A. Majid, A. Khan and Anwar M. Mirza,, "Combination of Support Vector Machines Using Genetic Programming," in the International Journal of Hybrid Intelligent Systems (IJHIS), 2006 (to appear).

# Combination of Support Vector Machines Using Genetic Programming

**Abdul Majid, Asifullah Khan and Anwar M. Mirza**

Faculty of Computer Science & Engineering, GIK Institute, Ghulam Ishaq Khan (GIK) Institute of Engineering Science & Technology, Topi-23460, Swabi, PAKISTAN { majid, akhan and mirza}@giki.edu.pk

**Abstract:** This paper describes the combination of support vector machine (SVM) classifiers using Genetic Programming (GP) for gender classification problem. In our scheme, individual SVM classifiers are constructed through the learning of different SVM kernel functions. The predictions of SVM classifiers are then combined using GP to develop Optimal Composite Classifier (OCC). In this way, the combined decision space is more informative and discriminant. OCC has shown improved performance than that of optimized individual SVM classifiers using grid search. Another advantage of our GP combination scheme is that it automatically incorporates the issues of optimal kernel function and model selection to achieve high performance classification model. The classification performance is reported by using Receiver Operating Characteristics (ROC) Curve. Experiments are conducted under various feature sets to show that OCC is more informative and robust as compared to their individual SVM classifiers. Specifically, it attains high margin of improvement for small feature sets.

**Keywords:** Support Vector Machines, Optimal Composite Classifiers, Receiver Operating Characteristics Curves, Area Under the Convex Hull (AUCH), Genetic Programming.

## 1 Introduction

There is a considerable interest in obtaining useful information from a large volume of data. Scientists and engineers are turning to computers to find automatic classification methods to make sense from data. Intelligent classification models are being developed in new fields of Bioinformatics, Machine Learning, Data Mining and Knowledge Discovery [20]-[23], [34]. This is the reason why researchers are always in search of high performance classification models. Improvements in such models might improve the overall quality of the system [1]. The main objective of a classification model is to achieve good generalization performance on new test samples. For example, in a disease diagnosis system, practitioners are interested in high performance classification models.

Support vector machine is developed for binary classification problems. It is a margin-based classifier with good generalization capabilities. SVMs are frequently used in real world applications of pattern classification and regression [2]. In recent years, SVMs are being widely used in various image processing and classification tasks [3]-[4]. SVM based discriminant approach is preferred in high dimension image space. SVM classification models could be developed by using different kernel functions. We have chosen SVM models for combination due to their high discrimination power and low generalization error.

Optimization of SVM models is an active area of research. In the optimization of SVM models, two main issues are encountered; selection of kernel function and its associated parameters (model selection) for the problem at hand [5]-[11]. In kernel selection, SVM are tested with various kernels for higher classification, while keeping minimum training error [4]. In model selection, mostly grid search based iterative methods are applied in order to optimize the parameters within the range of $[10^{-15} - 10^{15}]$ [8]. However, such search methods are based on trial and error. They become computationally inefficient when the number of parameters is more than two [5]. Optimization of SVM models is performed by linear combination of SVM kernel functions [18]. However, this combination model uses class conditional probabilities and nearest neighbor techniques. So far, no intelligent combination method, based on SVM kernel functions, is developed. However, research in the development of such methods is in progress.

Currently, the combination of multiple classifiers has attained a considerable attention for higher classification, in which the prediction accuracy is improved by parameters tuning. A combination of classifiers is expected to be more accurate than a single classifier [13]. In such systems, individual classifiers and their diversity could give us suitable combination of classifiers [14]-[15] with the deficiency of one classifier can be replaced by the advantage of other. However, there are many challenges how to generate the component classifiers and how to combine the results provided by these component classifiers in a best possible way. These problems are normally addressed independently through *coverage optimization* and *decision optimization techniques* [16].

There exist well-known combination techniques such as Bagging, Boosting and Adaboost [38] that manipulate the input training data for training diverse types of component classifiers. In Bagging a small set of training data is randomly picked to generate a component classifier for an ensemble [40]. Their output predictions are then combined by giving equal weights. In Boosting, first weak classifiers are created with accuracy greater than 0.5. Next, the distribution of the training data is changed. This distribution is based on the performance of previously trained component classifiers. The output of each component classifier is then weighted sum to give higher accuracy. Adaboost, a small variant of Boosting, allow adding weak classifiers until some desired low training error has been obtained. In Adaboost, each training example receives a weight that determines its chance to be selected in training set for a subsequent component classifier.

Our GP-based combination technique is different from these combination techniques. These techniques improve the classification performance by iteratively retraining the component classifiers with a subset of most informative training data. During training, each time the trained component classifier produces a single point on ROC curve (TPR and FPR coordinates). The value of decision threshold T is fixed and re-sampling of the training data is performed repeatedly. However, in this work, we want to evaluate the performance of composite classifiers against the whole threshold range of [0-1]. In our GP-based technique, effective combinations of fixed numeric classifiers are automatically created during GP evolution process. Their performance is then evaluated in each generation. There is no need to retrain them iteratively.

Another problem in the conventional combination methods is the complexity involved in finding an optimal search space. Usually, the possible solutions of combination functions are very large. To find out the optimal composite model by manually adjusting the parameters might be a tedious work. Moreover, there is a lack of general combination rules. Under such circumstances, GP-based combination technique may offer a good alternative approach. In such optimization problems, there are good chances for this technique to perform better. GP-based technique may be used successfully to address. GP has potential to develop target based complex numerical functions. Previously, GP has been used in combining different classification models like, kNN, Artificial Neural Networks, Decision Trees and Naïve Bayes [19]-[26]. In the current work, we address two main issues through the following contributions:

- We genetically combine individual SVM classifiers to construct an optimal decision space using the decision space of individual classifiers. Different kernel functions of SVMs can represent the complex feature space more accurately.
- Our GP-based intelligent method has eliminated the requirement of finding an optimal SVM model. This happens implicitly in the GP evolution mechanism. GP automatically incorporates the values of suitable constants in addition to variables terminals.

In our technique, first, individual SVM classifiers are tuned over [0-1] range of decision thresholds T. GP is then used to evolve appropriate combination functions using AUCH of ROC curve as a fitness function. This work is an extension of our previous works [25], [26] and [39]. Here, we are extending the applicability of GP method to combine SVM classifiers.

The remaining paper is organized as follows: In Section 2, we briefly describe SVMs. In Section 3, the proposed methodology and architecture of a classification system is explained. Implementation details are given in Section 4. Results and discussion are presented in Section 5. Finally, conclusions are given in Section 6.

## 2 SVM Classifiers

SVM performs pattern classification between two classes by finding a decision surface that has maximum distance to the closest points in the training set [2]. These points are called support vectors. SVM addresses the classification problem as a quadratic optimization problem by placing an upper bound on the margin between classes. The training principle of SVM is to find an optimal linear hyperplane such that the classification error for new test samples is minimized. For a linearly separable data, hyperplane is determined by maximizing the distance between the support vectors. Consider *n* training pairs $(x_i, y_i)$, where $x_i \in R^N$ and $y_i \in \{1, -1\}$ , the decision surface is defined as:

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i x_i^T . x + b \tag{1}$$

where the coefficient $\alpha_i > 0$ is the Langrange multiplier in an optimization problem. A vector $x_i$ that corresponds to $\alpha_i > 0$ is called a support vector. $f(x)$ is independent of the dimension of the feature space and the sign of $f(x)$ gives the membership class of $x$ . In case of linear SVM, the kernel function is simply the dot product of two points in the input space.

In order to find an optimal hyperplane for non-separable patterns, the solution of the following optimization problem is sought [2].

$$\Phi(w, \xi) = \frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i \tag{2}$$

subject to the condition $y_i \left( w^T \Phi(x_i) + b \right) \geq 1 - \xi_i, \quad \xi_i \geq 0.$

where $C > 0$ is the penalty parameter of the error term $\sum_{i=1}^{N} \xi_i$ and $\Phi(x)$ is nonlinear mapping. The weight vector $w$ minimizes the cost function term $w^T w$ . For nonlinear data, we have to map the data from the low dimension *N* to higher dimension *M* through $\Phi(x)$ such that $\Phi : R^N \rightarrow F^M, M >> N$ . Each point $\Phi(x)$ in the new space is subject to Mercer's theorem. Different kernel functions are defined as: $K(x_i, x_j) = \Phi(x_i).\Phi(x_j)$ . We can construct the nonlinear decision surface $f(x)$ in terms of $\alpha_i > 0$ and kernel function $K(x_i, x_j)$ as:

$$f(x) = \sum_{i=1}^{N_S} \alpha_i y_i K(x_i, x) + b = \sum_{i=1}^{N_S} \alpha_i y_i \Phi(x_i) \cdot \Phi(x) + b \tag{3}$$

where, $N_s$ is the number of support vectors.

