

# A Hierarchical Classifier Using New Support Vector Machine

Yu-Chiang Wang and David Casasent

Dept. ECE, Carnegie Mellon University, Pittsburgh, PA 15213

ycwang@cmu.edu, casasent@ece.cmu.edu

## Abstract

*A binary hierarchical classifier is proposed to solve the multi-class classification problem. We also require rejection of non-target inputs, which thus producing a very difficult problem. The SVRDM (support vector representation and discrimination machine) classifier is considered at each node in the hierarchy, since it offers good generalization and rejection ability. Using this hierarchical SVRDM classifier with magnitude Fourier transform features, initial recognition and rejection test results on simulated infra-red data are excellent.*

## 1. Introduction

Many pattern recognition applications, e.g. face recognition, automatic target recognition (ATR), etc., deal with a multi-class ( $C$  class) classification problem. Some well-known techniques, such as  $k$ -nearest neighbors (kNN) and neural networks (NNs), can be directly applied to these problems and they consider all classes at once. However, many classification problems are characterized not only by a large number of inputs, but also a large number of class labels. To solve this complicated and computationally expensive problem, a multi-class classifier is typically constructed by combining several binary classifiers [1], since it has been observed that designing a classifier which separates two classes (or two sets of classes) is easier than one which distinguishes among all classes simultaneously [2].

Two binary classification approaches, the one-vs-rest [1] and one-vs-one [3] strategies, are conventionally used to solve the multi-class classification problem, as we discuss in Sec. 2. Both methods can be considered as a two-level approach. A set of binary classifiers are formed in the first level, and some relatively simple mechanism is used in the second level to fuse the results from the first level. However, the one-vs-rest method does not consider the similarities between the classes, so there is no guarantee that good discrimination exists between one class and the remaining classes; the one-vs-one method requires  $O(C^2)$  pairwise classifiers, and it is thus very computationally expensive when  $C$  is large [4].

Inspired by the divide-and-conquer strategy, hierarchical classifiers [1] are attractive for the multi-class classification problem, where better results and fewer computations can be expected [4, 5]. If a binary classifier

is used at each node in the hierarchy, then this is called a binary hierarchical classifier; it involves a tree structure of  $C-1$  classifiers (see Sec. 3).

In this paper, a hierarchical classifier using the SVRDM classifier is proposed. The SVRDM algorithm is introduced in Sec. 4. The proposed method is illustrated on ATR problems. In Sec. 5, the design procedures of our hierarchical classifier are detailed. The proposed method is illustrated on an ATR problem. Recognition and rejection test results are shown in Sec. 6.

## 2. Background

### 2.1. One-vs-rest Method

The one-vs-rest method [1] is also known as the one-against-all or class-modular method. It decomposes the  $C$ -class problem into  $C$  binary sub-problems.  $C$  different classifiers (decision functions) are constructed; each of them separates a class from the remaining  $C-1$  classes. To accomplish this, each classifier needs to be trained on the whole training set. The class label of the test input is decided by combining the  $C$  classifier outputs using winner-take-all etc methods [6]. However, the one-vs-rest method results in imbalanced data problems, where the difference number of training set samples in the two classes is very different. Therefore, we cannot expect good classification results using this method.

### 2.2. One-vs-one Method

In the one-vs-one method (a.k.a. one-against-one or class-pairwise method) [3], a classifier is constructed for each class pair. In a  $C$ -class classification problem, there are thus  $C(C-1)/2 \approx C^2/2$  classifiers and each of them is trained on the data from only the two classes. The decision is made by combining these  $C^2/2$  classifier outputs using some voting strategy (e.g. majority vote or "Max Wins"). Since a  $C$ -class classification problem is exhaustively decomposed into a set of  $C^2/2$  classifiers, the number of classifiers and computations are prohibitive [4] when  $C$  is very large. In Sec. 3, a binary hierarchical classifier is thus proposed to solve the problems which exist in the one-vs-rest and one-vs-one methods.

## 3. Binary Hierarchical Classifier

A hierarchical classifier is inspired by the divide-and-conquer strategy, which makes a coarse discrimination between classes first (at upper levels in the hierarchy) and a finer classification later (at lower levels) [1, 4]. The root (top) node is a collection of all  $C$  classes. We first divide them into two disjoint subsets of classes (which is also called *macro-classes*). The macro-classes are further partitioned recursively in the subsequent levels until the partition reaches the leaf node at the bottom level in the hierarchy, which contains only one of the original  $C$  classes. Since a binary classifier is used at each node, a binary decision tree structure is constructed after all the macro-class pairs to be separated at each node are selected.

