

# A Comparison of Clustering Methods for Writer Identification and Verification

Marius Bulacu    Lambert Schomaker  
AI Institute, Groningen University, The Netherlands  
(bulacu, schomaker)@ai.rug.nl

## Abstract

*An effective method for writer identification and verification is based on assuming that each writer acts as a stochastic generator of ink-trace fragments, or graphemes. The probability distribution of these simple shapes in a given handwriting sample is characteristic for the writer and is computed using a common codebook of graphemes obtained by clustering. In previous studies we used contours to encode the graphemes, in the current paper we explore a complementary shape representation using normalized bitmaps. The most important aim of the current work is to compare three different clustering methods for generating the grapheme codebook: k-means, Kohonen SOM 1D and 2D. Large scale computational experiments show that the proposed method is robust to the underlying shape representation used (whether contours or normalized bitmaps), to the size of codebook used (stable performance for sizes from  $10^2$  to  $2.5 \times 10^3$ ) and to the clustering method used to generate the codebook (essentially the same performance was obtained for all three clustering methods).*

## 1. Introduction

Research in writer identification and verification has received significant interest in recent years due to its forensic applicability [8, 7, 13, 1, 10]. A *writer identification* system performs a one-to-many search in a large database with handwriting samples of known authorship and returns a likely list of candidates. This list is further scrutinized by the forensic expert who takes the final decision regarding the identity of the author of the questioned sample. *Writer verification* involves a one-to-one comparison with a decision whether or not the two samples are written by the same person. The decidability of this problem gives insight into the nature of handwriting individuality [13].

While *texture-level* approaches that use directional features (capturing slant, curvature, regularity) prove to be very efficient [3], they must be complemented by *allograph-level*, i.e., character-shape based approaches in order to ob-

tain adequate and robust results [14]. New results also show that writer-specialized handwriting recognizers can be used for writer identification and verification [9].

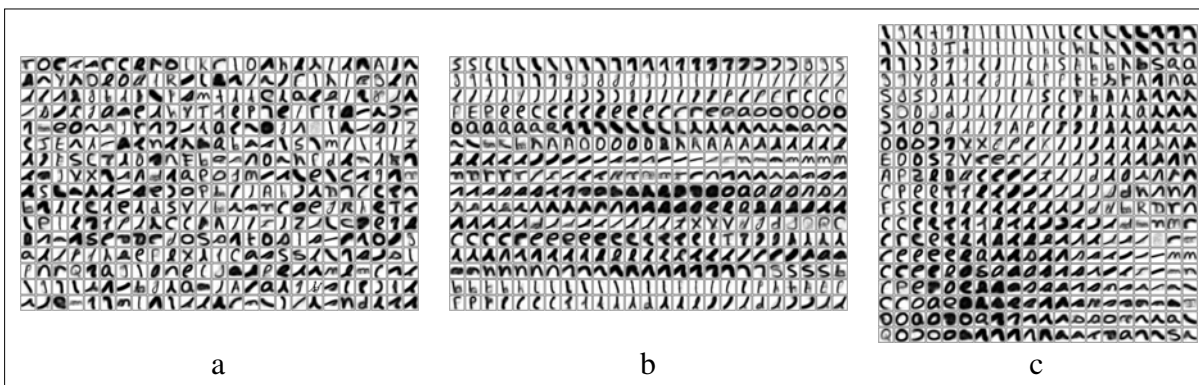
Recently, we have proposed an effective writer identification method in which the writer is assumed to act as a stochastic generator of ink-blob shapes, or graphemes. The probability distribution (PDF) of grapheme usage is characteristic of each writer and is computed using a common codebook obtained by clustering. This approach was first applied to isolated uppercase handwriting [10] and later it was extended to lowercase cursive handwriting by using a segmentation method [11].

In these previous studies, we have used contours for shape representation and a 2D Kohonen self-organizing map (KSOM) for generating the grapheme codebook. While contours possess definite advantages for shape matching, they are nevertheless susceptible to problems regarding the starting point, open/closed loops or the presence of multiple inner contours. On the other hand, pixel-based representations can be more robustly extracted from the handwriting images, but the matching process becomes more vulnerable in this case, e.g., due to quantization in rescaling. The first purpose of this paper is to explore the use of normalized bitmaps as the underlying shape representation. In this respect, our paper comes closest to the work reported in [1, 2] where an information-retrieval framework is used for writer identification. In contrast, our approach uses explicit probability distributions constructed on the basis of the shape codebooks to characterize writer individuality.

