LETTER

Reducing the Number of Training Samples for Fast Support Vector Machine Classification

Ravindra Koggalage and Saman Halgamuge
Department of Mechanical and Manufacturing Engineering, The University of Melbourne
Parkville, Victoria 3010, Australia
E-mail: rlwk@mame.mu.oz.au, sam@mame.mu.oz.au

(Submitted on February 16, 2004)

Abstract: Support Vector Machines (SVMs) have gained wide acceptance because of the high generalization ability for a wide range of classification applications. Although SVMs have shown potential and promising performance in classification, they have been limited by speed particularly when the training data set is large. The hyper plane constructed by SVM is dependent on only a portion of the training samples called support vectors that lie close to the decision boundary (hyper plane). Thus, removing any training samples that are not relevant to support vectors might have no effect on building the proper decision function. We propose the use of clustering techniques such as K-mean to find initial clusters that are further altered to identify non-relevant samples in deciding the decision boundary for SVM. This will help to reduce the number of training samples for SVM without degrading the classification result.

Keywords: Support Vector Machines, K-mean, Clustering, Classification

1. Introduction

1.1 SVM classification

Support vector machines (SVMs) developed by Vapnik[1], have gained wide acceptance because of their high generalization ability for a wide range of applications. There are many variations of SVM, including the soft margin classifier, adaptive margin classifier, and so on, depending on the margin used. Even though the two classes divided by the margin are slightly overlapped and noise exists, they all have the common property that the constructed hyper plane effectively separates two classes [2].

1.2 Problems with huge training sets in SVM

Recently SVMs have shown promising performance in many applications. However, they require the use of an iterative process such as quadratic programming to identify the support vectors from the labeled training set of samples. When the number of samples in the training set is huge, sometimes it is impossible to use all of them for training; otherwise heuristic methods have to be used to speed up the process.

One such heuristic approach is to use chunks of samples. A chunk is a pre-defined small number of samples, which is much less than the total number of samples in the whole training set. Each chunk of samples is used to iterate for support vectors, and the samples, which are support vectors are kept for further training[3] . Samples, which cannot be support vectors, are simply discarded. The process continues with different chunks until all training samples are being used. Although this is a feasible way of avoiding memory capacity and cost problems, it is still very time consuming for huge data sets.

Other possible approaches include the use of heuristic techniques to reduce the number of samples to be used for training. This could be considered as an effort to select an appropriate subset of available training data for the purpose of SVM training. One common way of doing this is to select a representative subset randomly.

However, in the context of SVM, there is no guarantee that this method would not remove possible support vectors that would otherwise directly affect the construction of decision hyper plane.

1.3 Proposed solution

Although SVMs have shown attractive potential and promising performance in classification, they have the limitation of speed and size in training large data sets. The hyper plane constructed by SVM is dependent on only a fraction of training samples called support vectors that lie close to the decision boundary (hyper plane). Thus, removing any training samples that are not relevant to support vectors may have no effect on building the proper decision function [4]. If it is possible to identify such non-relevant samples from the training set, it is possible to reduce the computational cost, and in turn allow us to use complex kernels that can increase the accuracy of the classification results. The problem to be solved here is how the non-relevant samples in the training data set can be identified. In [4], Sunghwan has proposed a method to use fuzzy class membership for each sample in the training set, where K-nearest neighbors are used to calculate the class membership. Since this has to be done for each and every sample of the training set, it requires huge computational overhead for large training sets. In this paper we propose an efficient way of identifying non-relevant samples, and hence reduce the size of the training set for SVM, without degrading the results.

In this method, any simple clustering technique can be applied to the whole training set to identify initial clusters. This can be a supervised or unsupervised clustering. Once the initial clusters are detected, a variable radius from the cluster center is used to identify 'crisp clusters'. Crisp clusters are the clusters with the same class samples. Crisp clusters are further refined to remove possible support vectors, and then the samples in refined crisp clusters are removed from the training set.

