Available online at www.sciencedirect.com
SCIENCE DIRECT®
Neural Networks 18 (2005) 541–548
Neural Networks
www.elsevier.com/locate/neunet

2005 Special Issue

A hierarchical classifier using new support vector machines for automatic target recognition

David Casasent, Yu-Chiang Wang*

Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA

Abstract

A binary hierarchical classifier is proposed for automatic target recognition. We also require rejection of non-object (non-target) inputs, which are not seen during training or validation, thus producing a very difficult problem. The SVRDM (support vector representation and discrimination machine) classifier is used at each node in the hierarchy, since it offers good generalization and rejection ability. Using this hierarchical SVRDM classifier with magnitude Fourier transform (|FT|) features, which provide shift-invariance, initial test results on infrared (IR) data are excellent.

Keywords: Automatic target recognition; Hierarchical classifier; Support vector machine

1. Introduction

Many pattern recognition applications deal with a multi-class (C class) classification problems, e.g. face recognition, speech recognition, handwritten character and symbol recognition, automatic target recognition (ATR), etc. Most of these classification problems are very complicated and the computation complexity is very high, since the number of classes C is usually very large.

We address an ATR problem in this paper. In general, an ATR system consists of four stages as shown in Fig. 1:a *detection* stage which operates on the original image and extracts regions containing possible targets (ROIs, regions of interest), a *preprocessing* stage which performs background removal and noise reduction within each ROI, a *feature extraction* stage which selects the features from each ROI for input to the *classification* stage, which decides on the type of input (non-object class or which object-class).

In this paper, we assume ROIs are provided. We emphasize the classifier, rejecting non-objects, and address reducing the number of features and providing shift and scale-invariant features, since the test object may not always be centered in each ROI.

To solve such a multi-class classification problem, some well-known techniques are k-nearest neighbor (kNN) or neural networks (NNs). However, these methods consider all the classes at once. A preferable method for multi-class classification is to combine several binary classifiers (Duda, Hart, & Stork, 2000), since it has been observed (Kumar, Ghosh, & Crawford, 1999) that designing a classifier which separates two classes is easier than designing one which distinguishes among all classes simultaneously. Thus better results occur (Kumar et al., 1999)

Two binary classification approaches, the *one-vs-rest* (Duda et al., 2000) and the *one-vs-one* (Hastie & Tibshirani, 1998) strategies, are typically used to solve the multi-class classification problem. The former decomposes the C-class classification problem into C binary classification sub-problems; each classifier separates one class from the remaining C-1 classes. The class label of the test input is decided by combining the C classifier outputs using winner-take-all etc methods (Anand, Mehrotra, Mohan, & Ranka, 1995). The latter forms a binary classifier for each class-pair and thus $C(C-1)/2 \approx C^2/2$ classifiers are required. For the test input, the decision is made by combining these $C^2/2$ classifier outputs using some relatively simple voting strategy, e.g. majority voting. However, in the one-vs-rest method, the similarity between the classes is not considered, so there is no guarantee that good discrimination exists

* Corresponding author.
E-mail addresses: casasent@ece.cmu.edu (D. Casasent), ycwang@cmu.edu (Y.-C. Wang).

0893-6080/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2005.06.033

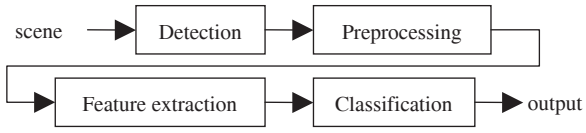

Fig. 1. A typical ATR system.

between one class and the remaining classes; in the one-vs-one method, a C-class classification problem is exhaustively decomposed into a set of $C^2/2$ classifiers, and the number of classifiers and computations are prohibitive (Kumar, Ghosh, & Crawford, 2002).

We considered a binary hierarchical classifier (Casasent & Wang, 2005)[1] using new SVRDM classifiers (Yuan & Casasent, 2003) at each node to achieve rejection. We choose the classes to be separated at each node using a top-down design. This is a modular learning problem, inspired by the divide-and-conquer strategy. Such modular learning has been found to learn concepts more effectively (better performance) and more efficiently (faster learning), since learning a large number of simple local decision boundaries is easier than learning complex global discriminations (Kumar & Ghosh, 1999). The hierarchical classifier has been shown to give better classification results than single complex classifiers, such as neural networks (NNs) and kNN classifiers (Kumar et al., 2002; Schwenker, 2000). The binary hierarchical classifier involves a tree structure of C-1 classifiers rather than a single layer of C classifiers (in the one-vs-rest approach) or $C^2/2$ classifiers (in the one-vs-one approach). At each node in the hierarchy, we divide the classes to be recognized into two smaller *macro-classes*; this procedure continues at subsequent levels and nodes. Only $\log_2 C$ classifiers are used in traversing a path from the top to a bottom decision node in the hierarchy (see Fig. 2). Thus, the number of required calculations is significantly reduced in this approach. Since each classifier is simpler, better classification rate P_C is also expected.

