Adaptive Boosting with Leader based Learners for Classification of Large Handwritten Data

T.Ravindra Babu, M.Narasimha Murty
Department of CSA, Indian Institute of Science
Bangalore-560012
trbabu,mnm@csa.iisc.ernet.in

V.K.Agrawal
ISRO Satellite Centre
Bangalore-560017
agrawal@isac.ernet.in

Abstract

Boosting is a general method for improving the accuracy of a learning algorithm. AdaBoost, short form for Adaptive Boosting method, consists of repeated use of a weak or a base learning algorithm to find corresponding weak hypothesis by adapting to the error rates of the individual weak hypotheses. A large, complex handwritten data is under study. A repeated use of weak learner on the huge data results in large amount of processing time. In view of this, instead of using the entire training data for learning, we propose to use only prototypes. Further, in the current work, the base learner consists of a nearest neighbour classifier that employs prototypes generated using "leader" clustering algorithm. The leader algorithm is a single pass algorithm and is linear in terms of time as well as computation complexity. The prototype set alone is used as training data. In the process of developing an algorithm, domain knowledge of the Handwritten data, which is under study, is made use of. With the fusion of clustering, prototype selection, AdaBoost and Nearest Neighbour classifier, a very high classification accuracy, which is better than reported earlier on the considered data, is obtained in less number of iterations. The procedure integrates clustering outcome in terms of prototypes with boosting.

1. Introduction

Boosting attempts to improve the accuracy of a learning algorithm. It is based on the theoretical framework provided by [16] for machine learning. [9], [10] observed that a weak learning algorithm that could perform better than random guessing can be boosted to an arbitrarily strong learning algorithm. [12] and [4] provided boosting algorithms subsequently.

The Adaptive Boosting Algorithm was introduced by [4]. Theoretical aspects of AdaBoost with a complete overview were provided by [12, 13]. Adaptive Boosting using Naive Bayes is carried out by [17].

Given a set of training data, AdaBoost makes use of a weak learner or a base learner. The training data for the weak learner is considered according to some weights and classification is performed. Initially equal weights are assigned to each training pattern. At every subsequent iteration, error in classification in the previous iteration is made use to update the weights for the next iteration. The updates of the weights is such that the weights of incorrectly classified examples are increased so that the weak learner is forced to focus on the hard examples in the training set. It was theoretically shown [12, 13] that with training algorithm slightly better than random, training error drops exponentially. Section-4 and 5 discuss AdaBoost.

From the above discussion and also from [12, 13], it is clear that in each iteration, choice of base learning algorithm and amount of training data that is provided for learning, influence the efficiency. As the number of iterations increases, the processing time becomes significant.

For the current study, a 10 category, 192 featured handwritten data set of 10003 patterns is considered. Out of this data 6670 patterns form the training data and 3333 patterns form the test data. 40% of the training data is considered as validation data. Preliminary analysis on the central tendency and dispersion of the data is provided in Section-2. Section-3 contains detailed discussion on the choice of base learning algorithm.

While applying AdaBoost to the current data, one option is to use the entire data at each iteration subject to the weight distribution. This would obviously consume large amount of processing time. At this stage, three considerations emerge. Firstly, if the training time could be reduced by some manner, use of AdaBoost would be more efficient. Secondly, efficiency also depends on the nature of base learning algorithm. Thirdly, while dealing with 10-category data, whether inherent similarities in the handwritten digits would help in designing an efficient algorithm, which brings the domain knowledge of the data into use. These considerations lead to an efficient, multi-stage algorithm. This is discussed in detail in Section-5.

Section-6 contains a discussion on the experimental results using the HW data as well as few datasets from UCI repository [11]. Section-7 provides summary and conclusions. Section-8 presents the scope for further work.

2. Preliminary Analysis of Handwritten Data

The handwritten digit data under study consists of training data of 6670 patterns. The data is equally divided into 10 classes, viz., 0 to 9. Each digit consisted of 192 binary values, resulting into 16X12 Handwritten(HW) digit.

