ELSEVIER

Pattern Recognition Letters 19 (1998) 351–356

Pattern Recognition Letters

Comparison of fast nearest neighbour classifiers for handwritten character recognition [1]

Luisa Micó [*], Jose Oncina [2]

Departamento de Lenguajes y Sistemas Informáticos, Ap. Correos 99, Universidad de Alicante, E-03080 Alicante, Spain

Received 13 August 1997; revised 12 December 1997

Abstract

Recently some fast methods (LAESA and TLAESA) have been proposed to find nearest neighbours in metric spaces. The average number of distances computed by these algorithms does not depend on the number of prototypes and they show linear space complexity. These results where obtained through vast experimentation using only artificial data. In this paper, we corroborate this behaviour when applied to handwritten character recognition tasks. Moreover, we compare LAESA and TLAESA with some classical algorithms also working in metric spaces. © 1998 Elsevier Science B.V. All rights reserved.

Keywords: Metric spaces; Nearest neighbour; Pattern recognition

1. Introduction

Given a set of prototypes whose classification is known, the nearest neighbour technique classifies a test sample in the class containing the prototype whose distance to the test sample is minimum. There are many problems of practical interest in which the distance computation is too expensive, making the brute-force approach impractical: isolated word recognition through dynamic time warping (DTW) (Rabiner and Levinson, 1984), attributed graph match searching (Shapiro and Haralik, 1985), or best-match string edit searching (Wagner and Fischer, 1974).

In this paper, the behaviour of the LAESA (Micó et al., 1994) and TLAESA (Micó et al., 1996) algorithms is studied in a real life task. In particular, these algorithms have been applied to the recognition of handwritten digits using strings of symbols obtained from the contour of characters. This technique provides good classification results in relation to the simplicity of the feature extraction method (Gómez et al., 1995). On the other hand, the high computational cost of the edit distance makes in principle LAESA and TLAESA specially adequate due to their low number of distance computations.

In this work, both algorithms will be compared with other classical algorithms that can be used in the same context (in the sense that they only make use of the triangle inequality property of the distance).

The algorithms used for comparison are:

- the Fukunaga and Narendra algorithm (1975) (FUKNA),
- the Kalantari and McDonald algorithm (1983) (KALMD).

[*] Corresponding author. Email: mico@dlsi.ua.es.

[1] Work partially supported by the Spanish grant CICYT TIC97-0941.

[2] Email: oncina@dlsi.ua.es.

Both algorithms have characteristics which are similar to those of the LAESA and TLAESA methods: (1) space complexity linear, (2) they only make use of the triangle inequality property of the distance for searching and (3) they work in sublinear time (as the TLAESA algorithm). There are no other methods in the literature with these characteristics.

We apply them to the NIST Special Database 3. This database consists of 128×128 bitmap images of handwritten digits and letters. Digits written by 100 different writers were used in the experiments.

2. Feature extraction

Each digit is coded as an octal string, the string representing the contour of the image. The procedure scans the bitmap let-to-right and starting from the top. When the first pixel on is found, it follows the border of the character until it returns to the first pixel. During this traversal, the algorithm builds a string with the absolute direction of the next pixel in the border (see Fig. 1).

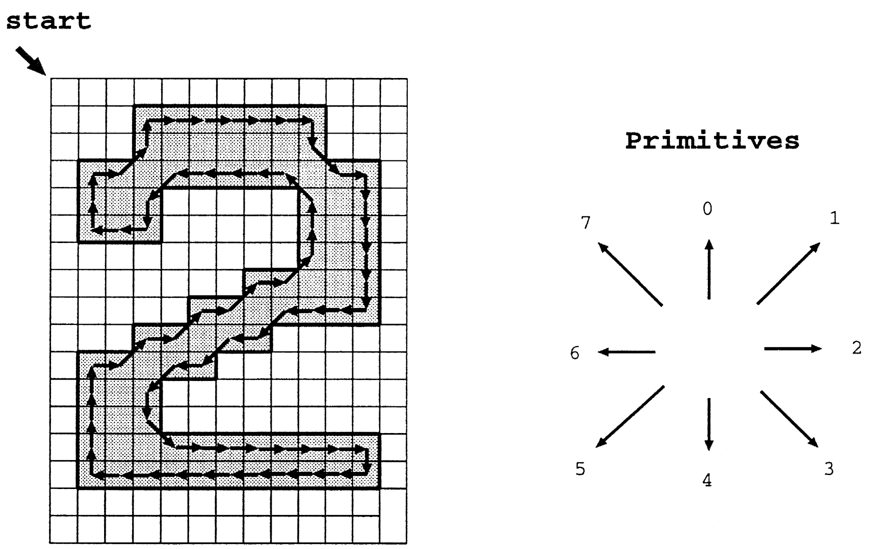
In order to reduce the length of the strings, the bitmap image has been previously compressed to a 64×64 one.

