A NEW CRITERION FOR APPROXIMATING IN A RECENT VERSION WITH LINEAR PREPROCESSING OF THE AESA ALGORITHM

LUISA MICÓ AND JOSE ONCINA
Departamento de Tecnología Informática y Computación, Universidad de Alicante
Alicante, Spain

ABSTRACT

The LAESA (Linear Approximating Eliminating Search Algorithm) was introduced in order to reduce the large space complexity of the AESA. This is achieved by using only a very small subset of points, called Base Prototypes. Moreover, as the AESA, this algorithm can be used for finding Nearest Neighbours in Metric Spaces with an average constant number of distance computations. The algorithm makes use of two generic functions, CONDITION and CHOICE, to allow for different heuristic strategies of Base Prototypes management. In this paper, we present a new strategy for the CHOICE function that improves the previous results. We also show that with this new criterion, the number of distance computations does not increase with the dimensionality, but with the average distance from the test sample to its nearest neighbour.

1. Introduction

A nearest neighbours classifier finds, given a set of prototypes, which of them is at a minimum distance from a test sample (the nearest neighbour). There are many Pattern Recognition techniques related to this type of classifiers[1]. For this reason, it is very interesting to reduce their complexity. The most interesting techniques are those making only use of the metric space properties. This is due to the large number of practical problems where it is not possible to know the coordinates of the data, and it is very expensive to compute the distance as, for example, in Isolated Words Recognition[2].

The AESA (Eliminating Approximating Search Algorithm), introduced by Vidal in 1986, reduces the number of distance computations compared to other methods previously introduced. Two facts make this algorithm specially interesting: 1) the algorithm makes only use of the metric properties of space, and 2) it finds the nearest neighbour calculating a number of distances (in average) constant with the size of the set[3]. This algorithm performs repeatedly two phases: in the first phase (Approximating) the distance between the test sample and an approximately close prototype is computed. If this distance is smaller than that of the nearest neighbour, updating takes place. In the second phase (Eliminating), those prototypes that cannot be closer than the nearest neighbour are eliminated by using the triangle inequality. The process finishes when the set of prototypes is empty.

Later, a new formulation for the algorithm using the Branch and Bound strategy was introduced[4]. The results, using this technique, were better than those of the first version of the algorithm, because of two facts: reduction in the number of distances and reduction of the overhead. However, the main drawback of this algorithm is the need of computing (and storing) a matrix of distances among all the prototypes during preprocessing time. For this reason, the use of the AESA algorithm is limited to a relatively small set of prototypes[2].

Recently, new techniques appeared reducing the space complexity with the same behaviour as AESA for the number of distance computations. Ramasubramanian and Paliwal[5,6] propose some techniques able to solve this problem in vectorial spaces. At the same time, a new version of the AESA algorithm, called Linear-AESA, was introduced with the same purpose[9]. This algorithm only uses the metric properties of space. The LAESA algorithm, like the AESA, has also two phases, but only makes use of a matrix of distances between a small subset of "Base Prototypes" (BP´s) and the remaining prototypes.

2. Choice of Base Prototypes

The efficiency of the LAESA algorithm depends on the number of selected Base Prototypes and their location with respect to the other prototypes. This last issue was already studied by Marvin Shapiro[10], whose results show better when the reference points (Base Prototypes) are located far away from data clusters. Ramasubramanian and Paliwal also use a similar technique for selecting Base Prototypes, but with the difference with respect to the LAESA that reference points do not belong to the set of data but are placed on the coordinate axis. In order to select Base Prototypes in LAESA, we make use of a greedy strategy with linear cost. This strategy selects incrementally the prototypes that are maximally separated among those previously selected.

3. Algorithm LAESA

The LAESA has, as AESA as well, two main phases: Approximating and Eliminating. The main difference between both algorithms is the use in LAESA of a matrix of distances among the complete set of prototypes (with size n) and a small subset of Base Prototypes (with size m<<n)[7,8,9], while AESA needs the complete matrix among all prototypes.

The Approximating criterion for selecting new candidates is the same one used by Vidal in[4]. The only difference is that updating of the lower bound for the distance (G) of every prototype *p* is only done when Base Prototypes are selected. This is so because we only know the distances among Base Prototypes and the complete set.

$$s = \underset{\forall q \in P-U}{\operatorname{argmin}} G(q) \qquad (1)$$

with

$$G(p) = \max_{\forall u \in B_u} | d(p,u) - d(x,u) | \qquad (2)$$

where B_u is the set of Base Prototypes already used in the approximating process, and U is the set of eliminated prototypes.