The second and most important purpose of the current work is to compare three different clustering methods for generating the grapheme codebook: k-means, Kohonen SOM 1D and 2D. We have run large scale computational experiments for comparing these three clustering methods over a large range of codebook sizes. Both writer identification and verification will be considered in our evaluation.

## 2. Datasets

We conducted our writer identification and verification study using two datasets: Firemaker and ImUnipen.



**Figure 1. Examples of codebooks with 400 graphemes. For kmeans (a) and ksom1D (b) the graphemes have been placed 25 in a row, while for ksom2D (c) the 20x20 original SOM organization has been maintained.**

The Firemaker set [12] contains handwriting collected from 250 Dutch subjects, predominantly students, who were required to write 4 different A4 pages. On page 1 they were asked to copy a text of 5 paragraphs using normal handwriting style (i.e. predominantly lowercase with some capital letters at the beginning of sentences and names). On page 2 they were asked to copy another text of 2 paragraphs using only uppercase letters. The category of page 3 ("forged") samples was not used here. On page 4 they were asked to describe the content of a given cartoon in their own words. These samples consist of mostly lowercase handwriting of varying text content and the amount of written ink varies significantly, from 2 lines up to a full page. The response sheets were scanned at 300 dpi, 8 bits / pixel, gray-scale. In the writer identification and verification experiments reported here, we performed searches/matches of page 1 vs. 4 (Firemaker lowercase) and paragraph 1 vs. 2 from page 2 (Firemaker uppercase).

The ImUnipen set contains handwriting from 215 subjects, 2 samples per writer. The images were derived from the Unipen database [5] of on-line handwriting. The time sequences of coordinates were transformed to simulated 300 dpi images using a Bresenham line generator and an appropriate brushing function. The samples contain lowercase handwriting with varying text content and amount of ink. The dataset was divided in two parts: 65 writers (130 samples) were used for training the grapheme codebook and the rest of 150 writers (300 samples) were used for testing.

### 3. Segmentation Method

In free-style cursive handwriting, connected-components may encompass several characters or syllables. A segmentation method that isolates individual characters remains an elusive goal for handwriting re-

search. Nevertheless, several heuristics can be applied, yielding graphemes (sub- or supra-allographic fragments) that may or may not overlap a complete character. While this represents a fundamental problem for handwriting recognition, the fraglets generated by the segmentation procedure can still be effectively used for writer identification. The essential idea is that the ensemble of these simple graphemes still manages to capture the shape details of the allographs emitted by the writer.

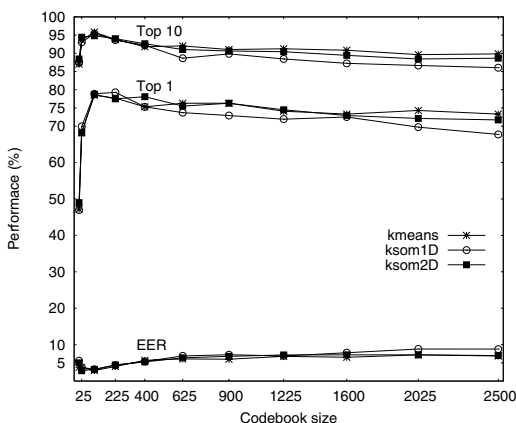


**Figure 2. Segmentation at the minima in the lower contour that are proximal to the upper contour.**

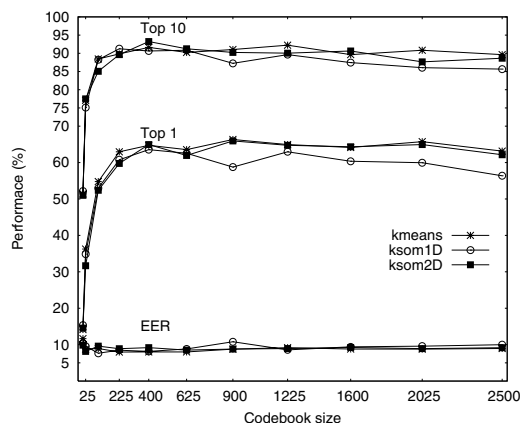
We segment handwriting at the minima in the lower contour with the added condition that the distance to the upper contour is in the order of the ink-trace width (see fig. 2). For contour extraction we use Moore's algorithm. After segmentation, graphemes are extracted as connected components, followed by a size normalization to 30x30 pixel bitmaps, preserving the aspect ratio of the original pattern.