A new support vector representation and discrimination machine (SVRDM) classifier, recently developed by Yuan et al (Yuan & Casasent, 2003), is used at each node in our hierarchical classifier. The SVRDM is an extended version of the SVM (support vector machine) that gives good rejection of non-class inputs, which is important in realistic applications. We introduce the SVRM (support vector representation machine) and the SVRDM algorithms in Section 2. Both are used in our automated hierarchical classifier design (selection of the macro-classes per node). In Section 3, we detail our automated selection (hierarchical designs) of the macro-classes to be separated at each node. Section 4 describes the database we used. The image preprocessing and the feature spaces considered are detailed in Section 5. Experimental results are presented in Section 6.

[1] An abbreviated version of some portions of this article appeared in Casasent and Wang (2005), published under the IEEE copyright.

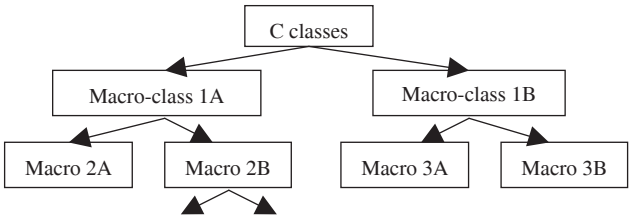

Fig. 2. Structure of a binary hierarchical classifier.

2. Support vector representation and discrimination machine (SVRDM)

The support vector machine (SVM) (Burges, 1998) is a powerful and promising binary classifier, which has been extensively used to solve many pattern recognition problems. The SVRDM, which provides good generalization (like the SVM) and can reject non-objects inputs well, is extended from the SVM. Section 2.1 first introduces the SVRM (one-class SVM), and the SVRDM algorithms are detailed in Section 2.2.

2.1. SVRM (Support vector representation machine)

In one of our hierarchical classifier design approaches to select the macro-classes to be separated at each node, we employ a SVRM (support vector representation machine), which is a one-class SVM that solves the data domain description problem (Tax & Duin, 1999) to recognize one object class in a higher dimensional transformed feature space with a Gaussian kernel and rejects unseen non-object inputs. To do this, we find the discriminant function $\boldsymbol{h}$ that satisfies $\boldsymbol{h}^T\Phi(\boldsymbol{x}) \geqq 1$ for all transformed versions $\Phi(\boldsymbol{x})$ of object class C_1 samples $\boldsymbol{x}$ and minimized $\|\boldsymbol{h}\|$. The evaluation function $f(\boldsymbol{x}) = \boldsymbol{h}^T\Phi(\boldsymbol{x})$ denotes the confidence that the input $\boldsymbol{x}$ is in the object class C_1. By minimizing $\|\boldsymbol{h}\|$, we minimize the decision region for the object class and thus achieve good rejection of non-object class C_0 inputs, which have never been seen. If $f(\boldsymbol{x}) < 1$, we reject the input as a non-object. Throughout this paper, we assume that *no non-object class samples to be rejected are available during training*. This is realistic for many applications, since we cannot expect data from all possible non-object classes. Others also make this assumption that no non-objects to be rejected are used in training or validation data (Chen, Zhou, & Huang, 2001).

Since the explicit form of the transform Φ is not necessary or available, only the kernel function is needed to compute the VIP (vector inner product) in the evaluation function. In the SVRM, we use a Gaussian kernel function (Yuan & Casasent, 2003) $exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/2\sigma^2)$. The Gaussian

kernel and $f(\mathbf{x})$ approaches zero as $\|\boldsymbol{x_i}-\boldsymbol{x_j}\|$ approaches infinity (or equivalently, if the input $\boldsymbol{x_i}$ is far from the support vector $\boldsymbol{x_j}$, $f(\mathbf{x})$ approaches zero (this is the case of non-objects to be rejected). Thus, if we train the SVM without a bias term, the evaluation function $f(\mathbf{x})$ approaches zero (a low value) when the input $\boldsymbol{x}$ is far from all support vectors of this one object class. When the bias term is present in the SVM, its value is determined by the standard SVM algorithm (Burges, 1998). If its value is close or larger than 1, this implies that the evaluation function value for a non-object class sample would be accepted with high confidence (recall that we regard one as a value of high confidence). This is not attractive. This is one reason why we choose to use the Gaussian kernel with bias $=0$ for our SVRM and SVRDM. Another reason for not using a bias term is that use of a Gaussian kernel without the bias term normalizes the data in the transformed feature space to lie on the unit sphere and this simplifies the task of estimation of the decision region's volume; i.e. this allows us to show that the SVRDM is best in terms of rejection ability (Yuan & Casasent, 2005). In this paper, the σ in the Gaussian kernel is selected automatically using the searching technique in (Yuan & Casasent, 2003).

