15th European Signal Processing Conference (EUSIPCO 2007), Poznan, Poland, September 3-7, 2007, copyright by EURASIP

A NEW ALGORITHM FOR FAST SEARCH OF THE K NEAREST PATTERNS

R. Gil-Pita, M. Rosa-Zurera, R. Vicen-Bueno and F. López Ferreras

Department of Signal Theory and Communications, University of Alcalá
Escuela Politécnica Superior, 28805, Alcalá de Henares, Spain
phone: + 34 918856698, fax: + 34 918856699, email: roberto.gil@uah.es

ABSTRACT

The computational cost associated to the k-nearest neighbor classifier depends on the amount of available patterns, which makes this method impractical in many real-time applications. This fact makes interesting the study of fast algorithms for finding the k-nearest patterns, like, for example, the kLAESA algorithm. In this paper we propose a novel algorithm for finding the k-nearest patterns, denominated k-tuned approximating and eliminating search algorithm (kTAESA). The algorithm is used to implement kNN classifiers, which are applied to three databases from the UCI machine learning benchmark repository. Results are compared with those achieved by the exhaustive search, the kAESA and the kLAESA algorithms, in terms of number of distances to evaluate, number of simple operations (sums, comparisons and products) needed to classify each pattern, and amount of required memory. Results demonstrate the best performance of the proposal, mainly when the number of operations is considered.

1. INTRODUCTION

The k-nearest neighbor (kNN) method is a distance-based classification technique that has been applied in many different real classification problems. It finds the k patterns from a set of pre-classified patterns (denominated training patterns) with less distance to the pattern to classify (denominated input pattern), and uses the classes of these k patterns. So, the decision is taken by majority voting of the classes of these k-nearest patterns. The parameter k is specified by the user, and it determines how many patterns are considered to determine the class of the input pattern.

In a first approach, the search procedure requires the calculus of the distance from the input pattern to each pattern in the training set. This exhaustive search is associated to a high amount of operations, almost in high-dimensioned problems with large training sets. In these cases, the application of search algorithms to reduce the number of evaluated distances becomes interesting.

In the literature, there are many papers dealing with the search of the k-nearest patterns. Data can be structured in trees [1] which can be useful to reduce the search space and, therefore, the number of required distances. Other alternatives determine the eigenvectors of the training set, and project the input vector over the main eigenvector, avoiding the calculus of some distances [2]. At last, there is a group of techniques that, applying triangular inequalities, reduce the number of evaluated distances. One of the main methods of this group is the k-linear approximating and eliminating search algorithm (kLAESA) [3], which has a considerably low associated computational complexity.

In this paper we propose a novel search algorithm based on triangular inequalities, denominated k-tuned approximating and eliminating search algorithm (kTAESA), as it refines the approximation of the nearest pattern by searching over the nearest patterns to the candidate. Results obtained by this proposed method are compared to those achieved by the exhaustive search, the kAESA and the kLAESA methods, in terms of computational cost evaluated considering the average number of calculated distances, the number of simple operations needed to classify one test pattern, and the required amount of memory. We study the application of the different algorithms in three different problems, extracted from the UCI machine learning benchmark repository: the breast cancer detection, the diabetes problem and the letter recognition.

2. MATERIALS AND METHODS

This section includes a description of the main characteristics of the databases (size, dimension, etc) and the theoretical basis of the algorithms and methods used in the paper. It firstly describes the three databases selected from the UCI machine learning benchmark repository: the breast cancer database, the diabetes database, and the letter database. In a second part, a brief theoretical analysis of the relevant characteristics of kLAESA implementation of the kNN classifier is tackled. These descriptions allow to better understand the proposed kTAESA method.

2.1 Description of the databases

Each database has been divided in three sets: the training set, which is composed of those patterns that are used to design the kNN classifier; the validation set, which is used to obtain the value of k in each problem, and to select the values of the user specific parameters in the different fast implementations; and the test set, which is used to evaluate the average computational complexity of the implementation of the kNN algorithm, measured in terms of both average number of distances to evaluate for each test pattern and average number of simple operations (sums, products and comparisons). This third set has not been used during the design of the classifiers.

Three UCI databases have been selected in order to evaluate the performance of the fast search algorithms in three different problems. These three UCI databases and the data preparation techniques are identical to those used in [4].

