Oriented Local Binary Patterns for Writer Identification

Anguelos Nicolaou
Institute of Computer Science and
Applied Mathematics University of Bern
Neubrückstrasse 10
3012 Bern, Switzerland
Email: anguelos.nicolaou@gmail.com

Marcus Liwicki and Rolf Ingolf
Document, Image and Voice Analysis (DIVA) Group
University of Fribourg
Bde des Perolles 90
Fribourg Switzerland
Email: firstname.lastname@unifr.ch

***Abstract*—In this paper we present an oriented texture feature set and apply it to the problem of offline writer identification. Our feature set is based on local binary patterns (LBP) which were broadly used for face recognition in the past. These features are inherently texture features. Thus, we approach the writer identification problem as an oriented texture recognition task and obtain remarkable results comparable to the state of the art. Our experiments were conducted on the ICDAR 2011 and ICHFR 2012 writer identification contest datasets. On these datasets we investigate the strengths of our approach as well its limitations.**

I. Introduction

A. Local Binary Patterns

Local binary patterns (LBP) were broadly popularized in 2002 with the work of Ojala et al [1] as a texture feature set extracted directly on grayscale images. As well demonstrated by Ojala, the histogram of some specific binary patterns is a very important feature-set. LBP are inherently texture features, but they have been used in a very broad range of applications in Computer Vision (CV), many of which exceed the typical texture recognition tasks. In 2004, Ahonen et al [2] used successfully LBP for face recognition. In 2007, Zhao et al [3] extended the operator as a 2D plus time voxel version of LBP, called VLBP, and used them successfully for facial gesture recognition. In 2009, Whang et al [4] combined LBP features with HOG features to address the problem of partial occlusions in the problem of human detection.

B. Writer Identification

While graphology, i.e. the detection of personality traits based on handwriting, has been associated with bad science [5] and has failed to provide experimentally sound significant results [6], handwriting style can be considered an invariant attribute of the individual. Writer identification has traditionally been performed by Forensic Document Examiners using visual examination. In recent decades there is an attempt to automate the process and codify this knowledge in to automated methods. In 2005, Bensefia et al [7] successfully used features derived from statistical analysis of graphemes, bigrams, and trigrams. In 2008, He et al [8] used Gabor filter derived features and in 2010 Du et al [9] introduced LBP on the wavelet domain. Even-though the method of Du uses LBP for feature extraction in writer identification, the similarities end there. Our method makes no assumptions specific to handwriting and treats the problem as a generic oriented binary texture classification problem. The extent to which handwriting contains invariant characteristics of the writer is an open question. While forensic document examiners have been tested in detecting disguised handwriting by Bird et al [10], Malik et al [11] have started to address the issue of different writing styles for automated offline writer identification systems. It remains an open question whether handwriting style can provide us with real biometric markers, invariant to the sample acquisition conditions. By preserving the generic attributes of our method, we can safely avoid addressing many complications that are specific to handwriting analysis and writer detection.

II. LBP Feature Set

Although writer identification seems to require scale invariant features, scale sensitive features might be suited as well. Writers tend to write with a specific size, therefore the scale of the texture tends to be directly dependent on the sampling rate. The task of writer identification is almost always done with respect to a dataset, where the sampling rate is defined or at least known when performing feature extraction. It is feasible and probably worth the effort of resampling all text images to a standard sampling resolution, rather than improvising a scale invariant feature-set. Our feature-set as is the norm, is derived from the histogram of occurring binary patterns.

A. The LBP operator

LBP were defined in [1] as a local structural operator, operating on the periphery of a circular neighborhood. LBP are encoded as integers, which in binary notation would map each sample on the periphery to a binary digit. As can be seen in Fig. 1 and (2), LBP are defined by the radius of the circular neighborhood and the number of pixels sampled on the periphery. The sampling neighborhood $N_{r,b}$ is formally defined in (1).

