Fast Multistage Algorithm for K-NN Classifiers

I. Soraluze, C. Rodriguez, F. Boto, and A. Cortes

Computer Architecture and Technology Department, Computer Science Faculty, UPV/EHU,
Aptdo. 649, 20080 San Sebastian, Spain
acrneura@si.ehu.es

Abstract. In this paper we present a way to reduce the computational cost of k-NN classifiers without losing classification power. Hierarchical or multistage classifiers have been built with this purpose. These classifiers are designed putting incrementally trained classifiers into a hierarchy and using rejection techniques in all the levels of the hierarchy apart from the last. Results are presented for different benchmark data sets: some standard data sets taken from the UCI Repository and the Statlog Project, and NIST Special Databases (digits and upper-case and lower-case letters). In all the cases a computational cost reduction is obtained maintaining the recognition rate of the best individual classifier obtained.

1 Introduction

Pattern classification based on the *k*-nearest-neighbors (*k*-NN) [1], [2] approach has been proven to perform well in pattern classification on many domains. It is well known that the main drawback of this kind of classifier in practice is the computational cost demand. In order to find the *k* nearest patterns, a dissimilarity measure between the test sample and a large number of samples in the training set is computed making this approach very computational. One method for solving this problem is to reduce the large data set to a small and representative one using condensing algorithms [4]. The objective of selecting a subset of the original training data is the computational efficiency in the classification phase, or/and making the resulting classification and generalization more reliable. Other kinds of methods try to reduce the data set by generating artificial prototypes summarizing representative characteristics of similar instances [10]. In many works has been proven that hierarchical systems are good too to reduce the computational cost in classifiers with a large data set. Besides, in recent years there is an increasing interest in schemes that combine multiple classifiers [5].

In this paper, multistage classifiers based on *k*-NN classifiers are proposed for the reduction of the computational cost. The multistage classifiers are built with *l* levels of individual *k*-NN classifiers trained incrementally and using rejection techniques in the levels. Each of the individual classifiers is built with different characteristics and the computational cost of classifying a pattern is proportional to the level where the pattern has been recognized. When a pattern has not enough classification certainty in

A. Sanfeliu and J. Ruiz-Shulcloper (Eds.): CIARP 2003, LNCS 2905, pp. 448–455, 2003.

a level of the hierarchy goes to the next level where a more complex classifier will try to classify it. A lot of the patterns are classified in the lowest levels of the hierarchy, where the computational cost is low because the simplest classifiers have been put there.

We have used different benchmark data sets to validate our system. Three data sets have been selected from the UCI Repository and the Statlog project [6]. The results obtained with these data sets are put together with previously obtained results with characters of the NIST Special Databases [12]: digits, upper-case and lower-case letters.

The paper has the following structure. In Section 2 a formal description of the multistage system is presented. In Section 3 we describe the particular classifiers used and the experimental results obtained. Finally in section 4 the conclusions are presented.

2 Formal Description of the System

We are going to give now some definitions and the notation used throughout the paper. After that the algorithms used in the training and recognition process to build the multistage system are presented.

A pattern is represented by x_i and is identified by the orderly pair (r_i,c_i), where $r_i=(r_{i,1}, r_{i,2},.., r_{i,d})$ are the d feature values that represent the pattern x_i and c_i is the class label assigned to the pattern.

The Training Set formed for all the training patterns, ***TS***, is defined by $TS = \{x_i \mid x_i = (r_i, c_i)\}$ and the Reduced Set, ***RS***, is a subset of ***TS***, obtained with a learning or training algorithm.

The following operators will appear in next sections:

- $C(x_i) = c_i$ is the class of the pattern x_i.
- $d(x_i, x_j) = |r_i - r_j|$ is the Euclidean distance between two feature vectors (r_i, r_j) associated to the patterns x_i and x_j respectively.
- $D(x) = \min\left(d(x, x_j)\right) \forall x_j \in RS \wedge C(x_j) = C(x)$ are the nearest patterns to x belonging to the same class.
- $\neg D(x) = \min\left(d(x, x_j)\right) \forall x_j \in RS \wedge C(x_j) \neq C(x)$ are the nearest patterns to x belonging to a different class. $\neg D(x) = \infty \,|\, C(x_j) = C(x)\ \forall x_j \in RS$
- $NN_k(x,S) = (x_1, x_2, ..., x_k) \,|\, x_i \in S \quad 1 \le i \le k$ is the distance-ordered set of the k-nearest patterns to x in the set S.

