A What-and-Where Fusion Neural Network for Recognition and Tracking of Multiple Radar Emitters

Eric Granger[1,2]**, Mark A. Rubin**[3,4] **Stephen Grossberg**[3] **and Pierre Lavoie**[1]

[1] Defence Research Establishment Ottawa, Department of National Defence, 3701 Carling Avenue, Ottawa, Ontario, K1A 0Z4, Canada.

[2] Department of Electrical and Computer Engineering, École Polytechnique de Montréal, C. P. 6079, Station "centre-ville," Montreal, Quebec, H3C 3A7, Canada.

[3] Department of Cognitive and Neural Systems and Center for Adaptive Systems, Boston University, 677 Beacon Street, Boston, MA 02215, USA.

December, 2000

Technical Report CAS/CNS-TR-2000-029
Boston, MA: Boston University

Key Words: radar, electronic support measures, pattern recognition, data fusion, neural network, ARTMAP, Kalman filter.

Running Head: What-and-Where Fusion Neural Network.

[4]Current address: Sensor Exploitation Group, M.I.T. Lincoln Laboratory, 244 Wood Street, Lexington, MA 02420, USA

Abstract

A neural network recognition and tracking system is proposed for classification of radar pulses in autonomous Electronic Support Measure systems. Radar type information is combined with position-specific information from active emitters in a scene. Type-specific parameters of the input pulse stream are fed to a neural network classifier trained on samples of data collected in the field. Meanwhile, a clustering algorithm is used to separate pulses from different emitters according to position-specific parameters of the input pulse stream. Classifier responses corresponding to different emitters are separated into tracks, or trajectories, one per active emitter, allowing for more accurate identification of radar types based on multiple views of emitter data along each emitter trajectory. Such a What-and-Where fusion strategy is motivated by a similar subdivision of labor in the brain.

The fuzzy ARTMAP neural network is used to classify streams of pulses according to radar type using their functional parameters. Simulation results obtained with a radar pulse data set indicate that fuzzy ARTMAP compares favorably to several other approaches when performance is measured in terms of accuracy and computational complexity. Incorporation into fuzzy ARTMAP of negative match tracking (from ARTMAP-IC) facilitated convergence during training with this data set. Other modifications improved classification of data that include missing input pattern components and missing training classes. Fuzzy ARTMAP was combined with a bank of Kalman filters to group pulses transmitted from different emitters based on their position-specific parameters, and with a module to accumulate evidence from fuzzy ARTMAP responses corresponding to the track defined for each emitter. Simulation results demonstrate that the system provides a high level of performance on complex, incomplete and overlapping radar data.

1 Introduction

Radar Electronic Support Measures (ESM) involve the search for, interception, location, analysis and identification of radiated electromagnetic energy for military purposes. ESM hereby provide valuable information for real-time situation awareness, threat detection, threat avoidance, and for timely deployment of counter-measures (Browns, 1998; Davies and Hollands, 1982; Grant and Collins, 1982; Schleher, 1986; Schleher, 1999; Sciortino, 1997; Tsui, 1986; Wiley, 1993).

A critical function of radar ESM is the real-time identification of the radar type associated with each pulse train that is intercepted. Current approaches typically involve sorting incoming radar pulses into individual pulse trains, then comparing the pulse train characterizations with a library of parametric descriptions, which yields a list of likely radar types. This task is challenging owing to increases in environment density (*e.g.*, pulse Doppler radars that transmit hundreds of thousands of pulses per second); dynamically changing environments; multiplication and dispersion of the modes for military radars; agility in parameters like pulse repetition interval, radio frequency and scan; unknown and reserve modes for which no ESM library entry exists; overlaps between the parameters of different radar types in the ESM library; and noise and propagation effects that lead to erroneous or incomplete signal characterization. These aspects of the problem place severe stress on current ESM systems.

It this paper, an alternative approach is examined. A new recognition system combines diverse sources of information in order to predict the most likely radar type for each intercepted pulse. Type-specific parameters of the input pulse stream are used to classify pulses according to radar type, while environment-specific parameters are used to separate pulses corresponding to active emitters. Such separation allows the system to accumulate the classifier's responses for each emitter, and therefore to predict an emitter's identity based on one or multiple responses.

A key component of the new recognition system is a neural network classifier that is trained to determine the types of radar emitters present in the environment. The system learns autonomously, directly from data collected in the field, to identify pulse parametric ranges corresponding to specific radar types. Aside from avoiding some of the pulse sorting, training on data from the actual environment to approximate an unknown mapping function may deliver greater predictive accuracy. Furthermore, the need for by-hand construction of an emitter library is obviated.

From an ESM standpoint, training a system directly on radar data is a radical departure from current practice. At present, data are collected, analyzed, combined with prior information, and distilled into ESM libraries off-line by skilled analysts. New libraries, containing explicit radar type descriptions, are disseminated to the field as needed. One inconvenience of the conventional approach is that it is very complex, time-consuming, and does not allow for rapid modifications of ESM libraries upon discovery of new radar modes in the field. Using a neural network able to learn incrementally offers a framework for refining familiar, or adding unfamiliar, radar type descriptions on the fly.

In a particular realization of the recognition system, fuzzy ARTMAP (Carpenter et al., 1991a, 1992) is considered for neural network classification of pulses from their type-specific parameters, whereas nearest-neighbor matching with a bank of Kalman filters (Bar-Shalom and Li, 1993; Blackman, 1986) is considered for separation of pulses from their environment-specific parameters. The features of the system include: (1) By virtue of the fast-learning capabilities of ARTMAP neural networks (Carpenter et al., 1991a, 1991b, 1992), new information

from familiar or unfamiliar radar type classes can be learned incrementally without retraining on the whole data set. (2) Classification decisions can be made on the basis of single pulses or, for greater accuracy, on the basis of streams of pulses that have been determined to come from a given emitter. This determination is performed either by a time-of-arrival (TOA) deinterleaver or, when TOA deinterleaving is not practical, by a Kalman filter that tracks the bearing and amplitude of the pulses. The system is thus an example of a neural system combining temporal — When — and positional — Where — information with featural — What — information to arrive at its decision. It is well-known that the mammalian brain also divides What and Where computations into separate, but mutually interacting, cortical processing streams. Our What-and-Where model shows how this strategy can generate higher accuracy in identifying radar emitters. (3) The "familiarity discrimination" extension of fuzzy ARTMAP, called ARTMAP-FD (Carpenter et al., 1997a, 1997b), allows the system not only to detect pulses from unfamiliar radar type classes (not presented during training), but to determine the threshold for rejection based on all of the training data, without the need for holding back a portion for a validation set. This ability to determine the reject threshold on-line makes possible on-line learning of pulses from unfamiliar classes (LUC) (Granger et al., 2000). (4) New extensions to fuzzy ARTMAP permit both training and testing on data with missing components, and the use of unlabeled training data (Granger et al., 2000).

Conventional approaches to, and challenges of, radar type identification in radar ESM systems are reviewed in the next section. A system-level overview of our novel neural network recognition system is provided in Section 3. A radar pulse data set used for proof-of-concept simulations is presented in Section 4. The three main components that form a specific implementation of the recognition system are described in Sections 5 through 7. In Section 5, the fuzzy ARTMAP neural network is applied to the classification of pulses according to radar type from functional, type-specific parameters. Then, aspects of this network for dealing with incomplete radar data are proposed and tested. In Section 6, a module for clustering incoming pulses by emitter based on environment-specific parameters is described. In Section 7, a module that accumulates evidence from fuzzy ARTMAP responses corresponding to the tracked emitters is proposed. Finally, these three components are connected, and global simulation results using this particular realization of the entire recognition system are presented and discussed.

2 Radar Electronic Support Measures

2.1 Overview. The basic functionality of current radar ESM approaches can be decomposed into three tasks: reception of radar signals, grouping of pulses according to emitter, and identification of corresponding radar types.

Radar signals are passively intercepted by the receiver portion of the ESM system. In typical theaters of operation, intercepted signals are a mixture of electromagnetic pulses transmitted from, typically, several sources. Simultaneous illumination by these sources causes overlap and interleaving of the received pulses. Upon detection of a radar pulse, most receivers measure the pulse amplitude (PA), pulse width (PW), radio frequency of the carrier wave (RF) and time-of-arrival (TOA). Direction-finding receivers also measure the bearing (Brg), while advanced receivers also measure the modulation on pulse (MOP). Once parameter values have been measured for a pulse, they are digitized and assembled into a data structure called a Pulse Descriptor Word (PDW). For the reader's convenience, a list of the radar ESM abbreviations used in this

Table 1: List of abbreviations

Abbreviations	**Definition**
Brg	bearing
ESM	electronic support measures
EW	electronic warfare
MOP	modulation on pulse
PA	pulse amplitude
PDW	pulse descriptor word
PPI	pulse-to-pulse interval
PRI	pulse repetition interval
PW	pulse width
RF	radio frequency
TOA	time of arrival

paper is given in Table 1.

The stream of successive PDWs is fed to a grouping module, which performs either TOA deinterleaving, or sorting, or both. In short, this module seeks to recover pulse trains and their inter-pulse structure prior to further analysis. This involves progressively grouping pulses that appear to have been transmitted from the same emitter. An emitter is an instance of a radar type, and it is not uncommon to observe several emitters of a same type all being active in a theater of operation. A single type of radar can also operate under several different modes to perform various functions. To each group of pulses is associated a *track*. A track consists of statistical PDW parameters, plus other parameters that are derived from the sequence of grouped PDWs, like the pulse repetition interval (PRI).

Pulse grouping techniques either exploit the difference in TOA between pulses, or the actual parameters in the PDWs. Parametric ranges are associated with tracks, and updated to reflect changes in the emitter's characteristics over time. TOA deinterleaving attempts to discover consistent patterns in the TOA of pulses using techniques such as TOA difference histogramming (Davies and Hollands, 1982; Mardia, 1989; Wiley, 1993). If TOA consistencies are found, and these correlate with radar definitions compiled in an ESM library, then the corresponding pulses are grouped based on PRI, and stripped away from the input stream of PDWs. Sorting attempts to group pulses based on the likeliness of their PDW parameters such as RF, PW and Brg. Gating (Chandra et al., 1988; Davies and Hollands, 1982; Rogers, 1985) or clustering (Anderberg, 1973; Dubes and Jain, 1988; Wilkinson and Watson, 1985) techniques are commonly used to this end.

Identification makes use of an ESM library where are stored the parametric descriptions of known radar types, and attempts to assign a single radar type to each track. Incidentally, the parametric ranges of various types can overlap in the library, and multiple candidates can appear plausible for the same track, a situation known as an "ambiguity." Therefore, a list of likely radar types is often displayed and monitored over time for every track, along with a confidence rating, threat level, latest bearings, and so on. Further analysis can assist an ESM operator in revealing

mode changes in emitters, links between emitters, and inferred platforms.

2.2 Challenges. Pulse grouping and radar type recognition keep evolving in response to the following defense trends: radar signals are more agile; power management and low probability of intercept waveforms in advanced threats reduce response time; ESM libraries are expensive to maintain; and unmanned platforms require autonomous ESM. These trends motivate this work, and call for more powerful ESM approaches.

The multiplication of radar modes is the result of computer control and the ease with which parameters such as RF and PRI can be changed. From an ESM standpoint, this means libraries that grow larger and more complex. Agility in parameters like RF and PRI can make pulse grouping very difficult.

A shorter response time requires faster pulse grouping, as well as identification using fewer pulses. In addition, the occurrence of low power waveforms implies that pulses near the receiver detection threshold may be dropped, and hence that pulse grouping must work satisfactorily on sparse data. Response time is critical if threats are to be avoided, or self-protection measures such as chaff dispensing, maneuvering, or electronic jamming, are to be successful.

It is difficult and expensive to maintain comprehensive ESM libraries that accurately reflect each specific operational environment. Library construction requires explicit modeling of known radar systems, based on prior information and data that is not necessarily extracted from the local environment. This task is complex, tedious, and prone to error because some radar types are difficult to describe. Owing to the multiplication of modes, it is not uncommon for a library to be incomplete and to contain erroneous data. In addition, threats could deliberately reserve some of their modes for use during war time. Radar type identification must therefore be tolerant to such shortcomings. For instance, classical parametric approaches to pattern classification (Duda and Hart, 1973; Fukunaga, 1990) are generally less effective when the underlying radar type class distributions are incomplete and/or uncertain.

Personnel reduction in the Armed Forces, as well as the deployment of ESM on autonomous platforms such as unmanned aerial vehicles raises the expectations for ESM. Without an operator to interpret ESM output and provide discernment, ESM systems must achieve enhanced accuracy and reliability. In light of these trends, alternative approaches are sought for pulse grouping and radar type identification.

