Combining Diverse Classifiers by Learning Weights Robust to the presence of Class Label Noise

Sannia Arshad , Shehzad Khalid

Abstract—**In this paper, we introduced a classifier ensemble approach to combine heterogeneous classifiers together in the presence of class label noise in the datasets. To enhance the performance of our proposed classifier ensemble, we give a preprocessing step to filter out class label noise present in the dataset. This noise free data is further used to learn individual classifier model. After that, a weight learning method is introduced to learn weights on each individual classifier to construct a classifier ensemble. We applied genetic algorithm to search for an optimal weight vector on which classifier ensemble is expected to give best accuracy. The proposed approach is evaluated on variety of real life datasets. The proposed technique is also compared with existing standard ensemble techniques such as Adaboost, Bagging and RSM to show the superiority of proposed ensemble method as compared to its competitors and also to show the sensitivity of competitors to class label noise. We have also given an analysis of the performance of proposed approach on the imbalanced and overlapping classes.**

Index Terms*— *Bagging, Adaboost, m-Mediods, SVM.

I. INTRODUCTION

The interest of Machine Learning research community in automatic data classification techniques is increasing from the last few decades. Until now, variety of classification techniques such as Support Vector Machine (SVM) [22], Gaussian Mixture Model (GMM) [23], Multivariate M-Mediod [24] etc. have been proposed. All of these classifiers differ in their implementation approaches as well as theories and hence they have diverse level of sophistication to handle various issues expected in real-life classification settings. These includes but not limited to patterns with overlapping distribution, multivariate distribution of samples within patterns, complex shape boundaries between samples belonging to different patterns and patterns with uneven number of training samples. Individual classifiers may be specialized in handling subsets of these problems but may give poor results in the presence of others. Resultantly, they achieve different levels of success for diverse application. To overcome this problem, classifier ensemble methods have been suggested [1-3] [6-10, 12-16]. In this method, many classifiers are combined i.e. their decisions are combined together to make a final strong classifier whose judgment is more precise and effective as compared to its individual members.

Sannia Arshad, Shehzad Khalid, Department of Computer Engineering, Bahria University Islamabad, 44000, Pakistan. Email: shehzad@bahria.edu.pk. Manuscript received March 17, 2014; revised July 19, 2014 and September 22, 2014.

These classifier ensembles can be divided into two ways according to their structure: homogeneous and heterogeneous. The homogeneous ensemble approaches combines classification algorithms of the same type whereas different types of classification algorithms are combined in heterogeneous ensemble. There exist some more classifier ensemble methods i.e. classifier selection methods [1, 2], feature selection methods [8, 9, 14], methods for diversity creation in classifiers of ensemble [16] and combination methods [1-4, 6, 12-15]. The most popular and standard ensemble methods include Bagging [12], Boosting (Adaboost) [13] and Random Subspace Methods (RSM) [14]. These are the dominant methods for diversifying and combining classification results. But all these discussed classifier ensemble methods only consider the construction of classical ensemble methods and put less attention toward handling noise problems which are obvious to occur in the real-life data sets. Although, lot of data cleaning methods have been introduced for individual classifier but not for classifier ensemble methods. A very few ensemble methods have been used to identify class label noise including boosting and voting filters [29, 30].

In this paper, we introduced a preprocessing approach to eliminate class label noise from the dataset. For this purpose, k-nearest neighbor algorithm is applied to get the specific number of nearest neighbors of sample's own class. A threshold value is defined to get the neighbors of sample's own class by specifying the ranges of data sizes. To evaluate the performance of proposed pre-processing approach, we give a method for adding artificial class label noise in training dataset. The filtered data is further used to learn individual classifier models which are then combined by introducing weight learning method to learn weights on these heterogeneous classifiers to construct an ensemble. To search for an optimal weight vector on which classifier ensemble is giving best accuracy, we applied genetic algorithm [25]. Our proposed work is related to the combination methods i.e. weight learning scheme and fall under the category of heterogeneous ensemble methods combining independent classifiers.

The remainder of the paper is organized as follows: Section II gives a brief review of existing techniques that have been used to combine classifiers in an ensemble. The classifiers used in the construction of proposed classifier ensemble are introduced in section III. The framework of proposed ensemble is detailed in section IV. Experiments are reported in Section V. The last section gives conclusion of the paper.

II. BACKGROUND AND RELATED WORK

The classifier ensemble techniques combine multiple classifiers together to produce a single stronger one. Until now, lots of classifier ensemble techniques have been introduced. Previous work includes the ensemble methods [1-4] [6-10, 12-16] that are based on different strategies. These methods are using different classifier combination [1- 4, 6, 12-15], weight assignment [3, 4, 10, 13], majority voting [2, 12], feature selection [8, 9, 14] and diversity creation methods [16]. Earlier work has put attention toward assigning weights to instances as well as classifiers to construct an ensemble by applying weight assignment methods [11] and voting algorithms [10,12-14][17-19]. The methods based on combining different classifiers together along with statistical methods have also been presented [2-3] [5-10]. Prior studies introduced the most prominent voting and weighting based standard ensemble methods i.e. Bagging [12] and Boosting [13]. Breiman [12] introduced the bagging algorithm to get better classification accuracy. Bootstrap samples are generated from training set which are obtained by sampling with replacement. Every time, a classifier is learned with different training set after which different outputs of learned classifiers are combined together to get a single classifier.