3. The edit distance

The edit distance between two strings x and y is defined as the minimum-cost set of transformations that must be done to turn a string into the other. The allowed transformations are insertions, deletions and substitutions of a single symbol in the string. This distance can be also defined recursively as follows:

$$d(\lambda,\lambda) = 0,$$
$$d(\lambda,yb) = d(\lambda,y) + W_{\mathrm{d}},$$
$$d(xa,\lambda) = d(x,\lambda) + W_{\mathrm{i}},$$
$$d(xa,yb) = \min\begin{cases} d(x,yb) + W_{\mathrm{d}}, \\ d(xa,y) + W_{\mathrm{i}}, \\ d(x,y) + W_{\mathrm{s}}(a,b), \end{cases}$$

where $a,b \in \Sigma$, $x,y \in \Sigma^*$, λ is the empty string, W_{d}, W_{i} are, respectively, the cost of making a deletion and the cost of making an insertion, and $W_{\mathrm{s}}(a,b)$ is the cost of substituting (mistaking) the direction a for b. The substitution costs are proportional to the relative angle between the directions (see Fig. 2) where, in particular, $W_{\mathrm{s}}(a,a) = 0$. This distance can be computed by a dynamic programming algorithm in $O(|x||y|)$ time (Wagner and Fischer, 1974). The



"2"=222222432444446665656543222222246
6666666660000212121210076666546600210

Fig. 1. Example of chain-coding extraction.

$W_d = 1$ $W_i = 1$

W_s	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	0	1	2	3	4	3	2	1
1	1	0	1	2	3	4	3	2
2	2	1	0	1	2	3	4	3
3	3	2	1	0	1	2	3	4
4	4	3	2	1	0	1	2	3
5	3	4	3	2	1	0	1	2
6	2	3	4	3	2	1	0	1
7	1	2	3	4	3	2	1	0

Fig. 2. Insertion, deletion and substitution costs.

edit distance was normalized dividing $d(x,y)$ by the sum of the lengths of the two strings, $|x|+|y|$.

4. The algorithms

All the fast algorithms used to find nearest neighbours consist of two parts: the preprocessing and the search. During the preprocessing, the algorithms build some structures that speed up the search part. During the search part, the nearest prototype is searched but the triangle inequality is used in order to avoid the calculation of some distances. This is done by discarding those that cannot be closer to the sample than a current one (elimination criterion).

A technique often used to reduce the number of distances involved in real applications is to allow some ‘‘looseness’’ in the triangle inequality (Vidal et al., 1988).

Given a representation space E, the *looseness* is defined for each $x,y,z \in E$ as

$$h(x,y,z) = d(x,y) + d(y,z) - d(x,z).$$

If a histogram of values $h(x,y,z)$ is computed for a great number of triplets (x,y,z), this histogram can be used as an estimator of the probability that the triangle inequality is satisfied with a given *looseness*. Therefore, a value H can be chosen such that the probability of a *looseness* smaller than H is negligible (Vidal et al., 1985).

This technique was used in the search part in order to reduce the number of distance computations. The value of the parameter H can be chosen in such a way that the error rate in the classification does not increase or has at most a negligible increase. The use of an adequate value of this parameter results in a drastic reduction of the distance computations. The optimum value of H can be calculated using the method proposed in (Vidal, 1985) or can be obtained experimentally as described below.

The introduction of the *looseness* makes some changes in the algorithms necessary. In particular, in the algorithm a prototype (or set of prototypes) is eliminated if its lower bound distance to the sample is larger than the current minimum distance D_{min}. However, with the use of the looseness, a prototype (or set of prototypes) is eliminated if its lower bound distance to the sample is larger than $D_{\text{min}} - H$.

5. Fitting parameters

A first set of experiments was made in order to study the dependence of the error rate with the *looseness* parameter. In these experiments, 10 writers were used for training and 10 writers as test (each subset contained about 1000 prototypes). As the LAESA and TLAESA algorithms use an additional parameter, the number of base prototypes (prototypes used in the calculation of the lower bound distance), in principle only the FUKNA and the KALMD algorithms were used. The results are shown in Figs. 3 and 4.

