JOURNAL OF PATTERN RECOGNITION RESEARCH 1 (2011) 56-64

Received October 3, 2010. Accepted February 5, 2011.

Training Set Compression by Incremental Clustering

Dalong Li *dalong.li@hp.com*
Hewlett-Packard Company
11445 Compaq Center Dr. West. Houston, TX 77070, USA

Steven Simske *steven.simske@hp.com*
Hewlett-Packard Company
3404 East Harmony Road, Fort Col lins , CO 80525, USA

Abstract

Compression of training sets is a technique for reducing training set size without degrading classification accuracy. By reducing the size of a training set, training will be more efficient in addition to saving storage space. In this paper, an incremental clustering algorithm, the Leader algorithm, is used to reduce the size of a training set by effectively subsampling the training set. Experiments on several standard data sets using SVM and KNN as classifiers indicate that the proposed method is more efficient than CONDENSE in reducing the size of training set without degrading the classification accuracy. While the compression ratio for the CONDENSE method is fixed, the proposed method offers variable compression ratio through the cluster threshold value.

Keywords: Clustering, Support vector machine, KNN, Pattern recognition, CONDENSE.

WWW.JPRR.ORG

1. Introduction

The training and/or testing complexity of a classifier usually depends on the size of the training set, e.g. the nearest neighbor (NN) classifier [1]. Nearest neighbor and its generalized form, the K-nearest neighbor (KNN) classifier, are among the most popular non-parametric classifiers. The membership of an unknown sample is classified based on the majority vote of the K nearest neighbors. There is no explicit learning from the training set. The entire training set itself defines the decision boundaries. It is conceptually simple and shows good performance in many applications, e.g. it was used in face recognition for visitor identification [2] and it was shown that it outperformed more sophisticated algorithms that use Principal Components Analysis (PCA) and neural networks. Unfortunately, when the size of the training set is high, it requires a lot of memory to store the entire training set and it also takes longer to search for the nearest neighbors of a given test pattern to make a single membership classification. Obviously, reducing the size of a training set can improve the space and time efficiency of KNN. There has been considerable interest in reducing the training set size by editing, especially in the context of NN. Different proximity graphs (such as Delaunay triangulation) may be used for editing NN rules [3, 4]. Complexities of such approaches are prohibitively high. For example, the Voronoi diagram has worst case complexity of

$$\Theta(n^{\frac{d}{2}}) \tag{1}$$

[5] where n is the number of samples and d is the number of dimensions. The complexity of the Gabriel graph approach is $O(dn^3)$. Besides high complexities, these approaches are proposed only for the nearest neighbor (1-NN) rule, which is not robust on noisy data. KNN ($k > 1$) is more robust than NN. Those graph approaches can not be directly used in editing KNN. The CONDENSE algorithm was first described by [6]. This algorithm selects a subset of the training examples whose 1-NN decision boundary would still classify correctly all of the initial training examples. Effectively, the CONDENSE algorithm removes interior examples in the training set, similar to support

© 2011 JPRR. All rights reserved. Permissions to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or to republish, requires a fee and/or special permission from JPRR.

vectors. In this paper, it is compared with the proposed method.

Support vector machines (SVMs) [7, 8] represent a new generation learning system based on recent advances in statistical learning theory [9]. SVMs deliver state-of-the-art performance in many real-world applications such as hand-written character recognition, image classification, image restoration [10], etc. SVMs have been established as one of the standard tools for machine learning and data mining. The LibSVM software package [11] is an implementation of SVM and is used in this study. The training complexity of LibSVM is #Iterations $\times O(n)$ if most columns of Q are cached during iterations. n is the number of samples in the training set and Q is an $n \times n$ positive semidefinite matrix, $Q_{ij} \equiv y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$, and $K(\mathbf{x}_i, \mathbf{x}_j) \equiv \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ is the kernel. Reducing the size of the training set n will speed up training of the SVM. In this work, both KNN and SVM are used as the classifiers for the purpose of comparing the two training set compression algorithms.