### 4. Grapheme Codebook Generation

A number of 130 samples corresponding to 65 writers have been taken from the ImUnipen dataset. The graphemes have been extracted from these samples using the described procedure yielding a training set containing a total of 41k patterns (normalized bitmaps).



**Figure 3. Performance on the Firemaker lowercase dataset as a function of codebook size.**



**Figure 4. Performance on Firemaker uppercase dataset as a function of codebook size.**

Three clustering methods will be used to generate the grapheme codebook: k-means, Kohonen SOM 1D and 2D. We use standard implementations of these methods. Complete and clear descriptions of the algorithms can be found in references [4, 6].

The size of the codebook (the number of clusters used) yielding optimal performance is an important parameter in our method. In the experiments, we will explore a large range of codebook sizes. This will allow a thorough comparison of the considered clustering algorithms.

Fig.1 shows examples of codebooks that have been obtained by training using each of the three clustering methods. The two codebooks obtained using Kohonen training show spatial order, while the one obtained using k-means is "disorderly". The ksom1D codebook must be understood as a long linear string of shapes and gradual transitions can be observed if the map is "read" in left-to-right top-to-bottom order. The ksom2D codebook shows a clear bidimensional organization.

## 5. Computing Writer-Specific Grapheme Emission PDFs

The writer is considered to be characterized by a stochastic pattern generator, producing a family of basic shapes. The individual shape emission probability is computed by building a histogram in which one bin is allocated to every grapheme in the codebook.

For every sample  $i$  of handwriting, the graphemes are extracted using the segmentation/connected-component-detection/size-normalization procedure described before. For every grapheme  $g$  in the sample, the nearest codebook prototype  $w$  (the winner) is found using the Euclidean distance and this occurrence is counted into the corresponding histogram bin:

$$w = \operatorname{argmin}_n [\operatorname{dist}(g, C_n)], h_{iw} \leftarrow h_{iw} + 1 \quad (1)$$

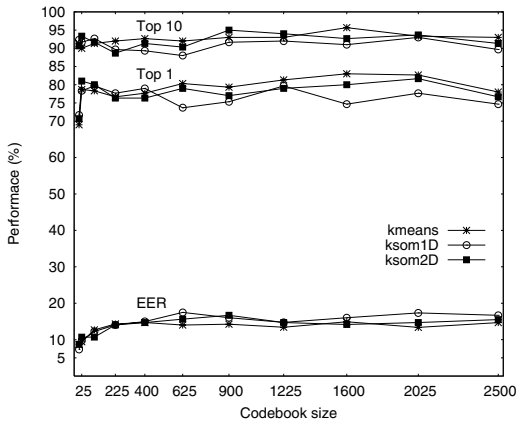
where  $n$  is an index that runs over the shapes in the codebook  $C$ . In the end, the histogram  $h_i$  is normalized to a probability distribution function  $p_i$  that sums up to 1. This PDF is the writer descriptor used for identification and verification.

## 6. Results

We performed large scale computational experiments to compare the three clustering methods over a large range of codebook sizes. The number of clusters used was varied from 9 (3x3) to 2500 (50x50). A number of 200 epochs have been used for training the Kohonen SOMs. Computations have been performed on a Beowulf high-performance Linux cluster with 1.7GHz/0.5GB nodes. Training times for codebooks of size 400: k-means - 1 hrs, ksom1D - 10 hrs, ksom2D - 17 hrs. Computation times for the grapheme emission PDF on codebooks of size 400: k-means - 0.5 s / sample, ksom1D - 1.5 s / sample, ksom2D - 3.1 s / sample. These computation times were obtained using the 'gcc' compiler with optimization for single-precision floating-point calculations. The total computation time used in the experiments amounts to approx. 800 CPU hrs.

### Writer Identification

Writer identification performance is computed using nearest-neighbor classification in a leave-one-out strategy. For a query sample  $q$ , the distances to all the other samples  $i \neq q$  are computed. Then all the samples  $i$  are ordered in a sorted hitlist with increasing distance to the query  $q$ . Ideally the first ranked sample (Top 1) should be the pair sample produced by the same writer (in all our experiments there are 2 samples per writer).



**Figure 5. Performance on the ImUnipen dataset as a function of codebook size.**

An appropriate dissimilarity measure between the grapheme PDFs is the  $\chi^2$  distance:

$$\chi_{qi}^2 = \sum_{n=1}^k \frac{(p_{qn} - p_{in})^2}{p_{qn} + p_{in}} \quad (2)$$

where  $p$  are entries in the PDF,  $n$  is the bin index and  $k$  is the number of bins in the PDF (equal to the size of the grapheme codebook). In our experiments,  $\chi^2$  outperformed other distance measures: Hamming, Euclid, Minkowski order 3, Bhattacharya.