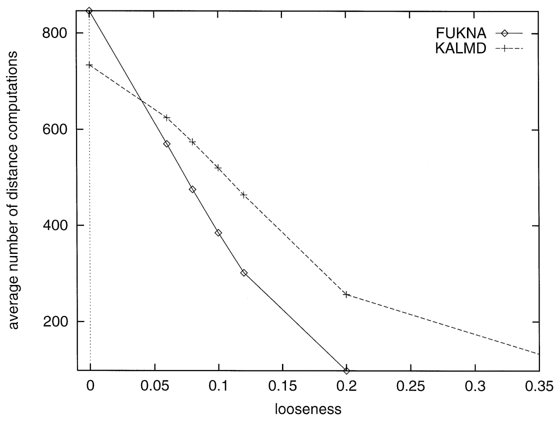


Fig. 3. Average number of distance computations as a function of the looseness for FUKNA and KALMD algorithms using a set of 8000 prototypes. The standard deviation for all estimations was below 4%.

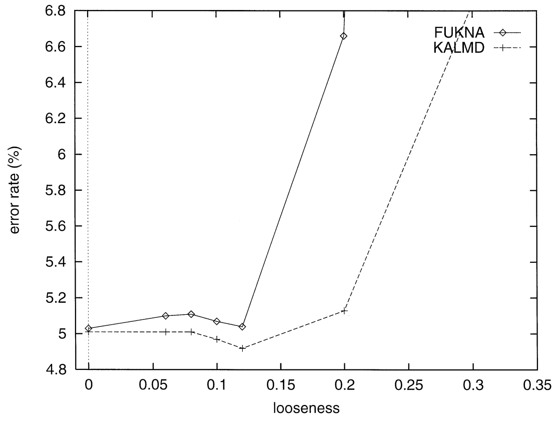


Fig. 4. Error rate as a function of the looseness for FUKNA and KALMD algorithms using a set of 8000 prototypes. The standard deviation for all estimations was below 4%.

It was observed that for values of the looseness lower than 0.1 the increase in the error rate is negligible while there is a significant fall in the number of distance computations. However, beyond this point the error rate increases dramatically. We will call this point *critical looseness*.

In order to obtain a suitable value of base prototypes for LAESA and TLAESA a second set of experiments was performed to show the dependency of the number of distance computations with the number of base prototypes (Fig. 5). These experiments were made using 80 writers as training set and

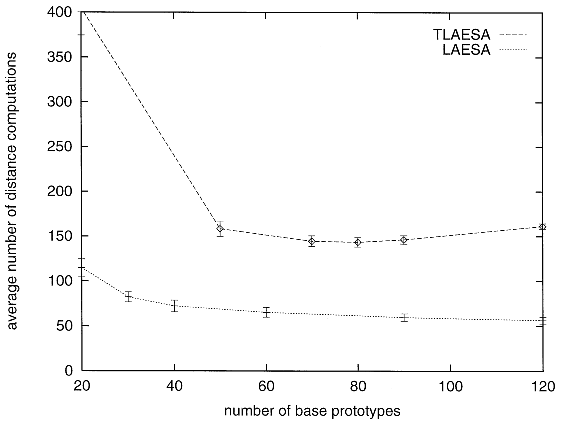


Fig. 5. Average number of distance computations for LAESA and TLAESA algorithms varying the number of base prototypes, for a set of 8000 prototypes on average.

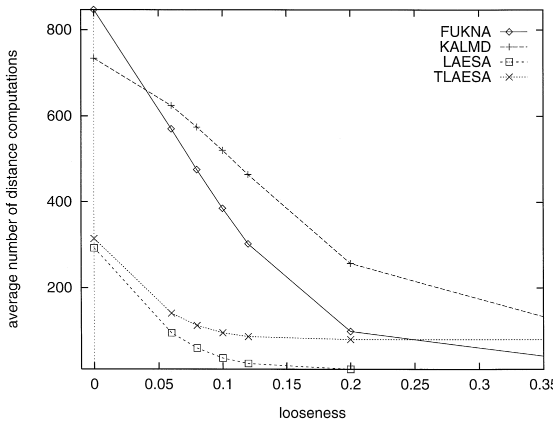


Fig. 6. Average number of distance computations as a function of the looseness for LAESA, TLAESA, FUKNA and KALMD algorithms using a set of 8000 prototypes. The number of base prototypes used in LAESA and TLAESA algorithms was 40 and 80, respectively. The standard deviation for all estimations was below 4%.