The purpose of compression is to remove redundancy in a training set. Sequential leader clustering [12] is one of classical clustering algorithms. This clustering algorithm can be used to remove redundancy. The discarded samples are redundant since they are close to one of the samples in the clusters, also known as codebook or dictionary in image compression. The process of the clustering can also be viewed as subsampling since the clustering guarantees that all the samples in the codebook are separated at a distance that is above a minimal threshold. In this paper, this subsampling based compression is introduced.

The rest of this paper is organized as follows. In Section 2, the CONDENSE algorithm and the leader clustering algorithm are reviewed. Some synthetic examples are shown to illustrate how they edit a training set by removing redundant samples and how the KNN and SVM decision boundaries vary as the training set is edited. In Section 3, the experiments results on several standard machine learning databases are reported. Some brief conclusions are in Section 4.

2. Editing training set

Given a training set: $\Omega = \{\mathbf{x}_i | i = 1, 2, 3, \ldots, n\}$, the goal of compression is to find a subset $\mathbf{R}$, $\mathbf{R} \subseteq \Omega$ so that the size of $\mathbf{R}$ is reduced but the reduction in the training set size shall not degrade the accuracy of a classifier trained on $\mathbf{R}$.

2.1 CONDENSE

Editing Ω by the CONDENSE algorithm takes the following steps.

1. A training sample $\mathbf{x}_i$ from each class is randomly selected from Ω and is put in set $\mathbf{R}$.
2. Each sample in Ω is classified using the 1-NN rule with $\mathbf{R}$ as the training set, and any misclassified sample is inserted into $\mathbf{R}$.
3. Repeat previous step if $\mathbf{R}$ has been incremented during the last pass.
4. When $\mathbf{R}$ is no longer updated, the final $\mathbf{R}$ is then the condensed training set of Ω.

2.2 Sequential Leader Clustering

The only parameter in the Leader clustering is the cluster threshold T. The following steps are taken in sequential leader clustering:

1. A cluster threshold value T is assumed.
2. $\mathbf{R}$, the codebook, is initialized by a randomly selected sample from Ω.
3. For each sample $\mathbf{x}_i$ in Ω, compute the distance d_{ij} between $\mathbf{x}_i$ and every sample $\mathbf{x}_j$ in $\mathbf{R}$.

4. Compute the minimum of d_{ij}, the distance between $\mathbf{x}_i$ and the closest sample in $\mathbf{R}$.

5. If the minimum distance is smaller than T, this sample is matched with one of the samples in the codebook and thus it is not added into the codebook. Otherwise, if the minimum distance is larger than T, meaning there is no match, the sample is a new cluster and it is added into $\mathbf{R}$.

6. When all samples are processed as in steps 3-5, the clustering finishes and $\mathbf{R}$ is the final codebook.

The above clustering algorithm will remove any sample that is matched with one of the samples in the codebook (i.e. close enough to one of the samples in the codebook). Thus, the density of the samples in the reduced set is effectively smaller than that of the original data set. In this sense, the clustering algorithm effectively subsamples the data set. The reduction in size depends on T: if T is too small, very few samples will be removed; on the other hand, a large T might remove too many samples. Fig. 1 shows the compression ratio v.s. the accuracy when T varies on the Breast-cancel dataset. It clearly indicates that as more samples are removed, the accuracy normally decreases. Details on the experiment are described in Section 3. Though T is adjustable, in this work, T is estimated by some statistics: first, a subset of samples are randomly chosen and the distance between each of those samples and its nearest neighbor is computed. Then the mean of those distances is used as T. In practise, one can choose a different method to estimate it. For example, one might want to compute a curve similar to the one show in Fig. 1 on a subset. Then, depending the goals and constraints such as memory and/or storage size of the computing environments, a proper T can be used.