This method decomposes the  $C$ -class problem into  $C-1$  binary sub-problems. This is less than the  $C^2/2$  classifiers required in the one-vs-one method. Among these  $C-1$  classifiers, only  $\log_2 C$  classifiers are used when traversing a path from the top to a bottom node in the hierarchy. As a result, less computation time is expected in the use of the binary hierarchical classifier.

The macro-class partitioning in a hierarchical classifier, which is also known as *hierarchical clustering* [1], cannot be decided arbitrarily or by intuition. There are two types of hierarchical clustering in the hierarchy design (Sec. 3.1) to select the macro-classes used at each node (Sec. 3.2).

### 3.1. Hierarchical Clustering

Clustering is an important unsupervised learning technique; it finds the most suitable set in a collection of unlabeled data. The unlabeled data in the same set (cluster or macro-class) are “similar” or “close” to each other, and they are dissimilar to the data belonging to other clusters. For a binary hierarchical classifier, we divide the classes into two macro-classes at each node, i.e. there are two clusters to be selected at each node in the hierarchy.

The hierarchy can be formed by *agglomerative* or *divisive* clustering [1]. The former constructs the hierarchy in a *bottom-up* way by merging the two “closest” classes (macro-classes) into a new group at each level; the latter divides a group of classes into two smaller groups of classes (macro-classes), where the separation between these two groups is the “best” among all the macro-class pair combinations. The divisive clustering is the *top-down* design for the hierarchical classifier. Although comparable classification results have also been found using the bottom-up design, it is not practical to implement this design when  $C$  (the total number of classes) is large [4]. Therefore, we chose to use the top-down hierarchy design. Sec. 3.2 details two methods which are used to solve the top-down design problem.

### 3.2. Automated Macro-class Partitioning

In the binary hierarchical classifier with a top-down design, each node separates the classes into two macro-

classes. Typically, there are two methods to automatically decompose this multi-class problem into binary sub-problems:

#### (1) Exhaustive searching

Each possible macro-class pair is generated and is evaluated by the amount of the training errors. For a node with  $N$  classes, there will be  $2^N/2-1 = 2^{N-1}-1$  macro-class combinations. This can be prohibitive, even the design is off-line, since the complexity is  $O(2^{N-1})$  for selecting a macro-class pair for one node, and it grows exponentially (not polynomially!) with  $N$ . The forward-selection method (a.k.a. sequential searching) [7] is not preferable, since it is similar to an exhaustive search and cause other problems (e.g. the nesting problem).

#### (2) K-means clustering

K-means clustering is one of the most commonly used unsupervised clustering algorithms. The goal is to divide the data into  $k$  clusters such that the distance between each sample in each cluster to its centroid is minimized. The squared error function is the criterion function:

$$J = \sum_j^k \sum_{x \in C_j} \|x_i - m_j\|^2 \quad (1)$$

For a given cluster  $C_j$ , its mean vector  $m_j$  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, thus  $k = 2$ .

Inspired by one-class SVM (SVRM) [8, 10] (Sec. 4.2) and the SVRDM [8] (see Sec. 4.3), we propose two different automated macro-class partitioning algorithms, which are extended from the exhaustive searching and k-means clustering, respectively. Our automated macro-class partitioning algorithms are detailed in Sec. 5.

## 4. SVRDM

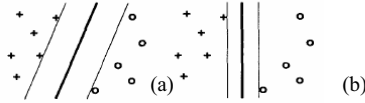
The SVRDM [8], which provides good generalization ability and can reject non-object inputs well, is extended from the SVM (support vector machine). We discuss the SVM in Sec. 4.1. The details of the SVRM (support vector representation machine) and the SVRDM are covered in Secs.4.2 and 4.3.