The first UCI database used in the experiments is the breast cancer database. It was obtained at the University of Wisconsin Hospitals, Madison. This is a binary problem ($C = 2$, where C is the number of classes), and its purpose is to be able to classify a tumor as either benign or malignant,

This work has been partially funded by the Comunidad de Madrid/Universidad de Alcal (CCG06-UAH/TIC-0378) and the Spanish Ministry of Education and Science (TEC2006-13883-C04-04/TCM)

based on cell descriptions gathered by microscopic examination. The data set contains $L = 9$ attributes and 699 examples of which 458 are benign examples and 241 are malignant examples. The input attributes are scaled between 0 and 1. In order to generate the different sets, the first 349 examples are used for the training set, the following 175 examples for the validation set, and the final 175 examples for the test set.

The second database, the Diabetes data set, was constructed by constrained selection from a larger database held by the National Institute of Diabetes and Digestive and Kidney Diseases. All patients represented in this data set are females of at least 21 years old and of Pima Indian heritage living near Phoenix, AZ. In this second database, the problem consists in predicting whether a patient would test positive for diabetes according to World Health Organization criteria, given a number of physiological measurements and medical test results. So, this is a two class problem, and therefore C is equal to 2. There are 500 examples of class $C = 1$ and 268 of class $C = 2$. There are $L = 8$ attributes for each example, and, like in the case of the Cancer problem, the input attributes are scaled between 0 and 1. In this case, the first 384 examples are used to generate the training set, the following 192 examples to generate the validation set, and the last 192 examples to generate the test set.

The last used database is the Letter Recognition database. This is a multi-class problem with $C = 26$ different classes (associated to an alphabet with 26 letters). Each pattern contains 16 numerical characteristics of a written letter of the alphabet, and therefore the dimension of the input space is $L = 16$. Again, the input attributes are scaled between 0 and 1. The database contains 20000 patterns. The first 16000 examples are used to generate the training set, the following 2000 examples to generate the validation set, and the last 2000 examples to generate the test set.

2.2 kNN description

The kNN method is a statistical technique based on the estimation of the posterior probabilities $p(H_i|\mathbf{x})$ of the hypothesis H_i, conditioned to the input patter $\mathbf{x}$. Taking a volume around the input pattern $\mathbf{x}$ that encompasses k patters of the training set and k_i patters of hypothesis H_i so that $k_i \leq k$, then the posterior probability $p(H_i|\mathbf{x})$ can be approached using equation (1) [5].

$$p(H_i|\mathbf{x}) \simeq \frac{k_i}{k} \tag{1}$$

So, applying a *Maximum A Posteriori* (MAP) criterion, the selected hypothesis H_i is the one that maximizes its posterior probability $p(H_i|\mathbf{x})$ and, therefore, maximizes k_i. The kNN method fixes the number k of patterns in the volume, being these k patterns the k nearest patterns to the observation point, and the decision is taken by evaluating the values of k_i, $i = 1,\ldots,C$, and selecting the class which obtains a highest k_i value. This is equivalent to the application of a majority voting criterion to the classes of the k nearest patterns.

The distance criterion and, therefore, the selected k nearest patterns depend on the selected metric, which must be fixed in the algorithm. In the literature, the use of the Euclidean distance is quite common, and, therefore, it has been selected for our experiments in the paper.

The value of k is also a parameter of the kNN classifier, that controls the smoothness of the implemented solution.

Table 1: Relevant characteristics of the databases used in the paper.

	Breast	Diabetes	Letter
kNN error probability	1.14%	21.88%	4.85%
Selected k value	3	27	4
Training set size (N)	349	384	16000
Input vector length (L)	9	8	16
Number of classes (C)	2	2	26

The best k value depends on the characteristics of the selected database, and, therefore, it must be fixed in the design stage. In order to guarantee a good generalization of the obtained results, we select its value using the validation set. So, the classification error over the validation set is evaluated for several k values, and the value that efforts the lowest error is selected. The characteristics of the three databases used in this paper are included in table 1, showing the error probability obtained by a kNN classifier, the value of k selected by the validation set, the training set size N, the vector length L, and the number of classes C.

2.3 kLAESA: a fast search algorithm

The implementation of a kNN classifier involves a search procedure in which the k patterns with less distance must be determined. In many cases, this search procedure supposes a high amount of computational requirements, which makes the kNN method not suitable for many real-time applications. To tackle this problem, there are some strategies that try to reduce the required number of distances to evaluate in the search process, by applying some properties of the distance measurement in the vectorial space of the data.

One of the main algorithms of this group of methods is the approximating and eliminating search algorithm (AESA) [6], in which the distance from the training patterns to each other are pre-calculated, and, applying triangular inequalities, a minimum level (lower bound) to the distance from the input pattern to every pattern in the training set is established. So, in many cases it is not necessary to evaluate all the distances from the input pattern to every pattern in the database, and therefore the number of evaluated distances is reduced. On the other hand, this reduction in the number of distances implies a high increment in memory requirements and operations for calculating and updating the lower bounds. This fact makes this method only suitable for applications with a very high computational cost associated to the calculus of distances, and with not very large training sets.