2. Issues to be Considered When Reducing Number of Samples

2.1 Effects for classification result

There can be advantages in reducing the number of training samples before trying to select support vectors, in terms of speed and space requirements. However, reducing samples from the training set should not affect the classification results. If it can improve the classification results when compared with using the whole training set, it would be an added benefit. On the other hand, if the classification rate is decreasing, it must be within an acceptable range, depending on the application requirements.

2.2 Keep support vectors

In SVMs, support vectors are the outcome of the training process. A separable, optimal hyper plane is decided based on the support vectors. It is important to keep the samples that could possibly become support vectors, in order to maintain the same accuracy. Generally, support vectors are the samples close to the boundary of two classes. Although, all samples close to the boundary may not necessarily be support vectors[1].

2.3 Remove samples that has less effect (far from boundary)

According to SVM theory, samples close to the decision boundaries have a higher likelihood of being a support vector. Therefore, the samples far from decision boundaries have less effect when identifying support vectors. This is a rule of thumb that can be used when discarding samples from the training set in order to reduce the size of huge training sets. The problem to be addressed is how to identify such samples that are far from the decision boundary.

3. Method and Examples

3.1 Example dataset

The idea behind this example data set is to have a non-linearly separable random data set for a two-class problem. For the purpose of illustration of the method, 1000 two dimensional points were created randomly in the range of [0,10]. A sinusoidal curve with mean 5 (mean of y value range) is used to separate the data points into two classes. Points above the curve are assigned +1 and those that are below the curve are assigned –1. As

shown in the Figure 1, signs belonging to class +1 are plotted as squares and the points belonging to the class –1 are plotted as crosses.

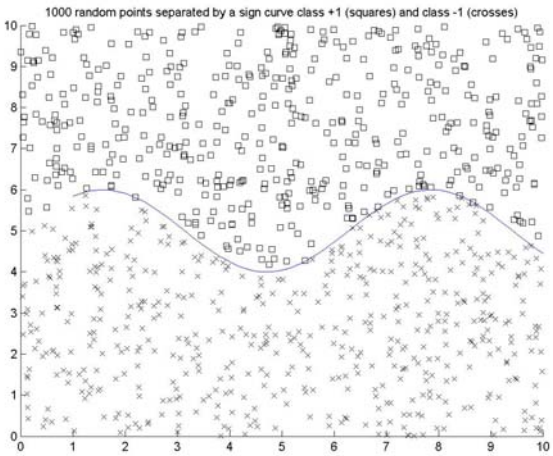

Figure 3.1: An example data set for a non-linearly separable two-class problem

3.2 Initial clusters

In the proposed technique, there are three major steps toward reducing the number of samples in the training set, without considerably effecting the SVM results.
They are as follows:

- Identify initial clusters or cluster centers
- Identify crisp clusters
- Detect samples to be removed

There are many well known clustering techniques which can be categorized into supervised or unsupervised clustering. In this case, any such method can be used to identify the initial clusters. The main consideration here is that the initial clustering method should be simple to implement and less costly in terms of computing power. A few examples of such methods could be Self Organizing Maps (SOM)[5], K-mean [6, 7], or fuzzy-c-mean[8]. In this paper, our algorithm employed the K-mean method for initial clustering.

3.3 Identification of crisp clusters

The next step is to identify clusters with samples from a single class (crisp clusters). Any clustering algorithm, either supervised or unsupervised can be used as the initial cluster selector. From these initial clusters, it is required to identify crisp clusters. Crisp clusters are clusters with the same class samples. There are two ways of finding crisp clusters. If the initial clusters are already defined as in a SOM, algorithm 1 (see below) can be applied. If only the cluster centers are known (k-mean), then the algorithm 2 might be a better option. In this paper, our algorithm employed the k-mean method for initial clustering.

Algorithm 1: Identify crisp clusters from known initial clusters

Note: This algorithm is defined only for the known initial clusters. If only the cluster centers of initial clusters are known, the use of algorithm 2 is recommended.

