Published in 2006 International Conference on Artificial Intelligence (ICAI '06) (Las Vegas, NV, 26-29 June 2006).

Tactical Alert Management

Mike Daily, Ron Azuma, Youngkwan Cho, Troy Rockwood
HRL Laboratories
and
Susan Gottschlich
Raytheon Network Centric Systems

Keywords: incremental learning, active learning, multimodal user interfaces, ontologies

Abstract

Emerging technology now in use in the military enables air vehicles, dismounted soldiers, persistent sensors, and higher headquarters to share multimedia information in the form of alerts that will increase the awareness of a dismounted soldier conducting urban operations. The challenge is to manage the complexity of such a tactical alerting system so that soldiers are not overwhelmed or distracted by new technology. There is a large number of ways to present soldiers with any specific information in the form of an alert, including the choice not to present the information. An effective tactical alert management system must accurately learn the preferences of the soldier as well as policy from higher-level command from a small number of examples in a small amount of time with a simple interface. This paper presents the development of an information management engine that accomplishes this difficult goal. We present the system concept, an ontology for tactical alerts, a supervised learning approach, and results of pilot study experiments that give preliminary validation for the use of such a system.

Introduction

The goal of this research is to manage the complexity of a tactical alerting system so that users are enabled and not encumbered by their technology. Even for a relatively simple alert format, the multidimensional space of all possible alerts is extremely large making it untenable for a commander or anyone else to develop alert management criteria by inspection. Alert management must also be specialized for each user within the context of each distinct mission. Further, we cannot expect every user to master a complicated user interface to conduct a dialog with the system and indicate their preferences. Consequently, we prefer to present the user with a set of sample alerts and ask them to accept or reject the alert and its presentation. While this is extremely intuitive, such simple input requires a huge number of training sample alerts using traditional supervised training methods. Users would likely not be willing to train the system for the hours it would require, and it would be extremely difficult to generate a complete and realistic alert set.

To address this set of difficult constraints, we chose to use a novel supervised training algorithm that allows a user to train the IME on a drastically reduced alert set randomly generated from the multidimensional alert space. This algorithm starts with baseline management criteria, represented by a lookup table, and adjusts the table based on binary

feedback from the user. The baseline is created using an ontology of the alert space and subject matter experts to develop rules by inspection of the ontology.

Related Work

While this paper offers a system level solution that learns from a small subset of training samples using only binary input from a user, most previous systems that learn a user's preferences require a substantial dialog with the user. Many are built around the common desktop user interface metaphor. Seo and Zhang describe research that depends on a rich set of interactions with the user in a desktop environment while web browsing to filter the information and determine their interests [1]. It operates over a relatively narrow domain (scientific documents). Goecks and Shavlik also attempt to learn user interests by monitoring mouse clicks, scrolling behavior, and other interactions [2]. By unobtrusively observing the user's browsing behaviour, they are able to learn web pages that may be of interest to the user. A number of systems have been developed that use variations of reinforcement learning of user preferences, in some cases with inconsistent feedback at every step, and a complex set of states that may not be fully determined [3]. Thompson et al. describe a system that learns user's preferences based on a rich, natural conversational interaction [4].

Recent efforts in the Small Unit Operations Situation Awareness System (SUOSAS) program have used agent-based methods to filter and control the flow of information to the soldier, but primarily for non-tactical information such as email, XML messages, and logs [5]. The new Army initiative "Every Soldier a Sensor," (ES2) attempts to build a more formal structure for the creation of tactical intelligence, but does not address issues of information presentation and filtering [6].

The Information Management Engine

The Information Management Engine (IME) is a supervised learning system that rapidly determines from binary soldier input the output mode of the alert. The system shows the soldier an alert, and the soldier provides either an "accept" or "reject" response, or no response at all. Based on responses, the IME determines the soldier's preferences for presentation of all possible alerts by running an incremental multi-class support vector machine classifier. We describe three variations of this system: an interactive vignette-based training system, a truth-set automatic training system, and an interactive, incremental training system. For pilot experiments described later, we merged the vignette and incremental training systems into a single system that, in response to a user request, can update the classifier at any point during the training session. Figure 1 shows the system architecture for these variations. The vignette-based system was designed to accommodate user input while a specific simulated vignette was played out. On the other hand, the incremental version was designed to enable the system to update the classifier as each new user response was added to the classification set. In merging the two, we enabled the user to decide when to perform incremental updates during a simulated vignette. The truth-set automatic training was designed to take as input a specification of the user's preferences in rule form and construct a complete table of all possible alert input values and output preferences. This form was used primarily to measure the key performance metric for the IME, the learning rate.