One variant of this method is the linear approximating and eliminating search algorithm (LAESA) [7], and its k searching version (kLAESA) [3]. This variant reduces the required amount of memory and operations for evaluating the bounds but with an increment in the number of evaluated distances. In a preprocessing stage, a subset of Q prototypes $\mathbf{y}_{q1}$, $q = 1,\ldots,Q$ is selected, which are used to sort the remaining training patterns. So, Q lists of sorted patterns are generated, and for the q-th list, the distance from its k-th pattern satisfaces $D(\mathbf{y}_{qk},\mathbf{y}_{q1}) \leq D(\mathbf{y}_{q(k+n)},\mathbf{y}_{q1})$ for every $n \in \mathbf{N}$.

So, the first stage of the algorithm consist in determining

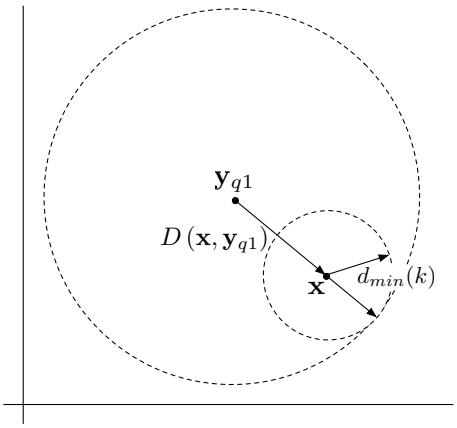

Figure 1: Two-dimensional example of the application of the triangular inequality in the kLAESA method.

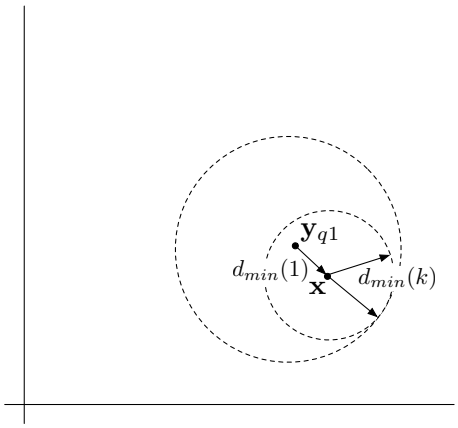

Figure 2: Two-dimensional example of the application of the triangular inequality in the kTAESA method.

the distance $D(\mathbf{x},\mathbf{y}_{q1})$ from the input pattern $\mathbf{x}$ to each prototype $\mathbf{y}_{q1}$, and the input pattern is assigned to the list which prototype is nearest. Then, the distances from the input pattern $\mathbf{x}$ to the sorted training patterns $\mathbf{y}_{qj}$,$j = 2,\ldots,N-Q+1$ are evaluated. This process can be early stopped applying triangular inequalities of the data (2).

$$D(\mathbf{x},\mathbf{y}_{q(k+n)}) \geq D(\mathbf{y}_{qk},\mathbf{y}_{q1}) - D(\mathbf{x},\mathbf{y}_{q1}) \tag{2}$$

Being $d_{min}(k)$ the distance to the k-th nearest pattern to the input pattern from the analyzed, there is no need to evaluate the distance to a training pattern if we can guarantee that it is greater than $d_{min}(k)$, and, therefore, we can early stop the search process when $d_{min}(k) < D(\mathbf{y}_{qk},\mathbf{y}_{q1}) - D(\mathbf{x},\mathbf{y}_{q1})$.

Figure 1 shows an example of the search procedure for an illustrative two-dimensional problem. Being $\mathbf{x}$ the input pattern to be classified and being $d_{min}(k)$ the distance to the k-th nearest pattern in a given step of the algorithm, then the small circle shows the region of the space in which there could be a k-nearest pattern and, therefore, the region of the space to be searched for candidates. So, let's consider $\mathbf{y}_{q1}$ is the prototype of the selected list, then the small circle is included in the large circle, and it determines the search region. Taking into account that the patterns in the list are sorted in function of the distance to the prototype, then we must just evaluate the distance to those patterns that are included in the large circle, and therefore we can early stop the process, reducing the number of required distances to evaluate.

The procedure to determine the prototypes can be implemented by a clustering algorithm. The number of needed distances depends on the data distribution and on the number of prototypes. A high value of Q makes necessary the evaluation of a high number of distances in the first stage of the implementation, but usually forces $D(\mathbf{x},\mathbf{y}_{q1})$ to be low. On the contrary, a low value of Q implies a low number of minimum distances calculated for each pattern, but a higher $D(\mathbf{x},\mathbf{y}_{q1})$ value.