3 A Neural Network for Radar Type Identification

3.1 Adaptive Learning and ESM. In this section, a new approach is described for radar type recognition. When collection platforms are brought into a theater of operations prior to military interventions, data from radars of interest can be collected and analyzed. Collection platforms include in-theater tactical aircraft and ships, unmanned aerial vehicles, and stand-off assets like electronic warfare (EW) aircraft. Data collected prior to, or during, the conflict and analyzed either on-line (*e.g.,* on a ship) or off-line from electronic intelligence readings. Whereas the data are normally combined with prior information from other environments, and distilled by analysts into library entries, this paper explores their use for training an artificial neural network recognition system. Once trained on the data, the network can classify the pulses without the grouping process, thereby making use of a priori information early in the processing chain. As discussed in the following, this approach offers several potential advantages.

Firstly, training on real data gathered in the field may yield higher classification accuracy. Adaptive learning algorithms used to train neural network classifiers constitute an interesting alternative to the explicit modeling currently employed in ESM libraries, since they can estimate unknown input-to-class mapping functions directly from the training set. Their supervised learning process involves a prescription to combine some prior assumptions (*i.e.*, a set of possible mapping functions) with the training data set, to approximate an unknown mapping function. This mapping is then used to generalize, that is, predict output classes (radar types) for unlabeled input patterns (pulses).

Secondly, an attractive feature of neural networks is the convenience of handling incomplete radar type descriptions and incomplete data. Since it is impossible to have an exhaustive data set to train a network, recognition of new radar types encountered during operations is important. Neural network classifiers that allow on-line incremental learning provide a consistent framework for automatically refining the description of familiar radar types, as well as for detecting unfamiliar radar types and learning their description as operations unfold.

Lastly, although it is not the focus of this paper, the massively parallel architecture of neural networks, when implemented on appropriate hardware, can provide extremely fast and fault tolerant processing of PDWs. Such a response would also be somewhat tolerant to incomplete and noisy data, yielding graceful performance degradation. The on-line nature of the networks also eliminates the need to collect batches of pulses prior to processing, as in current approaches.

Neural network techniques have previously been applied to several aspects of radar ESM processing (Sciortino, 1997; Self and Bourassa, 1991), including parameter measurement (Self and Bourassa, 1991), PDW sorting (Anderson et al., 1990; Kamgar-Parsi et al., 1996; Meiler and van Wezenbeek, 1990; Pape et al., 1997; Wang and Thompson, 1991), and radar type recognition (Macedo Filho and Griffiths, 1994; Maloney and Specht, 1989; Roe and Roe, 1994; Specht, 1989). A new neural network recognition and tracking system for classification of radar pulses is described next.

3.2 Overview of ESM Model. One possible embodiment of a neural network recognition system into an ESM system is depicted in Figure 1. First a TOA deinterleaver uncovers periodicities in the TOA of input PDWs. Whenever grouping pulses is straightforward, it forms tracks and assigns a track number and a PRI to each grouped pulse. TOA deinterleaving continues to play an important role in ESM since it suffices to group the pulses of emitters having simple PRI patterns, like many high duty cycle Pulse Doppler radars. Besides, the PRI parameters themselves are useful for classification of pulses according to radar type.

The neural network recognition system receives all the PDWs, some of which have track numbers and PRI parameters. It has already been trained off-line on data from known radar types. The neural network weights replace the ESM library, and can be periodically updated by learning from radar data collected during operations. The neural network outputs a prediction of the radar type for every PDW, and assigns a track number to the PDWs that did not get one from the TOA deinterleaver. Track assignment is autonomous, and takes place regardless of the radar type classification. The radar type classification does, however, take into account track assignment.

The remaining module in Figure 1 is a signal separator, which receives all the PDWs along with their track numbers, radar types, and, whenever available, PRI parameters. The signal separator is responsible for the final track assignment and for distilling the stream of PDWs into emitter reports that are periodically updated. The final assignment takes into account the

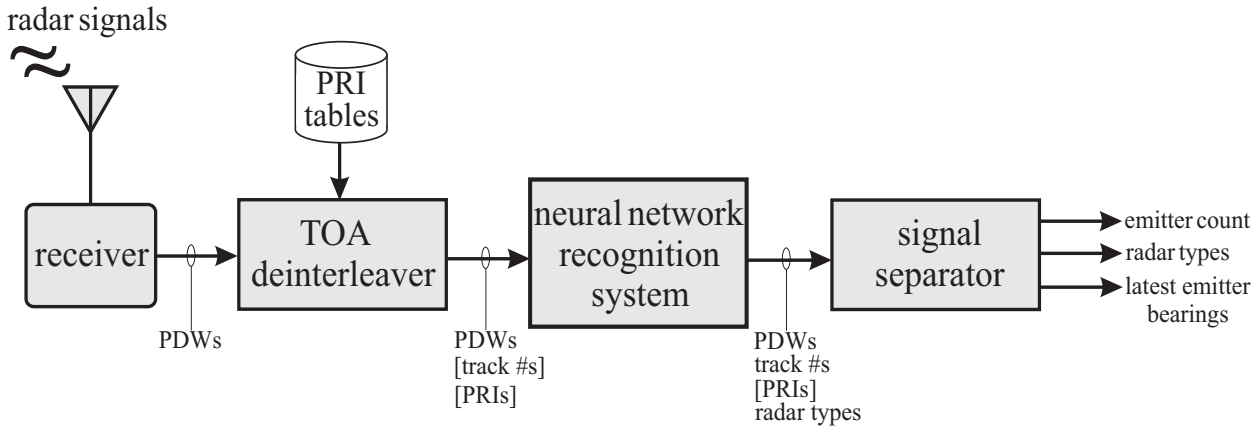


Figure 1: High level block diagram of a radar ESM system that uses a neural network recognition system. Brackets indicate that the corresponding field may be empty for some pulses.

radar type recognition previously performed by the neural network. Emitter reports contain, for instance, the type of each emitter with its latest bearing.

3.3 What-and-Where Model Architecture.

3.3.1 What and Where data streams. The PDW stream may be partitioned into two data streams called *What* and *Where*. This division is motivated by a similar subdivision of parallel processing in the primate cerebral cortex into a What stream for recognizing objects, and a Where stream for localizing their position in space. See Grossberg (2000) for a theoretical discussion of these processing streams. Here the What data stream consists of parameters that characterize the functional aspects of radar systems. Such parameters include RF, PW and PRI. Since these parameters correspond to data typically compiled in ESM libraries, they are directly useful for radar type recognition. The Where data stream consists of context-specific parameters. This stream is defined by parameters, such as Brg and PA, that indicate the status (*e.g.*, position) of specific emitters in the environment. These parameters are less useful than What parameters for radar type recognition, but are important for grouping pulses into tracks, or trajectories. The definition of Where parameters can be extended to include emitter specific parameters that cannot be recorded *a priori* due to practical or physical considerations. Such parameters may prove effective for pulse grouping, irrespective of their value for radar type recognition. Some MOP parameters could, for example, be assigned to this processing stream.

3.3.2 Distinct What and Where data processing. The internal architecture of a neural network recognition system is shown in Figure 2. It is composed of three subsystems: neural network classification, clustering and evidence accumulation. These subsystems cooperate to predict the most likely radar type for each incoming pulse.

Prior to on-line operation, the neural network classification module is trained, via supervised learning, using a data set of radar pulses collected in the field, and labeled with their respective radar type. Only What parameters are employed for training. PRI may be supplied if it is available.

During on-line operation, the recognition system accepts the stream of PDWs corresponding to intercepted radar pulses, as well as the track number and PRI of each pulse grouped by the

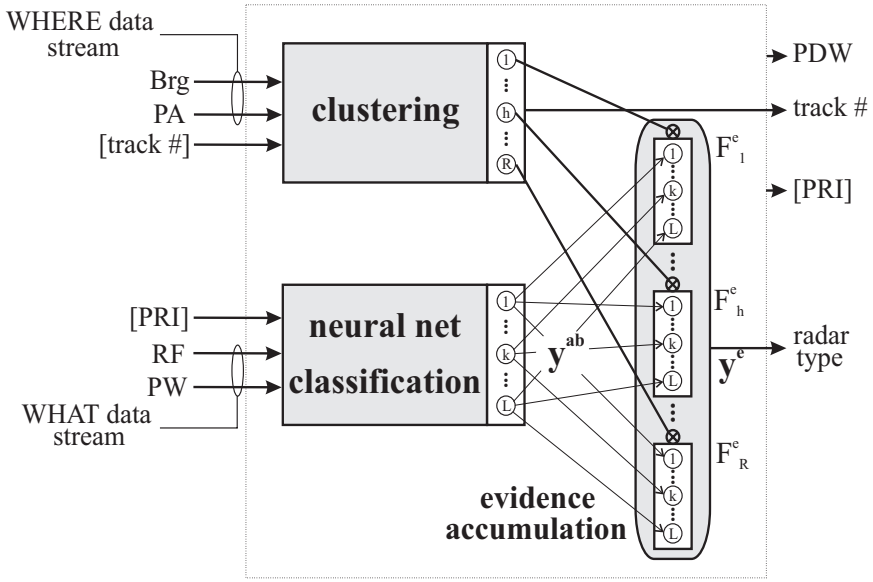


Figure 2: Internal architecture of the neural network recognition system.

TOA deinterleaver. Since each PDW is composed of predefined What and Where parameters, these can be automatically separated and fed to the neural network classification and clustering subsystems, respectively.

For each pulse, the neural network classification subsystem accepts What parameters, including PRI when available, and yields a prediction of the radar type. This prediction takes the form of a response pattern denoted by $\mathbf{y}^{ab}$ (refer to Figure 2). Meanwhile, the clustering subsystem attempts to group pulses into tracks based on Where parameters. If a track number is supplied by the TOA deinterleaver, then the output of this subsystem is bypassed. Whether the subsystem produces the track number itself, or receives it from the TOA deinterleaver, it maintains an up-to-date picture of the number and activity of radar emitters illuminating the ESM system.

3.3.3 Evidence accumulation. In conventional radar ESM systems, Where information is employed early in the processing chain (*i.e.*, during track formation) to reduce data and subsequent computational costs. The neural network recognition system embodies an alternative approach that integrates both What and Where information streams for better recognition prior to the data reduction. Fusion of responses from the classification subsystem and the clustering subsystem is accomplished via evidence accumulation, which emulates the brain process of working memory; *e.g.*, Bradski et al. (1994), and Bradski and Grossberg (1995). Response patterns $\mathbf{y}^{ab}$ obtained from classification are hereby accumulated over time according to tracks, that is, groupings determined from Where data.

Track numbers obtained from the clustering module dictate the emitters to which PDWs are associated, and drive the evidence accumulation. This evidence accumulation is implemented as a set of evidence accumulation fields, with each field F_h^e corresponding to a track $h = 1, 2, \ldots, R$. Assignment of a track $h = H$ to a PDW activates an evidence accumulation field F_H^e that accumulates the classification module's response pattern $\mathbf{y}^{ab}$. Such accumulation produces a radar type response pattern for each PDW, denoted by $\mathbf{y}^e$ (see Figure 2), which is obtained from one or more responses $\mathbf{y}^{ab}$. As discussed in Section 7, exploiting both What and Where information

sources can thereby enhance the system's classification accuracy.

4 Radar Pulse Data

The data set used for the computer simulation contains approximately 100,000 consecutive radar pulses gathered over 16 seconds by the Defense Research Establishment Ottawa during a field trial. After the trial, an ESM analyst manually separated trains of pulses coming from different emitters. Each pulse was then tagged with two labels: a radar type number and a mode number. Since ESM trials are complex and never totally controlled, not all pulses could be tagged and a sizable residue was obtained. Residue pulses were discarded for this study.

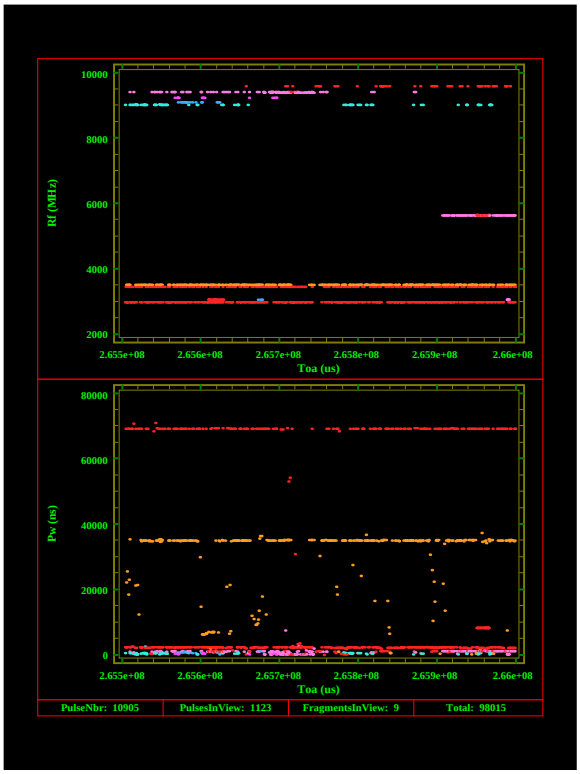
The parameters used are Brg, PA, PRI, PW and RF (refer to Table 1). From this point on, a PDW is denoted by $(\mathbf{a};\mathbf{b})$. Patterns in the What and Where data streams are defined by $\mathbf{a} =$ (PRI, PW, RF) and $\mathbf{b} =$ (Brg, PA), respectively. The two Where parameters, Brg and PA, are specific to the environment, and thus are not employed for training the neural network classification module.