2.1 Classic Training Algorithm (*TA*)

The training algorithm shown in Fig.1 generates a reduced data set *RS* taking the training set *TS*. Two rules, specified as (1) and (2) in the figure, have been applied to obtain the reduced set. The reduction obtained depends on the parameter *tt* (*training*

threshold). The threshold can take values between 0 and infinite. If the threshold takes value 0 the maximum reduction is obtained and this algorithm becomes into Hart's algorithm [4]. If the value assigned to this parameter is infinite, there is no reduction and *RS* and *TS* are the same data set.

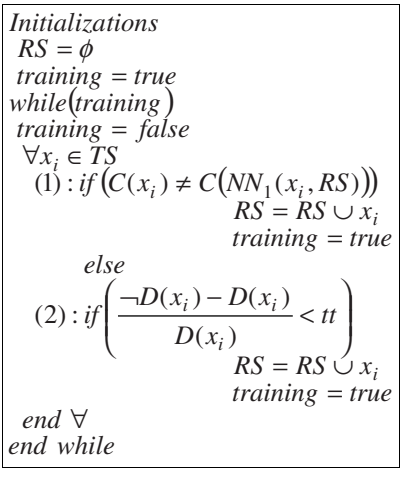

Fig. 1. Training Algorithm (*TA*)

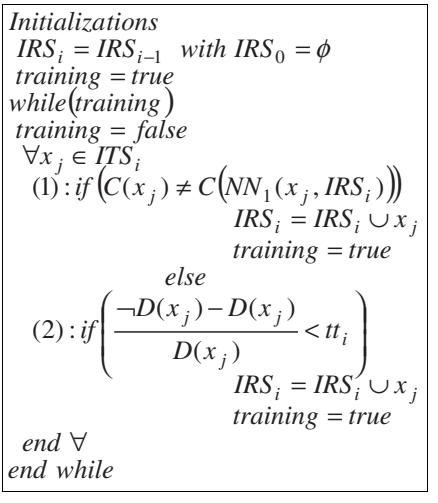

Fig. 2. Incremental Training Algorithm (*ITA*)

2.2 Incremental Training Algorithm (*ITA*)

The different classifiers used to build a hierarchy have been trained using an incremental training algorithm, which is a modification of the algorithm shown in previous section. The training set *TS* is divided in *s* subsets:

$$TS = \bigcup_{i=1}^{s} Ts_i \quad Ts_i \cap Ts_j = \phi \quad \forall i, j \mid i \neq j. \tag{1}$$

The incremental training set *i*, $\boldsymbol{ITS_i}$, is defined as following:

$$ITS_i = \bigcup_{j=1}^{i} Ts_j \mid ITS_i = ITS_{i-1} \cup TS_i \text{ with } ITS_0 = \phi. \tag{2}$$

The cardinal of ITS_i follows a exponential and uniform distribution.

The *I*ncremental *T*raining *A*lgorithm (***ITA***), creates an incremental reduced set, $\boldsymbol{IRS_i}$, taking the incremental training set, ITS_i, and the incremental reduced set *i-1*, IRS_{i-1}, by applying the same two rules shown previously in Fig. 1. The incremental algorithm is shown in Fig. 2.

Having used this algorithm, we can use the following property in the recognition process:

$$\forall i > 0 \; IRS_{i-1} \subseteq IRS_i \text{ with } IRS_0 = \varnothing \; . \tag{3}$$

The incremental training algorithm reduces the computational cost of the training process because the training starts from an already built classifier and only some new patterns are learnt to create the new classifier.

