# SANDIA REPORT

SAND2007-6058

Unlimited Release

Printed September 2007

# Coordinated Machine Learning and Decision Support for Situation Awareness

Nathan Brannon, Greg Conrad, Tim Draelos, John Seiffertt, Don Wunsch, and Pengchu Zhang

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation,
a Lockheed Martin Company, for the United States Department of Energy's
National Nuclear Security Administration under Contract DE-AC04-94AL85000.

Approved for public release; further dissemination unlimited.

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.



Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.osti.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd.
Springfield, VA 22161

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.fedworld.gov
Online order: http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND2007-6058
Unlimited Release
Printed September 2007

# Coordinated Machine Learning and Decision Support for Situation Awareness

Nathan Brannon
Reliability Assessment and Human Systems Integration Department

Timothy Draelos
Cryptography and Information Systems Surety Department

Greg Conrad and Pengchu Zhang
Knowledge Discovery and Extraction Department

Sandia National Laboratories
P. O. Box 5800
Albuquerque, NM 87185-0672

John Seiffertt and Donald Wunsch
Department of Electrical & Computer Engineering
University of Missouri/Rolla
131 Emerson Electric Co. Hall
131 Miner Circle
Rolla, MO 65409-0040

Abstract

For applications such as force protection, an effective decision maker needs to maintain an unambiguous grasp of the environment. Opportunities exist to leverage computational mechanisms for the adaptive fusion of diverse information sources. The current research employs neural networks and Markov chains to process information from sources including sensors, weather data, and law enforcement. Furthermore, the system operator's input is used as a point of reference for the machine learning algorithms. More detailed features of the approach are provided, along with an example force protection scenario.

**CONTENTS**

**FIGURES**



**TABLES**

# 1 INTRODUCTION

The supervisory decision maker in force protection scenarios is responsible for the management of security. Many facilities exist that are designed to protect critical material and human assets. Unlike forward operating bases in the military encountering numerous and frequent threats, many facilities have a low probability of being intruded or attacked, yet the consequences of such unlikely events are great. Maintaining vigilance during long periods of little to no activity can be challenging. Further, if the decision maker becomes too disengaged with the state of the environment, a genuine threat such as physical intrusion, cyber intrusion, and even weapons of mass destruction can highlight the potential loss of situation awareness (SA). Consequently, the decision maker could take erroneous action or panic and lose time struggling to regain situation awareness.

Modern information sources to support force protection decision makers are diverse. Ground, air, and space-based sensors continue to increase in capability. Information fusion algorithms can help to integrate a variety of sensor data into meaningful forms [11]. However, finding a single learning algorithm to address real-world applications can be difficult. Our approach coordinates multiple learning mechanisms to accommodate environments where ground-truth and feedback may not be consistently available. Data from law enforcement, transportation systems, cyberspace, and intelligence offer additional perspectives on the state and capability of threats. Despite the significant and increasing capabilities of advanced computational systems, it has been noted that decision support systems leveraging such capabilities and resources have limited value without sensitivity to the decision maker and their objectives [17]. At the same time, areas such as Information Operations recognize that the ultimate target is the decision maker [2]. The current work has pursued technological capabilities for decision support with the decision maker as a central and balanced element of the system.

The current work blends three focus areas including machine learning, information fusion, and decision support. The machine learning components provide adaptive mechanisms for information fusion, situation assessment, and the ability to continuously sense and influence the state of the environment. Information fusion offers the ability to integrate diverse data sources into meaningful and actionable information. Finally, decision support addresses the need to facilitate objective, effective, and consistent human performance.

Information fusion and decision support combine to provide a situation awareness resource from which decision makers can benefit. The kinds of information used as input to the system determine the kinds of situations to which decision makers can respond. The collection, fusion, and decision support of cyber information can provide a Cyber SA capability for modern force protection network administrators. The current work focuses on physical protection of a facility through the use of electronic ground sensors and environmental information, but the principles employed can be applied to cyber and other SA concerns.

## *1.1 Machine Learning*

Machine learning involves programming computers to optimize a performance criterion using example data or past experience [1]. Artificial neural networks are commonly used in machine learning and utilize supervised, unsupervised, and reinforcement learning approaches to achieve predictive properties based on example (training) data. Unsupervised learning (clustering) can be effective when ground truth is not available with a dataset. An excellent survey of clustering techniques is provided in [25]. Supervised learning (learning with a teacher) provides a means to use experience (examples with ground-truth) for correctly classifying yet unseen situations. Reinforcement learning offers promise for machine learning in difficult learning environments by taking advantage of sometimes crude feedback about the performance of a system. Real-world problems are rarely optimally addressed using a single learning approach. Ideally, ground-truth information is available with all training data. In this case, supervised learning is used. However, this is too optimistic for many real situations. More often, data is available

without ground-truth, a rather pessimistic situation for machine learning algorithms supporting situation awareness. In this case, unsupervised learning can still be used to offer structure to the data for a human to interpret. Even without ground-truth information, hints or general feedback regarding a data set are often available, in which case reinforcement learning is used. The challenge addressed by the current work is to coordinate all of these learning mechanisms and utilize the appropriate one based on the information at hand.

Neural nets offer an excellent assortment of high-performance, low-cost, distributed processing options. In particular, they can be embedded into appropriate sensors for operation at the lowest levels of information fusion with effective but low-complexity designs. At the highest levels of information fusion and situation assessment, reinforcement learning can be used with a human in the loop to provide operational feedback. Dealing with multiple sensor modalities and extracting meaningful information from massive datasets is a natural fit for these adaptive methods. Although neural networks have been applied to sensor fusion, their use in situation awareness has been limited, possibly because of the lack of rich training data for this problem.

Automated (computational) information fusion continues to suffer from very specific, ad-hoc solutions (i.e., there appears to be very little general-purpose technology to apply to this problem) [14]. For many applications, there exists a dearth of data to use for training a computational engine. This reveals a challenge for the application of machine learning techniques, which are data-driven and require training, whether via supervised, unsupervised, or reinforcement learning. On the other hand, because they are data-driven, the advantage of machine learning techniques is that they can learn solutions to problems that are difficult for humans to codify with explicit rules or models. In other words, they can represent rules/decisions that are implicit in the training data.

## *1.2 Information Fusion*

The fusion of information has been likened to the human ability to utilize multiple senses to derive a better understanding of a situation [11]. For example, one may hear a noise and, based on the sound pressure discrepancy between each ear, localize the area of the sound source. Vision can then be used to further define and understand the source of the sound. The analogy is helpful because fusion, and more generally situation assessment, is a process rather than simply a discrete event. The process leads one from raw data to understanding and actionable knowledge. Fusion can occur over various information (sensor) modalities, over geographic space, and over time.

The sources of information potentially available to decision makers continue to expand in depth and breadth. Sensor capabilities in particular are maturing rapidly, but a valid concern is that the pace of sensor development has not necessarily been consistent with advances in human effectiveness, which the sensors must ultimately support [17]. Fusion algorithms will better support human-in-the-loop system effectiveness when the decision maker is a central and balanced design element [19].

Previous research has often utilized the Joint Directors of Laboratories (JDL) fusion levels [22] to explain how various input (e.g., seismic sensor data) can be used to provide certain output (e.g., human on foot). Applications described using the JDL model are typically military in nature but can be transferred easily to other domains as well. In JDL terms, the current research focuses on supporting levels 2 (“situation refinement”) and 3 (“threat assessment”).

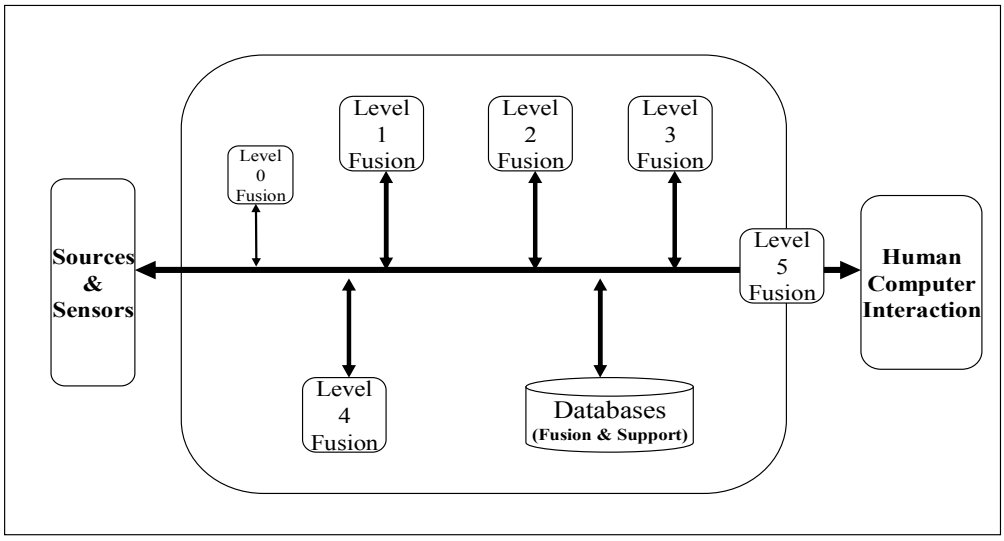


**Figure 1 The Data Fusion Model from the Joint Directors of Laboratories.**

In levels such as threat assessment, there are challenging issues to address, such as adversarial intent [20]. Inferring intent from raw data implies the filtering of some data during fusion. If too much data is filtered, the decision maker's understanding of the situation is inhibited, and, worse, trust in the automated resources can be lost [15]. The loss of trust can render the entire system ineffective. Fusion algorithms must therefore determine the optimum level of fusion necessary to avoid overloading the decision maker with data while keeping the decision maker sensitive to the context at hand.

## *1.3 Decision Support*

### *Addressing the Decision Maker*

Probably the greatest pitfalls in the development of decision support systems involve gross assumptions regarding who will use the system and how it will be used. The consequences of such assumptions include the late and painful confrontation of these assumptions only after significant design alternatives have been implemented. Designers must either return to earlier design phases or proceed at the expense of system effectiveness. As clearly and as early in the design process as possible, it is necessary first to understand who will be using the system. The user should not be considered some isolated entity, but rather a central component in the overall system.

The user is the greatest source of variability to a system. The user consists of many characteristics that can be divided into three broad categories including sensory (e.g., visual, auditory, tactile), physical (size, weight, strength), and cognitive (awareness, reasoning, memory, learning, decision-making). Each dimension brings with it capabilities and limitations that have often been empirically established and documented. Automated components are effective with structure and patterns and are ever vigilant. In contrast, humans are more agile, with the ability to change. In fact, variety is desirable in many cases. Furthermore, humans are capable of shaping the perception of others and detecting anomalies. General weaknesses include the need for regular rest, highly variable motivation, and limited vigilance. It is necessary to be aware of such capabilities and limitations when allocating functions to either the user or the hardware to facilitate human-in-the-loop system effectiveness.

Individual users of decision support systems introduce variation with respect to their backgrounds. Level and type of education will influence the nature of performance in a system. The level and type of education with decision support systems in particular will influence performance as well. The user may be accustomed to certain resources (e.g., information, automation) in reaching decisions or in general performance that could influence expectations and the acceptance of a new or alternative tool.

Developers must understand the nature of the user's task. What are they trying to accomplish? In the case of decision support systems, the user may need to plan, such as in the development of a defense perimeter. Digging deeper within planning, it is helpful to understand the constraints placed upon the decision maker, including factors like time constraints and resource availability. The user may be performing supervisory control, where issues such as the level of automation are relevant. Supervisory control is often tied to response activities in the event of, for example, a detected intrusion. Each task required of the user must be evaluated and designed such that the user is a central and balanced system element.

The complexity of individuals is significant and the complexity is magnified in the context of multiple/collaborative users. Careful analysis is needed to understand the dynamics of collaborative because each collaborator brings with them the variables already mentioned. Note also that individual differences not only vary person-to-person but also day-to-day, and it is cumbersome for many designers of high consequence systems to accept the fact that humans have bad days independent of training. Further, communication, authority, and shared awareness are a small sample of additional variables to assess in the decision support system design.

### *Decision Making*

A decision is a choice among alternatives. It is helpful to recognize that a choice does not necessarily have to result in some action that changes the environment (e.g., turning on a sensor). A decision can result in a hypothesis concerning a given situation or even an interim hypothesis in assessing a situation (e.g., the object may be a passenger car). A well-documented feature of human decision making in natural environments is that much of the process involves situation assessment [13]. Consequently, effective automated support can be directed towards situation assessment as much as decision support systems have traditionally supported the exploration of possible courses of action.

Engineered systems are commonly designed for normal and prescribed circumstances [24], but context drives human performance [21]. Therefore, unexpected events involving human error highlight the discrepancy between design assumptions and reality. Natural environments involve adaptive processes, again driven by circumstances or context [13]; therefore, advanced decision support systems must facilitate the adaptive processes of human-in-the-loop systems.

Adaptive processes observed in naturalistic decision making can be extremely effective. However, the utilization of incomplete information to draw conclusions can be costly as well [7].

# 2 Approach

In an attempt to effectively provide situation awareness for a decision maker, we have brought together three focus areas of research. An overview of these areas is depicted in Figure 2, where respective attributes and weaknesses are delineated. As noted earlier, much of our approach's success hinges on the effectiveness of the operator. If the approach is focused solely on the automated features, then the operator can become more disconnected from the tools and resources needed to assess situations and make objective and effective decisions. The functions allocated to automated processes leverage capabilities that humans cannot perform effectively. Still, the proposed approach does not control the course of action and allows the human to act at his discretion for specific contexts.

The fusion algorithms provide additional dimensions of perspective for a given situation and therefore generate rich input to the situation assessment algorithm. Machine learning algorithms are at the core of both sensor fusion and situation assessment. They provide analyses that improve with training and additional input. Their output facilitates the operator's trust in the automation [10] and provides a better grasp of a given situation. Fixed rules can augment these algorithms when relationships between inputs and outputs are already known.

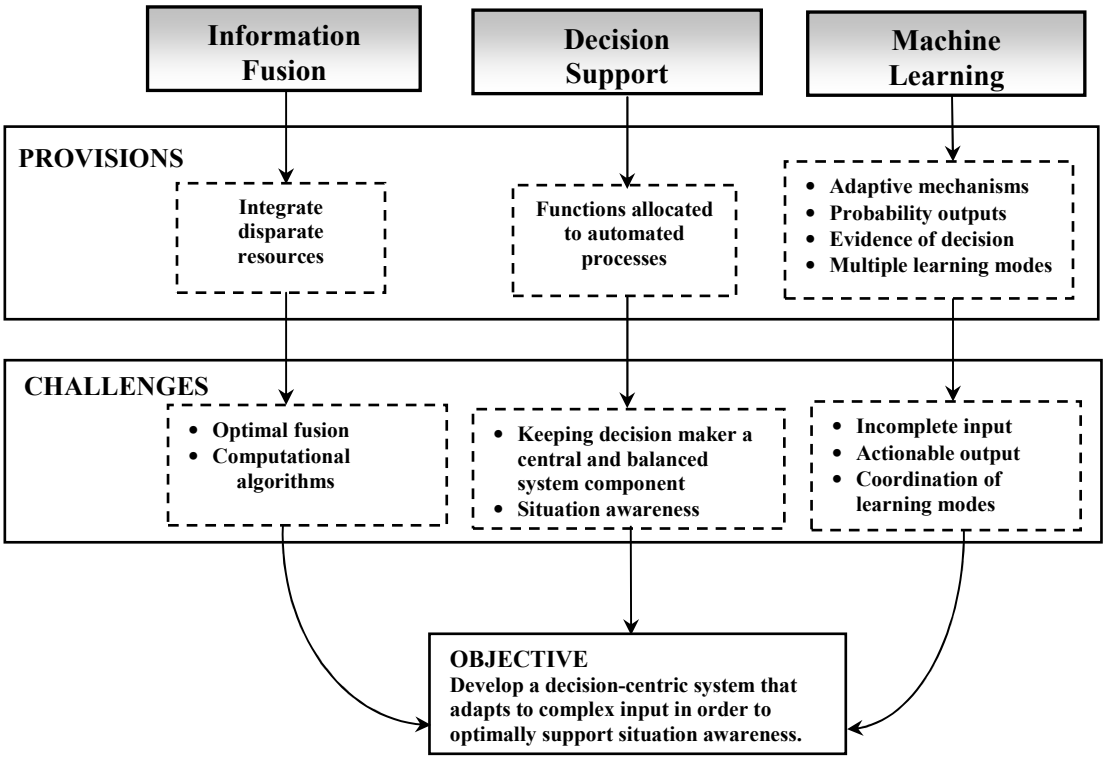


**Figure 2 Conceptual overview of situation awareness approach.**

## 2.1 System Architecture

This section provides the theoretical motivation for the design of the computational engine for information fusion and situation assessment, a design that takes advantage of the diverse utility of neural networks and integrates elements of supervised, unsupervised, and reinforcement learning. While this design is pushing the envelope of machine learning research, it matches extremely well the needs of situation awareness and human-in-the-loop decision support. Key design attributes of our system include the following.