Random Forest Method proposed by Breiman [14] uses un-pruned decision trees. It consists of a collection of tree like structured classifiers and uses a number of input variables to find the decision at a node of the tree. To classify a new pattern, votes are collected from every tree in the forest and then use majority voting to finalize the class label. Schapire came up with a well-known ensemble method boosting [13] to enhance the performance of weak learner by iteratively running it on training data. Adaboost [13] improves the classifier by an iterative process. It fully concentrates on those samples which are difficult to classify. When the algorithm starts, it gives equal weights to every sample in a training dataset. After every iteration, weights of misclassified instances are increased while decreasing the weights of correctly classified ones. It further assigns weight to individual classifier to measure its overall accuracy. The higher weights are given to those classifiers performing accurately which are then used to classify new samples. Some other ensemble methods combine classifiers using majority voting [2, 12] and Dempster-Shafer theory [6].

Jiang et al [2] combined the concept of bagging with Principal Component Analysis (PCA) to construct an ensemble. Nicolas et al [7] combines the concept of boosting with supervised projection method. The focus of this technique is on misclassified instances as they are used to find supervised projection. However, this approach does not employ majority voting or weighting scheme to combine the classifiers. Some dynamic ensemble learning approaches have also been proposed in literature [3, 8]. A weight adjusted voting algorithm [10] is introduced which use a weight vector for instances as well as for classifiers. The weight vector for instance gives higher weights to those instances which are difficult to classify. On the other side, the classifiers weight vector gives highest weights to only those classifiers giving better performance on these difficult instances. The methods for dynamic ensemble selection [9] using majority voting rule to combine the classifiers have also been introduced in literature. The method in [5] combines more than one ensemble in order to achieve improved accuracy and diversity of classifiers. But all of these previous studies fully concentrated on the construction of classical ensemble methods and have less noticed the handling of different types of noise in datasets which are obvious to occur while working in real world environment. So, there is not enough literature on handling noise using ensemble methods. About standard ensemble methods, Dietterich [31] reported in his experimental work that in situations where classification noise is present in data, Bagging shows better performance as compared to Adaboost, and also with some datasets Bagging gives better accuracy than Random Forest. He reported that Random Forest is not as much sensitive to noise as Boosting. There exist few methods [29, 30] that handle class label noise in a dataset in ensemble methods. The standard ensemble schemes i.e. Adaboost, Bagging and RSM are also not performing well on imbalanced and overlapping classes.

III. CLASSIFIERS USED TO CONSTRUCT ENSEMBLE

To produce a hybrid classifier system, we selected three classifiers including Support Vector Machine (SVM) [22], Gaussian Mixture Model (GMM) [23] and our previously proposed multivariate *m*-Mediods [24]. The proposed ensemble methodology combines all three classifiers together after learning each classifier model individually. These classifiers are intentionally selected due to their ability to handle different problems and scenarios that may arise in variety of class distributions. GMM [23] is good in handling classes with overlapping and distinct distribution but does not give good results with non-overlapping classes having complex and tight boundaries. SVM [22] handles the problem of complex and tight decision boundaries by looking for best possible hyper-dimensional decision surfaces separating the samples belonging to different classes. However, the effectiveness of SVM degrades with increasing amount of overlap amongst the classes. Another disadvantage of SVM is that it neglects classes with smaller membership count to correctly classify samples belonging to classes having large membership count. Multivariate *m*-Mediod [24] classifier has the capacity to cater for the presence of multivariate distribution of samples within a modeled class. It has the strength of modeling complex patterns without imposing any limitation on the shape of distribution of samples within a given pattern. Once the multivariate *m*-Mediods model for all the classes have been learnt, the classification of test samples is achieved using a soft classification technique that can handle for multimodal and overlapping distributions of samples among different patterns within a dataset. It also caters for patterns having small membership count. Combining classifiers with different skills and specialties will enable the ensemble to cover all possible aspects and problems of classification.

IV. PROPOSED CLASSIFIER ENSEMBLE SYSTEM

Our proposed ensemble approach is presented to combine classifiers in a hybrid framework to enhance the classification accuracies as compared to individual classifiers. In proposed method, model of individual classifiers are generated. These models are used to generate individual opinions of different classifiers regarding test sample. As classifier ensemble is effective only in situations where there is a difference in opinion regarding classification of test sample, we propose to select only those samples (referred to as confused samples) to be used for ensemble learning purpose.