Interface Ontology

We developed the interface ontology using three spirals of more sophisticated capability. In the first spiral, we focused on the development of an ontology that could support IME decisions for either display or no-display (see Figure 2). This enabled the focus to be on the prioritization of alert input values and the classifier method. In the second spiral, we generalized the ontology to support output modes of audio, text, imagery, and video in addition to no-display (see Figure 2). The final development spiral used specific heuristic constraints on the possible output modes (see Figure 3). Input modes could be transformed to a more basic output mode such that video -> imagery -> text -> audio (described below).

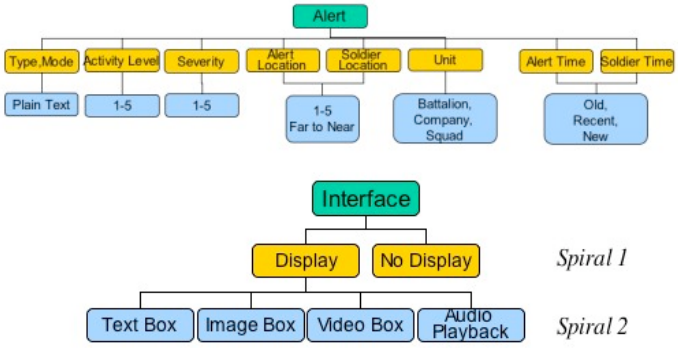

Figure 2. Ontology for Spiral 1 and 2 of IME.

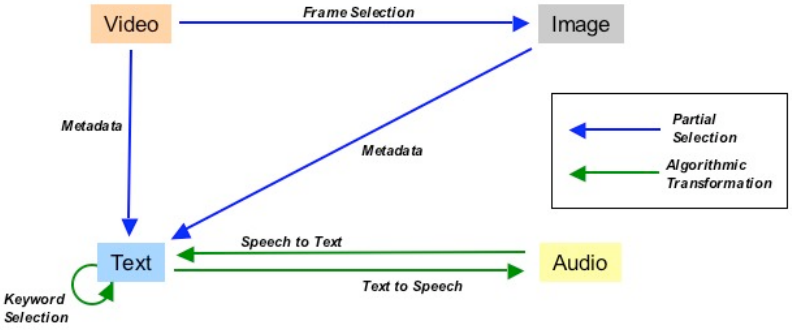

Figure 3. The IME can change an input mode to a different output mode by two means: partial selection chooses a new interface mode by eliminating data (i.e., full to partial display); algorithmic transformation processes the data from an interface mode into a new mode.

An alert was defined with the following input dimensions:

- Severity level (1 to 5, with 5 being high)
- Soldier activity level (1 to 5, with 5 being high activity)
- Relative Location (1 to 5, where 5 is close and 1 is far)
- Unit (1 to 5, 1 = lowest priority and 5 = highest priority)
- Display mode (1 = none, 2 = audio, 3 = text, 4 = image, 5 = video)

In addition, we defined the unit and alert locations relative to the receiver of the alert, requiring the alert values to be computed for that receiver based on a pre-defined policy. For the purposes of the experiments, we defined the policy as shown in Figure 4 below.

Using the above five dimensions, each with five discrete values, we have $5^5 = 3125$ possible alerts, each with an output value of +1 (yes) or -1 (no) indicating the user's preference for display. The training and "truth set" formats are very similar, and differ only in the number of values stored. Each line has the following format

classification 1:activity 2:severity 3:location 4:unit 5:mode

So for example, two lines could be

-1 1:5 2:1 3:3 4:2 5:2

+1 1:1 2:5 3:4 4:4 5:4

stating that the user says "reject" for an alert with activity=5, severity=1, location=3, unit=2, mode = audio, while the second line states that the user says "accept" for an alert with activity=1, severity=5, location=4, unit=4, mode = video.