### 4.1. Support Vector Machines

The support vector machine (SVM) [9] has been extensively used to solve many classification problems. It was originally designed for binary classification problems. The main idea is to separate the two classes with a decision boundary which maximizes the margin between them. For the linearly separable case, a set of  $M$  samples  $(x_i, y_i)$  can be divided into two classes by a decision hyperplane, where  $x_i \in \mathbf{R}^N$  and its corresponding class label  $y \in \{+1, -1\}$ . Such a hyperplane is determined by a weighting vector  $h \in \mathbf{R}^N$  and a bias  $b \in \mathbf{R}$ , which satisfy:

$$\left. \begin{array}{l} \text{Min } \|h\|^2 / 2 \\ \text{subject to } y_i(h^T x_i + b) \geq 1, i=1,2,\dots,M \end{array} \right\} \quad (2)$$

This problem is solved by quadratic programming optimization techniques. The training set samples satisfying the equality in (2) are called *support vectors*. The distance between this separating hyperplane and the closest data point of the training samples is called the *margin*. A larger margin means a better separation between the two classes, and it also provides higher generalization ability of this classifier. See Fig. 1 for examples of separating hyperplanes with different margins for a 2D data case.



**Figure 1.** Linearly separating hyperplane for 2D separable cases (a) optimal separation with the largest margin (b) separation with a smaller margin.

For the linearly non-separable case, a set of slack variables  $\xi_i$  is introduced. The summation of the slack variables indicates the amount of training errors. Paying a penalty proportional to this amount, separation margin maximized for this non-linearly separable case satisfies:

$$\left. \begin{aligned} & \text{Min } \|\mathbf{h}\|^2 / 2 + C \left( \sum_{i=1}^l \xi_i \right)^k \\ & \text{subject to } y_i (\mathbf{h}^T \mathbf{x}_i + b) \geq 1 - \xi_i, \quad i=1,2,\dots,M \\ & \quad \text{and } \xi_i \geq 0 \end{aligned} \right\} \quad (3)$$

where  $k$  is usually assumed to be 1, and the parameter  $C$  is the penalty which defines the cost of constraint violation.

Most realistic classification problems will not be solved by a linear separating plane. In the SVM, we thus extend the separating plane to a higher dimensional space, and we solve a linear separation problem in that space. The input  $\mathbf{x}$  is first mapped from the input space  $\mathbf{R}$  to a higher dimensional space  $\mathbf{F}$  by a transform  $\Phi$ . Since we only need to compute the inner product,  $\Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}_j)$ , the explicit form of  $\Phi$  and the computation of  $\Phi(\mathbf{x})$  are not necessary. A kernel function  $K(\mathbf{x}_i, \mathbf{x}_j)$  is thus considered in this nonlinear decision case, and the Gaussian kernel ( $K(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x}-\mathbf{y}\|^2/2\sigma^2)$ ) is used in this paper.

For multi-class classification problems, several extensions to the SVM classifiers have been considered (e.g. the one-vs-rest and the one-vs-one method). The SVM has also been extended to the data domain description problem [10] (a.k.a. 1-class classification, see Sec. 4.2), which is important in one of our automated macro-class partitioning methods (see Sec. 5.2).

## 4.2. One-class SVM and the SVRM

In the one class classification problem, there is an object class  $C_1$  and a non-object class  $C_0$ . We are only interested in detecting the object class  $C_1$ , and the data from the non-object class  $C_0$  need to be rejected. Therefore, the task is to find an evaluation function  $f(\mathbf{x}) =$

$\mathbf{h}^T \Phi(\mathbf{x}) + b$ , which gives the confidence of the input  $\mathbf{x}$  being in the object class  $C_1$ . We assume that *no non-object class samples are available during training*. This is realistic for many applications, since we cannot expect data from all possible non-object classes. The one-class SVM [10] was proposed to solve this problem. The objective function to be optimized is the same as (2), except only one class samples ( $y = +1$ ) are used in training.

The SVRM (support vector representation machine) [8, 10] is an one-class SVM; it recognizes the object class inputs in a higher dimensional transformed space and rejects non-object inputs. The Gaussian kernel is considered, since the transformed data are automatically normalized (they lie on the unit sphere centered at the origin in the transformed space, and no bias term  $b$  in  $f(\mathbf{x})$  is needed). Therefore, the evaluation function of the SVRM is  $f(\mathbf{x}) = \mathbf{h}^T \Phi(\mathbf{x})$ , which is the inner product between a vector  $\mathbf{h}$  and the transformed input  $\Phi(\mathbf{x})$ . The  $\mathbf{h}$  satisfies  $f(\mathbf{x}) \geq 1$  for all  $\Phi(\mathbf{x})$  of the object class  $C_1$ . By minimizing  $\|\mathbf{h}\|$ , we minimize the decision region for the object class and thus achieve good rejection of non-object class  $C_0$  inputs. The use of the Gaussian kernel also simplifies calculation of the minimization of the volume of the decision region of the object class [12]. In this paper, the  $\sigma$  in the Gaussian kernel is selected automatically using the searching technique in [8].