- Accept diverse inputs – binary, categorical (integers), real-valued
- Output situation assessment with
    - Probability/confidence level,
    - Evidence in support of output, and
    - Evidence against output.
- Graceful performance degradation
- Missing/noisy inputs
- Incorporation of human feedback

In order to address the desired design attributes of our situation awareness system, neural networks are employed for information fusion, and a Markov Decision Process (MDP) is employed for situation assessment. The ARTMAP is based on Adaptive Resonance Theory (ART), a widely implemented approach to modeling the learning capabilities of the human brain. Architectures based on ART have been used successfully in a variety of areas requiring a self-organizing pattern recognition neural network [5]. The basic ART element supports unsupervised learning and binary inputs. Fuzzy ART is an extension to accommodate categorical and analog inputs. ARTMAP supports supervised learning and can

accommodate analog inputs using fuzzy logic. ARTMAP can also support reinforcement learning, for example, by adding a mechanism to implement actor-critic methods such as Q-Learning (discussed below in Section 2.2). Coordinated ARTMAP (CARTMAP) is the name given to the integration of all three learning mechanisms in the same architecture. Appendix A provides an excellent introduction to, and summary of, adaptive resonance theory (ART), the core machine learning element used for situation awareness in this paper. This material regarding ART is taken directly from Clustering book by Wunsch and Xu [26].

The situation assessment module of the system architecture is designed to use an MDP for sequential decision making. It receives state information from the information fusion module and possibly other sources as well and outputs a threat assessment or action to be taken. A Partially Observable MDP (POMDP) is a crucial element for assessing a situation when state information of the environment is not entirely available and must be supplemented with a probabilistic description.

The SA/Decision Support GUI takes information from all previous modules and presents it to the human decision maker(s) in meaningful and actionable ways. Human feedback can be accepted in various ways to support on going learning in the fusion and situation assessment modules.

Figure 3 shows the steps involved in situation awareness for a sample force protection application, which is described further in Section 3. The Information Sources block in Figure 3 represents a diversity of data collected from the world, including, but not limited to, electronic sensors. The Information Fusion block represents a data processing step that combines various information sources. Fusion of information occurs over time, space, and sensor modality. A CARTMAP network is used to map collected data to observations of events in the world. The Situation Assessment block takes observations from the Information Fusion function and provides an assessment of the current situation. Its output includes a confidence level and evidence in support of and against its assessment for presentation to the human decision maker. The SA / Decision Support GUI block represents an operator interface tailored to force protection decision making. Finally, the human operator can provide feedback to the system, providing a reinforcement signal for perpetual learning.

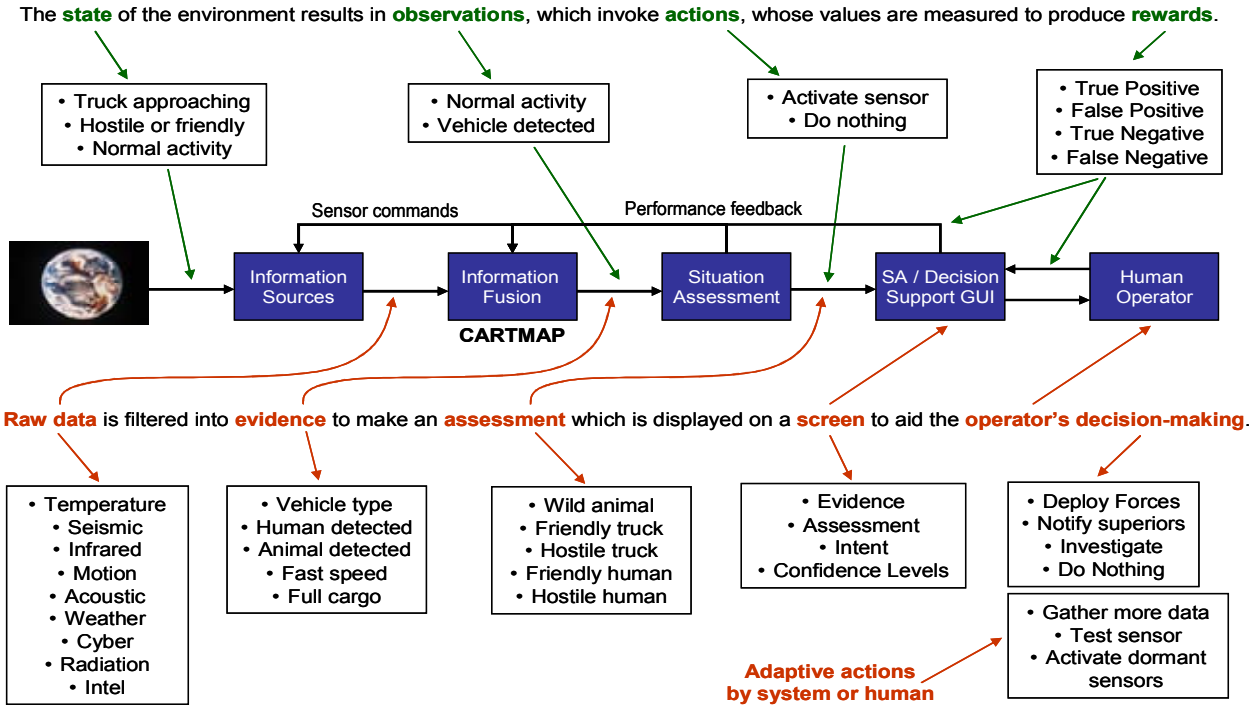


**Figure 3 Situation awareness architecture block diagram.**

## 2.2 Information Fusion Engine

Intelligent creatures exhibit an ability to switch seamlessly among unsupervised, supervised, and reinforcement learning as the needs arise. However, machine learning architectures, including artificial neural networks, have not yet achieved this goal. We believe that it is advantageous to develop this capability in a computational framework and that the ART architecture is an excellent choice for such an implementation. In the following text, we motivate, design, and give examples of this capability using smart sensors.

A well-designed sensor fusion algorithm, like an intelligent creature, can make informed use of all three types of learning in this environment on the data set given. Certain paths may be pre-trained prior to deployment, thus granting the human operators license to verify that the most obvious sensor patterns will be classified successfully. During operation, a reinforcement signal provided either by the environment or by the human operator acting off of the fusion algorithm's recommendations can adjust the current adaptive weight profile to curtail a faulty clustering in the ART algorithm. Finally, in the absence of any external signal, the algorithm will learn in an unsupervised manner, comparing current inputs to what it already knows.

This approach takes the form of an Adaptive Critic Design (ACD), a class of architectures designed to translate a reinforcement learning problem into a supervised learning problem. Our core algorithm—ARTMAP—already seamlessly handles both unsupervised and supervised learning problems. Therefore, the ACD super-structure is a natural fit, given our goals. In the core ACD paradigm, the function to be maximized is the Bellman Equation.

$$J(s) = r(s) + \gamma \sum_{s'} P(s', a) J(s', a)$$

This is the discounted expected reward optimality criterion. In this equation, *s* represents the current state of the system, *a* the action to be taken, J(s) represents the current value of a given state, s′ signifies the next-states, P(s′, a) is the transition probability matrix for the system's evolution, and a discount factor, $\gamma$, is applied to future rewards. This equation is to be maximized over all actions.

Narratively, the Bellman equation states that the current value of a state is equal to the immediate reward of taking an action plus the discounted future reward that accrues from that state. Other optimality criteria are possible to account for infinite horizon or nondiscounted models. The task at hand is to solve this equation given an appropriate reinforcement signal.

Barto and Sutton [23] discuss a wide variety of solution methods for these problems. Our algorithm will take advantage of one solution method in particular, a member of the temporal difference (TD)-lambda family of optimization algorithms called Q-learning. The Q-learning algorithm is presented in Figure 4. Q(s,a) is the valuation of each state-action pair, t is the iteration number, $\pi(s)$ is some method of calculating the next action (typically an e-greedy policy), $\delta$ and $\gamma$ are learning rates, a′ is the set of next actions, and s′ is the next state.

**Q-Learning Algorithm**

1. Initialize Q(s, a)
2. Set t = 1
3. Initialize s
4. Set a = $\pi$ (s), calculate s′
5. Update Q (s, a) = Q (s, a) + $\gamma$ [r (s′) + $\delta \max_{a'} Q(s', a') - Q(s, a)$]
6. Set s = s′
7. If s is not terminal, goto 4.
8. Increment t
9. If t is not equal to the maximum number of iterations, goto 3.

**Figure 4 Q-learning algorithm for Reinforcement Learning problems.**

Note that the Q-learning algorithm iteratively updates the value of each state-action pair. The appropriate modification is calculated based on the difference between the current and realized valuations, when maximized over all possible next actions. This is a key concept that sets the foundation for the more advanced techniques discussed in the following paragraphs.

This algorithm utilizes a lookup table to store the Q-values for each state-action pair. As the scale of the simulation grows, the amount of memory required to catalogue these values can grow at a staggering rate. A more computationally intensive version, called Heuristic Dynamic Programming, uses function approximators in place of the table. However, for our purposes in this architecture, the classic Q-learning approach will suffice.

With the ARTMAP unit taking the place of the Actor in the ACD implementation, the CARTMAP algorithm will behave in the following manner. Upon receipt of an unsupervised signal, the system will use its exemplar classification scheme to output an action choice, as usual. No updating of the lookup table will be necessary. When presented with a supervised signal, the internal adaptive weights will update appropriately, and the output action will be set equal to the supervised training signal. Finally, when a reinforcement learning input signal is received, it will be interpreted according to the Q-learning algorithm. The appropriate entry in the lookup table will be augmented with the new RL value, and the action selected will be the one with the most value accumulated in its column of the table. In our simulations, the values of the parameters delta and gamma are 0 and 1, respectively.

In summary, the information fusion engine accepts raw data from sensors and other information sources and processes/transforms/fuses them into inputs appropriate for the SA Assessment engine. It provides a mapping to a set of observations for the POMDP Situation Assessment Engine, as described in the next section.

Figure 5 illustrates the CARTMAP architecture, which uses three interdependent ART modules, one for each learning mode. The three ART networks are linked together by an inter-ART module (Associative Memory). One ART unit handles the inputs, another ART unit processes the supervisory (or target) signal, and the other processes the reinforcement signal as an adaptive critic. This architecture is capable of online learning without degrading previous input-target memories.

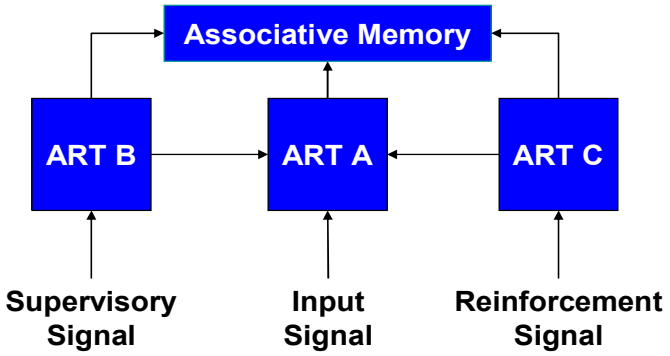


**Figure 5 Coordinated ARTMAP information fusion architecture.**

There are times when unsupervised learning is satisfactory, such as in the presentation of new input vectors to a pre-trained network. Supervised learning is appropriate and desired for initial training on fixed data. However, these two types of learning do not cover every possible complication. There are times when the correct classification is not known by the human operator, yet some feedback on the decision can be provided. These situations fall into the reinforcement learning category. One aspect of developing this information fusion engine, therefore, is adding the reinforcement learning capability to the ARTMAP neural network. Figure 6 shows the inputs and learning activity of the CARTMAP network for the three learning mechanisms as well as for standard operational use. Available inputs to the system are shown in green, as are the active elements involved in learning/use. The information fusion system will utilize appropriate elements of its architecture based on the data presented to it.

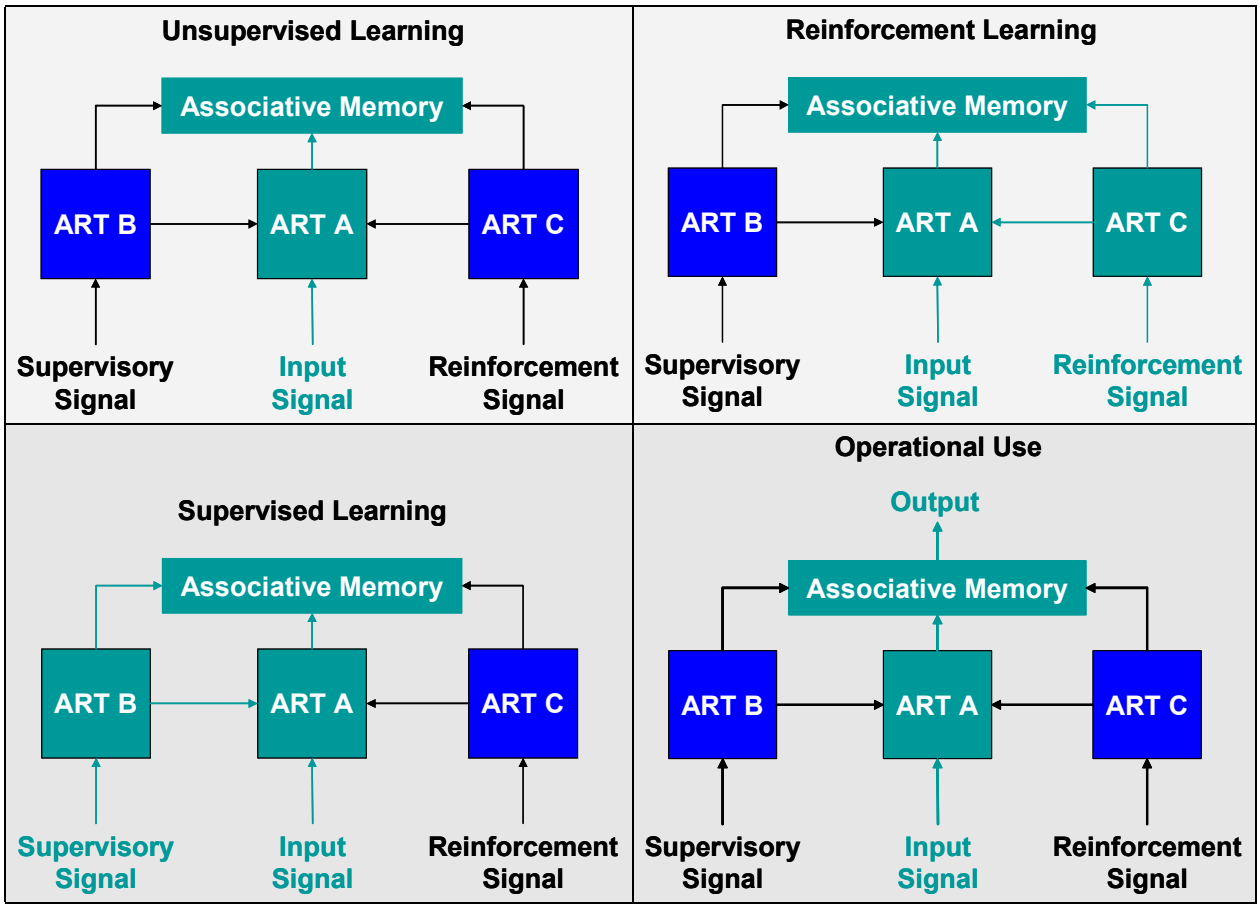


**Figure 6 CARTMAP input and system activity associated with different modes of use.**

It is the goal of any information fusion engine applied to this work to generate appropriate observations for the POMDP used in the situation assessment system. Whether this is accomplished using an advanced neural network architecture or using a simple clustering algorithm depends mostly on the particular problem statement at hand.

A practical consideration of this work includes the need to develop field-deployable hardware capable of performing intelligent sensor fusion quickly, efficiently, and with a minimum of overhead. Using a single architecture rather than a conglomeration of techniques hacked together in a computational sprawl will allow us to satisfy these requirements. Once trained, ART operates quickly and smoothly and is well-suited for operation on such a platform. The next section will describe how the information fusion engine's output is used in decision making.