Let $DT = (XT, Y)$ be a labeled dataset where $XT = \{x_1, x_2, x_3, ..., x_n\}$ a dataset is containing *n* samples and $Y = \{y_1, y_2, ..., y_n\}$ is a set containing the label information of corresponding instance in set *XT*. A sample x_i in a data set *DT* is represented by a *t*-dimensional feature vector $x_i = \{f_1, f_2, ..., f_t\}$. Let $XT^{(j)}$ represent the samples belonging to class *j*. From each class *j*, 70% of samples are selected as train set. The rest of 30% samples from each class are treated as test set. Let XT_{train} represents the train set and XT_{test} denotes the test set. The process of filtering class label noise is comprised of following steps:

1) Select a sample *x* from train data XT_{train}. Compute its distance from all the samples in XT_{train} and sort these computed distances in descending order. After that, identify *k* distances to get the labels of *k* nearest neighbors of current sample as:

$$KN\_N(x, TR, k) = \left\{ P \in TR \left| \begin{array}{l} \forall U \in P \wedge \\ \forall V \in TR - P, \\ \|x, U\| \le \\ \|x, V\| \wedge |P| \\ = k \end{array} \right. \right\} \quad (1)$$

where x is the current training sample. TR is a set of all training instances and P represents the set of k instances. The value of k finds out the number of nearest neighbors whose corresponding training instances are nearest to x. ||.,.|| is the function of Euclidean distance. U represents the each individual instance in set P. Similarly, V in set $TR - P$. Set P of k closest neighbors is updated in every iteration for each new sample x.

2) After getting *k* nearest neighbors of sample *x*, define the threshold value for achieving the specific number of nearest neighbors of sample's own class. The threshold value can be different for different data sets.
3) Count the number of instances from *KNN* set '*P'* that belongs to the same class as that of query sample *x*.
4) Initialize $X\acute{T}_{train} = \{\}$ as an empty train set.
5) If the counted number of instances achieves the defined threshold value, then maintain the sample *x* in $X\acute{T}_{train}$, otherwise not. Let â be the number of instances from *KNN* set '*N'* that belongs to the same class as that of query sample *x*. Update $X\acute{T}_{train}$ as

$$X\breve{T}_{train} = \begin{cases} X\acute{T}_{train} \, \mathrm{Y} \, x_i & if \quad \breve{a} \ge \tau, \\ X\breve{T}_{train} & otherwise \end{cases} \quad \forall x_i \in XT_{train} \quad (2)$$

Where τ is the defined threshold value. *i* is the number of sample in $X\acute{T}_{train}$.

6) Repeat steps 1-5 for all samples in XT_{train}.

After getting filtered train data, the methodology for learning individual classifiers and selection of confused samples consists of the following steps:

i. Set $XT_{confused} = \{\}$

ii. Divide dataset $X\acute{T}_{train}$ into *k*-folds. Let ₣$_k$ represent the *k*th fold of $X\acute{T}_{train}$, where $k = 1,2,...,\#cv$ and $\#cv$ is the total number of folds. Each fold F_k contains equal number of instances from all the classes in the dataset.

iii. Select *k*th fold and treat it as cross-validation set $XT_{cv} = F_k$. The remaining $\#cv - 1$ folds are selected to learn the classifier model as:

$$XT_{train} = \{XT_{train} \cup F_j \forall j = 1,2,..\#cv \wedge j \ne k\} \quad (3)$$

iv. Learn the model of GMM, SVM and multivariate m-Mediod classifiers using XT_{train}.

v. Select ith sample from XT_{train}, represented as $XT_{cv}^{(i)}$ and classify it separately using GMM, SVM and multivariate *m*-Mediod classifiers as:

$$class^{cl}_{X_{cv}^{(i)}} = \arg \max_{\forall class_j} P_{cl}\left(y = class_j \middle| X_{cv}^{(i)}\right) \quad (4)$$

where $P_{cl}\left(y = class_j \middle| X_{cv}^{(i)}\right)$ is the probability $class_j$ of given $X_{cv}^{(i)}$ using classifier C_l.

vi. Identify those samples for which at least two classifiers give different class predictions. This is achieved by filtering those samples from XT_{cv} for which all the concerned classifiers predict the same class as these will not contribute in weight learning process. Removing such samples will have a positive impact of significantly speeding up the weight learning process. The filtered cross validation set which is composed of confused samples, referred to as $XT_{confused}$, is obtained as:

$$XT_{confused} = XT_{confused} \cup \left\{ \begin{matrix} XT_{cv}^{(i)} \in XT_{cv} \mid class_{XT_{cv}^{(i)}}^{Cl} \neq \\ class_{XT_{cv}^{(i)}}^{Cp} \forall l, p \wedge l \neq p \end{matrix} \right\} \quad (5)$$

7) Repeat step 3-6 for all folds in the dataset.

