Online Active Learning for Automatic Target Recognition

Evan Kriminger, J. Tory Cobb, Member, IEEE,
and José C. Príncipe, Fellow, IEEE

Abstract

Automatic target recognition in side-scan sonar imagery is vital to many applications, particularly sea mine detection and classification. We expand upon the traditional offline supervised classification approach with an active learning method to automatically label new objects that are not present in the training set. This is facilitated by the option of sending difficult samples to an outlier bin, from which models can be built for new objects. The decisions of the classifier are improved by a novel active learning approach, called Model Trees, which builds an ensemble of hypotheses about the classification decisions, which grows proportionally to the amount of uncertainty the system has about the samples. Our system outperforms standard active learning methods, and is shown to correctly identify new objects much more accurately than a pure clustering approach, on a simulated side-scan sonar data set.

Index Terms

Automatic target recognition, sonar imaging, active learning.

I. INTRODUCTION

Side-look sonar images are used extensively in mine-countermeasure (MCM) applications where the sonar images are used to detect and classify various types of underwater unexploded ordinances (UXOs) [1] or sea mines [2]. It is not feasible for human operators to inspect each sonar image due to limitations in the underwater communication channel and the time burden
it would put on the operator. Automatic target recognition (ATR) systems aim to alleviate the burden from human operators by quickly and accurately finding targets of interest in these data.

The ATR problem possesses some interesting challenges. While many sonar images are acquired during scans of the seabed, few of these images actually contain objects of interest. Therefore, with respect to constructing models of these objects, the problem is data-poor. Features are extracted from the raw sonar images, which represent the shape of the objects present. To provide sufficient discriminative power, the dimensionality of the feature space can be quite large relative to the number of available samples. It is therefore difficult to estimate models, and self-labeled samples must often be used for training to improve the models. Furthermore, the survey platform acquires data from multiple viewing angles as it makes passes over the seabed. Thus, observations of the same object may be very scattered in feature space, owing to effects such as shadows and partial concealment by the seafloor substrate.

A full ATR system consists of sensor hardware, image/sonar processing, and the algorithms for object detection and classification. Many ATR systems incorporate some or all of these techniques. In [3], mine-like objects are detected using a Markov random field model, the object’s shadow is extracted using a co-operating statistical snake (CSS), and a classification decision is made by comparing to the shadows of known shapes. In [4], objects are compared to templates with a nonlinear extension of the matched filter based on correntropy. An active learning system is developed for use with side-scan sonar imaging systems in [5], in which a kernel classifier is built using a small number of samples labeled by a human expert. In [2], mine-like regions of the sonar images are detected, features are automatically selected, and objects are classified with a $k$-nearest neighbor attractor-based neural net classifier fused (or cascaded) with an optimal discrimination filter classifier.

The ATR system we propose here seeks to classify objects as they appear in a sonar survey, which is an online rather than batch classification task. An online approach is attractive in that it enables reactive behaviors from a sensing platform, such as changing a search pattern to improve detection performance mid-survey. Rather than maintaining a single classifier, an ensemble of classifiers is built adaptively, such that likely alternative hypotheses about new data may be captured. Unlikely samples are stored in a outlier bin, from which new objects can be detected using unsupervised techniques. An active learning scheme is proposed in conjunction with our ensemble, such that a small amount of human feedback may be utilized to improve performance.
in the online and nonstationary environment.

A. Supervised Methods

There are two main classes of algorithms commonly used in ATR: supervised and unsupervised. The supervised method of classification requires a fixed number of classes and a training set of labeled data points, from which the classifier can be built. Supervised learning is reliable, but only if the statistics of the training data are consistent with that of the data which will be encountered during operation. Moreover it is an oversimplification of the real world scenario of mine detection, where the number of different objects of interest is unknown a priori. Therefore, mine detection can only be solved with supervised techniques if only a few known types of mines are to be detected. Under this huge constraint (closed set classification), a training set can be built, comprised of templates for the desired mine types. Nevertheless, even in this case, threshold values have to be set to determine when a sample does not belong to any of the mine types.

The ability to make decisions on samples as they are acquired is important in ATR systems for a variety of reasons. The large amount of uninteresting data that is continuously acquired is a burden to store and process in batch mode. It is preferred to reject these points as they arrive, with an online system. In addition, the task may require real-time detection, and hence the decision on a sample cannot wait for more observations. Classification is a very natural way to group samples as they are acquired. While standard classifiers lack the ability to detect new objects, their ability to efficiently process ATR data online is quite useful.