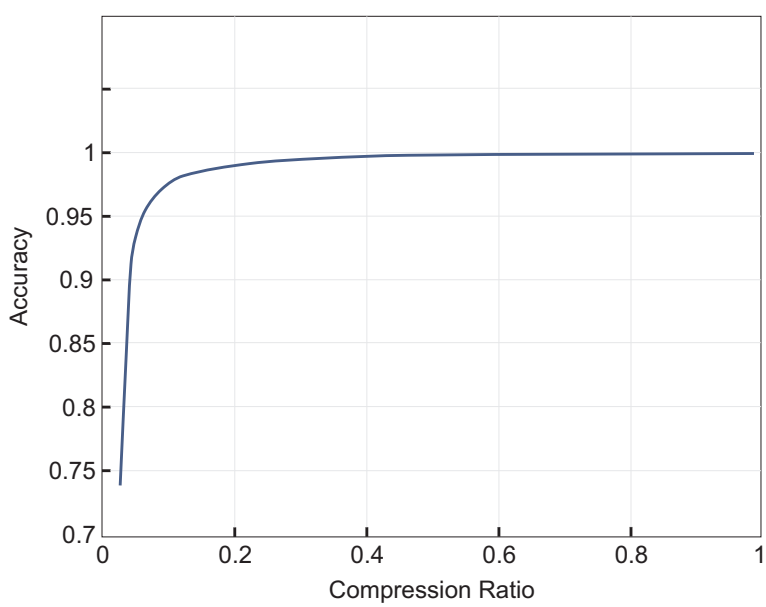

Fig. 1: Compression ratio v.s. accuracy when T varies for the proposed method on the Breast-cancel dataset with KNN (K=9).

Fig.2 shows two synthetic examples that compare leader and CONDENSE in reducing the size of the training sets. The two datasets are in the first row. The samples from the two classes are separated from each other in the first dataset. In such case, CONDENSE efficiently removes most of the training samples, the reduced training set is minimal. There are only 2 samples left in the

training set. When there is overlapping of the samples from the two classes, CONDENSE is less efficient in reducing the number of samples in the training set. Unlike CONDENSE, the leader clustering always reduces the size of the training set to a certain degree without being affected by how the training data is distributed. This observation is further confirmed by the experiments in Section 3.

Fig.3 shows another synthetic example. In this example, only one data set is used. The original data set before editing is shown in the first row; the second row is the data set edited by the leader algorithm; the third row is that edited by the CONDENSE algorithm. The left column shows the decision boundary of the NN classifier while the right column shows the decision boundary of the SVM. The decision boundaries of SVM in the synthetic example is computed by the program svmtoy, which is available in the LibSVM package [11]. The SVMtoys program in LibSVM has parameters of $-t2$ (Gaussian Kernel is used) $-c100$ (the cost in C-SVC). It can be seen that the decision boundaries training on the data set edited by the leader clustering (a2 and b2) are very similar to those on the original full data set (a1 and b1). In the case of the data set edited by the CONDENSE, though the NN boundary (c1) looks similar to the original one (a1), the SVM boundary (c2) significantly deviates from the original one (b1).

3. Experiments

Several datasets from the UCI machine learning repository [13] and the Stalog repository [14] are used in the experiments. All of the data sets are binary classification problems and the detailed information about the databases is listed in Table 1. For each data set, 80% of the data is for training and 20% of the data is for testing.

Table 1: Description of Data sets.

Collection Name	Num of Instance	Num of features
Breast-cancer(1)	683	10
Australian(2)	690	14
Diabetes(3)	768	8
German number (4)	1000	24
heart (5)	270	13
Ionosphere(6)	351	34

Table 2: Comparisons of leader and CONDENSE in reducing the size of the training sets. Training#: number of samples in the training set;

Dataset	No editing		Leader		Condense	
	Training #	Ratio	Training #	Ratio	Training #	Ratio
1	546	1	403	0.2619	**253**	**0.5366**
2	552	1	**394**	**0.2862**	524	0.0507
3	614	1	**451**	**0.2655**	533	0.1319
4	800	1	**582**	**0.2725**	679	0.1513
5	216	1	**160**	**0.2593**	183	0.1528
6	281	1	212	0.2456	**204**	**0.2740**