In SVMs, there are two types of kernel functions, i.e. local (Gaussian) kernels and global (linear, polynomial, sigmoidal) kernels. The measurement of local kernels is based on a distance function while the performance of global kernels depends on the dot product of data samples. Linear, polynomials and radial basis functions are mathematically defined as:

$K(x_i, x_j) = x_i^T . x_j$ (Linear kernel with parameter $C$ )

$K(x_i, x_j) = \exp(-\gamma \| x_i - x_j \|^2)$ (RBF with Gaussian kernel parameters $\gamma, C$ )

$K(x_i, x_j) = [\gamma < x_i, x_j > + r\ ]^d$ (Polynomials kernel with parameters $\gamma, r, d$ and $C$ )

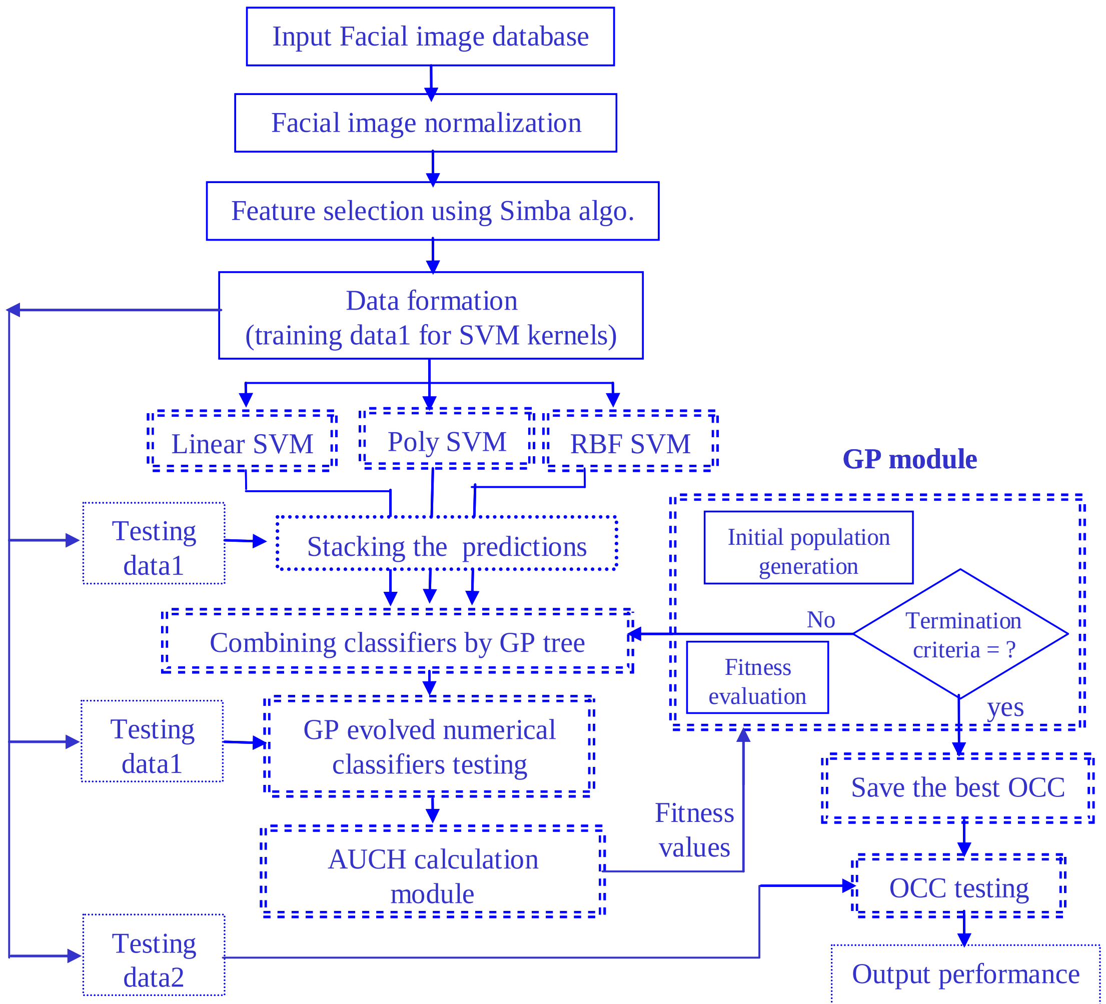
Linear, RBF and polynomial kernels have one, two, and four adjustable parameters respectively. All these kernels share one common cost parameter *C*, which represents the constraint violation of the data point occurring on the wrong side of the SVM boundary. The parameter $\gamma$ in the RBF shows the width of Gaussian functions. In order to obtain optimal values of these parameters, there is no general rule about the selection of grid range and step size. In the present work to select the optimal parameters of kernel functions, grid search method described in [8] is used.

## 3 Proposed Methodology

We have taken gender classification problem as a test case. ROC curve is a general performance measure of classifiers [30]. During GP evolution AUCH of ROC curve is taken as a fitness function [20]-[26]. In the proposed scheme, a combination of classifiers is carried out using the concept of stacking the predictions of classifiers to form a new feature space [12], [22]-[26]. Suppose, there are *m* kernel functions $K_1, \ldots, K_m$ . The dataset *S* is partitioned into three non-overlapping but equal training and testing sets, i.e. $\{X_1 \cap X_2 \cap X_3\} = \varnothing$ , *w*here $X_1$ = training data1, $X_2$= testing data1 and $X_3$= testing data2. Each data-label example $s_i$ is represented by $s_i = (x_i, y_i)$ . A set of individual classifiers $C_1, \ldots, C_m$ are constructed by training kernel functions on training dataset $X_1$ , i.e. $C_j = K_j(x_{1i}),\ x_{1i} \in X_1,\ i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, m$ . A new feature set (meta-data) $(\hat{y}_i^1, \ldots, \hat{y}_i^m)$ is constructed by stacking the predictions of SVM classifiers under the second testing dataset $X_2$ as: $\hat{y}_i^j = C_j(x_{2i})$, *w*here $x_{2i} \in X_2,\ \forall j = 1, 2, \ldots, m$ . GP meta-learning process is based on the new training data space of $(\hat{y}_i^1, \ldots, \hat{y}_i^m),\ \forall i = 1, \ldots, n$ .
GP optimally combine the predictions of individual classifiers $(\hat{y}_i^1, \ldots, \hat{y}_i^m)$ to obtain OCC. The predictions of individual classifiers are used as unary functions in the GP tree. These unary functions are mixed during GP crossover and mutation operations. In order to develop OCC various GP runs are carried out. The main modules of our scheme are shown with double dashed boxes in Figure 1. A brief description of each module is given as follows:

### 3.1 Normalization of face databases

Various databases are combined to form a generalized and unbiased database for gender classification. Different face images are collected from the standard databases of YALE [41], CVL [42], ORL [43] and Stanford medical student [44]. In this way, a more general face image database is formed. Images are taken under various conditions of illumination with different head orientations. In the normalization stage, CSU Face Identification Evaluation System [33] is used to convert all images in the uniform state. Images within the databases are of different sizes taken under various conditions. CSU system includes standardized image pre-processing software to study the unbiased performance of classifiers. In order to convert each image of size 103 by 150 into a normalized form, this system performs various image preprocessing tasks. First, human faces are aligned according to the chosen eye coordinates. Then, an elliptical mask is formed and images are cropped such that the face from forehead to chin and cheek to cheek is visible. At last, the histogram of the unmasked part of the image is equalized. A sample of a few images, their normalized form, as well as the block diagram for processing images is shown in Figure 2(a-b).



Input Facial image database
Facial image normalization
Feature selection using Simba algo.
Data formation (training data1 for SVM kernels)
Linear SVM
Poly SVM
RBF SVM
GP module
Testing data1
Stacking the predictions
Initial population generation
Combining classifiers by GP tree
No
Termination criteria = ?
Fitness evaluation
yes
Testing data1
GP evolved numerical classifiers testing
Save the best OCC
Fitness values
AUCH calculation module
OCC testing
Testing data2
Output performance

Figure 1: Architecture of the proposed classification system

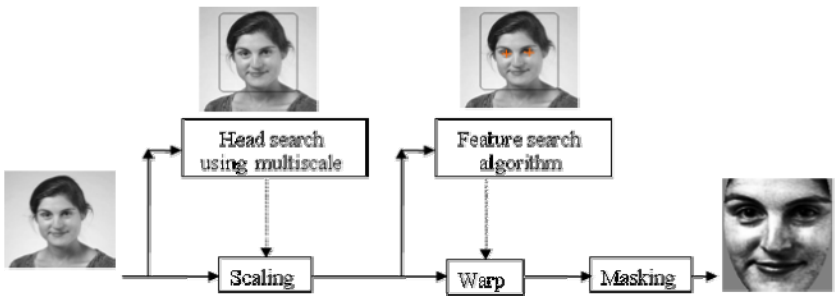


Figure 2(a): Face alignment system with different stages

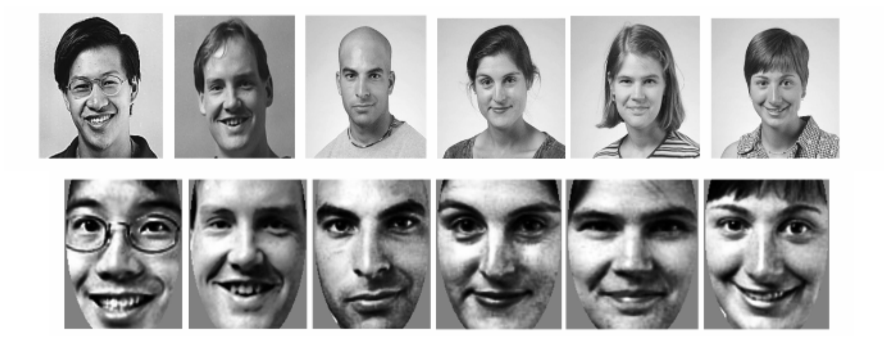


Figure 2(b): Sample of a few input faces and processed faces

## 3.2 Features selection

Feature selection is the task of reducing dimensions by selecting a small set of features. Feature selection reduces the feature space, which in turn speeds up the classification process. There exist many image pixels based feature selection techniques. However, we are using Iterative Search Margin Based Algorithm (*SIMBA*). Details about this algorithm are available in [32]. This algorithm along with 1-NN classifier combines large margin so that optimal image pixel may be selected. A stochastic gradient ascent is used over the evaluation function $e(W)$ in order to maximize this function. For a training set S having sample vectors $x$ , each of dimension $N$ and a weight vector $W$ , the feature evaluation function $e(W)$ is defined as:

$$e(W) = \sum_{x \in S} \mu^{W}_{S \setminus x}(x) \tag{4}$$

where µ represent the hypothesis margin of an instance $x$ . The gradient of $e(W)$ on set S is computed as:

$$\Delta\left(e(W)\right)_i = \frac{\partial e(W)}{\partial w_i} = \frac{1}{2}\left( \frac{\left(x_i - nearmiss(x)_i\right)^2}{\left\|x - nearmiss(x)\right\|_w} - \frac{\left(x_i - nearhit(x)_i\right)^2}{\left\|x - nearhit(x)\right\|_w} \right) w_i \tag{5}$$

where $i = 1, 2, \ldots, N$. $Nearhit(x)$ indicate the nearest point to $x$ having the same label as that of $x$ , while $nearmiss(x)$ indicate the nearest point to $x$ with different label. A weight vector $W$ is obtained as: $W = W + \Delta$ .

In order to analyze the performance of classifiers for various feature sets, different values of thresholds are used for this weight vector $W$. After picking the desired image pixels, three separate datasets; training data1, testing data1 and testing data2 are formed. In the original database *S*, we have selected 300 male and 300 female images. Thus, each part of this dataset contains 100 male and 100 female images. This data is then used in various training and testing stages as shown in Figure 1.

### 3.3 Evaluation of Classifiers: ROC curve

In the present work, the performance of classifiers are measured in terms of ROC curves. This measure is also used in GP fitness function. ROC curve is a general performance measure of a classifier. It summarizes the performance of classifier under different operating conditions for a particular problem. When there is no prior knowledge of the true ratio of misclassification costs for a classification system, ROC curve is a reasonable performance measure [28]. This ROC curve and its area rank the classifier decision values [27].

TPR (true positive rate) represents the number of correct positive cases divided by the total number of positive cases. FPR (false positive rate), on the other hand, is the number of negative cases predicted as positive cases divided by the total number of negative cases. FPR (X-axis) and TPR (Y-axis) values represent the specificity and sensitivity of the classification system respectively. In order to plot ROC curve, the predicted values of a classifier are scaled in the range of [0-1]. The values of TPR and FPR of the entire test samples are obtained by applying threshold T in the range of [0-1]. If the output of classifier is greater than the threshold T, then input sample is allocated to one class (male), else to the other class (female). ROC curve is then plotted between TPR and FPR and then AUCH of ROC curve is determined. AUCH of a classifier's ROC is the M*aximum Realizable* ROC [29]. It gives a scalar value representing the overall performance of a classifier. This performance measure has been used in various classification models [20]-[26]. A classifier is an optimal one, if AUCH of its ROC is near to one.

### 3.4 GP-based combination of SVM classifiers

In the combination of classifiers, first suitable component classifiers are selected. They may be trained on the same or different data. In the homogenous combination, only one type of component classifier is used to develop a composite classifier. In this way, OCC is a function of only one classifier $f(C_L), f(C_P) \, or \, f(C_R)$ . On the other hand, in heterogeneous combination, the composite classifier is a function of two or more component classifiers i.e. $f(C_L, C_P, C_R)$ [26]. These component classifiers may be trained on the same or different data. We have studied homogenous and heterogeneous combination of linear classifiers [31], statistical classifiers [26], kNN classifiers [19], and SVM classifiers [39].

In the present work, we are using heterogeneous combination of SVM classifiers trained using different SVM kernel functions. OCC trained using this scheme may usually have a better chance of delivering high performance. The procedure adopted is as such: first individual SVM classifiers are trained using training data1. Their predictions are extracted using the testing data1. Such predictions of individual classifiers are stacked in three arrays (L, P and R), which would be used as unary functions within the GP tree. In order to plot ROC curve, the decision threshold T is used as a variable terminal in GP tree as shown in figure 3. During GP evolution, combined decision space of individual SVM classifiers is developed. Finally, the performance of OCC and individual SVM classifiers is analyzed in terms of AUCH of ROC curve using testing data2. The main steps of our GP module are as follows:

### 3.4.1 GP module

GP is a type of Evolutionary Algorithms that are based on the mechanism of natural selection. In context of classification, GP-based technique comes under the category of stochastic methods, in which randomness plays a crucial role in searching and learning [1]. We represent a classifier as a candidate solution with a tree like-data structure. Initially, a population of individuals is created randomly as a possible solution space. Next, score of each classifier is obtained for a certain classification task. In this way, the fitness of each classifier is calculated. The survival of fittest is carried out by retaining the best classifiers. The rest are deleted and replaced by their offspring. These retained classifiers and their offspring are used for the next generation. During genetic evolution, each new generation has a slightly higher average score. In this way, the solution space is refined and converges to the optimal/near optimal solution [34]. We have used *GPLAB software* [35] to develop OCC. All the necessary settings are given in Table 1. In order to represent possible solutions in the form of a complex numerical function, suitable functions, terminals, and fitness criteria are defined. Different functions of our GP module are as follows:
*GP function set:* GP Function set is a collection of various mathematical functions available in the GP module. In GP run, we have used simple functions, including four binary floating arithmetic operators (+, -, *, and protected division), *LOG*, *EXP*, *SIN* and *COS*. Our functions set also consist of application specific logical statements e.g. greater than (gt) and less than (lt).
*GP terminals:* In the development of OCC, we have selected suitable random variables and constants. Threshold T is taken as a variable terminal. Random numbers in the range [0-1] are generated from uniform distribution. They are used as constant terminals in GP tree.
*Population initialization method:* We have generated an initial popoulation by using ramped half and half method. In this methood, first, equal number of individuals are initialized for each tree depth, while the number of depths are considered from two to the initial tree depth value. Now for each tree depth level, half of the individuals are

initialized using *Grow method* and the other half individuals using *Full method* [35].
Fitness function: The performance of individuals in GP population is assessed by using AUCH of ROC curve as a fitness function. A fitness function grades each individual. It provides feedback to the GP module representing the fitness of individuals. Figure 1 shows how this function is used to score each individual. Higher fitness score of an individual indicates higher performance in terms of AUCH of ROC curve.

Table 1: GP Parameters Setting

| ***Objective*** | ***To evolve a optimum combined classifier with maximum AUCH*** |
|---|---|
| Function Set: | +, -, *, protected division, gt,le, log, abs, sin and cos |
| Special Function: | SVM classifier prediction ( L, P, R ) are used as unary functions |
| Terminal Set: | Varaible threshhold T and random constants both are in the range of [0 - 1] |
| Fitness : | AUCH of ROC curve |
| Expected offspring: | rank85 |
| Selection: | Generational |
| Wrapper: | Positive if $\geq$ 0, else negative |
| Population & Generations: | 300 & 80 respectively |
| Initial Tree Depth Limit: | 6 |
| Initial population: | Ramped half and half |
| GP Operators prob: | Variable ratio of crossover & mutation is used |
| Sampling: | Tournament |
| Survival mechanism: | Keep best individuals |
| Real max. tree level: | 28 |

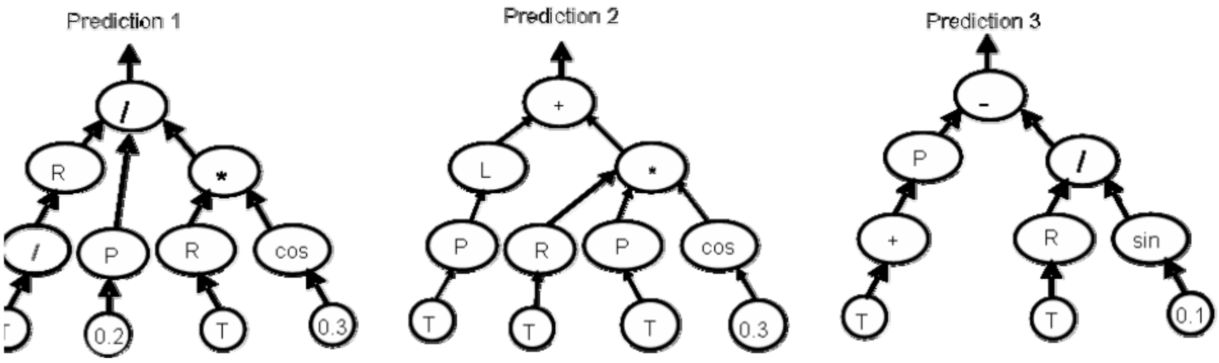


Fig. 3: Combination of SVM classifiers represented as unary functions in GP trees

GP operators used: We have used replication, mutation, and crossover operators for producing new generation. Replication is the copying of an individual into the next generation without any change. In mutation, a small part of an individual's genome is changed. This small random change often brings diversity in the solution space. Crossover creates an offspring by exchanging genetic material between two individual parents. It tries to mimic recombination and sexual reproduction. Crossover helps in converging onto an optimal/near optimal solution. In GP run, we have used variable ratio of crossover and mutation.