Real ATR tasks are quite dynamic and the classification of desired objects is confounded by the presence of natural and man-made clutter. Assumptions on noise and stationarity are violated, which can limit the effectiveness of templates generated in a much different environment. In addition, it is often useful for the ATR system to detect novel object types other than those which have been specified initially (open set classification). Therefore, we must consider training data to be a limited picture of the possible objects that will be observed in practice.

B. Unsupervised Methods

The unsupervised method, commonly called clustering, seeks to find structure in the data itself, without labels. Observations of an object are assumed to be found in clusters, which can be distinguished from observations of different objects. This is not necessarily the case, since there
may be overlaps between objects, or one object may be split into more than one cluster. The latter case requires cluster merging, which is automatically done in classification because of the label information. Moreover, clustering algorithms require the specification of the number of clusters, or in our case, the number of objects which are to be detected. This is quite a large assumption, and cannot be made in the ATR problem as a group of observations may be tied to one or many objects. Without specifying the number of clusters, this is a considerably more difficult problem. A few clustering algorithms exist which do not require the number of clusters to be specified. Examples include DBSCAN [6], a density-based scheme in which clusters are determined by connected regions of sufficiently high density. Dirichlet process mixture modeling (DPMM) [7] is a non-parametric Bayesian clustering approach that fits the data to a possibly infinite number of component densities. The Gaussian mean-shift (GMS) [8] is a method which seeks modes of the underlying density function of the data, by iteratively improving an estimate of these modes. Although there is no need to specify the number of clusters, the free kernel parameter in the algorithm controls the number of clusters discovered. Furthermore, many unsupervised methods are not designed for online use. Clustering the full data set can become infeasible as more and more samples are acquired.

Nonetheless, unsupervised methods are highly desirable in the ATR problem, because they allow for the discovery of new objects, and do not require the creation of labels, which is expensive and a bottleneck. Even if only a few objects are of interest, and there exists many training examples for these objects, many uninteresting objects will be encountered in the field. The superfluous objects include large rocks and debris, biological clutter, and artifacts from turns and shadows. Not accommodating these various objects puts the burden of excluding them on a properly tuned threshold or anomaly detection system. It is difficult to set these parameters for use in a nonstationary environment, and even small changes can result in missed detections or false alarms. If models can be built for the clutter objects as well, then the problem becomes one of discrimination, which requires no arbitrary thresholds.

C. Active Learning

In both supervised and unsupervised approaches, feedback from a human expert may be incorporated to ensure a high level of performance and clarify any ambiguities in the past decisions. The problem of incorporating human feedback in a learning system is the domain of
active learning [9]. There are many domains in which human feedback is available, however, it is usually the case that interaction with the human operator is extremely expensive and the number of queries must be minimized. This cost is often the time it takes for the human expert to label the sample. When a small number of labels can be obtained, doing so has potential to stabilize the system created by self-learning and ensure good performance. Active learning methods determine which samples to send to the human operator and how to adjust existing models in response to the operator’s feedback.

In active learning, the problem is to determine whether or not to obtain a correct label for an unlabeled or self-labeled sample by querying a human operator. There are a few problem domains, which differ in which samples are considered candidates for query. In an online setting, each sample is considered for query as it is acquired. Often it is more natural to consider a pool of candidate samples, and select the one which offers the largest reduction in uncertainty in the space of all samples. In some special cases, points in the input space, which have not been observed, can be synthesized and considered for query, thus allowing even more control over the information that is gained by the classifier.

Active learning algorithms are typically based on a feature of the sample that captures its suitability for query. One of the most popular active learning frameworks is Uncertainty Sampling [10], in which the feature of interest is a measure of uncertainty with respect to the classifier. The sample with the highest measure of uncertainty is selected for query. Consider samples $x \in \mathcal{X}$, where $\mathcal{X}$ is a pool of candidates for query. Let the class label be $y \in \mathcal{Y}$, with class posterior $P(y|x)$. Let the MAP class be

$$\hat{y}_1 = \arg \max_y P(y|x),$$

(1)

and the “runner-up” class be

$$\hat{y}_2 = \arg \max_{y \in \mathcal{Y}\backslash \hat{y}_1} P(y|x).$$

(2)

Then, some example Uncertainty Sampling strategies are least confident, margin sampling, and maximum entropy, represented by
\[ x^*_{\text{LC}} = \arg\max_x \{1 - P(\hat{y}_1|x)\} \] (3)

\[ x^*_M = \arg\min_x \{P(\hat{y}_1|x) - P(\hat{y}_2|x)\} \] (4)

\[ x^*_H = \arg\max_x \left\{ -\sum_{y \in \mathcal{Y}} P(y|x) \log P(y|x) \right\} \] (5)

respectively. In all these strategies, Uncertainty Sampling chooses the sample for which the classification decision is least clear.

