Offline Writer Identification using K-Adjacent Segments

Rajiv Jain and David Doermann
University of Maryland, College Park, USA
e-mail: rajivj@cs.umd.edu and doermann@umiacs.umd.edu

Abstract— **This paper presents a method for performing offline writer identification by using K-adjacent segment (KAS) features in a bag-of-features framework to model a user's handwriting. This approach achieves a top 1 recognition rate of 93% on the benchmark IAM English handwriting dataset, which outperforms current state of the art features. Results further demonstrate that identification performance improves as the number of training samples increase, and additionally, that the performance of the KAS features extend to Arabic handwriting found in the MADCAT dataset.**

Writer Identification; Handwriting; Codebook; Local Features; Document Forensics; K-Adjacent Segments

I. INTRODUCTION

Handwriting is a behavioral biometric, which can be used to uniquely identify or verify a document's author and is often admissible as evidence in legal proceedings. This paper addresses the research problem in offline writer identification of matching a handwritten sample from an unknown author to a set of handwriting samples with known authors. This is distinct from the related writer verification problem were one tries to authenticate the author of a handwritten sample.

There are many applications for offline writer identification in criminal forensics such as the analysis of ransom notes, where a document of interest is matched to a set of samples from a given suspect. For example, the FBI recently used handwriting analysis to identify and arrest an individual who mailed handwritten letters with threats along with white powder to political leaders in 2010 [1], demonstrating there continues to be a need for accurate writer identification. The main goal of this research is to design effective writer identification techniques to aid forensic document experts that would otherwise have to manually compare large numbers of documents.

This paper builds on previous work on writer identification, which is outlined in Section II. The approach of modeling character contours using K-adjacent segments (KAS) feature is described in Section III. In Section IV, we present how a codebook is constructed to represent a handwriting model as a vector of code words. Sections V and VI describe the experimental datasets and results.

II. RELATED WORK

Offline handwritten writer identification is a well-studied topic that has seen steady progress in the last ten years. Table 1 summarizes performance of some of the previous literature on this topic.

Author	Dataset Language	# of Writers	% Correct
Srihari [2]	English	900	87
Schlapbach [3]	English	50	94
Schlapbach [15]	English	100	98
Bulacu [5]	English	650	89
Schomaker[4]	Dutch	250	87
Bulacu [8]	Arabic	350	88
Abdi [7]	Arabic	82	90
Chen [14]	Arabic	60	75
He [6]	Chinese	20	80

Table 1: Performance of past writer identification approaches.

Previous research by Srihari [2] established the uniqueness of handwriting by showing that writer verification can be performed at a rate of 96%, and writer identification at a rate of 87%, for a dataset of over 1500 writers. They identify macro features that include intensity changes, slope, and contour features that operate at the paragraph, line and word levels. They also identify micro features, which include gradient, concavity, and structure at the character level. Micro features are shown to significantly outperform the macro features. While the results are impressive, the dataset set contains identical passages from all writers and the approach required manual segmentation, which may not be practical for real world scenarios.

In [3], Schlapbach uses a sliding window to extract nine simple geometric features from a line of text and builds a Hidden Markov Model (HMM) for each of the writers. The author uses the log likelihood output by the Viterbi algorithm to rank users and achieves an identification rate of 94% on 50 writers from the IAM dataset. This work is extended in [15] where Schlapbach uses a Gaussian Mixture Model and achieves an identification rate of 98.5% on 100 writers. Both of these techniques assume perfect line segmentation and require a substantial amount of training. The author uses a 4-fold cross validation on extracted lines during the experiments instead of entire pages, potentially mixing training and testing samples that occurred from the same page. Subsequent papers have used a leave-one-out methodology using between 300-650 writers in the experiments as has been done in this paper.

In [4], Schomaker models character allographs by creating a codebook of connected component contours (CO3) and matching using a bag of features model. In [5] Bulacu models the curvature of characters by introducing the edge hinge, which models the relative angle of two line segments on a character's contour. They combine this method with the CO3 slant features, and run lengths to achieve an identification accuracy of 89% on the IAM dataset for 650 writers. This approach is the current state of the art for writer identification.

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.159

Figure 1: This image illustrates how contours and edges are extracted from connected components in documents.

Recently the authors of [6], [7], [8] extended writer identification to Chinese and Arabic. In [6] the authors use Gabor wavelets for features and HMMs to classify Chinese with moderate success. In [7] and [8] the authors use features similar to those in [5] for Arabic datasets. [7] achieves a recognition rate of 90% on a dataset of 40 writers and [8] achieves an identification rate of 88% when using 5 training samples on a dataset of 350 writers.