Step 1:
Assign the label ‘undecided’ to each cluster.
Step 2:
Select a cluster with the label ‘undecided’, t = 1.
Check the output class of each sample within the cluster.
If all the samples are in same class, assign it as a 'crisp' cluster.
If the samples are combination of two (or more) classes, assign it as 'non_crisp'.
Go to the next cluster
Step 3:
Select a label with non_crisp. Find the cluster centre and the radius.
Reduce the radius by ∇R and create a new cluster using the same cluster centre.
Where, ∇R can be an arbitrary value depending on the cluster size.
If the number of samples is less than three, assign 'not_a_cluster'.
Else assign label as 'undecided'.
Step 4:
If undecided clusters are remaining, goto step 2, t=t+1.
Else STOP.

Algorithm 2: Identify crisp clusters from known initial cluster centers

Note: This algorithm is defined only for the known initial cluster centers.

Step 1:
For all the cluster centers, increase the cluster radius (n =1) until 3 or more samples are in the cluster. Assign label 'undecided'.
Step 2:
select a cluster with the label 'undecided', t = 1.
If no samples with label 'undecided', STOP.
check the output class of each sample within the cluster.
Step 3:
If all the samples are in same class, increase the radius by ∇R and check again (n = n+1).
Where, ∇R can be an arbitrary value depending on the cluster size.
Repeat this step until the radius is big enough to include a combination of different class samples.
Select the cluster with previous radius (n = n-1, all samples are in same class).
Assign it as a 'crisp' cluster. Go to Step 2, t = t+1.

By applying algorithm 1 or 2 it is possible to come up with crisp sample clusters, that is, a cluster with samples from a single class. The next step is to refine the crisp sample clusters in order to remove the possible support vector samples. Figure 2 depicts such crisp clusters obtained using k-mean initial clustering with 6 initial clusters, and Algorithm 2.

3.4 Detection of samples to be removed

This is the most important part of the proposed technique of reducing the size of the training set so that the effect on selecting support vectors to be a minimum. Samples that are likely to be support vectors must not be removed in order to have the same results, when compared to results obtained with the whole training set. Support vectors are selected from the samples close to the decision boundary. Therefore, it is safer to remove samples far from decision boundaries. Here, the problem is how to detect samples far from decision boundaries, or the samples that have less effect on selecting support vectors. For example, in the above figure with crisp clusters, it is obvious that some of the samples within crisp clusters are either on the decision boundary or very close to it. Hence it is not a good idea to discard all samples within the crisp cluster in order to reduce the training set size. The presence of this problem was the motivation for the following improvement in order to select samples that have less effect in SVM construction.

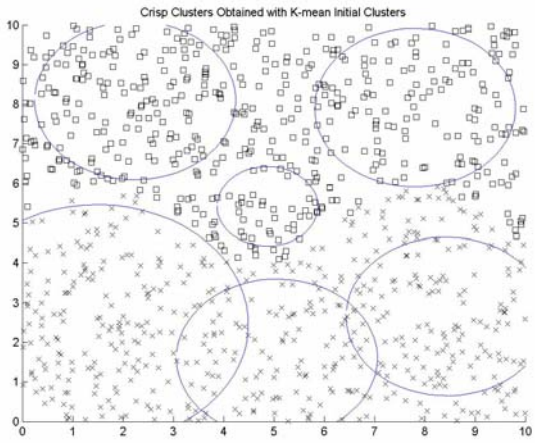

Figure 2. Crisp clusters obtained with Algorithm 2 and k-mean initial clusters with