The second major class of active learning algorithms is Query-by-Committee [11]. In the Query-by-Committee framework there exists an ensemble of \( N \) models \( \theta_1, \ldots, \theta_N \), each representing a classifier that is consistent with the labeled training data, but not with differing opinions on the unlabeled samples. Each model \( \theta_i \) in the ensemble provides its own label \( y_i(x) \) for a candidate sample \( x \). Therefore, samples which engender the most conflicting hypotheses among the models are considered interesting samples worthy of query. One measure of conflict is the vote entropy, and applying this rule makes the following query decision,

\[ x^*_\text{VE} = \arg\max_x -\sum_{y \in \mathcal{Y}} N_y \log \frac{N_y}{N} \] (6)

where \( N_y = \sum_{i=1}^N \mathbb{1}_y(y_i(x)) \), i.e. the number of models voting for sample \( x \) to be in class \( y \).

Other methods include Expected Error Reduction [12], which estimates the expected error of each sample over all possible decisions for that sample, and density methods [13], which considers both the uncertainty of a sample, and the number of unlabeled samples in its neighborhood, thus better characterizing the impact of that query.

These frameworks make query decisions based on the model, which in many cases, has been estimated from a small training set. If the training set is small enough, then self-labeled samples must also be used to train the classifier as it makes decisions. This poses a problem for active learning algorithms because we are now attempting to evaluate the uncertainty or the impact of samples with respect to a model that has been estimated from these very same uncertain samples.

One issue is that if a query is made for sample \( x(t_1) \), then the model has changed and all decisions made for times \( t > t_1 \) need to be re-evaluated. This can lead to other samples
requiring queries. The query at time $t_1$ also has information for samples at times $t < t_1$. Perhaps information about the queried sample changes a decision made on an early sample. It is obvious that self-trained models offer particular difficulty for active learning methods, as each query launches a cycle of retraining and reclassifying all samples. Not only is this computationally intensive, but this system is not very efficient with its queries. To improve the models enough to accurately determine where a query is truly needed may require more human intervention than the problem allows.

II. PROPOSED SYSTEM DESCRIPTION

We introduce an ATR system which is designed to exploit as much external information about the problem as possible, without making any assumptions that will be violated in the dynamic environment. A training set is used, condensing the best knowledge available on a set of targets of interest, which gives the system an idea of the dispersion present in the data as a characteristic of the environment and sensors, rather than simply a closed set of templates for future objects. Human feedback can be incorporated to verify previous decisions, but the system does not require this interaction, and can utilize it whenever it is available. The ATR system is designed with online functionality in mind, such that each new observation is processed without reprocessing the full data set. Previously unseen objects are detected, and no assumptions are made on the number of objects that will be encountered, allowing deployment in any unknown environment.

The system consists of three components:

- A classifier with information-theoretic features [14], which determines if a new observation belongs to an existing object or should be placed in the outlier bin.
- An active learning algorithm, which maintains an ensemble of these classifiers, representing alternate hypotheses. When human feedback is available, this feedback removes incorrect hypotheses.
- A new object detection algorithm, which works on the outlier bin to determine if a subset of those samples represents a new object.

The flow chart for processing a sonar observation with our system is shown in Figure 1.
Fig. 1. A sonar image is preprocessed, resulting in a new data point in feature space. The Model Trees active learning algorithm consists of an ensemble of classifiers. Querying the human results in a pruning of these models. New object detection is continually run, altering the models as new objects are detected.

A. Classification with Thresholds

In the ATR problem, a class represents an object, and the samples which make up the class represent the multiple observations of that object. Let the feature vector of an observation be $x$. Let there be $C$ known classes, designated by the label $y \in \{1, \ldots, C\}$. For each known class, $y$, we have a likelihood $f_{\theta_y}(x)$, where $\theta_y$ is an estimate of the likelihood parameters. These parameters are estimated with the Dynamic Tree (DT) estimator developed in previous work [15]. Dynamic trees are graphical models, where a group of sensor measurements are represented probabilistically as the leaf nodes in a forest of DTs, with root nodes representing the objects themselves. DTs are particularly well-suited to online estimation and in our tests have performed much better than methods such as maximum likelihood when very few training samples are available relative to the dimensionality of the data. In this paper the $f_{\theta_y}(\cdot)$ are multivariate normal distributions.

Rather than using the generative model for maximum likelihood or maximum a posteriori
classification, we use an information-theoretic feature based on a probability divergence, $D$. For each class, this feature is the change the likelihood undergoes when its parameters are re-estimated with the inclusion of the sample to be classified, $x_{new}$. The classification procedure with this feature is described in Algorithm 1.