Before embarking on to the algorithm development the measures of central tendency and dispersion are computed. They are computed by considering non-zero features of the binary pattern. The analysis helps in understanding the data, which is made use in developing the algorithm. For example in case of class-0, the patterns consist of features ranging from 39 to 121. High standard deviation values of around 12 are observed for the classes 0,2,3,5 and 8. Least standard deviation of 5.9 was observed for the class 1. Also, it was observed that patterns in the data vary in terms of orientation of the digit, width and height.

From this initial analysis and the knowledge of the character shapes, it is found that the sets of HW digits, viz., (0,6,3,5,8) and (1,2,4,7,9) share commonality, in terms of shape,features,inter-pattern distance and size. This observation is made use further in designing the algorithm.

3. Choice of Prototype Selection Algorithm

Boosting C4.5 [14] and comparing boosting C4.5 and boosting stumps [4] was earlier carried out. AdaBoosting Naive Bayes scoring was tried in [17]. In the current work, the base learner is a Nearest Neighbour Classifier(NNC) that employs prototype set generated from training data using Leader algorithm. The prototype set is used for NNC instead of entire data. One of the objectives of the current work is to present only prototypes from the data to the base learner instead of entire data. Prototype selection with the same set of HW digit data using Partition Around Medoids(PAM) [8], CLARA [8] and Leader [15] was earlier reported [1]. It was brought out [1] that Leader performed better than PAM and CLARA in terms of classification accuracy. Also the computation time for Leader is much less compared to both PAM and CLARA for the same HW data. The complexity of PAM is O(k$(n-k)^2$) and that of CLARA is of O(ks^2+k(n-k)). Leader has linear complexity. Although CLARA can handle larger data than PAM, its efficiency depends on sample size and unbiasedness of the sample.

In the current Section, a comparison of two algorithms, viz., Condensed Nearest Neighbour(CNN) [5] and Leader [15] clustering algorithms is carried out for prototype selection.

The outline of CNN [2] is provided in Figure-1.

1. *Set two bins called STORE and GRABBAG*
2. *The first sample is placed in STORE*
3. *The second sample is classified by NN rule, using current contents of store as reference set. If the second sample is classified correctly, it is placed in GRABBAG; Otherwise it is placed in STORE.*
4. *Proceeding in this manner, the i^{th} sample is classified by the current contents of STORE. If classified correctly, it is placed in GRABBAG; otherwise it is placed in STORE.*
5. *After one pass through the original sample set, the procedure continues to loop through GRABBAG until termination, in one of the following ways*
    - *The GRABBAG is exhausted, with all its members, transferred to store, or*
    - *One complete pass is made through GRABBAG with no transfers to STORE.*
6. *The final contents of STORE are used as reference points for the NN rule; the contents of GRABBAG are discarded*

Fig.1 Condensed Nearest Neighbour rule [5]

The Leader clustering algorithm [15] [7] is provided in Fig. 2. It is evident from Fig.2 that number of leaders depends on the choice of the threshold. As the threshold increases, the number of leaders reduces.

A comparative study is conducted between CNN and Leader, by providing entire 6670 patterns as training data and 3333 as test data for classifying them with the help of NNC. Table-1 provides the results.