*Termination criterion:* The simulation is stopped if one of the two conditions is encountered first;

- The fitness score exceeds 0.999.
- Number of generations reaches the maximum limit.

### 3.4.2 Testing phase

At the end of a GP run, the best expression of a GP individual is obtained. Its performance is then evaluated using testing data2.

## 4 Implementation Details

The experimental results are obtained using Pentium IV machine (1.6 GHz, 256MB RAM). OSU-SVM toolbox [36] is used in MATLAB 6.1 environment. In the first study, the performance of OCC is analyzed with individual SVM classifiers by priori fixing the values of kernel parameters, i.e. cost parameter $C=1$, kernel parameter $\gamma=1$, coefficient $r=0$ and degree $d=3$. In the 2nd study, the optimal values of these parameters are adjusted by using grid search. Suitable grid range and step size is estimated for SVM kernels. PolySVM has four adjustable parameters; $d, r, \gamma$ and $C$ . However, to simplify problem, the values of degree and coefficient are fixed at $d=3$, $r=1$ . The optimum values of $\gamma$ and $C$ are then selected. In case of PolySVM, a grid range of $C=[2^{-4}, 2^{5}]$ with step size $\Delta C=0.2$ and $\gamma=[2^{-7}, 2^{2}]$ with step size $\Delta\gamma=0.2$ are used. In case of RbfSVM, the range of grid and step size of $C$ and $\gamma$ are selected as: $C=[2^{-15}, 2^{10}]$, $\Delta C=0.2$ and $\gamma=[2^{-10}, 2^{15}]$ , $\Delta\gamma=$ 0.2 . The optimal value of $C$ parameter for linear kernel has been obtained by adjusting the grid range of $C=[2^{-1}, 2^{5}]$ with $\Delta C=0.2$ .

Efforts have also been made to minimize the problem of over-fitting in the training of both individual SVM and composite classification models. Appropriate size of training and testing data is selected in the holdout method. Tuning parameters of GP evolution are also selected carefully.

## 5 Results and Discussion

The following experiments have been conducted to study the behavior of OCC. Comparative performance analysis of OCC is also carried out with that of the individual SVM classifiers.

### 5.1 Analysis of accuracy versus complexity

We investigate the behavior of the best GP individuals developed for 1000 features set. Figure 4(a) shows how the fitness of best individual increases in one GP run. Figures 4(b-c) show the complexity of the best individuals expressed in terms of tree depth and no. of nodes. It is observed that in the search of better predictions, size of the best GP individual increases after each generation. As a result, the best genome's total number of nodes increases and its average tree depth becomes very large. We can observe from Figure 4(d), the large increasing behavior of median, average, and maximum fitness of the best individual.

During crossover and mutation operations, more and more constructive blocks build up that minimize the destruction of useful building blocks [34]. As a result, the size of GP program grows after each generation. This might be due to *bloating phenomenon* of GP evolution [37]. Due to this phenomenon, many branches within the best GP program generally do not contribute in improving its performance. However, such branches increase the size of GP program.

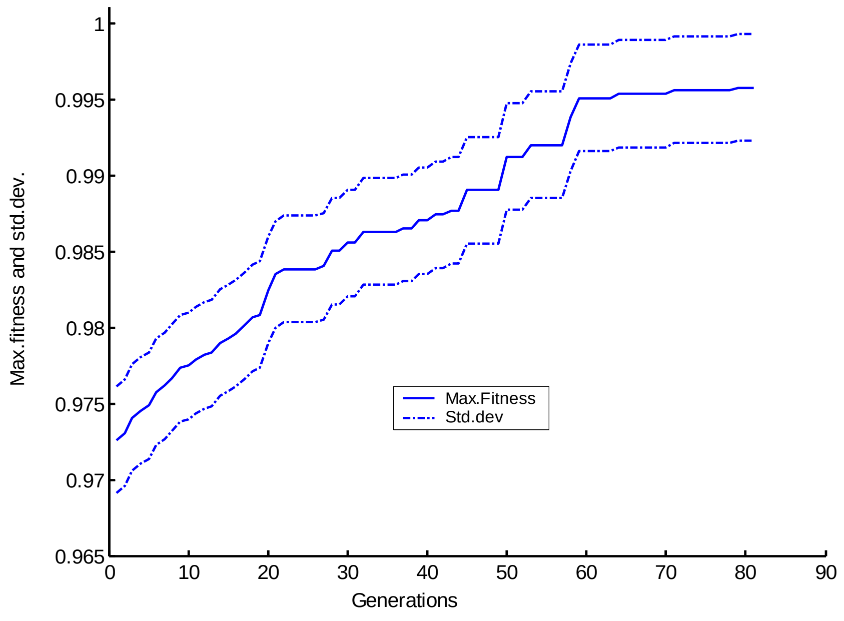


Figure 4(a): The maximum fitness of the best individuals in one GP run ($\sigma = 0.007$).

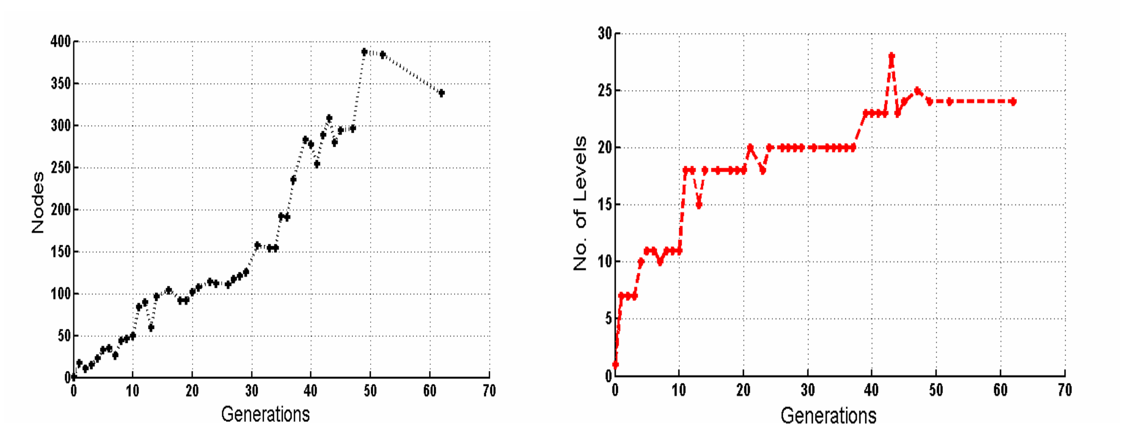


Figure 4(b)-(c): (Left) No. of nodes versus generations; (Right) Tree depth versus generations of the best individuals in one GP run.

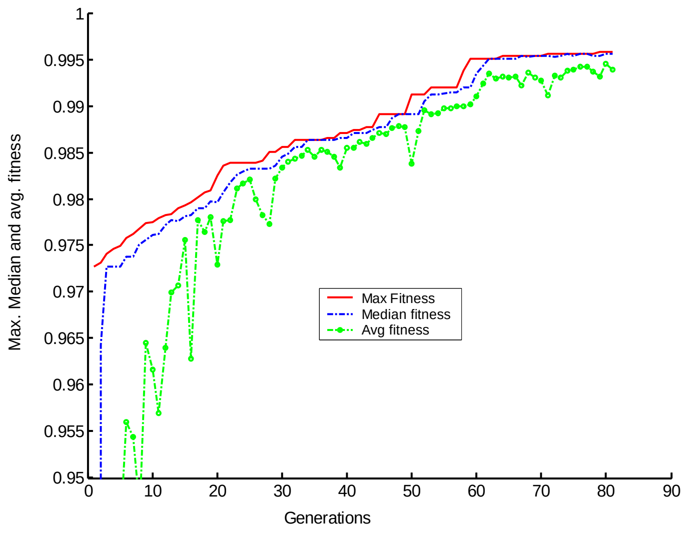


Figure 4(d): The behavior of maximum, median, and average fitness of the best individuals.

## 5.2 Polynomial SVM for different feature sets

In order to study the effect of increasing the size of feature sets, on the performance of individual SVM classifiers, with PolySVM is selected as a test case. Its performance in terms of ROC curves for different feature sets is shown in Figure 5. This Figure shows the improvement in ROC curves with the increase of feature sets, e.g. PolySVM-10 (trained on 10 features) has AUCH = 0.8831 and PolySVM-1000 has AUCH = 0.9575. AUCH of ROC curve gives the overall performance of the classifier in a summarized form and it is not necessary to have high TPR values for all FPR values. Practically, initial high values of TPR and low FPR are more crucial for significant overall performance. For example, PolySVM-100 has overall lower values of AUCH, even though, it has high values of TPR for FPR $\geq$ 0.4 than that of PolySVM-1000.