Once we have generated the model of different classifiers and extracted confused samples set $XT_{confused}$, we use them to learn weights for each individual classifier by applying genetic algorithm based learning approach. Let W=[wGMM wSVM wm-Mediods] represents a weight vector containing weights to scale the decision of different classifiers. The algorithm for learning optimal weight values to maximize ensemble accuracy using $XT_{confused}$ comprises the following steps:

i. Initialize the population by randomly generating 16 weight vectors. Normalize the weight vectors so that the sum of weights in each vector is equivalent to 1.

ii. Pad the population with 4 pre-defined weight vectors including [1 0 0], [0 1 0],[0 0 1] and [1/3 1/3 1/3]. [1 0 0], [0 1 0] and [0 0 1] weight vectors are padded to give maximum confidence to a single classifier. This will be helpful in class distributions which ideally suits the expertise of one of the classifier. [1/3 1/3 1/3] is padded to give equal confidence to all the classifiers. Resultantly, we have population comprising of 20 weight vectors. We represent the set of weight vectors as **W**.

iii. Set $acc_w = 0 \forall W \in \mathbf{W}$.

iv. Select a sample $XT_{confused}^{(i)}$ from confused sample and compute its probability to be classified to different classes using all the classifiers as:

$$\mathrm{Pr}ob_{Cl}^{j} = P_{Cl}\left(y = class_j \middle| XT_{confused}^{(i)}\right) \quad (6)$$

Where $\mathrm{Pr}ob_{Cl}^{j}$ is the probability of a given sample to be classified to class *j* according to classifier C_l.

v. Combine different classifiers using weight vector *W* from population set **W** as:

$$\begin{aligned} \mathrm{Pr}ob_{ens}^{j}(W) &= \omega_{GMM} \times \mathrm{Pr}ob_{GMM}\left(y = class_j \middle| XT_{confused}^{(i)}\right) \\ &+ \omega_{SVM} \times \mathrm{Pr}ob_{SVM}\left(y = class_j \middle| XT_{confused}^{(i)}\right) \\ &+ \omega_{m-Mediods} \times \mathrm{Pr}ob_{m-Mediods}\left(y = class_j \middle| XT_{confused}^{(i)}\right) \forall j \end{aligned} \quad (7)$$

where $\mathrm{Pr}ob_{ens}^{j}(W)$ is the probability of sample $XT_{confused}^{(i)}$ to belong to class *j* according to classifier ensemble created using weight vector W.

vi. Classify $XT_{confused}^{(i)}$ using:

$$class_{XT_{cv}^{(i)}}^{ens} = \arg\max_{\forall class_j} \mathrm{Pr}ob_{ens}^{j}(W) \quad (8)$$

vii. Increment acc_w by 1 if the predicted class of $XT_{confused}^{(i)}$ is equivalent to its true label.

viii. Repeat steps 4-5 for $\forall W \in \mathbf{W}$.

ix. Repeat steps 4-8 for all samples in $XT_{confused}$. The classification accuracies of ensemble computed using different weight vectors in **W** is the objective function that we like to optimize using genetic algorithm.

x. Generate new sets of weight vector $\mathbf{W}_{new}$ by selecting 10 best weight vectors from **W** w.r.t. their objective function values and applying the genetic operators i.e. crossover and mutation. The objective function using $\mathbf{W}_{new}$ is computed as specified in steps 3-9. The evolved population of best 20 weight vectors is obtained by selecting the top weight vectors from $\mathbf{W}_{new} \cup \mathbf{W}$. This step prevents us to lose track of any weight vector that gives us the optimal results during the weight learning procedure.

xi. Repeat steps 3-10 till there is no improvement in the classification accuracy for 5 consecutive iterations or the number of iterations over the genetic algorithm exceeds 50.

Once the weight vector to combine classifiers in an ensemble is learned, the classification of test sample *x* using proposed classifier ensemble is performed using eq. (9).

V. EXPERIMENTAL STUDIES

A. *Experimental Datasets*

To evaluate the performance of proposed classifier ensemble methodology, four different real life datasets have been used including Iris, Satimage, DiaretDB and Heart datasets. A Brief overview of these datasets is given in TABLE I. The distribution of instances in different classes within a dataset is presented in TABLE II to highlight the variation in distribution of samples among the classes.

B. *Experiment 1: Evaluation of Proposed Classifier Ensemble Approach*

The experiment is conducted to evaluate the classification accuracy and effectiveness of proposed ensemble approach using DiaretDB dataset. Training data is achieved by randomly getting 70% of the samples separately from each class. The remaining instances are treated as test data. We employ k-fold cross validation with *k=7* to learn the ensemble as specified in section IV. We managed to extract only 159 confused samples out of 2156 samples present in training Diaretdb data, which resultantly speeds up the weight learning process. Learning of optimal weights for the proposed ensemble is done as specified in section IV. The classification accuracies of classifier ensemble approach and its individual classifier members using test DiarestDB dataset are presented in TABLE III. It is observed that hybrid classifier yields best accuracy i.e. 99.02% as compared to its individual member classifiers which shows the effectiveness of grouping classifiers together as compared to applying them individually.