1. *Choose a dissimilarity threshold. Start with an arbitrary pattern. Call it Leader-k, where k=1*
2. *For i=1 to n(total number of patterns in the training data)*
    - *Find the dissimilarity between Leader-k and the training pattern.*
    - *If the distance is less than the threshold, place the pattern in already classified set. Else, consider the pattern as a new leader. Update k.*
3. *Repeat step 2 till end of training data*

Distance Threshold	No. of Prototypes	C.A.(%)	CPU Time(sec)
CNN			
-	1610	86.77	942.76
Leader			
5	6149	91.24	1171.81
10	5581	91.27	1066.54
15	4399	90.40	896.84
18	3564	90.20	735.53
20	3057	88.03	655.44
22	2542	87.04	559.52
25	1892	84.88	434.00
27	1526	81.70	363.00

Table 1. Comparison between CNN and Leader

Fig.2 Leader algorithm [15]

In Table-1, C.A. refers to Classification Accuracy and CPU Time refers the processing time taken on Pentium III 500 MHz computer. The table, apart from comparing both the methods, demonstrates the effect of threshold on number of prototypes selected, C.A. and processing time. A discrete set of threshold values is chosen to demonstrate the effect of distance threshold. For binary patterns, the Hamming distance provides equivalent information as Euclidean distance metric, while avoiding the need to compute the square root. Hence Hamming distance is chosen as dissimilarity measure, in view of binary patterns.

From the exercises, it is clear that CNN consumes more time as compared to Leader algorithm for obtaining same Classification Accuracy. On the other hand, CNN generates fewer but fixed set of prototypes for a given order of input data. Leader algorithm offers a way of improving classification accuracy by means of threshold value based prototype selection and thus provides greater flexibility to operate with. In view of this, Leader is considered in the base learner in the current study.

4. AdaBoost Procedure

The AdaBoost algorithm [14] iteratively calls a given weak or base learning algorithm. The outline of the algorithm is provided in Fig.3.

1. *Input: m-labeled two-class patterns, (x_i, y_i),i=1,2...m, where x_i are training data belonging to an instance space X and y_i are the corresponding two-class labels, belonging to Y={-1,+1}*
2. *Initialize weights, $D_1(i) = \frac{1}{m}$*
3. *For each of iteration,t=1,... T, carry out the following*
   - *Train weak learner using distribution D_t*
   - *Weak learners finds a weak hypothesis, $h_t : X - \to \{-1, 1\}$ for the given weight distribution, D_t.*
   - *The error in the weak hypothesis is given by, $\epsilon_t = P_{i \sim D}$ [$h_t \neq y_i$]*
     Here, D represents current weight distribution
   - *Choose $\alpha_t = \frac{1}{2} \ln \left(\frac{1-\epsilon_t}{\epsilon_t} \right)$*
   - *Update*

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times e^{-\alpha_t}, \text{ if } h_t(x_i) = y_i$$
$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times e^{\alpha_t}, \text{ if } h_t(x_i) \neq y_i$$
$$= \frac{D_t(i) exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

where Z_t is such that $\sum_{i=1}^{m} D_{t+1}(i) = 1$

4. *Output the final hypothesis:*

$$H(x) = sign(\sum_{i=1}^{T} \alpha_t h_t(x))$$

Fig.3 AdaBoost algorithm [14]

In the current implementation, selection of prototypes is made according to the weights D_t, instead of using them directly on the training examples [14].

4.1. Bootstrapping of data

The training data of 6670 samples is subdivided into a training data set of 4000 samples and validation data set of 2670 samples. A bootstrapped data set [6] of 10000 samples is generated from the 4000-sample training data set by means of simple random sampling with replacement. The proposed algorithm is trained on the leaders generated from this set of 10000 samples and validation is carried out with 2670 samples. The leader set, which provides best CA with validation dataset, is tested against the test data and the results are presented. The 10000-sample data is hereafter referred to as training data set and original 6670-sample data set as original training data.

5. Proposed Algorithm

The proposed procedure divides entire clustering and classification procedure as multiple 2-class problems. Based on the preliminary analysis on the data, similarity is observed among the two sets of classes, viz., set-1:(0,6,5,3,8) and set-2:(1,2,4,7,9). This observation is made use in devising the method.

At *stage-1*, the method consists of AdaBoost procedure for finding prototypes that would classify the given data into one of above two sets. The method is based on the base learner of NNC employing prototype set generated by leader-clustering algorithm in every iteration. In doing so, it considers complete training data of 10000 samples as input. The classes of set-1 are termed as an equivalent class of A and those of set-2 as B.

