A Comparison of Techniques for Automatic Clustering of Handwritten Characters

Vuokko Vuori and Jorma Laaksonen
Helsinki University of Technology
Laboratory of Computer and Information Science
P.O. Box 9800, FIN-02015 HUT, Finland
{vuokko.vuori,jorma.laaksonen}@hut.fi

Abstract

This work reports experiments with four hierarchical clustering algorithms and two clustering indices for on-line handwritten characters. The main motivation of the work is to develop an automatic method for finding a set of prototypical characters which would represent well the different writing styles present in a large international database. One of the major obstacles in achieving this goal is the uneven representation of different writing styles in the database. On the basis of the results of the experiments, we claim that a good set of prototypes can be formed from the combined results of the different clustering algorithms. However, the number of clusters cannot be determined automatically but some human intervention is required.

1. Introduction

One of the main problems in automatic handwriting recognition is the vast variety of different personal writing styles. A recognition system should simultaneously be able to model the differences by which words or characters of different classes can be distinguished from each other, and the variations which can be found within the classes. As recognition accuracy is one of the key factors in determining the usability of a handwriting recognition system and the whole application in which it is implemented, it should be as high as possible for all potential users.

If the recognition is based on comparison of the system's input to a set of known examples, say prototypical handwritten character samples, the higher accuracies can be achieved the better the different writing styles are covered and represented by the prototypes [10]. The same is true when recognition is based on statistical models, say Hidden Markov Models (HHMs). Better results can be obtained if each writing style is modeled with its own HMM instead of using one model per class and for several, sometimes significantly different, writing styles [3]. Initial recognition accuracies can be improved by adapting the recognition system to new writing styles. Nevertheless, a high initial accuracy is crucial – the user might not have enough patience to write many new training samples and to wait for the system to learn them. Therefore, all prior knowledge of different writing styles is of paramount importance, even with adaptive recognition systems.

In this work, we concentrate on the problem of automatically characterizing the possible ways of writing isolated alphanumeric characters. Two different approaches to this problem can be taken. We can try to understand what are the underlying reasons or mechanisms causing the different writing styles. Some of the obvious ones are the handedness, origin, education, age, and health of the writer, fine motorics of hands, and writing equipment. Alternatively, we can use data-driven clustering algorithms to divide the character classes into subclasses each corresponding to a different style of writing. We have taken the latter approach. We have performed experiments with a large character database, four clustering algorithms, and two clustering indices in order to automatically determine the number of different writing styles for digits and upper and lower case letters and to find good prototypes for them. We try to find a set of character prototypes which captures the within-class style variations well. Therefore, the character classes are treated separately. In this approach, the between-class variations are not taken in account and the found prototypes are not optimized in the sense of their classification capacity.

The rest of this paper is organized as follows: the dissimilarity measure between characters, clustering methods and indices are explained in sections 2 and 3, the database used in the experiments is described in section 4, the experiments are introduced and their results are reported in sections 5 and 6, and finally, conclusions are drawn in section 7.

2. Clustering methods

The dissimilarity measure used in the clustering of the characters is based on the Dynamic Time Warping (DTW) algorithm [8]. Connected parts of a drawn curve in which the pen is pressed down on the writing surface are considered as strokes. DTW dissimilarity measure is defined on stroke basis so that it is infinite between two characters having different numbers of strokes. Strokes and data points are matched in the same order as they have been drawn and the first and last data points of the two curves are matched against each other. The DTW algorithm finds the point-to-point correspondence between the curves which satisfies these constraints and yields the minimum sum of the costs associated with the matchings of the data points. The cost for matching two data points is their squared Euclidean distance. Prototype-based classifiers using DTW-based distances have been shown to be well suited for handwriting recognition task by several researchers and high recognition accuracies can be obtained if the prototype set has a good coverage of the different handwriting styles [10].