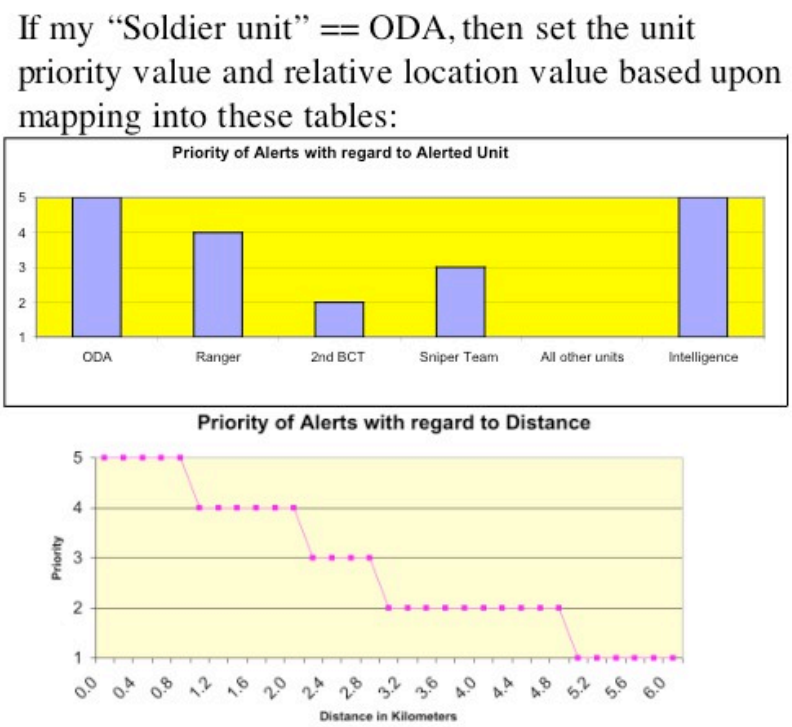

Figure 4. Unit and location values are computed relative to the receiver, in this case, for soldier unit ODA.

To store the results of the classifier, we use a truth table. The truth table has 3125 lines, containing all possible combinations of the five alert components. For each of those, the table states whether that alert has a yes or a no value. The IME decision engine reads the entire truth table and stores it into a lookup table, which is used to execute the decision logic for a particular alert. When an alert arrives, the IME looks up the truth table entry with those alert components. If the result is +1, then the system shows the alert in the specified output mode. Otherwise we examine, in order, the lower-numbered display modes to see if those have a positive output mode (+1). If they do, we convert to that lower-numbered mode and display. If none of those return +1, we don't display anything. For example, if the IME receives an alert vector of the form {-1 1:3 2:4 3:3 4:4 5:5} (i.e., severity=3, activity=4, location=3, unit=4, mode = 5 (video)), since it is scored –1, we first check severity=3, activity=4, location=3, unit=4, mode = 4. This also has output –1, so we check severity=3, activity=4, location=3, unit=4, mode = 3 (text). This vector has a defined output = +1, so we display this alert not as video but as text by displaying the text in the amplification field.

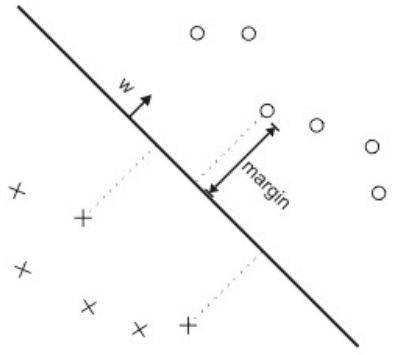

Figure 5. SVMs classify training samples by maximizing the margin between the training set and the decision surface.

