networks*, **8:4**, 619-41, 1995.
28. Milgram, J., Chriet, M. and Sabourin, R., "Estimating Accurate Multi-class Probabilities with Support Vector Machines," *International Joint Conference on Neural Networks 2005*, 1906-1911, Montral, Canada, August 1-4, 2005.
29. Murshed, N. A., Bortolozzi, F., and Sabourin, R., "A Cognitive Approach to Signature Verification," *International Journal of Pattern Recognition and Artificial Intelligence (Special issue on Bank Cheques Processing*, **11:7**, 801-825, 1997.
30. Oliveira, L. S., Sabourin, R., Bortolozzi, F., and Suen, C. Y., "Automatic Recognition of Handwritten Numerical Strings: A Recognition and Verification Strategy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24:11**, 1438-1454, 2002.
31. Parsons, O., and Carpenter, G. A., "ARTMAP neural network for information fusion and data mining: map production and target recognition methodologies," *Neural Networks*, **16**, 10751089, 2003.
32. Rubin, M.A., "Application of Fuzzy ARTMAP and ART-EMAP to Automatic Target Recognition Using Radar Range Profiles," *Neural Networks*, **8:7**, 1109-1116, 1995.
33. Schutte, J. F., Reinbolt, J. A., Fregly, B. J., Haftka, R. T., and George, A. D., "Parallel Global Optimization with Particle Swarm Algorithm," *International J. of Numerical Methods in Engineering*, **61**, 2296-2315, 2004.
34. Stone, M., "Cross-Validatory Choice and Assessment of Statistical Predictions," *Journal of the Royal Statistical Society*, 111-147, 1974.
35. Sumathi, S., Sivanandam, S. N., and Jagadeeswari, R., "Design of Soft Computing Models for Data Mining Applications," *Indian J. of Engineering and Materials Sciences*, **7:3**, 107-21, 2000.
36. Srinivasa, N., "Learning and Generalization of Noisy Mappings Using a Modified PROBART Neural Net," *IEEE Trans. on Signal Processing*, **45:10**, 2533-2550, 1997.
37. Taghi, M., Baghmisheh, V., and Pavesic, N., "A fast simplified Fuzzy ARTMAP network," *Neural Processing Letters*, **17**, 273316, 2003.
38. Valentini, G., "An Experimental Bias-Variance Analysis of SVM Ensembles Based on Resampling Techniques," *IEEE Trans. Systems, Man, and Cybernetics - Part B: Cybernetics*, **35:6**, 1252-1271, 2005.
39. Verzi, S. J., Heileman, G. L., Georgiopoulos, M., and Healy, M. J., "Boosting the Performance of ARTMAP", *IEEE International Joint Conference on Neural Networks Proceedings 1998*, Anchorage, USA, 396-401, 1998.
40. Waxman, A. M., Verly, J. G., Fay, D. A., Liu, F., Braun, M. I., Pugliese, B., Ross, W., Streilein, W., "A Prototype System for 3D Color Fusion and Mining of Multisensor/Spectral Imagery," *Proc. of the 4th International Conference on Information Fusion*, Vol. 1, pp. WeC1-(3-10), Montreal, Canada, August 7-10, 2001.
41. Williamson, J. R., "A Constructive, Incremental-Learning Neural Network for Mixture Modeling and Classification," *Neural Computation*, **9:7**, 1517-1543, 1997.