We point out that we do not make a separation between a training set and a test set, all the data is in one suite. This is actually a more difficult and realistic testing condition, with more distractors: not 1, but 2 per false writer and only one correct hit.

Figures 3, 4, 5 show our results obtained on the experimental datasets. Writer identification performance (Top-1 and Top-10) reaches a plateau for codebook sizes larger than about 100 (10x10) shapes. More remarkable is the fact that the same performance is achieved by all three clustering methods. Table 1 gives numerical results for codebooks of size 400 which was chosen as an anchor point.

The lower performance obtained on the Firemaker uppercase dataset can be explained by two factors: the amount of handwriting in these samples is very reduced (only one paragraph of 100-150 characters) and the codebooks have been generated based on samples that contain almost exclusively lowercase (cursive) handwriting. Nevertheless, the overall performance levels achieved on lowercase and uppercase are quite comparable. In a previous study using edge-based directional features under the condition that roughly the same amount of ink is present in all samples, the performance level achieved on lowercase and uppercase was the same [3]. These empirical results contradict the idea

Dataset / Method		<i>kmeans</i>	<i>ksom1D</i>	<i>ksom2D</i>
Firemaker lowercase (250 writers)	Top 1	75.3	75.3	78.1
	Top 10	91.8	92.2	92.6
	EER	5.7	5.4	5.3
Firemaker uppercase (250 writers)	Top 1	64.7	63.6	64.9
	Top 10	91.6	90.6	93.2
	EER	8.0	8.2	9.2
ImUnipen (150 writers)	Top 1	77.7	79.0	76.3
	Top 10	92.7	89.3	91.3
	ERR	14.7	15.0	14.7

**Table 1. Identification and verification accuracies (percentages) for codebooks of size 400 (20x20).**

(which one might intuitively expect) that it is always easier to identify the author of lowercase rather than uppercase handwriting. The slightly higher performance obtained on ImUnipen is due to the smaller number of writers in the dataset.

The writer identification results presented in this paper are in the same ballpark as the ones we reported in a previous study using contours for shape representation and Kohonen 2D for codebook training [11]. This constitutes additional evidence regarding the robustness of the proposed method of using grapheme emission PDFs for writer identification.

### Writer Verification

In the writer verification task, the distance  $x$  between two given handwriting samples is computed using the grapheme PDFs. Distances up to a predefined decision threshold  $T$  are deemed sufficiently low for considering that the two samples have been written by the same person. Beyond  $T$ , the samples are considered to have been written by different persons. Two types of error are possible: falsely accepting (FA) that two samples are written by the same person when in fact this is not true or falsely rejecting (FR) that two samples are written by the same person when in fact this is the case. The associated error rates are FAR and FRR. In a scenario in which a suspect must be found in a stream of documents, FAR becomes false alarm rate, while FRR becomes miss rate. These error rates can be empirically computed by integrating up-to/from the decision threshold  $T$  the probability distribution of distances between samples written by the same person  $P_S(x)$  and the probability distribution of distances between samples written by different persons  $P_D(x)$ :

$$FAR = \int_0^T P_D(x) dx, \quad FRR = \int_T^\infty P_S(x) dx. \quad (3)$$

By varying the threshold  $T$  a Receiver Operating Characteristic (ROC) curve is obtained that illustrates the in-

evitable trade-off between the two error rates. The Equal Error Rate (EER) corresponds to the point on the ROC curve where FAR = FRR and it quantifies in a single number the writer verification performance.

For the Firemaker dataset,  $P_S(x)$  has been constructed using the 250 same-writer distances, while  $P_D(x)$  has been constructed using all the  $C_{500}^2 - 250 = 124500$  different-writer distances arising in the dataset. Similarly for ImUnipen. In figures 3, 4, 5, the lower family of curves show the ERR as a function of codebook size. Here again the same performance is achieved by all three clustering methods.