10 writers as test set. The selected looseness was the critical ($H = 0.1$) for the FUKNA algorithm.

Fig. 5 shows that the minimum number of distance computations is obtained using about 40 base prototypes for the LAESA and 80 base prototypes for the TLAESA. The experiments in (Micó, 1996) showed that this optimal value does not depend on the number of prototypes.

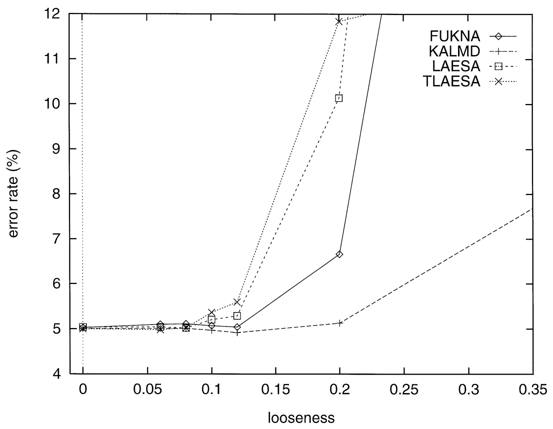


Fig. 7. Error rate as a function of the looseness for LAESA, TLAESA, FUKNA and KALMD algorithms using a set of 8000 prototypes. The number of base prototypes used in LAESA and TLAESA algorithms was 40 and 80, respectively. The standard deviation for all estimations was below 4%.

If the first set of experiments is repeated for the LAESA and TLAESA algorithms with this number of base prototypes, one finds the same critical looseness as in the FUKNA and KALMD experiment in Fig. 4 (see Figs. 6 and 7).

The parameters obtained in this section are also used for all the experiments appearing in next section, i.e., the *looseness* for all algorithms is 0.1, and the number of base prototypes for LAESA and TLAESA is 40 and 80, respectively.

6. Results

In order to compare the behaviour on this task of the four algorithms, a set of experiments was carried out increasing the size of the training set. In Figs. 8 and 9, the average number of distance computations and the consuming time for FUKNA, KALMD, TLAESA and LAESA are compared. It is observed the large difference in the number of distance computations between FUKNA and KALMD compared to TLAESA and LAESA. It is also observed that the number of distance computations for the LAESA and TLAESA algorithms grows very slowly with the size of the training set. The same slow increase is observed for the time consumed.

In (Micó, 1996) the LAESA and TLAESA algorithms were compared. There, it was shown that the time needed is linear for the LAESA and sublinear for the TLAESA as a function of the number of prototypes. However, the LAESA was faster than TLAESA for not very large sets of prototypes or for expensive distance measures. Indeed, the distance that we are using is very expensive, and the TLAESA tends to be slower than LAESA, as shown in Figs. 8 and 9.

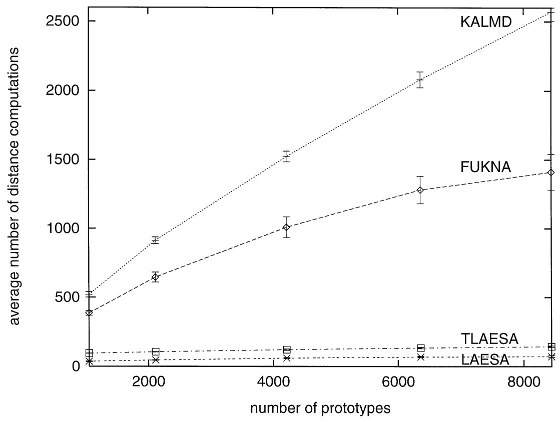


Fig. 8. Average number of distance computations for FUKNA, KALMD, TLAESA and LAESA as a function of the number of prototypes.

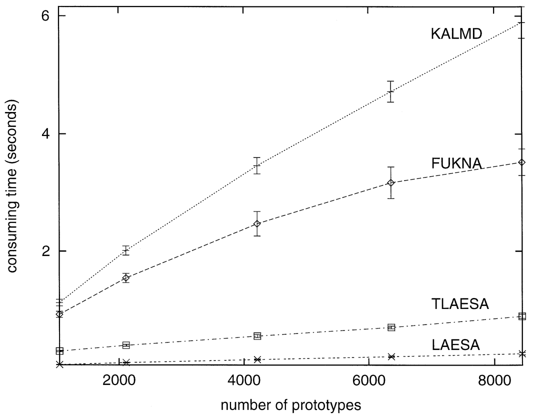


Fig. 9. Time needed (in seconds) for FUKNA, KALMD, TLAESA and LAESA as a function of the number of prototypes.