Brg, PA, PW and RF are automatically produced by the receiver on each individual pulse, whereas PRI is derived from the difference in time-of-arrival (TOA) between pulses from the same emitter. For simplicity, it is assumed that, as a part of the preprocessing, a simple TOA deinterleaver has grouped the pulses belonging to each active emitter mode, and then computed their respective PRI values. Note that, since at least two successive pulses are required to compute a PRI value, the first pattern from each active emitter mode was omitted from the simulations. Also, owing to the circular scanning action of some radar emitters, pulses are recorded in bursts. The first pulse of each scan (or burst) was also omitted. Finally, the parameters were linearly normalized so that $a_i, b_j \in [0,1]$, for $i = 1,2,3$ and $j = 1,2$.

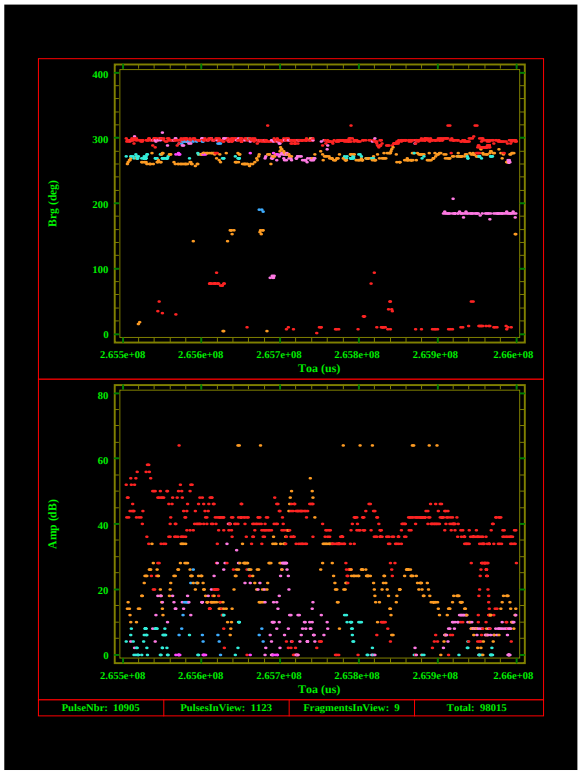
Once tagged and deinterleaved, the data used to train and test the neural network recognition system contain 52,192 radar pulses from 34 modes, each one belonging to one of 15 different radar types. The data feature bursts of high pulse densities, multiple emitters of the same type, modes with overlapping parametric ranges, radars transmitting at different pulse rates, and emitters switching modes. The sophistication of the radar types range from simple (constant RF and PRI) to fairly complex (pulse-to-pulse agility in RF and PRI). Figures **??** and **??** displays a 0.5 second sample of the radar pulse data set used for simulations. This particular example contains 1123 pulses from 8 emitters belonging to 7 different radar types, with agility and overlap of parameters, and an emitter switching modes.

5 An ARTMAP Neural Network for Classification

An enhanced ARTMAP neural network is used to classify incoming radar pulses according to radar type from parameters in the What data stream. ARTMAP refers to a family of neural network architectures capable of fast, stable, on-line, unsupervised or supervised, incremental learning, classification, and prediction (Carpenter et al., 1991b, 1992). ARTMAP networks have several attractive features for applications such as electronic support measures (ESM). Because they can perform fast, stable, on-line, incremental learning, they can learn from novel events encountered in the field. Neural network classifiers such as the popular Multilayer Perceptron (MLP) (Rumelhart et al., 1986) and Radial Basis Function (RBF) (Chen et al., 1991) require off-line retraining on the whole data set, through a lengthy iterative slow-learning procedure,



(a) What parameters (RF and PW) vs TOA.



(b) Where parameters (Brg and PA) vs TOA.

Figure 3: A sample of the radar pulse data set used for simulations.

to learn new patterns from existing radar type classes or a new radar type class. ARTMAP networks can also perform familiarity discrimination to avoid meaningless guesses on patterns from unfamiliar radar types classes (Carpenter et al., 1997a, 1997b; Granger et al., 1999a). Furthermore, they can represent radar type classes using one or more prototypes, which appears desirable for handling radar types having several modes of operation. The *k*-Nearest-Neighbor (*k*NN) (Cover and Hart, 1967) and Probabilistic Neural Network (PNN) (Specht, 1990) classifiers would usually require greater computational resources to store all the training set patterns, and to yield on-line predictions. Finally, ARTMAP networks lend themselves well to high speed parallel processing, which is critical for real-time identification.

5.1 Fuzzy ARTMAP. ARTMAP is often applied using the simplified version shown in Figure 4. It is obtained by combining an ART unsupervised neural network (Carpenter and Grossberg, 1987) with a map field. Fuzzy ARTMAP (Carpenter et al., 1992) can process both analog and binary-valued input patterns by employing fuzzy ART (Carpenter et al., 1991a) 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 is connected, through learned associative links, to an L node map field F^{ab}, where

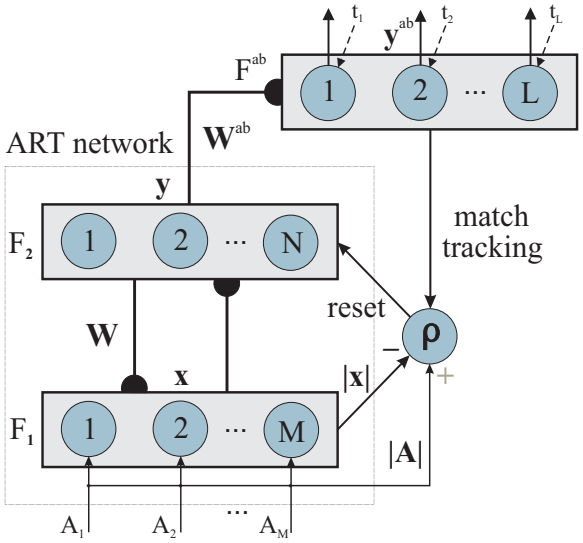


Figure 4: An ARTMAP neural network architecture specialized for pattern classification.

L is the number of classes in the output space. A set of binary weights $\mathbf{W}^{ab} = \{w_{jk}^{ab} \in \{0,1\} : j = 1,2,...,N; k = 1,2,...,L\}$ is associated with the F_2-to-F^{ab} connections. The vector $\mathbf{w}_j^{ab} = (w_{j1}^{ab}, w_{j2}^{ab}, ..., w_{jL}^{ab})$ links F_2 node j to one of the L output classes.

During training, ARTMAP classifiers perform supervised learning of the mapping between training set vectors $\mathbf{a} = (a_1, a_2, ..., a_m)$ and output labels $\mathbf{t} = (t_1, t_2, ..., t_L)$, where $t_K = 1$ if K is the target class label for $\mathbf{a}$, and zero elsewhere. The following algorithm describes fuzzy ARTMAP learning:

1. Initialization. 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 $\alpha > 0$, and the baseline vigilance $\bar{\rho} \in [0,1]$ 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 norm operator, $|\mathbf{w}_j| \equiv \sum_{i=1}^{M} |w_{ij}|$, $\wedge$ 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 ρ:

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

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. The depth of search before an uncommitted node is selected is determined by the choice parameter α.

4. Class prediction. Pattern **t** is fed directly to the map field F^{ab}, while the F_2 category **y** learns to activate the map field via associative weights $\mathbf{W}^{ab}$. The F^{ab} layer produces a binary pattern of activity $\mathbf{y^{ab}} = (y_1^{ab}, y_2^{ab}, \ldots, y_L^{ab})$ in which the most active F^{ab} node K yields the class prediction ($K = k(J)$). If node K constitutes an incorrect class prediction, then a *match tracking* signal raises the vigilance parameter ρ just enough to induce another search among F_2 nodes in Step 3. This search continues until either an uncommitted F_2 node becomes active (and learning directly ensues in Step 5), or a node J that has previously learned the correct class prediction K becomes active.

5. Learning. Learning input **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:

$$\mathbf{w}_J' = \beta(\mathbf{A} \wedge \mathbf{w}_J) + (1 - \beta)\mathbf{w}_J \ , \tag{3}$$

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 ART represents category j as an m-dimensional hyperrectangle R_j that is just large enough to enclose the cluster of training set patterns **a** to which it has been assigned. 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 **a**, and 0 otherwise. Once the weights **W** have converged for the training set patterns, 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 **a** that activates node J is predicted to belong to class $K = k(J)$.

5.2 Comparative Simulations.

Fuzzy ARTMAP and three other ARTMAP neural networks — ART-EMAP (Stage 1) (Carpenter and Ross, 1995), ARTMAP-IC (Carpenter and Markuzon, 1998) and Gaussian ARTMAP (Williamson, 1996, 1997) – have been compared using computer simulations. The *k*NN and RBF classifiers were included for non-parametric, and semi-parametric (Bishop, 1995) reference, respectively.

Prior to each simulation trial, the radar pulse data described in Section 4 was partitioned into training and test subsets. 50% of the data from each radar type was selected at random to form the training subset. Then, the training vectors **a**, along with their radar types labels **t**, were repeatedly presented, until convergence. The same random order was used across presentations. Emitter mode labels were ignored since this paper concerns the classification of pulses according to radar type. Convergence was reached when the sum-squared-fractional-change (SSFC) of prototype weights **W** was less than 0.001 for two successive epochs. An epoch is defined as a presentation of the training subset to a classifier in TOA order. The RBF classifier used in this comparison selects training subset patterns one by one to encode hidden layer nodes (Chen et al., 1991). Convergence was reached when the sum-squared-error between actual outputs

(resulting from training set patterns) and target outputs fell below 0.01. After convergence, the test subset was presented to the trained classifier for prediction. Throughout this paper, average results are obtained from several independent simulation trials, each one with a different random selection of the training data.

The What patterns consist of 3 parameters: $\mathbf{a} =$ (PW, PPI, RF). It is assumed that a TOA deinterleaver has correctly grouped the n_k pulses belonging to each active emitter mode k, and then computed the pulse-to-pulse intervals: $\mathrm{PPI}_k(i) = \mathrm{TOA}_k(i) - \mathrm{TOA}_k(i-1)$ for $i = 2, 3, \ldots, n_k$. Using the PPI to estimate the PRI allows for simple training and testing of neural network classifiers without concern for the PRI agility of some emitters.

The performance of each classifier was assessed in terms of both the amount of resources required and predictive accuracy. The amount of resources allocated during training is measured in the 3 following ways. *Compression* refers to the average ratio of training patterns to committed F_2 layer nodes. *Memory* is the number of normalized registers needed to store the set of learned prototype vectors, a normalized register being a fixed-size register whose number of bits suffices to store the classifier's real values such as a_i, w_{ji}, ρ, and so on. *Convergence time* is the number of epochs required for the classifier to converge. The predictive accuracy on the test subset is measured using the *classification rate* – the ratio of correctly classified patterns over all test patterns.

Average results from 20 simulation trials of fuzzy ARTMAP are given in Table 2, along with the standard error of the sample mean (in parentheses). Parameter settings were selected through trial and error to achieve the best classification rate for the least memory and convergence time during training. Results indicate that fuzzy ARTMAP and Gaussian ARTMAP consistently achieve the highest average classification rates, followed by ARTMAP-IC and ART-EMAP (Stage 1). The classification rates of fuzzy ARTMAP and Gaussian ARTMAP are comparable to those obtained using the *k*NN and RBF classifiers. ART-EMAP, ARTMAP-IC and fuzzy ARTMAP attain their classification rates with greater compression (and thus require less physical memory to store prototype vectors, and deliver faster fielded performance) than the other classifiers, and take fewer training epochs to converge than Gaussian ARTMAP and RBF. Overall, fuzzy ARTMAP performs at least as well as the other classifiers in both accuracy and computational complexity, and better than each of them in at least one of these aspects of performance. The reader is referred to Appendix A and to Granger et al. (1999b) for further details of these simulations and the classifiers used.

5.3 Convergence and Negative Match Tracking. A convergence problem occurs with the above fuzzy ARTMAP algorithm whenever the training subset contains identical patterns that belong to different classes. In the present application, this corresponds to radar pulses in a same resolution cell that belong to different radar types. The problem is aggravated because ARTMAP tends to segment the overlapping parts of classes into several tiny, often minimum-sized prototypes. The consequence is a proliferation of identical prototypes for certain training set patterns.