2.3 Multistage Recognition Algorithm (*MRA*)

The multistage classification process is obtained using classifiers trained in an incremental way and putting them into a hierarchy ordered according to the number of patterns they have. Having a pattern x, we will define a confidence value, $CV_{i,ki}$, used to decide if the pattern x has enough classification certainty to be classified in the level i of the hierarchy (this rule is known as k,l-Nearest Neighbor [1]). The expression can be seen in Equation 4.

$$CV_{i,k_i}(x) = \frac{\max\left(CPC_{c,k_i}\right)}{k_i}. \tag{4}$$

k_i is the number of nearest neighbors taken in level i and $CPC_{c,ki}$ determines how many of these patterns belong to class c. The expression is shown in Equation 5.

$$CPC_{c,k}(x) = \left|\left\{x_j \mid C(x_j) = c \wedge x_j \in NN_k(x, IRS_i)\right\}\right|. \tag{5}$$

The multistage recognition algorithm is shown in Fig. 3. This recurrent algorithm starts with the call *MRA(x,1)* and represents the classification process of a pattern x of the test set. This algorithm reduces the computational cost in the classification process, because the discriminating capacity of the hierarchical classifier is adapted to the recognition difficulty of each pattern.

```
MRA(x, i)
{
  if (i = max_level) return WR_{i,k}(x);
  else
    if (CV_{i,k_i}(x) > rt_i) return c
    else return (MRA(x, i+1));
}
max_level is the number of levels
rt_i is the rejection threshold associated to the level i.
```

Fig. 3. Multistage Recognition Algorithm

When the pattern reaches to the last level of the hierarchy we use the Weighted rule $WR_{i,k}(x)$ [1] modified (Equation 6) to classify it.

$$WR_{i,k}(x) = \max_c\left(D_{c,k}(x)\right) \tag{6}$$
$$with\ D_{c,k}(x) = \sum_{j=1}^{k} d(x_j, x_k)^2 \mid x_j, x_k \in NN_k(x, IRS_i) \wedge C(x_j) = c.$$

2.4 Multistage Recognition Algorithm with Active Memory (*MRAMA*)

In order to reduce the computational cost of the previous algorithm, a modification has been introduced. This new algorithm keeps memory about what happened in the previous levels [3] of the hierarchy and uses it when a pattern is going to be classified in a later level. This is a memory-based method or a method with "*active memory*". Particularly, the information known about the previous levels is the set of classes found among the *k* nearest patterns of that level, referred as $CS_{i,k}$. The expression for this set is in Equation 7.

$$CS_{i,k_i}(x) = \left\{ C(x_j) \mid x_j \in NN_{k_i}(x, IRS_i) \wedge C(x_j) \in CS_{i-1,k_{i-1}} \right\} \forall i > 0 \quad (7)$$
where CS_{0,k_0} *contains all the possible classes.*

The description of the new algorithm is shown in Fig. 4. The algorithm starts with the call *MRAMA(x,1,*$CS_{0,k0}$*)*. The only difference between the expressions $MAWR_{i,k}(x)$ and $MACV_{i,ki}(x)$ and the presented in previous section $WR_{i,k}(x)$ and $CV_{i,ki}(x)$ is that in the new expressions the k_i patterns are searched only among the patterns of the classes in the set $CS_{i,k}$, which reduces the cost.

```
MRAMA(x, i, CS_{i-1,k_{i-1}})
{
  if (i = max_level) return MAWR_{i,k}(x);
  else
    if (MACV_{i,k_i}(x) > rt_i) return c
    else return (MRAMA(x, i+1, CS_{i,k_i}));
 }
max_level is the number of levels
rt_i is the rejection threshold associated to the level i.
```

Fig. 4. Multistage Recognition Algorithm with Active Memory

The computational cost associated to recognize a pattern *x* in the level *i* using this algorithm and incremental training is defined with the following expression:

$$RCC_i(x) = RCC_{i-1}(x) + \left| \left\{ x_j \mid x_j \in \{IRS_i - IRS_{i-1}\} \wedge C(x_j) \in CS_{i-1,k_{i-1}}(x) \right\} \right| \quad (8)$$
$$\forall i > 1 \wedge RCC_1(x) = |IRS_1|.$$