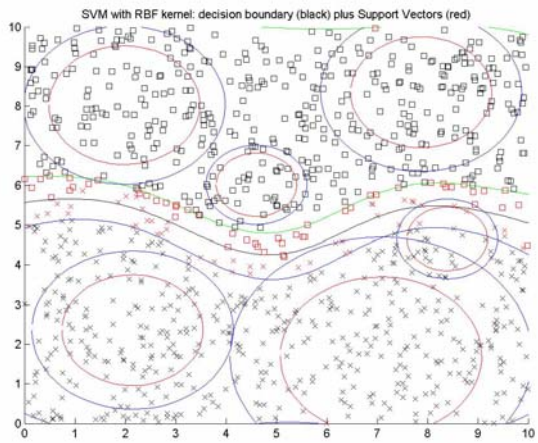

Figure 3. Problems associated with the Fixed Safety Region

3.4.1 Fixed Safety Region

Samples close to the cluster boundary have a high probability of affecting the support vector selection. Therefore, it is better to keep a safety region close to the cluster boundary, so that no samples would be removed from that region. This region is defined based on the cluster radius or number of samples within the cluster. The outcome would be an inner cluster, where samples within are far from the decision boundary. Samples within this inner cluster are the detected samples to be removed. Samples between the inner cluster and the outer cluster are the samples within the safety region and are not to be removed. To decide the inner radius, a threshold percentage value based on the number of samples, is used and defined as:

$$\text{Threshold samples percentage} = \frac{\text{samples within the inner cluster}}{\text{samples within the outer cluster}} * 100$$

If only the number of samples are considered, there is a possibility of having possible support vectors in the inner cluster (figure 3.3). In that figure, even though five out of six inner clusters are finding samples not likely to be support vectors, there is one cluster (mid, right) with support vectors. Here, k-mean initial clustering is applied to identify six cluster centers and then Algorithm 2 is used to identify crisp clusters. Inner clusters are obtained using a fixed threshold percentage of 50%. If the number of samples in the outer cluster = N, then the number of samples in the inner cluster $\geq$ 50% N. SVM decision boundaries using the RBF kernel are also plotted in the same figure. In order to overcome this problem, we have suggested combining both radius and the number of samples within the cluster to identify inner cluster with samples far from the decision boundary.

3.4.2 Variable safety region

In our proposed techniques, both the radius and number of samples were taken into consideration in deciding an inner cluster radius within the crisp cluster as follows:

If the threshold percentage is high, the samples to be removed from a crisp cluster are high and vice versa. It is noticeable that the smaller the outer radius, the closer the cluster centers are to the decision boundary. Similarly, the higher the outer radius, the further the cluster center is from the decision boundary. Taking this fact into consideration, the algorithm is further refined to have a variable threshold percentage with the cluster radius. If the cluster radius is small, the threshold percentage is lower, and on the other hand if the cluster radius is high, the threshold percentage is higher. This way it is possible to have a variable safety region within the crisp cluster based on its radius. In Figure 4 with a variable threshold percentage, it has managed to overcome the above problem. For the crisp clusters with a small radius that are close to the boundary, a smaller threshold percentage of 25% is applied, whereas, 50% is applied for higher radius clusters. For example,

If the outer radius is small (say $\leq$ 1 unit), threshold = 25 % and if the outer radius is small (say > 1 unit), threshold = 50 %.