The training sets are edited by the proposed leader algorithm and the CONDENSE algorithm. Then the edited training sets and the original training sets are used to train both KNN ($k = 9$) and

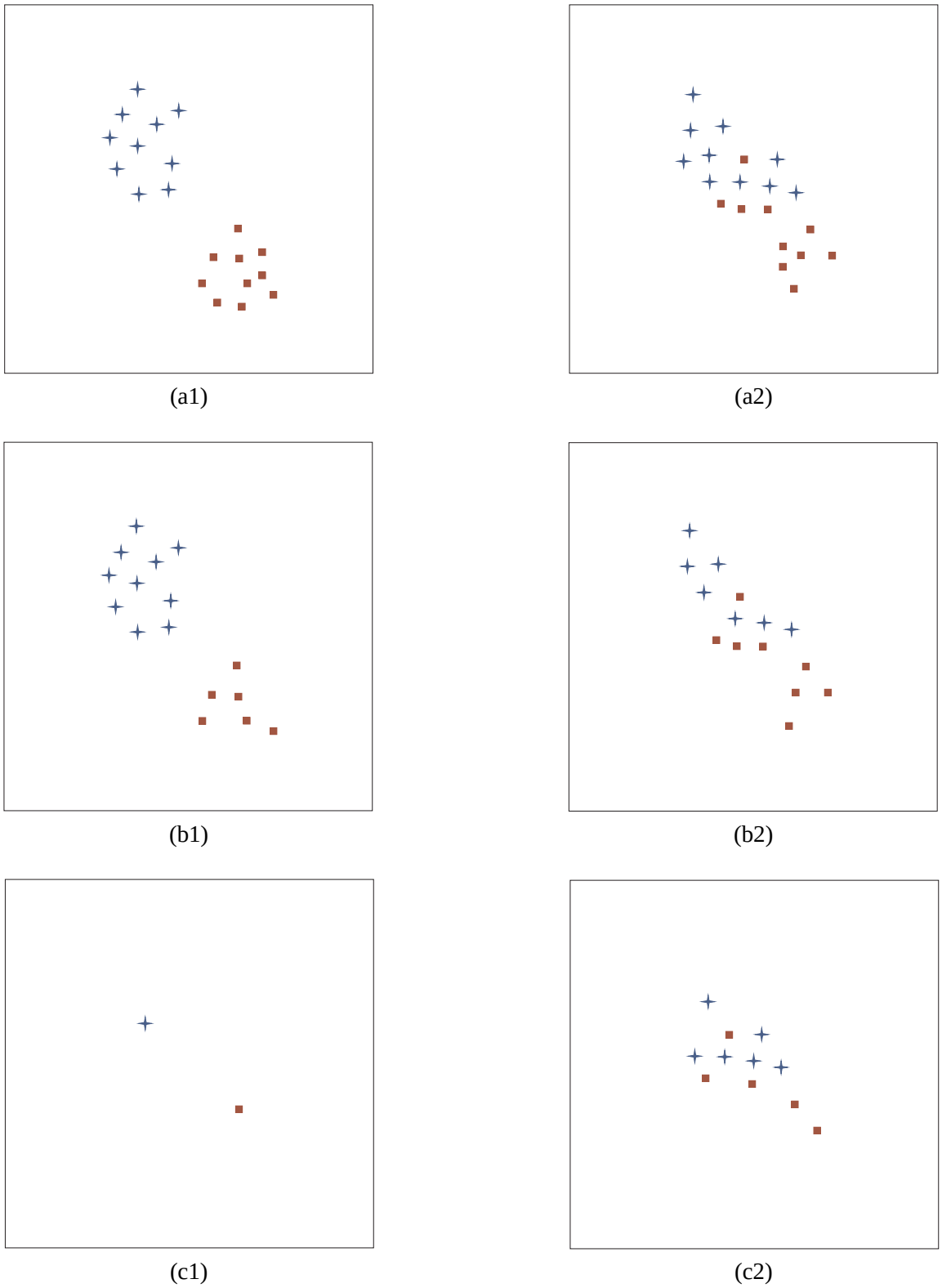

Fig. 2: Two synthetic data sets (1,2) examples of editing by leader and CONDENSE. The first row (a1) and (a2) show the two data sets (1 and 2). The second row (b1) and (b2) are the sets edited by leader. The third row (c1) and (c2) are the sets edited by CONDENSE.