**Algorithm 1 Classification with divergence of likelihoods feature**

| Input: | Sample to classify $x$, likelihood parameters $\{\theta_i\}_{i=1}^C$, training sets $\{X_i\}_{i=1}^C$ |
| Output: | Class label $y$ |
| for $i = 1$ to $C$ do |
| $X'_i \leftarrow X_i \cup x$ |
| $\theta'_i \leftarrow \text{DynamicTree}(X'_i)$ |
| $\phi_i(x) \leftarrow D(f_{\theta'_i} \parallel f_{\theta_i})$ |
| end for |
| $y \leftarrow \arg \min_i \phi_i(x)$ |

This divergence-based feature performs comparably to negative log-likelihood for classification. However, the value of divergence quantifies the impact a sample has on our existing models, which is important in our data-sparse ATR problem, where individual samples can influence the model substantially. It is therefore a more natural feature for selecting outlier bin thresholds than likelihoods, as will be developed below.

For $D$, the Cauchy-Schwartz (CS) divergence is used instead of the Kullback-Leibler because it is more efficient to compute for mixtures of multivariate normal distributions [16]. For two continuous pdfs, $f$ and $g$, the CS divergence is defined to be

$$D_{CS}(f; g) = -\log \left( \frac{\int f(x)g(x)dx}{\sqrt{\int f(x)^2g(x)^2dx}} \right).$$  \hspace{1cm} (7)

Classification with DT estimators and CS divergence features was introduced in prior work [17].

In this formulation, we have assumed that the number of classes is known and that there exist training samples for each class. This is only possible when the set of objects to be detected is known in advance and training examples of these objects from multiple views are available. For an ATR system capable of detecting new objects as well as known ones, a means of accommodating new classes of objects must be developed.

For some samples, the likelihood divergence is high for all classes. This is the case when a new object has been detected, for instance. Rather than assign these samples to the most likely class,
we set them aside in an outlier bin. Binned samples can be grouped with other bin samples if a new object is present, or simply remain in the bin instead of corrupting existing object models if they are not observations of interest. We establish a threshold $T_i$ on the likelihood divergence $\phi_i$ for each class $i$, based on the maximum value that is encountered in the training set.

$$T_i = \max_{x \in X_i} \phi_i(x)$$  \hspace{1cm} (8)

We define $y = 0$ for samples in the outlier bin and there is no accompanying $\theta_0$. The likelihood divergence for the bin, $\phi_0(x)$, will be defined as the threshold of the minimum divergence class

$$\phi_0(x) = T_{\arg \min_i \phi_i(x)}.$$  \hspace{1cm} (9)

This definition will be used for stochastically designating samples to the outlier bin in our active learning algorithm.

**B. Model Trees for Active Learning**

Since sonar scans may only acquire a few usable views of each object before moving on, it is necessary to use self-labeled samples when estimating models. Classifiers that learn from their own decisions face the possibility that the parameters can become increasingly poor if an incorrect label is applied, and performance will drop instead of improving. When feedback from a human operator is available, it can be used to ensure that a classifier practicing self-learning remains reliable.

The Query-by-Committee framework is a promising ATR strategy for ATR, since there is no longer the difficulties of retraining, reclassifying, and reapplying active learning algorithms each time a query is issued, as the committee members are simply removed. The difficulty in this framework is in generating a diverse committee of models, each consistent with the training data. If the range of hypotheses present in the committee does not capture the uncertainty of each sample, then disagreement among the various models in the committee will not serve as a good basis for query, since this disagreement is only founded on the randomness with which the committee was synthesized. We propose to fuse the philosophies of Uncertainty Sampling and Query-by-Committee, by dynamically generating committees that reflect the uncertainty of
the samples. We adapt this framework to the problem of online classification with self-learning and rectify one of the unaddressed problems in active learning, that is, when to make a query.

Our active learning algorithm *model trees* is initialized with the likelihoods $f_{\theta_y}$ for each class $y \in \{1, \ldots, C\}$, which were estimated from a training set of known objects. For a new sample $x$ that is encountered, the features $\phi_i(x)$ for $i = [0, \ldots, C]$ are calculated as in Algorithm 1. $\phi_i(x)$ is converted into a Gibbs distribution,

$$p(y = i | x) = \frac{\exp(-\beta \phi_i(x))}{\sum_{i=0}^C \exp(-\beta \phi_i(x))},$$  \hspace{1cm} (10)

where $\beta > 0$ is a temperature constant controlling the degree to which the distribution is flattened.