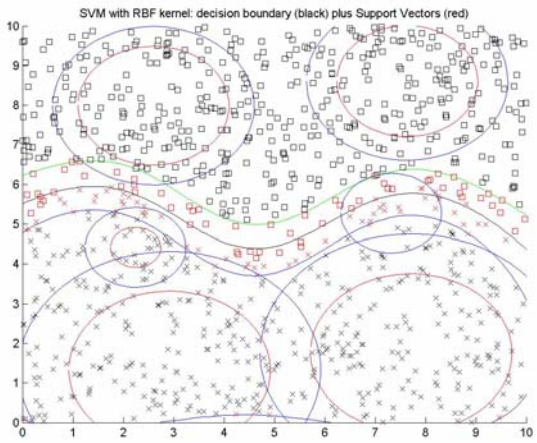

Figure 4. Inner clusters obtained with variable threshold percentage

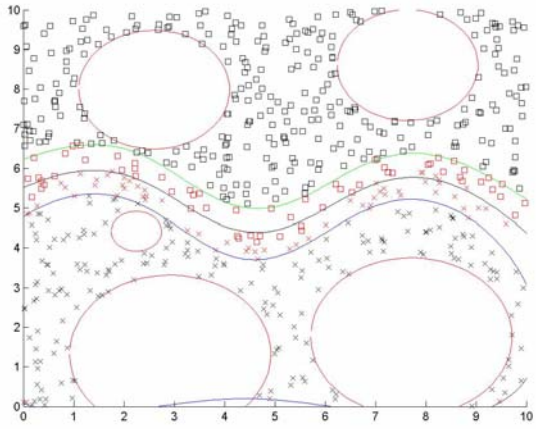

Figure 5. Reduced training set with the proposed algorithm.

Figure 5 shows the results obtained after removing the samples within the inner clusters and then applying SVMs with the same RBF kernel. It is interesting to see, when comparing with the results with those obtained using the whole training set, it has managed to get almost same results with the reduced training set.

4. Results

4.1 IRIS Dataset

The Iris dataset is a collection of 150 Iris flowers of 3 kinds, with four attributes, leaf and petal width and length in cm. Three classes are the iris-setosa, iris-versicolor and iris-virginica. 75 samples are used as the training set and the remaining 75 are the test set. The SVM polynomial kernel is applied to the training set. The first row of Table 1 shows values obtained for the whole training set. The following rows show the reduced training sets obtained by the proposed algorithm with k-mean clustering with parameter 4 (four cluster centers). This process has been repeated 5 times in order to get different results. The percentage number of samples removed from a cluster is 50 or more. The first column is the success rate of classification from the 75 samples of the test set. The last column shows the time taken for SVM calculations with the reduced training set. The best values are indicated in bold characters.

Table 1. Results summary for IRIS-one class vs Others.

IRIS-Setosa vs others			IRIS-Versicolor vs others			IRIS-Virginica vs others		
correctly classified %	Percentage number of samples removed	Total time taken for SVM calculation	correctly classified %	Percentage number of samples removed	Total time taken for SVM calculation	correctly classified %	Percentage number of samples removed	Total time taken for SVM calculation
100	0	1	97.33	0	4.3	86.67	0	2.1
100	64	0.2	95.67	32	2.6	86.67	33.33	1
100	77.33	0.1	97.33	42.67	1.2	86.67	52	0.5
100	77.33	0.1	97.33	42.67	1.2	86.67	53.33	0.4
100	81.33	0.1	97.33	42.67	1.2	86.67	53.33	0.5
100	**86.67**	**<0.05**	**97.33**	**46.67**	**1.2**	**86.67**	**57.33**	**0.4**

Table 2. Results summary for Monks problem with 16 initial clusters.

	Monks problem 1			Monks problem 2			Monks problem 3		
	Correctly classified %	Number of samples removed	Total time taken in seconds	Correctly classified %	Number of samples removed	Total time taken in seconds	Correctly classified %	Number of samples removed	Total time taken in seconds
Normal SVM	85.65	0	4.6	69.44	0	13.7	95.37	0	3.2
RTS Average	84.98	18.12	2.66	69.65	5.80	8.93	95.30	22.70	1.97
RTS maximum samples removed	85.42	**22.58**	2.1	68.06	**10.06**	7.1	92.82	**29.51**	1.4
RTS Best classification	**85.65**	16.13	2.7	**70.60**	7.69	9.6	**96.53**	24.59	2.2
RTS Fastest	85.42	22.58	**2.1**	68.06	10.06	**7.1**	92.82	29.51	**1.4**

4.2 Monks Dataset

The three problem sets defined for Monk's data set in Thrun et al [9] were used in this experiment. Monks problem 1 is in standard disjunctive normal form and is supposed to be easily learnable by most of the algorithms and decision trees. Conversely, Monk's problem 2 is similar to parity problems. It combines different attributes in a way that makes it complicated to describe using the given attributes only. Monks problem 3 serves to evaluate the algorithms under the presence of noise. Results are shown in the Table 2. The first row is the results of normal SVM and the other rows represent the Reduced Training Set (RTS). Best values are shown in bold characters.