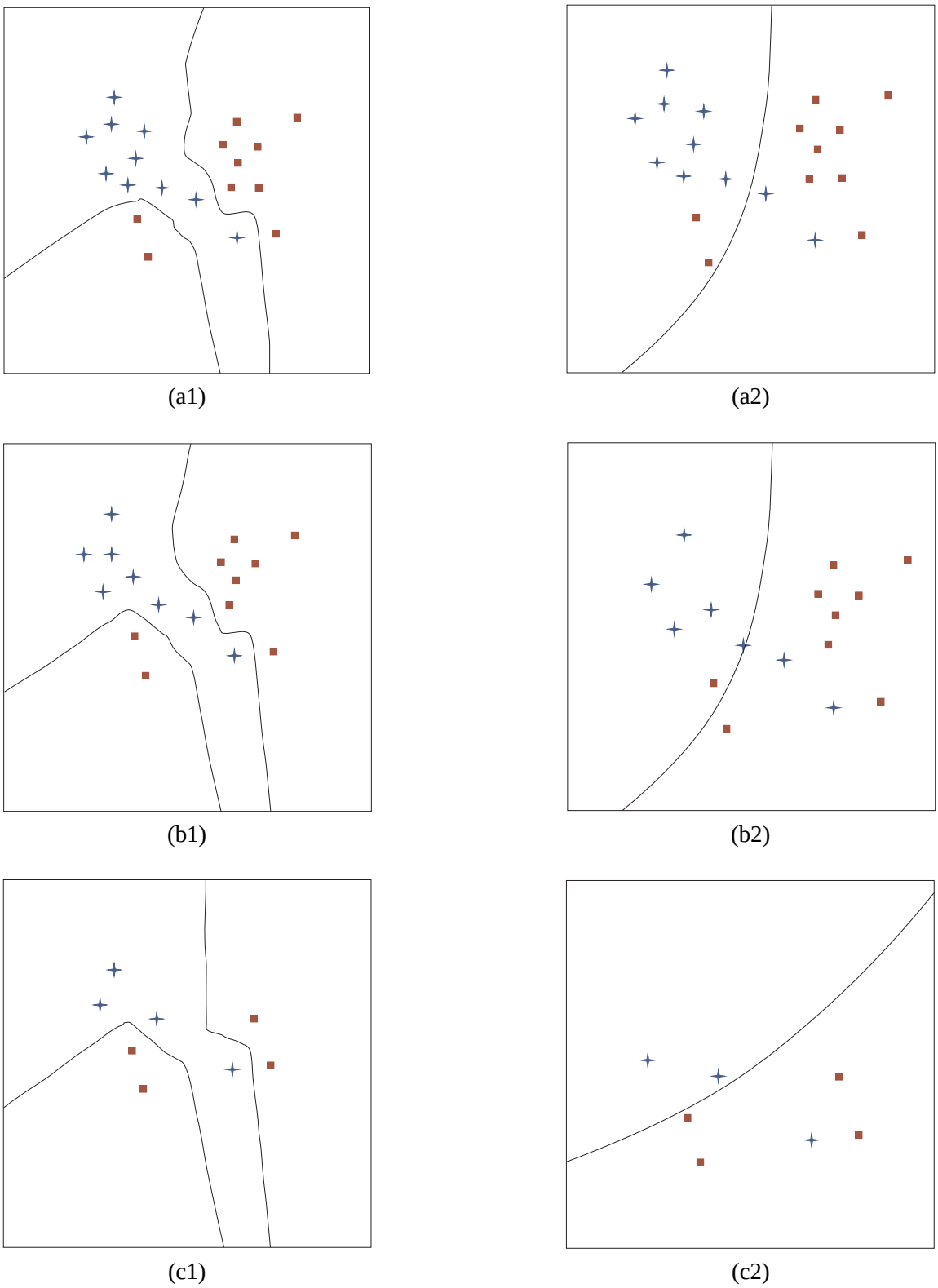

Fig. 3: Another synthetic example. In this example, both the data sets and classification boundaries of NN and SVM are shown. First column shows the decision boundary determined by NN; second column shows the decision boundary defined by SVM. First row shows the original training set. Second row shows the sets edited by leader and the last row shows the sets edited by CONDENSE.

the SVM classifiers. The parameters of LibSVM are all the default ones. The default kernel is Gaussian. The accuracies are computed and compared. Table 2 summarizes the comparative results of editing by the leader and the CONDENSE. The ratio in the table reflects the efficiency of the editing methods in reducing the numbers of samples in the training sets. It is defined as:

$$ratio = \frac{Full\,Training\# - Reduced\,Training\#}{Full\,Training\#} \tag{2}$$

The training set sizes are listed in the table. Except for the dataset 1 and 6, the proposed subsampling based editing removes significantly more samples. Comparing the reduction ratio shown in Fig.2, we suspect that the samples in dataset 1 and 6 are well separated, similar to Fig. 2. (a) in the synthetic example; in all the other datasets, the samples are overlapped to a large degree. Therefore, CONDENSE removes a very small percentages of samples under these conditions.

Table 3 shows the accuracy comparison using the KNN classifier. The accuracies are very similar. As a matter of fact, the editing might even increase classification accuracies, e.g, in data set 3 where the classifiers trained on reduced sets achieve higher accuracies.

