A NEW ENVIRONMENTALLY ADAPTIVE CLASSIFICATION ALGORITHM FOR UNDERWATER MINES IN SAS IMAGERY

David P. Williams[a] and Elias Fakiris[b]

[a]NATO Science and Technology Organization, Centre for Maritime Research and Experimentation, Viale San Bartolomeo 400, 19126 La Spezia, Italy
[b]University of Patras, 26504 Patras, Greece

Email: williams@cmre.nato.int; fax: +39 0187-527-330

Abstract: *A new classification framework is developed to explicitly account for environmental dependence in the underwater mine classification problem. Information describing the local environment of each object is quantified in the form of a novel feature characterizing the seabed; this auxiliary feature is then used in the classifier construction stage to automatically adjust the relative contribution of each training data point to the learning process. Multiple classifiers are constructed, with each classifier associated with a particular range of environmental feature values. The class prediction for a new unlabeled data point is a weighted average of the classifiers' outputs, with the relative weightings determined by environmental similarity. An extension to handle the case in which the environment is characterized by multiple features is also provided. Importantly, all algorithm parameters are learned automatically from the data itself, with no specialized tuning required. Experimental results on an underwater mine classification task using a large database of synthetic aperture sonar (SAS) imagery collected at sea demonstrate the promise of the proposed approach.*

Keywords: *Classification, synthetic aperture sonar (SAS), environmental adaptation, underwater mines, automatic target recognition (ATR).*

1. INTRODUCTION

An implicit assumption made in most statistical learning algorithms is that the labeled data used to train a classifier will be representative of – i.e., generated by the same underlying distribution as – the unlabeled testing data for which predictions must subsequently be made. For the task of discriminating underwater mines from natural clutter objects in sonar imagery, this assumption of data homogeneity can be violated because of a strong dependence on the environment in which the data is collected. For instance, it is common to encounter different types of clutter objects at different geographical locations.

Additionally, the seabed *composition* can also lead to mismatched feature distributions. For example, features that are based on the segmentation of an object's shadows or highlights can be systematically distorted by the shadows and highlights associated with background sand ripples. The undesirable influence of the environment can also manifest in ostensibly robust features, such as those tied to physical properties of an object. Consider a feature that measures the height of an object on the seafloor (e.g., using interferometry data, or using geometry based on the length of the shadow cast and the range from, and altitude of, the imaging sonar). If a seabed is composed of soft mud, objects can sink into the seafloor and become partially buried, thereby decreasing the observable object height. In contrast, on a seabed of hard-packed sand, objects are likely to remain proud on the seafloor, so the measured heights will be correspondingly larger. The result is that the height-feature measurement *for the same given object* can be very different in these two environments. If the environmental characteristics causing fundamental mismatches between the data sets used for training and testing are not recognized and addressed, classification performance can suffer.

In this work, we create a new classification framework that adroitly compensates for data mismatch by first quantifying the environmental conditions under which each data point is collected. This auxiliary information is then incorporated into a learning process that constructs multiple classifiers. The key is that the relative importance of each object (i.e., data point) during the learning phase for a given classifier is controlled via a modulating factor computed by comparing the object's environment feature with an analogous environment feature assigned to each classifier.

Substantial research has explored various versions of the transfer learning problem [1-4], which seeks to improve classification performance when the underlying distributions generating training data and (future) testing data differ. However, to our knowledge, no one has addressed the specific scenario considered here: purely supervised classification in which no test data – neither features nor labels – are available during the training phase, but auxiliary information in the form of a meta-feature associated with each training data point *is* available.

The remainder of this paper is organized as follows. The proposed classification algorithm that exploits auxiliary environmental information is described in Sec. 2. Experimental results are shown in Sec. 3, before concluding remarks are made in Sec. 4.