We have developed four different algorithms for clustering character samples: *TreeClust* [11], *MinSwap*, and two variations of C-means algorithm [5], named here *CMeans 1* and *CMeans 2*. All the four clustering algorithms are agglomerative and hierarchical. Clusters are represented by prototypes which are the samples having the minimum sum of distances to the other samples in the same cluster. *TreeClust*, *MinSwap*, and *CMeans 2* start form a situation in which all the samples are prototypes, i.e. form their own clusters, while in the beginning of *CMeans 1*-algorithm, only a random subset of the samples is selected to be an initial prototype set. As the clustering algorithms proceed, the number of clusters is reduced by merging clusters. In *TreeClust*-, *CMeans 1*-, and *CMeans 2*-algorithms those two clusters whose prototypes are the most similar pair are merged into one. *MinSwap*-algorithm tries several alternative mergings, first the clusters with the most similar prototype pair, then with the next similar pair etc. Next, a prototype is selected among the samples which belong to the new cluster. After that, *MinSwap*, *CMeans 1*, and *CMeans 2* reassign the samples into the clusters according to the closest prototypes and then reselect the prototypes. This is repeated until a stable division is found. *MinSwap* does the same thing but also calculates how many of the samples are swapped out from the new cluster into the other clusters, or vice versa, and selects that merging which causes the minimum number of these swappings.

3. Clustering indices

For determining the number of clusters automatically, two indices were measured: Davies and Bouldin (DB) [4] and Calinski and Harabasz (CH) [2]. Both the indices were modified so that the squared Euclidean distances between two samples or a sample and cluster prototype were replaced by DTW-based distances. In addition, the sample which had the minimum sum of distances to the other samples was used as mean. The DB-index was selected because it is one of the most commonly used clustering indices. It is the average of the similarity measures between each cluster and its most similar cluster. Hence, a sensible number of clusters can be determined by minimizing the DB-index. The CH-index was selected as it outperformed several other clustering measures, including the DB-index, in a thorough comparison carried out by Milligan et al. [7]. In our work, the CH-index is defined as follows:

$$\mathit{CH}(k) = \frac{\sum_{l=1}^{k} n_l d(P_l, M)/(k-1)}{\sum_{l=1}^{k} \sum_{j=1}^{n_l} d(P_l, C_l^j)/(n-k)},$$

where $d(\cdot,\cdot)$ is a DTW-based distance, k is the number of clusters, n is the total number of samples, n_l is the number of samples in the lth cluster, P_l is the prototype of the lth cluster, M is the mean sample, and C_l^j is the jth sample in the lth cluster. The numerator part of the CH-index measures how much the cluster prototypes differ from the mean sample and the denominator part tells how much the samples differ from their cluster prototypes. Therefore, a reasonable number of clusters can be found by maximizing the CH-index.

4. Character database

The experiments were performed with two public databases, named IRONOFF [9] and UNIPEN train_r01_v07 [6]. Only isolated digits and upper and lower case letters were used in the experiments. The two databases were combined into one, all the character samples were manually checked and obviously erroneous ones were removed. Most of the erroneous samples were incorrectly segmented (a whole word instead of a letter, several digits of the same class, lower case letter consisting of single vertical stroke and labeled, for example, as 'k'). In total, 3 174 erroneous samples were found. The total number of samples in the cleaned database is 130 831. These samples were written by 728 subjects. The subjects were of various ages and from several countries, and both genders and handedness groups were represented. In our opinion, it is justified to assume that the database has a rather good coverage of the possible writing styles.

The character samples were collected with pressure sensitive displays or tablets which were able to record the x- and y-coordinates of the moving pen point. As there were several contributors and therefore many different collection softwares and devices, all the samples were preprocessed so that their data points were similarly equidistant in space. In addition, the size and location variations of the characters were normalized. This way, all the characters could be reasonably compared with each other using the DTW distance.

5. Experiments

First, all the characters were divided into subclasses according to their classes and stroke numbers. Next, all the subclasses were clustered independently using the four algorithms introduced in section 2. After that, the DB- and CH-indices were evaluated and plotted for each of the character subclasses as functions of the number of clusters. The minimum values of the DB-index and the maximum values of the CH-index were found and the corresponding numbers of clusters were recorded. These recorded cluster numbers were the optimal prototype set sizes suggested by the clustering indices. In the case *MinSwap* algorithm, 5 alternative mergings were tried. *CMeans 1* was initialized with 50 and 20 randomly selected prototypes in the case of digits and letters, respectively. If there were less samples than that, all the samples were selected as initial prototypes.

Our main objective was to find clearly distinct prototypes for each character class which would represent well all the writing styles present in the large database. No attention was paid to how well the prototypes could capture differences between the classes i.e. would perform in a classification task. Missing rare styles were considered to be a much worse problem than over-represented common styles. Different clustering results were judged and compared with each other on the basis of the knowledge on the writing styles established during the manual examination and cleaning of the character database.