While there has been a steady increase in writer identification performance, much of the recent progress has come from combining weaker features together. The edge-hinge feature developed in 2003 continues to be the best performing independent feature [5] and past research shows that similar features that can model properties of the underlying stroke patterns such as curvature and slant perform the best. With a maximum accuracy rate of 80% for a single feature, there is certainly room for improvement. This paper takes the approach of finding a feature that can more accurately model properties of the underlying stroke rather than identifying a novel characteristic of handwriting and combining it with previous features.

III. K-Adjacent Segments (KAS)

K-adjacent segments were introduced by Ferrari, 2008 [9] as a feature to represent the relationship between sets of neighboring edges in an image for object detection. It has since been successfully extended for a number of applications in handwritten text including language identification [10] and text zone detection and classification [11] based on the feature's ability to capture discriminative local stroke information in document images. This work aims to build upon the work in [5] by modeling the character contours using a codebook of KAS features.

In order to extract KAS features from a document image, a set of edges must be found. In color or gray scale images, Ferrari uses a Canny edge detector. Document images are typically binary, so contours that capture the shape and curvature are extracted. A line fitting algorithm is then used to decompose the smooth curves into a set of lines. This process is illustrated in Figure 1.

As the name K-adjacent segments implies, this feature describes any number of K neighboring line segments, but for this paper only 2, 3 and 4 adjacent segments (2AS, 3AS, 4AS) are tested. Any two lines are said to be adjacent if they share an endpoint. The lines that make up the KAS feature must be ordered in a consistent and repeatable manner so that KAS features can be directly compared against each other. The primary line segment is defined as the line with its midpoint closest to the center of the midpoints from all the lines. The remaining lines are ordered by their midpoints from left to right and then top to bottom. Each of the K lines can then be described by the following features:

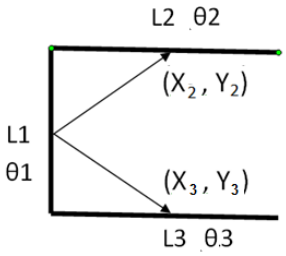

Figure 2: This figure illustrates the segment ordering and features captured for a 3AS, with the primary segment numbered 1.

$$\frac{r^{x_2}}{N_d}, \frac{r^{y_2}}{N_d}, \ldots, \frac{r^{x_k}}{N_d}, \frac{r^{y_k}}{N_d}, \theta_1, \ldots, \theta_k, \frac{l_1}{N_d}, \ldots, \frac{l_k}{N_d} \quad (1)$$

Here (r^x , r^y) define the vector that connects the midpoint of a given segment and the midpoint of the primary segment. Θ and l are the orientation and length of a given segment that makes up the KAS feature. N is the length of the largest segment and is used as a normalization factor to make the feature scale invariant. Features for a 3AS are illustrated in Figure 2. Two KAS features, A and B, can be compared using the distance function D(A,B):

$$D(a,b) = w^r \sum_{i=2}^{k} \left\| r_i^a - r_i^b \right\| + \quad (2)$$

$$w^\theta \sum_{i=1}^{k} \left| \theta_i^a - \theta r_i^b \right| \quad + w^l \sum_{i=1}^{k} \left| \log\left(l_i^a / l_i^b\right) \right|$$

The weights w_r, w_Θ, and w_l can be adjusted to assign more importance to particular features as needed. For this work we use weights of w_r=4, w_Θ=2, and w_l=1 as done in [9] because the segment size is the least stable portion of this feature.

IV. KAS Codebook

A bag of features (BOF) model is used to compare the writers from two documents by converting the KAS features extracted from a document into a vector of code words. We use a clustering technique known as affinity propagation [6] to cluster KAS features from a set of training data to construct a codebook for the BOF model. The input to the affinity propagation algorithm is a distance matrix between all features. Initially all points are considered exemplar clusters and each cluster is combined with neighboring clusters using a message passing algorithm. Two types of messages are passed that represent the responsibility and availability for a given exemplar. The responsibility message, sent from point *i* to point *k*, is defined by *r(i,k)* and represents accumulated evidence for how well suited a point *i* is to be an exemplar for point *k*. The availability message,

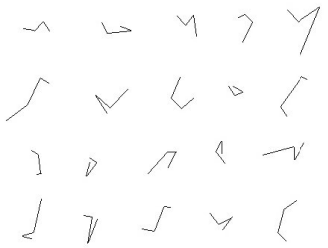

Figure 3: Example of TAS code words.

sent from point k to point i, is defined by $a(i,k)$ and represents how appropriate it would be for point i to represent the exemplar of point k. The equations for both can be seen below.

$$r(i,k) \leftarrow s(i,k) - \max_{k',k' \neq k} \{a(i,k') + s(i,k')\} \quad (3)$$

$$a(i,k) \leftarrow \min\{0, r(k,k) - \sum_{i',i' \neq \{k,i\}} max\{0, r(i',k)\}\} \quad (4)$$

These messages continue to pass until a "preference" threshold is met. It should be noted that unlike K-means this algorithm does not require the number of clusters ahead of time and that the number of clusters is instead controlled by the preference threshold. This approach is used instead of K-means mainly because it converges quicker and is less likely to get stuck in local minima [6].