2.2. SVRDM algorithm

The SVRDM (Yuan & Casasent, 2003) is an extension of the SVM that provides better rejection performance, which is vital for many applications. As we discussed in Section 2.1, we consider only the Gaussian kernel function to calculate the VIPs, for reasons noted above. To classify an input $\boldsymbol{x}$ into one of two classes, we thus compute the VIP (vector inner product) evaluation function output of the transformed test input $\Phi(\boldsymbol{x})$ and our nonlinear evaluation function $\boldsymbol{h}$ as $f(\boldsymbol{x})=\boldsymbol{h}^{\mathrm{T}}\Phi(\boldsymbol{x})$. The filter $\boldsymbol{h}$ is computed from the training set and solving the following quadratic programming problem

$$\begin{aligned}&\mathrm{Min}\|\boldsymbol{h}\|^2/2 \quad \boldsymbol{h}^T\Phi(x_i)\geq T,\ i=1,2,\ldots,N_1\\&\boldsymbol{h}^T\Phi(x_j)\leq -T, j=1,2,\ldots,N_2,\end{aligned}\tag{1}$$

where N_1 and N_2 are the number of training set samples in each class and the threshold T is ideally chosen to be 1 for an SVM, i.e. the VIP outputs are $\geqq 1$ for one class and ≤ -1 for the other class. When an SVM is used at each node in the hierarchical classifier, at each node, the classifier separates one macro-class from another. The VIP determines to which macro-class the input belongs.

Since the standard SVM cannot reject non-object inputs well, we extended the SVM to the new SVRDM with its better rejection performance in the multiple-object-class case (Yuan & Casasent, 2003). In our new hierarchical classifier, each node handles classification of two macro-classes. The two solution vectors $\boldsymbol{h_1}$ and $\boldsymbol{h_2}$ for macro-classes 1 and 2 in our SVRDM for a given node must satisfy

$$\begin{aligned}&\mathrm{Min}\|\boldsymbol{h}_1\|^2/2 \quad \boldsymbol{h}_1^T\Phi(x_{1i})\geq T,\ i=1,2,\ldots,N_1\\&\boldsymbol{h}_1^T\Phi(x_{2j})\leq p,\ j=1,2,\ldots,N_2, \quad \mathrm{Min}\|\boldsymbol{h}_2\|^2/2\\&\boldsymbol{h}_2^T\Phi(x_{2j})\geq T,\ j=1,2,\ldots,N_2\\&\boldsymbol{h}_2^T\Phi(x_{1i})\leq p,\ i=1,2,\ldots,N_1.\end{aligned}\tag{2}$$

In (2), there are N_1 and N_2 samples in each macro-class. Note that the second class output is specified to be $\leq p$ (and not $-\mathrm{T}$ or -1, as in the standard SVM). This improves rejection. Typically, we choose p in the range $[-1, 0.6]$. If $p=-1$, then (2) describes the standard SVM. In the presence of outliers (training class errors), slack variables ξ are used in both $\boldsymbol{h_1}$ and $\boldsymbol{h_2}$. Thus, the final $\boldsymbol{h_1}$ in (2) satisfies

$$\begin{aligned}&\mathrm{Min}\{\|\boldsymbol{h}_1\|^2/2+C(\sum\xi_{1i}+\sum\xi_{2j})\},\xi_{1i}\geq 0,\xi_{2j}\geq 0\\&\boldsymbol{h}_1^T\Phi(x_{1i})\geq T-\xi_{1i},\ i=1,2,\ldots,N_1\\&\boldsymbol{h}_1^T\Phi(x_{2j})\leq p+\xi_{2j},\ j=1,2,\ldots,N_2.\end{aligned}\tag{3}$$

In (3), C is the weight of the penalty term for the slack variables. The final version of $\boldsymbol{h_2}$ is similar.

At each node in our hierarchical classifier, we evaluate the associated two VIPs of the transformed input $\Phi(\boldsymbol{x})$ and the $\boldsymbol{h_1}$ and $\boldsymbol{h_2}$ (at that node). The VIP with the largest output ($\geqq$T) denotes the macro-class decision made at that node. If neither VIP gives an output $\geqq$T, the test input is rejected as a non-object (clutter in our ATR case).

2.3. SVRDM parameter selection

For each SVRDM in the hierarchy, we must choose values for several parameters. If we choose too small of a value for σ in the Gaussian kernel function, the bounded region for the class samples will be tight, more samples will be viewed as support vectors. This causes an over-fitted problem, and the generalization will be poor. We developed a new automated method to select σ (Yuan & Casasent, 2003).

The selection of the threshold T decides the acceptable range of the input $\boldsymbol{x}$ in (3). Lower T values result in worse rejection but better classification. From initial tests, we chose $T=0.9$, reducing T to 0.7 in increments of 0.1 until $\mathrm{P_C}$ on the validation set is good. Note that no test set images are present in the training or validation sets and that no non-object data be rejected are used in training or validation tests.