For Firemaker uppercase, the EER hovers around 8%. For Firemaker lowercase, the EER reaches a minimum of about 3% for a codebook size of 100 and increases to about 7% for larger codebooks. A similar increase in the ERR for larger codebooks can be seen also for the ImUnipen set, from 8% (codebook with 9 shapes) to 14% (codebooks with  $10^3$  shapes). This effect can be explained considering that, as the codebook size increases, the grapheme emission PDFs reside in increasingly higher dimensional spaces that progressively become less and less populated. The distances between the individual handwriting samples increase in relative terms. As a result it becomes gradually more difficult to find a unique threshold distance that separates the sample pairs written by the same person from those written by different persons. Clearly an individualized threshold is needed that depends on the variability in feature space of the handwriting belonging to that particular person. However estimating this within-writer variability using a limited amount of handwritten material is a difficult problem that requires further research. The described dimensionality problem does not significantly affect the distance rankings with respect to a chosen sample and consequently writer identification performance remains essentially stable over a large range of codebook sizes. A slight decrease in the writer identification performance with increasing codebook size can however be notice in fig. 3.

We must point out that the essence of the proposed method does not consist in an exhaustive enumeration of all possible allographic part shapes. Rather, the grapheme codebook spans up a shape space by providing a set of nearest-neighbor attractors for the ink fraglets extracted from a given handwritten sample. The three clustering methods considered in this paper seem to perform this task equally well.

## 7. Conclusion

The use of grapheme emission PDFs in writer identification and verification yields valuable results. Ultimately, writing style is determined by allographic shape variations and small style elements which are present within a character are the result of the writer's physiological make up

as well as education and personal preference. The proposed method proves to be robust to the underlying shape representation used (whether contours or normalized bitmaps), to the size of codebook used (stable performance for sizes from  $10^2$  to  $2.5 \times 10^3$ ) and to the clustering method used to generate the codebook (essentially the same performance was obtained for k-means, ksom1D and ksom2D). More research is needed to improve writer verification performance especially for larger codebooks and experiments must be extended to bigger datasets containing more writers.

## References

- [1] A. Bensefia, T. Paquet, and L. Heutte. Information retrieval-based writer identification. In *Proc. of 7th ICDAR*, volume II, pages 946–950, Edinburgh, Scotland, 3-6 August 2003.
- [2] A. Bensefia, T. Paquet, and L. Heutte. Handwriting analysis for writer verification. In *Proc. of 9th IWFHR*, pages 196–201, Tokyo, Japan, 26-29 October 2004.
- [3] M. Bulacu and L. Schomaker. Writer style from oriented edge fragments. In *Proc. of 10th Int. Conf. on Computer Analysis of Images and Patterns: LNCS 2756*, pages 460–469, Groningen, The Netherlands, 25-27 August 2003.
- [4] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley Interscience, New York, second edition, 2001.
- [5] I. Guyon, L. Schomaker, R. Plamondon, R. Liberman, and S. Janet. UNIPEN project of on-line data exchange and recognizer benchmarks. In *Proc. of 12th ICPR*, pages 29–33, Jerusalem, Israel, October 1994.
- [6] T. Kohonen. *Self-Organization and Associative Memory*. Springer Verlag, Berlin, second edition, 1988.
- [7] U.-V. Marti, R. Messerli, and H. Bunke. Writer identification using text line based features. In *Proc. of 6th ICDAR*, pages 101–105, Seattle, USA, September 2001.
- [8] H. Said, T. Tan, and K. Baker. Personal identification based on handwriting. *Pattern Recognition*, 33(1):149–160, January 2000.
- [9] A. Schlupbach and H. Bunke. Using HMM based recognizers for writer identification and verification. In *Proc. of 9th IWFHR*, pages 167–172, Tokyo, Japan, 26-29 October 2004.
- [10] L. Schomaker and M. Bulacu. Automatic writer identification using connected-component contours and edge-based features of uppercase western script. *IEEE Trans. on PAMI*, 26(6):787–798, June 2004.
- [11] L. Schomaker, M. Bulacu, and K. Franke. Automatic writer identification using fragmented connected-component contours. In *Proc. of 9th IWFHR*, pages 185–190, Tokyo, Japan, 26-29 October 2004.
- [12] L. Schomaker and L. Vuurpijl. Forensic writer identification: A benchmark data set and a comparison of two systems [internal report for the Netherlands Forensic Institute]. Technical report, Nijmegen: NICI, 2000.
- [13] S. Srihari, S. Cha, H. Arora, and S. Lee. Individuality of handwriting. *J. of Forensic Sciences*, 47(4):1–17, July 2002.
- [14] B. Zhang, S. Srihari, and S. Lee. Individuality of handwritten characters. In *Proc. of 7th ICDAR*, volume II, pages 1086–1090, Edinburgh, Scotland, 3-6 August 2003.