6. Results

Manual inspection of the clustering results showed that only the solutions with less than 20 prototypes were interesting: if the rare styles were not present among them, they would not be found unless the number of clusters, and simultaneously the number prototypes representing similar common styles, were increased considerably. Therefore clustering indices were evaluated only for solutions with 20 clusters or less. The best DB- and CH-indices and corresponding numbers of clusters for digits are shown in Table 1. The table shows that the indices agree rarely and a manual examination of the clustering results revealed that neither one of them does make much sense in practice. The plots of clustering indices were jagged with many local optima and usually with no clear depressive or elevative trend around the global optimum point. In addition, different clustering algorithms found different rare writing styles. Naturally, the same problems were present in the clustering results of letters. Prototypes found for common styles were not dependent on the choice of the clustering algorithm.

Table 1. The numbers of clusters yielding the minimum and maximum values of the DB- and CH-indices for digit subclasses. The number of clusters picked by hand is given in the last column.

Class	Strokes	CMeans 1		CMeans 2		TreeClust		MinSwap		By hand
		DB	CH	DB	CH	DB	CH	DB	CH	
0	1	2	9	9	4	4	2	19	4	19
0	2	2	11	7	13	3	19	9	18	11
0	4	1	1	1	1	1	1	1	1	0
1	1	2	16	2	3	2	2	2	7	14
1	2	3	4	4	17	2	20	18	19	14
1	3	2	20	2	20	2	20	9	20	11
2	1	2	2	4	20	2	2	17	7	16
2	2	3	5	2	5	2	4	3	5	1
3	1	2	4	2	16	2	3	2	3	17
3	2	3	6	4	6	2	4	4	6	0
4	1	2	4	2	4	2	2	14	4	23
4	2	7	5	3	9	4	2	10	18	22
4	3	2	12	2	12	2	2	2	12	6
5	1	2	7	2	19	2	12	16	5	16
5	2	2	10	2	5	2	3	14	6	11
5	3	2	1	2	1	2	4	2	1	1
6	1	2	5	4	2	2	1	19	13	20
6	2	2	3	2	3	2	18	2	3	0
7	1	2	3	2	3	2	2	2	5	18
7	2	4	9	14	10	2	8	6	4	20
7	3	2	20	9	19	2	2	2	20	20
8	1	2	4	4	7	4	20	14	4	18
8	2	2	17	2	10	2	3	17	13	17
8	3	1	1	1	1	1	1	1	1	0
9	1	2	3	2	3	2	19	20	5	19
9	2	3	20	9	20	2	14	10	20	17
Sum	-	59	200	97	231	55	188	233	222	330

CMeans 1 could find the rare styles hardly at all unless they were selected as initial prototypes, which was not very probable as the common styles were heavily over-represented. *CMeans 2* performed little better in that sense as initially all the samples were prototypes. However, due to the reassignments of the samples into the clusters, the small clusters of high internal variance were often absorbed into the big clusters. There were no clear difference between *CMeans 2* and *MinSwap* – perhaps there should have been some kind of normalization by the size of the new cluster for the number of swappings. The current version of the *MinSwap*-algorithm probably favors merging small clusters as there simply can not be much swapping with its small number of samples. Of all the four algorithms *TreeClust* could best preserve the small clusters of rare styles in the merging process. However, assignment of the samples into the clusters is always final and the prototypes of common styles are not always the best representatives of all the samples of that style. Instead of selecting one general prototype, *TreeClust* algorithm tends to select a couple of more extreme samples of the same style. This problem can be alleviated by fine-tuning the final prototypes for example with a Learning Vector Quantization-based algorithm [1, 11].