The value of p in (2) and (3) mainly affects P_C and the rejection rate. It is also affected by the distribution of non-object class data. If the non-object class samples (to be rejected) are distributed in the entire input space, a higher p

is preferred and it will reject samples not close to training set data (this also reduces P_C for true samples), as it provides a smaller decision region. We detailed this recently (Yuan & Casasent, 2005). For the TRIM-2 database we use here, the non-object class samples are different aircraft and background clutter images; thus, we varied p from 0.4 to 0.6 in increments of 0.1 and found that a larger $p=0.6$ is preferable. The last parameter to choose is the factor C in (3). We set $C=20$. This choice is not critical in our initial tests, since our scores are nearly perfect, i.e. there were no slack variables to be weighted in (3) during training our SVRDM classifiers.

3. Hierarchical classifier design

Our binary hierarchical classifier decomposes the C-class classification problem into C-1 sub-problems, each separating a pair of macro-classes. The structure of the binary hierarchical classifier is shown in Fig. 2. However, the macro-class partitioning in a hierarchical classifier cannot be decided arbitrarily or by intuition. Hierarchical clustering, which is an important unsupervised learning solution, can be used to solve this problem.

There are two types of hierarchical clustering approaches: *agglomerative* and *divisive* (Duda et al., 2000). The agglomerative clustering is a *bottom-up* design that searches all possible class pairs at the bottom level of the hierarchy and merges the two "closest" classes into a new group (macro-class) at the next level up the tree. This continues with two macro-classes produced at each node. The number of classes in each macro-class increases as we go from the bottom to the top of the hierarchy. The computational complexity is $\boldsymbol{O}(\mathrm{N}^2)$ at each node (N is the number of classes at a given node) and the overall complexity is $\boldsymbol{O}(\mathrm{C}^3)$, where C is the total number of classes. It is thus not practical to design a hierarchical classifier with a bottom-up design, when C is very large.

Divisive clustering divides a group of classes into two smaller disjoint groups at each node proceeding from the top to the bottom of the hierarchy, where the separation between these two groups of classes (macro-classes) is the "best" among all the macro-class pair combinations. Divisive clustering is also known as the top-down design. Similar to the divide-and-conquer strategy, this design approach makes a coarse discrimination between classes at the upper levels in the hierarchy and a finer classification later (at lower levels). In this paper, we propose two different divisive (top-down) methods to solve this unsupervised hierarchical clustering problem. We now detail them in Sections 3.1 and 3.2.

3.1. Balanced binary hierarchy design

In the top-down design, for a node with N classes, there are $2^{\mathrm{N}}/2-1=2^{\mathrm{N}-1}-1$ macro-class pair combinations to be evaluated. This is prohibitive, since the complexity is $\boldsymbol{O}(2^{\mathrm{N}-1})$, and it grows exponentially (not polynomially!) with N.

To alleviate this problem, we consider only a balanced binary hierarchical structure, in which the N classes at a node are divided into two macro-classes, each with the same number of classes (N/2). If N is odd, the difference between the class numbers in the two macro-classes is one. We assume N is even for simplicity, each macro-class then contains N/2 classes, and there are thus $0.5\times \boldsymbol{C}^N_{N/2}$ SVRDMs to be formed at each node, each for different sets of macro-classes. To select the macro-class pair at one node, an SVRDM is designed for each macro-class pair, using the training set data. The performance of these SVRDMs is then evaluated using the validation set data. To do this, we form the VIPs between the validation set inputs and the two macro-class SVRDM discriminant vectors in transformed space, as detailed in Section 2.2. The largest $p=0.6$ gives the best rejection. Thus, we fix $p=0.6$ and for each macro-class choice we vary T and select the threshold T and the macro-class pair whose SVRDM gives the highest classification rate $\mathrm{P_C}$ and the largest margin on the validation set (Casasent & Wang, 2005 for more details).

3.2. K-means SVRM clustering design

K-means clustering (Duda et al., 2000) is one of the most commonly used unsupervised clustering algorithm. It divides the data into k clusters such that the distance between each point $\boldsymbol{x_i}$ in each cluster j and its centroid $\boldsymbol{m_j}$ is minimized. The squared error criterion to be minimized is thus:

$$\boldsymbol{J}=\sum_{j}^{k}\sum_{\boldsymbol{x}\in C_j}||\boldsymbol{x}_i-\boldsymbol{m}_j||^2, \tag{4}$$

The centroid $\boldsymbol{m}_{\mathrm{j}}$ of class $\mathrm{C_j}$ is the best representative of the samples collected in that class. In the binary hierarchical classifier, there are only two clusters (macro-classes) to be determined at each node, and thus $k=2$.