## *2.3 Situation Assessment Engine*

A Markov chain is a mathematical object used in modeling the evolution of the states of systems for which all salient information for the future is embedded in the current state description (called the Markov property). The states of a board game such as chess or checkers, for example, embody all necessary information to analyze the game moving forward. Therefore, this system is said to have the Markov property and can be analyzed using the methods described below. In practice, real systems of interest often have error in the sensor measurements and therefore do not technically fulfill the Markov property definition. However, approximations can be made, and the use of Markov chain modeling can still be justified in many cases [18].

When Markov chains are used for sequential decision making, the resulting model is called a Markov Decision Process (MDP). An MDP consists of the relevant state information, a set of actions that can be taken at each state along with the accompanying reward or penalty, and transition probabilities, which detail the stochastic evolution of the system when a particular action is taken in a given state.

MDP's have been studied extensively and applied in such areas as inventory management [3], communications networks [18], and behavioral biology [12]. A solution to an MDP consists of a set of state-action pairs that gives the best formal policy according to some optimality criterion (typically one is looking for the set of actions that maximizes total expected reward). Standard solution techniques are available and well understood [18].

It is not always the case that a system can be expressed adequately as a vanilla MDP. Sometimes the state information is not entirely available, and the model must be supplemented with a probabilistic description of the current state. Called a belief space, this concept is the core idea of a Partially Observable Markov Decision Process (POMDP). A POMDP is essentially an MDP augmented with a set of observations providing probability information for the states, and it demonstrates a richer ability to model a variety of systems compared with an MDP. For example, POMDPs have been used in dynamic price modeling when the exact demand faced by the vendor is unknown [4]. The extensive utility of the POMDP framework is crucial for use in the situation awareness engine described in this work.

Standard solution methods for POMDPs work only on specific models and take massive amounts of computing power. To avoid these problems, it is common to use a technique such as a Bayesian Filter to transform a POMDP into an MDP once the observations are known. The solution techniques for MDPs can then be applied to the POMDP, and the optimal policy can be determined.

# 3 Application

We designed our situation awareness system to operate in an environment involving distributed sensors and a central collection site for protection of a facility, a typical anti-terrorism scenario. Information sources for our system might include seismic, magnetic, acoustic, passive infrared (PIR), and imaging sensors as well as weather, time/day information, various intelligence information, local/regional/federal

threat levels or law enforcement bulletins, and any other information that might be relevant to the security of a particular facility, such as current traffic situations and health issues.

Conditions of interest to force protection decision makers include no activity, severe weather, unauthorized people or vehicles in certain locations, and certain types of unauthorized vehicles or humans with weapons in any areas. Actions include doing nothing, identifying the type and location of a moving object (vehicle or human), commanding that sensors be turned on or off, deploying of forces, and/or notifying of higher authorities. The information sources can include binary data, such as motion detection, categorical data, such as the type of day (weekend, holiday, etc.), and digitized analog time-series data, such as seismic, acoustic, and magnetic energy levels.

Before being deployed, the system must be pre-trained with information that the human operator knows about the broader (e.g., force protection) system and the application environment. For example, if the data signature of a thunderstorm is easy to demonstrate (due to specific acoustic, magnetic, etc. levels), then that information can be included in the supervised training portion of the system. The information fusion engine will adaptively learn many more data-observation relationships during online operation, but having basic readings pre-trained will aid in initial operation.

When an intruder, be it an unauthorized vehicle or a human with a weapon, breaches the sensor range of a protected facility, the triggered sensor data stream into the information fusion engine. The CARTMAP network then maps these data into observations. These pairings represent novel data readings that were not anticipated, which are then categorized via the CARTMAP algorithm in relation to the pre-trained data.

The observation is then sent to the situation assessment engine, which follows the POMDP formulation to calculate a probability distribution over the state space. This information represents a confidence level that the system is in any given state. The state with the highest confidence from this calculation represents the system's choice for the current state. All this probability information is then passed to the human operator, who uses this evidence in making a final decision about how to respond to the situation.

Adapting online is an important element of the system and is accomplished through reinforcement signals that can be back-propagated through the system in two ways. First, if the probabilities of each state are too low, so that the human operator would not be able to distinguish the state from simple background noise, then the situation assessment engine may issue a command to gather more information from additional sensors. Second, the human operator may disagree with the system's assessment of the current state. A reinforcement signal is then sent to the information fusion engine, and the data-observation mappings will adapt online. Both of these reinforcement signal loops are noted functionally in the block diagram in Figure 3. This feature of the system allows it to maintain relevance in a changing environment.

It is an important point that the final action decisions are fully the domain of the human operator and are not automated. This system is an aid to the human; it is not a replacement.

The decision support graphical user interface (GUI) consists of three screens (see Figure 8, Figure 10, and Figure 11 in Section 3.2 for examples of these screens in an example force protection application). The center screen is primarily imagery (i.e., from cameras, photography augmented with graphics, and/or fully synthetic renderings) (see Figure 8). The second screen displays a log of temporal track data (see Figure 11). The log reflects temporal features, such as how long ago an unauthorized vehicle breached a sensor field and how soon another track might reach a key threshold (e.g., a fence or different sensor field). The third and most detailed screen provides track detail and assessment bases (see Figure 10).

The log screen and track detail screen borrow heavily from the Tactical Decision Making Under Stress (TADMUS) system [16]. The TADMUS system has similar motivations to the current research in that more content needs to be devoted to supporting an understanding of a given context. In both TADMUS

and our SA approach, less emphasis is placed upon supporting an appreciation of possible courses of action.

The track detail GUI provides typical track parameters such as the course and speed, but significant detail is provided with respect to the basis for assessment. Evidence in support of and against a given assessment is displayed. The machine learning algorithms share the evidence used to derive assessments with the operator. Such an approach provides greater transparency and allows the operator to interrogate assessments.

For the example scenario of an unauthorized vehicle, the assessment could be a "threat." Evidence in support of such an assessment includes sensor data, such as explosives detected, but also local law enforcement data that the license plate returns as a stolen vehicle. Evidence against the assessment could include a relatively slow speed and the use of the vehicle for construction when there has been ongoing construction activity. Alternative assessments are shown along with their respective evidence in support of or against.

The operator can investigate various assessments along with corresponding courses of action. For example, a patrol vehicle is in the vicinity of the unauthorized vehicle and could be directed closer to the possible threat. Further, more powerful sensors can be activated to generate additional points of reference and work towards higher levels of assessments such as possible intent.

## *3.1 Data Issues*

A data-driven approach to situation awareness requires data that can be used to design, implement, and/or test an actual system. While our machine-learning approach requires data to train and test the system, it can accommodate various kinds of data at various stages of system development. The ideal situation is to have plenty of rich data for supervised training. This implies ground-truth information regarding all input data, which is often not the case. In fact, it is quite rare in real-world problems. At times, no ground truth information is available with a set of data, in which case unsupervised learning can be utilized. A middle ground between these optimistic and pessimistic cases occurs when one has some information (hints) about the nature of the data, short of full ground-truth. In this case, reinforcement learning can be applied. Since our architecture employs all three learning modes (supervised, unsupervised, and reinforcement), it is well-suited to address real-world problems. However, data remains hard to acquire. We pursued several attempts at acquiring data for developing, training, and testing our SA system to demonstrate an SA capability to address a problem important to Sandia.

**Umbra** – Since quality training data is difficult to find for information fusion and situation awareness applications, one option is to generate your own data by simulating application scenarios. One advantage of this approach is that virtually any condition might be simulated, including those that can be difficult or impossible to create in reality. Another advantage is that simulations can be run as many times as necessary, each with slight variations if desired. This is also difficult to achieve in real-life exercises. Unfortunately, simulation capabilities for force protection applications are severely limited.

Umbra is a Sandia-developed Modeling and Simulation Framework that allows us to collect acoustic, magnetic, and seismic sensor data (energy levels and binary detections) during simulations of vehicle and human movement through a sensor field. Umbra provides a capability for gathering training data of simulated force protection scenarios. Figure 7 shows a map of the force protection scenarios executed with Umbra. The red dots are locations of sensor nodes grouped into three large and three small clusters. The blue dots are locations of a car as it traveled from the upper right of the map to the center. Dots are not removed, so all previous locations are shown. The larger the dot, the more recent in time an event occurred. The green and violet dots look like lines because they are adjacent dots representing locations of two different people who leave the car and walk away.

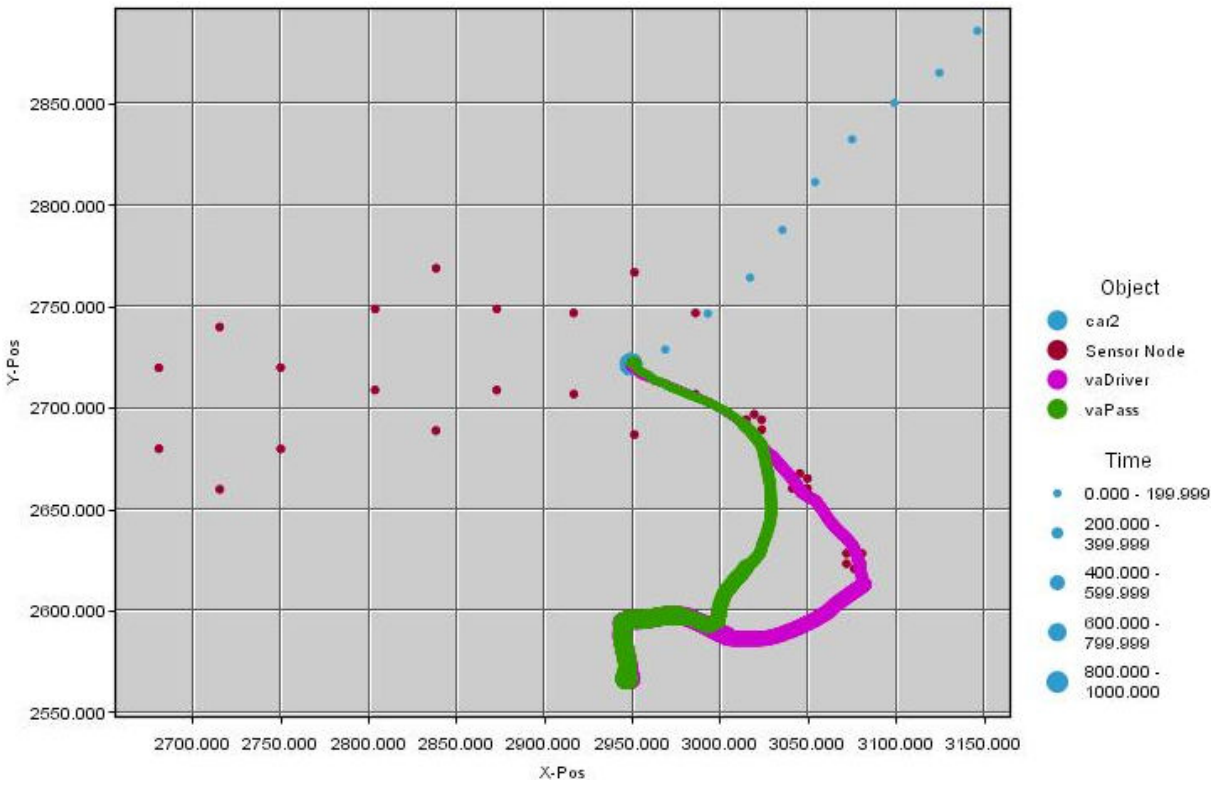


**Figure 7 Umbra simulation scenario presented as a map.**

The problems with this approach are that the simulated sensors do not accurately reflect the physics of actual sensors to the degree necessary. The other problem involves limitations in conditions (e.g., weather, vegetation, and animal). Therefore, we did not use Umbra to generate training data for our algorithms and to conduct virtual force protection experiments. As simulation capabilities increase, though, information fusion and situation awareness development may benefit greatly.

**Radiation Detection, DTRA TEAMS** – The detection of radioactive sources entering a compound is another force protection application that we considered addressing. We acquired data for detection of radiation sources and conducted a preliminary analysis of its use in our research. The data consisted of radiation measurements and pictures of vehicles. One can imagine a scenario in which the fusion of vehicle velocity measurements, rain detectors, radiation measurements, time, day, threat level, and other data allow a decision maker at a gate to arrive at a confident conclusion regarding threat of harmful radiation entering the compound under his protection. Real-time connection to medical databases (to assess nuclear medicine patients) and construction activity databases (to assess the need for soil density equipment) could further improve the quality of decision making.

**SNL protective forces** – We interacted with SNL Protective Forces personnel for the purposes of interviewing decision makers and acquiring sensor data for the development of our sensor fusion and situation awareness / decision support technology. Support of this application is important and would provide valuable experience for future military force protection applications. However, access to sensitive data and interruption of their activity make this application very difficult to complete.

**Coordination with VPED** – The Virtual Perimeter Extended Defenses (VPED) system (formerly VPS for Virtual Perimeter Security) is a Sandia force protection project that employs an array of sensors (seismic, microwave, magnetic, acoustic, passive infrared) to detect intrusions into access-restricted areas. We discussed integrating our information fusion and situation awareness algorithms into their system. However, their upcoming deployments made interruption to support our integration problematic.

Nonetheless, our algorithms and approach are very complementary with their sensor fusion needs. Importantly, the availability of ground truth information is limited, making unsupervised and reinforcement learning necessary elements of a machine learning system.

**University of Wisconsin (UW) SensIT data** – A well-documented, highly usable set of DARPA SensIT Program data is held in a repository at the University of Wisconsin (UW). Although somewhat limited, the data collection experiment was well controlled, executed, and documented. It is a good data set to use to develop and test sensor fusion methods on realistic force protection problems. The UW data contains acoustic and seismic sensor readings from 23 sensor nodes distributed along three roads during vehicle tracking exercises at Twenty-Nine Palms. We used this data set to conduct our SA experiments.

## *3.2 Vehicle Tracking*

The primary application of the developed situation awareness technology involved tracking of vehicles in the vicinity of a facility under force protection. A data set suitable for testing and demonstrating our technology was collected during a DARPA SensIT program in November, 2001 at Twenty-Nine Palms, CA and exists at the University of Wisconsin (UW) [9]. The data set consists of raw time series (acoustic and seismic) and binary detection decisions from 23 sensor nodes distributed along three intersecting roads as one of two vehicles travels along a road. Figure 8 includes a map illustrating the force protection scenario, with a fence line and an Entry Control Point (ECP) providing protection for a facility on the North Road. The two vehicles used in the scenario are a light armored vehicle (AAV) and a heavier, tracked transport vehicle (DW). We developed a scenario whereby a facility under protection is assumed to exist along one of the roads, and binary sensor data processed by our fusion and situation assessment algorithms are used to inform a human decision maker with our decision support techniques.


**Figure 8 Vehicle tracking scenario map. Blue dots are seismic/acoustic sensor nodes.**

### *Data Analysis*

In preparation for testing our own sensor fusion and situation assessment algorithms, an initial analysis was performed on the UW data set to determine its value in predicting the location of the AAV vehicle during a run (travel along a road). The Clementine data mining work bench from SPSS was used to accumulate, prepare, and analyze the data.

The geographic vehicle test site was divided into a 10 x 10 grid (100 distinct cells). Using ground truth data, the grid location of the vehicle was calculated for each time step in each run. The location was then correlated with the sensor node decision report (0 – no detection; 1 – detection) for each sensor node for each time step. Additionally, the sensor decision reports for the previous four time steps for each sensor node were included at each time step. Therefore, the input (feature) vector consists of 23 x 5 = 115 sensor decision spanning a 5-step time window. With the data thus organized for each AAV run, all of the AAV runs were appended into a single data set with each record being designated as either a training or a testing record.

Clementine has five standard predictive modeling algorithms available (Neural Net, C5.0, C&R Tree, Quest, and Chaid). Each algorithm was used with its default parameterization to provide a quick indication of the ability to predict vehicle location based on sensor decisions over multiple time steps. Two experiments were performed: one attempted to predict the grid location from the 23 current sensor node decision reports, and the other used the current node reports and the node reports from the four previous time steps. Tables 1 and 2 show the results of this experiment.