C. *Experiment 2: Comparison of Proposed Classifier Ensemble Approach with Competitors*

This experiment evaluates the proposed ensemble approach on different real-life datasets and compares it with competitors including Adaboost, Bagging and Random Subspace Method. The experiment is conducted on DiaretDB, Iris, Satellite Image and Heart datasets. The learning of competitor ensembles is performed for 50, 100 and 150 iterations. The accuracy results of competitors at these three different iterations are given in TABLE IV. It is observed that their performance is affected by the number of iterations, especially of Adaboost which shows more sensitivity toward iterations with Iris, DiaretDB and Satimage datasets. Bagging and Random Subspace are also affected by the number of iterations but to a relatively smaller level. We selected the best accuracy results of these competitors at different iterations which are highlighted in TABLE IV and presented them in TABLE V. We also present the results obtained using our proposed classifier ensemble in TABLE V. The comparison of classification accuracies of proposed classifier ensemble approach with its competitors shows that proposed approach gives best results as compared to competitors. The proposed approach is not affected by the number of iterations used for learning the ensemble

$$class(x) = \arg\max_{\forall class_j} \omega_{GMM} \times \Pr ob_{GMM}(y = class_j)|x + \omega_{SVM} \times \Pr ob_{SVM}(y = class_j)|x \\ + \omega_{m-Mediods} \times \Pr ob_{m-Mediods}(y = class_j)|x \forall j \tag{9}$$

TABLE I. A BRIEF OVERVIEW OF EXPERIMENTAL DATASETS

Dataset	Dataset Description	# of instances	# of features	# of Classes
Iris	A flower dataset [27] that is publically available dataset, used as a test case for classification techniques.	150	4	3
DiaretDB	An eye's retina dataset [28] used for diabetic retinopathy detection methods.	3080	15	4
Satimage	A publically available satellite images dataset [27] generated from Landsat scanner image data.	6435	36	6
Heart	A good dataset [27] to test the ML algorithms.	267	23	2

TABLE II. DESCRIPTION OF NUMBER OF INSTANCES PRESENT IN EACH CLASS OF DATASET

Dataset	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6
Iris	50	50	50			
Diaretdb	1028	725	1262	65		
Satimage	1533	703	1358	626	707	1508
Heart	55	212				

TABLE III: CLASSIFICATION ACCURACY OF PROPOSED APPROACH AND INDIVIDUAL CLASSIFIER WITH DIARETDB.

Method	Accuracy (%)
Proposed Hybrid Approach	**99.02**
M-Mediod	97.94
GMM	98.59
SVM	93.72

D. *Experiment 3: Comparing Proposed Ensemble Approach with Competitors in the presence of Noise*

In order to study the sensitivity of proposed approach and its competitors in the presence of class label noise, we have induced the artificial noise to class labels of 70% train dataset. To add *L%* class label noise to 70% train dataset *TD*, pick $L \times TD/_{100}$ samples randomly and alter their true class labels with wrong ones at random. The noise is only induced in 70% train dataset and not in test set.To make our preprocessing approach generic we define the ranges of k-nearest neighbors and the threshold value τ for different sizes of datasets as given in TABLE VI. The description of how to achieve threshold value is specified in section IV.

TABLE IV. CLASSIFICATION ACCURACY OF COMPETITORS AT DIFFERENT NUMBER OF ITERATIONS.

	Adaboost			**Bagging**			**RSM**		
	Iter = 50	***Iter= 100***	***Iter= 150***	***Iter = 50***	***Iter= 100***	***Iter= 150***	***Iter= 50***	***Iter= 100***	***Iter= 150***
Iris	**100**	97.8	97.8	**100**	97.8	**100**	**100**	**100**	**100**
Diaretdb	91.92	**92.35**	**92.35**	97.68	**97.9**	97.68	**97.24**	97.1	97
Satimage	76.1	77.1	**78.1**	91.25	91.5	**91.7**	**90.99**	90	90
Heart	67.5	**67.5**	66.3	62.5	62.5	**63.8**	**68.8**	**68.8**	**68.8**

TABLE V. COMPARISON OF PROPOSED APPROACH WITH COMPETITORS

	Adaboost	**Bagging**	**RSM**	**Proposed Approach**
Iris	**100**	**100**	**100**	**100**
Diaretdb	92.35	97.9	97.24	**99.02**
Satimage	78.1	91.7	90.99	**93.38**
Heart	67.5	63.8	68.8	**69.47**

TABLE VI. SPECIFIED RANGES FOR K-NEAREST NEIGHBORS & THRESHOLD τ

Range of data size	**Range of k**	**Range of τ**
0 – 300	5 – 7	2 – 3
0 – 4000	9 – 11	3 – 4
0 – 7000	11 – 13	4 – 6