2. PROPOSED CLASSIFICATION ALGORITHM

The proposed classification algorithm exploiting auxiliary environmental information is outlined in detail here. To avoid interrupting the flow of the derivation, a more thorough discussion explaining the rationale surrounding various aspects is withheld until Sec. 2.7. For the sake of clarity, we first present the algorithm assuming the environment is represented by a single scalar meta-feature; later in Sec. 2.6 we present the extension to the general case of a vector of meta-features.

2.1 Preliminaries

Let $\boldsymbol{x}_i \in \mathbb{R}^d$ denote a (column) vector of d features representing the ith object of a training set of N such objects. Let $z_i \in \mathbb{R}$ denote a scalar meta-feature that quantifies auxiliary information about the conditions under which the ith object was collected. We refer to this meta-feature as the *environment* feature. Let $y_i \in \{+1, -1\}$ denote the class label (e.g., mine or clutter) that corresponds to the ith object, $\boldsymbol{x}_i$. Collect the N sets of object features, class labels, and environment features as $\boldsymbol{X} = \{\boldsymbol{x}_i\}_{i=1}^N$, $Y = \{y_i\}_{i=1}^N$, and $Z = \{z_i\}_{i=1}^N$, respectively.

The objective is to perform binary classification using the training data $\{\boldsymbol{X}, Y, Z\}$ where the presence of the auxiliary environment information, Z, distinguishes the task from standard supervised classification tasks. It should be noted that the proposed algorithm is a purely supervised approach, assuming no knowledge of, or access to, the testing data on which classification is to be performed subsequently.

2.2 Establishing Data Importance

In the proposed algorithm, rather than learning a single classifier, a set of $C>2$ classifiers are learned. The jth such classifier will be associated with an assigned environment feature value, z_j^*. The specification of C and the construction of the set $Z^* = \{z_j^*\}_{j=1}^C$ will be addressed below shortly.

A weight modulating the relative importance of the ith object during the learning of the jth classifier is calculated using the Boltzmann distribution:

$$\omega(z_i, z_j^*) = \frac{\exp\{-d(z_i, z_j^*)/\beta\}}{\sum_{k=1}^C \exp\{-d(z_i, z_k^*)/\beta\}} \tag{1}$$

where $\beta>0$ is a fixed scaling parameter and $d(z_i, z_j^*) = |z_i - z_j^*|$ is the distance between the ith object's environment feature and the environment feature associated with the jth classifier. It is here that the key auxiliary environment information is exploited. It should also be noted that the denominator in (1) is a normalizing constant ensuring $0 < \omega(z_i, z_j^*) < 1$.

2.3 Parameter Selection

The above formulation relies on three as-yet-unspecified quantities: C, the number of classifiers to be learned; Z^*, the set containing the environment feature associated with each classifier; and the scaling parameter β. These quantities are determined in the following manner. The first and last elements of Z^* are set to be the smallest and largest values of the training set's environment features, respectively: $z_1^* = \min_i z_i$ and $z_C^* = \max_i z_i$. The remaining C-*2* elements, z_j^*, are then assigned values that divide $[z_1^*, z_C^*]$ into equal partitions. By selecting the extrema of Z for inclusion in Z^*, the learned classifiers will span the greatest range of potential testing data environment values (which are unknown *a priori*) for which training data exists.

Next, $\beta \in \mathbb{R}^+$ and $C \in \mathbb{Z}^+$ are determined jointly by performing a brute-force (yet very easy and fast) search to find the (β, C) pair that maximizes the entropy of the importance weights $\omega(z_i, z_j^*)$ calculated using all N training data points. (Recall that $\omega(z_i, z_j^*)$ depends on both β and C.) That is, for a given (β, C) pair, the entropy

$$H(\omega|\beta, C) = -\sum_{\omega_k \in \Omega_\Delta} p(\omega_k) \log_2 p(\omega_k) \tag{2}$$

is calculated, where $p(\omega_k)$ is the relative frequency with which the (continuous-valued) weights $\omega(z_i, z_j^*) \forall i, j$ are mapped to the kth element in the discrete alphabet of quantized weights Ω_Δ. The (β, C) pair that maximizes the entropy of the weights is then selected.