So, the number of lists (Q) is a user-specific parameter of the algorithm. In this paper we vary the value of Q and we use the k-means algorithm to determine the prototypes. We repeat the experiment several times, and we select the value of Q and the selected prototypes that minimize the average number of distances needed to classify the validation patterns.

3. DESCRIPTION OF THE PROPOSAL

In this section we describe the proposed implementation of a kNN classifier, that we denominate k-tuned approximating and eliminating search algorithm (kTAESA). In order to implement this algorithm, it is necessary to evaluate and record the distances from the N training patterns each other. So, these distances are used to generate N lists of indexes, that determine the position of each training pattern sorted in function of the distance to each pattern. The value of these distances are also used to early stop the search process.

The main issue of the algorithm is that, on the contrary that with the kLAESA algorithm, the input pattern is not assigned to a list of sorted patterns at the beginning of the process, but it changes during the search procedure to the list which first pattern is nearer to the input pattern. So, the distance from the input pattern to the selected prototype $D(\mathbf{x},\mathbf{y}_{q1})$ is always equal to the distance to the nearest pattern from those evaluated ($d_{min}(1)$), and, therefore, the early stopping condition is improved. Replacing $D(\mathbf{x},\mathbf{y}_{q1})$ by $d_{min}(1)$ in equation (2) and rearranging, we obtain the stopping condition given by (3).

$$d_{min}(1) + d_{min}(k) \leq D(\mathbf{y}_{qk},\mathbf{y}_{q1}) \tag{3}$$

The main disadvantage of this method is that, when the value of $d_{min}(1)$ changes, it is necessary to change the list and start in the first pattern of the new list. To avoid the repetition of the calculus of the distance to the same training pattern, the use o a set of flags that determine if a pattern has been evaluated or not is needed.

Figure 2 shows the application of the new stopping criterion in a two-dimensional space. The region of the candidates to be nearer than $d_{min}(k)$ (small circle) is included in the region of the points which distance to $\mathbf{y}_{q1}$ is lower than $d_{min}(1) + d_{min}(k)$. Comparing this figure with figure 1, we demonstrate that the search region is smaller in the proposed kTAESA method, and, therefore, the required number of distances is reduced.

The procedure to implement the proposed algorithm is described as follows:

1. PREPROCESSING: The process starts by measuring the distance from the input pattern to k patterns of the training set. They are considered as the first k candidates to be

Table 2: Average number of measured distances needed to classify one pattern for the different databases considered in the paper.

	Breast	Diabetes	Cancer
Exhaustive search	349	384	16000
kAESA	24	104	203
kLAESA	103	251	2023
kTAESA	71	193	296

Table 3: Thousand of simple operations (sums, products and comparisons) needed to classify one pattern for the different databases considered in the paper.

	Breast	Diabetes	Cancer
Exhaustive search	10.1	10.0	816.0
kAESA	199.0	10.6	2902.8
kLAESA	3.1	9.3	103.9
kTAESA	3.1	8.9	33.4

the k-th nearest neighbors. The index of the list q starts equal to the pattern with less distance, and the index i of the list starts with value equal to 2 (the first element is prototype of the list, and its distance has already been calculated).

2. It is necessary to determine if the i-th pattern of the q-th list has been evaluated yet. In that case, the index i is incremented and the algorithm continue in step 6.
3. The distance from the prototype to the i-th pattern to evaluate of the q-th list is compared to $d_{min}(1) + d_{min}(k)$. If it is larger, then the search procedure is stopped and the classes of the k candidates are used to classify the input pattern by majority voting. If not, then it is necessary to evaluate the distance from the input pattern to the corresponding i-th pattern of the q-th list.
4. If this distance is lower than $d_{min}(k)$, then this i-th pattern substitutes the k-th candidate.
5. If this distance is lower than $d_{min}(1)$, then the list index q is updated so that the new prototype is the new nearest pattern, and $i = 2$. If not, then i is incremented, in order to evaluate the next pattern of the list.
6. If i is larger than the training set size, then the searching process is stopped, and the input pattern is classified in function of the classes of the k nearest candidates. If not, the process iterates in step 2.

This method guarantees that the patterns to evaluate are those nearest to the best candidate. It shows a strong reduction in the number of operations when compared to other methods like, for example, the kLAESA, but on the other hand it requires the calculus and storage of the distances of all the training patterns to each others. For example, considering the distances stored in 4 bytes float numbers, and the indexes in 2 bytes integers, then it is necessary to have available $6N^2$ bytes of memory in order to store the distances and the indexes from each N pattern to each N pattern. This fact can suppose a drawback in those environments with very large training sets.