With this formulation, a prototype *p* can be eliminated for subsequent search if the following condition is fullfilled

$$G(p) \geq d(x,n) \qquad (3)$$

In the Approximating phase, in order to choose new candidates to nearest neighbour, we select two prototypes each time. One of them is Base Prototype and the other is non-Base Prototype (in AESA only one prototype is selected, because all of them are in fact Base Prototypes). Afterwards, we select one of the two prototypes by means of the CHOICE function and the distance to the test sample is computed. Elimination of base prototypes is controlled by a Boolean function, called CONDITION, which only allows for the elimination if it is true, provided that the elimination condition is simultaneously fullfilled (3).

Base Prototypes must satisfy the eliminating criterion and the CONDITION function must be true. We have considered different functions, where elimination of BP's is allowed[7,8]:

$$\text{CONDITION} = (nc > m/k) \qquad (4)$$

where *nc* is the number of Base prototypes previously selected and *m* is the total number of Base prototypes. For values of $k = 1, 2, \ldots, \infty$, this function leads to the BP management policies hereafter referred to as EC1, EC2,...,EC∞.

We have also studied another CONDITION function, called ECELIM, wich allow elimination of BP's if the last selected prototype contributed no elimination

$$\text{CONDITION} = (\text{no prototype was eliminated in the previous step}) \qquad (5)$$

The CONDITION function we are using is the ECELIM criterion, because from a small size of the set of BP's, the increase of this size does not depend on the number of distance computations. This behaviour is shown in Figure 3, corresponding to rc=∞.

The results using these strategies are quite interesting (the number of distances is smaller than 1.5 times the number of distance computations with AESA). This number, as in AESA, seems to increase rapidly with the course of the dimensionality. However, this is not the case[11]. This behaviour is mainly due to the increase in the average distance from the test samples to their nearest neighbours. Still, if we use in the LAESA algorithm the only CHOICE criterion that we have until now, this desired behaviour is not noticed. This criterion only allows the CHOICE of BP's for computing the distance in the Approximating phase, if this set is not empty.

```
Algorithm LAESA
  Input:   $P \subset E$,  $n = |P|$;                     // finite set of prototypes //
           $B \subseteq P$,  $m = |B|$;                   // set of Base Prototypes //
           $D \in \Re^{n \times m}$;        // precomputed $n \times m$ interprototype distances//
           $x \in E$;                                     // test sample //
  Output: $p^* \in P$; $d^* \in \Re$    // nearest neighbour prototype and its distance to x//
  Functions:  d: $E \times E \rightarrow \Re$                    // distance function //
              CONDITION: Boolean  // controls the elimination of Base Prototypes //
              CHOICE: $B \times (P-B) \rightarrow P$// CHOICE of Base or non-Base Prototypes  //
  Variables:  p, q, s, b $\in$ P ;
              $G \in \Re^n$;                              // lower bound array //
              dxs, gp, gq, gb $\in \Re$ ;
              nc $\in \mathbb{N}$;                        // number of computed distances //
  d*:= $\infty$; p*:= indeterminate; G:= [0]; s:= arbitrary_element (B); nc:=0;
  while |P| > 0 do
    dxs:= d(x,s); P:= P-{s}; nc:=nc+1;                    // distance computing //
    if dxs < d* then p*:= s; d*:= dxs; endif              // updating p*, d* //
    q:= indeterminate; gq:= $\infty$; b:= indeterminate; gb:= $\infty$;
    for every p $\in$ P do                      // eliminating and approximating loop //
      if s $\in$ B then G[p]:= max( G[p], | D[p,s]-dxs | )  // updating G, if possible //
      endif
      gp:= G[p];
      if p $\in$ B then
        if ( gp $\geq$ d* & CONDITION ) then P:= P-{p}        // eliminating from B //
        else
          if gp<gb then gb:= gp; b:= p endif   // approximating: selecting from B //
        endif
      else
        if gp $\geq$ d* then P:= P-{p}                       // eliminating from P-B //
        else                                   // approximating: selecting from P-B //
          if gp<gq then gq:= gp; q:= p endif
        endif
      endif
    endfor
    s:= CHOICE( b, q );
  endwhile
endalgorithm
```

Figure 1. The LAESA algorithm.