2.4 Learning of Classifiers

With C selected, all z_j^* are specified. With β determined, all $\omega(z_i, z_j^*)$ can be readily computed as well, via (1). Let $\Omega_{(j)} = \{\omega(z_i, z_j^*)\}_{i=1}^N$ be the set of weights associated with the training data, $\{\boldsymbol{X}, Y, Z\}$, for the jth classifier. The jth classifier is then learned using $\{\boldsymbol{X}, Y, \Omega_{(j)}\}$ – the information contained in Z having been fully transferred to $\Omega_{(j)}$ – by modulating the contribution of the ith object, $\boldsymbol{x}_i$, by $\omega(z_i, z_j^*)$ in the (base) classifier's objective function. This weighting effectively controls the trust placed in each data point for the given classifier.

Many standard classification algorithms can be employed here as the base classifier within this framework, but we do assume that the classifier used will produce probabilistic predictions. In the experiments presented here, we use a modified form of the relevance vector machine (RVM) [5] with no kernel, so the classifier parameters are weights on the features themselves rather than on basis functions. (This choice has the added benefit that the learned parameters can be analyzed in terms of feature selection.) The RVM is also convenient because it provides probabilistic predictions that can be easily combined. Moreover, employing the RVM requires only a minor modification to the original objective function and, for the learning phase, its gradient and Hessian with respect to the classifier parameters, $\boldsymbol{w}_{(j)}$. (This particular modification is straightforward and does not affect the theoretical properties of the RVM, but care must be taken to ensure the same if one chooses to use a different base classification method.) The modified RVM objective function to be maximized under the proposed framework for the jth classifier becomes

$$J_{(j)} = \sum_{i=1}^{N} \omega\left(z_i, z_j^*\right) \log \sigma\left(y_i \boldsymbol{w}_{(j)}^T \boldsymbol{x}_i\right) - \frac{1}{2} \boldsymbol{w}_{(j)}^T \boldsymbol{A}_{(j)} \boldsymbol{w}_{(j)} \quad (3)$$

where $\boldsymbol{A}_{(j)}$ is a diagonal matrix of hyperparameters associated with the sparsity-promoting prior, $\boldsymbol{w}_{(j)}$ is the vector of classifier parameters to be learned, and $\sigma(u) = (1 + \exp\{-u\})^{-1}$ is the sigmoid function. (To recover the original objective function, one must simply remove the $\omega\left(z_i, z_j^*\right)$ factor.) Standard classifier learning is then undertaken as one normally would; the culmination of this process for the *j*th classifier is the vector of learned classifier parameters, $\boldsymbol{w}_{(j)}$.

2.5 Prediction

Let $\boldsymbol{W} = \left\{\boldsymbol{w}_{(j)}\right\}_{j=1}^{C}$ collect all of the learned classifiers. Then given a new unlabeled test object, $\boldsymbol{x}_\ell$, with environment meta-feature z_ℓ, class prediction is made using a weighted average of the *C* classifiers' predictions; this weighting is again specified by $\omega\left(z_\ell, z_j^*\right)$, measuring the similarity of the test object's environment feature with each classifier's environment feature, computed using (1). Thus, the probability that test object $\boldsymbol{x}_\ell$ belongs to class $y_\ell = +1$ is given by

$$p(y_\ell = +1 | \boldsymbol{x}_\ell, z_\ell, \boldsymbol{W}) = \sum_{j=1}^{C} \omega\left(z_\ell, z_j^*\right) p\left(y_\ell = +1 \middle| \boldsymbol{x}_\ell, \boldsymbol{w}_{(j)}\right), \quad (4)$$

where $p\left(y_\ell = +1 \middle| \boldsymbol{x}_\ell, \boldsymbol{w}_{(j)}\right)$ is the prediction of the *j*th classifier. For the modified RVM used in this work, $p\left(y_\ell = +1 \middle| \boldsymbol{x}_\ell, \boldsymbol{w}_{(j)}\right) = \sigma\left(\boldsymbol{w}_{(j)}^T \boldsymbol{x}_\ell\right)$.