In real-world problems, nonlinear transformed data $\Phi(\boldsymbol{x})$ is used, and thus k-means clustering must be applied to the transformed data $\Phi(\boldsymbol{x})$. The form of Φ is not necessary or available. We calculate an SVRM for each class $\mathrm{C_j}$ and use its solution vector $\boldsymbol{h}_j$ as the representative of the data in that class. The vector $\boldsymbol{h}_j$ is a linear combination of all support vectors for class C_j, and it can be considered as the best representative of each class in the transformed space (Section 2.1). In our new k-means SVRM clustering, we thus solve a k-means ($k=2$ macro-classes) clustering problem with only N vectors at each node in the transformed space ($\boldsymbol{h}_i$ for the N classes at that node). Each $\boldsymbol{x}_i$ in (4) is now replaced by $\boldsymbol{h}_i$. The pair of macro-classes that minimizes J in (4) denotes the macro-class pair for that node. As different classes are assigned to different clusters, the mean $\boldsymbol{m}_j$ of each macro-class cluster in (4) changes and is now more

easily calculated. We note that, unlike the balanced binary hierarchy, the numbers of classes in each macro-class are now not necessarily equal in k-means SVRM clustering. Also no validation set data is needed in this method. SVRDMs are then formed for the different macro-class pairs at each node.

4. Database description (TRIM-2)

The TRIM-2 database used contains 21 aspect views of each of twelve military objects (eight vehicles and four aircraft) in several different thermal states at a range of 1.5 km. Each image is 150×50 pixels. The eight vehicles were chosen as our targets; all four aircraft in all thermal states and backgrounds and thirty clutter regions in different backgrounds were chosen as the 366 false alarms to be rejected (Fig. 3).

The 21 aspect view images for each object-class were divided into: a *training set* of 9 aspect views (0°, +10°, +20°, +45°, +90°, −90°, −45°, −20°, and −10°) used for synthesis of the SVRDM classifiers, a *validation set* of five other aspect views (+5°, +25°, +75°, −30°, and −15°) used to select the SVRDM classifier parameters p and T and the macro-classes used at each node in our hierarchy, and a *test set* of seven aspect views (+15°,+30°,+60°, −75°, −60°, −25°, and −5°) used to obtain final P_C scores. The test set data differs from training set data by up to 15°. To test the rejection ability of our classifier (P_{FA} or percent false alarms not rejected), we used 21 aspect×4 aircraft×4 thermal states plus 30 clutter chips (ten each from the three backgrounds)=366 *non-object chips* at the same 1.5 km range. *No non-object data are used in training or validation*. The chip for each image includes many non-zero background pixels (See Fig. 1). The false alarm database used only the central 150×50 pixel region of each aircraft and of each clutter chip.

5. Preprocessing and feature extraction

We assume that regions of interest in a scene are provided that contain possible targets. Prior to training and testing, some preprocessing is required to reduce or remove background noise. A proper feature space is necessary with some desired properties, such as shift invariance. We detail our image preprocessing steps in Section 5.1, and we discuss several feature spaces we considered in Section 5.2.

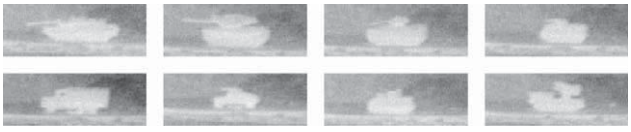

Fig. 3. Broadside view of eight classes to be recognized in the TRIM-2 database.

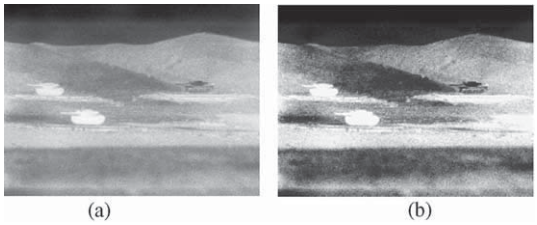

Fig. 4. Traditional contrast enhancement techniques: (a) linear normalization and (b) histogram equalization.

5.1. Image preprocessing

We initially included contrast enhancement in our preprocessing, using linear normalization (mapping the original grayscale image linearly to the full [0, 255] range) and histogram equalization. However, since the objects in the TRIM 2 database have negligible internal details, image enhancement emphasizes the outer boundary of the object too much; thus small errors in segmentation of the boundary of the object will be amplified. In addition, the Fourier transform of a sharp object boundary will ring in the frequency domain and thus cause problems with the magnitude Fourier transform (|FT|) feature space we use. As seen in Fig. 4, contrast enhancement also amplifies noise. Histogram equalization also maps different input gray levels to the same output gray level, which causes problems when the object and background pixel levels are similar and when the two are spatially close. Hence, we want to remove the background around the target, but not enhance the target's contrast. We do this by replacing the background with a constant uniform level. We achieve background removal by several morphological image processing techniques, as we now discuss.