We also picked the prototypes by hand from the clustering results obtained with the four clustering algorithms. In total, 330, 851, and 1 410 prototypes were selected for

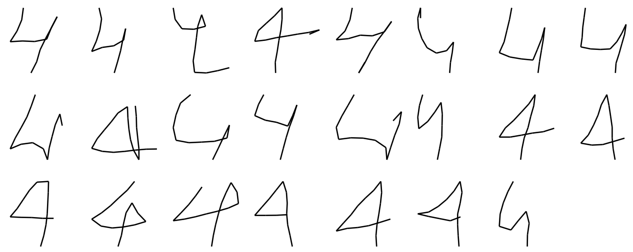

Figure 1. Hand-picked prototypes for one-stroke digit '4'.

digits, lower, and upper case letters, respectively. The prototypes for rare styles were mainly picked from the results of *TreeClust*-algorithm and prototypes for common styles from the results of *CMeans 2*- and *MinSwap*-algorithms. The number of picked prototypes per each digit subclass are shown in the last column of Table 1. These numbers show what is our idea of the correct number of clusters. This is of course only a subjective opinion. The prototypes picked for one-stroke digit '4' are shown in Figure 1. It should be noted that the number of clusters determined automatically by a clustering index and the number decided by a human can not be compared straightaway. The DTW-based distance used by the clustering algorithms and indices and the way how humans perceive the similarities and dissimilarities between two characters are far from equal.

7. Conclusions

We performed experiments with four agglomerative clustering algorithms and two clustering indices in order to automatically find a good set of prototypes of different handwriting styles from a large database of isolated handwritten characters. The major problem was that different writing styles are not equally represented in the database: there are hundreds of examples of some common styles but only a few samples of the rare, more personal styles. If the samples were redistributed among the clusters after a merging of clusters, the small cluster of the rare styles were easily lost as they tended to be merged into the large clusters. None of the four clustering algorithms performed sufficiently by their own but a good prototype set could be formed from their combined results. The examination of the clustering results, approximately 12 000 character samples from which about 22% were selected as prototypes, required only a fraction of the time spent when going through over 130 000 samples in the database.

The results obtained with the clustering indexes were not promising at all: the number of different writing styles had to be determined by a human inspector and the final prototype set was picked by hand. However, the amount of hand-work would be considerably reduced and the over-representation problem alleviated by introducing a two-phased clustering scheme. In the first phase, the database would be clustered with several different algorithms and the number of clusters would be left rather high. In the second phase, the clustering results would be combined and a new set of initial prototypes would be defined on the basis of that. The clustering process would then be continued with some automatic method. The decision on the size of the final prototype set would still be left to humans. We have already performed some experiments with a two-phased clustering algorithm. The preliminary results are encouraging. We compared the automatically selected prototypes to the hand-picked ones and noticed that in the early stages of the two- and single-phased clustering algorithms there were no significant differences if all the samples were used as initial prototypes. However, the prototype set found with a two-phased algorithm contained more rare styles and therefore reminded more the hand-picked prototype set as the size of the prototype set was decreased.

References

[1] F. Andrianasy and M. Milgram. A new learning scheme for the recognition of dynamical handwritten characters. In *Proceedings of the 5th IEEE Workshop on Neural Networks for Signal Processing*, pages 371–379, USA, 1995.

[2] T. Calinski and J. Harabasz. A dendrite method for cluster analysis. *Communications in Statistics*, 3:1–27, 1974.

[3] S. D. Connell. *Online handwriting recognition using multiple pattern class models*. PhD thesis, Michigan State University, 2000.

[4] D. L. Davies and D. W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(2):224–227, April 1979.

[5] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, 1973.

[6] I. Guyon, L. Schomaker, R. Plamondon, M. Liberman, and S. Janet. Unipen project of on-line data exchange and recognizer benchmark. In *Proceedings of International Conference on Pattern Recognition*, pages 29–33, 1994.

[7] G. W. Milligan and M. C. Cooper. An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, 50(2):159–179, June 1985.

[8] D. Sankoff and J. B. Kruskal. *Time warps, string edits, and macromolecules: the theory and practice of sequence comparison*. Addison-Wesley, 1983.

[9] C. Viard-Gaudin, P. M. Lallican, S. Knerr, and P. Binter. The IRESTE on/off (IRONOFF) dual handwriting database. In *Proceedings of 5th International Conference on Document Analysis and Recognition*, pages 455–458, 1999.

[10] V. Vuori, J. Laaksonen, E. Oja, and J. Kangas. Experiments with adaptation strategies for a prototype-based recognition system for isolated handwritten characters. *Int. Journal on Document Analysis and Recognition*, 3(3):150–159, 2001.

[11] V. Vuori and E. Oja. Analysis of different writing styles with the self-organizing map. In *Proceedings of the 7th International Conference on Neural Information Processing*, volume 2, pages 1243–1247, November 2000.