## 4.3. SVRDM Algorithm

The SVRDM [8] is a binary classifier which is an extension of the SVRM to the multi-class problem. A pair of discriminant vectors  $\mathbf{h}_1$  and  $\mathbf{h}_2$  is formed, and we calculate the inner product between the input and each discriminant vector. Similar to the SVM, the discriminant vector  $\mathbf{h}_1$  satisfies the following equations and constraints:

$$\left. \begin{aligned} & \text{Min } \left\{ \|\mathbf{w}_1\|^2 / 2 + C (\sum \xi_{1i} + \sum \xi_{2j}) \right\} \\ & \mathbf{h}_1^T \Phi(\mathbf{x}_{1i}) \geq 1 - \xi_{1i}, \quad i=1,2,\dots,N_1 \\ & \mathbf{h}_1^T \Phi(\mathbf{x}_{2j}) \leq p + \xi_{2j}, \quad j=1,2,\dots,N_2 \\ & \quad \xi_{1i} \geq 0, \xi_{2j} \geq 0, \end{aligned} \right\} \quad (4)$$

and  $\mathbf{h}_2$  is similar. In (4),  $N_1$  and  $N_2$  are the number of training samples in classes (macro-classes) 1 and 2, and slack variables  $\xi_1$  and  $\xi_2$  are the classification errors of the training samples in the two classes. The parameter  $p$  in (4) controls the evaluation function for the other class and thus affects the size of the decision volume. If  $p = -1$ , the SVRDM is the standard SVM. As  $p$  increases, the decision region for both classes decreases. When training an SVRDM, one vector (e.g.  $\mathbf{h}_1$ ) gives an output  $\geq 1$  for class (macro-class) 1 inputs (e.g. a macro-class at the  $i$ th node in the hierarchy) and for the samples in class 2 it gives an output  $\leq p$ . The other vector  $\mathbf{h}_2$  gives outputs  $\geq 1$  for class 2 training set inputs and outputs  $\leq p$  for class 1. For the test input, the vector  $\mathbf{h}$  ( $\mathbf{h}_1$  or  $\mathbf{h}_2$ ) with the largest

output  $\geq$  threshold  $T$  determines the class of the input. Test inputs with inner products  $< T$  for both  $\mathbf{h}_1$  and  $\mathbf{h}_2$  are rejected as non-objects (false alarms). The SVRDM thus provides good discrimination between the two classes. It also provides more suitable decision regions than the SVM does to reject non-object class inputs.

## 5. Hierarchical Classifier Design

We propose two types of automated macro-class partitioning methods for the top-down hierarchical classifier design. We now detail them.

### 5.1. Balanced Binary Hierarchy Design

To alleviate the computational complexity in the exhaustive search, the balanced binary hierarchy [11] is used to select the macro-class pairs at each node. In a balanced binary hierarchy, a set of  $N$  classes are divided into two smaller groups of classes (macro-classes) at each node in the hierarchy, 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 that  $N$  is even for simplicity, each macro-class contains  $N/2$  classes and there are thus  $0.5 \times C_{N/2}^N$  SVRDM classifiers formed at each node. To select the macro-class pair at one node, an SVRDM is formed for each macro-class pair, using the training set samples. The performance of these SVRDMs is evaluated using the validation set data. We form the inner products between the validation set inputs and the two SVRDM discriminant vectors, as detailed in Sec. 4.3. For a given fixed  $p$  and threshold  $T$ , the macro-class pair with the highest classification rate  $P_C$  and the largest margin denotes the macro-class choice at that node [11].

### 5.2. K-means SVRM Clustering Design

The second method which we propose is k-means SVRM clustering. Traditionally in k-means clustering, the data and their corresponding cluster centers are analyzed in the original space. However, if the data distributions are complex, the centroid of each cluster cannot be determined appropriately.