The approach we use is to remove the object, leaving only the estimated background; then subtract this from the original and threshold, thus reducing structures in the background. We morphologically open the image by an *erosion* followed by a *dilation*. *Opening* removes any bright regions whose size is less than the structuring element. We use a structuring element slightly larger than the largest object aspect view (a 2D 30×80 pixel rectangle structuring element is used). The erosion removes bright regions smaller than the structuring element; for bright region larger than the structuring element, it reduces the size of these regions. The dilation operation restores these larger bright regions. These are gray scale morphological operations with a binary structuring element. Fig. 5 shows the results for the different steps. We apply the structuring element to the original target chips (Fig. 5a); the structuring element is slid pixel by pixel over the image (as in a correlation). The image pixel at the center of the structuring element is called the *origin*, i.e. the image pixel being processed. The pixels in the structuring element that are 1 define the *neighborhood* of the image being processed. The pixel at the origin is replaced by the minimum

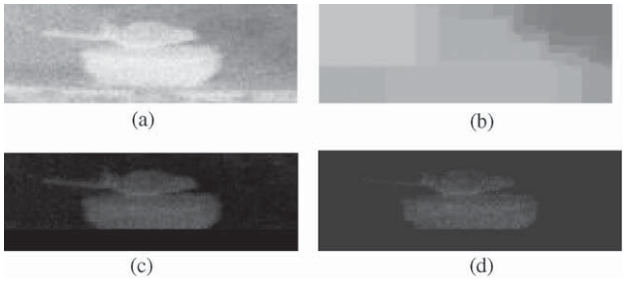

Fig. 5. (a) original target (b) estimated background (c) background reduction by (a)–(b), and (d) thresholded result of (c).

(for an erosion) or the maximum (for a dilation) input image pixel value in the neighborhood defined by the structuring element. Opening (closing) removes any bright (dark) region whose size (in 1D) is less than that of the structuring element. After opening, the result (Fig. 5b) is an estimate of the background around the object. After subtracting the estimated background from the original grayscale image, background and structure noise are greatly reduced (Fig. 5c). The remaining background pixel values are all less than those on the target. We thus apply a simple threshold to remove the structured background (Fig. 5d). We then assign the same uniform background level (50 was used) to the background regions. After preprocessing, all target pixel values are above 50 and less than 90. The contrast between the target and the background is now similar to that in the original grayscale image, and we do not overly emphasize object edges. We will compare P_C results with and without this background removal preprocessing (Section 6).

5.2. Feature spaces used

We considered three different types of feature spaces as input to our hierarchical classifiers. The first were iconic (pixel-based) features; specifically, the 150×50 pixel target images are lexicographically ordered into a 7500×1 element column feature vector. However, these features are not shift-invariant. The P_C performance will significantly decrease if the test inputs are not centered within a few pixels of the center of the object, as we discuss in Section 6.

To solve the problem of shift-invariance, we considered magnitude Fourier Transform (|FT|) features. This is also a feature space in which compression is possible. Analysis of the |FT| of the images showed negligible energy in higher spatial frequencies. Thus, only the lowest 50×50 spatial frequencies were retained to reduce the number of |FT| features from 7500 to 2500. Since the |FT| is symmetric, we use only the lower 50×25 |FT| components in the first and second quadrants, with no loss of information. For greater compression, we also considered use of only one quadrant of this |FT| feature space, the lowest 25×25 spatial frequencies in the first |FT| quadrant. This is now only 625 |FT| features that we refer to as *reduced |FT| features*.

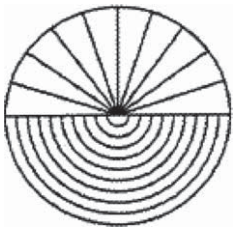

Fig. 6. Wedge-ring detector (WRD) |FT| concept.

Wedge-Ring Detector (WRD) |FT| features add distortion-invariance (scale and rotation invariance) to the shift-invariance provided by |FT| features (Lendaris & Stanley, 1970). Fig. 6 shows the wedge-ring detector concept. The |FT| plane is sampled with N wedge and M ring shaped detectors. Wedge |FT| features are scale-invariant, and ring |FT| features are rotation-invariant. All |FT| energy presents in each wedge and ring shaped detector region is summed. This feature selection method results in N+M WRD |FT| features vs. many more |FT| features. The use of the WRD |FT| features thus greatly reduces feature vector size.

|FT| features allow compression from $150\times50=7500$ features (using iconic or all |FT| features) to $25\times25=625$ features (using one reduced |FT| quadrant) with shift-invariance. Use of only $N=12$ wedge |FT| features is also considered, as it further compresses the number of features further to only 12. This represents a significant reduction in computation time; however, we note that computation time for the classification stage of the ATR problem is much less than in other stages and thus reducing it may not be of major concern.

6. Hierarchical SVRDM classifier results

Table 1 lists the performance of the hierarchical SVRDM classifiers using the balanced binary hierarchy design using different features. In the tests listed in Table 1, we do not consider image preprocessing, and we use the original images for training, validation, and testing, since the main purpose is to compare test results for different feature spaces.