4. New CHOICE Criterion

We propose in this work a new criterion allowing the CHOICE of non-base prototypes from the beginning and thus, it is not necessary for the set of BP´s to be empty. With this new criterion, the nearest neighbour to a very close sample can be found very quickly, due to the flexibility in the CHOICE criterion. If we know that the selected non-base prototype is very close to the sample, it will be very interesting to use it at the Elimination phase. Moreover, it is not useful selecting many non-Base Prototypes in the Approximating phase. Selection of non-Base Protoypes increases the number of distances without updating the lower bound of the remaining data. Due to this reason, we propose the following criterion:

```
CHOICE_rc(b, q) =    if in the rc previous iterations q it is the same        (6)
                         then q
                         else b
                     endif
```

for every rc=1,2..∞

This criterion does not allow the choice of non-base prototypes continuously. This is interesting because every time a non-base prototype is selected, we do not update the lower bound of distances of the remaining prototypes. If rc = ∞, we obtain the first criterion, corresponding to the case where we can only select non-base prototypes if the set of BP´s is empty. Thus, we keep on calling this criterion CHOICE_1 and we will call CHOICE_{rc} the new criterion, dependent on the value of rc.

5. Experiments and Results

We have used "tolerance intervals" around each prototype in order to study the behaviour of the algorithm. Test samples are selected in these intervals (Figure 2). The tolerance is given in percentage compared to the corresponding unit hypercube. The samples are very close to its nearest neighbours dependent on the value of the tolerance.

The strategy of elimination used for the criterion CHOICE_{rc} is ECELIM. The number of distance computations does not depend on the increase of the size of the set of BP's using this elimination criterion.

In Figure 3, one can see that the expected behaviour for low tolerances (independence of the number of distances with the dimensionality) is not observed with the CHOICE_∞ criterion. This is so because this criterion only allows the choice of non-base prototypes when this set is empty. Indeed, the algorithm shows the same behaviour in the first part of the execution using test samples very close to their nearest neighbours.

With the new CHOICE criterion this is not the case. In Figure 4, we present some experiments varying the parameter rc in the criterion CHOICE_{rc}. In Figure 5 one can see the behaviour for rc= 3, 4, 6, 8, 10, ∞, in a uniform distribution with respect to the unit hypercube of dimension 6, and for different sizes of the set of BP´s.

In all the experiments with the new criterion and using "tolerance intervals" for the test samples, we have applied the "optimum" number of BP's obtained with the first elimination criterion, EC1 (CONDITION=false) together with the criterion CHOICE_∞[7,8] for each dimension. These sizes are the following:

DIMENSION	2	4	6	8	10
NUMBER OF BP's	3	6	14	28	48

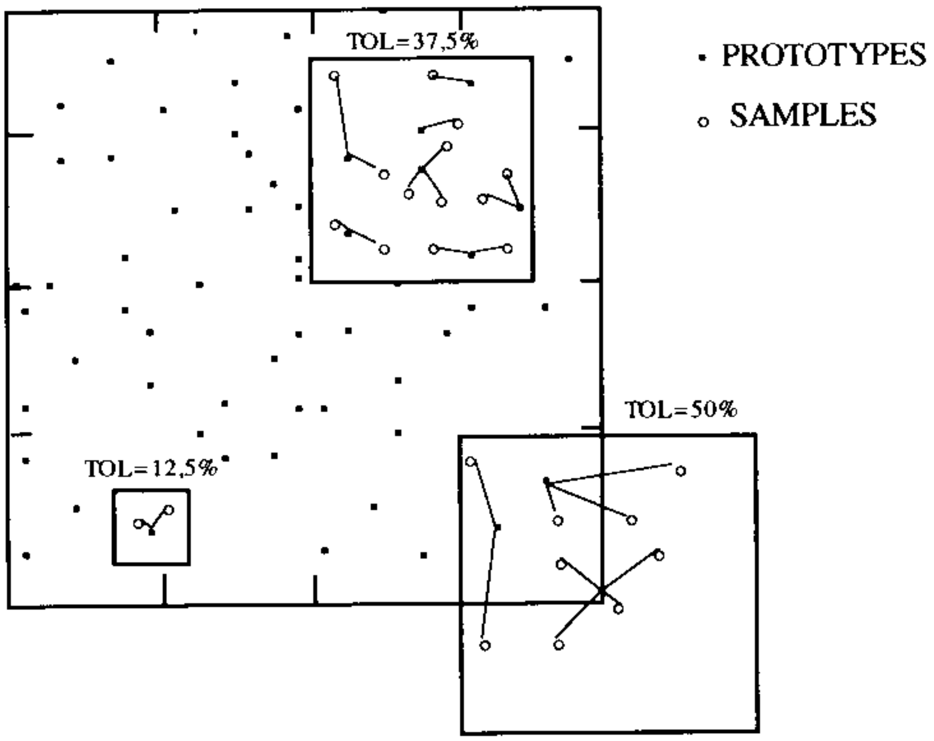

Figure 2. Uniform distribution of prototypes and samples with illustration of the "tolerance interval" concept. The percentages represent the relative size of these intervals (small squares) about the bigger square [11].