Table 2: Average classification results for the radar data. The "†" indicates that the classifier was unable to converge for the training set on each trial. (Numbers in parentheses are the standard error of the sample mean.)

Classifier	**Evaluation criteria (std error)**			
	Accuracy	**Resources**		
	Classification rate	**Compression**	**Memory**	**Convergence time**
***k*-Nearest-Neighbor** ($k = 1$, $d_{\text{cityblock}}$)	99.64% (0.01%)	1.0 (0.0)	80311 (0)	N/A
***k*-Nearest-Neighbor** ($k = 1$, $d_{\text{Euclidean}}$)	99.64% (0.01%)	1.0 (0.0)	80311 (0)	N/A
Radial Basis Function† (spread of RB kernel = 0.05)	99.68% (0.01%)	4.1 (0.1)	6367.9 (3.3)	2123.6 (1.1)
ART-EMAP† (Stage 1) ($\varepsilon = 10^{-4}$, $\alpha = .001$, $\beta = 1$, $\bar{\rho} = 0$)	78.28% (1.65%)	222.9 (5.2)	709.2 (15.9)	stopped at 3.6 (0.1)
ARTMAP-IC ($\varepsilon = 10^{-4}$, $\alpha = .001$, $\beta = 1$, $\bar{\rho} = 0$)	83.65% (1.90%)	214.0 (4.6)	737.4 (14.5)	3.8 (0.1)
fuzzy ARTMAP† ($\varepsilon = 10^{-4}$, $\alpha = .001$, $\beta = 1$, $\bar{\rho} = 0$)	99.56% (0.02%)	217.5 (5.9)	729.6 (19.2)	stopped at 3.6 (0.1)
Gaussian ARTMAP† ($\varepsilon = 10^{-3}$, $\gamma = .0025$, $\bar{\rho} = 0$)	99.69% (0.01%)	99.1 (1.2)	1583.4 (17.4)	stopped at 5.9 (0.2)
fuzzy ARTMAP with MT- ($\varepsilon = 10^{-4}$, $\alpha = .001$, $\beta = 1$, $\bar{\rho} = 0$)	99.56% (0.01%)	213.2 (6.0)	744.3 (19.2)	3.8 (0.1)

Consider the following example. Assume that in 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 $T_{A.1} = T_{B.1}$ 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 raises the vigilance parameter ρ to $(|\mathbf{A}_2 \wedge \mathbf{w}_{A.1}|/M) + \varepsilon$, 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 creates another minimum-sized prototype $\mathbf{w}_{B.2} = \mathbf{w}_{B.1}$. From epoch to epoch, the same phenomenon repeats itself, yielding ever more identical prototypes $\mathbf{w}_{B.\mathrm{n}} = \mathbf{w}_{B.1}$ for $\mathrm{n} = 3, 4, \ldots, \infty$.

This phenomenon was observed while training fuzzy ARTMAP, ART-EMAP and Gaussian ARTMAP on the radar data set. Results in Table 2 were obtained through manual termination by: (1) detecting, from epoch to epoch, the repeated creation of identical prototypes for the same training patterns, (2) pruning non-unique prototypes from memory, and (3) defining the convergence time as the number of epochs leading to the creation of non-duplicate prototypes only.

ARTMAP-IC converged incrementally, by itself, on the radar data. The feature of ARTMAP-IC that circumvents the convergence problem is called *negative match tracking*, denoted MT-, which consists of using a negative ε value (Carpenter and Markuzon, 1998). In the above example, mismatch raises ρ but $\mathbf{w}_{B.1}$ would still pass the vigilance test. This allows learning of fully overlapping non-unique prototypes for training set patterns that belong to different classes. As shown in Table 2, fuzzy ARTMAP with MT- performs as well, to within standard error, as without MT-, yet circumvents the convergence problem. As pointed out by Carpenter and Markuzon (1998), MT- is a better algorithmic approximation to the continuous-time version of fuzzy ARTMAP.

5.4 Classification of Incomplete Data. A neural network classifier applied to radar ESM may be subjected to data, either during training or testing, that is incomplete in one or more of the following ways (Granger et al., 2000):

1. Limited number of training cases. Collecting and analyzing ESM data from the field of operation to train a neural network can be a costly undertaking. Yet, if the number of training cases is insufficient, the classifier may not achieve good generalization during operations. It is therefore of interest to know how the performance of the classifier declines as the amount of training data is decreased, so that, *e.g.*, more training data may be gathered, if necessary, before the classifier is fielded. The effect on fuzzy ARTMAP performance of reducing the amount of training data from each radar type was characterized by Granger et al. (2000). Overall, fuzzy ARTMAP achieves a high level of accuracy when training with very few pulses from each radar type.

2. Missing components of the input patterns. The information in the different components of the PDW comes from a number of sources. Absence of components in radar ESM processing arises due to sensor limitations and/or delay in deriving parameters (*e.g.*, PRI). This implies that the classifier may encounter partial input patterns. A strategy is presented later in this section that allows fuzzy ARTMAP to effectively process partial input patterns during both training and testing.

3. Missing class labels during training. The task of analyzing radar ESM data collected in the field can be difficult owing to the complexity and lack of control of the environment. Expertise and experience are required for manual separation and labeling of pulse trains transmitted

by different emitters. This problem raises the question of whether the classifier may benefit from training on data with missing class labels. Training on such data is referred to as "semi-supervised learning" (Demiriz et al., 1999) or "partially supervised clustering" (Bensaid, 1996; Pedrycz, 1985).

To assess the effect on performance of training fuzzy ARTMAP using data with missing class labels, the network was trained in two phases (Granger et al., 2000). During the first phase, involving supervised learning, the network was trained as usual until convergence with a fixed amount of labeled training data from each radar type. During the second phase, involving unsupervised learning, the network was presented with a varying percentage of unlabeled data from each radar type until the weights **W** converged once again. Using fuzzy ARTMAP without the class prediction (*i.e.,* without Step 4 of the fuzzy ARTMAP algorithm), plus other modifications discussed by Granger et al. (2000), the network associated each unlabeled training pattern with one of the already-existing F_2 category nodes and adjusted the corresponding prototype vectors through slow learning ($0 < \beta < 1$). In all simulations with the radar data, the classification rates observed were never greater than those achieved by simply discarding all unlabeled data (Granger et al., 2000). Such an approach is most effective to the extent that clusters of data from different emitters are separable and well clustered, which is not necessarily the case for radar ESM data.

4. Missing classes during training. New radar types (not represented in the training set) may be encountered during operations. When the classifier receives a pattern transmitted by an new radar type, it would be desirable to "flag" the pattern as unfamiliar, rather than make a meaningless guess as to its class label. This may be implemented by *familiarity discrimination* (Carpenter et al., 1997a; Parra et al., 1996). The importance in radar ESM of familiarity discrimination during operations is evident: radar emitters can exhibit new modes at any time. The ability to learn unfamiliar classes is anticipated to be just as important. Presently, the task of providing training data to a neural network classifier requires time from an ESM analyst, so it cannot be expected that more than a small fraction of the large amount of available data would be labeled for training. Furthermore, it cannot be assumed that all of the unlabeled training data belongs to one of the radar types identified by the ESM analyst. (Although not explored here, this would involve performing familiarity discrimination during semi-supervised training on unlabeled data.) The modifications presented later in this section enable fuzzy ARTMAP to mitigate performance degradation due to missing class labels, while allowing it to benefit from learning information hidden in unlabeled data about as-yet unfamiliar classes.

Modifications to fuzzy ARTMAP with MT- are now introduced for dealing with missing components of the input pattern (Section 5.5), and missing classes during training (Sections 5.6 and 5.7). Performance obtained with these modifications is assessed via computer simulation using the methodology, evaluation criteria and radar pulse data described in Sections 4 and 5.1.

5.5 Indicator Vector Strategy for Missing Components.

A strategy to address missing components in the input patterns consists in using *indicator vectors* (Ghahramani and Jordan, 1994; Granger et al., 2000; Little and Rubin, 1987). An indicator vector $\delta = (\delta_1, \delta_2, ..., \delta_M)$ informs the fuzzy ARTMAP network about the presence or absence of each component in an input pattern: $\delta_i = 1$ if component i is present, and $\delta_i = 0$ if component i is missing, with $\delta_{i+m} \equiv \delta_i$ for $i = 1, \ldots, m$. Unlike strategies that involve replacement by "0" or by "1" (Granger et al., 2000), the indicator vector strategy modifies the prototype vectors as well

Table 3: Algorithmic modifications to fuzzy ARTMAP required for implementation of the indicator vector strategy. (Refer to fuzzy ARTMAP equations in Table A.1.)

Algorithmic step	**fuzzy ARTMAP**	**fuzzy ARTMAP with indicator vector**
Prototype selection:		
– choice function	$T_j(\mathbf{A}) = \dfrac{\lvert\mathbf{w}_j \wedge \mathbf{A}\rvert}{\alpha + \lvert\mathbf{w}_j\rvert}$	$T_j(\mathbf{A},\delta) = \dfrac{\lvert\mathbf{w}_j \wedge \mathbf{A} \wedge \delta\rvert}{\alpha + \lvert\mathbf{w}_j \wedge \delta\rvert}$
– vigilance test	$\lvert\mathbf{w}_j \wedge \mathbf{A}\rvert \geq \rho\lvert\mathbf{A}\rvert$	$\lvert\mathbf{w}_j \wedge \mathbf{A} \wedge \delta\rvert \geq \rho\lvert\mathbf{A} \wedge \delta\rvert$
Learning:		
– prototype update	$\mathbf{w}'_J = \beta(\mathbf{A} \wedge \mathbf{w}_J) + (1-\beta)\mathbf{w}_J$	$\mathbf{w}'_J = \beta((\mathbf{A} \vee \delta^c) \wedge \mathbf{w}_J) + (1-\beta)\mathbf{w}_J$

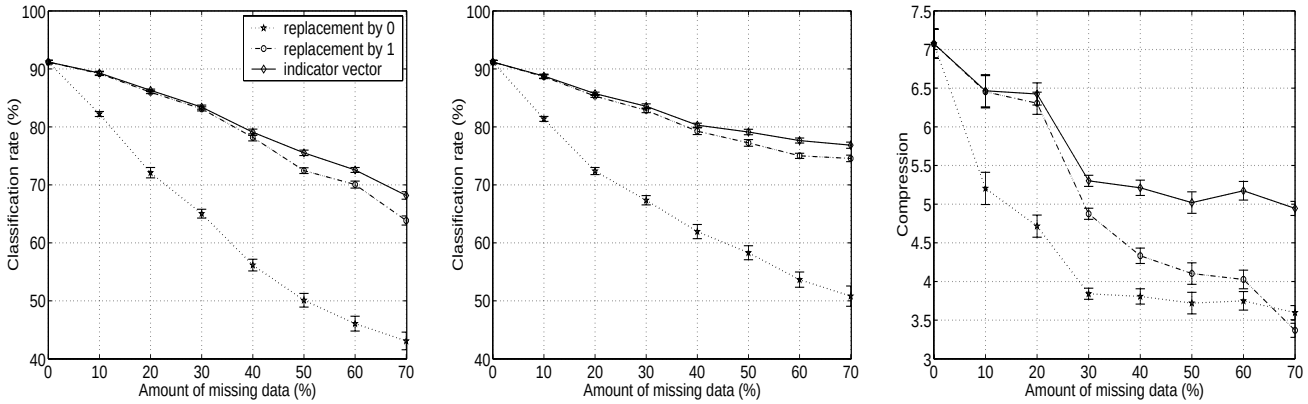
as the input pattern in response to missing components. This approach circumvents a bias in the order of prototype selections by F_2 nodes. The adjustments to the fuzzy ARTMAP algorithm that realized the indicator vector strategy are summarized in Table 3.

To verify the effectiveness of this strategy, fuzzy ARTMAP with MT- and indicator vector was trained using a randomly-selected 0.5% of the available training set from each radar type class. (Recall from Section 5.2 that the available training subset consists of a random selection of 50% of the pulses from each class. The remaining data forms the test set.) For each class in the training set, a variable percentage, between 0% and 70%, of the components were randomly removed from the patterns and declared to be "missing" (although, if a particular choice of missing components would have left the pattern with *no* components, another random choice was made). The classification rate and compression are shown in Figure 5 for the indicator vector strategy, as well as for replacement by "1" and replacement by "0." Also shown are the results obtained when components are removed from the test set. Whether components are missing during training or testing, the indicator vector strategy provides a simple and effective means of handling the absence of components, as its performance degrades gracefully as a function of the percentage of missing data.

Also noteworthy are the results obtained when there are no missing components. With a randomly-selected 0.5% of the available training data from each class (about 130 pulses total), the classification rate on the test set is 91.4%, compared to 99.6% when all the training data (about 26000 pulses total) are used. The slow decline in accuracy is in part due to the uneven distribution of pulses among radar type classes in the data set.