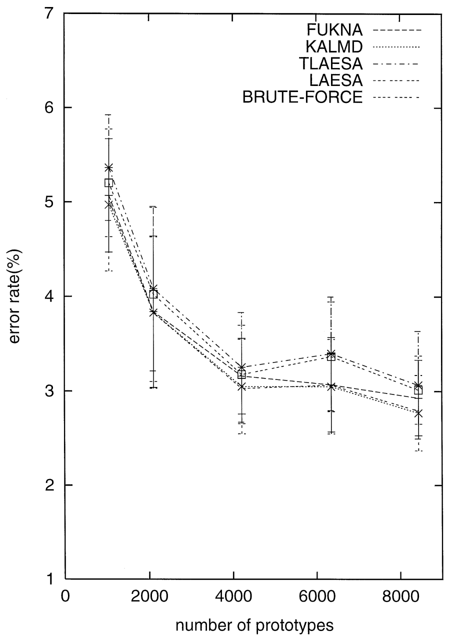


Fig. 10. Classification error rate for BRUTE-FORCE, FUKNA, KALMD, TLAESA and LAESA as a function of the number of prototypes. The error bars correspond to a 95% of confidence level.

A final experiment was made in order to study the error rates when the number of prototypes increases for the four algorithms. The same experiment was repeated using a brute-force algorithm to find the exact nearest neighbour (i.e., $H=0$), as seen in Fig. 10. As expected, the behaviour of the four algorithms is similar to the behaviour of the brute-force algorithm.

7. Conclusions

In this paper we have applied four fast nearest neighbour search algorithms to a handwritten character recognition task in order to compare their behaviour. The selected algorithms were KALMD (Kalantari and McDonald, 1983), FUKNA (Fukunaga and Narendra, 1975), LAESA (Micó et al., 1994) and TLAESA (Micó et al., 1996) The LAESA and TLAESA are algorithms that, with artificial data, showed an excellent performance (the average number of distance computations grows very slowly as the size of the training set increases).

In all the experiments the looseness technique was used in order to reduce the number of distance computations.

The experiments show that LAESA and TLAESA have the same properties observed with artificial data. These algorithms are significantly faster than KALMD and FUKNA.

Acknowledgements

The authors wish to thank Rafael Carrasco for interesting comments.

References

Fukunaga, K., Narendra, M., 1975. A branch and bound algorithm for computing *k*-nearest neighbors. IEEE Trans. Comput. 24, 750–753.

Gómez, E., Micó, L., Oncina, J., 1995. Testing the linear approximating eliminating search algorithm in handwritten character recognition tasks. In: Actas del VI Simposium Nacional de Reconocimiento de Formas y Análisis de Imágenes, Córdoba, pp. 212–217.

Kalantari, I., McDonald, G., 1983. A data structure and an algorithm for the nearest point problem. IEEE Trans. Software Engineering 9, 631–634.

Micó, L., Oncina, J., Vidal, E., 1994. A new version of the nearest-neighbour approximating and eliminating search algorithm (AESA) with linear preprocessing-time and memory requirements. Pattern Recognition Lett. 15, 9–17.

Micó, L., 1996. Algoritmos de búsqueda de vecinos más próximos en espacios métricos. Ph.D. Dissertation. Universidad Politécnica de Valencia.

Micó, L., Oncina, J., Carrasco, R., 1996. A fast branch and bound nearest neighbour classifier in metric spaces. Pattern Recognition Lett. 17, 731–739.

Rabiner, L., Levinson, S.E., 1984. Isolated and connected word recognition – Theory and selected applications. IEEE Trans. Comm. 29, 621–659.

Shapiro, L.G., Haralik, R.M., 1985. A metric for comparing relational descriptions. IEEE Trans. Pattern Anal. Machine Intell. 7, 90–94.

Vidal, E., 1985. Some contributions to automatic speech recognition tasks (in Spanish). Ph.D. Dissertation. Universidad de Valencia.

Vidal, E., Casacuberta, F., Rulot, H., 1985. Is the DTW distance really a metric? An algorithm reducing the number of DTW comparisons in isolated word recognition. Speech Communication 4, 333–344.

1988. On the use of a metric-space search algorithm (AESA) for fast DTW-based recognition of isolated words. IEEE Trans. Acoustics Speech Signal Process. 36, 651–660.

Wagner, R.A., Fischer, M.J., 1974. The string-to-string correction problem. J. ACM 21 (1), 168–173.