Typically, we would use $x$ to update the likelihood parameters of the class

$$y^* := \arg \max_i p(y = i | x)$$

(or ignore the sample if $y^* = 0$). However, if the decision is very difficult, i.e. $p(\cdot | x)$ has a small margin, then not as much confidence can be put on that decision. It is advisable then, to consider other likely alternatives. Rather than making a hard decision on $x$, a soft decision can be made by drawing from 10. Let the result of this draw be $y_{alt}$. If $y^*$ is the same as $y_{alt}$, then only one model is updated, and the number of hypotheses stays the same. If $y_{alt}$ is a different outcome, then we first update $\theta_{y^*}$ with $x$, then update $\theta_{y_{alt}}$. The superscripts $(1)$ and $(2)$ represent different hypotheses which form our committee.

By continuing in this way, a “tree” of models is built, with each path through the tree representing different hypotheses. When many of our classification decisions carry a large amount of uncertainty, the number of branches in the tree will grow. This is indication that the classifier could be significantly improved with feedback from a human operator. By setting a limit on the number of branches, we have a means of determining when to initiate a query that is related to the amount of uncertainty in our models. The sample that is to be queried should best reduce the number of competing hypotheses. The amount of disagreement among the hypotheses can be quantified by the entropy of the labels given to that sample.

Let the label given to $x$ by the $j^{th}$ member of the committee be $y^{(j)}(x)$. We can construct a histogram of the committee’s votes to compute the entropy for each sample. The label distribution
across a committee of size $M$ is given by

$$p_{com}(Y(x) = l) = \sum_{j=1}^{M} \delta(l, y^{(j)}(x)). \quad (11)$$

for a label outcome $l$ and where $\delta$ is the Kronecker delta. Then the Shannon entropy for the label of $x$ is

$$H(Y(x)) = \sum_{i=0}^{C} -p_{com}(Y(x) = i) \log p_{com}(Y(x) = i). \quad (12)$$

The sample chosen for query is simply the one with maximum label entropy,

$$x_{\text{query}} = \arg \max_x H(Y(x)). \quad (13)$$

The sample in question is then presented to the human operator, who specifies its true label. All branches which incorrectly labeled the sample are removed, thus pruning the tree, with the more accurate models remaining. If all hypotheses are assumed to be equally likely, then the maximum label entropy query decision maximizes the expected number of branches which are removed from the tree.

**C. New Object Detection**

Each hypothesis in our committee places sample in their own outlier bins. It is from these bins that the samples comprising a new object are detected as clusters, and a model is built for that object. The statistics of these undesired observations are unknown, but at least from the known classes we have some estimate of the variation present in a group of observations. Since clustering algorithms typically have a parameter which determines the scale on which clusters are identified, we can tune this parameter to the known classes. This should provide a more problem specific balance of false positives and missed detections than if the parameters were chosen a priori.

To determine these clustering parameters, first a base clustering algorithm is chosen. This algorithm must be nonparametric (the number of clusters does not need to be specified) because it is unknown how many objects exist in the outlier bins. The parameters vary for each base
Algorithm 2 Model Trees active learning

$\textbf{Input:}$ Temperature constant $\beta$, max committee size $\text{MaxPaths}$, likelihood parameters $\{\theta_i\}_{i=1}^{C}$

$\textbf{Output:}$ Committee of labels

$\textbf{Initialize:}$ $M = 1$

$\textbf{for}$ each new test sample $x$ $\textbf{do}$

$\quad \textbf{for}$ $j \leftarrow 1 \textbf{ to } M$ $\textbf{do}$

$\quad \quad \text{Compute } p(\cdot|x)$ (Eq. 10)

$\quad \quad y^* \leftarrow \arg \max_i p(y = i|x)$

$\quad \quad \text{Update } \theta_{y^*}^{(j)} \text{ with } x$

$\quad \quad \text{Draw } y_{\text{alt}} \sim p(\cdot|x)$

$\quad \quad \textbf{if } y_{\text{alt}} \neq y^* \textbf{ then}$

$\quad \quad \quad M += 1$

$\quad \quad \quad \theta_{y_{\text{alt}}}^{(M)} \leftarrow \text{update } \theta_{y^*}^{(j)} \text{ with } x$

$\quad \quad \textbf{end if}$

$\quad \textbf{end for}$

$\textbf{Queries}$

$\quad \textbf{if } M > \text{MaxPaths} \textbf{ then}$

$\quad \quad \text{Find } x_{\text{query}}$ (Eq. 13)

$\quad \quad \text{Receive query response, } y_{\text{true}}$

$\quad \quad \text{Delete committee members } \{j \mid y^{(j)}(x_{\text{query}}) \neq y_{\text{true}}\}$

$\quad \textbf{end if}$

$\textbf{end for}$

Algorithm 3 New object detection

Run in Algorithm 2 for each member of the committee at periodic intervals

$\textbf{Input:}$ Unlabeled data set $\mathcal{X}^u$, labeled data set $(\mathcal{X}^l, \mathcal{Y}^l)$, base clustering method $\text{Cluster}$, set of candidate clustering parameters $\Sigma$