The average computational cost of the hierarchical or multistage classifier with *L* levels taking a test pattern set *X* is defined in Equation 9

$$RCC(X) = 1/|X| \sum_{j=1}^{|X|} RCC_{L_{x_j}}(x_j) \quad (9)$$
Where L_{x_j} *is the level where the pattern* x_j *has been recognized .*

3 Experimental Results

The experimentation has been carried out with different data sets: Statlog LandSat Satellite (Satimage), Statlog Shuttle (Shuttle) and UCI Letter Recognition database (Letter), taken from the UCI Repository and the Statlog project. Table 1 presents the characteristics of these benchmark data sets.

Table 1. Benchmark data sets used

	TS	Test Set	Classes	Features
Shuttle	43,500	14,500	7	9
Letters	15,600	4,400	26	16
Satimage	4,435	2,000	7	36

The first experimentation has been carried out for individual classifiers using the training algorithm *TA* and the $WR_{i,k}(x)$ rule. Different *training thresholds* and *TS* sizes have been proven in order to find the classifier with best recognition rate. For Satimage data sets have been defined 3 *TS* of 1,200, 2,400 and 4435 patterns respectively. For Shuttle data sets 5 *TS* have been used with 2,100, 4,200, 8,400, 16,800 and 43,500 patterns. Finally for Letter data set 4 *TS* have been defined with 2,600, 5,200, 10,400 and 15,600 patterns. For all the data set a bigger *TS* contains the patterns of the smaller ones. We can see in Table 2 the best results obtained for these three data sets and the characteristics of the classifiers that provide these results. The recognition rate achieved is similar to the presented by other authors [7], [11]. Together with the results of these three benchmarks, the results previously obtained with characters of NIST Special Databases, digits [8] and letters [9] are presented. We can see that the best results for all the data sets are obtained with a reduced set of the bigger *TS* used. Selecting some patterns a computational cost reduction is obtained.

Table 2. Results obtained with individual classifiers

	TS	RS	Recognition	tt	[7]	[11]
NIST, digits	160,000	23,493	99.50	0.5%		
NIST, upper-case	60,194	24,348	95.44	0.5%		
NIST, lower-case	49,842	24,503	88.34	0.5%		
Shuttle	43,500	567	99.88	0.75%	99.86	99.94
Letters	15,600	8,406	95.73	0.75%	96.6	68.30
Satimage	4,435	1,924	91.15	0.25%	90.15	86.25

The best recognition rate presented can be obtained with less computational cost if we use the multistage classifiers with incremental training and active memory. We have built hierarchies with different number of levels for each data set depending on the number of training subsets we have. The *TS* subsets created for each data set have been trained incrementally now, using the previous trained subset as described in the section 2.2 and put into a hierarchy ordered by the number of patterns they have. The *tt* used to obtain *IRS* has been 0 except for the last *IRS*. The last *TS* (the one with more patterns) has been trained using the value 0 for the *tt*, and using the *tt* which provides

the best recognition rate. The incremental training helps to reduce the computational cost in the recognition process because in each level of the hierarchy only the new patterns of the classifier need to be check.

These classifiers have been put into a hierarchy and applying the algorithm presented in section 2.4 and important computational cost reduction has been obtained maintaining the hit rate presented before (The computational cost is calculated with the expression in Equation 8, and compared with the computational cost of the individual classifier in the last level, which gives the same recognition rate). These results are shown in Table 3. In this case too, results obtained with NIST Databases are presented in order to make a comparison.

Table 3. Results obtained with multistage and incremental classifiers

	Levels	Recognition	Computational cost / Speed-up
NIST digits	12	99.48	957.69 / 167
NIST upper-case	12	95.44	744.23 / 80.88
NIST lower-case	12	88.12	1145 / 43.53
Shuttle	6	99.88	329 / 132.2
Letters	5	95.54	4070 / 3.82
Satimage	2	91.5	1093 / 4.05

We can observe that although there is always a computational cost reduction (speed-up) applying the hierarchical recognition algorithm, when the data set contains few patterns the reduction is not very high. In problems with large *TS* or in easy problems, the computational cost reduction is higher.