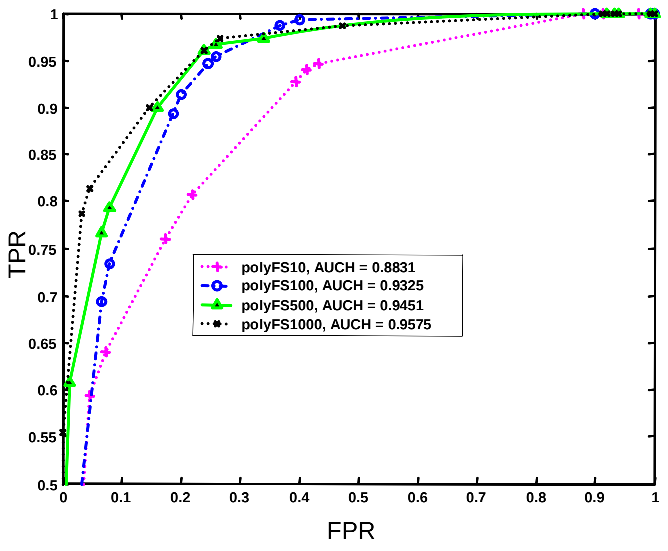


Figure 5: ROC curves of PolySVM. (For simplicity, in this figure and in the next figures, only those points that recline on the convex hull of the ROC curve are shown.)

## 5.3 Performance comparison of OCC with individual SVM classifiers

This experiment is carried out to analyze the performance of OCC and SVM classifiers. Figures 6(a-d) show AUCHs of ROC curves obtained for various feature sets. These figures demonstrate that with the provision of more information, TPR increases, while FPR decreases. Our OCC has outperformed individual SVM classifiers. This improvement in ROC curves is due to low values of FPR and high values of TPR. These values help the points shifting towards upper left corner and thus providing better decision. Such kind of behavior is desirable in those applications where the cost of FPR is too important. For example, a weak patient cannot afford high FPR. Minor damage of healthy tissues may be a matter of life and death. On the other hand, if attempts are made to reduce FPR by simply adjusting decision threshold T, the risk of false negative cases might rise in a poor prediction model. Such kind of prediction models, specifically in medical applications, might cause high misclassification cost in various fatal diseases such as lungs, liver, and breast cancer [31].

Bar chart in Figure 7 shows the comparison of experimental results in a more summarized form for various feature sets. It is observed that linear SVM has the lowest AUCH values. Due to the nonlinearity in the feature space, Linear SVM is unable to learn this space effectively. However, with the increase of feature sets, it improves AUCH performance. To promote diversity, usually diverse types of component classifiers are used in the development of composite classifiers [14]. As a result, linear classifier is included to enhance diversity in the decision space.

As far as the performances of PolySVM and RbfSVM classifier are concerned, they have shown relatively equal performance. Both are capable of constructing a nonlinear decision boundary [2]. However, the performance of our OCC is superior to both these individual SVM classifiers. During GP evolution process, composite classifiers might have extracted useful information from the decision boundaries of constituent kernels. Another advantage gained is that composite classifiers have shown higher performance, specifically for small feature sets of sizes 5, 10, and 20. The general order of performance of classifiers is:

$$OCC_{AUCH} > (PolySVM_{AUCH} \cong RbfSVM_{AUCH}) > LinSVM_{AUCH}$$

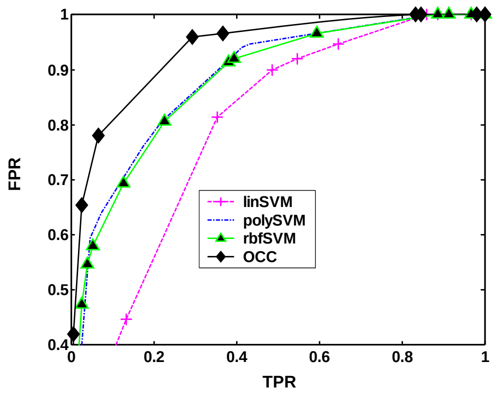


Figure 6(a): ROC for 10 features

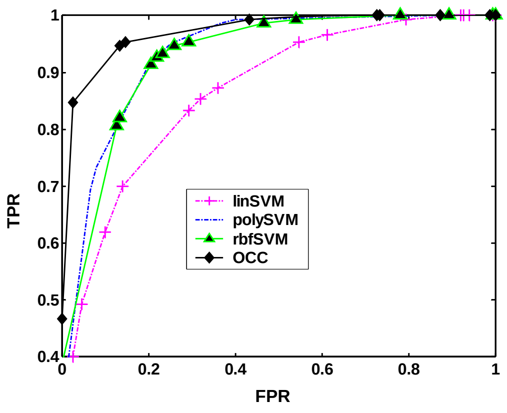


Figure 6(b): ROC for 100 features

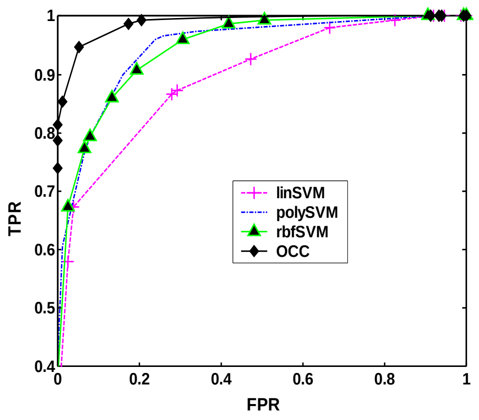


Figure 6(c): ROC for 500 features

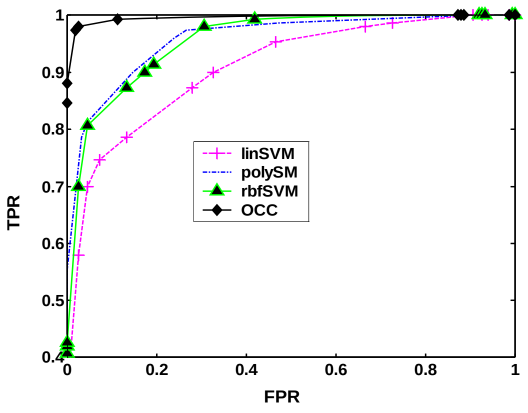


Figure 6(d): ROC for 1000 features

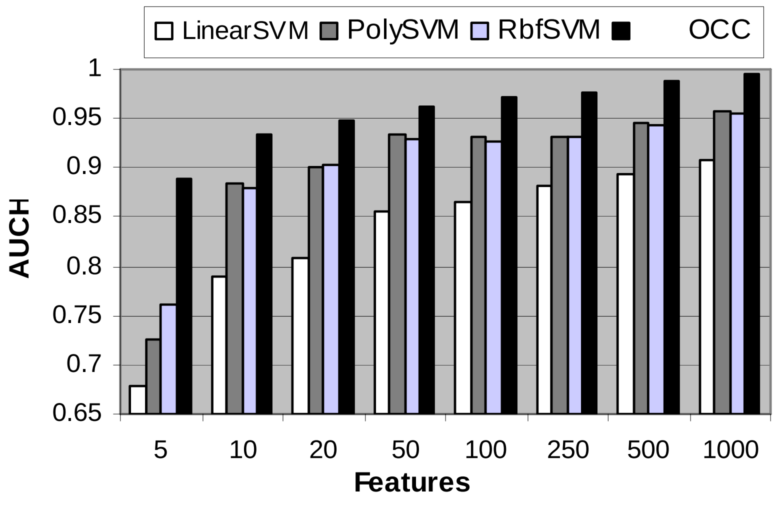


Figure 7: AUCHs of ROC curves of different classifiers

## 5.4 Overall classifier performance

It is observed from Figure 7 that AUCH of classifiers is enhanced with the increase of features. Further analysis is carried out to study the performance of classifier in terms of AUCH of AUCHs [26]. This measure has shown more compactness by incorporating the performance of a classifier with respect to the variation in the feature sets. The procedure adopted is as follows: in the first step, different AUCH values of a classifier for different feature sets are obtained. Graph is plotted between AUCH versus different feature sets as shown in Figure 8. Finally, AUCH of these AUCH curves is computed. In the second step, average AUCH of each classifier is also calculated. The difference between AUCH of AUCH and average AUCH of each classifier is determined. The value of difference represents the variation in classifier's performance with respect to the size of feature set. Higher difference indicates lower robustness of a classifier.

Bar chart in the Figure 9 shows the overall performance of classifiers in terms of AUCH of AUCHs, average AUCH and their percentile difference. It is observed that linear SVM has the lowest AUCH of AUCHs value of 0.9205 and the largest percentile difference of 7.45 (0.9205-0.846). The other two component classifiers have comparatively the same AUCH of AUCH values and relatively small percentile difference. However, OCC has the largest AUCH of AUCHs value of 0.988 and the smallest percentile difference of 2.9 (0.988-0.96). These results illustrate two main advantages of OCC, i.e. higher optimality and robustness against the variation in feature sets.