$\textbf{Output:}$ Labels for $\mathcal{X}^u$

$\textbf{for}$ $\sigma$ in $\Sigma$ $\textbf{do}$

$\quad \mathcal{Y}_{\sigma}^l \leftarrow \text{Cluster}(\mathcal{X}^l; \sigma)$

$\quad r_{\sigma} \leftarrow \text{RandInd}(\mathcal{Y}_\sigma^l, \mathcal{Y}_\sigma^l)$

$\textbf{end for}$

$\quad \text{Compute } \sigma^* \leftarrow \arg \max_{\sigma} r_{\sigma}$

$\quad \mathcal{Y}^u \leftarrow \text{Cluster}(\mathcal{X}^u; \sigma^*)$

clustering method: kernel bandwidth $\sigma$ for GMS and concentration parameter $\alpha$ for DPMM.

By applying the clustering algorithm to the data for known classes, we can find the parameter set which yields a clustering of the data which best matches existing labels. With the properly set parameter, the outlier bin can be clustered at desired times. The stored labels within the committee are changed when their outlier bins are clustered. See Algorithm 2 and 3 for a full
description of Model Trees with new object detection.

III. EXPERIMENTS

We test the active learning and new object recognition components of our ATR system and compare with other methods. The data set used is fully described in [14] and is a partially synthetic simulation of a target field of objects. Each trial will run on a group of 3 closely spaced objects, with 24 observations of each object. The feature vector is 8-dimensional. The first two features are the approximate $x - y$ coordinates of the observations. These coordinates are synthetic and are drawn from gamma, beta, and normal distributions. The difficulty of the classification task varies throughout the data set and is largely a function of the closeness of the object centers. The next three features are shape features acquired by applying edge detection techniques on the raw sonar images and extracting edges. There are 3 objects found in our data set, cones, cylinders, and boxes. The final three features are surface features, which characterize the seafloor textures and are also acquired from the raw sonar images. There are 6 seafloor types, with two varieties each, of grass, rock, and sand. Some example sonar images are shown in Figure 2. All possible combinations of the 3 shapes, 6 backgrounds, and coordinate distributions are present in the data set. Thus, the 3 objects that appear in each trial may have the same or different backgrounds and shapes. It is the most difficult to detect distinct objects when they possess the same shape and appear over the same background.

Fig. 2. Example sonar images with examples of rock, grass, and sand style seabed textures and cylinder, box, and cone shaped objects.
One natural way to evaluate the benefit of an active learning methodology is to characterize how the models improve over time. To measure this improvement, in each trial, we measure the divergence between the learned model at each time step in testing, and the best possible model. In this case, the best model is the one which is trained with all the correctly labeled samples. By computing this divergence for each class, and taking the norm of the divergences of all classes, we have a metric for evaluating the classification system. If we define the estimated likelihood for class $y$ to be $\hat{p}_y := f_{\theta_y}(\cdot)$ and the best likelihood to be $p_y^*$, then our metric is,

$$\sqrt{\mathcal{D}_{CS}(\hat{p}_1; p_1^*)^2 + \ldots + \mathcal{D}_{CS}(\hat{p}_C; p_C^*)^2}$$  \hspace{1cm} (14)$$

This metric is computed for each time step, as the system trains its models with the DT, using self-labeled testing samples. By plotting the metric, averaged over all trials, as a function of time, the progress of learning can be visualized. In fact, this plot can reveal interesting behavior that is not captured in the classification error rate. In Figure 3, plotting the model’s divergence over time reveals an instance where a classifier utilizing self-learning actually deteriorates as it encounters new samples. A data set consisting of 3 classes, drawn from 2-dimensional Normal distributions, was broken into a training set with 8 samples per class, and a testing set of 126 samples. As seen in the figure, the divergence grows as more samples are classified and used to update the models. When a simple implementation of Uncertainty Sampling with a 9% query
Active learning causes the learned models to approach the true models much more quickly than with a plain classifier. With Model Trees, queries are utilized much more efficiently than with Uncertainty Sampling.

<table>
<thead>
<tr>
<th>Queries</th>
<th>Plain</th>
<th>US</th>
<th>US+Density</th>
<th>EER</th>
<th>MT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>83.2%</td>
<td>84.2%</td>
<td>84.3%</td>
<td>84.3%</td>
<td>86.1%</td>
</tr>
<tr>
<td>2</td>
<td>83.2%</td>
<td>85.9%</td>
<td>85.9%</td>
<td>86.1%</td>
<td>86.9%</td>
</tr>
<tr>
<td>3</td>
<td>83.2%</td>
<td>86.2%</td>
<td>86.4%</td>
<td>87.1%</td>
<td>87.0%</td>
</tr>
</tbody>
</table>

**TABLE I**

The plain classifier is compared with Uncertainty Sampling (US), Density-Based Uncertainty Sampling, Expected Error Reduction (EER), and Model Trees (MT). The scenarios are 1, 2, and 3 queries, made at regular intervals of 10 time steps.