Besides, for the Satimage data set we have selected a classifier of two levels although we primarily took a classifier with 4 levels as described before. The first two classifiers in this case had very few patterns so they rejected all the patterns and selected a class set not correct that increment the error rate in the next levels. For this reason these first classifiers have been remove from the hierarchy, building a hierarchy of only 2 levels.

4 Conclusions

It is well known that *k*-NN classifier provides good recognition results on pattern recognition systems of different domains. In this paper we have presented a multistage incremental algorithm that reduces the computational cost associated to *k*-NN classifiers, solving one of the most important problems of these algorithms.

The multistage classifier is built with different *k*-NN classifiers (with different number of patterns used in the classification) trained in an incremental way. The discriminating capacity of the classifiers is adapted to the classification needs of each pattern in the classification process; the easiest patterns are classified in the first levels of the hierarchy, with low computational cost. Only the most difficult patterns reach the last level of the hierarchy where the classifiers with more patterns have been put.

The experimental results have been presented for different benchmark data set. We have observed that although always a computational cost reduction is obtained main-

taining the recognition rate of the best classifiers, the speed-up is higher when large training sets are taken. The speed-up obtained with data sets as NIST digits databases where we have taken 160,000 patterns for training is of 167 times whereas the reduction for data sets as Satimage with 4435 training patterns is of 4.05 times. In all the cases a good recognition rate is maintained.

Acknowledgments. This work was supported in part by the Spanish Government (CICYT project number TIC2000-0389). We also want to thank the collaboration of the company ADHOC Synectic Systems, S.A.

References

1. Dasarathy, B.V.: "Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques". Ed: IEEE Computer Society Press (1991)
2. Devroye, L., Györfi, L. and Lugosi, G.: "A Probabilistic Theory of Pattern Recognition". Ed. Springer-Verlag, New York, (1996)
3. Giusti, N., Masulli, F., Sperduti, A.: "Theoretical and Experimental Analysis of a Two-Stage System for Classification". IEEE Trans. PAMI, Vol. 24 Nº 7 pp. 893–905 (2002)
4. Hart, P.E., "The Condensed Nearest Neighbour Rule". IEEE Transactions on Information Theory, Vol. 14, pp. 515–516, (1968)
5. Ho, T.K, Hull, J. J. and Srihari, S.N.: "Decision Combination in Multiple Classifier Systems". IEEE Trans. PAMI, Vol. 16 Nº 1 pp. 66–75 (1994)
6. Murphy, P.M. and Aha, D.W.: 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 (1994)
7. Paredes, R., Vidal, E.: "A class-dependent weighted dissimilarity measure for nearest neighbor classification problems". Pattern Recognition Letters, Vol. 21 pp. 1027–1036 (2000)
8. Soraluze, I., Rodriguez, C., Boto, F., Perez, A.: "Multidimensional Multistage K-NN classifiers for handwritten digit recognition" Proceedings of eighth IWFHR, Niagara-on-the-lake (Ontario), Canada, pp 19–23 (2002)
9. Soraluze, I., Rodriguez, C., Boto, F., Perez, A.: "An Incremental and Hierarchical k-NN classifier for Handwritten characters" 16th International conference on Pattern Recognition Quebec, Canada (2002)
10. Vijaya Saradhi, V., Narasimha Murty, M.: "Bootstrapping for efficient handwritten digit recognition". Pattern recognition, Vol. 34 pp. 1047–1056 (2001)
11. Murthy, S.K. and Salzberg, S.: A system for the induction of oblique decision trees. Journal of Artificial Intelligence Research 2, 133 (1994)
12. Wilkinson, R.A., Geist, J., Janet, S., Groter, P., Burges, R., Creecy, R., Hammond, B., J. Hull, N. Larsen, T. Vogl and C. Wilson, *First Census Optical Character Recognition System Conference*. National Institute of Standards and Technology (NIST) 1992