Learning Classifiers

The IME uses a multi-class adaptation of the support vector machine classifier to generate a decision surface. Support vector machines (SVMs) are "large margin" classifiers that attempt to construct a decision surface that maximizes the

distance from the training set to the decision surface (see [7],[8]). The samples that are closest to the decision surface are called support vectors. SVMs have strong theoretical foundations and have been used successfully in a variety of real world tasks. In their simplest form, SVM decision surfaces are hyperplanes that separate the training data by a maximal margin such that all vectors lying on one side of the hyperplane (in a binary classification setting) are labeled as a –1 and all vectors lying on the other side are labeled +1 (Figure 5). The decision surface, or kernel, may be defined using a variety of different functions including polynomial approximations, radial basis functions, or user selected kernels. We used the freely available "SVM-Light" implementation of SVMs and modified it for multi-class use and incremental updates. For a more detailed discussion of SVMs, see the references.

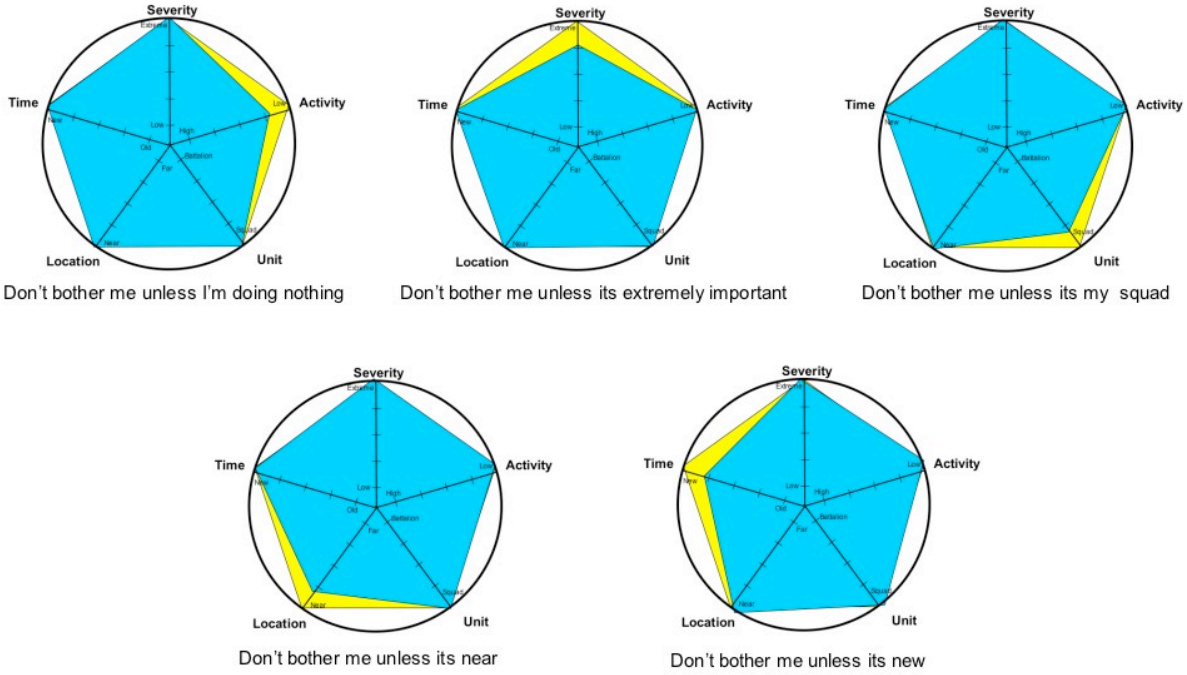

Figure 6. Historical reference models that vary in one dimension of the input vector and their interpretation. Blue regions correspond to no display decision, while yellow regions represent display.

Historical Reference Models

In a fully developed TAMS/IME system, we envision a set of case histories (historical reference models) that represent the various user preference characteristics for choice of display. A historical reference model is a set of decision surfaces for several users that capture essential differences or bias in preferences for alert display. They may change with different vignettes or other conditions. For example, Figure 6 shows single dimensional changes with a bias toward no display. These case histories would be used to initialize the IME prior to training, as described in the experiments section.