In Figure 4, the model divergence is plotted over time to compare the Model Trees algorithm with Uncertainty Sampling and with the classifier without active learning. The plot is an average over 200 randomly selected trials from the simulated sonar data set. The DT+likelihood divergence classifier is the default classification scheme. The results for query rates of 9% and 16% are shown. These query rates were determined by setting the $\beta$ parameter of the Model Trees algorithm to 6 and 3, respectively. The maximum number of paths parameters was fixed at 20. As the Uncertainty Sampling algorithm does not determine when to query automatically, it was allotted as many queries as made by the Model Trees. A large improvement can be seen by employing active learning, however Model Trees is more query efficient, producing similar results with a 9% query rate as Uncertainty Sampling achieves with 16%.

In Table I, the classification rates are shown for our classifier equipped with several forms of
active learning. The “plain” classifier is the standard classifier used throughout this paper, and uses no active learning. We select three standard active learning methods that can be used with our DT classifier: Uncertainty Sampling, Density-Based Uncertainty Sampling, and Expected Error Reduction, and compare them with Model Trees. For each of the three objects in our simulated sonar data set, 10 samples are designated for training, and 14 remain for testing. The 42 testing samples are presented in random order, and the models for each object are learned online via self-learning with the classified samples. For the sake of determining an accurate classification rate for the Model Trees algorithm, no samples are assigned to the outlier bin, which is ensured by setting its probability to 0. The maximum number of paths in our tree is set to 20, and \( \beta \) is set to 8. However, the query rate is fixed, so these parameters only affect the number of models in our tree. The implementations of the other three active learning algorithms are taken from [9], and are parameter free (other than the query times).

For comparison we evaluate the algorithms with 1, 2, and 3 queries per trial. These queries are made in intervals of 10 time steps, thus queries would be made at time steps 10, 20, and 30 in the case of 3 queries per trial. At the time of query, all encountered testing samples are considered candidates for query. Querying a sample returns the correct object label of that sample. The classification rate is calculated as the number of correctly classified samples at the end of testing, out of the 42 testing samples. The results are averaged over 100 trials.

All active learning methods significantly improve the classification rate of the plain classifier. Model Trees is shown to perform particularly well when very few queries are possible, while the other methods improve quickly as the number of queries rises. This can be attributed to our use of fixed queries times in this experiment, rather than querying at the times automatically specified by Model Trees. By making unnecessary queries, the diversity of our ensemble can be decreased. As a Query-by-Committee method, the quality of the Model Trees query decision depends on disagreement amongst the models of the committee. If there is no disagreement, then a query is not likely to have as much impact.

The sensitivity of the query rate of Model Trees to its two parameters, \( \beta \) and maximum number of paths, is shown in Figure 5. This figure is produced using the same testing setup as was used for calculating the classification rates, and the plotted query rates are averaged over 50 trials. The query rate reduces exponentially in \( \beta \), while changing the maximum number of paths in the tree merely induces a shift in the query rate. In practice, the maximum number of paths can be
set based on the computational requirements, and $\beta$ can be set to induce the desired query rate.

We evaluate the ability of the new object detection system in a final experiment in which only one of the three objects in each trial is presented in training (10 training samples). Observations of the unknown two objects arise only during the testing phase, mixed with the remaining 14 observations of the known object. We evaluate algorithms based on their ability to correctly detect the presence of the new objects from the testing observations. We compare our classification-based approach aided by our new object detection method with the pure clustering method. We use the Gaussian Mean Shift both as the base clustering method in our new object detection and as a separate algorithm for determining the number of objects in a set of observations. In the latter case, a purely unsupervised method is used, in which all samples are clustered (including training data). It may seem redundant to cluster the training data as well, but these samples help ensure that the known objects are clustered properly.

In Table II, we show the statistics of the number of objects detected by each method across 100 trials. The kernel bandwidth of the GMS was set with a heuristic rule based on the mean nearest neighbor distance of the clustered samples. The ensemble size in our new object detection algorithm was set to 10. The parameters of the Model Trees algorithm are the same as in the classification rate experiment. In one scenario, no queries are allowed, while in the other, a single query is made at the end of testing. The outlier bin is enabled, with the thresholds given by (8). At every time step, the new object detection method is run on the outlier bin samples of each
TABLE II
THE TRUE NUMBER OF OBJECTS IS 3, BUT LABELED DATA FOR ONLY ONE OF THESE OBJECTS IS PROVIDED. THE PERFORMANCE OF GAUSSIAN-MEAN SHIFT (GMS) CLUSTERING ON THE WHOLE DATA SET IS COMPARED WITH OUR METHOD OF MODEL TREES WITH NEW OBJECT DETECTION (NOD), AND OUR METHOD WITH 1 ALLOTTED QUERY.