**Table 1 Comparison of algorithm performance – Current sensor node decision reports only.**

| Algorithm | 'Partition' | Train | | Test | |
|---|---|---|---|---|---|
| Neural Net | Correct | 340 | 48.43% | 291 | 46.12% |
| | Wrong | 362 | 51.57% | 340 | 53.88% |
| C 5.0 | Correct | 357 | 50.85% | 312 | 49.45% |
| | Wrong | 345 | 49.15% | 319 | 50.55% |
| CRT | Correct | 333 | 47.44% | 288 | 45.64% |
| | Wrong | 369 | 52.56% | 343 | 54.36% |
| Quest | Correct | 311 | 44.3% | 263 | 41.68% |
| | Wrong | 391 | 55.7% | 368 | 58.32% |
| Chaid | Correct | 317 | 45.16% | 257 | 40.73% |
| | Wrong | 385 | 54.84% | 374 | 59.27% |

**Table 2 Comparison of algorithm performance – Current and previous 4 node reports**

| Algorithm | 'Partition' | Train | | Test | |
|---|---|---|---|---|---|
| Neural Net | Correct | 357 | 50.85% | 312 | 49.45% |
| | Wrong | 345 | 49.15% | 319 | 50.55% |
| C 5.0 | Correct | 382 | 54.42% | 323 | 51.19% |
| | Wrong | 320 | 45.58% | 308 | 48.81% |
| CRT | Correct | 337 | 48.01% | 276 | 43.74% |
| | Wrong | 365 | 51.99% | 355 | 56.26% |
| Quest | Correct | 312 | 44.44% | 263 | 41.68% |
| | Wrong | 390 | 55.56% | 368 | 58.32% |
| Chaid | Correct | 328 | 46.72% | 269 | 42.63% |
| | Wrong | 374 | 53.28% | 362 | 57.37% |

These results suggest that there is an improvement across all algorithms when using historical sensor node reports. However, even at its best (the results from C 5.0), the prediction barely represents an improvement over a coin toss. Extending the experiment produced an interesting synergy when using two algorithms in tandem. We used the two best performing algorithms, the C 5.0 and the Neural Net, in tandem. Tables 3 and 4 show that when the algorithms are used in tandem and when they agree, the result is a highly accurate prediction.

**Table 3 Algorithm agreement matrix – C 5.0 and Neural Net**

| 'Partition' | Train | | Test | |
|---|---|---|---|---|
| Agree | 288 | 41.03% | 247 | 39.14% |
| Disagree | 414 | 58.97% | 384 | 60.86% |
| Total | 702 | | 631 | |

**Table 4 Algorithm agreement accuracy - C 5.0 and Neural Net**

| 'Partition' | Train | | Test | |
|---|---|---|---|---|
| Correct | 269 | 93.4% | 232 | 93.93% |
| Wrong | 19 | 6.6% | 15 | 6.07% |
| Total | 288 | | 247 | |

### *Force Protection Experiments*

In order to demonstrate the capabilities of our situation awareness system, neural networks were trained to perform sensor fusion, a situation assessment formula was constructed/calculated, and a GUI was developed, all to increase the awareness of a human decision maker of the situation around that facility under his protection. The scenario consists of a virtual checkpoint partway up the north road on the way to a sensitive facility (see Figure 8) with 23 sensor nodes scattered along three intersecting roads. Each sensor node outputs a binary detection decision at fixed time intervals (0.75 seconds in the original test set). The sensor detections derive from seismic, acoustic, and passive infrared energy levels. There are two different vehicles (AAV and DW) that make runs from one end of a road through the intersection to the end of another road. The total number of runs is 40. The direction of half (20) of the runs was artificially reversed to create additional data sets. It is plausible that the information is accurately represented in these runs because the data is based on binary decisions and the ground is relatively flat so that the engine speed and noise are presumably similar in both directions.

A prominent piece of information that a decision maker wants to know is the current threat level around his facility. The threat level is a function of the location, speed, heading, and type of vehicle detected by the sensor array and other variables that are independent of the sensor array, such as DHS advisory level, wind speed, average batter level of the sensors, time of day, and day of week.

The system used to produce the threat level is illustrated in Figure 9. The system consists of three modules: 1) Information Fusion, 2) Situation Assessment, and 3) a Graphical User Interface (GUI) focused on human decision makers in force protection applications. Multiple time steps of binary sensor data serve as input to the Information Fusion module, which implements the Coordinated ARTMAP algorithm. This introduces an element of relative time, which is a necessary component in estimating speed and heading. The output from the Fusion module consists of vehicle type, speed, location, and heading, each with a corresponding confidence level, and will serve as input to the Situation Assessment module. The Assessment module uses outputs from the Fusion module as well as other information, and, being designed to represent the conditions under which a Threat is defined, outputs an overall Threat

Level. The output of the assessment module will feed the graphical user interface (GUI) with a threat level (low, medium, high), an associated confidence level, a suggested response, and evidence in support of or against its output. The GUI will also have access to the output from the fusion module, maps, and other available data, such as time, date, and environmental data.

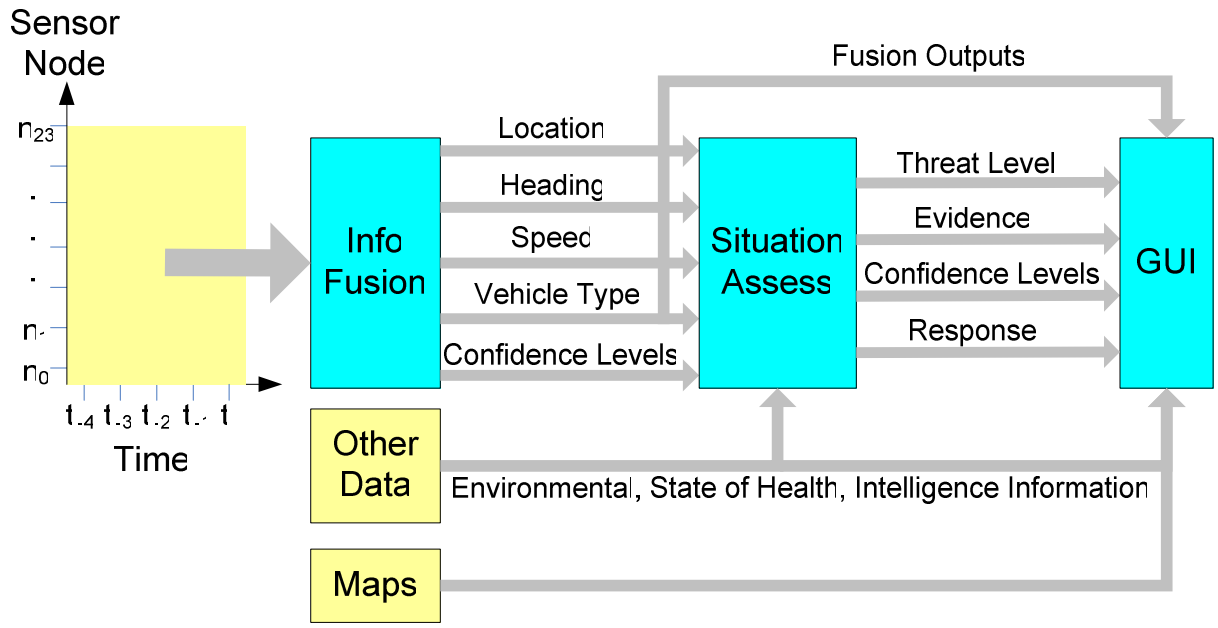


**Figure 9 Force protection experiment using UW vehicle data.**

# 4 Force Protection Experiment Results

Section 3.2 introduced a force protection experiment involving a vehicle traveling along roads near a facility with restricted access. The highest level output of the situation awareness system is a threat level, which is defined in the experimental scenario to be a function of all of the available information regarding the environment. Some of this information is relatively static, such as DHS advisory levels, and some information is a real-time estimate of a vehicle attribute, such as its speed. Conditions of a high threat include a vehicle at high speed, in a location near the facility, of a certain type (heavier vehicles are more worrisome than lighter ones), heading north towards the facility. Other conditions contributing to a high threat level are a high DHS advisory level, off-hours times and days, and extreme environmental conditions. Experimental results of estimating vehicles' attributes with the Information Fusion module are presented in Section 4.1. Two approaches to situation assessment are presented in Section 4.2. Finally, decision support GUIs are presented in Section 4.3 for this realistic scenario.

## *4.1 Information Fusion Module Results*

The fusion module consists of four different CARTMAP networks, one for each fusion output (location, heading, speed, and vehicle type). The output of a network will be of a categorical type or class except for the confidence levels, which will be a real number. Table 5 presents the classes for each information fusion network. Note that for each network, if the input is all zeros, the output will be zero by virtue of a simple rule (i.e., no learning is involved).

**Table 5 List of classes and class values for each of the information fusion outputs.**

| Vehicle Type Classes | Location Classes | Heading Classes | Speed Classes |
|---|---|---|---|
| 0: zero input<br>1: AAV<br>2: DW | 0: zero input<br>1: West Road<br>2: North Road<br>3: East Road<br>4: Intersection | 0: zero input<br>11: N<br>14: NE<br>13: E<br>8: SE<br>4: S<br>1: SW<br>2: W<br>7: NW | 0: zero input<br>1: < 10 km/hr<br>2: 10-20 km/hr<br>3: 20-30 km/hr<br>4: 30-40 km/hr<br>5: 40-50 km/hr<br>6: 50-60 km/hr<br>7: 60-70 km/hr<br>8: 70-80 km/hr<br>9: 80-90 km/hr<br>10: > 90 km/hr |

Out of the 40 total runs available for the force protection experiments, 30% were used for testing and the remainder for training. Table 6 show the number of runs used in the six experiments. In real-world applications, it is expected that the amount of supervised training data is limited. In the force protection experiments, only 2 of the 30 training runs are used for supervised learning.

**Table 6 Distribution of vehicle runs for the force protection experiments.**

| Experiment # | # Supervised Runs | # Unsupervised Runs | # Reinforcement Runs | # Test Runs |
|---|---|---|---|---|
| 1, 4 | 2 | 26 | 0 | 12 |
| 2, 5 | 2 | 13 | 13 | 12 |
| 3, 6 | 2 | 0 | 26 | 12 |

Experiments 1-3 use the same runs as Experiments 4-6, but the order of training is reversed. In Experiments 1-3, supervised learning is conducted first, followed by reinforcement learning, and finally unsupervised learning. Experiments 4-6 use the opposite order of learning, using the data with the least amount of information first and finishing with supervised learning, which utilizes training data with the most amount of information. In this case, one expects the richer data sets and training modes to correct errors and refine the classification performance of previous learning modes.

For each force protection experiment conducted, the same test set was used, consisting of 12 runs with 1755 input/output pairs. The performance (% correct classification) was computed based on this test set. For some sensor modes, such as speed and heading, a classification error may not necessarily indicate poor performance. For example, if the ground truth heading of a vehicle is North and the fusion module output is Northeast, it would be counted as a classification error even though the output is quite satisfactory. The same vigilance value of 0.89 was used for all experiments. Results from experiments conducted using various combinations of learning modes for each of the information fusion outputs are presented in Table 7-Table 10. In the Classified Correct (%) column of the tables, there are three numbers separated by colons. These numbers are explained in the three bullets below in the order of their presentation in the table (e.g., 1 : 2 : 3).

1. These numbers represent the percentage of test samples that have a target value that exactly matches the output value from a CARTMAP network.
2. These numbers represent the percentage of test samples that have a target value that exactly or partially matches the output value from a CARTMAP network. An exact match increments the total number of correct classifications by 1, whereas a partial match increases the number by 0.5.

Partial matches are possible only with the Heading and Speed networks, where the class adjacent to the target class is considered a partial match. For example if the target class is N, then a network output of NW or NE would result in a partial match. Note that for the Vehicle Type and Location networks, there are no partial matches so the first and second numbers in the Classified Correct should be the same.

3. These numbers represent correct classification percentages of networks which have had two passes through the training set. During the first pass, the reinforcement lookup table is updated during reinforcement learning. The updated table may be an advantage for second pass unsupervised and reinforcement learning. Correct classification percentages are computed using partial matches.

Since 54.4% of the input patterns in the test set are all zeros, correct classification percentages should never be less than 54.4%.

**Table 7 Force protection experiment results for Vehicle Type.**

| Experiment # | Learning Mode | Training Samples | # Categories | Classified Correct (%) |
|---|---|---|---|---|
| 1 | Supervised<br>Unsupervised | 462<br>3304 | 19<br>79 | 82.2 : 82.2 : 82.2<br>77.8 : 77.8 : 77.8 |
| 2 | Supervised<br>Reinforcement<br>Unsupervised | 462<br>1782<br>1582 | 54<br>57<br>58 | 82.2 : 82.2 : 82.2<br>79.8 : 79.8 : 88.1<br>79.9 : 79.9 : 85.9 |
| 3 | Supervised<br>Reinforcement | 462<br>3304 | 54<br>57 | 82.2 : 82.2 : 82.2<br>82.6 : 82.6 : 85.6 |
| 4 | Unsupervised<br>Supervised | 3304<br>462 | 53<br>99 | 55.0 : 55.0 : 55.0<br>63.5 : 63.5 : 63.5 |
| 5 | Unsupervised<br>Reinforcement<br>Supervised | 1582<br>1782<br>462 | 37<br>40<br>89 | 55.0 : 55.0: 64.8<br>62.4 : 62.4: 83.6<br>66.7 : 66.7: 84.3 |
| 6 | Reinforcement<br>Supervised | 3304<br>462 | 10<br>58 | 84.2 : 84.2 : 86.1<br>85.2 : 85.2 : 88.5 |

**Table 8 Force protection experiment results for vehicle Location.**

| Experiment # | Learning Mode | Training Samples | # Categories | Classified Correct (%) |
|---|---|---|---|---|
| 1 | Supervised<br>Unsupervised | 462<br>3304 | 23<br>63 | 95.4 : 95.4 : 95.4<br>93.8 : 93.8 : 93.8 |
| 2 | Supervised<br>Reinforcement<br>Unsupervised | 462<br>1928<br>1673 | 23<br>100<br>129 | 95.4 : 95.4 : 95.4<br>95.4 : 95.4 : 96.2<br>93.6 : 93.6 : 95.7 |
| 3 | Supervised<br>Reinforcement | 462<br>3304 | 23<br>179 | 95.4 : 95.4 : 95.4<br>95.1 : 95.1 : 90.4 |
| 4 | Unsupervised<br>Supervised | 3304<br>462 | 68<br>90 | 54.4 : 54.4 : 54.4<br>71.7 : 71.7 : 71.7 |
| 5 | Unsupervised<br>Reinforcement<br>Supervised | 1673<br>1928<br>462 | 31<br>227<br>249 | 54.4 : 54.4 : 54.5<br>54.4 : 54.4 : 63.1<br>93.7 : 93.7 : 62.2 |
| 6 | Reinforcement<br>Supervised | 3304<br>462 | 380<br>402 | 54.4 : 54.4 : 84.3<br>91.9 : 91.9 : 85.0 |

**Table 9 Force protection experiment results for vehicle Heading.**

| Experiment # | Learning Mode | Training Samples | # Categories | Classified Correct (%) |
|---|---|---|---|---|
| 1 | Supervised<br>Unsupervised | 462<br>3304 | 49<br>78 | 68.2 : 69.2 : 69.2<br>65.5 : 66.9 : 66.9 |
| 2 | Supervised<br>Reinforcement<br>Unsupervised | 462<br>1782<br>1582 | 49<br>55<br>55 | 68.2 : 69.2 : 69.2<br>60.9 : 65.7 : 64.8<br>61.0 : 65.8 : 63.5 |
| 3 | Supervised<br>Reinforcement | 462<br>3304 | 49<br>60 | 68.2 : 69.2 : 69.2<br>60.1 : 63.1 : 64.6 |
| 4 | Unsupervised<br>Supervised | 3304<br>462 | 53<br>98 | 54.4 : 54.4 : 54.4<br>60.7 : 60.8 : 60.8 |
| 5 | Unsupervised<br>Reinforcement<br>Supervised | 1522<br>1782<br>462 | 31<br>37<br>80 | 54.4 : 54.4 : 67.2<br>56.5 : 57.7 : 62.7<br>58.0 : 59.1 : 63.9 |
| 6 | Reinforcement<br>Supervised | 3304<br>462 | 15<br>56 | 61.0 : 67.2 : 65.4<br>61.9 : 67.8 : 65.7 |

**Table 10 Force protection experiment results for vehicle Speed.**

| Experiment # | Learning Mode | Training Samples | # Categories | Classified Correct (%) |
|---|---|---|---|---|
| 1 | Supervised<br>Unsupervised | 462<br>3304 | 55<br>90 | 69.7 : 78.7 : 78.7<br>71.2 : 78.5 : 78.5 |
| 2 | Supervised<br>Reinforcement<br>Unsupervised | 462<br>1782<br>1582 | 55<br>57<br>57 | 69.7 : 78.7 : 78.7<br>65.8 : 77.7 : 71.7<br>66.0 : 77.8 : 74.6 |
| 3 | Supervised<br>Reinforcement | 462<br>3304 | 55<br>58 | 69.7 : 78.7 : 78.7<br>69.7 : 78.1 : 76.1 |
| 4 | Unsupervised<br>Supervised | 3304<br>462 | 51<br>100 | 55.0 : 55.0 : 55.0<br>58.6 : 60.4 : 60.4 |
| 5 | Unsupervised<br>Reinforcement<br>Supervised | 1582<br>1782<br>462 | 27<br>32<br>79 | 55.0 : 55.0 : 55.6<br>63.5 : 67.0 : 69.0<br>63.9 : 67.6 : 68.6 |
| 6 | Reinforcement<br>Supervised | 3304<br>462 | 13<br>58 | 71.3 : 78.8 : 75.7<br>71.0 : 78.5 : 77.3 |

The following notes may be helpful in interpreting the classification performance results presented in Table 7-Table 10 (UL – Unsupervised Learning, RL – Reinforcement Learning, SL – Supervised Learning).