Once a codebook is constructed, the source document is represented by a feature vector of KAS "code words" present in the document. This feature vector is normalized to sum up to 1 so that it is invariant to the size of the input. The two feature vectors can then be compared by their Euclidean distance. Figure 3 shows examples of the 20 most popular 3AS code words present in the IAM training dataset.

V. DATASETS

In our experiments described in the next section, we use two datasets to show that the performance of the KAS features is not dependent on a particular dataset and extends to multiple languages.

A. IAM Handwriting Dataset

The IAM dataset is made up of samples of handwritten English text from 650 different writers. This dataset has been used by a number of other authors and can be considered the primary benchmark dataset for writer identification [5]. The samples are each scanned as 300 DPI grayscale images. Each of the samples is made up of two or three sentences. The number of pages per writer varies from 1 to 59. 356 writers only provided a single sample, 301 writers provided at least two samples, 159 writers have provided at least three samples and 127 writers provided at least four samples. In order to process this data, each image is preprocessed by binarizing the data using a threshold of 70% and removing connected components with a mass less than 30 pixels in order to remove speckled noise from the binarization. Figure 4 illustrates samples from two different writers.

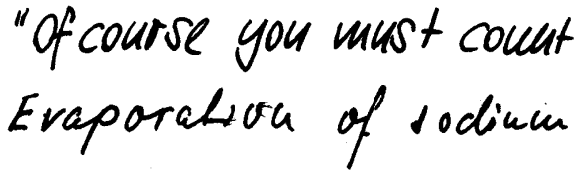

Figure 4: Two writer samples from the IAM dataset.

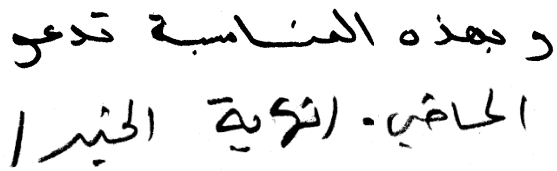

Figure 5: Two writer samples from the MADCAT dataset.

B. DARPA MADCAT/GALE Dataset

The DARPA MADCAT/ GALE dataset is made up of over 10,000 pages of scanned handwritten Arabic text from over 325 writers. A more detailed outlined of this dataset can be found in [10]. The images are sampled at 600 dpi, are already binarized, and are significantly noisier and less structured then the IAM dataset. For example, writers that contributed to this dataset were directed to write at various speeds using various writing instruments (pencils, pens and markers) and to add natural variation into the handwriting samples. Ten documents were chosen randomly from the collection for each of the 302 writers that had at least ten samples. As before, small components were removed to reduce the effect of speckled noise from the binarization process. Connected components with a mass less than 100 pixels were removed here instead of 30 due to the higher resolution of the images. Only one paper [14] has reported results for writer identification on the MADCAT data and they report an accuracy of 75% on similar data with 60 writers. Figure 5 shows samples from two different writers.

VI. EXPERIMENTS AND RESULTS

Four experiments are conducted on the IAM and MADCAT dataset to determine the effectiveness of the KAS feature for writer identification. The KAS feature is implemented in C++ and all experiments are run on 3.0 GHZ Quad Core PC. KAS features can be extracted from a 5 megapixel binary document image in approximately .14 seconds and all the nearest neighbor comparisons for the experiments were completed in less than one second for as many as 650 images.

A. Optimal number of K-adjacent line segments

The first experiment tested 2AS, 3AS, and 4AS for varying codebook sizes on the IAM dataset to determine which codebook size and segment size resulted in the best approach. Samples from the 350 writers who wrote a single page were used for creating the three codebooks and two

Rank	2AS	3AS	4AS
Top1	89.6%	**93.3%**	92.0%
Top2	92.5%	**94.1%**	93.6%
Top5	94.0%	**95.3%**	95.0%
Top10	94.6%	**96.0%**	95.8%

Table 2: This table shows the performance for 2AS, 3AS, and 4AS.