$$\forall n, \phi : n \in [0..b-1] \land \phi = (n * 2 * \pi)/b$$
$$\forall f(x_1, x_2) : R^2 \implies \{0, 1\}$$

$$N_{r,b}(I(x,y), n) = I(x + sin(\phi) * r, y + cos(\phi) * r) \quad (1)$$

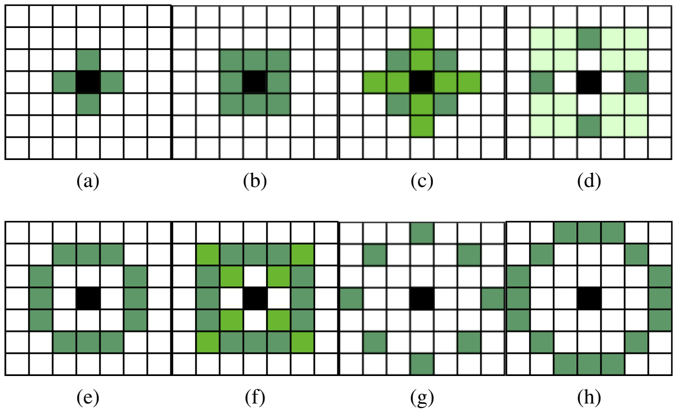
Fig. 1: Indicative LBP operators: $LBP_{1,4}$ (a), $LBP_{1,8}$ (b), $LBP_{1.5,8}$ (c), $LBP_{2,8}$(d),$LBP_{2,12}$ (e), $LBP_{2,16}$ (f), $LBP_{3,8}$ (g), $LBP_{3,16}$ (e). Dark green represents pixels with 100% contribution, green represents pixels with 50%, light green pixels with 25%, and black is the reference pixel.

$$\begin{aligned} LBP_{r,b,f}(x,y) = f(N_{r,b}(I(x,y),n) * 2^n, I(x,y))+ \\ f(N_{r,b}(I(x,y),n-1) * 2^{n-1}), I(x,y)) + ... \\ ... + f(N_{r,b}(I(x,y),0) * 2^0, I(x,y)) \end{aligned} \quad (2)$$

When defined on grayscale images, LBP are obtained by thresholding each pixel on the periphery by the central pixel. Because we worked on binary images as input, a lot more operations than greater or equal (thresholding) were possible as a binary operation. We generalized our definition of the LBP in (2), to consider the boolean operator marked as f a third defining characteristic of the LBP operator $LBP_{r,b,f}$ along with the radius r and the number of samples b.

We took into account several factors for selecting the appropriate LBP binary operator. In what concerns the bit count, a bit-count of 8 presents us with many benefits. Implementation wise, the LBP transform is an image that uses one byte per pixel. Its histogram has 256 bins providing a high feature-vector dimentionality and good discriminative properties. Additionally, containing the distinct LBP count to 256, guaranties highly representative sampling in relatively small surfaces of text.

B. The LBP function

While LBP are traditionally derived from grayscale images, when dealing with text, its better to use binarized text images as input, thus avoiding all information coming from the text background. We considered many different binary operations and chose the binary operator "equals" (3) as $f()$ in (2).

$$f(x_{ceter}, x_{periphery}) = \begin{cases} 1 & : x_{ceter} = x_{periphery} \\ 0 & : x_{ceter} \neq x_{periphery} \end{cases} \quad (3)$$

"Equals" as a boolean function on an image means true for any background pixel in the peripheral neighborhood of a background pixel, true for any foreground pixel in the peripheral neighborhood of a foreground pixel, and false for everything else. When using the "equals" function as the binary function in a 8 bit-count LBP, all pixels with only foreground and only background have an LBP value of 255. By suppressing (ignoring) the 255 bin, we make the LBP histogram surface invariant. All occurrences left in the histogram represent the pixels in the border between foreground and background. The core of the feature set comprises of the 255 histogram bins normalized to a sum of 1. This normalization renders the features derived from the histogram invariant to the number of signal pixels in the image.

