

SINGLE TO MULTIPLE KERNEL LEARNING WITH FOUR POPULAR SVM KERNELS (SURVEY)

Yassein Eltayeb Mohamed Idris¹, Li Jun²

¹MS. Student, School of Automation & Electric Engineering, Lanzhou Jiaotong University, Lanzhou, Gansu, China

²Professor, School of Automation & Electric Engineering, Lanzhou Jiaotong University, Lanzhou, Gansu, China

Email Correspondence: yasinaltyb@yahoo.com

Abstract

Machine learning applications and pattern recognition have gained great attention recently because of the variety of applications depend on machine learning techniques, these techniques could make many processes easier and also reduce the amount of human interference (more automation). This paper research four of the most popular kernels used with Support Vector Machines (SVM) for Classification purposes. This survey uses Linear, Polynomial, Gaussian and Sigmoid kernels, each in a single form and all together as un-weighted sum of kernels as form of Multi-Kernel Learning (MKL), with eleven datasets, these data sets are benchmark datasets with different types of features and different number of classes, so some will be used with Two-Classes Classification (Binary Classification) and some with Multi-Class Classification. Shogun machine learning Toolbox is used with Python programming language to perform the classification and also to handle the pre-classification operations like Feature Scaling (Normalization). The Cross Validation technique is used to find the best performance Out of the suggested different kernels' methods. To compare the final results two performance measuring techniques are used; classification accuracy and Area Under Receiver Operating Characteristic (ROC). General basics of SVM and used Kernels with classification parameters are given through the first part of the paper, then experimental details are explained in steps and after those steps, experimental results from the steps are given with final histograms represent the differences of the outputs' accuracies and the areas under ROC curves (AUC). Finally best methods obtained are applied on remote sensing data sets and the results are compared to a state of art work published in the field using the same set of data.

Keywords: Machine Learning, Classification, SVM, MKL, Cross Validation and ROC.

1. INTRODUCTION

This paper studies a classification technique, which is sorted under supervised learning. Supervised learning involves using examples with known labels for the training process to find a general relationship (rule), that rule maps the inputs (features of the training examples) to the outputs (labels of the training examples) [1]. the classification methods and techniques used in this paper could be presented as follows:

1.1 Support Vector Machines (SVM)

SVM is a supervised learning method which is used for classification, regression and other operations. In this research it can be defined as the state-of-art approach for classification problems [2, 3]. The basic principal behind the SVM is to classify a given set of training data points into two categories (two classes), this mission achieved by forming a separating hyper plane maximizes the distance (the margin) to the closest training data points of each class, this operation would form a geometric or functional margin. This would ensure a better generalization i.e. enhances the ability of the classifier to classify new (unseen) points. As it is shown in Figure 1, where three possible hyper planes to separate the two groups of points are drawn. The hyper plane H_3 does not classify well all the training data points. H_1 classifies all the training data points, but it's not at the maximum margin from their nearest training data points, so

any new instance that would be to the right of the black circles could be misclassified. The hyper plane H_2 classifies all the training data points well and it is the separating hyper plane with the maximum margin [4,5,6].

The functional keys of SVM are that it maximizes the functional margin and minimizes the structural risk error which gives it better generalization ability.

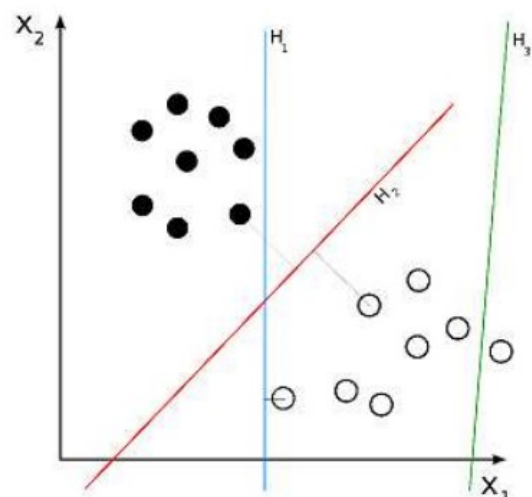


Fig -1: SVM Separating Hyper Planes

- For binary classification it could be represented as follows:
Given a set of points as training data T as follows [7, 8]:

$$T = \{(x_i, y_i) | x_i \in \mathbf{R}^D, y_i \in (-1, 1)\}_{i=1}^N \quad (1)$$

Where x_i represents D-dimensional features for one instance (point) of data set, y_i represents the label or the class of the given instance as positive (1) or negative (-1), D represents the number of features and N represents the number of training instances (points).

SVM Finds the separating hyper plane to maximize the margin between two parallel hyper planes, as shown in figure 2 and divides the points into their classes in which they belongs.

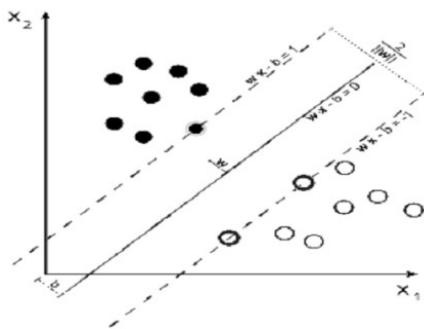


Fig -2: Support Vectors separating hyper plane

The equation of the separating hyper plane is given as:

$$w \cdot x - b = 0 \quad (2)$$

Where w is a normal vector perpendicular to the separating hyper plane and b is the hyper plane offset, where the offset of the hyper plane from the origin of the direction of the normal vector w is calculated from the parameter $\frac{b}{\|w\|}$.

To maximize the margins, SVM chooses suitable values of normal vector w and plane offset b .

Figure 2 shows two parallel hyper planes, those represented as follows:

$$w \cdot x - b = 1 \ \& \ w \cdot x - b = -1 \quad (3)$$

If a linearly separable set of data is given, then the classifier selects two parallel hyper planes (support vectors) with no data instance lies between them. The distance between the two support vectors is calculated geometrically through $\frac{2}{\|w\|}$, SVM then suppose to minimize $\|w\|$ to maximize the distance between the support vectors.

Due to mathematical difficulty in optimization that results because of the dependency on the absolute value of w , which is a non-convex optimization problem that is very difficult to solve, the equation should be possible to solve by replacing $\|w\|$ with $\frac{1}{2}\|w\|^2$. This would generate the following quadratic programming (QP) optimization:

$$\min \frac{1}{2} \|w\|^2 + C \sum_{i=1}^N \xi_i \quad (4)$$

$$s. t. \ y_i(w^T x_i + b) \geq 1 - \xi_i \quad \forall i$$

Where C is predefined positive trade-off parameter between classification error and model simplicity and ξ is a vector of slack variables. Due to SVM objective which is maximizing the margins between the two classes, then the optimization problem could be solved indirect through Lagrangian dual function [9] as follows:

$$\max \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i,j=1}^N \alpha_i y_i x_i^T x_j$$

$$w. r. t \ \alpha \in R_+^N \quad (5)$$

$$s. t \ \sum_{i=1}^N \alpha_i y_i = 0 \ \& \ C \geq \alpha_i \geq 0 \quad \forall i$$

Where α is a vector of dual variables that corresponds to separation constraints. Solving this yields $w = \sum_{i=1}^N \alpha_i y_i x_i$ and the discriminate function can be represented in the following new form:

$$f(x) = \sum_{i \in s} \alpha_i y_i x_i^T x + b \quad (6)$$

Where s is the support vectors indices.

- For multi-class classification many approaches could be used, but the proposed toolbox (Shogun Machine Learning Toolbox) uses One against All method. For conventional One Against All method one SVM is constructed per class to distinguish the samples of one class from the samples of all other classes.

Let consider an n -class problem, using One Against All method, then n direct decision functions are determined to separate each class from the rest.

Let the i^{th} decision function that has a maximum margin to separate class i from the remaining classes, be

$$f_i(x) = w_i^T \Phi(x) + b_i \quad (7)$$

Where $\Phi(x)$ is the mapping function that maps x into l -dimensional feature space. After optimization the training data from class i should satisfy $f_i(x) \geq 1$, and other classes training points should satisfy $f_i(x) \leq -1$. this would give the general decision function to classify a given point or instance of data as it belongs to class i if:

$$f_i(x) > 0 \quad (8)$$

But a problem appears when a given point or instance of data is classified into more than one classes like the black square data sample 1 in figure 3, or not classified by any class like black circle data sample 2 in figure 3.