Table 1
Test results of hierarchical SVRDM classifiers with balanced binary hierarchy design using different features

Feature	Iconic	50×50 pixel |FT|	Reduced 25×25 pixel |FT|	WRD |FT| (12 wedges)
P_C	98.22% (55/56)	100% (56/56)	94.65% (53/56)	96.4% (54/56)
P_{FA}	0%	0%	0%	0.27% (1/366)

Table 2
Performance comparison of balanced binary hierarchical SVRDM classifiers with and without preprocessed data

Feature used	Without preprocessing	With preprocessing
25×25 pixel |FT| features	P_C=53/56=94.65% (2 misclassification +1 miss)	P_C=55/56=98.21% (1 miss)

As shown in Table 1, the classification rate is P_C=98.2% using iconic features (55 of 56 test inputs are correct) and false alarm rejection P_{FA} is perfect; however, this feature space is not shift-invariant and the dimension of each input is very high (7500×1 pixels). In tests, we determined that if the chip images of the true inputs are shifted by three pixels, then many true class inputs are rejected as non-objects. This is why we consider |FT| features in further designs. The next entry in the top row of Table 1 shows the good performance obtained with the |FT| feature spaces with shift-invariance. Rejection P_{FA} is perfect. This is because of our SVRDM used. A standard SVM performs worse (Yuan & Casasent, 2003). Classification is perfect (all 56 test objects correctly identified) using fewer 2500 |FT| features vs 7500 iconic features. Reduced 25×25 pixel |FT| features and twelve-wedge |FT| features both gave comparable results (P_C=94.65 and 96.4%) in the bottom row in Table 1. Using twelve-wedge |FT| features, there was one non-object input which gave a VIP score above the threshold T and was thus not rejected. For other feature spaces, all non-object chips to be rejected gave VIP scores below the threshold T, and they were successfully rejected as false alarms (FAs). These test results are encouraging as the computational complexity of the classifier training and testing is greatly reduced.

Table 2 lists the performance (P_C and error types) for the hierarchical SVRDM classifier using only 25×25 |FT| features with and without our image preprocessing, using the balanced binary hierarchy design (Section 3.1). We note that a miss is a true object test input which gives a VIP score below the threshold and is thus rejected as a non-object input; a misclassification is a true object input, which gives a VIP score above the threshold and is assigned to an incorrect object class. Unlike a miss, which can often be recovered by adjusting the VIP threshold in the classifier, a misclassification error is not recoverable. Thus misses are preferable to misclassification errors. From Table 2, we note that two more (misclassification) errors occur if preprocessed data is not used. Thus, preprocessing noticeably improves results. The rejection for both cases is perfect, i.e. P_{FA}=0. Fig. 7a and b show the macro-class divisions chosen at the different nodes and levels in the hierarchy (using the balanced binary hierarchy design) and 25×25 |FT| features with and without using preprocessed images. Note that the two designs are different.

We next chose to use 25×50 |FT| features (the lower 25×25 |FT| components in the first and the second quadrants), since they contain more information. Using preprocessed data, Table 3 lists the performance for the hierarchical SVRDM classifiers designed by our two different automated macro-class selection methods. Both give excellent results (P_C=98.21% and P_{FA}=0).

Fig. 8 shows the hierarchical structure chosen using 25×50 |FT| features and the k-means SVRM clustering design (with preprocessing). Although the number of classes in each macro-class need not be equal using the k-means SVRM clustering design; however, this occurred in Fig. 8. As noted earlier, no validation set is needed in the k-means SVRM clustering design. Therefore, the k-means SVRM clustering design saves much training and evaluation time. We note that in the balanced binary hierarchy design, the margins for several different macro-class pairs, which gave the same highest P_C on the validation set, were very close (Casasent & Wang, 2005). Thus, the specific macro-class pair chosen is not critical. For this reason, and since k-means SVRM clustering results in a balanced binary hierarchy (like Fig. 7), we do not expect it to perform much better than the one using the balanced binary searching design. This occurs, as Table 3 showed.

In both hierarchical structures in Figs. 7b and 8, we note that the tanks (classes 1 and 2) are not in the same macro-class. Compared to the design of a hierarchy by intuition (which would place all tanks in the same macro-class), automated methods for selecting macro-classes (as we use)

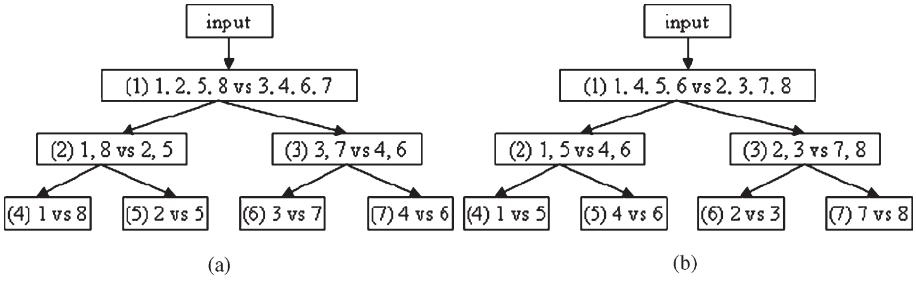

Fig. 7. Hierarchical classifier structures (balanced binary hierarchy design) using 25×25 pixel |FT| features (a) with preprocessed data (b) without using preprocessed data.