5.6 Familiarity Discrimination. An extension to fuzzy ARTMAP that allows for detection of patterns from unfamiliar classes is called ARTMAP-FD. The ARTMAP-FD algorithm has been shown to effectively perform *familiarity discrimination* on simulated radar range profiles (Carpenter et al., 1997a), and radar pulse data (Granger et al., 1999a). In addition to the classification rate on patterns from familiar classes, the performance of the classifier can be measured in terms of a *hit rate* (H) — fraction of familiar-class test patterns correctly predicted to belong to one of the familiar classes — and a *false alarm rate* (F) — fraction of unfamiliar-class test patterns incorrectly predicted as familiar by the classifier.

5.6.1 ARTMAP-FD algorithm. With complement coding and fast learning ($\beta = 1$), the M-dimensional prototype vector $\mathbf{w}_j = (w_{j1}, w_{j2}, \ldots, w_{jM})$ associated with F_2 layer category node j



(a) Accuracy for missing components during testing only.

(b) Accuracy for missing components during training only.

(c) Compression associated with Figure 5(b).

Figure 5: Average performance of fuzzy ARTMAP with MT-, over 20 simulation trials, using strategies to manage input patterns with missing input components. (Error bars are standard error of the sample mean.)

of fuzzy ARTMAP defines a hyperrectangle in the *m*-dimensional space of input pattern components, with edges parallel to the coordinate axes. Such a hyperrectangle records the largest and smallest component value of the training set patterns assigned category j. A pattern that is associated with an F_2 node during testing is "completely familiar" if it falls within the hyperrectangle, and "less-than-completely familiar" to the extent that it falls outside. This notion can be quantified by the *familiarity measure*:

$$\phi(\mathbf{A}) = \frac{T_J(\mathbf{A})}{T_J^{\max}} = \frac{|\,\mathbf{A} \wedge \mathbf{w}_J\,|}{|\,\mathbf{w}_J\,|}\ , \tag{4}$$

where $T_J^{\max} = |\,\mathbf{w}_J\,|\,/(\alpha + |\,\mathbf{w}_J\,|)$. The maximal value of $T_J = T_J^{\max}$ is attained for an input $\mathbf{a}$ if it lies inside the hyperrectangle associated with node J, for then $|\,\mathbf{A} \wedge \mathbf{w}_J\,| = |\,\mathbf{w}_J\,|$. In other words, an input $\mathbf{a}$ that is assigned category J during testing has the maximum familiarity value $\phi(\mathbf{A}) = 1$ if and only if $\mathbf{a}$ lies within hyperrectangle R_J.

ARTMAP-FD is identical to fuzzy ARTMAP during training. During testing, $\phi(\mathbf{A})$ is computed for each input pattern $\mathbf{a}$ after fuzzy ARTMAP has selected node J, and tentatively predicted class $K = k(J)$. An input $\mathbf{a}$ is declared to belong to a *familiar* class if the value of the familiarity measure $\phi(\mathbf{A})$ is greater than a decision threshold γ. In this case, ARTMAP-FD outputs the prediction of class K for $\mathbf{a}$. Otherwise ($\phi(\mathbf{A}) \leq \gamma$), input $\mathbf{a}$ is flagged as belonging to an unfamiliar class, and ARTMAP-FD makes no prediction.

5.6.2 Familiarity threshold selection. The choice of a particular familiarity threshold $\gamma = \Gamma$ for use during operations depends upon the relative cost of errors due to *misses* (patterns belonging to familiar classes that the network flags as unfamiliar) and *false alarms*. Since familiarity discrimination involves placing an input into one of two sets, familiar or unfamiliar, the Receiver Operating Characteristic (ROC) formalism (Helstrom, 1995) can be used to measure the effectiveness of ARTMAP-FD. Optimizing Γ corresponds to choosing a point on the parameterized ROC curve that is close to the upper left-hand corner of the unit square. This maximizes correct

selection of familiar patterns (H) while minimizing incorrect selection of unfamiliar patterns (F). Two methods for predicting a familiarity threshold Γ value are described in (Carpenter et al., 1997b). A variant of the "on-line threshold determination" method has been chosen for this work and is now described.

During the first ARTMAP-FD training epoch, every time a category node J wins the competition for a pattern $\mathbf{a}$, fast learning expands R_J just enough to enclose $\mathbf{a}$. Before learning takes place, $\phi(\mathbf{A})$ is be computed, and can have a value less than one. The degree to which $\phi(\mathbf{A})$ is less than 1 reflects the distance from the training pattern to R_J. A training pattern successfully coded by a category node (without reset) is taken to be representative of familiar test-set patterns. The corresponding familiarity measure $\phi(\mathbf{A})$ contributes to the generation of a training hit rate curve, where $H(\gamma)$ equals the fraction of training inputs with $\phi(\mathbf{A}) > \gamma$. In contrast, a reset event during the first training epoch resembles the arrival of an unfamiliar pattern during testing, where reset occurs when a category node J that predicts class K wins the competition for a pattern that actually belongs to a different class k, $k \neq K$. The set of $\phi(\mathbf{A})$ values corresponding to these events are used to generate a training false-alarm rate curve, where $F(\gamma)$ equals the fraction of match-tracking inputs with $\phi(\mathbf{A}) > \gamma$.

After training, the predicted familiarity threshold is given by $\Gamma = \arg\max_\gamma\{H(\gamma) - F(\gamma)\}$. Predictive accuracy is improved by use of a reduced set of $\phi(\mathbf{A})$ values in the training-set ROC curve construction process; namely, training patterns that fall inside a hyperrectangle ($\phi(\mathbf{A}) = 1$), are not used because these exemplars tend to distort the miss-rate curve. In addition, the *first* incorrect response to a training input is the best predictor of the network's response to an unfamiliar testing input, since sequential search will not be available during testing. Finally, giving more weight to events occurring later in the training process improves accuracy. In this paper, this is accomplished by computing the training curves $H(\gamma)$ and $F(\gamma)$, and predicting the threshold Γ, from the data presented only after the system has created a number of category nodes equal to L (the number of training set classes). Note that this threshold determination method requires storage of all of the training patterns, so as to obtain $H(\gamma)$ and $F(\gamma)$ and thereby predict Γ. For the sake of computational efficiency, it should be possible to approximate $H(\gamma)$ and $F(\gamma)$ from a reduced set of $\phi(\mathbf{A})$ values which would be updated incrementally as new data are obtained.

5.7 Learning of Unfamiliar Classes. With Learning of Unfamiliar Classes (LUC), a classifier continues during the testing phase to adjust its weights via semi-supervised learning. The criteria for familiarity discrimination is also adjusted on-line, and when a test pattern is flagged as unfamiliar, the classifier defines a new class. Subsequent test patterns may be declared by the classifier to be "familiar" and classified as belonging either to classes encountered during training (*i.e.,* training-set classes) *or* to the "newly-minted" classes; or they may be declared to be "unfamiliar," in which case another new class is defined. The hit rate for an LUC classifier (H^*) can be defined as the fraction of test patterns from training-set classes that are correctly declared to belong to one of the training-set classes. The false alarm rate for an LUC classifier (F^*) is the fraction of unfamiliar-class (*i.e.*, not encountered during the training phase) test patterns not either flagged as unfamiliar nor assigned to a "new" class defined during testing. An additional figure of merit for an LUC classifier is a "purity measure," such as the Rand clustering score (Hubert and Arabie, 1985), which rewards the classifier for learning the right number of unfamiliar classes, and correctly assigning them to unfamiliar patterns.

The algorithm for fuzzy ARTMAP was modified as follows to incorporate LUC. First, in

order to focus on the effects of LUC (as opposed to learning with missing class labels), the weights associated with F_2 category nodes allocated during the training phase (training classes) are kept at fixed values during testing. Only "new" F_2 category nodes created during the test phase are allowed to change through slow learning $(0 < \beta < 1)$.

In addition, to prevent the generation of an excessive number of new F_2 nodes, patterns that are declared to be unfamiliar are given a "second chance" to be associated with an already-existing new F_2 node before defining a new class. Specifically, a pattern **a** declared unfamiliar by the network is subjected to a vigilance test at each of the new nodes. If it passes the test for one or more of these nodes, it is associated with the node j_{new} among these new nodes which has the strongest activation $T_{j_{\text{new}}}$. If the pattern cannot in this way be associated with an already-existing new node, and the node J to which the pattern was tentatively assigned is not a new node, then a *coactivation test* is performed. If the activation level T_J of node J is not stronger than the activation level $T_{j_{\text{new}}}$ for all of the new nodes j_{new} by a sufficient amount — that is, if $T_J - T_{j_{\text{new}}} < \varepsilon_{co}$ — then the pattern is associated with the node j_{new} for which $T_J - T_{j_{\text{new}}}$ is smallest. If either of these two tests is passed, then no weight adjustment takes place for $\mathbf{w}_{j_{\text{new}}}$. Only if neither of these options for association with an already-existing new node succeeds is a new class defined.

When a new class is defined, a new F_2 category node J_{new} is allocated to encode the input $(\mathbf{w}_{J_{\text{new}}} \leftarrow \mathbf{A})$, and linked to a new F^{ab} map node K_{new} through $\mathbf{w}^{ab}_{J_{\text{new}}}$. Slow semi-supervised learning subsequently adjusts the prototype weights of new F_2 nodes upon their assignment to familiar patterns. Notice that newly-defined classes are interpreted differently: each new F_2 or F^{ab} node represents a fragment of data from an emitter mode that belongs to an unfamiliar radar type.

Finally, fuzzy ARTMAP-LUC extends the on-line method to allow for enhancement of the familiarity threshold Γ from test-set patterns. If pattern **a** is declared familiar, $\phi(\mathbf{A})$ is added to the $\{\phi\}$ values used to generate the training set hit rate curve $H^*(\gamma)$; otherwise $\phi(\mathbf{A})$ is added to the $\{\phi\}$ values used to generate the false alarm rate curve $F^*(\gamma)$. The threshold $\Gamma^* = \arg\max_\gamma\{H^*(\gamma) - F^*(\gamma)\}$ is recomputed following each input pattern assignment. For faster execution, it is possible to adjusts Γ every time a pattern is assigned a new F_2 node; that is, when $\mathbf{W}$ is modified.

5.8 Simulations with Familiarity Discrimination and Unfamiliar Classes.

In computer simulations, 13 out of the 15 radar types were declared to be familiar (thus training classes). Familiar class selection was performed at random, with the restriction that an insufficient number of unfamiliar-class data patterns (less than a thousand) was not allowed. A randomly-selected 50% of the data from each of the 13 training classes were presented to the network during the training phase. The familiarity threshold Γ was determined during the training phase using the on-line method described in Section 5.6.2. Patterns remaining from the 13 training classes, plus all the patterns from the 2 unfamiliar radar type classes formed the test set.

Average results obtained for fuzzy ARTMAP with FD (pure ARTMAP-FD) and for fuzzy ARTMAP with FD and LUC are shown in Table 4. Overall results indicate that fuzzy ARTMAP with FD has a high hit rate (99.60%), but only marginal performance with regards to avoiding false alarms: 14.33% of patterns belonging to unfamiliar classes are mistakenly assigned familiar training classes. This is a consequence of the overlap, scattering, and uneven mixture of pulses from different radar types. By defining new classes and assigning them to unfamiliar-

Table 4: Average results, over the same 20 simulation trials, using ARTMAP-FD with and without LUC. (Numbers in parentheses are the standard error of the sample mean.)

Evaluation criteria	**fuzzy ARTMAP with FD**	**fuzzy ARTMAP with FD and LUC**
Hit rate	99.60% (0.07%)	99.63% (0.05%)
False alarm rate	14.33% (8.89%)	7.46% (4.25%)
Classification rate	99.51% (0.05%)	99.49% (0.05%)
Memory	806.2 (40.1)	931.1 (41.4)
Rand clustering score	N/A	0.7640 (0.0597)

class patterns, LUC reduces this false alarm rate by about half, to 7.46%, without loss of accuracy.

On average, fuzzy ARTMAP with FD and LUC requires an additional 124.9 registers to store the prototype vectors, and thus about 21 new F_2 (plus F^{ab}) nodes to represent the different emitter modes from the 2 unfamiliar classes. Despite the creation of these additional nodes, the network's predictive accuracy on data from familiar classes is not significantly degraded. The moderate Rand score, 0.7640, indicates the ability of LUC to recover some of the true cluster structure in data from the 2 unfamiliar classes.

6 Pattern Clustering

The objective of pattern clustering in the What-and-Where recognition architecture shown in Figure 2 is to group patterns from the Where data stream into tracks. Impinging signals contain information about emitter status, which may change in time. Desirable features for such on-line clustering include the ability to initialize new tracks whenever new emitters are detected, to adjust tracks in response to emitter maneuvers, and to delete tracks as emitters leave or stop transmitting.