The experiment is conducted to illustrate the effectiveness of proposed ensemble approach to class label noise. The comparison of proposed approach with competitors using Diaretdb, Iris and Satimage datasets with class label noise is also presented. The sensitivity of proposed methodology and competitors to 5% and 10% level of class label noise are presented in TABLE VII. It is observed that, the performance of Adaboost is affected by the 5% and 10% level of class label noise very much with Satimage data. With Diaretdb, the performance of Adaboost goes down at 10% level of noise. The reported results show the sensitivity of Adaboost to class label noise. From TABLE VII, we can observe that Bagging is less sensitive to noise as compared to Adaboost with DiaretDB and Satimage datasets. RSM shows more sensitivity to 5% level of noise with DiaretDB dataset. From TABLE VII, it is observed that our proposed ensemble approach shows less sensitivity toward class label noise as compared to its competitors. By applying our preprocessing approach, the proposed ensemble approach does not remain much sensitive to noise and give more accurate results as compared to competitors.

E. Sensitivity of ensembling approaches to class imbalance in datasets

The performance of proposed hybrid approach is also measured for class imbalance problem in datasets. The competitors are also evaluated and compared with proposed method. The confusion matrices of competitors and proposed approach with different datasets are given in TABLE VIII, IX and X. Adaboost, Bagging and RSM are evaluated on DiaretDB, Satimage and Heart dataset. It is observed form Table VIII (a) that, Adaboost is performing poorly on class imbalance problem such as on class 4 of Diaretdb which has very small number of instances in it. Due to this, the *false positive* rate is increasing. There is also ambiguity between class 3 and 4 of DiaretDB. Adaboost has shown poor performance on ambiguous classes in Diaretdb dataset by:

TABLE VII. COMPARISON OF PROPOSED APPROACH WITH COMPETITORS AT 5% & 10% NOISE

	AB.M1/M2		**Bagging**		**RSM**		**Proposed Approach**	
	5% noise	***10% noise***	***5% noise***	***10% noise***	***5% noise***	***10% noise***	***5% noise***	***10% noise***
Iris	**98**	**97.8**	95.6	95.6	97.6	97.6	**98**	**97.8**
Diaretdb	90.95	88.6	97.2	96.1	95.9	94.1	**98.2**	**97.2**
Satimage	75.5	72.2	91.2	90.8	90.5	90.3	**92.87**	**91.73**

Classifying class 4 into class 3. Similarly in TABLE VIII (b) & (c) we have seen that, Bagging and RSM are also performing badly on minority and ambiguous class but their performance is a little bit better than Adaboost. The performance of Adaboost is also poor on Satimage dataset for class imbalance and ambiguity among classes 4 and 6 as shown in TABLE IX (a). From Table IX (b) & (c) we can see that, Bagging and RSM are performing well to some extent with Satimage for both issues but still confused to classify class 4. The same is the behavior of Adaboost, Bagging and RSM with Heart dataset which can be observed in TABLE X (a), (b) & (c). The reason behind this behavior of competitors is that, the focus of these techniques is to improve their overall classification accuracy. They learn their models on majority class very well but model the minority class poorly due to insufficient data.

As compared to competitors, our ensemble approach is performing well on DiaretDB, Satimage and Heart datasets as shown in Table VIII (d), IX (d) & X (d). It is observed that, by combining individual classifiers together having different capabilities we can handle the issue of imbalance and overlapping classes very well as *m*-Mediod and GMM can perform well on imbalance instances. GMM can handle the overlapping classes very well. *m*-Mediod can handle the multivariate distribution of samples within the class. It can also handle the overlapping classes as well as classes having small number of instances very well. By doing so, the *false positive* and *false negative* rates can be decreased.

VI. CONCLUSION

This paper presents a preprocessing approach to filter out class label noise from the dataset. K-nearest neighbor algorithm is applied to get the specific number of nearest neighbors of sample's own class. For this, we define a threshold value to get the neighbors of sample's own class by specifying the ranges of data sizes. Filtered data is then used to learn individual classifier models. These individual models are combined by introducing weight learning method to learn weights on these heterogeneous classifiers to construct an ensemble. To search for an optimal weight vector on which classifier ensemble is giving best accuracy, we applied genetic algorithm. From experimental studies, it is observed that our proposed approach gives superior results as compared to the competitors such as AdaBoost, Bagging and Random Subspace Method. It also gives best retrieval accuracies for all real life datasets selected from UCI Repository. The proposed approach is not much sensitive to the class label noise. Our proposed approach has shown less sensitivity toward overlapping and imbalance classes as compared to its competitors.

REFERENCES

[1] P,Garcia.,N,Garcia-Osorio.,C, Fyfe. "Nonlinear boosting projections for ensemble construction." Journal of Machine Learning Research, vol. 8. pp 1–33, 2007

[2] Z, Chun-Xia., Z, Jiang-She. "A novel method for constructing ensemble classifiers". Statistics and Computing, vol. 19. pp 317–327. 2009.