2.6 Extension: Multiple Environment Meta-features

The above algorithm can easily be extended to handle the case in which the environment is represented by multiple meta-features, rather than a single scalar meta-feature. Let $\boldsymbol{z}_i = [z_{i1} \quad z_{i2} \quad \cdots \quad z_{iF}]^T$ denote a vector of *F* meta-features that quantifies auxiliary information about the conditions under which the *i*th object was collected.

The weight modulating the relative importance of the *i*th object during the learning of the *j*th of *C* total classifiers is then modified to be calculated as

$$\omega\left(\boldsymbol{z}_i, \boldsymbol{z}_j^*\right) = \frac{\exp\{-\sum_{f=1}^{F} d(z_{if}, z_{jf}^*)/\beta_f\}}{\sum_{k=1}^{C} \exp\{-\sum_{f=1}^{F} d(z_{if}, z_{kf}^*)/\beta_f\}} \quad (5)$$

where $\beta_f > 0$ is a fixed scaling parameter and $d\left(z_{if}, z_{jf}^*\right) = \left|z_{if} - z_{jf}^*\right|$ is the distance between the *i*th object's *f*th environment feature and the *f*th environment feature associated with the *j*th classifier. The distance calculation is made feature-by-feature, and a unique scaling parameter is included for each meta-feature, to prevent the contribution of one feature from unfairly dominating.

In the case of a scalar meta-feature, *C* was the number of classifiers to be learned because there were *C* unique meta-feature values associated with the classifiers. When

there is a vector of meta-features, C_f will correspond to the number of unique classifier-associated values of the *f*th meta-feature. The assumption is that the vector of meta-feature values associated with a given classifier will be formed from the Cartesian product of the individual meta-feature value sets. This means the total number of classifiers to be learned will be $C = \prod_{f=1}^{F} C_f$. Despite the unfavorable scaling, if the meta-feature dimension F is low, jointly learning the unknowns C_f and β_f for all f by maximizing the entropy of the weights will still be feasible. Once all C_f, β_f, and $z_{.f}^{*}$ are obtained, classifier learning and prediction proceeds as in the scalar meta-feature case.

2.7 Discussion

The principal insight being leveraged in the proposed framework is that the values of features extracted to represent objects at a given site can be strongly influenced by (and correlated with) a meta-feature summarizing environmental properties of the area. This environmental dependence is exploited by learning multiple classifiers, each associated with a particular environment. The data that are used to learn each classifier are automatically weighted according to their relevance (i.e, similarity) to the environment under consideration. In this way, all available data are always used to learn each classifier, yet classifier diversity (across different environments) is still achievable via unique weightings.

To enhance the rigor of this weighting, we appealed to the idea of Boltzmann distributions and the concept of energy states of a system. In this analogy, the probability that the system is in a specified state is equivalent to the contribution of an object to the learning process of the specified classifier. Just as low-energy states of a system are more probable, the contribution of an object to a classifier will be stronger when the environments associated with the object and classifier have low dissimilarity.

It should be noted that the normalization in (1) ensures that the total (summed) weight associated with each data point is unity. That is, although a given data point may contribute a different amount to each classifier, each data point will, in aggregate, contribute the same amount to the overall training process. So, to paraphrase George Orwell, “All data are equal, but some data are more equal than others.”