Several techniques can perform on-line sequential clustering. For instance, adaptive vector quantization (Gray, 1984) algorithms may be used if parameter values vary slowly. More sophisticated tracking algorithms (Bar-Shalom and Li, 1993; Blackman, 1986) are needed to update tracks for parameters that exhibit rapid linear or nonlinear variations. In this section, on-line clustering of Brg and PA parameters is implemented by combining nearest-neighbor matching with linear Kalman filtering. A breakdown of the recursive processing required for on-line clustering is given in Figure 6. The three basic functions — data association, track maintenance, and filtering and prediction — are now examined.

6.1 Data Association. An incoming pattern **b** from the Where data stream is initially considered for association with existing tracks. This association involves computing a *match* $s_h(\mathbf{b})$ between input **b** and the *next predicted* position of every track ($h = 1, 2, \ldots, R$) in the Where environment. Assume that track positions are drawn independently and identically from a mixture of Gaussian distributions, in which one distribution is associated with each track.

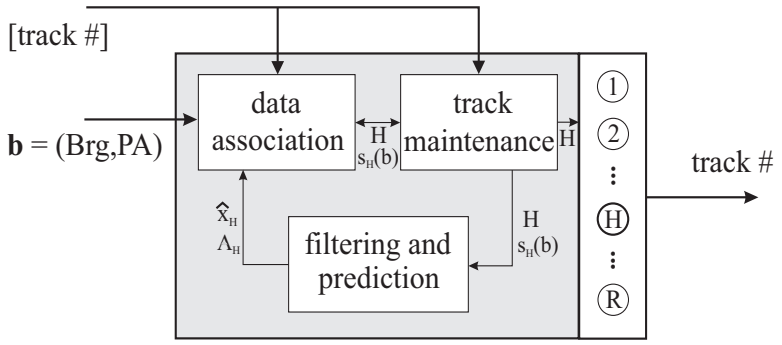


Figure 6: Pattern clustering system based on nearest-neighbor matching and Kalman filtering.

Then, the match can be taken to be a probability, and written:

$$s_h(\mathbf{b}) = \frac{1}{(2\pi)^{\frac{M}{2}} |\Lambda_h|^{\frac{1}{2}}} \exp\{-\frac{1}{2}(\mathbf{b}-\widehat{\mathbf{x}}_h)^T \Lambda_h^{-1}(\mathbf{b}-\widehat{\mathbf{x}}_h)\} \tag{5}$$

where M is the number of dimensions in the Where space, and $\widehat{\mathbf{x}}_h$ and Λ_h are, respectively, the predicted position and covariance matrix for track h. Assuming that all tracks have equal prior probabilities, the track $h = H$ that maximizes Equation 5 is associated with $\mathbf{b}$:

$$H = \arg\max_h \{s_h(\mathbf{b}) : h = 1, 2, ..., R\}. \tag{6}$$

Kalman filtering (Bar-Shalom and Li, 1993; Blackman, 1986) is employed to predict the next position $\widehat{\mathbf{x}}_h$, and covariance matrix Λ_h of each track h. Recall from Section 3 that the usual clustering is bypassed when $\mathbf{b}$ corresponds to a PDW that has been assigned a previously-established track through TOA deinterleaving. In this case, $\mathbf{b}$ retains its track, and does not perform data association, nor track maintenance. Kalman filtering and prediction are however still performed in order to sustain a consistent description of all active emitters in the environment.

6.2 Track Maintenance. Once associated with pattern $\mathbf{b}$, track H undergoes two tests. In the first, the match $s_H(\mathbf{b})$ is compared to a threshold δ_c that regulates the creation of new tracks, $\delta_c \in [0,1]$. If $s_H(\mathbf{b}) \geq \delta_c$, then the test is passed. In the second test, the cumulative average of match value S_H is computed:

$$S_H = \frac{\sum s_H(\mathbf{b})}{Q_H}\ , \tag{7}$$

where Q_H is the number of patterns to which track H was assigned. S_H is compared to another threshold δ_d, that regulates the overall quality of existing tracks, $\delta_d \in [\delta_c, 1]$. If $S_H \geq \delta_d$, then the test is passed. If both tests are passed, then track H is *assigned* to $\mathbf{b}$.

If either test is failed, then a new track is initiated for pattern $\mathbf{b}$. When a new track H is initiated, $s_H(\mathbf{b}) = 1$, $\widehat{\mathbf{x}}_H$ is set equal to $\mathbf{b}$ and Λ_H is set equal to $\sigma^2 I_M$, where I_M is the identity matrix, and $\sigma = (\sigma_1, \sigma_2, ..., \sigma_M)$ represents the resolution of Where parameter measurements. Furthermore, if the second test is failed (*i.e.*, $S_H < \delta_d$), then the previously-established track H is deleted.

After assignment of H to either an existing or newly-initiated track, any track h that has not been assigned to an input pattern for a time greater than an ageout parameter $\tau > 0$ is deleted.

That is, a track is deleted if:

$$\mathrm{TOA}(\mathbf{b}) - \mathrm{TOA}_h > \tau \,, \tag{8}$$

where TOA_h is the time at which track h was last assigned to an input. Deleting a track frees up resources, and reduces the chances of future miss-assignments. The track number H is the output from track maintenance.

A high quality track is one that is assigned pulses transmitted, to a large extent, from the same emitter. To ensure high quality tracks, new tracks are initiated rapidly by setting δ_c close to 1, whereas poor quality tracks are deleted more slowly (as S_H progressively declines). This reduces ambiguity during track assignment, but may lead to the initiation of a greater number of tracks, and thus predictions K^e based on the accumulation of short sequences of pulses.

6.3 Kalman Filtering and Prediction. Kalman filtering and prediction is implemented with a bank of standard Kalman filters, one per track. Every track h is associated with a Kalman filter, and is represented by a unimodal Gaussian distribution. Upon the assignment of a track H to $\mathbf{b}$, the filter of H is employed to predict its next position and covariance matrix.

7 Sequential Evidence Accumulation

Sequential evidence accumulation exploits Where information by combining the responses of pattern clustering with neural network classification. In short, the classifier's responses are accumulated according to track, thus offering predictions from multiple views of an emitter.

7.1 Fusion of What and Where information. As mentioned in Section 6, clustering produces a track number $h = H$ for each pattern $\mathbf{b}$ from the Where data stream. The track number indicates the specific emitter assigned to $\mathbf{b}$.

Sequential evidence accumulation is implemented by means of identical evidence accumulation fields $F_1^e, F_2^e, \ldots, F_R^e$, where each field F_h^e is connected to a track h, and replicates the neural network classifier's output field, that is, contains L nodes, one per radar type class. The classifier's output nodes are linked to their respective nodes in all fields F_h^e, $h = 1, 2, \ldots, R$. Each field F_h^e incorporates a short-term memory capable of accumulating its input patterns. The memory for F_h^e is characterized by a field accumulation pattern $\mathbf{T}_h^e = (T_{h1}^e, T_{h2}^e, \ldots, T_{hL}^e)$.

Upon initiation of a track h, $\mathbf{T}_h^e$ is set equal to $\mathbf{0}$. When track $h = H$ is assigned to pattern $\mathbf{b}$, F_H^e becomes active. The activity pattern $\mathbf{y}^{ab}$ output by the classifier accumulates onto F_H^e according to:

$$(\mathbf{T}_H^e)' = \mathbf{T}_H^e + \mathbf{y}^{ab} \,. \tag{9}$$

Accumulation of activity patterns in F_h^e continues until track h is deleted.

For a given input PDW, the activity pattern $\mathbf{y}^e$ output from evidence accumulation is equal to $\mathbf{T}_H^e$. The radar type is predicted to be:

$$K^e = \arg\max_{k^e} \{ T_{Hk^e}^e : k^e = 1, 2, \ldots, L \} \tag{10}$$

Besides discrete prediction, evidence accumulation fields can be used to feed an emitter table in the signal separator of Figure 1. The fields can also describe multiple radar types associated

with same Where features, for instance the same location, which can assist in linking emitters to platforms.

7.2 Prediction from multiple views. Sequential evidence accumulation may improve the overall classification rate of the recognition system, since the accumulated prediction K^e is tolerant to errors committed by the neural network classifier (prediction K). The concept of predicting classes from multiple "views" of a source, that are accumulated through time, has been successfully developed in a number of neural network architectures (Baloch and Waxman, 1991; Bradski and Grossberg, 1995; Carpenter and Ross, 1995). In the present case, information on the origin of input patterns is provided through clustering of the Where data. The effectiveness of evidence accumulation here depends on the quality of the tracks that are computed as in Section 6.

7.3 Simulations with Evidence Accumulation. This section summarizes simulations of how the entire What-and-Where system performs on the radar pulse data set. Software was written in the Matlab language to implement the neural network classification, clustering and evidence accumulation modules, as well as their integration. The parameter settings used for clustering were $\delta_c = 0.98$, $\delta_d = \delta_c + 0.01$, and $\tau = 10$ ms. The neural network classifier was fuzzy ARTMAP with negative match tracking (MT$-$), indicator vector strategy (IVS), familiarity discrimination (FD) and learning of unfamiliar classes (LUC), as described in Section 5. The vigilance and coactivation parameters used for associating unfamiliar patterns with already-existing new nodes in LUC were $\rho = 0.8$ and $\varepsilon_{co} = 0.05$. A learning rate of $\beta = 0.5$ was used for semi-supervised adjustment of prototype weights for new nodes.

Patterns presented to the fuzzy ARTMAP classifier contained 3 What parameters — namely PRI, PW and RF — whereas patterns presented to the clustering module contained 2 Where parameters — namely Brg and PA. The system normally receives the track number and PRI value for PDWs to which a track was assigned by the TOA deinterleaver. TOA deinterleaving is a difficult task due to the agility (jitter, stagger, etc.) of the PRI in modern radar systems. For simulation purposes, it was assumed that TOA deinterleaving can, during on-line operation, be reliably achieved only for constant PRIs. Approximately 30,000 pulses from the data set belong to constant PRI emitters, and the rest (about 22,000 pulses) belong to complex PRI emitters.

During fuzzy ARTMAP training, all What patterns had a PRI component. For emitters with complex PRI patterns, the PRI values were computed as in Section 5. During testing, the PRI components were declared missing from test-set patterns that belonged to emitters with complex PRI patterns. In addition, a randomly-selected 10% of the What components from each emitter mode were also declared missing. The IVS was used by fuzzy ARTMAP to deal with missing components.

For emitters with constant PRI, the PRI values used for both training and testing were mean PRI values, estimated using a moving average that accounts for dropped pulses. Assume that the TOA deinterleaver has correctly sorted the n_k pulses belonging to emitter mode k, and computed the $\mathrm{PPI}_k(i)$ and $q_k(i)$ values for $i = 2, 3, \ldots, n_k$, where $q_k(i)$ is the closest multiple of the nominal PRI_k value (taken from a PRI table for emitter mode k) to the $\mathrm{PPI}_k(i)$ value; namely $q_k(i) = \arg\min_q \{q : |q \cdot \mathrm{PRI}_k - \mathrm{PPI}_k(i)| : q = 1, 2, 3 \ldots\}$. The moving average is:

$$\overline{\mathrm{PRI}}_k(i) = \frac{1}{\Delta} \sum_{m=i-(l+1)}^{i} \frac{\mathrm{PPI}_k(m)}{q_k(m)} \ , \tag{11}$$

where Δ is the number of pulses in the moving average window. Observing the PPI of pulses inside a window defined by the last 5 consecutive PRIs is consistent with the clustering parameter $\tau = 10$ms, which corresponds to 5 times the longest nominal PRI value that is expected. If the PPI elapsed for a pulse goes beyond this window, the pulse is assumed to belong to the next burst of pulses from the emitter.