- It is expected that performance will improve when the vigilance parameter is optimized for the type of fusion mode and the type of learning. The vigilance for modes that have very distinct classes that don't change much, such as vehicle type and location, may need to be different than for speed and heading, which have more classes and change more frequently throughout a run. In addition, using a different vigilance value for each of the learning modes is expected to improve performance.

- Since 54.4% of the input patterns are all zeros, if a correct classification percentage of greater than 54.4% is achieved after UL alone, then the reinforcement lookup table must have been used to correctly label some patterns.
- During reinforcement learning, better results (higher classification percentages) were achieved when retraining was performed only when the reinforcement signal was negative. So, the following steps are taken during reinforcement learning.
    - When a positive reinforcement signal is received, the LUT is updated, but no training is performed.
    - When a negative reinforcement signal is received, the LUT is updated, and
        - Supervised learning is performed if the input pattern is found in the LUT (the action associated with the input pattern with the highest value is used as the target).
    - Unsupervised learning is performed if the input pattern is not found in the LUT regardless of the value of the reinforcement signal.
- RL followed by SL is far superior to UL followed by SL. This stands to reason since RL brings more information about potential class labels than UL. When UL comes after SL, no class label information is explicitly available. Therefore, any new ART categories that are created will be left without a class assignment and these categories will not contribute to test performance. However, unsupervised input patterns that get encoded by existing categories with a class label can contribute to the quality of the category in representing the class in feature space. In addition, since the CARTMAP has access to a reinforcement lookup table (RLUT), if an unlabeled pattern is found in the RLUT during UL, then its class label can be assigned to the pattern. Originally, the RLUT is generated from the supervised training data and it expands when new patterns with labels are added to the RLUT.
- In general, user control of the machine learning process (e.g., setting the vigilance parameter) should maintain the value of SL as much as possible. In other words, once SL has been performed, UL and RL should not degrade, but improve the quality of the network.

The coordination of three machine learning modes offers potential benefit from every sample of data available in an application. However, the details of their integration are non-trivial.

## *4.2 Situation Assessment Module*

The situation assessment module takes as input the information from the Information Fusion module and any other information relevant to the evaluation of the situation in the current environment. The inputs used in our vehicle tracking scenario are given below.

- Vehicle type, location, heading, and speed
- Wind speed
- DHS Advisory level
- Average battery life of sensor modules
- Day of week
- Time of day

The situation assessment module provides the highest level information about the situation to the human decision maker as well as meta-information about its assessment. Its outputs include the following information.

- Threat Level (high, moderate, or low)
- Evidence in support of the threat level
- Evidence against the threat level
- Confidence in the threat level
- Suggested response(s) to the threat level

### *Situation Assessment using a Weighted Rule*

The weighted rule approach to situation assessment first transforms each input into a threat category according to Table 11.

**Table 11 Threat categories of inputs used in Weighted Rule situation assessment.**

| Input | Range | Category | Numeric Value |
|---|---|---|---|
| Vehicle Location (L) | North Road<br>Intersection<br>East / West Road | High<br>Moderate<br>Low | 2<br>1<br>0 |
| Vehicle Speed (S) | > 40 km/hr<br>20 – 40 km/hr<br>< 20 km/hr | High<br>Moderate<br>Low | 2<br>1<br>0 |
| Vehicle Heading (H) | NW, N, NE<br>W, E<br>SW, S, SE | High<br>Moderate<br>Low | 2<br>1<br>0 |
| Vehicle Type (V) | Tracked (DW)<br>Light (AAV)<br>Anomaly | High<br>Moderate<br>Low | 2<br>1<br>0 |
| Wind Speed (W) | > 40 km/hr<br>≤ 25 km/hr | Moderate<br>Low | 1<br>0 |
| Battery Capacity (B) | ≤ 50%<br>> 50% | Moderate<br>Low | 1<br>0 |
| DHS Advisory Level (D) | High, Severe<br>Elevated, Guarded<br>Low | High<br>Moderate<br>Low | 2<br>1<br>0 |
| Time of Day (T) | Off hours<br>Normal hours | Moderate<br>Low | 1<br>0 |
| Day of Week (DW) | Weekend, Holiday, Special Day<br>Normal weekday | Moderate<br>Low | 1<br>0 |

The next step is to compute the assessed threat level from a linear combination of all of the input categories, weighted according to their relative importance.

Weighted Rule: Threat Index = 5L + 4S + 3H + 2V + W + B + D + T + DW

The Threat Index is then converted to a Threat Level, to be presented to the decision maker.

Threat Index > 22: High Threat

11 ≤ Threat Index ≤ 22: Moderate Threat

Threat Index < 11: Low Threat

### *Situation Assessment using a Bayesian Filter*

Standard solution methods for POMDPs work only on specific models and take massive amounts of computing power. To avoid these problems, it is common to use a technique such as a Bayesian Filter to transform a POMDP into an MDP once the observations are known. The solution techniques for MDPs can then be applied to the POMDP and the optimal policy can be determined.

Our approach leverages the laws of conditional probability given by

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)},$$

where A and B are events of interest. We generate reasonable estimates of the various probabilities, given expectations about the environment and interactions. For example, quantities such as the probability that a vehicle is a threat, given that it is moving at a certain speed is set 'a priori'. Our method runs the above calculation for each of the four "state" calculations and then selects the maximum of that set. Further research may upgrade the efficacy of the 'a priori' estimates while the system runs online.

The weighted rule formula used in the previous section can be used to establish initial conditional probabilities for the Bayesian Filter. These are shown in Table 12.

**Table 12 Conditional probabilities situation assessment used in the Bayesian Filter.**

| | Loc=High | Loc=Mod | Loc=Low |
|---|---|---|---|
| P(Threat = High \| Loc = *) | 0.779084967 | 0.213071895 | 0.007843137 |
| P(Threat = Mod \| Loc = *) | 0.294317218 | 0.411365564 | 0.294317218 |
| P(Threat = Low \| Loc = *) | 0.007843137 | 0.213071895 | 0.779084967 |
| | | | |
| | Speed=High | Speed=Mod | Speed=Low |
| P(Threat = High \| Speed = *) | 0.675816993 | 0.274509804 | 0.049673203 |
| P(Threat = Mod \| Speed = *) | 0.314249364 | 0.371501272 | 0.314249364 |
| P(Threat = Low \| Speed = *) | 0.049673203 | 0.274509804 | 0.675816993 |
| | | | |
| | Heading=High | Heading=Mod | Heading=Low |
| P(Threat = High \| Heading = *) | 0.57254902 | 0.308496732 | 0.118954248 |
| P(Threat = Mod \| Heading = *) | 0.325275657 | 0.349448685 | 0.325275657 |
| P(Threat = Low \| Heading = *) | 0.118954248 | 0.308496732 | 0.57254902 |
| | | | |
| | Type=High | Type=Mod | Type=Low |
| P(Threat = High \| Type = *) | 0.488888889 | 0.321568627 | 0.189542484 |
| P(Threat = Mod \| Type = *) | 0.329516539 | 0.340966921 | 0.329516539 |
| P(Threat = Low \| Type = *) | 0.189542484 | 0.321568627 | 0.488888889 |

## *4.3 Graphical User Interface Module*

The GUI designed to provide decision support for a force protection decision maker includes three screens, the Track Detail screen, the Log screen, and the Map screen.

The Track Detail screen (Figure 10) consists of four general sections of information. The upper-most left section provides basic track parameters that are largely generated by the fusion module. Beneath this section (titled, "Basis for Assessment") are fields of information that convey how the assessments were

derived. Further, there are details that show how the assessment may be invalid ("Against Evidence"). Such an approach offers some transparency that facilitates objective situation assessments. The lowest section that spans the width of the screen is a list of all tracks, including friendly forces that have most recently arrived in the track cue and are available to be specified in greater detail in the screen sections above. The section on the right is generally the course of action information. A list of possible operational activities by the threat is listed alongside how defense forces should respond. A basis for the corresponding defense operations is provided that conveys practical capabilities in the current context and possible constraints.

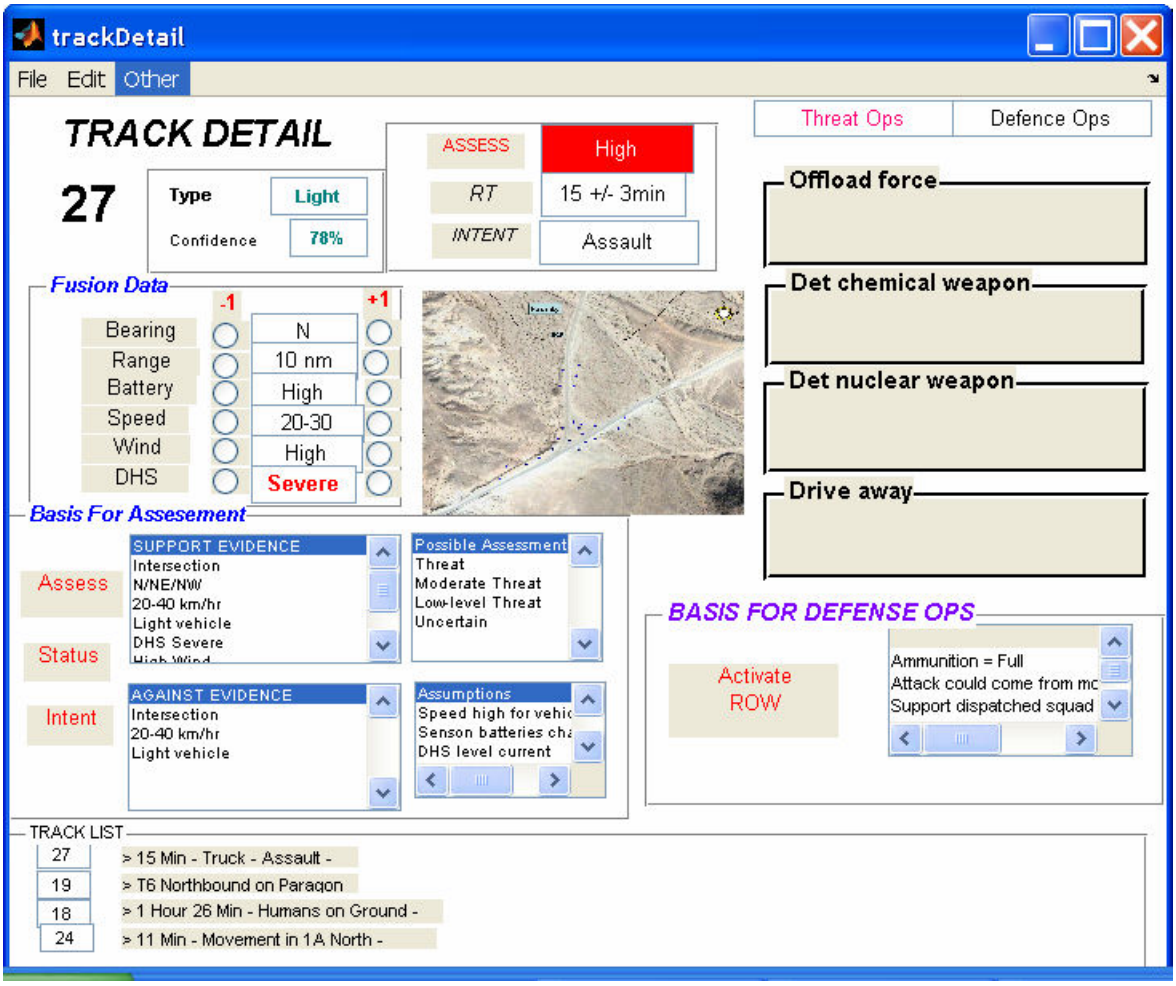


**Figure 10 Track Detail GUI Screen.**

The Log screen (Figure 11) affords the decision maker temporal information. A critical element in situation awareness is time-oriented information. Pace of events and time available to decide and act facilitate situation awareness and more effective decision making. The vertical bar in Figure 11 indicates the current time (i.e., "now"). The numbers across the top are time increments and move right to left in the application. The boxes represent events or tracks and are organized vertically with respect to priority, so the green box at the top is the most important event involving a track that may be approaching the gate. The green coloring corresponds to a low level threat. The box would turn yellow for a moderate threat and red for a high threat. The threat assessment is driven by the same data used to drive threat assessments in the track detail screen. If the track were moving towards the gate, the box would move right to left, and the estimated time to reach the gate would decrease. In this case, the decision maker

would know that there are at least 28.2 minutes until the track could reach the gate. If the vehicle were to pass the gate, then the box would have passed the vertical bar, and the box would indicate how long ago the vehicle passed the gate. The log screen shows general event data, such as computer network activity/announcements, and events that are significant for perimeter security, such as sunset times and high winds that may affect sensor reliability.

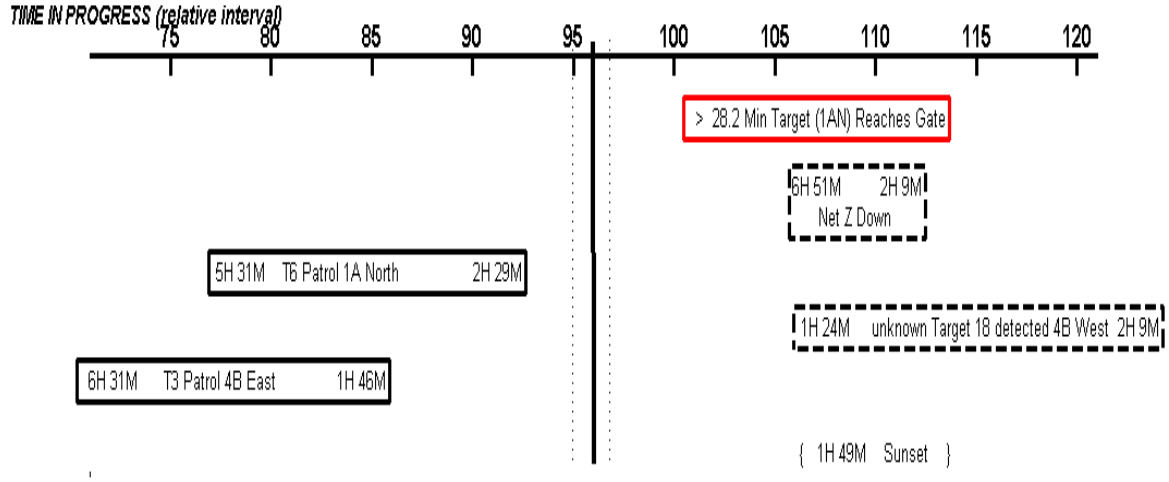


**Figure 11 Decision support Log screen.**

The Map screen (Figure 8) should present pertinent visual information regarding the environment of interest.

# 5 Future work

Arguably the most immediate area of future work is in establishing principles and practices for employing the three learning modes. There are different ways of combining three modes of machine learning and many options for how and when to employ each mode. We have just touched the surface of the possibilities for leveraging each learning mode for highest system performance. It stands to reason that a CARTMAP network can be tailored for each information fusion mode (vehicle type, speed, heading, and location). The vigilance parameter may be different for each mode. The vigilance may also require adjustment based on the type and ordering of the learning modes.