Inspired by the SVRM, we transform the data to a high dimensional feature space and find the discriminant vector  $\mathbf{h}_i$  for each class. The vector  $\mathbf{h}_i$  is a linear combination of all the support vectors from the class  $i$ , and it can be considered as a best representative of each class in the transformed space (Sec. 4.2). In k-means SVRM clustering, we solve a k-means ( $k = 2$ ) clustering problem with only  $N$  data points (one vector  $\mathbf{h}_i$  for each of the  $N$  classes at that node) at each node in the transformed space. Each  $x_i$  in (1) is now substituted by  $\mathbf{h}_i$ . The two clusters (macro-classes) are determined at each node, when (1) is optimized for each of them. We note

that, unlike the balanced binary hierarchy, the numbers of classes in each macro-class are not necessary equal in this case.

Once the macro-class pairs are selected for each node in the hierarchy, the design procedure of our binary hierarchical classifier is complete. We form an SVRDM classifier for each macro-class pair using training set samples, and we use them as the binary classifiers at each node in the hierarchy to classify the test inputs.

## 6. Experimental Results

The proposed hierarchical SVRDM classifier is applied to a simulated infra-red database. This database contains 12 military objects (8 vehicles and 4 aircraft). Each is present at 21 aspect views (over  $\pm 90^\circ$ ). We chose to recognize 8 vehicles and used the 4 aircraft and background clutter as non-objects to be rejected. Fig.2 shows the target chips (50 x 150 pixels) of the 8 vehicles to be classified. The 21 aspect view images for each object were divided in to: a *training set* of 9 aspect views ( $0^\circ, +10^\circ, +20^\circ, +45^\circ, +90^\circ, -90^\circ, -45^\circ, -20^\circ,$  and  $-10^\circ$ ) used for synthesis of the SVRDM classifiers. A *validation set* of 5 other aspect views ( $+5^\circ, +25^\circ, +75^\circ, -30^\circ,$  and  $-15^\circ$ ) was used to evaluate the SVRDMs and select the macro-classes to be separated at each node in our hierarchy. A *test set* of 7 aspect views ( $+15^\circ, +30^\circ, +60^\circ, -75^\circ, -60^\circ, -25^\circ,$  and  $-5^\circ$ ) was used to obtain final  $P_C$  scores. The training set of aspect views was chosen to cover the  $\pm 90^\circ$  and thus to represent the data.

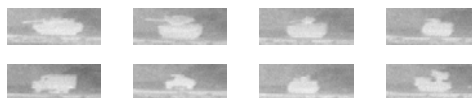


Figure 2. The eight classes to be recognized

Preprocessing and segmentation were performed prior to classification. We first morphologically opened the original image (Fig. 3a) to remove the target in the grayscale image. Opening is an erosion followed by a dilation. The 2D structuring element (SE) used is slightly larger than the largest target in the database. After opening, the result (Fig. 3b) is an estimate of the background. We thus subtracted the estimated background from the original image to greatly reduce the background noise (Fig. 3c). We thresholded the output and remove structured background (Fig. 3d). We set the background to a constant value so as not to overly emphasize object edges.

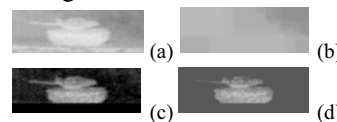


Figure 3. (a) original target, (b) estimated background, (c) background reduction by (a)-(b), and (d) thresholded output from (c)

We chose to use magnitude Fourier transform (|FT|) features for training and testing, since they provide shift-invariance and thus the target does not need to be centered in each image. Since there is negligible energy in the higher spatial frequency components, the lower 25 x 25 |FT| components in the first quadrant were used initially. Using the balanced binary hierarchy design, Table 1 lists the performance ( $P_C$  and error types) for the hierarchical SVRDM classifier with and without using preprocessed data. Without preprocessing, more errors were found than the one using preprocessed data. The rejection ability for both cases is perfect, i.e.  $P_{FA} = 0$ . Fig. 4a shows the class divisions chosen at the different nodes and levels in the hierarchy (by the balanced binary hierarchy design) using 25 x 25 |FT| features and preprocessed images.

**Table 1.** Performance comparison of balanced binary hierarchical SVRDM classifiers with/without preprocessing

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

We also chose to use 25 x 50 |FT| features (the lower 25 x 25 |FT| components in the first and the second quadrants), since they contain more information. Using the preprocessed data, Table 2 lists the performance for the hierarchical SVRDM classifiers designed by the two different automated macro-class partitioning methods (using preprocessed images). Both results were excellent ( $P_C = 98.21\%$  and  $P_{FA} = 0$ ).