To account for the chronological evolution of the environment, the training set was formed by selecting the first 0.5% of data encountered from each emitter mode in TOA order. Furthermore, this training data was taken only from the emitter modes corresponding to 13 of the 15 radar types selected at random, and declared to be familiar. In practice, this could correspond to training the neural network classifier on the first few bursts of pulses intercepted from each familiar-class emitter. After training, all the remaining data from the 13 familiar classes (99.5% from each emitter mode), plus all the patterns from the 2 unfamiliar classes, were presented to the recognition system in TOA order for prediction. To deal with missing classes during training, FD and LUC (as described in Section 5) were used. When applied with the IVS, the familiarity measure of Equation 4 becomes:

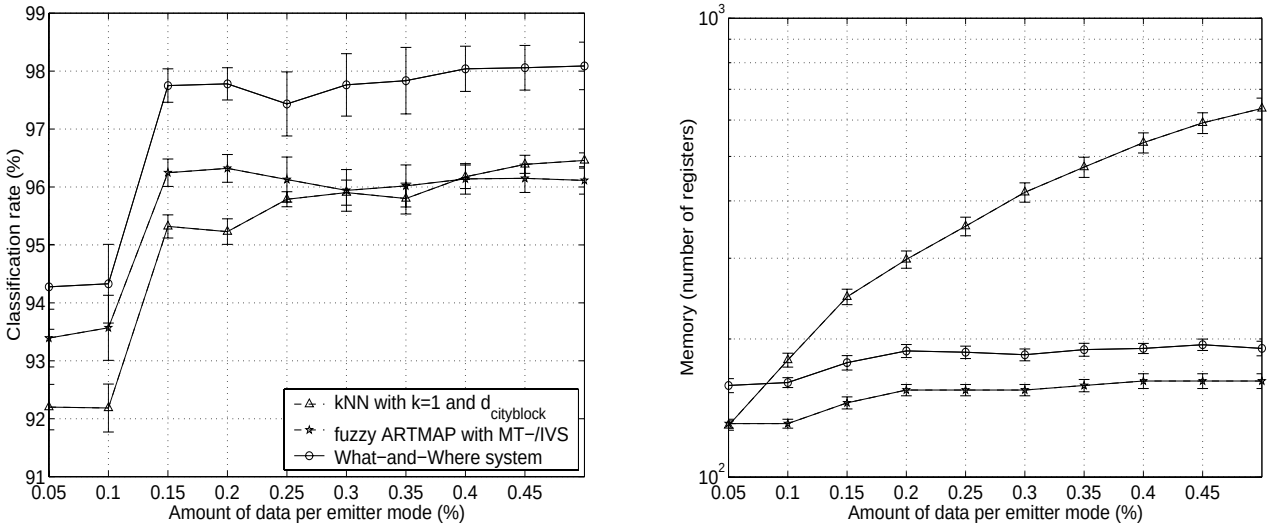
$$\phi(\mathbf{A}) = \frac{|\, \mathbf{A} \wedge \mathbf{w}_J \wedge \delta \,|}{|\, \mathbf{w}_J \wedge \delta \,|} \,. \tag{12}$$

When using fuzzy ARTMAP with FD alone, evidence accumulation of responses $\mathbf{y}^{ab}$ onto field F_H^e occurs only if the input $\mathbf{a}$ is declared familiar. Using fuzzy ARTMAP with FD and LUC, evidence accumulation always occurs since new classes are defined for unfamiliar-class patterns. To support the ability to accumulate unfamiliar-class patterns, whenever a new class is defined, a new node is initialized within each evidence accumulation field F_h^e, $h = 1, 2, \ldots, R$, as well as in the F_2 and F^{ab} layers.

The performance of the What-and-Where system as a function of the amount of data used for training is summarized in Figure 7. The amount of data is a randomly-selected percentage of patterns from each emitter mode in the training subset. This percentage was varied between 10% and 100% of the training data (between 0.05% and 0.5% of the entire data set). Results obtained with the fuzzy ARTMAP with MT- and IVS, and with *k*NN classifiers, are included for comparison. When components are missing, test-set patterns are classified with *k*NN on the basis of parameters that are present.

Figure 7(a) indicates that the What-and-Where system, and thus the fusion of What and Where information, significantly improves the classification rate of fuzzy ARTMAP with MT-/IVS on familiar-class patterns by about 2%. The system achieves a classification rate of about 98% with a training set consisting of as little as 0.15% of the whole data set, or about 80 pulses. This level of performance is attained along with a capability for detecting and learning patterns from unfamiliar classes. When trained on just 0.5% of the data per familiar class, the recognition system yields an average hit rate of $H^* = 96.9\%$ and a false alarm rate of $F^* = 12.9\%$ on the test subset.

The notion that additional training examples beyond a certain point become "redundant" is borne out of Figure 7(b), which shows memory growing as the number of training patterns is increased. The figure also shows memory cost due to defining new classes during testing. When trained on 0.5% of the data per familiar-class emitter modes, the system creates on average 5 new nodes on F_2, F^{ab}, and all F_h^e layers. The Rand clustering quality score of the unfamiliar-class patterns assigned to these nodes is on average 0.73. This is a slight improvement to the score of 0.70 obtained with fuzzy ARTMAP with MT-/FD/LUC.



(a) Classification rate on familiar-class patterns. (b) Memory requirements during training.

Figure 7: Average performance, over 20 simulation trials, of the What-and-Where system. (Error bars are standard error of the sample mean.) (a) (b) Memory requirements during training.

8 Conclusions

A novel What-and-Where architecture has been proposed for recognition and tracking of radar emitters for Electronic Support Measures (ESM). This architecture combines a neural network classifier, an on-line clustering algorithm, and an evidence accumulation module. Once trained on samples of data gathered in the field of operation, the neural network classifier can predict the radar type of intercepted pulses based on their What parameters. Meanwhile, the clustering algorithm separates these pulses according to emitter based on their Where parameters. The evidence accumulation module permits fusion of the classifier's What responses with the clustering algorithm's Where estimates, and thus allows prediction of the radar type from classifications along an entire emitter trajectory. For proof-of-concept computer simulations using a radar data set, a particular realization of the recognition system has been considered. It consists in using a variant of fuzzy ARTMAP for classification, and nearest-neighbor matching with a bank of Kalman filters for on-line clustering.

Simulations results show that fuzzy ARTMAP with negative match tracking (MT-), a core concept of the ARTMAP-IC algorithm (Carpenter and Markuzon, 1998), consistently delivers a high level of accuracy and compression on the radar pulse data set, even when the amount of training data is limited. Compared to several other ARTMAP variants, as well as the reference RBF and *k*NN classifiers, it yields one of the best classification rates, yet requires among the least resources (shortest convergence time and least storage for prototypes) and computational complexity for on-line predictions. The MT- feature allows fuzzy ARTMAP to converge naturally, by circumventing a node proliferation problem that can arise when identical or nearly-identical input patterns in the training data correspond to different classes. Modifications of fuzzy ARTMAP with MT- have also been introduced for dealing with missing components of the input pattern, and missing classes during training. The indicator vector strategy (IVS) provides an effective means of processing partial input patterns, whenever components of the

data set are missing during training or testing. Familiarity discrimination (FD) allows fuzzy ARTMAP to detect patterns belonging to unfamiliar classes during training and testing, and enables learning of unfamiliar classes (LUC) to take place during testing.

Computer simulations that test the entire What-and-Where system improve accuracy significantly over those obtained with fuzzy ARTMAP with MT- and IVS, and with *k*NN, while also detecting and learning patterns from unfamiliar classes. These results support the general approach of integrating What and Where information via evidence accumulation, and offer promise for application in autonomous ESM systems which may be subjected to complex, incomplete, and overlapping radar data.

9 Acknowledgements

This research was supported in part by the Defense Advanced Research Projects Agency and the Office of Naval Research ONR N00014-95-1-0409 (S. G. and M. A. R.), the National Science Foundation NSF IRI-97-20333 (S. G.), the Natural Sciences and Engineering Research Council of Canada (E. G.), and the Office of Naval Research ONR N00014-95-1-0657 (S. G.).

Bibliography

Anderberg, M. R. (1973). *Cluster Analysis for Applications*, New York, NY: Academic Press.

Browne, J. P. R, & Thurbon, M. T. (1998). *Electronic Warfare*, London, UK: Brassey's

Anderson, J. A., Gately, M. T., Penz, P. A., & Collins, D. R. (1990). Radar Signal Categorization Using a Neural Network. *Proc. IEEE*, 78 (10), 1646-1656.

Baloch, A. A., & Waxman, A. M. (1991). Visual Learning, Adaptive Expectation, and Behavioral Conditioning of the Mobile Robot MAVIN. *Neural Networks*, 4, 271-302.

Bar-Shalom, Y., & Li, R. A. (1993). *Estimation and Tracking: Principles, Techniques and Software*, Boston, MA: Artech House.

Bensaid, A. M., Hall, L. O., Bezdek, J. C., & Clarke, L. P. (1996). Partially Supervised Clustering for Image Segmentation. *Pattern Recognition*, 29, 859-871.

Bishop, C. (1995). *Neural Networks for Pattern Recognition*, Oxford, UK: Clarendon Press.

Blackman, S. S. (1986). *Multiple-Target Tracking with Radar Applications*, Norwood, MA: Artech House.

Bradski, G., Carpenter, G. & Grossberg, S. (1994). STORE working memory networks for storage and recall of arbitrary temporal sequences. *Biological Cybernetics*, 71, 469-480.

Bradski, G., & Grossberg, S. (1995). Fast-Learning VIEWNET Architectures for Recognizing Three-Dimensional Objects from Multiple Two-Dimensional Views. *Neural Networks*, 8 (7), 1053-1080.

Carpenter, G. A., & Grossberg, S. (1987). A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine. *Computer, Vision, Graphics and Image Processing*, 37, 54-115.

Carpenter, G. A., Grossberg, S., & Rosen, D, B. (1991a). Fuzzy ART: Fast Stable Learning and Categorization of Analog Patterns by an Adaptive Resonance System. *Neural Networks*, 4 (6), 759-771.

Carpenter, G. A., Grossberg, S., & Reynolds, J. H.. (1991b). ARTMAP: Supervised Real-Time Learning and Classification of Nonstationary Data by a Self-Organizing Neural Network. *Neural Networks*, 4, 565-588.

Carpenter, G. A., Grossberg, S., Markuzon, N., Reynolds, J. H., & Rosen, D. B. (1992). Fuzzy ARTMAP: A Neural Network Architecture for Incremental Supervised Learning of Analog Multidimensional Maps. *IEEE Trans. on Neural Networks*, 3 (5), 698-713.

Carpenter, G. A., & Ross, W. D. (1995). ART-EMAP: A Neural Network Architecture for Object Recognition by Evidence Accumulation. *IEEE Trans. on Neural Networks*, 6 (4), 805-818.

Carpenter, G. A, Rubin, M. A., & Streilein, W. W. (1997a). ARTMAP-FD: Familiarity Discrimination Applied to Radar Target Recognition. *Proc. Int'l Conference on Neural Networks* (Vol. III, pp. 1459-1464).

Carpenter, G. A, Rubin, M. A., & Streilein, W. W. (1997b). Threshold Determination for ARTMAP-FD Familiarity Discrimination. In C. H. Dagli *et al.* (Eds.), *Intelligent Engineering Systems Through Artificial Neural Networks 7* (pp. 23-28), New York, NY: ASME Press.

Carpenter, G. A., & Markuzon, N. (1998). ARTMAP-IC and Medical Diagnosis: Instance Counting and Inconsistent Cases. *Neural Networks*, 11 (2), 323-336.

Chandra, V., Jyotishi, B. K., & Bajpai, R. C. (1988). Some New Algorithms for ESM Data Processing. *Proc. 20th Southeastern Symposium on System Theory* (pp. 108-112).

Chen, S., Cowan, C. F. N., & Grant, P. M. (1991). Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks. *IEEE Trans. on Neural Networks*, 2 (2), 302-309.

Cover, T. M., & Hart, P. E. (1967). Nearest Neighbor Patterns Classification. *IEEE Trans. on Information Theory*, 13 (1), 21-27.

Davies, C. L., & Hollands, P. (1982). Automatic Processing for ESM. *Proc. IEE*, 129 (3), Part F, 164-171.

Demiriz, A., Bennett, K. P., & Embrechts, M. J. (1999). Semi-Supervised Clustering Using Genetic Algorithms. In C. H. Dagli *et al.* (Eds.), *Intelligent Engineering Systems Through Artificial Neural Networks 9* (pp. 809-814), New York, NY: ASME Press.

Dubes, R. C., & Jain, A. K. (1988). *Algorithms for Clustering Data*. Englewood Cliffs, NJ: Prentice Hall.

Duda, R. O., & Hart, P. E. (1973). *Pattern Classification and Scene Analysis*. New York, NY: John Weiley.

Fukunaga, K. (1990). *Introduction to Statistical Pattern Recognition.* 2nd ed., San Diego, CA: Academic Press.

Ghahramani, Z., & Jordan, M. I. (1994). Learning from Incomplete Data. Technical Report, Cambridge, MA: MIT. MIT A. I. LAB Memo No. 1509.

Granger, E., Grossberg, S., Rubin, M. A., & Streilein, W. W. (1999a). Familiarity Discrimination of Radar Pulses. In M. S. Kearns et al. (Eds.), *Advances in Neural Information Processing Systems 11* (pp. 875-881), Cambridge, MA: MIT Press.

Granger, E., Grossberg, S., Lavoie, P., & Rubin, M. A. (1999b). A Comparison of Classifiers for Radar Emitter Type Identification. In C. H. Dagli *et al.* (Eds.), *Intelligent Engineering Systems Through Artificial Neural Networks 9* (pp. 3-11), New York, NY: ASME Press.

Granger, E., Rubin, M. A., Grossberg, S., & Lavoie, P. (2000). Classification of Incomplete Data Using the Fuzzy ARTMAP Neural Network. *Proc. Int'l Joint Conference on Neural Networks* (Vol. IV, pp. 35-40).

Grant, P. M., & Collins, J. H. (1982). Introduction to Electronic Warfare. *Proc. IEE*, 129 (3), Part. F, 621-625.