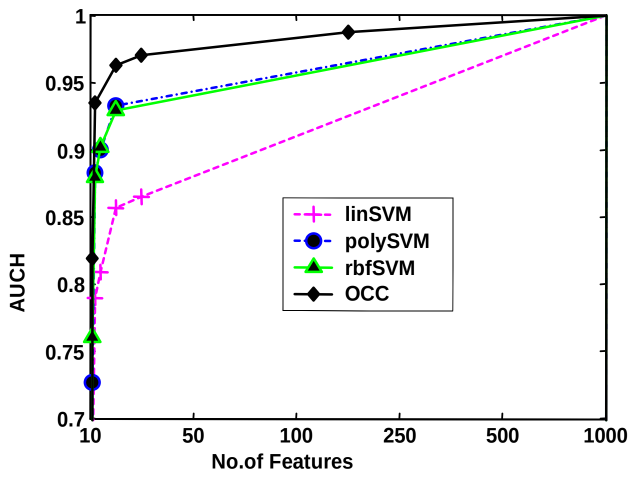
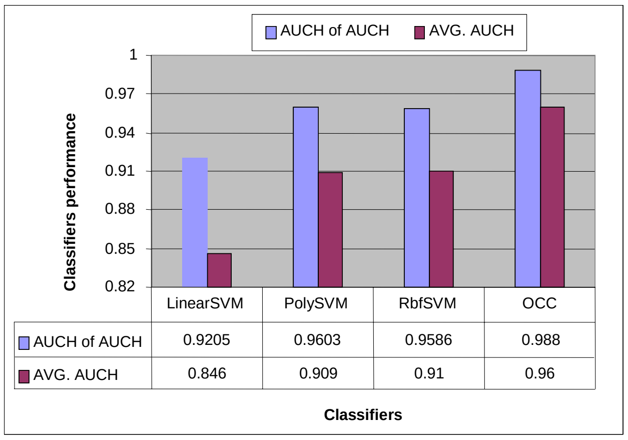


Figure 8: AUCH versus no. of features



| | LinearSVM | PolySVM | RbfSVM | OCC |
|---|---|---|---|---|
| ▨ AUCH of AUCH | 0.9205 | 0.9603 | 0.9586 | 0.988 |
| ▨ AVG. AUCH | 0.846 | 0.909 | 0.91 | 0.96 |

Figure 9: Overall performance of classifiers

## 5.5 Performance comparison of OCC with optimized SVM classifiers

The quantitative results in the Table 2 and Figure 10 highlight the improved performance exhibited by optimized SVM classifiers. Even though, the component SVM classifiers of OCC are not optimized, still OCC gives a margin of improvement as compared with optimized SVM classifiers. Table 2 shows that the optimum values of $\gamma$ and $C$ depend on the type of kernel function and the size of the feature set. It is observed that the optimal values of $\gamma$ and $C$ varies randomly with the increase of features. This table also indicates an improvement for SVM classifiers after optimization. For example, the performance of linear SVM is improved but not appreciably. SVM classifier based on RBF kernel is more accurate than linear SVM. It is also more efficient than polynomial kernel due to the lesser number of parameters to be adjusted.

Table 2: Performance comparison of OCC with Optimized SVM classifiers

| Classifiers /Feature sets | opt_LinSVM | | Opt_PolySVM | | | Opt_RbfSVM | | | OCC |
|---|---|---|---|---|---|---|---|---|---|
| | AUCH | Optimal C | AUCH | Optimal $\gamma$ | C | AUCH | Optimal $\gamma$ | C | AUCH |
| 5 | 0.6779 | 1.1 | 0.7655 | 0. 25 | 16 | 0.8587 | 8.0 | 32 | 0.8900 |
| 10 | 0.7896 | 1.2 | 0.8864 | 2.0 | 0. 06 | 0.9196 | 4.0 | 2.0 | 0.9344 |
| 20 | 0.8167 | 4.6 | 0.9071 | 0.25 | 5.65 | 0.9319 | 2.0 | 4.0 | 0.9482 |
| 50 | 0.8726 | 3.8 | 0.9609 | 0.35 | 22.62 | 0.9603 | 0.25 | 128 | 0.9629 |
| 100 | 0.9007 | 5.1 | 0.9689 | 0.17 | 32.01 | 0.9699 | 0.12 | 128 | 0.9705 |
| 250 | 0.927 | 4.4 | 0.9701 | 0.71 | 2.02 | 0.9728 | 2.01 | 4.02 | 0.9766 |
| 500 | 0.9263 | 4.8 | 0.9730 | 2.82 | 0.08 | 0.9785 | 8.1 | 2.2 | 0.9871 |
| 1000 | 0.9329 | 4.8 | 0.9754 | 2.82 | 0.176 | 0.9784 | 2.1 | 4.2 | 0.9947 |
| 5000 | 0.9529 | 5.1 | 0.9816 | 3.1 | 0.21 | 0.9846 | 2.0 | 8.0 | 0.9926 |

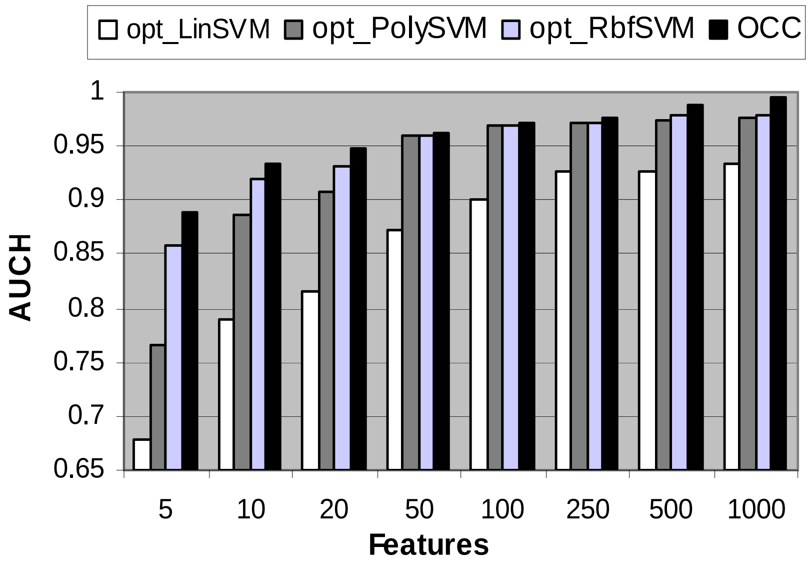


Figure 10: AUCHs of ROC curves of different classifiers

Figures 11 to 13 show the effect of varying $\gamma$ and $C$ on the performance of different kernels for 100 features. It is observed that as compared to parameter $C$ , parameter $\gamma$ has more significant effect on the performance of SVM kernels. Figure 11 shows the behavior of RbfSVM with respect to $\gamma$ and $C$ parameters. Its highest AUCH value is 0.9699 at $\gamma = 0.12$ and $C=128$. In Figure 12, the highest AUCH value of PolySVM is 0.9689 at $\gamma = 0.17$ and C=32. Figure 13 indicate the highest AUCH value of Linear SVM is 0.9007 at $C = 5.1$.

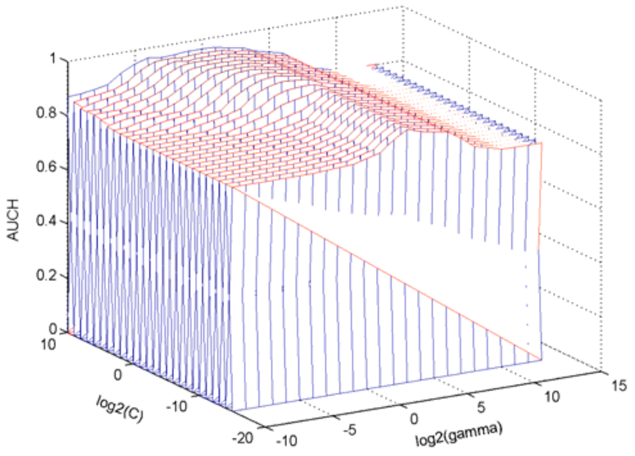


Figure 11: AUCH surface of RBF kernel parameterized by $\gamma$ and $C$ for 100 features

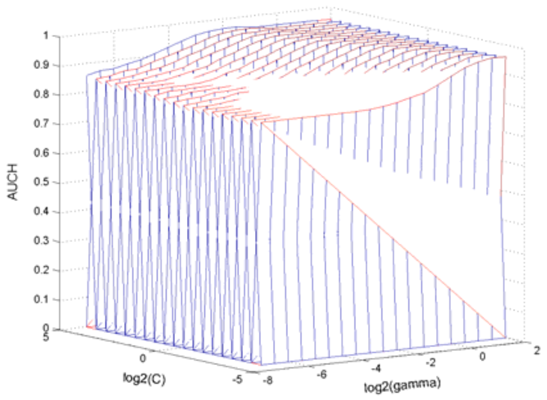


Figure 12: AUCH surface of polynomial kernel parameterized by $\gamma$ and $C$ for 100 features at $d = 3$, coefficient $r = 1$