Table 3: Comparisons of leader and CONDENSE using KNN (K=9) as classifier. #correct is the number of correctly classified samples.

Dataset	Test#	No editing		Leader		Condense	
		correct#	Accuracy	correct#	Accuracy	correct#	Accuracy
1	137	137	1.0000	137	1.0000	137	1.0000
2	138	116	0.8406	114	0.8261	116	0.8406
3	154	115	0.7468	118	0.7662	118	0.7662
4	200	148	0.7400	151	0.755	149	0.745
5	54	43	0.7963	41	0.7593	42	0.7778
6	70	68	0.9714	69	0.9857	68	0.9714

Table 4 summarizes the comparative results using the SVM as the classifier. The kernel is Gaussian. The accuracies of SVM trained on the edited sets are very similar to those of SVM on the original data set. This implies that the samples discarded by both CONDENSE and the leader clustering are likely non-contributors to the support vectors.

Table 4: Comparisons of leader and CONDENSE using SVM (Gaussian Kernel). #correct is the number of correctly classified samples.

Dataset	Test#	No editing		Leader		Condense	
		correct#	Accuracy	correct#	Accuracy	correct#	Accuracy
1	137	136	0.9927	136	0.9927	136	0.9927
2	138	120	0.8696	119	0.8623	121	0.8768
3	154	117	0.7597	116	0.7532	116	0.7532
4	200	154	0.7700	158	0.79	154	0.77
5	54	44	0.8148	43	0.7963	43	0.7963
6	70	68	0.9714	69	0.9857	68	0.9714

Table 5 shows comparative classification results using SVM classifiers trained on full and reduced training set (by the Leader algorithm). Different SVM kernels are used in this experiment. From the results, we can see that sometimes a reduced training set generates a slightly better classifier,

Table 5: Comparisons of classification accuracy on full and leader clustering reduced training set using different SVM kernels.