The core of our machine learning approach is an ART neural network. Other algorithms and architectures should be explored with the same goal in mind, that of integrating multiple learning modes. Reinforcement learning is a general area of research worth pursuing in the area of situation awareness where there is often not a clear win or lose outcome from which to measure success. There are also many ways of performing reinforcement learning, some closer to supervised learning, with stronger hints, and others that provide rare, but consistent hints about the system's performance.

The development of a complete POMDP for situation assessment is a research area deserving more attention. We have noted that it offers meaningful attributes to SA applications, but its implementation is non-trivial. The approach of using Bayesian Filtering deserves further research as does the Q-Learning approach. How many iterations to use in reinforcement learning on this problem is a legitimate research question, as is how best to acquire feedback from human decision makers or the overall force protection system, either directly or indirectly.

Another avenue of future machine learning research is to explore the use of ensembles or bagging for supervised learning [8]. The use of ensembles employs multiple "experts" that train the same network

using a different sampling with replacement from the original supervised training set. The combination of the experts' solutions results in higher performance than the use of a single network trained on the original data set.

Among the many system outputs explored, such as whether the vehicle was going North, one of the most challenging forms of assessments was adversarial intent [20]. An easier problem is to determine friendly person's intent, and even this is challenging. A malicious adversary that could apply forms of deception represents an enormous challenge. Progress in this direction would be appropriate for Sandia to pursue, leveraging the current work, which would benefit force protection customers.

The malicious context represents an additional domain that could leverage the current work. The sensor data used to determine vehicle parameters in this project was assumed to be correct and trustworthy. A significant variation in context would be sensor data that was deliberately manipulated for some strategic purpose. An interesting and valuable research agenda would investigate means of performing machine learning effectively despite malicious activity.

# 6 Conclusions

Situation awareness involves a host of difficult technical issues, including many regarding human factors. The human perspective played a part in every aspect of our system. The current research integrates three general areas including information fusion, decision support, and machine learning. The attributes of combining diverse sources of information facilitate a more complete picture of a discrete situation. Machine learning algorithms provide adaptive mechanisms that more continuously assess the environment over time. Moreover, the machine learning algorithms were selected to provide the proper amount and kind of information fusion and situation assessment, as well as to allow human feedback to the algorithms. Finally, maintaining the operator as a central and balanced system component helps the overall system consistently perform effectively.

The coordination of the three major machine learning approaches in a single architecture, using ARTMAP at its core, is an innovation that should prove valuable in addressing real-world problems. Life many times offers a limited amount of information with ground truth that can be used with supervised learning algorithms. More available is data with hints from the environment that can be used with reinforcement learning. Almost always, data is available without labels that can be used with unsupervised learning. Allowing these three modes of learning to be used in the same framework is an important contribution. Interesting advantages emerge, however, when these three approaches leverage one another. Reinforcement learning can utilize supervised learning when enough information about class labels is available from the environment. Unsupervised learning can take advantage of stored reinforcement learning information to go beyond mere clustering. The combination and leveraging of the learning modes results in a system that is greater than the sum of its individual parts.

# 7 References


[1] Alpaydin, E. *Introduction to Machine Learning*. Cambridge, MA: MIT Press, 2004.

[2] Armistead, L. (Ed.). (2004). Information Operations: Warfare and the Hard Reality of Soft Power. Dulles, VA: Brassey's, Inc.

[3] Arrow, K.J. Historical background. In *Studies in the Mathematical Theory of Inventory and Production*, edited by K.J. Arrow, S. Karlin, and H. Scarf. Stanford, CA: Stanford University Press, 1958.

[4] Aviv, Y. and Pazgal, A. A Partially Observed Markov Decision Process for Dynamic Pricing. *Management Science*, Vol. 51, No 9, 2005, pp 1400-1416.

[5] Carpenter, G. and Grossberg, S. The ART of Adaptive Pattern Recognition by a Self-Organizing Neural Network. *Computer*, Vol. 21, No 3, 1988, pp 77-87.

[6] Carpenter, G., Grossberg, S., Markuzon, N., Reynolds, J., and Rosen, D. Fuzzy ARTMAP: A Neural Network Architecture for Incremental Supervised Learning of Analog Multidimensional Maps. *IEEE Transactions on Neural Networks*, Vol. 3, No 5, 1992.

[7] Commission on the Intelligence Capabilities of the United States Regarding Weapons of Mass Destruction. Report to the President of the United States. March 31, 2005.

[8] Dietterich, T., An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization, *Machine Learning*, v.40 n.2, p.139-157, Aug. 2000.

[9] Duarte, M.F., Hu, Y. H. Vehicle Classification in Distributed Sensor Networks. *Journal of Parallel and Distributed Computing*, Vol. 64, No 7, pp 826-838, July 2004.

[10] Duggan, G., Banbury, S., Howes, A., Patrick, J, & Waldron, S. Too much, too little, or just right: Designing data fusion for situation awareness. In *Proceedings of the Human Factors and Ergonomics Society 48th Annual Meeting*. New Orleans, USA, 2004.

[11] Hall, D. and Llinas, J. An introduction to multisensor data fusion. *Proceedings of the IEEE*, *85*(1), 6-23, 1997.

[12] Kelly, E. J. and Kennedy, P.L. A Dynamic Stochastic Model of Mate Desertion. *Ecology*, Vol. 74, 1993, pp 351-366.

[13] Klein, G. A. Recognition-primed decisions. In W. B. Rouse (Ed.), *Advances in man-machine systems research*, Vol. 5, pp 47-92. Greenwich, CT: JAI Press, 1989.

[14] Kokar, M, Tomasik, T, and Weyman, J. (2004). Formalizing classes of information fusion systems. *Information Fusion,* Vol. 5(3), 2004, pp 189-202.

[15] Llinas, J., Bisantz, A., Drury, C., Seong, Y., and Jian, J. Studies and analyses of aided adversarial decision making, Phase 2: Research on human trust in automation. *Report AFRL-HE-WP-TR-1999-0216*. United States Air Force Research Laboratory: Wright-Patterson AFB, OH, 1998.

[16] Morrison, J., Kelly, R., Moore, R., & Hutchins, S. Tactical decision making under stress (TADMUS) decision support system. *Proceedings of IRIS National Symposium on Sensor and Data Fusion, MIT Lincoln Laboratory*. Lexington, MA, 1997.

[17] Paul, Jeffrey L. Smart sensor web: Web-based exploitation of sensor fusion for visualization of the tactical battlefield. *IEEE AESS Systems Magazine*, May, 2001.

[18] Puterman, M. Markov Decision Processes: Discrete Stochastic Dynamic Programming. *Wiley Series in Probability and Mathematical Statistics*, 1994.

[19] Rouse, W. B. Design for Success: A Human-Centered Approach to Designing Successful Products and Systems. New York: John Wiley & Sons, 1991.

[20] Santos, E. A cognitive architecture for adversary intent inferencing: Structure of knowledge and computation. *Proceedings of SPIE*, *5091*, 2003, pp 182-193.

[21] Simon, H. Sciences of the Artificial (3rd ed.). Cambridge, MA: MIT Press, 1981.

[22] Steinberg, A. and Bowman, C. Revisions to the JDL Data Fusion Model. In D. Hall & J. Llinas (Eds), *Handbook of Multisensor Data Fusion*. Boca Raton, FL: CRC Press, 2001.

[23] Sutton, R. and Barto, A. *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.

[24] Vicente, K. Cognitive Work Analysis: Toward Safe, Productive, and Healthy Computer-Based Work. Mahwah, NJ: Lawrence Erlbaum Associates, 1999.

[25] Xu, R. and Wunsch, D. Survey of Clustering Algorithms. *IEEE Transactions on Neural Networks*, Vol. 16, No 3, 2005.

[26] Xu, R. and Wunsch, D. *Clustering*. IEEE Press/Wiley, 2008, to appear.

# 8 Appendix A - Adaptive Resonance Theory (ART)

This appendix is taken verbatim from [26] without references and the equation and figure numbers are unchanged from the original.

An important problem with competitive learning-based clustering is stability. Moore (1989) defines the stability of an incremental clustering algorithm in terms of two conditions: "(1) No prototype vector can cycle, or take on a value that it had at a previous time (provided it has changed in the meantime). (2) Only a finite number of clusters are formed with infinite presentation of the data." The first condition considers the stability of individual prototype vectors of the clusters, and the second one concentrates on the stability of all the cluster vectors. In this sense, the algorithms discussed before do not always produce stable clusters, as pointed out by Moore (1989) and Grossberg (1976a). The reason for this instability lies in the algorithms' plasticity, which is required to adapt to important new patterns. However, this plasticity may cause the memories of prior learning to be lost, worn away by the recently-learned knowledge. Grossberg (1987a, 1988) refers to this problem as the plasticity and stability dilemma, i.e., how adaptable (plastic) should a learning system be so that it does not suffer from catastrophic forgetting of previously-learned rules (stability)?

Adaptive resonance theory (ART) was developed by Carpenter and Grossberg (1987a, 1988) as a solution to the plasticity and stability dilemma. ART can learn arbitrary input patterns in a stable, fast, and self-organizing way, thus overcoming the effect of learning instability that plagues many other competitive networks. ART is not, as is popularly imagined, a neural network architecture. It is a learning theory hypothesizing that resonance in neural circuits can trigger fast learning. As such, it subsumes a large family of current and future neural network architectures with many variants. ART1 is the first member, which only deals with binary input patterns (Carpenter and Grossberg, 1987a, 1988), although it can be extended to arbitrary input patterns by utilizing a variety of coding mechanisms. ART2 extends the applications to analog input patterns (Carpenter and Grossberg, 1987b), and ART3 introduces a new mechanism originating from elaborate biological processes to achieve more efficient parallel searches in hierarchical structures (Carpenter and Grossberg, 1990). Fuzzy ART (FA) incorporates fuzzy set theory and ART and can work for all real data sets (Carpenter et al., 1991b). Linares-Barranco et al. (1998) demonstrated the hardware implementations and very-large-scale integration (VLSI) design of ART systems. In Wunsch (1991) and Wunsch et al. (1993), the optical correlator-based ART implementation, instead of the implementation of ART in electronics, was also discussed.

## *8.1 ART1*

As depicted in Fig. 2, the basic ART1 architecture consists of two-layer nodes or neurons, the feature representation field $F_1$, and the category representation field $F_2$, whose present state is known as short-term memory (STM). The neurons in layer $F_1$ are activated by the input pattern, while the prototypes of the formed clusters are stored in layer $F_2$. The neurons in layer $F_2$ that are already being used as representations of input patterns are said to be committed. Correspondingly, the uncommitted neuron encodes no input patterns. The two layers are connected via adaptive weights: a bottom-up weight matrix $\mathbf{W}^{12}=\{w_{ij}^{12}\}$, where the index represents the connection from the $i^{th}$ neuron in layer $F_1$ to the $j^{th}$ neuron in layer $F_2$ , and a top-down weight matrix $\mathbf{W}^{21}=\{w_{ji}^{21}\}$, which is also called long-term memory (LTM). After layer $F_2$ is activated according to the winner-take-all competition, which occurs between a certain number of committed neurons and one uncommitted neuron, an expectation is reflected in layer $F_1$ and compared with the input pattern. The orienting subsystem with the pre-specified vigilance parameter $\rho$ ($0\le\rho\le1$) determines whether the expectation and the input pattern are closely matched. If the match meets the vigilance criterion, weight adaptation occurs, where both bottom-up and top-down weights are updated simultaneously. This procedure is called resonance, which suggests the name of ART. On the other hand, if the vigilance criterion is not met, a reset signal is sent back to layer $F_2$ to shut off the current winning neuron, which will remain disabled for the entire duration of the presentation of this input

pattern, and a new competition is performed among the rest of the neurons. This new expectation is then projected into layer $F_1$, and this process repeats until the vigilance criterion is met. In the case that an uncommitted neuron is selected for coding, a new uncommitted neuron is created to represent a potential new cluster. It is clear that the vigilance parameter $\rho$ has a function similar to that of the threshold parameter $\theta$ of the leader-follower algorithm. The larger $\rho$ is, the fewer mismatches are going to be tolerated; therefore, the more clusters are likely to be generated.

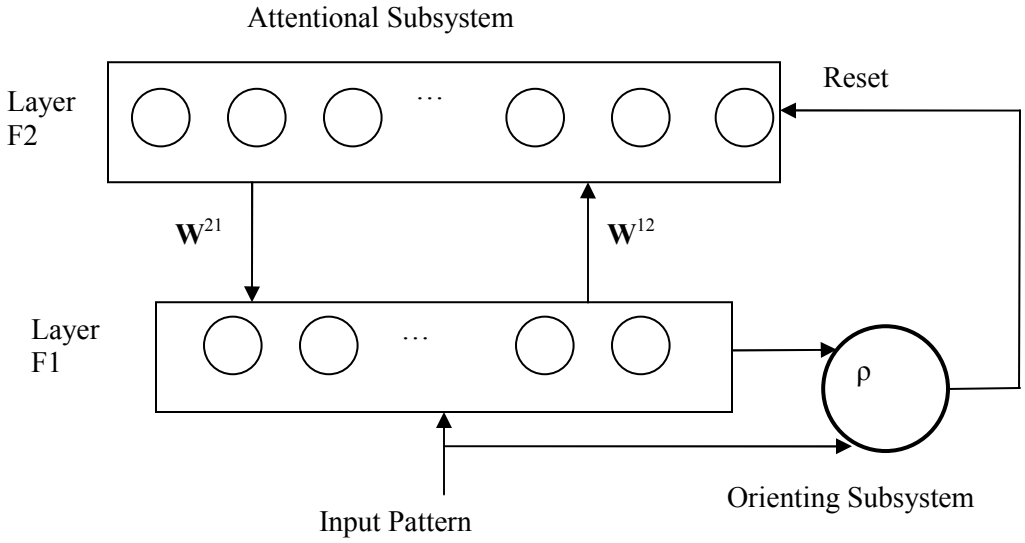


Figure 2 ART1 architecture. Two layers are included in the attentional subsystem, connected via bottom-up and top-down adaptive weights. Their interactions are controlled by the orienting subsystem through a vigilance parameter.

At this point, we summarize the basic steps of ART1 as follows, which are also depicted in Fig. 3:

1. Initialize the weights as $w_{ij}^{12}=\xi/(\xi-1+d)$, where $d$ is the dimensionality of the binary input pattern $\mathbf{x}$, $\xi$ is a parameter that is larger than one, and $w_{ji}^{21}=1$;
2. Present a new pattern $\mathbf{x}$ and calculate the input from layer $F_1$ to layer $F_2$ as

$$T_j = \sum_{i=1}^{d} w_{ij}^{12} x_i \quad ; \qquad (16)$$

3. Activate layer $F_2$ by choosing neuron $J$ with the winner-take-all rule,

$$T_J = \max_j \{T_j\} \quad ; \qquad (17)$$

4. Compare the expectation from layer $F_2$ with the input pattern. If

$$\rho \le \frac{\left|\mathbf{x} \cap \mathbf{W}_J^{21}\right|}{\left|\mathbf{x}\right|} \quad , \qquad (18)$$

where $\cap$ represents the logic AND operation, go to step 5a; otherwise, go to step 5b.