Interface Mode Issues

We considered a large number of issues in defining each of the possible output interface modes. Some of the important considerations are summarized here:

- Icons can reduce cognitive workload but require training. Too many can be confusing, leading to mistakes. Difficult to design for intuitive, easy to understand.
- Overloading different features can reduce use of display space but increase cognitive workload. For example, severity~color, time~brightness, unit~icon, severity~loudness.
- There is a tradeoff between divergent modes such as all text requiring reading or overloading features requiring training.
- Combinations of multiple interface modes increase awareness while increasing cognitive workload. For example, Visual + Aural, Visual+Aural+Tactile. interface methods include text+audio, map+text+image
- Different contexts requires different selections. Aural may be preferred for short message, immediate attention, or significant details while visual for spatially oriented meaning, noisy environment, need for silence.
- Video requires significant attention, and should be used only when activity level is low. Context is difficult to add with video, so use maps, 3D models, animations.
- Peripheral display vs full (foveal+peripheral) display. The best resting position for eyes is at -5° below horizon. Maintaining view at peripheral location (>15° from center) causes discomfort, eye strain and is useful for short duration interfaces (glance and go) but not for longer (reading, video). Next generation head-mounted displays are full displays (e.g. DARPA MANTIS).
- Multiple simultaneous alerts require filtering, prioritization, scheduling. For example, severity=5 before severity=1-4. Show pending alerts in summary form as prioritized list.
- How and when to interrupt the soldier is critical. For example, should we interrupt when activity=5 and severity=5?

In addition, several approaches to avoid include flashing alerts, use of layered menus (burying in a menu), requiring constant, repeated interaction, and adding useless or too much clutter.

Experimental Results

We considered a number of different metrics to characterize the performance of the IME. The primary metrics we decided on were accuracy, learning rate, and a reduction in the number of incorrect display choices. We assume that the user's opinion is not changing over time, or that they do not make mistakes in accepting/rejecting alerts. Conflicting responses are ignored.

Accuracy: Accuracy is defined as the number of alerts correctly presented divided by the total number of alerts. A value of 1 means we presented all the alerts received by the system correctly. A value of 0 means the IME presented no alerts correctly. The IME

deciding not to present an alert is considered a correct action, so if a soldier wanted to turn off all alerts and the IME learned not to show any, the accuracy would be 1.

Learning rate: The IME goal is to have high accuracy with a minimum of user input required. A learning rate of 1 is best, while a learning rate of 0 means learning is too slow for the level of accuracy achieved. So learning rate is defined as the accuracy minus the quotient of the number of user inputs and the possible number of inputs.

Learning Rate = Accuracy - (number user inputs / possible number inputs)

We cap it at 0 so there are no negative values. For example, if we ask no questions (i.e. use case histories) and have an accuracy of 1, then the learning rate is highest at 1. If we ask all possible questions and have an accuracy of 1, the learning rate is 0, meaning it took too much input. If we ask no questions and get none correct, we have a learning rate of 0. If we ask 10% of possible questions and get 90% of them correct, the learning rate is 0.8.

Reduction in incorrect choices: This metric only provides an indication that the IME is learning the user's preference, and is only useful in the context of a vignette or possibly in an operational setting. These type of training sessions have the value that they can more accurately represent the context in which an alert is received. On the other hand, there is no control over the specific order of the alerts since they flow based on the simulation (or live operations).

Our experiments using a truth set were either ontology-based or rule-based since we pre-defined the characteristics of the alerts and interface modes. In the case of a rule-base, we used a tool to take descriptive rules from a soldier and generate a truth set from the rules. This method has the significant disadvantage that the soldier must be able to define preferences across a 5 dimensional input space, and therefore is not a good method for constructing an actual decision classifier. It is useful for assessing the ability of the IME to learn a pre-defined truth set. For ontology-based methods, we applied the "presentation graph" reasoning to the ontology to decide which training examples are the best to ask about to maximize the learning rate. This approach requires active learning methods that we did not address in the seedling (but which we believe are central to developing a TAMS system).