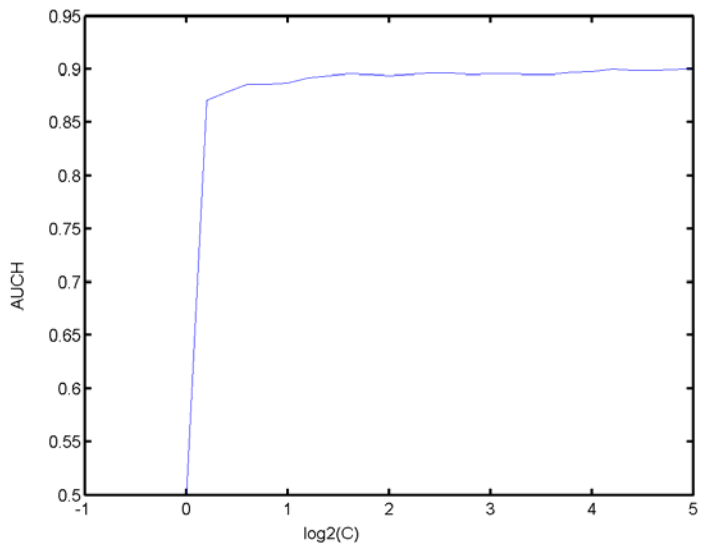


Figure 13: AUCH versus parameter C of Linear kernel for 100 features

## 5.6 Temporal cost of Optimized SVM classifiers and OCC

Table 3 shows comparison between the training and testing times of the optimized SVM classifiers and OCC. It is observed that temporal cost increases with the increase of size of feature set. In case of optimizing kernel parameters through grid search, the temporal cost depends on the grid range and its step size. In this table the training and testing time is reported for 200 data samples. Overall temporal cost of the classification models is:

$$OCC > opt\_PolySVM > opt\_RbfSVM > opt\_LinSVM$$

The optimized Linear SVM takes less training and testing time but its classification performance is poor. Our OCC takes comparatively more training and testing time but it keeps better classification performance. OCC training/testing time depends on various factors, like, training data size, length of feature set, maximum tree depth, tree nodes and the size of search space.

Table 3: Comparison between the training and testing time (in sec.)

| Classifiers /Feature sets | Opt_LinSVM | | opt_PolySVM | | Opt_RbfSVM | | OCC | |
|---|---|---|---|---|---|---|---|---|
| | Train | Test | Train | Test | Train | Test | Train | Test |
| 5 | 2.781 | 0.0050 | 201.02 | 0.0160 | 2278.4 | 0.0470 | 3612.4 | 1.470 |
| 10 | 3.001 | 0.0051 | 217.80 | 0.0310 | 1343.4 | 0.0580 | 4343.7 | 2.580 |
| 50 | 3.105 | 0.0051 | 220.11 | 0.0150 | 1864.1 | 0.0560 | 5643.1 | 2.860 |
| 100 | 4.328 | 0.0051 | 271.81 | 0.0320 | 5491.4 | 0.0780 | 9021.4 | 2.070 |
| 500 | 18.42 | 0.0151 | 1263.2 | 0.2500 | 19984.1 | 0.4780 | 40402.1 | 3.480 |
| 1000 | 36.71 | 0.0151 | 2393.1 | 0.500 | 40231.2 | 0.7651 | 57611.2 | 3.751 |

## 5.7 OCC behavior in a partially different feature space

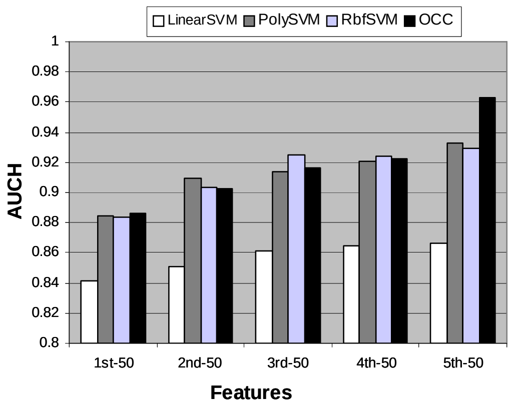
OCC is trained on a particular feature space and we study its behavior on a partially different feature space. In table 4, OCC-10 is trained for 10 feature sets. However, its performance is analyzed on the same feature space as well as on other feature spaces. This table shows that AUCH value of OCC-10 is maximum (0.934) for 10 features as compared to other partially different feature spaces. Thus, as expected, improved behavior of OCC along the diagonal path is observed . In each column, from top to bottom, there is a gradual increase in the values of AUCH. This behavior of OCC resembles a normal classifier, i.e. more information would result in higher performance. However, OCC perform randomly along horizontal. This might be due to the diverse nature of the GP search space in each run. In each run, the optimal solution may be partially/entirely different from the previous GP solution.

Table 4: OCC performance for various feature spaces

| Classifiers /Feature sets | OCC-10 | OCC-20 | OCC-50 | OCC-100 | OCC-500 | OCC-1000 | OCC-5000 |
|---|---|---|---|---|---|---|---|
| FS = 10 | **0.9344** | 0.8701 | 0.8902 | 0.8631 | 0.8776 | 0.7999 | 0.8751 |
| FS = 20 | 0.8901 | **0.9482** | 0.8920 | 0.8842 | 0.9002 | 0.8179 | 0.902 |
| FS = 50 | 0.9001 | 0.9061 | **0.9629** | 0.9111 | 0.9032 | 0.8431 | 0.9276 |
| FS = 100 | 0.9130 | 0.9190 | 0.9381 | **0.9705** | 0.9316 | 0.8930 | 0.9313 |
| FS = 500 | 0.9370 | 0.9470 | 0.9412 | 0.9481 | **0.9871** | 0.9084 | 0.9458 |
| FS=1000 | 0.9451 | 0.9500 | 0.9581 | 0.9472 | 0.9532 | **0.9947** | 0.9537 |
| FS=5000 | 0.9571 | 0.9591 | 0.9591 | 0.9651 | 0.9426 | 0.9262 | **0.9826** |

## 5.8 OCC behavior in an entirely different feature space

In this case, we analyze the performance of OCC on entirely different feature spaces but of equal size. In the first experiment, the features in the testing dataset2 are sorted and the last 250 most distinctive features are selected in ascending order. These selected features are divided into five separate but equal feature sets. Each feature set contain 50 features and the last feature set contains the most discriminant features. OCC is trained for the last feature space. Same steps are carried out in the second experiment to construct a feature space of 100 feature sets. Their experimental results are given in bar charts of Figures 14 and 15. It is observed from these figures that OCC performs well for the last feature set. Because, OCC was trained on this specific feature space, on the other feature spaces, the performance of OCC is equal to that of PolySVM or RbfSVM. This is because that SVM classifiers are tested and trained on their own feature spaces, but this is not the case for OCC. Even then, the performance of OCC is not less than the best of its component classifiers. These figures also illustrate continuous degradation in the performance of classifiers. This may be due to the gradual decrease in the discrimination power of *SIMBA* feature selection algorithm [32]. These figures show that the last feature spaces are the most discriminant.



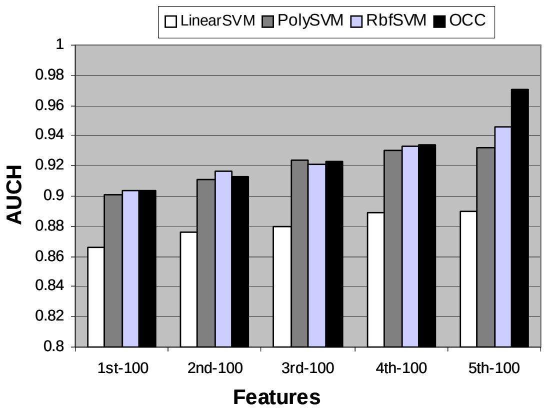


Figure 14: (left) and 15: (right) show the performance of classifiers for 50 and 100 feature sets, respectively.

An exemplary numerical expression classifier, in prefix form, developed by GP for 1000 features is given as:

*f(L,R,P)=plus(le(sin(P),le(plus(sin(L),sin(sin(P))),abs(minus(L,abs(minus(sin(sin(L)),0.12826)))))),plus(minus(R,0.15),plus(plus(sin(plus(le(0.047,abs(le(plus(R,plus(times(kozadivide(P,plus(sin(abs(L)),plus(cos(L),sin(sin(R))))),0.51),P)),le(sin(sin(L)),abs(minus(sin(L),plus(sin(plus(le(0.047,abs(minus(sin(sin(L)),0.12))),sin(L))),minus(sin(L),abs(minus(sin(sin(L)),0.12)))))))))),sin(abs(le(plus(0.047,sin(absus(minus(R,0.15),plus(plus(sin(plus(le(0.047,abs(le(plus(R,plus(times(kozadivide(P,plus(sin(abs(L)),plus(cos(L),sin(sin(R))))),0.51),P)),le(sin(sin(L)),abs(minus(sin(L),plus(sin(plus(le(0.047,abs(minus(sin(sin(L)),0.12))),sin(L))),minus(sin(L),abs(minus(sin(sin(L)),0.12))))))))))),(mnus(R,0.15),pmnus(L,plus(sin(plus(le(0.04,abs(minus(sin(abs(minus(sin(L),plus(sin(plus(le(sin(L),abs(minus(sin(sin(L)),0.12))),sin(L))),minus(sin(L),abs(minus(sin(sin(L)),0.12))))))),0.12))),sin(R))),R))))),le(sin(sin(abs(minus(sin(L),plus))))))).*

This expression shows that OCC function depends on predicted arrays (L, R, P) of the kernels, random constants, arithmetic operators and other special operators.