5. a. Update the corresponding weights for the active neuron as

$$\mathbf{W}_J^{21}(\text{new}) = \mathbf{x} \cap \mathbf{W}_J^{21}(\text{old}) \quad , \qquad (19)$$

and

$$\mathbf{W}_J^{12}(\text{new}) = \frac{\xi \mathbf{W}_J^{21}(\text{new})}{\xi - 1 + \left|\mathbf{W}_J^{21}(\text{new})\right|}. \quad (20)$$

If $J$ is an uncommitted neuron, create a new uncommitted neuron with the initial values set as in Step 1;

b. Send a reset signal to disable the current active neuron by the orienting subsystem, and return to step 3;

6. Return to step 2 until all patterns are processed.

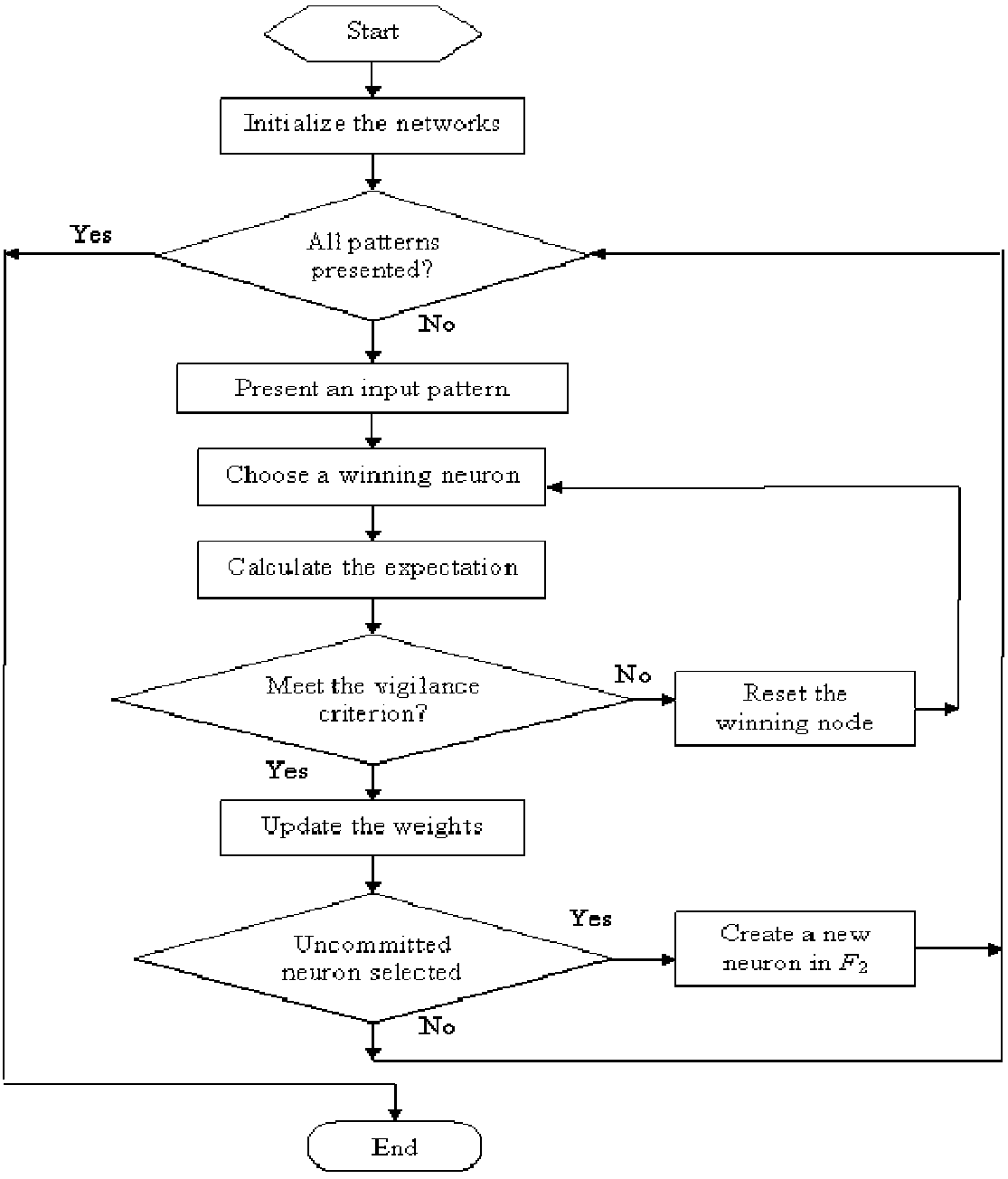


Figure 3 Flowchart of ART1.

In brief, ART1 obtains its adaptability in terms of dynamically creating new clusters in order to learn new patterns or events. At the same time, the problem of instability is solved by allowing the cluster weight vectors to move only in one direction during learning, as clearly shown in Eqs.19 and 20 (Moore, 1989). Moore (1989) and Linares-Barranco et al. (1998) also discussed a number of important properties of ART1, such as self-scaling, direct access to a stored cluster, learning of rare events, and direct access to subset and superset.

It is worth mentioning that, by incorporating two ART1 modules, which receive input patterns ($ART_a$) and corresponding labels ($ART_b$), respectively, with an inter-ART module, the resulting ARTMAP system can be used for supervised classifications (Carpenter et al., 1991a). The ART1 modules can be replaced with FA modules, Gaussian ART (GA) modules (Williamson, 1996), or ellipsoid ART (EA) modules (Anagnostopoulos and Georgiopoulos, 2001), which correspond to the supervised classification system known as fuzzy ARTMAP, as illustrated in Fig. 4 (Carpenter et al., 1992), Gaussian ARTMAP, and ellipsoid ARTMAP, respectively. A similar idea, omitting the inter-ART module, is known as laterally primed adaptive resonance theory (LAPART) (Healy et al., 1993). Carpenter (2003) used a nested sequence to describe the relations among several variants of ARTMAP, as fuzzy ARTMAP $\subset$ default ARTMAP (Carpenter, 2003) $\subset$ ARTMAP-IC (Carpenter and Markuzon, 1998) $\subset$ distributed ARTMAP (Carpenter et al., 1998).

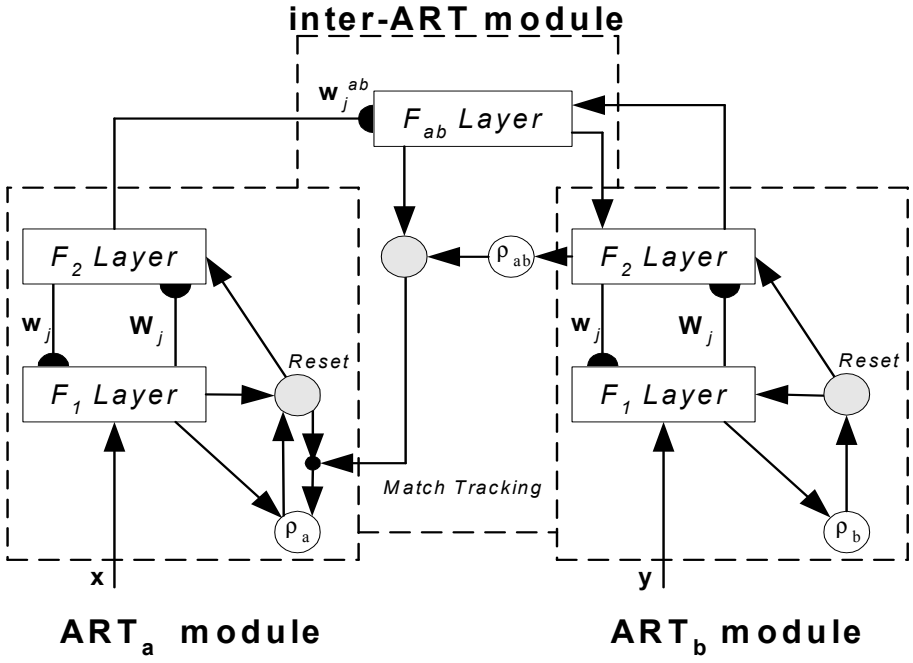


Figure 4 Fuzzy ARTMAP block diagram. Fuzzy ARTMAP consists of two FA modules (ARTa and ARTb) interconnected via an inter-ART module. The ARTa module clusters patterns of the input domain, and ARTb the ones of the output domain. The match tracking strategy ensures the consistency of category prediction between two ART modules by dynamically adjusting the vigilance parameter of ARTa.

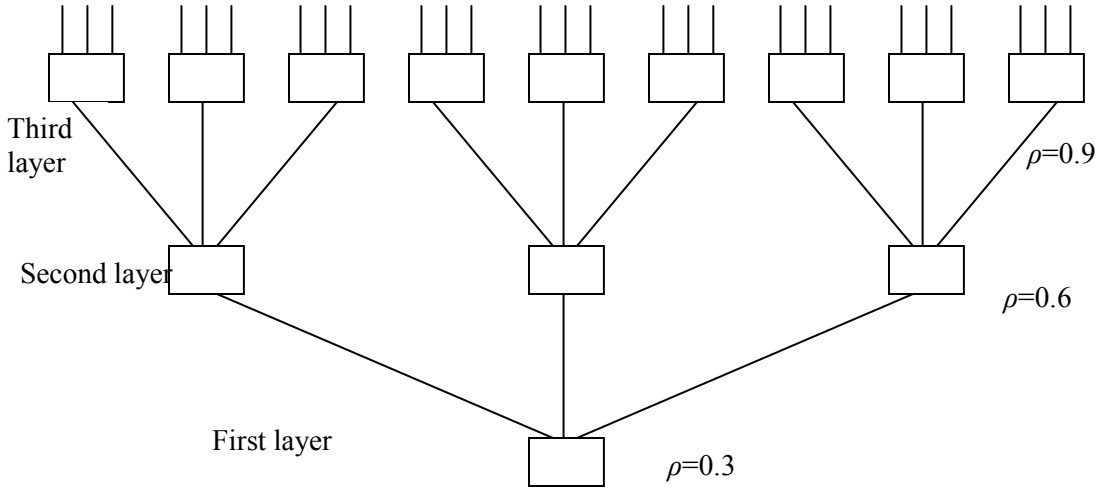


Figure 5 A hierarchy of ART1 units. The input pattern is fed in at the bottom, and the winning output is read out at the top. The prototype vectors of the nodes in layer F2 are used as inputs to the next layer ART1 unit. This makes the different prototypes nevertheless contain information in common. Therefore, it is possible to have a higher-layer ART unit that has a lower vigilance threshold and can combine what were previously separate clusters.

Wunsch (1991) and Wunsch et al. (1993) discussed the ease with which ART may be used for hierarchical clustering. The proposed method, called ART tree, is a hierarchy in which the same input pattern is sent to every level. The ART units in a given level that get to look at the input are determined by the winning nodes of layer $F_2$ at a lower level. Thus, all nodes of layer $F_2$ in the entire hierarchy see the same input pattern, or nothing at all. This allows ART to perform hierarchical clustering in that the lower-level clusters will form perfect subsets of the higher-level clusters. An ART1 hierarchy with a total of 39 prototypes is illustrated in Fig. 5. Also, two ART-based approaches for hierarchical clustering were presented by Bartfai and White (1997), known as hierarchical ART with joining (HART-J) and hierarchical ART with splitting (HART-S).

## *8.2 Fuzzy ART*

Fuzzy ART extends the ART family by being capable of learning stable recognition clusters in response to both binary and real-valued input patterns with either fast or slow learning (Carpenter et al., 1991b). FA maintains architecture and operations similar to ART1 while using the fuzzy set operators to replace the binary operators so that it can work for all real data sets. We describe FA by emphasizing its main difference with ART1 in terms of the following five phases, known as preprocessing, initialization, category choice, category match, and learning.

- Preprocessing. Each component of a $d$-dimensional input pattern $\mathbf{x}=(x_1,\ldots,x_d)$ must be in the interval [0,1].
- Initialization. The real-valued adaptive weights $\mathbf{W}=\{w_{ij}\}$, representing the connection from the $i^{th}$ neuron in layer $F_2$ to the $j^{th}$ neuron in layer $F_1$, include both the bottom-up and top-down weights of ART1. Initially, the weights of an uncommitted node are set to one. Larger values may also be used, however, this will bias the tendency of the system to select committed nodes (Carpenter et al., 1991b).
- Category choice. After an input pattern is presented, the nodes in layer $F_2$ compete by calculating the category choice function, defined as

$$T_j = \frac{|\mathbf{x} \wedge \mathbf{w}_j|}{\alpha + |\mathbf{w}_j|}, \quad (21)$$

where $\wedge$ is the fuzzy AND operator defined by

$$(\mathbf{x} \wedge \mathbf{y})_i = \min(x_i, y_i), \quad (22)$$

and $\alpha>0$ is the choice parameter to break the tie when more than one prototype vector is a fuzzy subset of the input pattern. Particularly, the limit $\alpha \to 0$ is called the conservative limit (Carpenter et al., 1991b). Also, $\alpha$ is related to the vigilance parameter $\rho$; $\alpha$ should decrease as $\rho$ decreases (Huang et al., 1995).
Similar to ART1, the neuron $J$ becomes activated with the winner-take-all rule,

$$T_J = \max_j \{T_j\}. \quad (23)$$

- Category match. The category match function of the winning neuron is then tested with the vigilance criterion. If

$$\rho \le \frac{|\mathbf{x} \wedge \mathbf{w}_J|}{|\mathbf{x}|}, \quad (24)$$

resonance occurs. Otherwise, the current winning neuron is disabled and a new neuron in layer $F_2$ is selected and examined with the vigilance criterion. This search process continues until Eq. 24 is satisfied.

- Learning. The weight vector of the winning neuron that passes the vigilance test at the same time is updated using the following learning rule,

$$\mathbf{w}_J(\text{new}) = \beta(\mathbf{x} \wedge \mathbf{w}_J(\text{old})) + (1-\beta)\mathbf{w}_J(\text{old}), \quad (25)$$

where $\beta \in [0,1]$ is the learning rate parameter. Carpenter et al. (1991b) introduced a method, called fast-commit slow-recode, for achieving efficient coding of noisy input patterns. In this context, $\beta$ is set to one when an uncommitted node is selected to represent the current input pattern. Correspondingly, Eq. 25 becomes

$$\mathbf{w}_J(\text{new}) = \mathbf{x}, \quad (26)$$

which indicates that the input pattern is directly copied as the prototype of the new cluster. On the other hand, committed prototypes are updated with a slow learning rate, $\beta<1$, to prevent them from being corrupted by noise.

A practical problem in applying FA is the possibility of cluster proliferation, which occurs as a result of an arbitrarily small norm of input patterns (Moore, 1989; Carpenter et al., 1991b). Since the norm of weight vectors does not increase during learning, many low-valued prototypes may be generated without further access. The solution to the cluster proliferation problem is to normalize the input patters (Carpenter et al., 1991b) so that,

$$|\mathbf{x}| = \varsigma, \quad \varsigma > 0. \quad (27)$$

This is an extended step of the preprocessing phase.

One way to normalize an input pattern $\mathbf{x}$ is to divide it by its norm, written as,

$$\mathbf{x}^* = \frac{\mathbf{x}}{|\mathbf{x}|}. \quad (28)$$

However, this method does not maintain the amplitude information of the input patterns. Alternately, Carpenter et al. (1991b) proposed a normalization rule, known as complement coding, to normalize input patterns without losing the amplitude information. Specifically, an input pattern $d$-dimensional $\mathbf{x}=(x_1,\ldots,x_d)$ is expanded as a $2d$-dimensional vector

$$\mathbf{x}^* = (\mathbf{x}, \mathbf{x}^c) = (x_1,\ldots,x_d, x_1^c,\ldots,x_d^c), \quad (29)$$

where $x_i^c=1-x_i$ for all $i$. A direct mathematical manipulation shows that input patterns in complement coding form are automatically normalized,

$$|\mathbf{x}^*| = |(\mathbf{x}, \mathbf{x}^c)| = \sum_{i=1}^{d} x_i + \sum_{i=1}^{d} x_i^c = \sum_{i=1}^{d} x_i + d - \sum_{i=1}^{d} x_i = d. \quad (30)$$

Corresponding to the expansion of the input patterns, now, the adaptive weight vectors $\mathbf{w}_j$ are also in the $2d$-dimensional form, represented as,

$$\mathbf{w}_j = (\mathbf{u}_j, \mathbf{v}_j^c). \quad (31)$$

Initially, $\mathbf{w}_j$ are still set to one, which causes $\mathbf{u}_j$ to be set to one and $\mathbf{v}_j$ to be set to zero. The adaptation of $\mathbf{w}_j$ also follows the same rule in Eq. 25.

A 2-dimensional geometric interpretation of FA cluster update with complement coding and fast learning is illustrated in Fig. 6, where each category is represented as a rectangle. Another method that has hyper-rectangular representations of clusters is called fuzzy min-max clustering neural networks (Simpson, 1993; Gabrys and Bargiela, 2000). As can be seen in Fig. 6, $\mathbf{u}_j$ and $\mathbf{v}_j$ in Eq. 31 are both 2-dimensional vectors defining two corners of rectangle $R_j$, which is considered a geometric representation of cluster $j$. The size of $R_j$ can be calculated using

$$|R_j| = |\mathbf{v}_j - \mathbf{u}_j|. \quad (32)$$

Note that when an uncommitted node $j$ is eligible to encode an input pattern $\mathbf{x}^*=(\mathbf{x}, \mathbf{x}^c)$, the fast learning in Eq. 26 leads to

$$\mathbf{w}_j(\text{new}) = \mathbf{x}^* = (\mathbf{x}, \mathbf{x}^c), \quad (33)$$

which implies that both $\mathbf{u}_j$ and $\mathbf{v}_j$ are equal to $\mathbf{x}$. In this situation, rectangle $R_j$ coincides with the point $\mathbf{x}$ with zero size.