In experiments we conducted using the truth set training architecture, we found that the percentage of correct choices for display was improved by the choice of which initialization was used, independent of the number of training samples. We also found

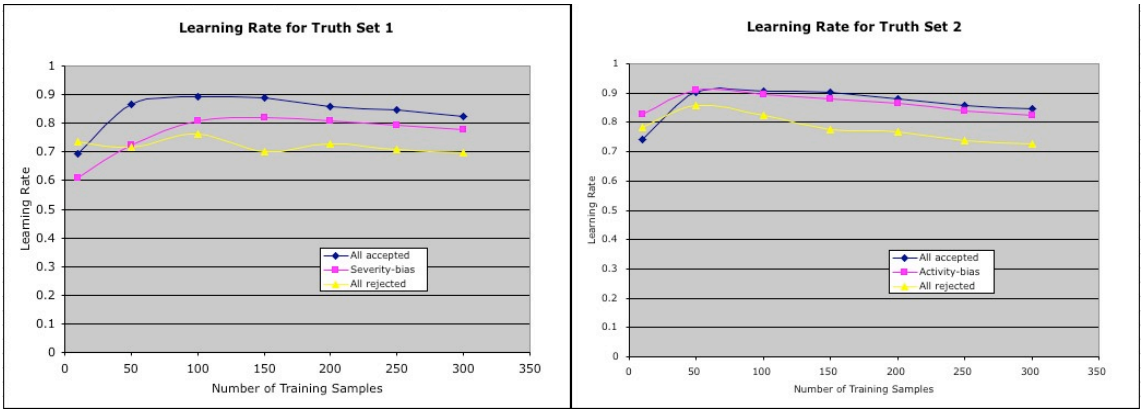

Figure 7. The learning rate for Tables 1 and 2 based on a total possible training sample size of 2500 alerts.

that as the number of training samples increased, the accuracy of the system improved (e.g. from 69% for only 10 training samples to 94% for 150, shown in Figure 7). In Figure 7, we show results for two experiments each with three examples based on different initializations: always display, never display, display under conditions defined by a set of rules.

Truth Set 1. Experimental results for initialization bias from historical reference models. In this case the truth table emphasis is on alert severity, and is defined by the following rules:

- If Alert Severity is 5, produce alert
- If Alert Severity is 4, unit generating alert is 2 or better and Activity Level is 4 or lower, produce alert
- If Alert Severity is 3, unit generating alert is 3 or better, Activity Level is 3 or lower and Mode is unpunished, produce alert
- If Alert Severity is 2, unit generating alert is 4 or better, Activity Level is 2 or lower, Alert Distance is Near and Mode is unpunished, produce alert
- If Alert Severity is 1, unit generating alert is 5, Activity Level is 2 or lower, Alert Distance is Near and Mode is unpunished, produce alert

Truth Set 2. In this case, the truth table emphasis is on soldier activity, defined by the following rules: If all of severity, unit, and distance are greater or equal to (activity – 1) and some of them are greater than or equal to activity level, accept all interface modes. However, if activity is 5, do not accept video mode.
Figure 7 shows the graph of the learning rate for Tables 1 and 2. Note that the learning rate improves rapidly up to a point and then levels out or drops.

The primary metric for the vignette-based user experiments was the number of rejects after re-training. This indicates the ability of the IME to correctly determine the user's preferences during the first training session. This type of training does not attempt to use a specific strategy for presentation of alerts that will maximize the performance of the system, and is not useful for measuring the learning rate. In these experiments, we had difficulty separating late responses, resulting in more 'rejects' from no response. If we remove the late responses from consideration, the number of rejects were slightly reduced after the re-training.

Discussion and Recommendations

The implementation of the IME using a multi-class SVM with incremental updates enables the system to learn the user's preferences fairly quickly for the types of alert input vectors we have defined. In an operational system, the IME may need to function with a larger number of dimensions for both input and output. For example, the choice of output modes of only four types does not address many of the possible output options that a user may prefer under different circumstances. Table 1 lists a number of possible variations on output modes that may provide improved understanding of alerts and could be used in training the IME.