<table>
<thead>
<tr>
<th>Objects Detected</th>
<th>GMS</th>
<th>MT+NOD</th>
<th>MT + NOD + Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10%</td>
<td>4%</td>
<td>0%</td>
</tr>
<tr>
<td>2</td>
<td>22%</td>
<td>4%</td>
<td>3%</td>
</tr>
<tr>
<td>3</td>
<td>54%</td>
<td>89%</td>
<td>95%</td>
</tr>
<tr>
<td>4+</td>
<td>14%</td>
<td>3%</td>
<td>2%</td>
</tr>
</tbody>
</table>

branch in the tree.

In this experiment, our method greatly outperforms the pure clustering approach, which reflects the amount of information utilized by our method in comparison to a fully unsupervised approach. By first classifying the more obvious samples, and then only clustering the outlier bin, there is much less room for error. Allowing just one query improves the performance even further, as there is higher label entropy for samples which have been labeled as belonging to both old and new classes in the tree.

IV. CONCLUSION

In this paper we have introduced a method for detecting and classifying objects in sonar data, which is not limited to a template set of objects and uses human feedback, when available, to improve the fusion of observations. Our active learning method, Model Trees, is a fusion of Query-by-Committee and Uncertainty Sampling approaches, generating a tree of likely hypotheses about the data. Interaction with the operator removes incorrect hypotheses. The decision of when to query is made automatically by the algorithm, which improves upon existing methods, in which this decision is arbitrary. New objects are recognized from samples that are set aside in the outlier bin using an approach which estimates good clustering parameters from existing samples. Model Trees is shown to greatly improve a classifier, and outperforms three standard active learning methods. Our new object detection method is much more reliable than clustering approaches, as the outlier bin offers a more facilitating subset of the data for recognizing new clusters, and the clustering parameters may be effectively set by using information present in the previously classified samples. The Model Trees algorithm works seamlessly with the new object
detection, thus allowing for an ATR system which can function autonomously in an unknown environment, and confirm its decisions with human feedback, when available.

REFERENCES


Evan Kriminger  Evan Kriminger is a Ph.D. candidate in electrical engineering at the University of Florida in Gainesville, FL. He received an M.S. degree in electrical engineering from the University of Florida in 2010, and a B.S. in engineering science from the University of Miami in Coral Gables, FL in 2009.

As a member of the Computational NeuroEngineering Laboratory (CNEL) he has done work on time series analysis, reinforcement learning, and active learning. His dissertation work focuses on active learning methods to discover the true number of classes in open-set classification problems, such as automatic target recognition.

Tory Cobb  J. Tory Cobb (S’99–M’01–SM’13) received the B.S. degree in electrical engineering from the United States Coast Guard Academy, New London, CT, USA in 1994, the M.S. degree in electrical engineering from Auburn University, Auburn, AL, USA, in 2001, and the Ph.D. degree from the University of Florida, Gainesville, FL, USA, in 2011.

From 1994 to 1999 he was an active duty officer in the Coast Guard. Since 2001 he has been employed as a Research Engineer at the Naval Surface Warfare Center, Panama City, FL, USA. He has served as Principal Investigator or Co-Principal Investigator for various automatic target recognition and sensor fusion projects funded by the Office of Naval Research. His research interests include statistical modeling of sonar signals with applications to automatic target recognition, automated environmental characterization of seabeds in side-look sonar images, and sonar image segmentation algorithm development.

Dr. Cobb is an Associate Editor of the IEEE Journal of Oceanic Engineering.

Jose Principe  Jose C. Principe (M’83–SM’90–F’00) received the Ph.D. degree in electrical engineering from the University of Florida, Gainesville, in 1979.

He was a Professor at the University of Aveiro, Portugal, from 1980 to 1987. He is Distinguished Professor of Electrical and Biomedical Engineering at the University of Florida, Gainesville, where he teaches advanced signal processing and machine learning. He is BellSouth Professor and Founding Director of the University of Florida Computational NeuroEngineering Laboratory (CNEL). His research interests are centered on advanced signal processing and machine learning, brainmachine interfaces, and the modeling and applications of cognitive systems. He has authored five books and more than 200 publications in refereed journals, and over 400 conference papers. He has directed 68 Ph.D. dissertations and 67 Masters theses.