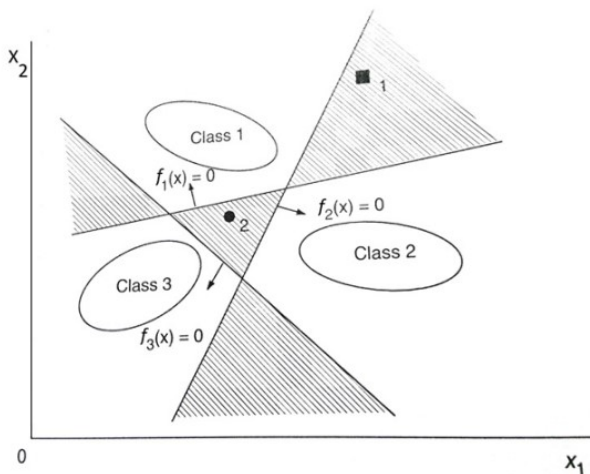


Fig -3: Three-class problem with two-dimensional input to show unclassifiable regions by one-against-all formulation

To solve this, continuous decision functions are proposed for classification instead of the direct decision function. That means, data instance or point (x) is classified due to the following rule:

$$\arg \max_{i=1, \dots, n} f_i(x) \tag{9}$$

Following this rule, data sample 1 in the figure could be classified into the second class (class 2), and data sample 2 could be classified into the first class (class 1).

1.2 Kernel Concept

In the case of training a non-linearly separable data, a hyper plane may not provide high generalization ability.

There for to enhance the classification, the original input space is mapped into a new high dimensional space (Feature Space) [7, 6, 10]. That would create a nonlinear vector function $\Phi(x) = \langle \phi_1(x), \dots, \phi_l(x) \rangle^T$ which maps the feature space input vector from N-dimensional to l-dimensional. Due to the previous function the linear decision function for a given features could be rewritten as:

$$f(x) = w^T \Phi(x) + b \tag{10}$$

Now introducing the general representation for the kernel function $k(x_i, x_j)$ the optimization problem can be represented as follows:

$$\max \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j k(x_i, x_j)$$

$$w.r.t \ \alpha \in \mathbf{R}_+^N \tag{11}$$

$$s.t \ \sum_{i=1}^N \alpha_i y_i = 0 \ \& \ C \geq \alpha_i \geq 0 \ \forall i$$

The discriminate function can be written as:

$$f(x) = \sum_{i=1}^N \alpha_i y_i k(x_i, x) + b \tag{12}$$

SVM has many kernel functions $k(x_i, x_j)$, this research uses four of them as single kernel and to make a linear combination of them. These Kernels are:

1.2.1 Linear Kernel

Linear kernel is the basic form of the SVM that is usually used for linear separable data sets [11]. The kernel uses separation hyper-plane with maximum margins to enhance the generalization process [4, 6]. The mathematical formula of the kernel could be presented as follows:

$$K(x, x') = \phi^T(x) \phi(x') \tag{13}$$

For $K(x, x')$ is a positive semidefinite kernel, the optimization problem comes as a quadratic programming problem. And because equation (1.6) $\alpha = 0$ is a feasible solution, so the problem has a global optimum solution. The Karush-Kuhn –Tucker [6, 12] (KKT) complementary conditions given by:

$$\alpha_i \left\{ y_i \left\langle \sum_{j=1}^N y_j \alpha_j K(x_i, x_j) + b \right\rangle - 1 + \xi_i \right\} = 0$$

$$\text{for } i = 1, \dots, N \tag{14}$$

$$(C - \alpha_i) \xi_i = 0 \quad \text{for } i = 1, \dots, N$$

$$\alpha_i \geq 0, \quad \xi_i \geq 0 \quad \text{for } i = 1, \dots, N$$

The decision function is given as:

$$f(x) = \sum_{i \in S} \alpha_i y_i K(x_i, x) + b \tag{15}$$

Where b is set as:

$$b = y_i - \sum_{i \in S} \alpha_i y_i K(x_i, x_j) \tag{16}$$

Where x_j is an unbounded support vector. The average is taken to ensure the stability of the calculation as follows:

$$b = \frac{1}{|U|} \sum_{j \in U} \{ y_j - \sum_{i \in S} \alpha_i y_i K(x_i, x_j) \} \tag{17}$$

U is the set of the unbounded support vector indices. And when given a new set of data for classification, the following decision function would be used:

$$X \in \begin{cases} \text{Class 1} & \text{if } f(x) > 0 \\ \text{Class 2} & \text{if } f(x) < 0 \end{cases} \tag{18}$$

If $f(x) = 0$, then X is unclassifiable. The pervious explanation is for two classes' classification, and for multi-classes classification refer to equation (9) .

1.2.2 Polynomial Kernel

Polynomial kernel is one of the famous kernels used with SVM in the case of linearly inseparable datasets. The kernel is represented by:

$$K(x, x') = (x^T x' + 1)^d \tag{19}$$

Where d is a natural number that represents the polynomial kernel degree. The inseparable one-dimensional example of three instances shown in figure 4, could be solved using the polynomial kernel with $d = 2$, the dual problem is as follows:

$$\max Q(\alpha) = \alpha_1 + \alpha_2 + \alpha_3 - (2\alpha_1^2 + \frac{1}{2}\alpha_2^2 + 2\alpha_3^2 - \alpha_2(\alpha_1 + \alpha_3)) \tag{20}$$

$$\text{s.t } \alpha_1 - \alpha_2 + \alpha_3 = 0, \quad C \geq \alpha_i \geq 0 \tag{21}$$

$$\text{for } i = 1, 2, 3. \tag{21}$$

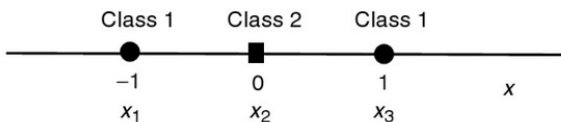


Fig -4: Three data instances from inseparable one-dimensional case

From equation (21), $\alpha_2 = \alpha_1 + \alpha_3$. Applying the value of α_2 into (20) gives:

$$Q(\alpha) = 2\alpha_1 + 2\alpha_3 - (2\alpha_1^2 - \frac{1}{2}(\alpha_1 + \alpha_3)^2 + 2\alpha_3^2), \tag{22}$$

$$C \geq \alpha_i \geq 0 \quad \text{for } i = 1, 2, 3.$$

From:

$$\frac{\partial Q(\alpha)}{\partial \alpha_1} = 2 - 3\alpha_1 + \alpha_3 = 0, \tag{23}$$

$$\frac{\partial Q(\alpha)}{\partial \alpha_3} = 2 + \alpha_1 - 3\alpha_3 = 0$$

Solving (23) gives $\alpha_1 = \alpha_3 = 1$, therefore the optimum solution for $C \geq 2$ is:

$$\alpha_1 = 1, \quad \alpha_2 = 2, \quad \alpha_3 = 1.$$

When $C \geq 2$, then $x = -1, 0,$ and $1,$ are support vectors. Applying in equation (16), gives $b = -1$, and the decision function would be:

$$f(x) = (x - 1)^2 + (x + 1)^2 - 3 = 2x^2 - 1$$

The boundaries of the decision are given as $x = \pm\sqrt{2}/2$. It is clear through following the value of $f(x)$ in figure 5 that the margin for class 2 is larger than that for class one in the input space.

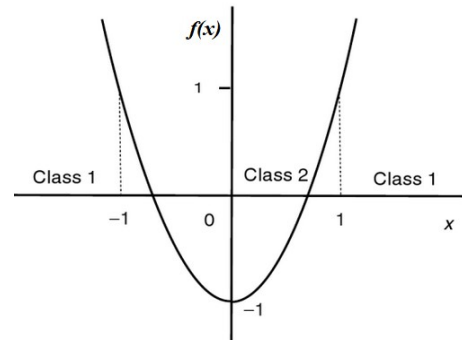


Fig -5: Polynomial kernel's decision function for inseparable one-dimensional case

1.2.3 Gaussian Kernel

The Gaussian Radial Basis Function kernel or RBF is one of the popular kernel functions used in various classification techniques [6, 13]. This kernel and other kernels provide the SVM with nonlinearity ability of classification. The kernel function could be shown as follows:

$$K(x, x') = \exp(-\gamma \|x - x'\|^2) \tag{24}$$

Where γ is a positive parameter to control Gaussian function radius. Applying the kernel function to equation (15), this yields the following decision function:

$$f(x) = \sum_{i \in S} \alpha_i y_i \exp(-\gamma \|x_i - x\|^2) + b \tag{25}$$

Figure 6 shows the classification mapping effect of the RBF's decision function.