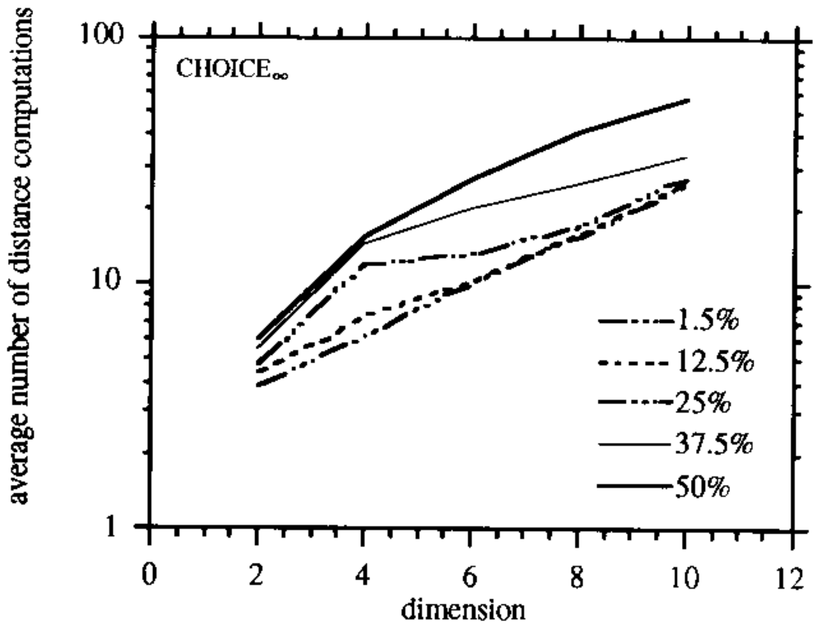

Figure 3. Average number of distance computations for the first criterion of CHOICE, varying the dimension and the tolerance of samples to theirs nearest neighbours.

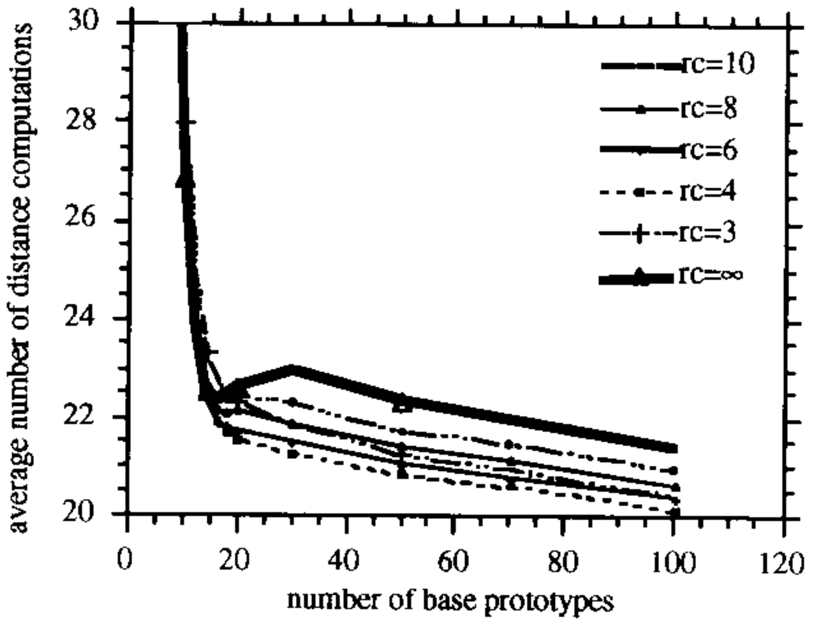

Figure 4. Average number of calculated distances as a function of the number of base prototypes for CHOICE$_{rc}$ using the Euclidean metric with a set of prototypes in dimension 6.