4.3 Forest Dataset

The actual forest cover type for a given observation (30 x 30 meter cell) was determined from the US Forest Service (USFS) Region 2 Resource Information System (RIS) data[10]. Independent variables were derived from data originally obtained from the US Geological Survey (USGS) and USFS data. Data is in raw form (not scaled) and contains binary (0 or 1) columns of data for qualitative independent variables (wilderness areas and soil types). Initial 10,000 samples were selected and divided into two groups of 5,000 each, randomly. One such set is considered as the training set and the other as the test set. Three cover types, 1,2 and 5 were considered for the classification. SVM with RBF kernel is applied to support vector calculations. For the initial clustering in the proposed algorithm, k-mean clustering is used with the parameter value 6. Results obtained for the forest dataset can be found in the Table 3. The second column of the table shows percentage number of samples removed from the training set. The first column is the success rate of classification from the test set. The last column shows the time taken for SVM calculations with the reduced training set. The first row represents the results with whole training set, whereas the other five rows are the results with the reduced number of training samples using k-mean algorithm with k = 6. Best performances are shown in bold characters.

Table 3. Results summary for forest cover type vs the rest.

Forest cover type 1 vs the rest			Forest cover 2 type vs the rest			Forest cover type 5 vs the rest		
correctly classified %	Percentage number of samples removed	Total time in seconds T1 + T2	correctly classified %	Percentage number of samples removed	Total time in seconds T1 + T2	correctly classified %	Percentage number of samples removed	Total time in seconds T1 + T2
86.42	0	225.8	89.72	0	259.5	87.24	0	244.8
86.42	21.62	177.9	89.72	4.56	246.4	87.24	20.94	127.7
86.42	21.8	**175.3**	89.72	4.92	248.6	87.24	20.94	131.4
86.42	21.8	177.5	89.72	4.92	249.0	87.24	20.94	129.3
86.42	21.8	178.7	89.72	4.96	246.3	87.24	26.82	101.6
86.42	**21.8**	177.5	89.72	**5.66**	**242**	87.24	**26.88**	**100.3**

5. Discussion

A method to overcome the problems with huge training datasets for SVM is proposed. When the training set is huge, it is impossible to use it in its entirety for training process due to space and computational limitations. Although random or other heuristic methods can be used to reduce the size of the training sets, there is no guarantee of how much it will affect the final classification result. In the proposed technique, the behaviour of selecting support vectors was taken into account in attempting to minimize the effects on the final result caused by reducing the training set. To overcome the possible shortcomings of the proposed technique, a concept of a 'safety region' is introduced. It has been further enhanced by using a 'variable safety region' (3.4.2).

In SVM theory, it is easy to observe that samples close to the decision boundary are more likely to be selected as support vectors, and samples far away from the decision boundary have no effect on selecting support

vectors. Therefore, a technique is developed to identify the samples that are far from the decision boundary and to remove them in order to reduce the training set size. This ensures that the effects of selecting support vectors are minimum and hence has less effect on the final classification result. Introduction of a 'safety region' and a technique to decide a 'variable safety region' has further minimized such effects. A non-linearly separable artificial data set is used to demonstrate the feasibility of the proposed technique. Results of the example set shows that it is possible to reduce the size of training set without affecting the SVM results.

Results obtained with IRIS-Setosa (Table 1), show that it is possible to reduce the training set even up to 86%, without affecting the correctly classified rate. Since only a few samples were remaining in this case, the time taken for the calculations is minimal. In the case of IRIS-Versicolor (Table 1), there was one case, which has the reduced classification rate. In this case it has reduced 32% of the training set and the success rate from 97% to 95%. When the number of training samples is small as in the IRIS, there is a high possibility of selecting the samples that can affect the support vectors. Therefore, such variations in the success rate might be due to the removal of samples, which can affect support vector selection. This is one weakness of the proposed technique when applied with the small datasets. In IRIS-Virginica (Table 1), it has managed to reduce the training set up to 57% without affecting the classification rate. In all the cases, the time taken for SVM training is reduced with the reduced set of training samples. Again, since IRIS dataset has a small number of training samples with four features each, the time taken for SVM training is smaller and not a critical issue. However, results obtained with IRIS dataset shows that the proposed technique is a valid way to reduce the number of training samples of the SVM training set, with minimum effects on the support vector selection.