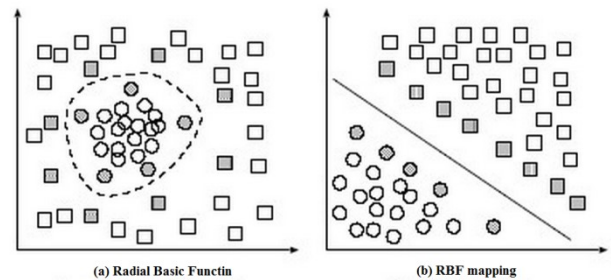


Fig -6: RBF decision function's mapping for two-dimensional case

1.2.4 Sigmoid Kernel

The sigmoid kernel is mostly known with the neural networks, but it is still a useful kernel that could be used with SVM to work out many applications [14, 15]. The kernel function could be represented as follows:

$$K(x, x') = \tanh(ax^T x' + r) \tag{26}$$

Where a and r are parameters to control the polynomial function, Applying the kernel function to equation (15) yields the following decision function:

$$f(x) = \sum_{i \in S} \alpha_i y_i \tanh(ax^T x' + r) + b \tag{27}$$

1.3 MKL

Multiple Kernel Learning (MKL), is an approach used with SVM that involves combining a number of kernels to overcome the process of kernel selection [16,17]. This study applies an un-weighted linear combination of the previous mentioned kernels and compare it to a single kernel performance. A general function for combination of kernels $k_\eta(\mathbf{x}_i, \mathbf{x}_j)$ could be represented as follows:

$$k_\eta(\mathbf{x}_i, \mathbf{x}_j) = f_\eta(\{k_m(\mathbf{x}_i^m, \mathbf{x}_j^m)\}_{m=1}^P) \tag{28}$$

Where P is feature representations of given data instances for $\mathbf{x}_i = \{\mathbf{x}_i^m\}_{m=1}^P$, where $\mathbf{x}_i^m \in R^{D_m}$, and D_m is feature representation dimensionality. η (Eta) parameterizes the combination function of kernels (weights of the kernels). The linear combination function could be presented as follows:

$$k_\eta(\mathbf{x}_i, \mathbf{x}_j) = \sum_{m=1}^P \eta_m k_m(\mathbf{x}_i^m, \mathbf{x}_j^m) \tag{29}$$

The primal form of MKL optimization problem could be obtained from the set of equations (4) which could be presented as follows [18]:

$$\begin{aligned} \min \quad & \frac{1}{2} \left(\sum_{m=1}^P \|\mathbf{w}_m\| \right)^2 + C \sum_{i=1}^N \xi_i \\ \text{w.r.t. } \quad & \mathbf{w}_m \in R^{D_m}, \xi \in R^N, b \in R \end{aligned} \tag{30}$$

$$\text{s.t. } \xi_i \geq 0 \text{ and } y_i \left\{ \sum_{m=1}^P \langle \mathbf{w}_m, \Phi_m(\mathbf{x}_i) \rangle + b \right\} \geq 1 - \xi_i, \forall i = 1, \dots, N$$

Where the problem's solution could be written as $\mathbf{w}_m = \eta_m \mathbf{w}'_m$ with $\eta_m \geq 0, \forall m = 1, \dots, P$ and $\sum_{m=1}^P \eta_m = 1$. It should be noticed that the ℓ_1 -norm of η is constrained to one, while the ℓ_2 -norm of \mathbf{w}_m is penalized in each block m separately. The constrained or penalized variables of ℓ_1 -norm tend to have sparse optimal solutions, while penalized variables of ℓ_2 -norm don not.

The dual for the previous problem was driven by taking problem (D_m), squaring the constraint on beta, multiplying the constraint by 1/2 and finally substituting $\frac{1}{2}\beta^2 \mapsto \beta$ which would lead to the following MKL dual [19]:

$$\begin{aligned} \min \quad & \beta - \sum_{i=1}^N \alpha_i \\ \text{w.r.t. } \quad & \beta \in R, \alpha \in R^N \end{aligned} \tag{31}$$

$$\begin{aligned} \text{s.t. } \quad & 0 \leq \alpha_i \leq 1C, \sum_{i=1}^N \alpha_i y_i = 0, \\ & \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j k_m(\mathbf{x}_i, \mathbf{x}_j) \leq \beta, \\ & \forall m = 1, \dots, P \end{aligned}$$

Where $k_m(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi_m(\mathbf{x}_i), \Phi_m(\mathbf{x}_j) \rangle$. The previous set represents the formula used in shogun toolbox.

1.4 Remote Sensing

Remote sensing could be defined in general as the methods or techniques used to acquire information about an object or phenomena without making physical contact with it [19, 20,21]. but it is most known for acquiring information about the surface of the earth without being in direct physical contact with that surface.

As part of this paper to use the best kernel method in an advanced application, study results are used on earth surface remote sensing data set used in a paper titled "Classifying a high resolution image of an urban area using super-object 4 information." [22], then accuracy results obtained from this research are compared to results in the mentioned paper.

2. EXPERIMENTAL SETUP

This part presents the data sets, tools and techniques those used to perform the classification and find the performance differences between the different suggested methods.

2.1 Data Sets

The data sets used in this research are bench mark data from real world sources [23], they have different ranges in classes from 2 to 11, and also there features are different in the range and data types. For some of the sets like the DNA set, they had to be treated before they could be fed into the kernels, the processes of transforming features, selecting features or extracting features from given information, are known as Feature Engineering.

The variety in data sets used with the thesis, allows a better chance for judging the performance of the different suggested methods. Main attributes of those data sets are given in table 1.

Table -1: Data sets' general attributes

Data Set	Number of Instances	Number of Features	Number of Classes
Breast Cancer	569	30	2
DNA	106	57	2
ECG	65536	9	2
EEG Eye State	14980	14	2
SPECT Heart	267	22	2
Abalone	4177	8	3
Balance Scale	625	4	3
Iris	150	4	3
Wine	13	178	3
Car Evaluation	1728	6	4
Vowel Recognition	10	990	11

Finally the data set used for remote sensing are for high resolution aerial image with 9 types of urban land cover. Multi-scale spectral, size, shape, and texture information are used for classification. There are a low number of training samples for each class (14-30) and a high number of classification features (148). the source of data is Institute for Global Environmental Strategies; 2108-11 Kamiyamaguchi, Hayama, Kanagawa, 240-0115 Japan[24]. Also available on UCI machine learning repository [23]. these data have been used in two papers [22, 25]. Figure 7 shows a color infrared image of the study area.

Classes were sampled into 9 categories; Asphalt 59 instances, Building 122 instances, Car 36 instances, Concrete 116 instances, Grass 112 instances, Pool 29 instances, Shadow 61 instances, Soil 34 instances and Tree 106 instances.



Fig -7: Infrared Image of remote sensing study area

2.2 Normalization

Input variable (features) those used in classification or in other machine learning operations, are usually obtained from different sources with different physical meaning, the variety of sources gives data sets different ranges. The dispersion in values of features could affect the performance of the classification, as slowing the classification process or causing computation problems due to the limited memory access in some computers.

Normalization Technique [2,6] is used to solve most of the pervious problems. In this dissertation data sets are classified with and without normalization under the four suggested kernels, to show the effect of normalization for different data sets and different kernels. Normalization scales the input variables' range into $\{[0, 1]$ or $[-1, 1]\}$.

In this paper normalization is used to get the feature scaling into the range $\{[0, 1]\}$, through equation (32).

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (32)$$

The same Scale of normalization used on training data should be used on testing and cross validation sets, which may lead to values greater than 1 or less than 0 in the tested data. To achieve this mission a library from scikit Learn (Python programming language machine learning toolkit) [26] is used. The code to normalize data is as follows:

```
>>> min_max_scaler=preprocessing.MinMaxScaler()
>>>Normalized_Train_Data=min_max_scaler.fit_transform(Original_Train_Set)
>>>Normalized_Test_Data=min_max_scaler.transform(Original_Test_Set)
```

2.3 Cross Validation Technique

This technique used to ensure the classifier ability of generalization, within given set of data, applying this method by dividing data into training, cross validation and testing groups would allow a chance of testing the classifier after choosing a parameter or set of parameters through cross validation technique.

In this research when this technique introduces data were divided as 60% training, 20% cross validation and 20% testing except for the remote sensing set, where 508 instances were provided for testing , almost 23% of them used for cross validation and the rest for the final testing. The number of training data was kept as the original source provided for the sake of results' comparison.

2.4 Performance Measuring

Among a number of techniques used to measure classification performance, is the Classification Accuracy and Area Under Curve (AUC), where the curve is the Receiver Operating Characteristic curve, or ROC curve.