## 6 Conclusions

Our GP-based technique of developing composite classifier has effectively extracted useful information from individual SVM classifiers. This gain in the performance of OCC is achieved through the genetic combination of SVM classifiers without resorting to the manual exhaustive grid search. From experimental results, it is also concluded that the performance of OCC is better than the optimized SVM classifiers. During GP evolution process, OCC learns the most favorable distribution within the data space. Using the proposed combination technique, OCC can be tuned at any binary classification problem. Our investigations [25]-[26], [31] and [39] have explored the GP potential to combine the decision information from its constituent classifiers. In future, we intend to analyze the performance of GP combination method on medical data sets and other performance evaluation datasets.

**Acknowledgements**

This work is sponsored by the Ministry of Science and Technology and Higher Education Commission, Government of Pakistan under the scholarship grants No. 17-6 (0246) /Sch / 2001 and No. 17-6 (176) /Sch / 2001.

## References

[1] Duda R. O., Hart P. E., and Stork D. G. (2001), “Pattern Classification,” *John Wiley & Sons, Inc.*, New York, 2nd edition.

[2] Vapnik V. (1998), “Statistical Learning Theory,” New York: *John Wiley & Sons Inc*.

[3] Rajpoot K., and Rajpoot N. (2004), “SVM Optimization for Hyperspectral Colon Tissue Cell Classification,” *LNCS3217*, Springer-Verlag, 829-837.

[4] Moghaddam B. and Yang M. H. (2002), “Learning Gender with support faces,” IEEE Transaction on *Pattern Analysis and Machine Learning*, Vol. 24, No. 5, 707–711 .

[5] Chapelle O., Vapnik V., Bousquet O., and Mukherjee S. (2002), “Choosing Multiple Parameters for Support Vector Machines,” *Journal of Machine Learning*, Vol. 46, No. 1, 131-159.

[6] Yu-Yen Ou, Chien-Yu Chen, Shien-Ching Hwang, and Yen-Jen Oyang (Oct. 2003), “Expediting Model Selection for Support Vector Machines Based on Data Reduction,” *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, Washington D.C, USA.

[7] Runarsson T.P. and Sigurdsson S. (2004), “Asynchronous Parallel Evolutionary Model Selection for Support Vector Machines,” *Neural Information Processing - Letters and Reviews*, Vol. 3, No. 3, 59-68.

[8] Hsu C.W., Chang C.C., and Lin C.J. (2003), “A practical guide to support vector machines,” Technical report, *Department of Computer Science & Information Engineering, National* Taiwan University.

[9] Guermeur Y., Maumy M. and Sur F. (2005), “Model selection for multi-class SVMs,” *ASMDA'05*, Brest, France, 507-516.

[10] Weston J., Mukherjee S., Chapelle O., Pontil M., Poggio T., and Vapnik V. (2000), “Feature selection for SVMs,” *Advances in Neural Information Processing Systems* (NIPS), Vol. 13, MIT Press, 668-674.

[11] Staelin C. (2002), “Parameter selection for support vector machines,” Technical report, *HP Labs*, Israel.

[12] Dzeroski S. and Zenko B. (2004), “Is combining classifiers with stacking better than selecting the best one? ” *Journal of Machine Learning*, 54(3):255–273.

[13] Kittler J. & Roli F. (2001), “Multiple Classifier Systems,” *Proceedings of 2nd International Workshop, MCS2001, Cambridge*, UK, 369-377.

[14] Brown G., Wyatt J., Harris R. and Yao X. (2005), “Diversity creation methods: A survey and categorization,” *Journal of Information Fusion*, 6(1), 5–20.

[15] Ruta D. and Gabrys B. (2001), “Analysis of the correlation between majority voting error and the diversity measures, in multiple classifier systems,” *Proceedings* of *4th International Symposium on Soft Computing*, Paisley, UK, Paper No. 1824-025.

[16] Ho T. K. (2001), “Data complexity analysis for classifier combination, Multiple Classifier Systems,” *Proceedings of 2nd International Workshop, MCS2001*, Cambridge, UK, 53-67.

[17] Roli F., and Giacinto G. (2002), “Hybrid Methods in Pattern Recognition,” Chapter design of Multiple Classifier Systems, *World Scientific Publishing*, Vol. 36, 199-226.

[18] Moguerza1 J. M., Muñoz A., and Diego I. M. D. (2004), “Improving Support Vector Classification via the Combination of Multiple Sources of Information,” *Multiple Classifier Systems I, Springer-Verlag*, Vol. 3138.

[19] Majid A., Khan A. and Mirza A. M., (2005) “Combination of Nearest Neighborhood Classifiers Using Genetic Programming”, *Proceedings of International IEEE Conference (INMIC2005)*, Karachi, Pakistan.

[20] Langdon W. B. and Barrett S. J. (2004), “Genetic Programming in Data Mining for Drug Discovery,” *Evolutionary Computing in Data Mining*, Physica Verlag, 211-235.

[21] Langdon W. B. and Barrett S. J., and Buxton. B. F. (2002), “Combining Decision Trees and Neural Networks for Drug Discovery in Genetic Programming,” *Proceedings of the 5th European Conference, EuroGP'2002*, 60-70, Springer-Verlag.

[22] Langdon W. B. and Barrett S. J., and Buxton B. F. (2002), “Genetic Programming for Combining Neural Networks for Drug Discovery,” *Soft Computing and Industry Recent Applications*, Springer-Verlag, 597-608.

[23] Buxton B. F., Langdon W. B. and Barrett S. J. (2001), “Data Fusion by Intelligent Classifier Combination,” *Measurement and Control*, Vol. 34, No. 8, 229-234.

[24] Langdon W. B. and Buxton B. F (2001), “Genetic programming for combining classifiers,” *Proceedings of GECCO2001, Morgan Kaufmann*, 66-73.

[25] Majid A., Khan A., and Mirza A. M. (2004), “Improving Performance of Nearest Neighborhood Classifier Using Genetic Programming,” *Proceedings. of International conference on machine learning and its applications ICMLA’04*, Louisville, KY, USA.

[26] Khan A., Majid A., and Mirza A. M. (2004), “Combination and Optimization of Classifiers in Gender Classification Using Genetic Programming,” *Journal of Knowledge-Based Intelligent Engineering Systems (KES)*, Vol. 8, 1-11.

[27] Agarwal S., Graepel T., Herbrich R., Peled S. H., and Roth D. (2005), “Generalization bounds for the area under the ROC curve,” *Journal of Machine Learning Research*, Vol. 6, 393-425.

[28] Fawcett T. (2004), “ROC graphs: Notes and practical considerations for researchers,” Technical report, *HP Laboratories*, MS 1143, Mill Road, Palo Alto, CA 94304, USA.

[29] Langdon W. B. and Buxton B. F. (2001), “Genetic Programming for Improved Receiver Operating Characteristics,” *Multiple Classifier System, LNCS2096, Springer-Verlag*, 2-4.

[30] Scott M. J. J., Niranjan M., and Prager R. W. (1998), “Realizable classifiers: Improving operating performance on variable cost problems,” *Proceedings of Ninth British Machine Vision Conference*, Vol. 1, 304-315.

[31] Majid. A. (Dec. 2005), “Optimization and combination of classifiers using Genetic Programming,” PhD Thesis, *Faculty of Computer Science, GIK institute*, Pakistan.

[32] Bachrach R.G., Navot A., and Tishby N. (2004), “Margin based feature selection - theory and algorithms,” *Proceeding*s of the 21'st *international conference on Machine Learning (ICML’04), ACM Press*, Vol. 69, 43-49.

[33] Beveridge R. (2004), “Evaluation of face recognition algorithms version 5.0,” web site. http://www.cs.colostate.edu/evalfacerec/algorithms5.html.

[34] Banzhaf W., Nordin P., Keller R. E. and Francone F. D. (1998), “Genetic Programming: An Introduction: On the Automatic Evolution of Computer Programs and Its Applications,” *Morgan Kaufmann, Inc*. San Francisco, California.

[35] http://www.gplab.sourceforge.net/download.html

[36] http://prdownloads.sourceforge.net/svm/osu-svm-3.0.zip

[37] Langdon W. B. (2000), “Size fair and homologous tree genetic programming crossovers”, *Genetic Programming and Evolvable Machines*, Vol.1, No.1-2, 95-119.

[38] Freund Y. and Schapire R. E, (1996) “Experiments with a new boosting algorithm”, *Proceedings* of *13th International Conference on Machine Learning, Morgan Kaufmann*, 148-156.

[39] Majid A., Khan A. and Mirza Anwar M. (2005), “Intelligent Combination of Kernels Information for Improved Classification”, *Proceedings of International Conference on Machine Learning and its Applications ICMLA’05*, Los Angeles, USA.

[40] Bauer E. and Kohavi R., (1999) “An empirical comparison of voting classification algorithms: Bagging, boosting and variants,” *Journal of Machine Learning*, 24(3):173–202.

[41] http://cvc.yale.edu/projects/yalefaces/yalefaces.html

[42] http://www.lrv.fri.uni-lj.si/facedb.html

[43] http://www.uk.research.att.com/pub/data/att_faces.tar.Z

[44] http://ise.stanford.edu/class/ee368/projects2001/dropbox/project16/appendix.html