Rank	3 training samples	2 training samples	1 training sample
Top1	**99.8%**	99.2%	98.4%
Top2	**99.8%**	99.6%	99.3%
Top5	**100.0%**	99.9%	99.7%
Top10	**100.0%**	100.0%	99.8%

Table 3: Results on 127 writers from IAM dataset.

samples from the remaining 300 writers were randomly chosen for testing. Recognition was performed using a K-nearest neighbor (KNN) approach in a leave one out configuration similar to [7], meaning each query image was run against 599 other documents with only 1 possible positive match. The results are shown Figure 6 and Table 2.

Figure 6 shows that once the codebook reaches a size of 300 clusters, the identification performance does not increase significantly. The results in Table 2 indicate that 3AS is the best feature representation, with a Top 1 recognition rate of 93.3%. This could be because 2AS does not capture as much information and there were less repeatable 4AS features found in a given document. Still, each of the three approaches is within a couple percentage points and all outperform the state of the art features shown in [5]. While the features are similar to the edge hinge and slant features used previously, the improved performance is likely due to the extra segment found in the 3AS feature, the addition of segment size in the feature representation, and the use of a codebook of clusters rather than coarse quantization. Given the superior performance of the 3AS features, it is used for the remaining experiments.

A close examination of the errors revealed that five of the writers changed their writing style making it very difficult to identify them and resulting in an error rate of 1.7%. The remaining errors are primarily due to too few lines resulting in few KAS features or a large discrepancy between lowercase and uppercase characters between the writing samples. The top 10 score of 96-97% likely represents an upper-bound for this dataset and is consistent with other published results for this dataset.

B. *Improvement from Additional Training Samples*

A second experiment was performed on the IAM dataset to determine if more than one training sample benefits the writer identification performance. The 3AS feature and codebook are reused from the first experiment. Four samples were randomly chosen from 127 writers to create the experimental dataset. Between 1 and 3 of the samples were used for training and the remaining unused samples were used for testing using a KNN search as before. All possible combinations of training and testing were performed. When using more than one sample for training, the distances between the feature vectors found during the nearest neighbor search were averaged for the multiple "trained" samples. This process is repeated using only one dataset for testing and one for training for comparison purposes.

The results indicate that there is a 50% and 87.5% decrease in the error rate when using 2 and 3 training samples respectively. This experiment also shows that the 3AS feature is extremely effective for a population of around 127 writers. These results may be skewed due to the high precision when using just one training sample due to the limited numbers of writers available in the IAM dataset. For this reason, a further review of the relationship between the number of training samples and identification performance is continued on the MADCAT dataset in the third experiment.

C. *Performance on Arabic*

The third experiment again used the 3AS features on the Arabic MADCAT data containing 10 samples for 302 writers. A new codebook was trained from the unused portion of the dataset. This experiment followed the same procedure as experiment 2 in order to take advantage of the 10 pages per writer. A range between 1 to 7 pages were used for training to determine if having more training data was beneficial. Every possible combination of training and testing was run using a KNN search and the results are shown in Figure 7.

Figure 7 shows that increasing the number of training samples is certainly beneficial for identification performance. The top 1 accuracy rises from 80 percent with

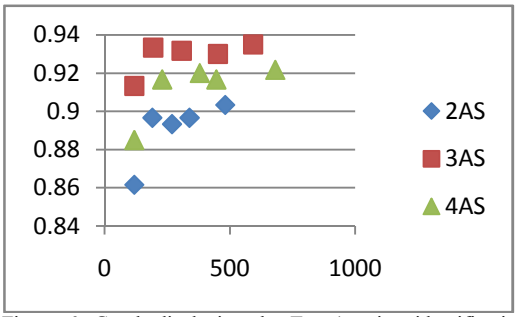

Figure 6: Graph displaying the Top 1 writer identification performance versus the number of clusters for 2AS, 3AS, and 4AS.

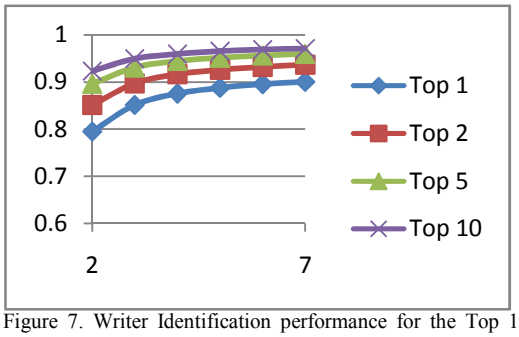

Figure 7. Writer Identification performance for the Top 1, 2, 5, and 10 score versus the number of training samples.