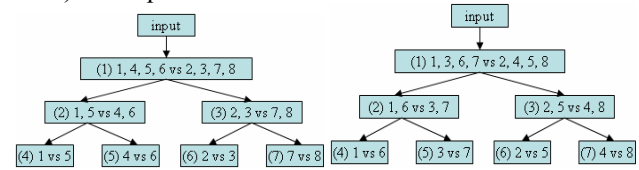
**Table 2.** Performance comparison for hierarchical classifiers with balanced binary hierarchy and k-means clustering designs

Features	Balanced binary hierarchy searching	K-means SVRDM clustering
25 x 50  FT  pixels	$P_C = 55 / 56 = 98.21\%$ (1 miss)	$P_C = 55 / 56 = 98.21\%$ (1 misclassification)

Fig. 4b shows the hierarchical structure using 25 x 50 |FT| features with the k-means SVRDM clustering design. Although the numbers of classes in each macro-class are not necessary to be equal using the k-means SVRDM clustering design, they turn out to be the same in our test result (Fig. 4b is a balanced binary hierarchical structure). This macro-class selection method does not evaluate all possible macro-class pairs; thus, no validation set is needed to be used. Therefore, k-means SVRDM clustering design saves much training and evaluation time. We note that in the balanced binary hierarchy method, the margins of several possible macro-class pairs, which gave the same highest  $P_C$  on the validation set, were actually very close [11], and the specific macro-class pair chosen is thus not critical. Since k-means clustering results in a balanced binary hierarchy (like Fig. 4b), we do not expect

that it to perform much better than the one using the balanced binary search design.

In both hierarchical structures in Fig. 4, we note that the tanks (classes 1 & 2) are not in the same macro-class. Compared to the design of a hierarchy by intuition, automated methods for selecting macro-classes (as we have) are important.



**Figure 4.** Hierarchical classifier structures using (a) 25x25 pixel |FT| features with balanced binary hierarchy search design and (b) 25x50 pixel |FT| features with k-means clustering design

## 7. Conclusions

A novel hierarchical SVRDM classifier with automated macro-class selection is proposed to solve the multi-class problem. Our method shows remarkable classification and rejection test results on a simulated IR database.

## 8. Acknowledgment

The support of this work by ARO STTR Phase-2 contract W911NF-04-C-0099 is gratefully acknowledged.

## 9. References

- [1] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, 2<sup>nd</sup> Ed., Wiley-interscience, 2001.
- [2] S. Kumar, J. Ghosh, M. Crawford, "A versatile framework for labeling imagery with larger number of classes," *Proc. Int'l Joint Conference on Neural Networks (IJCNN)*, Washington, DC, 1999.
- [3] T. Hastie and R. Tibshirani, "Classification by pairwise coupling, Advances in neural information processing systems," The MIT Press, Vol. 10, 1998.
- [4] S. Kumar, J. Ghosh, and M. Crawford, "Hierarchical fusion of multiple classifiers for hyperspectral data analysis," *Pattern Recognition and Applications*, 5, 2002, pp. 210-220.
- [5] M. Perrone, "A soft-competitive splitting rule for adaptive tree-structured neural networks," *Proc. of IEEE IJCNN*, 1992, pp. 689-693.
- [6] R. Arand et al, "Efficient classification for multiclass problems using modular neural networks," *IEEE Trans on NNs*, Vol. 6, 1995, pp. 117-124.
- [7] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, Academic Press Inc. New York 2<sup>nd</sup> ed., 1992.
- [8] D. Casasent and Y. Chao, "Face recognition with pose and illumination variations using new SVRDM support-vector machine," *Opt. Engineering*, 43(8), 2004, pp.1804-1813.
- [9] C. Cortes and V. Vapnik, "Support-vector network," *Machine learning*, Vol. 20, 1995, pp. 273-297.
- [10] D. Tax and R. Duin, "Data domain description using support vectors," *Proc. of European Symposium on Artificial Neural Networks (ESANN'99)*, 1999, pp. 251-256.
- [11] D. Casasent and Y.-C. Wang, "New distortion-invariant SVRDM hierarchical classifier," *Proc. of SPIE*, Vol. 5608(45), Oct. 2004.
- [12] C. Yuan and D. Casasent, "Face recognition and verification with pose and illumination variations and imposter rejection," *Proc. of SPIE*, Vol. 5779(29), Marh 2005.