	linear		cubic		gaussian		sigmoid	
	reduced	full	reduced	full	reduced	full	reduced	full
1	0.9854(135)	0.9854(135)	0.9854(135)	0.9854(135)	0.9927(136)	0.9927(136)	1.0000(137)	1.00(137)
2	0.8623(119)	0.8623(119)	0.8551(118)	0.8623(119)	0.8623(119)	0.8696(120)	0.8623(119)	0.8623(119)
3	0.7532(116)	0.7468(115)	0.6688(103)	0.6494(100)	0.7532(116)	0.7597(117)	0.7922(122)	0.7662(118)
4	0.7800(156)	0.7750(155)	0.7550(151)	0.7350(147)	0.7900(158)	0.7700(154)	0.7700(154)	0.7750(155)
5	0.8333(45)	0.8333(45)	0.8148(44)	0.8148(44)	0.7963(43)	0.8145(44)	0.8519(46)	0.8333(45)
6	0.9714(68)	0.9714(68)	0.3857(27)	0.6429(45)	0.9857(69)	0.9714(68)	0.9571(67)	0.9714(68)

although sometimes classifiers trained on the full training set without compressing perform better. It also shows that when different kernels are used, the performance might differ significantly as shown in dataset 6 with polynomial as kernel. Generally, the default kernel, Gaussian, is a good choice. The choice of kernel is beyond the scope of this paper.

4. Conclusion

In this paper, the Leader clustering algorithm is proposed to compress the training set. This compression is achieved by a subsampling process. Despite how the samples in the training set are distributed (i.e. overlapped or not), the proposed method can effectively reduce the size of the training set while maintaining the same level of classification accuracies. Comparing with the CONDENSE algorithm, the proposed subsampling based compression generally reduces more samples from the training set while the classification accuracies are comparable. The CONDENSE was designed for the NN classifier. The proposed subsampling approach is not classifier dependent, nor is it data distribution dependent. Moreover, the compression ratio is adjustable through the threshold in the Leader clustering algorithm. Those are the main advantages of the Leader algorithm in compressing a training set.

References

[1] T. M. Cover and P. E. Hart, ”Nearest neighbor pattern classification”, IEEE Trans. Information Theory, Vol. 13, No. 1, pp. 21-27, 1967.

[2] T. Sim, R. Sukthankar, M. Mullin, and S. Baluja, ”Memory-based face recognition for visitor identification”, in Proceedings of Face and Gesture, March 2000.

[3] B. Bhattacharya, R. Poulsen, and G. Toussaint, ”Application of proximity graphs to editing nearest neighbor decision rule”, in Proc. International Symposium on Information Theory, Santa Monica, CA, USA, 1981.

[4] G. Toussaint, ”The relative neighborhood graph of a finite planar set”, Pattern recognition, Vol. 12, No. 4, pp. 261-268, 1980.

[5] V. Klee, ”On the complexity of d-dimensional voronoi diagrams”, Archiv der Mathematik, Vol. 34, pp. 75-80, 1980.

[6] P. E. Hart, ”The condensed nearest neighbor rule”, IEEE Trans. Information Theory, Vol. 14, No. 3, pp. 515-516, 1967.

[7] C. Cortes and V. Vapnik, ”Support-vector networks”, Machine Learning, Vol. 20, September 1995.

[8] N. Cristianini and J. Shawe-Taylor, ”Support Vector Machines and other kernel-based learning methods”, Cambridge University Press, Cambridge, 2000.

[9] V. Vapnik, ”The Nature of Statistical Learning Theory”, Springer Verlag, New York, 1995.

[10] D. Li, R. M. Mersereau, and S. Simske, ”Blind image deconvolution through support vector regression,” IEEE Trans. Neural Networks, vol. 18, no. 3, pp. 931-935, 2007.

[11] C. C. Chang and C. J. Lin, ”LIBSVM: a library for support vector machines”, Software available at, 2001, http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[12] J. A. Hartigan, ”Clustering Algorithms”, John Wiley & Sons, New York, 1975.

[13] C. L. Blake D. J. Newman, S. Hettich and C. J. Merz, ”UCI repository of machine learning databases”, 1998.

[14] D. Michie, D. J. Spiegelhalter, and C. C. Taylor, ”Machine Learning, Neural and Statistical Classification”, Ellis Horwood Series in Artificial Intelligence, 1994.