[3] Z, Yuquan,. O, JiShun,. C ,Geng,. Y, Haiping. "An Approach for Dynamic Weighting Ensemble Classifiers Based on Cross-validation." Journal of Computational Information Systems, vol. 6 no. 1. pp 297-305. 2010.

[4] N, Toh Koon,. V, Dan. "A direct boosting algorithm for the k-nearest neighbor classifier via local warping of the distance metric". Journal of Pattern Recognition Letters, vol. 33. pp. 92–102. 2012.

[5] K, Sotiris. "Combining bagging, boosting, rotation forest and random subspace methods". Journal of Artificial Intelligence Research, vol. 35. pp. 223–240. 2011.

[6] A, Ahmad., D, Mohamed. "A New Technique for Combining Multiple Classifiers using the Dempster-Shafer Theory of Evidence". Journal of Artificial Intelligence Research, vol. 17. pp. 333-361. 2002.

[7] P, Nicolas., G, Cesar. "Constructing ensembles of classifiers using supervised projection methods based on misclassified instances". Experts Systems with Applications, vol. 38, no. 1 2011.

[8] Z, Zhang,. P,Yang. "An ensemble of classifiers with genetic algorithm based feature selection". IEEE Intelligent Informatics Bulletin, vol. 9, no. 1, pp. 18-24. 2008.

[9] H R, Albert., S, Robart., B, Alceu. "From dynamic classifier selection to dynamic ensemble selection". Pattern Recognition, vol. 41. pp 1718 – 1731. 2008.

[10] K, Hyunjoong,. K, Hyeuk., M, Hojin., A, hongshik. "A Weight-Adjusted Voting Algorithm for Ensemble of Classifiers". Journal of Korean Statistical Society, vol. 40, no. 4, pp 437-449. 2011.

[11] L, Rokach. "Ensemble-Based Classifiers". Artificial Intelligence Review, vol. 33. pp 1–39. 2010.

[12] Breiman L. "Bagging predictors". Journal of Machine Learning, vol. 24, no. 2, pp 123–140. 1996.

[13] Freund Y, Schapire RE. "Experiments with a new boosting algorithm". Proceedings of the 13th International Conference on Machine Learning. pp 325–332. 1996.

[14] Breiman L. "Random Forests". Machine Learning, vol. 45, pp 5–32. 2001.

[15] Xu, L., Krzyzak, A., Suen, C. "Methods of combining multiple classifiers and their applications to handwriting recognition". IEEE Transactions on Systems, Man and Cybernetics, vol. 22, pp 418–435. 1992.

[16] Kuncheva, L. "Diversity in multiple classifier systems. Information Fusion, vol. 6, pp 3-4. 2005.

[17] Bauer E, Kohavi R. "An empirical comparison of voting classification algorithms: bagging, boosting, and variants". Journal of Machine Learning, vol. 35, pp 1–38. 1999.

[18] Merler S, Caprile B, Furlanello C. "Parallelizing AdaBoost by weights dynamics". Journal of Computational Statistics & Data Analysis, vol. 51, pp 2487–2498. 2007.

[19] Zhang CX, Zhang JS. "A local boosting algorithm for solving classification problems". Journal of Computational Statistics Data Analysis, vol. 52, no. 4. pp 1928–1941. 2008.

[20] L, Rokach. "Taxonomy for Characterizing Ensemble Methods in Classification Tasks: A review and annotated bibliography". Journal of Computational Statistics & Data Analysis, vol. 53, no. 12, pp 4046-4072. 2009.

[21] P Cunningham, J D Sarah. "k-Nearest Neighbour Classifiers". Technical Report UCD-CSI-2007-4. March 27, 2007.

[22] M . D Hanif, P.S Yashwant. "Cybercrime detection techniques based on support vector machines". Artificial Intelligence Research, vol. 2, no. 1. 2013.

[23] S Bertrand. "Gaussian Mixture Model Classifiers". 2007

[24] K. Shehzad, R. Shahid. "Frameworks for multivariate m-mediods based modeling and classification in Euclidean and general feature spaces". Journal of Pattern Recognition, vol 45, pp 1092–1103. 2012.

[25] J Shapiro. "Genetic Algorithm in Machine Learning. Machine learning and its Application". Lecture Notes in Computer Science, vol. 2049, pp 146-168. 2001.

[26] S Arlot. "A survey of cross-validation procedures for model selection". Statistics Surveys, vol. 4, pp 40-79. 2010.

[27] UCI Machine Learning Data Repository. http://archive.ics.uci.edu/ml/datasets

[28] K Tomi et al. DIARETDB0: Evaluation Database and Methodology for Diabetic RetinopathyAlgorithms.