The input data for training the base learner is selected according to the weight distribution, D_t. Leaders are computed with such selected data and the chosen distance threshold as inputs. The leaders form prototypes. The learning based on NNC and prototypes is tested against complete training data. This is repeated at every iteration.

At each iteration of AdaBoost, the error in weak hypothesis "ϵ_t" and the parameter "α_t", which is the weight assigned to the each weak hypothesis, are computed. Depending on the error in classification, the weights of training dataset are re-computed at every subsequent stage. This procedure will pass through multiple iterations. At the end of the chosen number of iterations, the iteration-wise data of leaders, α_t parameters are obtained.

The validation patterns are classified based on the leaders and the α_t parameters. The classification is carried out using a Nearest Neighbour Classifier [3] [2] , which forms the weak hypothesis. The sign of linear combination of products of α_t and the weak hypotheses provides the final hypothesis. This is compared against the labels of validation patterns to obtain the classification accuracy(CA) with the validation data. The entire procedure is repeated for different distance threshold values. The set providing best CA with validation data is used to compute CA with the test data.

At *stage-2*, two possibilities emerge. The given test pattern is classified to belong to either of the two sets. In case of set-1, it is further considered as five 2-class problems, viz., 0 vs. rest of set-1, 3 vs. rest of set-1, 5 vs. rest of set-1, 6 vs. rest of set-1, 8 vs rest of set-1. Alternately, if it belonged to set-2, it is again considered as five 2-class problems, viz., 1 vs. rest of set-2, 2 vs.rest of set-2, 4 vs. rest of set-2, 7 vs rest of set-2 and 9 vs. rest of set-2. AdaBoost procedure is applied in a similar manner as discussed above.

Thus at the end of the two stages, the given test pattern is classified into one of the 10 classes.

5.1. Training and Validation Data

In case of stage-1, the training data consists of 10000 patterns of bootstrapped data belonging all 10 classes. The validation data consists of 2670 patterns. In case of stage-2, depending on the classification, the training data consists of 5000 patterns belonging to one of the sets of 5 classes and validation data consists of 1335 patterns corresponding to the same set.

Distance Threshold	Ave. No. of prototypes	Ave. CA with trg data	CA with validn data
2.0	1623	97.50	97.60
2.5	1568	97.60	97.60
3.0	1523	97.30	*97.68*
3.5	1335	97.21	97.38
4.0	1222	96.86	97.12
4.5	861	95.95	96.33
5.0	658	93.96	95.36

Table 2. Results for set-1 vs set-2

Distance Threshold	Ave. No. of prototypes	Ave. CA with trg data	CA with validn data
2.0	1288	99.19	98.58
2.5	1277	99.13	98.50
3.0	1255	99.14	98.43
3.5	1174	*99.18*	*98.43*
4.0	1006	98.90	98.28
4.5	706	97.88	98.20
5.0	536	97.28	98.28

Table 3. Results for 0 vs rest of set-1

5.2. Test Data

The case which provided best classification accuracy with the validation is considered for testing against the test data. The test data consists of 3333 patterns.

6. Results and Discussion

In the first step, the entire data is considered. The training consisting of 10 classes is labeled into two equivalent classes. AdaBoost is applied to the data. The exercise is repeated for different threshold values. The threshold values affect the number of prototypes and thereby the classification accuracy on validation data. Table-2 contains the results.

It can be observed from Table-2 that with increasing threshold the number of prototypes reduces. The exercises carried out by changing threshold values from 2.0 to 5.0 in steps of 0.1. The table contains summary of the results. Out of different threshold values, the best C.A. with validation is obtained is 97.68% for the distance threshold of 3.0. The corresponding C.A. with test data is 97.30%.