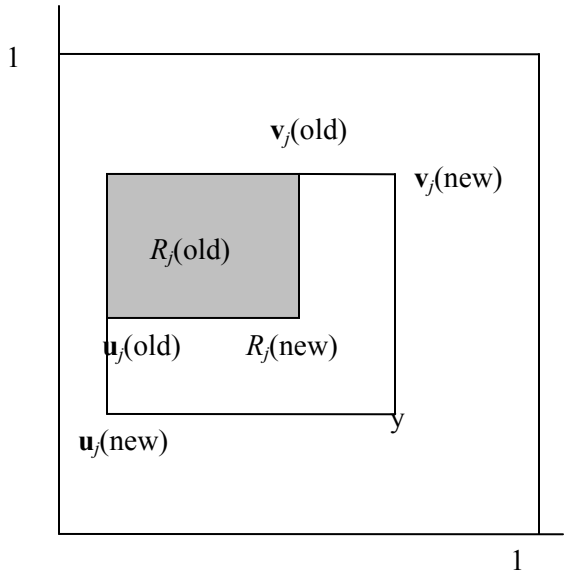


Figure 6 Category update of FA with complement coding and fast learning. Each category j has a geometric representation as a rectangle Rj. The shaded rectangle expands to the smallest rectangle to incorporate the presented input pattern x into the cluster.

Suppose cluster *j* corresponding to the shaded area is now eligible to encode a new input pattern $\mathbf{y}^*=(\mathbf{y}, \mathbf{y}^c)$. Again, following the fast learning rule in Eq. 26, we have

$$\begin{aligned}\mathbf{w}_j(\text{new}) &= \mathbf{y}^* \wedge \mathbf{w}_j(\text{old}) \\ &= \left(\mathbf{y} \wedge \mathbf{u}_j(\text{old}), \mathbf{y}^c \wedge \mathbf{v}_j^c(\text{old})\right) \\ &= \left(\mathbf{y} \wedge \mathbf{u}_j(\text{old}), \left(\mathbf{y} \vee \mathbf{v}_j(\text{old})\right)^c\right) \\ &= \left(\mathbf{u}_j(\text{new}), \mathbf{v}_j^c(\text{new})\right)\end{aligned}, \quad (34)$$

where $\vee$ represents the fuzzy OR operator

$$\left(\mathbf{x} \vee \mathbf{y}\right)_i = \max\left(x_i, y_i\right). \quad (35)$$

As can be seen from Eq. 34, the rectangle $R_j$ expands with the smallest size to include both the previous representation region and the new input pattern. It is also interesting to see that if $\mathbf{y}$ is already inside $R_j$, there will be no change for the weight vector and, correspondingly, for the rectangle $R_j$.

Now we turn to examine the relation between the vigilance parameter $\rho$ and the size of the rectangle $R_j$. we have already shown, learning will occur only if the winning cluster *j* meets the vigilance criterion in Eq. 24. Particularly, when the input pattern is 2-dimensional and complement coding is used, we have $|\mathbf{y}^*|=2$. Then, we can rewrite Eq. 24 as

$$2\rho \le \left|\mathbf{y}^* \wedge \mathbf{w}_j\right|. \quad (36)$$

By using Eq. 34, we have

$$
\begin{aligned}
\left|\mathbf{y}^* \wedge \mathbf{w}_j\right| &= \left|\left(\mathbf{y}, \mathbf{y}^c\right) \wedge \left(\mathbf{u}_j, \mathbf{v}_j^c\right)\right| \\
&= \left|\left(\mathbf{y} \wedge \mathbf{u}_j\right), \left(\mathbf{y}^c \wedge \mathbf{v}_j^c\right)\right| \\
&= \left|\left(\mathbf{y} \wedge \mathbf{u}_j\right), \left(\mathbf{y} \vee \mathbf{v}_j\right)^c\right| \\
&= \left|\left(\mathbf{y} \wedge \mathbf{u}_j\right)\right| + 2 - \left|\mathbf{y} \vee \mathbf{v}_j\right| \\
&= 2 - \left|R_j(\text{new})\right| \qquad . \qquad (37)
\end{aligned}
$$

By combining Eqs. 36 and 37, we see that resonance will occur when the expanded rectangle meets

$$\left|R_j(\text{new})\right| \le 2(1-\rho) \qquad . \qquad (38)$$

Clearly, the closer the vigilance parameter $\rho$ is to 1, the smaller the size of the rectangles will be, and correspondingly, the smaller the number of input patterns that are represented by the cluster prototypes, as discussed previously.

Similar manipulations can be applied to a more general situation with $d$-dimensional input patterns, which infers the size of a hyper-rectangle $R_j$,

$$\left|R_j\right| \le d - \left|\mathbf{w}_j\right| , \qquad (39)$$

together with the constraint on its maximum size,

$$\left|R_j\right| \le d(1-\rho) \qquad . \qquad (40)$$

The discussions above can be summarized with the stable category learning theorem (Carpenter et al., 1991b):

> "In response to an arbitrary sequence of analog or binary input vectors, a Fuzzy ART system with complement coding and fast learning forms stable hyper-rectangular categories $R_j$, which grow during learning to a maximum size $|R_j| \le d(1-\rho)$ as $|\mathbf{w}_j|$ monotonically decreases. In the conservative limit, one-pass learning obtains such that no reset or additional learning occurs on subsequent presentations of any input. Similar properties hold for the fast-learn slow-recode case, except that repeated presentations of an input may be needed before stabilization occurs."

As we have already seen, FA exhibits many desirable characteristics, such as fast and stable learning, transparent learning paradigm, and atypical pattern detection. Huang et al. (1995) investigated and discussed more properties of FA in terms of prototype, access, reset, and the number of learning epochs required for weight stabilization. A comparison of the performance of FA and ART2 was presented by Frank et al. (1998).

## *8.3 Other ART networks*

FA produces a hyper-rectangular representation of clusters in the feature space, which is more suitable for representing data that are uniformly distributed within hyper-rectangles (Williamson, 1996). When this assumption does not hold, the fuzzy categories may become an inefficient geometrical representation for exploring the potential data structures (Anagnostopoulos and Georgiopoulos, 2001; Williamson, 1996). Moreover, FA is sensitive to noise and has a problem of category proliferation in noisy data (Baraldi and

Alpaydin, 2002; Baraldi and Blonda, 1999; Williamson, 1996). Williamson (1996) pointed out two possible causes of the category proliferation problem: (1) both the category choice and category match functions are flat within a cluster's hyper-rectangle and (2) fast learning is performed. As a solution to this problem, Williamson (1996) further suggested the Gaussian-defined category choice and match functions, which monotonically increase toward the center of a cluster, to replace those of FA. The obtained new ART module, in which each cluster is represented as a hyper-ellipsoid geometrically, is called Gaussian ART.

In the context of Gaussian distributions, each GA cluster $j$, representing $d$-dimensional input patterns, is described by a $(2d+1)$-dimensional prototype vector $\mathbf{w}_j$ consisting of three components: $\boldsymbol{\mu}_j$ is the $d$-dimensional mean vector, $\boldsymbol{\sigma}_j$ is the $d$-dimensional standard deviation vector, and $N_j$ is a scalar recording the number of patterns cluster $j$ has encoded. Correspondingly, the category choice function is defined as a discriminant function examining the posteriori probability of cluster $j$ given an input pattern $\mathbf{x}$,

$$T_j = -\frac{1}{2}\sum_{i=1}^{d}\left(\frac{\mu_{ji}-x_i}{\sigma_{ji}}\right)^2 - \log\left(\prod_{i=1}^{d}\sigma_{ji}\right) + \log P(j), \quad (41)$$

where the priori probability of cluster $j$ is calculated as

$$P(j) = \frac{N_j}{\sum_{i=1}^{C} N_i}, \quad (42)$$

with $C$ being the number of clusters.

After the cluster $J$ with the maximum discriminant function is activated, the vigilance test is performed via the calculation of the value of the category match function, written as,

$$\rho_J = -\frac{1}{2}\sum_{i=1}^{d}\left(\frac{\mu_{Ji}-x_i}{\sigma_{Ji}}\right)^2, \quad (43)$$

which determines how well $\mathbf{x}$ matches with $J$ in terms of the measurement of its distance to the mean of $J$, relative to the standard deviation.

The learning of the GA winning cluster $J$ includes the update of the three elements of the prototype vector, given as follows,

$$N_J = N_J + 1, \quad (44)$$

$$\mu_{Ji}(\text{new}) = \left(1-\frac{1}{N_J}\right)\mu_{Ji}(\text{old}) + \frac{1}{N_J}x_i, \quad (45)$$

$$\sigma_{Ji}(\text{new}) = \begin{cases} \sqrt{\left(1-\frac{1}{N_J}\right)\sigma_{Ji}^2(\text{old}) + \frac{1}{N_J}\left(x_i-\mu_{Ji}(\text{new})\right)^2}, & \text{if } N_J > 1 \\ \gamma, & \text{otherwise} \end{cases}, \quad (46)$$

where $\gamma$ is the initial standard deviation.

GA is designed as a class of probability density function estimators for Gaussian mixtures, using the maximum likelihood (ML) method (Baraldi and Alpaydin, 2002). The relation between GA and the expectation-maximization (EM) approach for modeling mixture density was discussed by Williamson (1997). Furthermore, Baraldi and Alpaydin (2002) generalized GA in their defined constructive incremental clustering framework, called simplified ART (SART), which includes two other ART networks, known as symmetric fuzzy ART (SFART) and fully self-organizing SART (FOSART) networks. It is interesting to point out that FOSART uses a "soft-to-hard competitive model transition" to minimize the distortion error (Baraldi and Alpaydin, 2002), which makes it fall out of the category of hard competitive learning to which other ART networks belong.

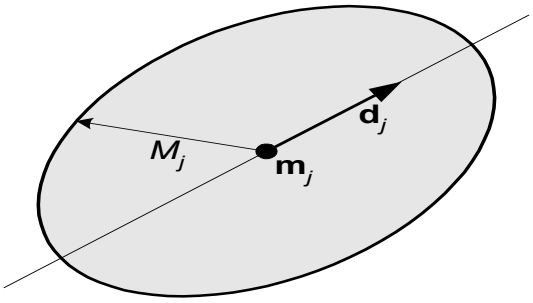


Figure 7 Example of the geometric representation of an EA cluster j in a 2-dimensional feature space.

Noticing the disadvantage of the lack of the fast learning law in GA, Anagnostopoulos and Georgiopoulos (2001) proposed ellipsoid ART, which evolved as a generalization of an early ART network, called hyper-sphere ART (HA) for hyper-spherical clusters (Anagnostopoulos and Georgiopoulos, 2000), to explore a hyper-ellipsoidal representation of EA clusters while following the same learning and functional principles of FA. A typical example of such a cluster representation, when the input space is 2-dimensional, is depicted in Fig. 7, where each category $j$ is described by a center location vector $\mathbf{m}_j$, orientation vector $\mathbf{d}_j$, and Mahalanobis radius $M_j$, which are collected as the prototype vector $\mathbf{w}_j$=[ $\mathbf{m}_j$, $\mathbf{d}_j$, $M_j$]. The orientation vector will be constant once it is set. If we define the distance between an input pattern $\mathbf{x}$ and a category $j$ as

$$D(\mathbf{x}, \mathbf{w}_j) = \max\left\{\left\|\mathbf{x}-\mathbf{m}_j\right\|_{\mathbf{S}_j}, M_j\right\} - M_j \quad , \qquad (47)$$

$$\left\|\mathbf{x}-\mathbf{m}_j\right\|_{\mathbf{S}_j} = \sqrt{(\mathbf{x}-\mathbf{m}_j)^T \mathbf{S}_j (\mathbf{x}-\mathbf{m}_j)} \quad , \qquad (48)$$

where $\mathbf{S}_j$ is the cluster's shape matrix, defined as

$$\mathbf{S}_j = 1/\mu^2 \left(\mathbf{I} - (1-\mu^2)\mathbf{d}_j \mathbf{d}_j{}^T\right) \quad , \qquad (49)$$

and $\mu \in (0,1]$ is a constant ratio between the length of the hyper-ellipsoid's minor axes (with equal length) and major axis (for $\mu$=1, the geometric representations become hyper-spheres, in which case the network is called HA), then the representation region of $j$, which is the shaded area in Fig. 7, can be defined as a set of points in the input space, satisfying the condition

$$D(\mathbf{x}, \mathbf{w}_j) = 0 \Rightarrow \left\|\mathbf{x}-\mathbf{m}_j\right\|_{\mathbf{S}_j} \le M_j \quad . \qquad (50)$$

Similar to FA, the competition is performed via the category choice function, defined as,

$$T_j = \frac{D_{\max} - 2M_j - D(\mathbf{x}, \mathbf{w}_j)}{D_{\max} - 2M_j + a}, \qquad (51)$$

where $a>0$ is the choice parameter, $D_{\max}$ is a parameter also greater than 0, and the match between the input pattern and the winning category's representation region is examined through the category match function

$$\rho_j = \frac{D_{\max} - 2M_j - D(\mathbf{x}, \mathbf{w}_j)}{D_{\max}}. \qquad (52)$$

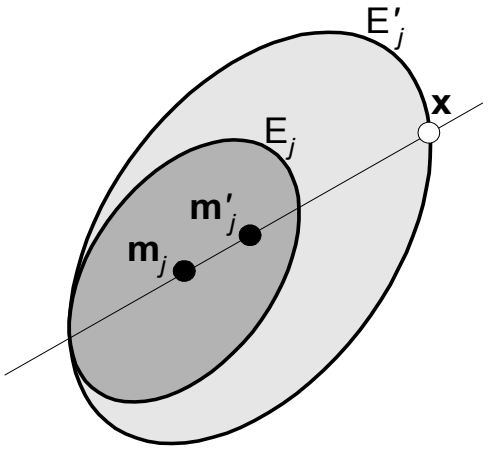


Figure 8 Update of an EAM category j due to a training pattern x when the feature space is 2-dimensional. The representation region expands to contain the original region and the new pattern.

When it has been decided that a category $j$ must be updated by a training pattern $\mathbf{x}$, its representation region expands so that it becomes the minimum-volume hyper-ellipsoid that contains the entire, original representation region and the new pattern. An example of this process for a 2-dimensional feature space is shown in Fig. 8, where the original representation region $E_j$ expands to become $E'_j$. More specifically, the center location vector, orientation vector, and Mahalanobis radius are updated with the following equations:

$$\mathbf{m}_j(\text{new}) = \mathbf{m}_j(\text{old}) + \frac{\eta}{2}\left(1 - \frac{\min\left(M_j(\text{old}), \left\|\mathbf{x} - \mathbf{m}_j(\text{old})\right\|_{\mathbf{S}_j(\text{old})}\right)}{\left\|\mathbf{x} - \mathbf{m}_j(\text{old})\right\|_{\mathbf{S}_j(\text{old})}}\right)\left(\mathbf{x} - \mathbf{m}_j(\text{old})\right), \qquad (53)$$

where $\eta$ is the learning rate,

$$\mathbf{d}_j = \frac{\mathbf{x}^{(2)} - \mathbf{m}_j}{\left\|\mathbf{x}^{(2)} - \mathbf{m}_j\right\|_2}, \mathbf{x}^{(2)} \neq \mathbf{m}_j, \qquad (54)$$

where $\mathbf{x}^{(2)}$ represents the second pattern encoded by cluster $j$, and

$$M_j(\text{new}) = M_j(\text{old}) + \frac{\eta}{2}\left(\max\left(M_j(\text{old}), \left\|\mathbf{x} - \mathbf{m}_j(\text{old})\right\|_{\mathbf{S}_j(\text{old})}\right) - M_j(\text{old})\right). \qquad (55)$$

Notice that if $\mathbf{x}$ falls inside the representation region of $j$, no update occurs because $j$ has already taken into account the presence of $\mathbf{x}$.

## Distribution:

| | | |
|---|---|---|
| 1 | MS 1221 | M. Scott, 5600 |
| 1 | MS 1235 | W. Cook, 5630 |
| 1 | MS 0671 | G. Rivord, 5610 |
| 1 | MS 0672 | T. Draelos, 5614 |
| 1 | MS 0188 | D. Chavez, LDRD Office, 1011 |
| 2 | MS 9018 | Central Technical Files, 8944 |
| 2 | MS 0899 | Technical Library, 9536 |