[29] V. Sofie, V A. “Anneleen. Ensemble methods for noise elimination in classification problems”. Proceeding of the 4th international conference on multiple classifier systems.

[30] M Prem, S Nishit, M Lilyana, R J. “Mooney. Experiments on Ensembles with Missing and Noisy Data”. Proceedings of 5th International Workshop on Multiple Classifier Systems. vol. 3077, pp 293-302, Cagliari, Italy.

[31] T. G. Dietterich. “An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization”. Machine Learning, vol. 40, pp 139–157. 1999.

TABLE VIII. EVALUATION MATRICES OF (a) ADABOOST (b) BAGGING (c) RSM & (d) PROPOSED APPROACH ON DIARETDB.

		Predicted Class			
		Clas1	Class 2	Class 3	Class 4
Actual Class	Class 1	284	14	10	0
	Class 2	13	203	2	0
	Class 3	2	10	367	0
	Class 4	0	2	17	0

(a)

		Predicted Class			
		Clas1	Class2	Class 3	Class 4
Actual Class	Class 1	307	0	1	0
	Class 2	2	215	1	0
	Class 3	2	3	374	0
	Class 4	0	2	8	9

(b)

		Predicted Class			
		Class1	Class2	Class3	Class4
Actual Class	Class 1	305	0	3	0
	Class 2	5	212	1	0
	Class 3	1	0	378	0
	Class 4	0	2	13	4

(c)

		Predicted Class			
		Class1	Class2	Class3	Class4
Actual Class	Class1	300	5	3	0
	Class2	0	271	1	0
	Class3	0	0	379	0
	Class4	0	0	0	19

(d)

TABLE IX. EVALUATION MATRICES OF (a) ADABOOST (b) BAGGING (c) RSM (d) PROPOSED APPROACH ON SATIMAGE.

		Predicted Class					
		Class1	Class2	Class3	Class4	Class5	Class6
Actual Class	Class1	363	0	22	2	4	69
	Class2	13	177	0	2	13	6
	Class3	2	0	394	5	0	6
	Class4	4	0	60	14	0	110
	Class5	30	2	1	3	137	39
	Class6	1	0	17	11	2	421

(a)

		Predicted Class					
		Class1	Class2	Class3	Class4	Class5	Class6
Actual Class	Class1	446	1	9	0	4	0
	Class2	0	206	0	3	1	1
	Class3	2	2	391	10	0	2
	Class4	2	1	34	120	1	30
	Class5	9	2	0	0	188	13
	Class6	0	2	6	21	5	418

(b)

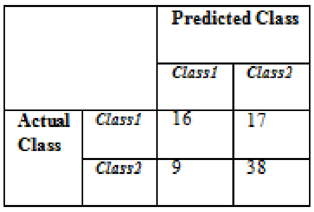
		Predicted Class					
		Class1	Class2	Class3	Class4	Class5	Class6
Actual Class	Class1	450	1	9	0	0	0
	Class2	0	205	0	2	3	1
	Class3	1	3	390	8	0	5
	Class4	1	1	34	120	1	31
	Class5	13	2	0	1	177	19
	Class6	0	1	7	27	4	413

(c)

		Predicted Class					
		Class1	Class2	Class3	Class4	Class5	Class6
Actual Class	Class1	437	7	12	2	0	2
	Class2	1	205	0	3	2	0
	Class3	4	0	387	6	7	2
	Class4	3	4	11	149	16	5
	Class5	1	0	1	6	202	2
	Class6	0	5	3	11	23	410

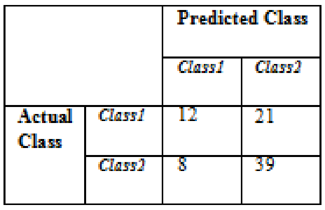
(d)

TABLE X. EVALUATION MATRICES OF (a) ADABOOST (b) BAGGING (c) RSM (d) PROPOSED APPROACH ON HEART.



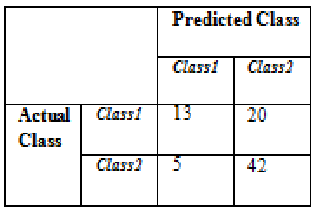
		Predicted Class	
		Class1	*Class2*
Actual Class	*Class1*	16	17
	Class2	9	38

(a)



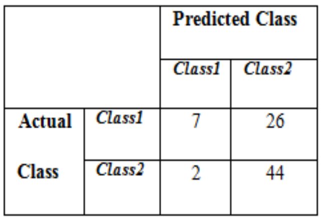
		Predicted Class	
		Class1	*Class2*
Actual Class	*Class1*	12	21
	Class2	8	39

(b)



		Predicted Class	
		Class1	*Class2*
Actual Class	*Class1*	13	20
	Class2	5	42

(c)



		Predicted Class	
		Class1	*Class2*
Actual Class	*Class1*	7	26
	Class2	2	44

(d)

ISSN – 1999-4974