Table-3 and 4 contain, distance threshold-wise classification accuracy with validation data, average classification accuracy on training data during learning, average number of prototypes for Classes 0 and 1 respectively. Table-5 provides summary of results of the entire exercise. The table contains best classification accuracy obtained with valida-

Distance Threshold	Ave. No. of prototypes	Ave. CA with trg data	CA with validn data
2.0	1148	99.32	98.43
2.5	1054	99.42	98.65
3.0	966	99.39	98.80
3.5	827	*99.22*	*98.80*
4.0	645	98.25	98.65
4.5	417	97.40	98.05
5.0	317	91.73	98.28

Table 4. Results for 1 vs rest of set-2

Class	Distance Thrshld	Ave. num. of prototypes per iteration	CA with validation data	CA with test data
set-1 vs set-2	3.0	1523	97.68	97.30
0	3.5	1174	98.43	98.20
1	3.5	827	98.80	99.04
2	5.0	358	98.88	98.68
3	3.5	1176	95.21	95.74
4	3.5	867	96.93	96.10
5	3.5	1152	95.51	95.38
6	4.5	727	99.26	98.26
7	3.0	1060	95.66	95.92
8	3.5	1190	95.58	96.70
9	2.0	1216	94.90	94.00

Table 5. Summary of class-wise results

tion data with each class and the corresponding classification accuracy with the test data.

In the current set of exercises, in some cases, more than one threshold provided same classification accuracy. In such a scenario, largest threshold value which provides least number of prototypes is considered for reporting. This is because less number of prototypes decreases classification time.

It can be observed from results that at any stage of the iteration, the number of prototypes is much smaller than the entire data. For example, in case of set-1 vs set-2(Table-2), best classification accuracy is obtained with an average number of prototypes per iteration as 1523 against the entire training data of 10000 patterns. Also it can be observed from Table-2, that in any iteration, the classification accuracy with validation data is always higher than 95%.

6.1. Results using bench mark data

The proposed algorithm is applied on three different datasets other than the above mentioned HW data, viz., WINE, THYROID and SONAR. The data is obtained from UCI Repository [11] .

Name of Dataset	Training data size	Test data size	Number of features	Number of classes
WINE	100	78	13	3
THYROID	3772	3428	21	3
SONAR	104	104	60	2

Table 6. Details on bench mark data

Name of the Dataset	Case Description	Dist. Thrshld	Average Num. of leaders	C.A. (%)
WINE	1 vs non-1	3.0	23	98.72%
	2 vs non-2	1.5	43	93.59%
	3 vs non-3	3.7	8	98.72%
THY	1 vs non-1	2.0	261	98.83%
	2 vs non-2	3.7	104	94.31%
	3 vs non-3	3.0	156	93.84%
SONAR	0 vs non-0	4.0	65	95.19%

Table 7. Results with bench mark data

Table-6 consists of details on each of the bench mark data. Table-7 contains CA(Classification Accuracy) obtained using the current method.

The datasets are all numerical. Each feature-wise value across all the training and test patterns, are normalized to have zero mean and unit standard deviation. Euclidean dissimilarity measure is employed. The proposed algorithm is applied on the data. Classification Accuracies(CA) with NNC on WINE, THYROID and SONAR are 92.31%, 93.26% and 95.19% respectively. It should be noted that these CAs are obtained by considering entire data.

It can be observed from Table-7 that in the first two data sets of WINE and THYROID, the average CAs obtained viz., *97.01%* and *95.66%* are higher than those of NNC. The average number of leaders 25 and 174 is much less than the full data size of 100 and 3772 respectively. In case of the third data set, viz., SONAR, the CA obtained is same as NNC, but with average number of prototypes, viz., 65 is less than original data size of 104 patterns. The CA values are higher than those reported in [18].