Rank	350 writers	650 writers
Top1	**92.3%**	92.1%
Top2	**93.8%**	93.6%
Top5	**94.8%**	94.5%
Top10	**95.8%**	95.8%

Table 4: Results on the IAM dataset using the Arabic codebook.

1 training sample, to 85% with 2 training samples and 90% with 7 training samples. This outperforms the previously reported rate of 75% in [14] and the 97% top 10 rate exceeded expectations given the amount of noise and variation present in this data.

D. Universality of the codebook

The last experiment uses the Arabic codebook on the IAM dataset. This was done in order to show that a codebook is not unique to a language as shown in [12]. This also allows all 650 writers from the IAM dataset to be tested on for a direct comparison to the state of the art results presented in [5]. Since the additional 300 writers have only published one page, they are retrieved against, but not queried for.

The results in Table 4 show that the codebook is indeed largely independent from language as there is only a 1% drop in accuracy between the English and Arabic codebooks for 350 writers on the IAM dataset. The addition of 300 more writers hardly impacts the accuracy showing that this approach should be robust to many more additional writers. The 92.1% precision on 650 writers represents a 27% reduction in error over the results presented in [5].

VII. CONCLUSION

This paper demonstrates the effectiveness of the KAS feature at modeling handwriting for writer identification. The KAS feature improves upon previous approaches where many features have to be combined and still do not reach the same performance. The identification rate of 93% on the IAM dataset and 90% on the MADCAT dataset outperforms the current state of the art. The experiments show that the codebook is generic between languages and writers so it does not have to be continually recreated and the technique is extremely robust. Additional improvements to the precision could also be made by combining the KAS feature with the approaches presented in past papers. Given the performance of KAS, other local features such as SIFT and shape context should be further examined as they could potentially provide an orthogonal approach to boost the overall performance.

REFERENCES

[1] "Aurora Man Sentenced for Mailing Threats Which Included White Powder to Government Officials", FBI Press Release, http://www.fbi.gov/denver/press-releases/2011/dn012811.htm. Jan. 28, 2011.

[2] S. Srihari, S. Cha, H. Arora, and S. Lee, "Individuality of Handwriting," J. Forensic Sciences, vol. 47, no. 4, pp. 1-17, July 2002

[3] A. Schlapbach and H. Bunke, "Using HMM-Based Recognizers for Writer Identification and Verification", ICFHR, pp. 167-172, Oct. 2004.

[4] L. Schomaker and M. Bulacu, "Automatic writer identification using connected-component contours and edge-based features of upper-case western script". *PAMI*, 26(6):787–798, 2004.

[5] M. Bulacu and L. Schomaker, "Text-independent writer identification and verification using textural and allographic features". *PAMI*, 29(4):701–717, April 2007.

[6] Z. He, X. You, Y. Tang, "Writer identification of Chinese handwriting documents using hidden Markov tree model, Pattern Recognition", Volume 41, Issue 4, April 2008, Pages 1295-1307

[7] M. N. Abdi, M. Khemakhem, and H. Ben-Abdallah, "A novel approach for off-line Arabic writer identification based on stroke feature combination" ISCIS 2009. 24th International Symposium on, Sep 2009, pp. 597 –600.

[8] M. Bulacu, L. Schomaker, A. Brink, "Text-Independent Writer Identification and Verification on Offline Arabic Handwriting," *ICDAR 2007*. pp.769-773, 23-26 Sept. 2007

[9] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid, "Groups of adjacent contour segments for object detection", IEEE Trans. PAMI, No. 30, 2008, pp. 36 – 51.

[10] S. Strassel, "Linguistic resources for Arabic handwriting recognition". In Proceedings of the Second International Conference on Arabic Language Resources and Tools, Cairo, Egypt, April 2009.

[11] J. Kumar et. al. "Shape Codebook based Handwritten and Machine Printed Text Zone Extraction." Document Recognition and Retrieval XVIII, pp. 1-8. January 2011.

[12] G. Zhu, X. Yu, Y. Li, D. Doermann, "Language Identification for Handwritten Document Images Using A Shape Codebook. Pattern Recognition", No. 42, pp 3184-3191, 2009.

[13] B. Frey and D. Dueck, "Clustering by Passing Messages Between Data Points," Science 315, 972-976.

[14] J. Chen, D. Lopresti, and E. Kavallieratou, "The Impact of Ruling Lines on Writer Identification". ICFHR. pp. 439-444, Nov, 2010.

[15] A. Schlapbach and H. Bunke. "Off-line Writer Identification Using Gaussian Mixture Models", ICPR. pp. 992-995. Sep. 2006.

.