Mode	Interface Options	Variables	Alert variations	Examples

Text	• No Display • Display Visual -Draw text at screen coordinates -In box -Across screen -Ticker (repeating) -Draw text icon at screen coordinates -Draw keyword at screen coordinates •Display Aural -Convert text to speech and play monaural -Convert text to speech and play spatial at world coordinates -Play "earcon" • Combination text display + audio	• Timing of display • Choice of location • Text size, style, color, (loudness) • Representation of icon (earcon) • Method of interaction with icon (earcon)	• Display relevant alert values as text • Severity maps to color (5=red, 1=green) • Unit displayed with icon • Show location with map or icon • Message newness maps to brightness of font (bold=new, dull=old)	S2 Report of suspicious activity Unit=2/6 Time=281100ZAPR2005 Location=38SMB38998580  Text only (top) and text on map (bottom)
Audio	• No Display • Display Visual - Convert to text and draw at screen coordinates - In box - Across screen - Ticker (repeating) - Keyword(s) • Aural -Play monaural -Play spatial at world coordinates -Play "earcon" • Combination audio + icon	• Timing of display • Choice of location • Audio loudness • Representation of earcon • Method of interaction with earcon	• Display relevant alert values as text (not good to use audio) • Severity maps to loudness • Unit displayed with icon • Show location with map or icon (best to use spatial) • Newness maps to ?	**Mode = Audio** Example = "Roof is clear"  Play audio (top) and link audio to earcon on map (bottom)
Image	• No Display • Display Visual - Draw at screen coordinates - In box - Relative to map (satellite image) - Relative to world -Draw icon at screen coordinates - Draw keyword at screen coordinates - Combinations (e.g. box, image, keywords, map)	• Timing of display • Choice of location • Image enhancements (e.g., reduced or enlarged) • Image overlays	• Display relevant alert values as text • Severity maps to color (5=red, 1=green) • Unit displayed with icon • Show location with map or icon • Message newness maps to brightness of font (bold=new, dull=old)	  Image in box (top) and combination of image with overlays (bottom)

Video	Same as image	• Timing of display • Choice of location • Playback speed • Video enhancements (e.g., reduced or enlarged) • Video overlays	• Embed values in video • Use other methods as before	 Sarnoff VideoFlashlight

Table 1. Variations and examples of output modes the user may select during training.

Thus far, we have assumed that the user's activity level is obtained through human observation. With additional input from biometric sensors such as body temperature, heart rate, respiration, movement, etc., the IME could learn a users activity level in either a supervised mode during training, or unsupervised during training or operations. We have also assumed that the unit and location values are determined using a "policy" based method in advance, and then when the alert is received by the soldier, these values are computed relative to the soldier's unit or location (and are different for each soldier). The soldier may prefer to establish their own policy, which would enable them to prioritize alerts differently for display than the policy.

Future IME functionality should move toward an active learning strategy combined with historical reference models. Active learning may be described as a method for minimizing the size of the training set while maximizing the value of each training example. This is analogous to an excellent teacher who is able to choose the training examples that will be fastest for a student to learn while providing the most understanding of the subject. Active learning systems must maintain an accurate model of the learning system's state so that they can choose the next training sample and then update their model based on the user's responses. The IME learning task is actually a case where the learning system (the IME) has access to a pool of unlabeled data (the multi-dimensional "truth set") and can ask an expert (the user) for the true label in a certain small number of instances. Furthermore, the performance of the learning system is assessed against the remaining training instances that are extracted from the database. In the learning literature this is called a pool-based, transductive active learning system (see [9]). These systems have been built using a variety of classifiers including SVMs.
Figure 8 shows the time required to train a user for a given training rate and number of training vectors. We clearly want to limit the training time to the smallest possible duration, while maximizing the accuracy of the system (conflicting goals). For a presentation rate of one alert training sample every 10 seconds (6 per minute), we can use only 180 training samples in a 30 minute session. A reasonable training accuracy of 1 alert presentation error every 15 minutes (~1% error rate assuming 6 alerts/minute) will require the use of accurate historical reference models to initialize the classifier.