To determine the values of the scaling parameter β and the number of classifiers *C*, the entropy of the weights, ω, was maximized. The rationale behind this decision is that when the entropy of the weights is maximized, the diversity of the different learned classifiers will tend to be large because the contributions (weights) associated with each data point will be highly varied. This classifier diversity is important because we want to encourage different classifiers to be learned in different environments (as much as the data can support such a result). If β is too small, each data point will have one weight near unity and all others near zero. In this case, effectively, each classifier would be learned using only a subset of the data (namely the data points whose environment is most similar to the classifier’s under consideration). If β is too large, each data point will have virtually equal contributions (weights) for each classifier. As a result, each classifier learned would be nearly identical, thereby eliminating any potential for improved performance.

Because the environmental meta-feature is a continuous variable, we quantize this space into *C* discrete values. The discretization is particularly important because if *C* is too small, the classifiers will not be tailored finely enough to the environment of interest. Similarly, if *C* is too large, the contribution of each data point to learning each classifier will be weakened, in turn decreasing the data set diversity among the classifiers, and the

resulting classifiers will be too similar. In the Boltzmann distribution analogy referenced earlier, *C*, the number of different environments possible, is the number of possible states of the system.

3. EXPERIMENTAL RESULTS

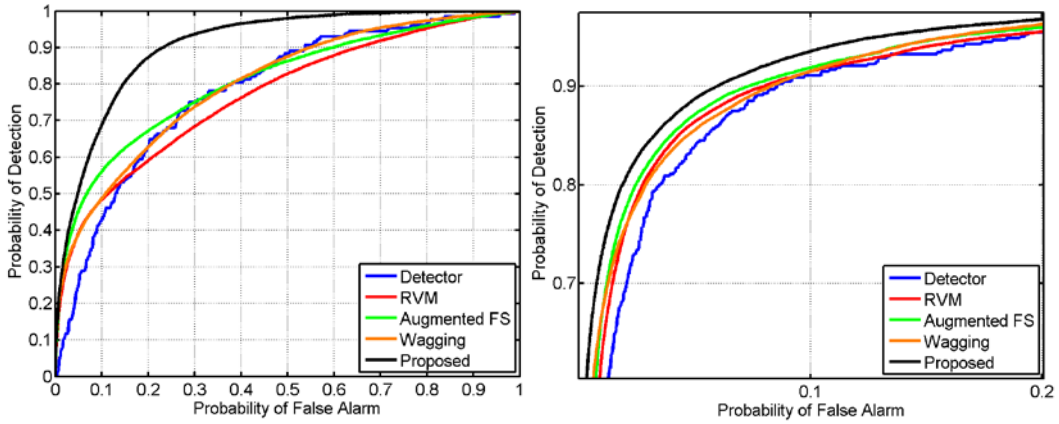
All of the data used in this study were collected by CMRE's MUSCLE autonomous underwater vehicle (AUV), which is equipped with a synthetic aperture sonar (SAS) system. The data, which spans eight different geographical sites, encompasses diverse environments in terms of seafloor characteristics, including flat hard-packed sand, soft mud, seabed characterized by sand ripples, and seabed covered in posidonia.

The detection algorithm described in [6] was applied to a huge database of SAS imagery containing over a thousand views of various man-made mine-like targets. A set of 27 object features was then extracted for each alarm (i.e., detection). In addition, two environment meta-features were also extracted for each alarm. The two environment features considered measure the anisotropy and complexity of the seabed. These features were introduced in [7], but they are computed here using the modifications described in [8]. The anisotropy feature can discern the presence of sand ripples, while the complexity feature can characterize the amount of background clutter in an area. All of this data was then used to perform binary classification with the goal of discriminating mine-like targets from clutter.