There were three problems defined in the Monks dataset. In all cases, the variation of success rate due to reducing the number of samples in the training set is minimum. This means that the proposed algorithm works well with the monk's dataset, even though it has a higher number of features when compared to the IRIS data. Those results confirm that the reduction of the number of training samples would lead to higher speeds when compared to the normal SVM results (using the whole training set). Another interesting observation of the result is that in certain cases, the classification rate has been increased more than the normal SVM. In general, the average classification rate is much closer to the normal SVM, and the average time taken is lower. Therefore the proposed technique is suitable for data sets similar to Monks, in order to increase the speed.

Best performances in terms of success rate and the time taken for SVM calculation were observed with the forest dataset. This could be due to the fact that the number of samples is huge when compared to the other two datasets. In all three cases with the forest dataset there was no change in the classification rate. The proposed algorithm has managed to remove samples of 21%, 5%, and 26% from the forest cover types 1,2 and 5 respectively. Those differences could be due to the variations of sample distributions. In all cases, the time taken for the SVM training has significantly reduced. For example, in the forest cover type 5, the time taken for SVM calculation has reduced from 244 seconds to 100 seconds.

In general, results obtained with all three datasets (IRIS, Monks, and forest) have shown that the proposed technique can be successfully applied to them, and the speed of SVM training can be increased without significant effect upon the classification results. With the IRIS dataset it managed to reduce a higher percentage of samples, when compared to the other two. But since it has few samples and attributes, it takes only a second or two for the support vector calculation. In contrast, the forest dataset had the highest number of samples and attributes, and hence the time taken for the SVM training is significant for the whole training set when compared to the reduced training set. Therefore with the proposed technique, advantages are more significant in huge data sets.

Acknowledgement

We thank Jock A.Blackard and Colarado State University for providing the Forest dataset. We also thank Murphy P.M., & Aha,D.W. (1994). UCI Repository of machine learning databases [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science. The authors also wish to acknowledge the patial funding provided by ARC.

References

[1] V.Vapnik, *the Nature of statistical learning theory*: Springer-Verlag, New York, 1995.

[2] B.W. Hwang, Jae-Jin Kim, S.-W. Lee;, "Retrieval of the top N matches with support vector machines," *Pattern Recognition, 2000. Proceedings. 15th International Conference on , IEEE*, vol. 2, pp. 716 -719 vol.2, 2000.

[3] T. Joachims, *Making large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning,*: MIT-Press, 1999.

[4] S. Sohn, Dagl C.H., "Advantages of using fuzzy class memberships in self-organizing map and support vector machines," presented at Proceedings. IJCNN '01. International Joint Conference on Neural Networks, 2001.
[5] Kohonen, *Self Organizing Maps*. Berlin: Springer, 1997.
[6] G. A. F. Seber, *Multivariate Observations*. New York: Wiley, 1984.
[7] H. Spath, *Dissection and Analysis: Theory, FORTRAN Programs, Examples*. New York: Halsted Press, 1985.
[8] J. C. Bezdek, 1939-, *Pattern recognition with fuzzy objective function algorithms*. New York: Plenum Press, 1981.
[9] B. J. Thrun S.B., "The MONK's Problems - A Performance Comparison of Different Learning algorithms." http://www.tech.port.ac.uk/~ml-algs/Data-sets/Database/Monks/.
[10] J. A. Blackard, "Forest CoverType Data." http://kdd.ics.uci.edu/databases/covertype/covertype.data.html: Remote Sensing and GIS Program, Department of Forest Sciences, College of Natural Resources, Colorado State University, Fort Collins, CO 80523, 1998.