The Classification Accuracy measures how often a given classifier finds the correct prediction. It's simply the ratio between the correctly classified instants and the total number of instances.

$$\text{accuracy} = \frac{\# \text{ correctly classified instances}}{\# \text{ total instances}} \quad (33)$$

For Accuracy measuring does not make any distinction between classes, i.e. correctly identified instances for class 0 and class 1, are treated the same. In some applications, some classes are more important to be classified correctly, so in such case accuracy measuring would not give a proper view of the classification. To see more of the classification performance, other methods are used like confusion matrix, per-class accuracy, log-loss, F1 score and AUC [27].

In this paper AUC is used through generating ROC. Receiver Operating Characteristic curve (ROC) got its name from a paper published in the 1970s by Charles Metz with the name "Basic Principles of ROC analysis." [28]. ROC curve shows the classification sensitivity through plotting the rate of true positive instances to false positive instances. That means it shows how many correct positive instances could be gained if more and more false positive are allowed.

A perfect classifier with no mistake would yield a true positive rate of 100% without a single false positive in it.

Through this paper experiments, ROC curves are shown for all classes found by different kernels, then AUC is obtained from these ROC curves as a single value to evaluate and compare the different classifiers. A good classifier yields a ROC curve with a wider area under the curve, i.e. the true positive rate gets to 1.0 (equals to 100%) as quickly as possible.

For multi-classes classification the AUC is calculated for each class alone (one class vs. all classes), finally average AUC of all classes is used to evaluate the classifier.

2.5 Software and System Specification

Hardware Intel Processor i7, Frequency 2.67 GHz, Ram 8 GB.

Operating System Ubuntu 12.04 (precise) 64 bit.

Programming Language Implementation is in python language v 2.7 [29, 30].

IDE Programming is done with the Eclipse IDE environment.

SHOGUN Toolbox API for machine learning tool [31], version 2.1.0.

3. EXPERIMENTAL STEPS AND RESULTS

This part explains the steps followed to find kernel methods' best performance using single kernel and then using un-weighted combination of the proposed kernels. It also shows the most important results through the experiments. the final part shows the experiments and the results on remote sensing data sets and compare the results to other results obtained by other research.

3.1 First Step (Single Kernel Initial Values using 70% Training)

In this step arbitrary values are assigned to the parameters, then data shuffled 100 times under each of the proposed kernels. The aim of this stage is to try the kernels with normalization and without normalization, and then due to the outcome a decision of using or not using the normalization technique is taken for each kernel with each data set.

This part gets the results of training 70% of the data and using the rest for testing the classification accuracy. The aim is to show the difference in the accuracy under different training rates of the data (later using 60% as training) and to find the efficiency of using the normalization technique. Kernels' parameters are set as follows:

- Linear Kernel: $C = 1$
- Polynomial Kernel: $C = 1$ and $d = 3$
- Gaussian Kernel: $C = 1$ and $\gamma = 0.1$
- Sigmoid Kernel: $C = 1$ and $a = 0.01$

some of this stage results are given in the following tables:

Table -2: Shuffling Abalone data set for 100 times without normalization

Kernel	Highest Accuracy	Lowest Accuracy	Average Accuracy
Linear	0.564940239044	0.521115537849	0.544541832669
Polynomial	0.548207171315	0.498007968127	0.521314741036
Gaussian	0.565737051793	0.509163346614	0.537657370518
Sigmoid	0.214342629482	0.145019920319	0.180605577689

Table -3: Shuffling Abalone data set for 100 times with normalization

Kernel	Highest Accuracy	Lowest Accuracy	Average Accuracy
Linear	0.569721115583	0.521912350598	0.544111553785
Polynomial	0.572908366534	0.527490039841	0.547569721116
Gaussian	0.581673306773	0.530677290837	0.557354581673
Sigmoid	0.537848605578	0.500398406375	0.520374501992

The average accuracy is calculated through the shuffling as follows:

$$\text{Average Accuracy} = \frac{\sum_{i=1}^{\# \text{ Shuffling}} \text{Accuracy}_i}{\# \text{ Shuffling}} \quad (34)$$

Table -4: Shuffling Vowel Recognition data set for 100 times without normalization

Kernel	Highest Accuracy	Lowest Accuracy	Average Accuracy
Linear	0.808080808081	0.707070707071	0.75430976431
Polynomial	0.673400673401	0.538720538721	0.604276094276
Gaussian	0.858585858586	0.703703703704	0.782861952862
Sigmoid	0.56228956229	0.434343434343	0.505218855219

Table -5: Shuffling Vowel Recognition data set for 100 times with normalization

Kernel	Highest Accuracy	Lowest Accuracy	Average Accuracy
Linear	0.700336700337	0.592592592593	0.647878787879
Polynomial	0.794612794613	0.686868686869	0.733569023569
Gaussian	0.993265993266	0.939393939394	0.977306397306
Sigmoid	0.542087542088	0.417508417508	0.471043771044

3.2 Second Step (Single Kernel using 60% as Training)

In this step just 60% of the data set is used to train the classifier, while the rest is used as testing set. Based on normalization results from the previous step, each kernel is used once with each data set. The aim of this step is to compare the difference in accuracy by decreasing the training groups by 10% and also to prepare new specific sets out of the shuffling for the next stage, where the concept of the cross validation would be introduced.

Parameters are set to the same values as the previous stage. This operation illustrated for the linear kernel in table 6.

3.3 Third Step (Kernels' Tuning)

This step is made for setting the proper values for the kernels used in previous tables. To find the best values for the given parameters, data sets had to be divided into three sets (training set, cross validation set and testing set), where the cross validation set is used to find the best value (tune the kernel) and finally the testing set is used to make the final judgment of the results.

Table -6: The performance of shuffling Linear Kernel 100 times under 60% as training data

Data Set	Highest Accuracy	Lowest Accuracy	Average Accuracy	Normalization
Abalone	0.5639952153	0.5233253588	0.544659090909	No
Balance Scale	0.9484126984	0.8888888889	0.917103174603	No
Breast Cancer	0.995614035	0.951754386	0.97399122807	Yes
Car Evaluation	0.8656069364	0.8236994219	0.843598265896	No
DNA	0.9090909091	0.6136363636	0.76295454555	Yes
ECG	0.9792485218	0.9748998664	0.976785809651	No
EEG Eye State	0.6504254964	0.6257300184	0.641181378275	No
Iris	1.0	0.933333	0.978	No
SPECT Heart	0.9065420561	0.7196261682	0.808878504673	No
Vowel Recognition	0.797979798	0.6944444444	0.749393939394	No
Wine	1.0	0.91780821918	0.978219178082	No

Results from the third step are shown in tables starting from table 7, with some figures show the tuning process are given starting from figure 8.

Table -7: Setting Linear Kernel parameter

Data Set	Best C parameter	Validation data Accuracy	Test data Accuracy	Area Under ROC Curve
Abalone	0.614928	57.12%	55.08%	0.753
Balance Scale	2	96.82%	93.65%	0.727
Breast Cancer	1	100%	99.13%	1.0
Car Evaluation	0.605576	89.01%	85.55%	0.916
DNA	0.07	95.45%	68.18%	0.868
ECG	0.016636	97.02%	96.49%	0.977
EEG Eye State	0.000311	66.38%	65.56%	0.685
Iris	0.456777	93.33%	100%	0.863
SPECT Heart	1	92.45%	88.89%	0.875
Vowel Recognition	17.044785	85.35%	82.32%	0.892
Wine	0.1	100%	100%	1.0

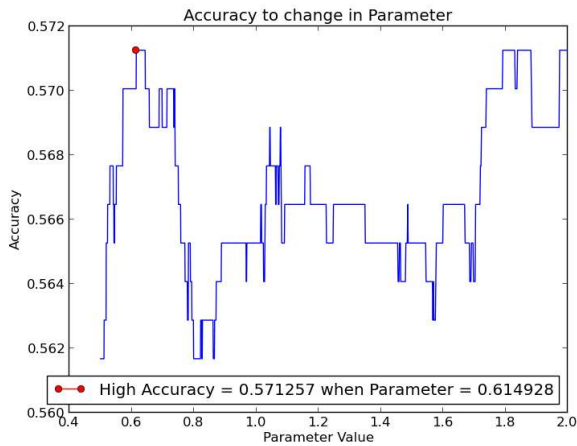


Fig -8: Change in accuracy while C changes for classifying Abalone Data set under linear kernel