7. Summary and Conclusions

AdaBoost was earlier used with different base learners. In the current work, NNC employing prototypes using clustering based on leaders is proposed as base learner. The clustering provides summarization of the entire data. The leaders computed on the large bootstrapped training data are used as prototypes. The current study proposed a multi-stage dichotomizer, based on AdaBoost and leader based clustering algorithm. In order to prove the proposed procedure, a large bootstrapped training data of 10000 patterns are considered. The design phase consists of data being classified into two categories at every stage of the algorithm and classification accuracy assessed using validation data. The test phase consists of considering set providing highest CA with validation data. This set is used to classify the test data of 3333 patterns. The overall average classification accuracy thus obtained is 96.85%. The values are higher than those achieved earlier [1, 18] on the same data.

In addition, the leader algorithm has a memory requirement of O(L) and is scalable. Efficient clustering algorithm is integrated in the overall fusion framework. This is being reported for the first time.

8. Future work

The selection of training examples from iteration to iteration is carried out based on the distribution D_t. The selection is based on cumulative distribution(weights) and the random numbers following Uniform Distribution. The different selection schemes that are used in Genetic Algorithms can be experimented in future. This could provide better representatives for the next iteration and reduce number of iterations.

References

[1] T. R. Babu and M. N. Murty. Comparison of genetic algorithm based prototype selection schemes. *Pattern Recognition*, 34(2):523–525, February 2001.

[2] T. Cover and P. Hart. Nearest neighbour pattern classification. *IEEE Transactions on Information Theory*, IT-13:21–27, January 1967.

[3] B. Dasarathy. *Nearest Neighbour Classification Techniques*. IEEE Computer Society Press, Los Alamitos, California, 1991.

[4] Y. Freund and R. E. Schapire. A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, 14(5):771–780, September 1999.

[5] P. E. Hart. The condensed nearest neighbour rule. *IEEE Transactions on Information Theory*, 14:515–516, May 1968.

[6] A. Jain, R. P. Duin, and J. Mao. Statistical pattern recognition: A review. *IEEE Trans. on PAMI*, 22(1):40–37, January 2000.

[7] A. Jain, M. N. Murty, and P. Flynn. Data clustering: A review. *ACM Computing Surveys*, 32(3), September 1999.

[8] L. Kaufman and P. Rousseeuw. *Finding Groups in Data - An Introduction to Cluster Analysis*. Wiley, NY, 1989.

[9] M. Kearns and L. Valiant. Learning boolean formulae or finite autotmata is as hard as factoring. Technical Report TR-14-88, Harvard University Aiken Computation Laboratory, 1988.

[10] M. Kearns and L. Valiant. Cryptographic limitations on learning boolean formulae and finite automata. *Journal of Association for computing Machinery*, 41(1):67–95, January 1994.

[11] P. Murphy. *UCI Repository of Machine Learning Databases*. http://www.ics.uci.edu /mlearn /MLRepository.html, Department of Information and Computer Science, University of California, Irvine, CA, 1994.

[12] R. E. Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, February 1990.

[13] R. E. Schapire. Theoretical views of boosting and applications. In *Proceedings of Algorithmic Learning Theory*, 1999.

[14] R. E. Schapire. The boosting approach to machine learning: An overview. In *MSRI Workshop on Nonlinear Estimation and Classification*, 2002.

[15] H. Spath. *Cluster Analysis - Algorithms for Data Reduction and Classification of Objects*. Ellis Horwood Limited, West Sussex, UK, 1980.

[16] L. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, November 1984.

[17] S. Viaenne, R. A. Darrig, and G. Dedene. A case study of applying boosting naive bayes to claim fraud diagnosis. *IEEE Transactions on Knowledge and Data Engineering*, 16(5):612–620, May 2004.

[18] P. Viswanath, M. N. Murty, and S. Bhatnagar. Fusion of multiple approximate nearest neighbour classifiers for fast and efficient classification. *Accepted for publication in Information Fusion*, 2004.

IEEE COMPUTER SOCIETY