Gray, R. M. (1984). Vector Quantization. *IEEE ASSP Magazine*, 1 (2), 4-29.

Grossberg, S. (2000). The complementary brain: A unifying view of brain specialization and modularity. *Trends in Cognitive Sciences*, in press.

Helstrom, C. W. (1995). *Elements of Signal Detection and Estimation*. Englewood Cliffs, NJ: Prentice Hall.

Hubert, L., & Arabie, P. (1985). Comparing Partitions. *Journal of Classification*, 2, 193-218.

Kamgar-Parsi, Behrooz, Kamgar-Parsi, Behzad, and Sciortino, J. C.(1996). Automatic Data Sorting Using Neural Network Techniques. Technical Report, Washington: Naval Research Laboratory, Report NRL/FR/5720-96-9803.

Little, R. J. A., and Rubin, D. B. (1987). *Statistical Analysis with Missing Data*. New York, NY: Wiley.

Macedo Filho, A. D., and Griffiths, H. D. (1994). Radar Signal Clustering and Identification by Means of Microwave Neural Networks. *Proc. Int'l Radar Conference* (pp. 470-475).

Maloney, P. S., and Specht, D. F. (1989). The Use of Probabilistic Neural Networks to Improve Solution Times for Emitter-to-Hull Correlation Problems. *Proc. Int'l Joint Conference on Neural Networks* (Vol. I, pp. 289-294).

Mardia, H. K. (1989). New Techniques for the Deinterleaving of Repetitive Sequences. *Proc. IEE*, 136 (4), Part F, 149-154.

Meiler, P. P., and van Wezenbeek, A. M. (1990). BSB Radar Pulse Classification. Technical report, Netherlands: TNO Physics and Electronics Laboratory Report, Report FEL-90-B023.

Pape, D. R., Anderson, J. A., Carter, J. A., and Wasilousky, P. A. (1997). Advanced Signal Waveform Classifier. *Proc. SPIE – The Int'l Society of Optical Engineers*, 3160, 162-169.

Parra, L., Deco, G., and Miesbach, S. (1996). Statistical Independence and Novelty Detection with Information Preserving Non-linear Maps. *Neural Computation*, 8, 260-269.

Pedrycz, W. (1985). Algorithms for Fuzzy Clustering with Partial Supervision. *Pattern Recognition Letters*, 3, 13-20.

Roe, A. L., and Roe, J. (1994). The Application of Artificial Intelligence Techniques to Naval ESM Radar Identification. *Proc. 7th Int'l Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems (IEA/AIE-94)* (pp. 565-572).

Rogers, J. A. V. (1985). ESM Processor System for High Pulse Density Radar Environments. *Proc. IEE*, 132 (7), Part F, 621-625.

Rumelhart, D. E., Hinton, G., and Williams, R. J. (1986). Learning Internal Representations by Error Propagation. In Rumelhart D. E., and McClelland, J, L., eds., *Parallel Distributed Processing: Explorations in the Microstructure of Cognition,* 1, 318-362, Cambridge, MA: MIT Press.

Schleher, D. C. (1986). *Introduction to Electronic Warfare*, Dedham, MA: Artech House.

Schleher, D. C. (1999). *Electronic Warfare in the Information Age*, Norwood, MA: Artech House.

Sciortino, J. C. (November 1997). Autonomous ESM Systems. *Naval Engineers Journal*, 73-84.

Self, A., and Bourassa, G. (1991). Future ESM Systems and the Potential for Neural Processing. *AGARD'91* (Vol. IX, pp. 1-10).

Specht, D. F. (1989). Probabilistic Neural Networks (A One-Pass Learning Method) and Potential Applications. *WESCON'89* (pp. 780-785).

Specht, D. F. (1990). Probabilistic Neural Networks. *Neural Networks*, 3, 109-118.

Tsui, J, B. (1986). *Microwave Receivers with Electronic Warfare Applications*, New York, NY: John Wiley and Sons.

Wang, D. C., and Thompson, J. (1991). An Adaptive Data Sorter Based on Probabilistic Neural Networks. *Proc. IEEE NAECOM'91* (Vol. III, pp. 1-12).

Wiley, R. G. (1993). *Electronic Intelligence: The Analysis of Radar Signals*, 2nd ed., Norwood, MA:Artech House.

Wilkinson, M. A., and Watson, M. A. (1985). Use of Metric Techniques in ESM Data Processing. *Proc. IEE*, 132 (4), Part F, 229-232.

Williamson, J. R. (1996). Gaussian ARTMAP: A Neural Network for Fast Incremental Learning of Noisy Multidimensional Maps. *Neural Networks*, 9 (5), 881-897.

Williamson, J. R. (1997). A Constructive, Incremental-Learning Neural Network for Mixture Modeling and Classification. *Neural Computation*, 9 (7), 1517-1543.

A Appendix A

A.1 ARTMAP Neural Network Classifiers. ART-EMAP (Stage 1) and ARTMAP-IC extend fuzzy ARTMAP to produce a distributed activation of coded F_2 nodes during testing. Furthermore, ARTMAP-IC biases distributed test set predictions according to the number of times F_2 nodes are assigned to training set patterns. It also uses negative match tracking (*i.e.*, negative ε values), to address the problem of inconsistent cases, whereby identical training set patterns correspond to different classes labels. After an incorrect prediction during training, the vigilance parameter is raised just enough to induce a search for another internal category node J, and then lowered by a small amount $\varepsilon > 0$.

Gaussian ARTMAP represents each category j as a separable Gaussian density function, defined by its mean $\mu_j = (\mu_{j1}, \mu_{j2}, ..., \mu_{jM})$ and its standard deviation $\sigma_j = (\sigma_{j1}, \sigma_{j2}, ..., \sigma_{jM})$ vectors. During training, the number of committed F_2 nodes, N_c grows. All committed F_2 nodes that pass the vigilance test for pattern **a** activate, and distribute a pattern of activity $\mathbf{y} = (y_1, y_2, ..., y_{N_c})$. Match tracking and learning are performed according to the relative activation over the "ensemble" E_K of F_2 nodes linked to the predicted F^{ab} node K. The relative activation over E_K is defined by the distributed pattern $\mathbf{y}^* = (y_1^*, y_2^*, ..., y_{N_c}^*)$, where $y_j^* = y_j / \sum_{l \in E_K} y_l$ only if $j \in E_K$, and $y_j^* = 0$ otherwise. Finally, the learning rate of category j is gradually decreased according to y_j^*/n_j. Table 5 highlights the main algorithmic differences between fuzzy ARTMAP and Gaussian ARTMAP.

A.2 Distributing and Biasing Test Set Activation. Simulation trials showed that the Q-max rule (Carpenter and Markuzon, 1998) for distributing F_2 layer activation in the ART-EMAP (Stage 1) and ARTMAP-IC classifiers gives better results than either power or threshold rules (Carpenter and Ross, 1995) with the radar data. The following choice of Q was found to give good results: $Q = \min\{\lceil N_c/2L \rceil, 2L\}$, where L is the number of classes (15 in this study) and N_c is the number of committed F_2 nodes. In particular, bounding Q to be below $2L$ reduces performance fluctuations.

Regardless of distributed activation, fuzzy ARTMAP performs better than its two extensions, ART-EMAP (Stage 1) and ARTMAP-IC, on the radar data; see Table 5. Distributed activation of the test patterns (in ART-EMAP, for example) yields more prediction errors because radar types in the data set can be dispersed, fragmented, and overlap one another. These data properties work against class predictions that are based on the distribution of strongly activated F_2 nodes among radar types, rather than on the most active F_2 node. Indeed, an F^{ab} class node k that receives a very strong activation from one F_2 node may have weaker overall activation than an F^{ab} class node h that receives a moderately strong activation from several F_2 nodes. Perhaps this explains why kNN gives its best performance for $k = 1$, and degrades slowly as k is increased. For example, its classification rate is 0.992 for $k = 9$ and $d_{\text{cityblock}}$.

Gaussian ARTMAP (Table 5) involves training and testing with a distributed pattern of activity. F_2 layer activation is distributed among nodes that pass the vigilance test. When training, each category $j \in E_K$ learns according to its relative activation for **a**. The learning rate of each category j is also gradually decreased as a function of n_j. F_2 nodes learn a Gaussian mixture model of the input space. Although computationally intensive, this learning strategy allows Gaussian ARTMAP to achieve high classification rates.

Table 5: Distinctive equations used by the ARTMAP neural networks in the comparison. ART-EMAP (Stage 1), ARTMAP-IC are extensions of fuzzy ARTMAP. With Gaussian ARTMAP, γ is the initial standard deviation assigned to newly-committed $F2$ nodes, and The scalar n_j accumulates the amount of relative activation obtained by F_2 node j on training set patterns.

Algorithmic step	**ARTMAP classifier**	
	fuzzy ARTMAP	Gaussian ARTMAP
1. Initialization:	$\alpha > 0$, $\beta \in [0,1]$	$\mu_J = \mathbf{A}$, $\sigma_{Ji} = \gamma\,(\gamma > 0)$, $w_{JK}^{ab} = 1$, $n_J = 1$
2. Input pattern coding:	$\mathbf{A} = (\mathbf{a}, \mathbf{a}^c)$	$\mathbf{A} = \mathbf{a}$
3. Prototype selection:		
– choice function	$T_j(\mathbf{A}) = \lvert\mathbf{A} \wedge \mathbf{w}_j\rvert / (\alpha + \lvert\mathbf{w}_j\rvert)$	$g_j(\mathbf{A}) = \begin{cases} \frac{n_j}{\prod_{i=1}^{M}\sigma_{ji}} G_j(\mathbf{A}) & \text{if } G_j(\mathbf{A}) > \rho \\ 0 & \text{otherwise} \end{cases}$
– vigilance test	$\lvert\mathbf{A} \wedge \mathbf{w}_J\rvert \geq \rho m$	$G_j(\mathbf{A}) = \exp\left\{-\frac{1}{2}\sum_{i=1}^{M}\frac{(A_i-\mu_{ji})^2}{\sigma_{ji}^2}\right\} > \rho$
– $F2$ activation	$y_j = 1$ only if $j = J$ (winning node)	$y_j = g_j / (0.01 + \sum_{l=1}^{N_c} g_l)$
4. Class prediction:		
– match tracking ($\varepsilon = 0^+$)	$\rho = (\lvert\mathbf{A} \wedge \mathbf{w}_J\rvert / m) + \varepsilon$	$\rho = \exp\left\{-\frac{1}{2}\sum_{j\in E_K} y_j^* \sum_{i=1}^{M}\frac{(A_i-\mu_{ji})^2}{\sigma_{ji}^2}\right\} + \varepsilon$
5. Learning:		$n_j' = n_j + y_j^*$
– prototype update	$\mathbf{w}_J' = \beta(\mathbf{A} \wedge \mathbf{w}_J) + (1-\beta)\mathbf{w}_J$	$\mu_j' = \frac{y_j^*}{n_j}\mathbf{A} + (1-\frac{y_j^*}{n_j})\mu_j$
– standard deviation update	N/A	$\sigma_j' = \sqrt{\frac{y_j^*}{n_j}(\mathbf{A}-\mu_j)^2 + (1-\frac{y_j^*}{n_j})\sigma_j^2}$

ARTMAP-IC and Gaussian ARTMAP accumulate weighting factors that depend on the quantity of training subset patterns assigned to each F_2 node. This frequency information is used to bias predictions towards classes assigned the most training patterns. This is a problem in radar ESM since some critical radars transmit very few pulses, while others transmit hundreds of thousands of pulses per second. Biasing prototype choices according to patterns in the TOA of pulses would, for instance, be more appropriate.

Fuzzy ARTMAP with MT- breaks ties (during prototype selection) by choosing the F_2 node with the smallest index. Even though it is not necessarily appropriate in our context, it may be useful on other data sets for fuzzy ARTMAP with MT- to use instance-weighted outputs only in the case where winning nodes are "inconsistent-case siblings," since in this case a basis on which to choose one of the winning nodes over another may be the frequency with which they were winning nodes during training. When using fuzzy ARTMAP with "limited instance counting," instances are still counted for all F_2 nodes. However, it distributes activation weighted by the instance counts if and only if nodes J are a set that code for the same test pattern but map to different classes. Limited IC does not harm accuracy on the radar data set (Granger et al., 1999b).

A.3 Prototype representations. Given the quantization of parameter measurements, intercepted radar pulses fall into resolution cells. The measurement uncertainty of the three parameters used (RF, PRI and PW) is uncorrelated, therefore the radar type definitions are essentially rectangular. ART-EMAP, ARTMAP-IC and fuzzy ARTMAP use hyperrectangles to represent prototypes in the input space, and appear to be a better match for this type of data. Gaussian ARTMAP and RBF, on the other hand, represent prototypes with Gaussian density functions. This results in substantial fragmentation of radar type classes, and in low compression for these two classifiers.