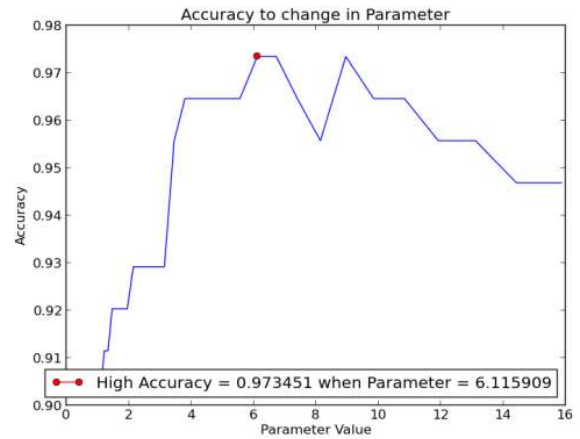


Fig -9: Tuning of Breast Cancer γ parameter under Gaussian kernel

Table -8: Setting Polynomial Kernel parameters

Data Set	Best C parameter	Best Degree parameter	Validation data Accuracy	Test data Accuracy	Area Under ROC Curve	Normalization
Abalone	0.707	2	54.61%	56.63%	0.708	Yes
Balance Scale	300	2	100%	99.2 %	0.949	Yes
Breast Cancer	62	2	95.57%	92.17%	0.986	Yes
Car Evaluation	25	7	97.68%	95.95%	0.989	Yes
DNA	1.5	2	90.91%	95.45%	1	No
ECG	57	19	99.30%	99.27%	0.998	No
EEG Eye State	1	2	55.10%	55.12%	0.64	Yes
Iris	5	2	93.33%	90.0%	0.948	No
SPECT Heart	1.3	2	83.02%	88.89%	0.877	No
Vowel Recognition	3500	11	95.45%	94.95%	0.991	No
Wine	2650	3	100%	89.19%	0.987	No

Table -9: Setting Gaussian Kernel parameters

Data Set	Best C Parameter	Best Gamma parameter	Validation data Accuracy	Test data Accuracy	Area Under ROC Curve	Normalization
Abalone	1.648774	0.114	56.64%	57.47%	0.693	Yes
Balance Scale	0.7	1.5	89.68%	91.27%	0.976	Yes
Breast Cancer	0.359742	6.5	97.34%	90.43%	0.99	Yes
Car Evaluation	2	0.15	98.84%	98.26%	0.999	Yes
DNA	0.5	3	90.91%	86.36%	0.917	Yes
ECG	5.35	0.3	99.42%	99.46%	0.999	No
EEG Eye State	1	3.5	55.47%	55.09%	0.6	Yes
Iris	1	1.35	96.66%	96.66%	0.838	Yes
SPECT Heart	1	4	90.56%	81.48%	0.897	No
Vowel Recognition	1.51	0.12	97.98%	95.96%	0.969	Yes
Wine	1.5	5	100%	97.3%	0.996	Yes

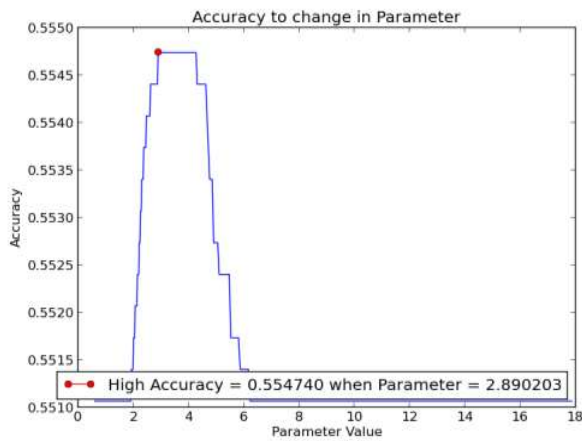


Fig -10: Tuning of EEG Eye State γ parameter under Gaussian kernel

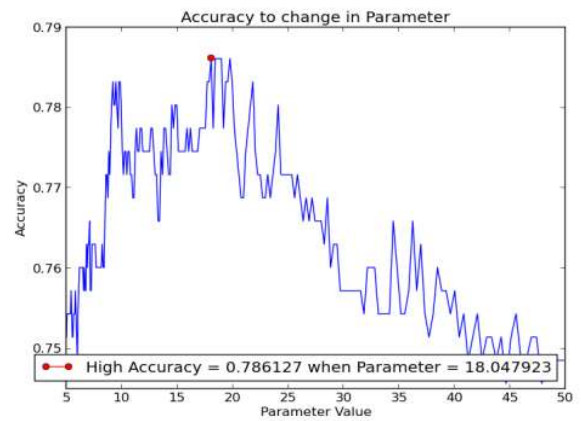


Fig -12: Tuning of Car Evaluation α parameter under sigmoid kernel

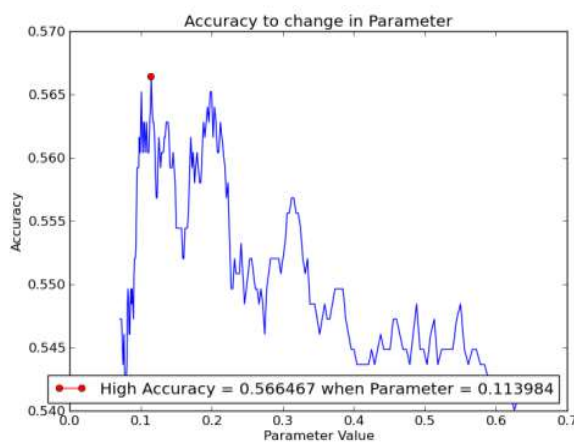


Fig -11: Tuning of Abalone γ parameter under Gaussian kernel

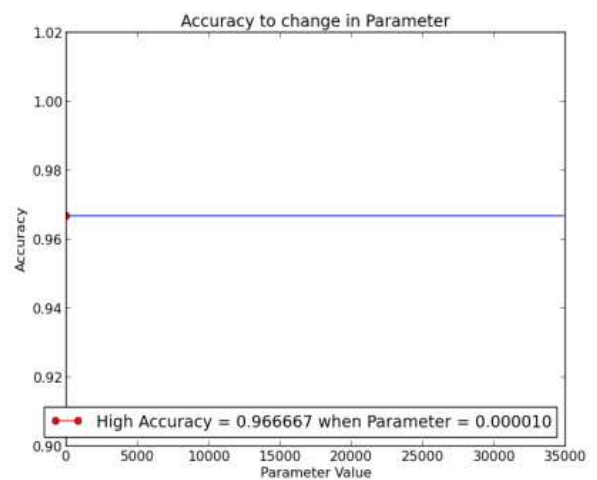


Fig -13: Tuning of Iris α parameter under sigmoid kernel

Table -10: Setting Sigmoid Kernel parameters

Data Set	Best C parameter	Best α parameter	Validation data Accuracy	Test data Accuracy	Area Under ROC Curve	Normalization
Abalone	7	0.192031	54%	51%	0.72	Yes
Balance Scale	3320	0.01	96%	89.68%	0.745	Yes
Breast Cancer	10	0.095	95.56%	97.39%	0.999	Yes
Car Evaluation	18.5	0.1	78.61%	74.56%	0.855	Yes
DNA	1	0.05	90.91%	86.36%	0.893	Yes
ECG	1.83	0.0222	97.31%	97.29%	0.979	No
EEG Eye State	N.E	N.E	55.1%	55.12%	0.586	Yes/No
Iris	1	0.0001	96.66%	90%	0.882	Yes
SPECT Heart	0.6	3	92.45%	85.18%	0.509	No
Vowel Recognition	61.5	0.0093	71.21%	61.11%	0.705	No
Wine	3	0.3	97.22%	91.89%	0.868	Yes

Notice that values stated as N.E (Not Effective) means that the performance of the classifier was not affected with the change in the parameter value.

3.4 Fourth Step (MKL Initial Values using 70% Training)

In this step MKL algorithm is trained with un-weighted summation of the stated kernels using initial values of parameters close or equal to those used in single kernel operations with training data percentage equal to 70% and testing data percentage equal to 30%. Each data set shuffled 100 times if possible, except for big sets they were shuffled only 10 times because of computational issues related to MKL made those data processing highly time consuming.

After shuffling the average accuracy was obtained to be compared to the single kernel performance later. Parameters are set to; $C = 1, d = 2, \gamma = 0.1$ and $a = 0.001$.

While data normalization or scaling is an option with the single kernel operation, it is **necessary** with MKL training processes, especially when using shogun toolbox through Ubuntu operating system 12.4 [32], because of features combining process and data chunking are highly computationally expensive.

The results out of the previous paragraph are presented in table 11.