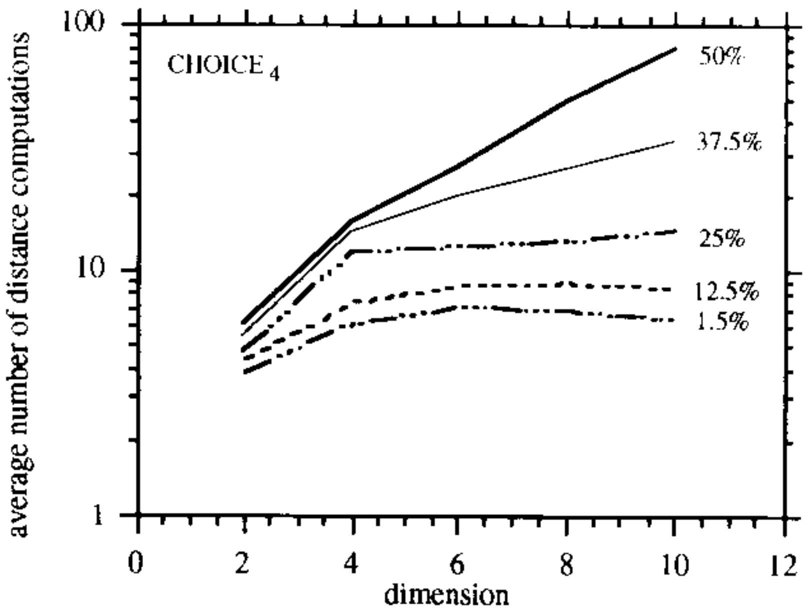

Figure 5. Average number of distances for the new CHOICE criterion varying the dimension and the tolerance (closeness) of test samples to their nearest neighbours.

Comparison of Figures 3 and 4, shows that for high tolerances both criteria have a similar behaviour. However, for lower tolerances the use of the new CHOICE criterion does not increase the number of distances with the dimension and sometimes it even decreases.

6. Conclusions

The results with the new CHOICE criterion show that the number of distances to compute in order to find the nearest neighbour to a test sample does not increase with the dimension, and they even decrease when the data are grouped (tolerance smaller than 15%). In other words, the average number of distance computations depends on the average distance from the test samples to their nearest prototypes. This is the reason for the introduction of the new CHOICE criterion, because for real data, the average number of distances of the samples to their corresponding prototypes does not increase with the addition of new characteristics to the representation of objects. In this way, the behaviour of the algorithm is in practice insensitive to the space dimensionality.

7. References

1. B. Dasarathy, *Nearest Neighbour(NN) norms: NN Pattern Classification Techniques*, IEEE Computer Society Press, 1991.
2. E. Vidal, H. Rulot, F. Casacuberta and J. Benedí, *On the use of a metric-space search algorithm (AESA) for fast DTW-bassed recognition of isolated words*, IEEE Transactions on Acoustics, Speech, and Signal Processing., vol. 36, p. 651-660, 1988.
3. E. Vidal, *An algorithm for finding nearest neighbours in (approximately) constant average time complexity.*, Pattern Recognition Letters, vol. 4, p. 145-157, 1986.
4. E. Vidal, *New formulation and improvements of the Nearest-Neighbour Approximating and Eliminating Search Algorithm (AESA).* To appear in Pattern Recognition Letters.
5. V. Ramasubramanian and K. K. Paliwal, *An Efficient Approximation-Elimination Algorithm for Fast Nearest-Neighbour Search Based on a Spherical Distance Coordinate Formulation.*, Signal Processing V:Theories and Applications., p. 1323-1326, 1990.
6. V. Ramasubramanian and K. K. Paliwal, *Fast Algorithms for Nearest-Neighbour Search and Application to Vector Quantization*, PhD dissertation. University of Bombay, 1991.
7. L. Micó, 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.* To appear in Pattern Recognition Letters.
8. L. Micó, J. Oncina and E. Vidal, *An algorithm for finding nearest neighbours in constant average time with a linear space complexity.* Int. Conference on Pattern Recognition (ICPR), Le Hague, September 1992.
9. L. Micó, J. Oncina and E. Vidal, *Algoritmo para encontrar el vecino más próximo en un tiempo medio constante con una complejidad espacial lineal*, Tech. Report DSIC II/14-91. Universidad Politécnica de Valencia, 1991.
10. M. Shapiro, *The Choice of Reference Points in Best-Match File Searching.*, Artificial Intelligence/Language Processing., vol. 20, pp. 339-343, 1977.
11. E. Vidal, *Diversas aportaciones al Reconocimiento Automático del Habla*, Tesis Doctoral. Fac. Físicas. Universidad de Valencia, 1985.