Table 3
Performance comparison for hierarchical classifiers with binary balanced hierarchy and k-means clustering designs

Features	Balanced binary hierarchy searching	K-means SVRM clustering
25×50 |FT| pixels	P_C=55/56=98.21% (1 miss)	P_C=55/56=98.21% (1 misclassification)

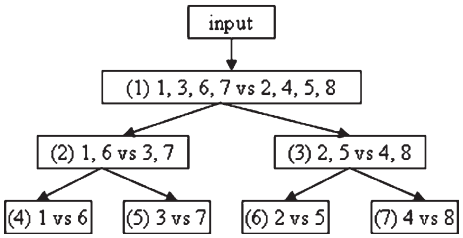

Fig. 8. Hierarchical classifier structure with preprocessed images using 25×50 pixel |FT| features with k-means SVRM clustering design.

are important. In the future, we will use the k-means SVRM clustering design, since it does not require us to exhaustively evaluate all balanced macro-class pairs when designing a hierarchy (and thus no validation set data is needed). This is attractive not only in terms of computation time in the training stage but also in cases in which only a limited dataset size is available.

We also calculated the average time to classify an input (after its feature vector has been formed). This includes traversing the SVRDMs in the tree structure. The average calculation time was 0.13 s for the 7500 (150×50) iconic features, 0.04 s for the 2500 (50×50) |FT| features, and 0.01 sec for the reduced 625 (25×25) |FT| features. These computations were made in Matlab on a P4 1.8 GHz PC with 256 MB RAM.

7. Conclusion

A new hierarchical SVRDM classifier with automated macro-class selection was proposed to solve the distortion-invariant ATR problem with excellent rejection ability. Our hierarchical SVRDM classifier achieved excellent performance (for both classification and rejection) on the TRIM-2 infra-red database.

Acknowledgements

The support of this work by ARO STTR Phase-2 contract W911NF-04-C-0099 is gratefully acknowledged. We also sincerely thank Clare Walters and Anita Burrell from NVESD for providing us the TRIM-2 database.

References

Anand, R., Mehrotra, K., Mohan, C. K., & Ranka, S. (1995). Efficient classification for multiclass problems using modular neural networks. *IEEE Transactions on Neural Networks*, *6*, 117–124.

Burges, C. J. C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, *2*(2), 121–167.

Casasent, D., & Wang, Y.-C. (2005). *Automatic target recognition using new support vector machine Proceedings of the international joint conference on neural networks, July 31–August 4, 2005, Montreal, Canada*.

Chen, Y. Q., Zhou, X. S., & Huang, T. S. (2001). One-Class SVM for learning in image retrieval. *International Conference on Image Processing* , 34–37.

Duda, R., Hart, P., & Stork, D. (2000). *Pattern classification*. New York, NY: Wiley-interscience.

Hastie, T., & Tibshirani, R. (1998). *Classification by pairwise coupling 1997 Conference on advances in neural information processing systems*, Vol. 10. The MIT Press (pp. 507–513).

Kumar, S., & Ghosh, J. (1999). *GAMLS: A generalized framework for associative modular learning systems Proceedings of SPIE conference on applications and science of computational intelligence II, SPIE Proceedings, Orlando, Fl*, Vol. 3722 pp. 24–35.

Kumar, S., Ghosh, J., & Crawford, M. M. (1999). A versatile framework for labeling imagery with a large number of classes. *International Joint Conference on Neural Networks*, *4*, 2829–2833.

Kumar, S., Ghosh, J., & Crawford, M. M. (2002). Hierarchical fusion of multiple classifiers for hyperspectral data analysis. In *Pattern analysis and applications*, spl issue on fusion of multiple classifiers, Vol. 5, No. 2. (pp. 210–220).

Lendaris, G. G., & Stanley, G. L. (1970). Diffraction-pattern sampling for automatic pattern recognition. *Proceedings of IEEE*, *58*(2), 198–221.

Schwenker, F. (2000). *Hierarchical support vector machines for multi-class pattern recognition 4th International conference on knowledge-based intelligent engineering systems & allied technologies,Proceedings of 4th KES, Brighton, UK*, Vol. 2 (pp. 561–565).

Tax, D., & Duin, R. (1999). *Data domain description using support vectors Proceedings of European symposium on artificial neural networks (ESANN'99), Bruges, Belgium* (pp. 251–256).

Yuan, C., & Casasent, D. (2003). *Support vector machines for class representation and discrimination 2003 International joint conference on neural networks, Portland* (pp. 1611–1616).

Yuan, C., & Casasent, D. (2005). Face recognition and imposter rejection using a new SVRDM modified support vector machine. *Proceedings of SPIE*, *5779*(29).