Table -11: Training MKL with 70% of data sets and shuffling for 100 or 10 times

Data Set	Best Accuracy	Lowest Accuracy	Average Accuracy
Abalone (10 times)	0.567330677291	0.543426294821	0.553466135458
Balance Scale	0.989417989418	0.957671957672	0.973968253968
Breast Cancer	0.953488372093	0.866279069767	0.915
Car Evaluation	0.857692307692	0.792307692308	0.833326923077
DNA	0.90625	0.65625	0.77625
ECG (10 times)	0.988760044756	0.931288780389	0.97031329468
EEG Eye State (10 times)	0.753726362625	0.739043381535	0.746184649611
Iris	1.0	0.88888889	0.95111111
SPECT Heart	0.888888888889	0.740740740741	0.826790123457
Vowel Recognition	0.946127946128	0.878787878788	0.913737373737
Wine	0.890909090	0.6363636364	0.785454545

3.5 Fifth Step (MKL using 60% as Training)

In this step the same process from the previous step is repeated with one difference, which is 60% of the data are used as training set and 40% as testing set. The aim of this step is to choose proper sets for the next step. Properly chosen sets can reveal the ability of MKL classifier to

enhance the classification accuracy. In some cases Like Iris data set the set with the worst accuracy out of the second step is chosen for further processing, because the best performance out of the Iris set has 100% accuracy, which is not good for judging the ability of the classifier to enhance the accuracy. Results from second step are presented in table 12.

Table -12: Training MKL with 60% of data sets and shuffling for 100 or 10 times

Data Set	Best Accuracy	Lowest Accuracy	Average Accuracy
Abalone (10 times)	0.570574162679	0.546052631579	0.555921052632
Balance Scale	0.992063492063	0.940476190476	0.973134920635
Breast Cancer	0.951754385965	0.868421052632	0.91649122807
Car Evaluation	0.851156069364	0.799132947977	0.826719653179
DNA	0.886363636364	0.659090909091	0.770681818182
ECG (10 times)	0.990120160214	0.933778371162	0.9730650391
EEG Eye State (10 times)	0.743534123144	0.729017186718	0.737460370432
Iris	1.0	0.9	0.9505
SPECT Heart	0.88785046729	0.766355140187	0.82691588785
Vowel Recognition	0.944444444444	0.856060606061	0.894166666667
Wine	0.849315068493	0.616438356164	0.756849315068

3.6 Sixth Step (Tuning MKL)

In this step the technique of cross validation is used to find the best parameters through tuning the classifier using each parameter range. Data sets are chosen from the previous step

with new distribution in this step, which is 60% training, 20% cross validation and 20 percent for testing.

Final results' performances are measured through the accuracy and AUC from ROC.

Since un-weighted linear combination is used with MKL method, the parameters in use would be the kernels' parameters allowed to control by shogun toolbox [31] plus regularization parameter on kernels' weights to prevent them from shooting off to infinity, this parameter is C . Another parameter available through shogun toolbox is **lp-norm** as described in reference [31] and Marius Kloft, Ulf Brefeld, Soeren Sonnenburg, and Alexander Zien. Efficient and accurate lp-norm multiple kernel learning [33]. To summarize the previous part, the following parameters are used for MKL algorithm training:

- C Parameter: kernels' weights regularization parameter.
- $Norm$ Parameter: stands for lp-norm technique to enhance the performance.
- d Parameter: the polynomial kernel degree.
- γ Parameter: the RBF positive parameter to control the radius.
- a Parameter: stands for one of the Sigmoid kernel parameters.

Final results from the third step are presented in table 13, then some of the results' graphs are given.

Table -13: Final Results out of Tuning MKL algorithm

Data Set	C	γ	d	a	$Norm$	Cross Validation	Test Set	AUC
Abalone (10 times)	0.009	1.066	2	0.001	12	55.81%	53.16%	0.757
Balance Scale	0.05	0.5	8	N.E	6	93.65%	89.68%	0.823
Breast Cancer	N.E	0.01	2	N.E	2	98.23%	99.13%	0.997
Car Evaluation	10	0.5	2	0.01	2	99.13%	99.42%	0.992
DNA	N.E	0.1	N.E	N.E	N.E	90.91%	86.36%	0.942
ECG (10 times)	N.E	0.1726	2	N.E	2	98.76%	99.07%	0.998
EEG Eye State (10 times)	N.E	0.000007074	2	N.E	2	84.78%	83.72%	0.898
Iris	0.005	0.1	N.E	N.E	N.E	96.66%	100%	1.0
SPECT Heart	N.E	0.1	N.E	N.E	2	92.45%	85.18%	0.775
Vowel Recognition	15	0.1	2	N.E	2	98.48%	92.42%	0.996
Wine	1	0.4	10	N.E	N.E	100%	100%	1.0

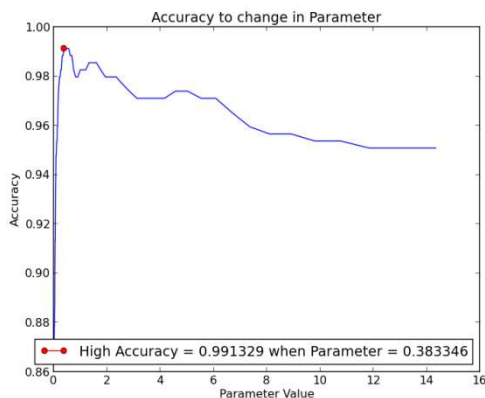


Fig -12: MKL Car evaluation γ tuning

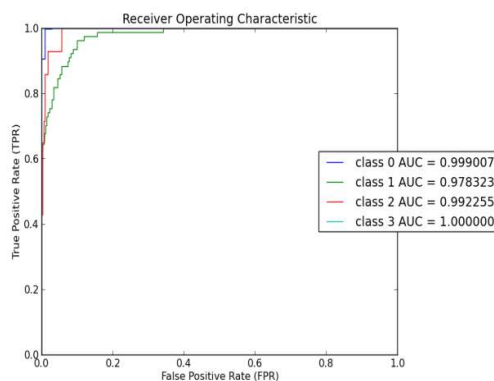


Fig -13: ROCs of classes under MKL for Car evaluation

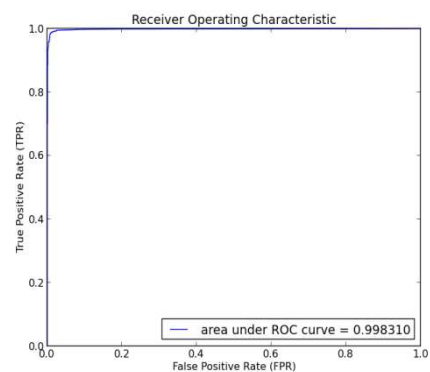


Fig -14: MKL's ROC for ECG dataset

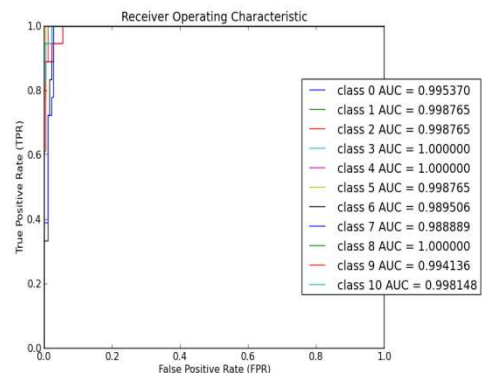


Fig -15: ROCs of classes under MKL for Vowel Recognition data set

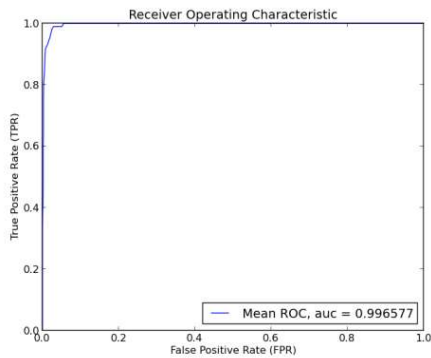


Fig -16: Vowel Recognition dataset's mean ROC under MKL

4. RESULTS COMPARISON & METHOD SELECTION

this part is aimed to present the data in an analogous form so one method or more could be selected to be used in classifying remote sensing data. To achieve this goal histograms have been made for the results obtained from all the previous steps, the following charts represent those results' histograms.