4. RESULTS

This section includes the results achieved by the proposed fast kNN method, the kTAESA, compared to the exhaustive search procedure, the kAESA and the kLAESA. The comparisons are done considering the three different application problems, described in subsection 2.1. Tables 2, 3 and 4 show the results obtained with the different databases, in terms of average number of distances to evaluate in order to classify one pattern, number of simple operations and required amount of memory, respectively.

Table 4: Required memory (Kilobytes) for the different databases considered in the paper.

	Breast	Diabetes	Cancer
Exhaustive search	2	13	379
kAESA	121	589	250379
kLAESA	16	64	5020
kTAESA	240	877	750379

From the obtained results we can extract some analysis:

- The exhaustive search for implementing the kNN classifier requires the highest number of distances to evaluate but the lowest amount of memory in all the cases considered in the paper. Both the number of simple operations and the memory requirements increase linearly with the training set size.
- The kAESA algorithm achieves the highest reduction in terms of number of distances to evaluate, but it has the highest associated number of simple operations. In this algorithm the required memory increases quadratically with the training set size. Due to the excessive amount of comparisons and sums in the eliminating and approximating stages, the number of operations is higher than the direct implementation of the kNN with an exhaustive search. Notice that in this paper we consider that the multiplications are processed in a timing cycle, and therefore they have the same associated computational cost as a sum or a comparison. In other cases in which the multiplications have a cost higher than sums or comparisons, then the kAESA method can produce an improvement in the computational cost, when compared to the exhaustive search.
- The best results among the methods described in the literature are achieved by the kLAESA algorithm. This method significantly reduces the number of operations needed to classify each patter, with a linear increase in the amount of needed memory. The number of distances is not so low as in the case of the kAESA method, but it has a very low associated number of simple operations.
- At last, it is important to highlight that the proposed kTAESA method achieves the lowest number of simple operations in most of the considered cases. Like with the kAESA, the memory requirements of the proposed method quadratically increase with the training set size. The number of required distances is not so low as in the kAESA case, but the number of simple oper-

ations (sums, comparison and products) is dramatically reduced, mainly with medium and large training sets.

5. CONCLUSIONS

In this paper we propose of a novel implementation of the kNN algorithm: the *k*-tuned approximating and eliminating search algorithm (kTAESA). Results are compared with those achieved by the implementation of the kNN classifier with exhaustive search, the kAESA and the kLAESA algorithms, in terms of number of distances to evaluate, number of simple operations needed to classify each pattern, and amount of required memory. All classifiers are applied to three databases from the UCI machine learning benchmark repository.

Results demonstrate the best performance of the proposal, when the number of simple operations (sums, comparisons and products) is considered. On the other hand, the proposed method requires the highest amount of memory. This fact makes necessary a trade off between the available memory and the CPU speed, in order to select the best fast searching algorithms for implementing kNN classifiers. In those cases in which a real-time classifier is needed, and there is enough memory in the system, then the proposed method could be a suitable choice. And in those cases in which the training set size is too large to store all the variables of the kTAESA algorithm, the kLAESA method could be more interesting.

REFERENCES

[1] K. Fukunaga and M. Narendra, "A branch and bound algorithm for computing k-nearest neighbors", *IEEE Transactions on Computation*, vol. 24, pp. 750-753, 1975.

[2] SeongJoon Baek and Koeng-Mo Sung, "Fast K-Nearest neighbour search algorithm for nonparametric classification", *Electronic Letters*, vol. 36, pp. 1821-1822, 2000.

[3] F. Moreno-Seco, L. Mico and J. Oncina, "Extending LAESA fast nearest neighbour algorithm to find the k nearest neighbours", *Lecture Notes in Computer Science*, vol. 2396, pp. 718-724, 2002.

[4] M.M. Islam, X. Yao and K. Murase, "A constructive algorithm for training cooperative neural network ensembles", *IEEE Transactions on Neural Networks*, vol. 14, no. 4, pp. 820-834, 2003.

[5] C.M. Bishop, *Neural networks for pattern recognition*, Oxford University Press Inc, New York 1995.

[6] E. Vidal, "An algorithm for finding the nearest neighbour in (approximately) constant time", *Pattern Recognition Letter*, vol. 4, pp. 145-157, 1986.

[7] L. Mico, J. Oncina and E. Vidal, "A new version of the nearest neighbour approximating and eliminating search algorithm (AESA) with linear preprocessing-time and memory requirements", *Pattern Recognition Letter*, vol. 15, pp. 9-17, 1994.