The experiments considered here exploit the extension of Sec. 2.6 permitting the use of multiple environment meta-features (here, anisotropy and complexity). (Similar results were obtained when only the seabed anisotropy feature was used.) We compare the classification performance of five approaches: (i) a standard classifier constructed on the 27-d feature data; (ii) a classifier constructed on the augmented 29-d feature space (FS) that includes the two environment meta-features as additional features; (iii) weight aggregation (or "wagging") [9], which learns *C* different classifiers after adding Gaussian noise to the contribution of each data point, and then averages the *C* predictions; (iv) the proposed method; and (v) a method that treats an alarm's score from the detection stage as its final classification prediction. A modified RVM with no kernel is employed as the base classifier for all methods in these experiments. For the case of wagging, the noise added to each weight is mean zero with a standard deviation of 2, as suggested in [9]; for direct comparisons, *C* is set to the value learned by the proposed method. The alternative methods are considered to demonstrate that performance gains achieved by the proposed approach are due to the comprehensive algorithm architecture as a whole, rather than one particular component (such as the use of multiple classifiers or the availability of additional environment features).

For each experiment, data from seven different sites are used as training data and then data from an eighth site are used as testing data. Each of the sites was treated as the test site once, for a total of eight experiments. The performance of the five methods is shown in terms of receiver operating characteristic (ROC) curves in Fig. 1 for two representative cases. (Space constraints prevent showing the results from all eight experiments here.) In Fig. 1(a), the test site (from the Colossus 2 sea trial) spans both benign flat seabed as well as seabed characterized by sand ripples; in Fig. 1(b), the test site (from the AMiCa sea trial) consists of only flat seabed. The training sites are characterized by multiple, different environments. As can be seen from the figures, the proposed method achieves superior performance. When the environment of the test data is diverse – and hence the

classification problem is more challenging – the gains in performance are more pronounced.



(a) (b)

Fig. 1: Classification performance for (a) a case in which the test data spanned both flat and rippled seabeds, and (b) a case for which test data was exclusively from flat seabed.

4. CONCLUSION

A new classification framework that incorporates auxiliary information about the environment in which the data has been collected was introduced. One of the particularly attractive aspects of the proposed algorithm is that there are no free parameters ("knobs") that must be tuned or tweaked. All of the necessary quantities are automatically learned from the data by the algorithm itself. Experimental results on an underwater mine classification task using SAS data demonstrated the superiority of the approach over alternative methods in which the environmental information is ignored or used differently.

REFERENCES

[1] **M. Sugiyama, M. Krauledat, and K. Muller**, "Covariate Shift Adaptation by Importance Weighted Cross Validation," *J. Machine Learning Research*, Vol. 8, pp. 985-1005, 2007.
[2] **L. Bruzzone and M. Marconcini**, "Domain Adaptation Problems: A DASVM Classification Technique and a Circular Validation Strategy," *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 32, No. 5, pp. 770-787, 2010.
[3] **R. Caruana**, "Multitask Learning," *Machine Learning*, Vol. 28, pp. 41–75, 1997.
[4] **S. Yuksel, J. Wilson, and P. Gader**, "Twenty Years of Mixture of Experts," *IEEE Trans. Neural Networks and Learning Systems*, Vol. 23, No. 8, pp. 1177-1193, 2012.
[5] **M. Tipping**, "Sparse Bayesian Learning and the Relevance Vector Machine," *J. Machine Learning Research*, Vol. 1, pp. 211-244, 2001.
[6] **D. Williams and J. Groen**, "A Fast Physics-Based, Environmentally Adaptive Underwater Object Detection Algorithm," in *Proc. OCEANS*, 2011.
[7] **O. Daniell, Y. Petillot, and S. Reed**, "Unsupervised Sea-Floor Classification for Automatic Target Recognition," in *Proc. International Conference on Underwater Remote Sensing*, 2012.
[8] **E. Fakiris, D. Williams, M. Couillard, and W. Fox**, "Sea-Floor Acoustic Anisotropy and Complexity Assessment Towards Prediction of ATR Performance," in *Proc. International Conference on Underwater Acoustics*, 2013.
[9] **E. Bauer and R. Kohavi**, "An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants," *Machine Learning*, Vol. 36, pp. 105-142, 1999.