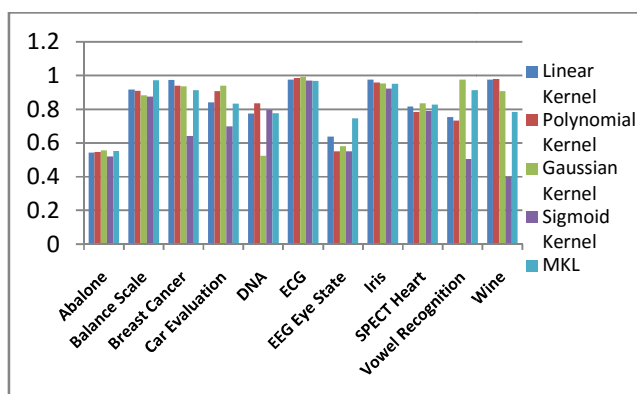


Chart -1: Accuracies using 70% as training data before setting the parameters

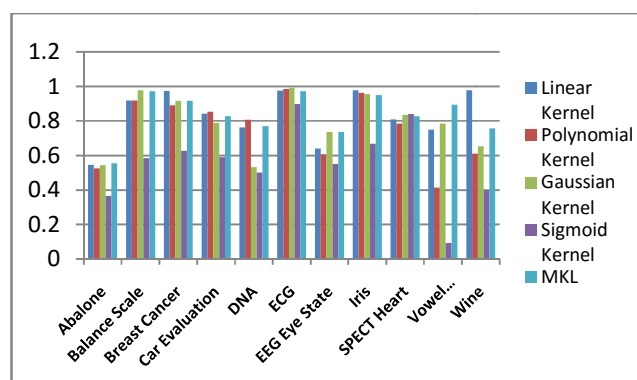


Chart -2: Accuracies using 60% as training data before setting the parameters

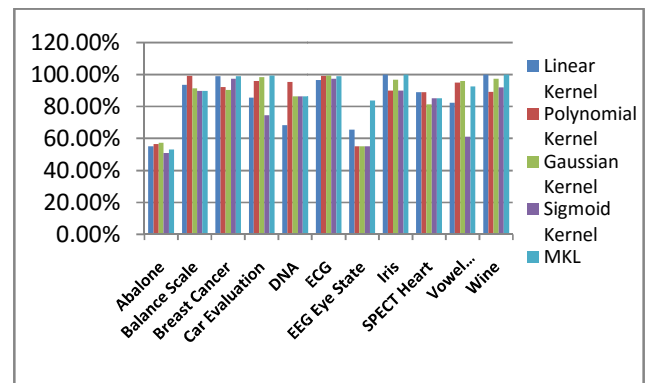


Chart -3: Final Test Accuracies after setting the parameters

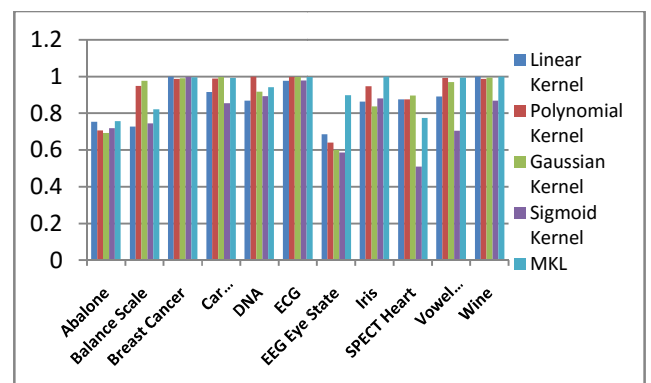


Chart -3: Final Test AUCs after setting the parameters

Theoretically MKL method supposed to achieve better results than single kernel method, but practically saying in some few cases a single kernel technique could overcome MKL and the reason for that is because some data sets have a specific nature of features, those require special treatment that may not be applicable in the case of MKL.

It is obvious from the histograms that in most cases, polynomial kernel could deliver reasonable results, especially when MKL failed to match single kernel performance, so in the next step of using MKL in remote sensing, both MKL from suggested kernels and Polynomial Kernel with normalization as single Kernel are used.

5. MKL & POLYNOMIAL KERNEL FOR REMOTE SENSING

This part shows the results of using un-weighted summation of the proposed kernels and Polynomial kernel on the data set that had been chosen for remote sensing task, this data set information and specifications were mentioned previously in this paper with all needed details.

Figures 17 and 18 present the results obtained from scale parameters in this paper and figure 19 provides an extraction of the paper in reference [22], where the paper's results during scale parameters are shown.

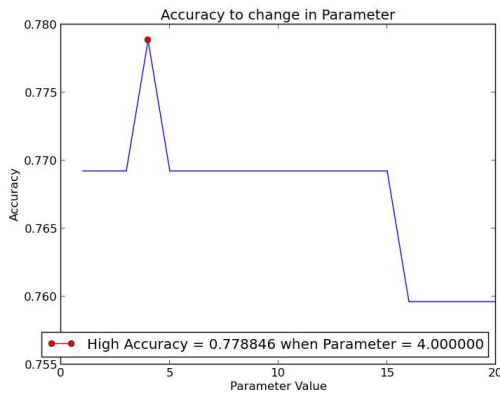


Fig -17: Cross Validation best accuracy obtained during the Scale of MKL parameters (Tuning)

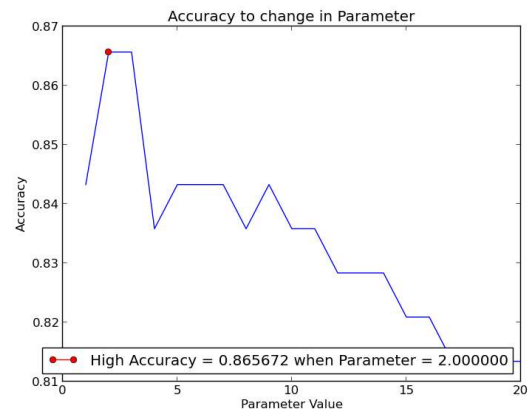


Fig -18: Cross Validation best accuracy obtained during the Scale of Polynomial parameters (Tuning)

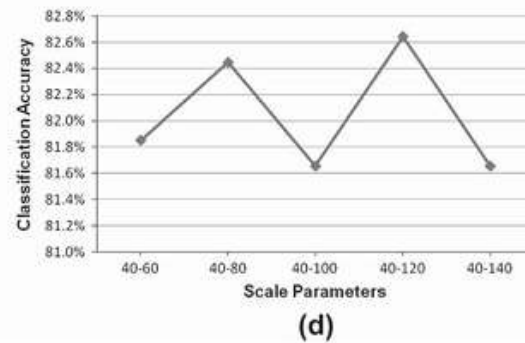
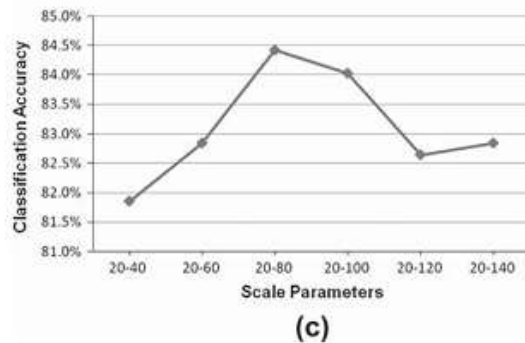
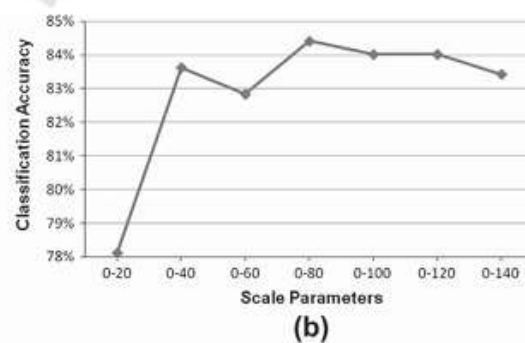
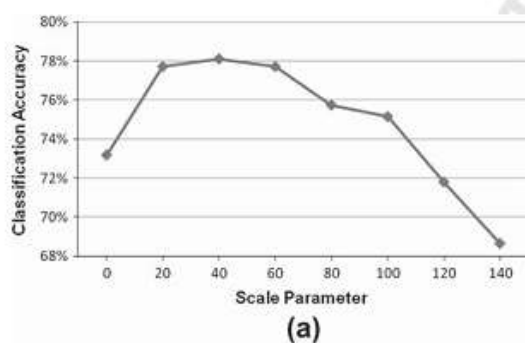


Fig 6. Overall classification accuracies for the pixel-based and single-scale classifications (a), and the multi-scale classifications with: single pixels (b), scale 20 segments (c), and scale 40 segments (d) as the base units for classification. Scale parameter of 0 indicates a pixel-based classification. For the multi-scale classifications, "Scale Parameters" indicate which segmentations were used for classification (e.g. "Scale Parameters" of 0-80 indicate that super-object variables from the scale 20, 40, 60, and 80 segmentations was assigned to single pixels for classification).