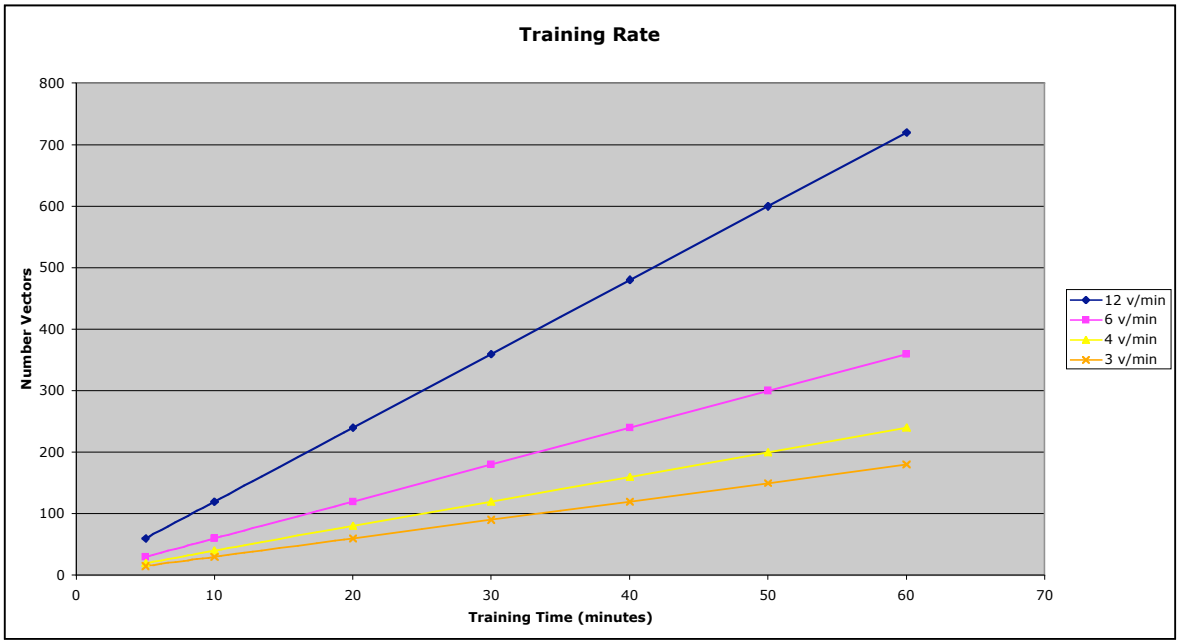

Figure 8. Training time for different rates and numbers of training samples.

Acknowledgements

This work was funded by DARPA. We thank Mr. Jeffrey Paul of DARPA, Mr. Bob Schulte of Solers Inc. (DARPA SETA Contractor), and Dr. Darrel G. Hopper of AFRL for their keen technical insights and guidance of this effort on behalf of the government. We also thank Robert Gray for his help in conducting experiments.

References

[1] Y. Seo and B. Zhang, "A reinforcement learning agent for personalized information filtering," Proc. of Int'l Conf. on Intelligent User Interface 2000 (IUI '2000), 2000, pp. 248 – 251.

[2] Jeremy Goecks, Jude Shavlik, " Learning Users' Interests by Unobtrusively Observing Their Normal Behavior" Proc. of Int'l Conf. on Intelligent User Interface 2000 (IUI '2000), 2000, pp. 248 – 251.

[3] Matthew Rudary, Satinder Singh and Martha Pollack, "Adaptive Cognitive Orthotics: Combining Reinforcement Learning and Constraint-Based Temporal Reasoning," In Proceedings of the Twenty-First International Conference on Machine Learning (ICML), pages 719-726, 2004.

[4] Thompson, C., M. Goker, P. Langley, "A Personalized System for Conversational Interactions," J. AI Research, Vol. 21, 2004.

[5] C. Thompson, B. Brown, P. Morris "SUO Communicator: Agent-based Support for Small Unit Operations." IEEE Integration of Knowledge Intensive Multi-Agent Systems (KIMAS), Cambridge, MA, October 1-3, 2003.

[6] Association of the United States Army, ES2: Every Soldier is a Sensor, discussion paper, August 2004

[7] Burges, C. “A Tutorial on Support Vector Machines for Pattern Recognition,” Data Mining and Knowledge Discovery, 2:121-167, 1998.

[8] Vapnik, V. “Statistical Learning Theory,” Wiley, 1998.

[9] Tong, S. and D. Koller, “Support Vector Machine Active Learning with Applications to Text Classification,” Journal of Machine Learning Research, November, 2001.