Fig -19: Best An extraction from reference [22] to show its scale of parameters (Tuning) results

From results depicted by the previous figures it is so clear that the polynomial kernel could obtain the best results. Final results of testing data are as follows; MKL accuracy 79.35% for $c = 3, d = 4, \gamma = 0.1, a = 0.01$ & $norm = 2$; polynomial kernel accuracy 84.29% for $c = 2$ & $d = 2$.

6. CONCLUSIONS

1. Classification performance or accuracy obtained through any classifier, does not depend only on the classification technique or algorithm. There are many other Factors could affect the classification, like features' transparency

that reveals the differences between their classes. So the aim that any classification technique needs to achieve is not to obtain 100% accuracy but to obtain the maximum possible accuracy out of given data. To enhance the data transparency of revealing classes other techniques beyond this research scope are used.

2. There is a difference between a theoretical expected outcome of an algorithm and the practical outcome, since those two may not match because of computational issues related to the practical application of the algorithm, like what happens in optimization or data segmentation.

3. It is found that many methods may achieve the maximum possible accuracy, so it would be a good approach to compare their computational efficiency due to time and computational resources needed to achieve the classification.
4. When dealing the four kernels used in this paper the best option is to combine them and also try polynomial kernel alone, in case of combination fail to achieve the maximum accuracy or performance.
5. Normalization technique usage does not always make a better results specially with the linear kernel, but it is still needed to reduce the computation load and speed the optimization process.
6. For beginners in machine learning or when MKL is not needed, we advice using toolboxes other than shogun for the lack of tutorials and difficulties through the setup process.

ACKNOWLEDGEMENT

The authors want to thank professor *Andrew Yan-Tak Ng* for his prolific contributions in the field of teaching machine learning courses, Python Programming language open society for their generous contributions to help others and all those contributed data sets to help others to practice and learn.

REFERENCES

- [1]. Fritzke, Bernd. Growing Cell Structures: A Self-Organizing Network for Unsupervised and Supervised Learning. Berkeley. International Computer Science Institute, 1993. 7-1441-1460.
- [2]. Hamel, Lutz. Knowledge Discovery with Support Vector Machines. Hoboken: Wiley, 2009. 10.1002/9780470503065.ch7.
- [3]. Vapnik, Vladimir N. Direct Methods in Statistical Learning Theory. The Nature of Statistical Learning Theory (2000): 225-265.
- [4]. Bishop, Christopher M, ed. Pattern Recognition and Machine Learning. New York: Springer, 2006. 978-0-387-31073-2.
- [5]. Deng, Weiping. A Study on Lane-Change Recognition Using Support Vector Machine. N.p., 2013. Print.
- [6]. Shigeo Abe. Advanced in Pattern Recognition – Support Vector Machines for Pattern Classification 2nd Edition. Springer-Verlag London Limited, 2010.978-1-84996-098-4.
- [7]. Perner, Petra, ed. Machine Learning and Data Mining in Pattern Recognition. 6th International Conference, Mldm 2009, Leipzig, Germany, July 23-25, 2009 : Proceedings. Berlin: Springer, 2009. 978-3-540-73499-4.
- [8]. A. Rakotomamonjy. Analysis of SVM regression bounds for variable ranking. Perception, Systemes et Informations CNRS FRE 2645 INSA de Rouen. Avenue de l'Universit'e . 76801 Saint Etienne du Rouvray, France
- [9]. Vapnik, Vladimir N. The Nature of Statistical Learning Theory. New York: Springer, 2000. 978-1-4419-3160-3
- [10]. John C. Platt. Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines. 1998, MIT Press.
- [11]. Kumar, M. A., and M. Gopal. An Investigation on Linear SVM and Its Variants for Text Categorization. 2010. doi:10.1109/ICMLC.2010.64.
- [12]. Lange, Kenneth. Optimization . (2004): 69-91. 978-1-4757-4182-7.
- [13]. "Radial Basis Function Kernel." Wikipedia, the Free Encyclopedia. Wikimedia Foundation, Inc, n.d.Web. https://en.wikipedia.org/wiki/Radial_basis_function_kernel.
- [14]. Fernandes, Silas E., Andre L. Pilastri, Luis A. Pereira, Rafael G. Pires, and Joao P. Papa. Learning Kernels for Support Vector Machines with Polynomial Powers of Sigmoid. 2014 27th SIBGRAPI Conference on Graphics, Patterns and Images (2014): n. pag. Print.
- [15]. Hsuan-Tien, Lin, Chih-Jen Lin. A Study on Sigmoid Kernels for SVM and the Training of non-PSD Kernels by SMO-type Methods. Department of Computer Science and Information Engineering National Taiwan University Taipei 106, Taiwan.
- [16]. Mehmet Gonen n, Ethem Alpaydın. Localized Algorithms for Multiple Kernel Learning. Pattern Recognition ,2013,46:795–807.
- [17]. Alexander Zien, Cheng Soon Ong. Multiclass Multiple Kernel Learning. 2007 the 24th international conference on Machine learning. ACM New York, NY, USA 978-1-59593-793-3.
- [18]. Sören Sonnenburg, Gunnar Rätsch. Large Scale Multiple Kernel Learning. 12/1/2006. The Journal of Machine Learning Research archive Volume 7. 1531-1565.
- [19]. Morton John Canty. Image Analysis and Pattern Recognition for Remote Sensing with Algorithms in ENVI/IDL. 2005, Forschungszentrum Julich GmbH.
- [20]. Thomas Lillesand, Ralph W. Kiefer, Jonathan Chipman. Remote Sensing and Image Interpretation. 17 Mar 2015. wiley. 978-1-118-34328-9.
- [21]. James B. Campbell, Randolph H. Wynne .Introduction to Remote Sensing. 17 Aug 2011. The Guilford Press. 978-1609181765.
- [22]. Johnson, B., Xie, Z., 2013. Classifying a high resolution image of an urban area using super-object information. ISPRS Journal of Photogrammetry and Remote Sensing, 83, 40-49.
- [23]. Lichman, M. UCI Machine Learning Repository [<http://archive.ics.uci.edu/ml>]. Irvine, CA: University of California, School of Information and Computer Science. 2013.
- [24]. "Institute for Global Environmental Strategies (IGES)." 公益財団法人地球環境戦略研究機関 (IGES : アイジェス) . N.p., n.d. Web. <http://www.iges.or.jp/en/>.
- [25]. Johnson, B., 2013. High resolution urban land cover classification using a competitive multi-scale object-based approach. Remote Sensing Letters, 4 (2), 131-140.
- [26]. Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.
- [27]. "How to Evaluate Machine Learning Models: Classification Metrics." Dato's Data Science Blog | Dato. N.p., n.d. Web. <http://blog.dato.com/how-to-evaluate-machine-learning-models-part-2a-classification-metrics>
- [28]. Metz, Charles E. Basic principles of ROC analysis. Seminars in Nuclear Medicine8.4 (1978): 283-298.

- [29]. Beazley, David M. Python Essential Reference. Indianapolis: Sams, 2006. Print.
- [30]. Raschka, Sebastian. Python Machine Learning: Unlock Deeper Insights into Machine Learning with This Vital Guide to Cutting-Edge Predictive Analytics. N.p., 2015. 1783555130.
- [31]. The SHOGUN Machine Learning Toolbox. N.p., n.d. Web. <http://www.shogun-toolbox.org/>.
- [32]. "MKL (SVM) Segmentation Fault on Ubuntu 12.04 • Issue #1720 • Shogun-toolbox/shogun • GitHub." GitHub. N.p., n.d. Web. <https://github.com/shogun-toolbox/shogun/issues/1720>.
- [33]. Marius Kloft, Ulf Brefeld, Soeren Sonnenburg, and Alexander Zien. Efficient and accurate lp-norm multiple kernel learning. In Advances in Neural Information Processing Systems. 2009. MIT Press, Cambridge, MA.

BIOGRAPHIES



Li Jun, received the Bachelor degree from Lanzhou Railway Institute, Lanzhou, China, in 1991. The master degree from Department of Telecommunications in Lanzhou Railway Institute, in 1999. In 2006 the Electrical Engineering PhD degree from Xi'an Jiaotong University, Xian, China. Right now a full time professor in Lanzhou Jiaotong University, Lanzhou, China.



Yassein Eltayeb, received the Bachelor degree in Computer Engineering from Omdurman Ahlia University, Khartoum, Sudan, in 2009. Admitted as Teaching Assistant in Omdurman Ahlia University in 2011. Currently pursuing master degree in Pattern Recognition & Intelligent Systems in Lanzhou Jiaotong University, Lanzhou China .