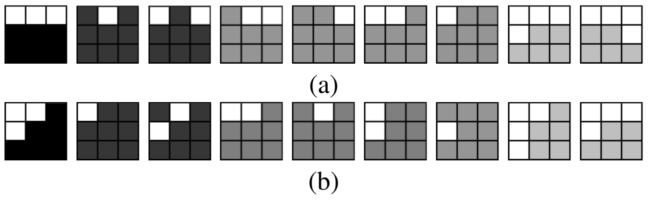
Fig. 2: LBP edge patterns. In (a) the top-edge contributing patterns and in (b) the top-left edge contributing patterns can be seen. Contribution: black 100%, dark gray 50%, gray 25%, and light gray 12.5%

C. Redundant Features

Having the normalized 255 bins from the histogram as the core of the feature set, we calculate some redundant features that will amplify some aspects of the LBP we consider significant in the writer identification task. Our goal is to have a feature-set discriminative enough to work well with naive classifiers such as nearest neighbor or, even more, classify writers by clustering the samples without any training.

The first redundant feature group we use, is edge participation. We consider each pattern to have a specific probability of belonging to an edge of a specific orientation; from now on we call that contribution. The sum of the number of occurrences of each pattern, multiplied by its contribution factor makes up the oriented edge occurrences. In Fig. 2a all top-edge patterns can be seen along with their probability, in 2b we can see the patterns of the top-left-edge patterns and their probabilities which are derived from the top-edge patterns by rotating them counter-clock-wise. By rotating the contributing patterns of the top-edge, we can obtain the contributing patterns of all eight edge-orientations. We also add the more general edge-orientations: horizontal, vertical, ascending, and descending as separate features which are calculated as the sum of the respective pair of edge-orientations. In the end we calculate the two final aggregations of perpendicular and diagonal, which are the sum of horizontal and vertical and respectively ascending and descending. In total we obtain 14 edge-features, which we then normalize to a sum of 1. One of our aims when introducing these redundant features is to enhance characteristics that have been associated with writer identification such as text slant.

The second redundant feature-group we implemented are the rotation invariant hashes. We grouped all patterns, so that each pattern in a group can be transformed in to any other pattern in that group by rotating. When having an 8 sample LBP, the distinct rotation invariant patterns are 36 in total [1]. Some pattern groups contain only one pattern eg. pattern 0, while other groups contain up to 8 patterns, such as all one bit true patterns 1,2,4,8,16,32,64,128. We took the number of occurrences for each group in the input image and normalized them to a sum of 1, thus providing 36 rotation invariant features. A complementary feature-group to the rotation invariant patterns is what we named rotation phase. For each group of rotation invariant features, we took the minimum, with respect to the numeric value, pattern in the group and designated it as group-hash. The number of clockwise rotations each pattern needs to become its groups-hash, is what we call the rotation phase. By definition, the distinct phases in an LBP image, are as many as the number of samples of the LBP. The frequency of all phases normalized to the sum of 1, provides us with 8 more redundant features that are complementary to the rotation invariant hashes.

A third group of redundant features we introduced to our feature-set is what we called beta-function as defined in (4) along with the bit-count of every pattern.

$$\begin{aligned}
&\forall n \in [1..bitcount] \\
&\forall lbp \in [0..2^{bitcount-1}] \\
&d(lbp,n) = \begin{cases} 1 & : \text{bit } n \text{ is set in } lbp \wedge \\ & \quad \text{bit } n-1 \text{ is not set in lbp} \\ 0 & : otherwise \end{cases} \\
&\beta(lbp) = \sum_{n} d(lbp,n)
\end{aligned} \tag{4}$$

When the sample count is 8, the β function, has up to 5 distinct values. The histogram of the β function (5 bins) normalized to a sum of 1 and the histogram of the bit-count of every pattern normalized to 1 as well, are the last redundant feature-group we defined. The β function becomes an important feature when the LBP radius is greater than pen stroke thickness. In those situations, e.g. a β count of one, would indicate the ending of a line, and a β count of three or four would indicate lines crossing.

If we put it all together, we have 255 bins of the histogram, plus 36 rotation invariant features, plus 8 rotation phase features, plus 14 edge-features, plus 5 β function features, plus 9 sample-count features, makes a total of 327 features; these are the proposed feature-set. The redundant features make the features well suited for naive classifiers. By setting the 255 histogram bin to 0, the feature set ignores all non signal areas in the image. The normalization of all bins to a sum of 1, as well as the nullification of the last bin, renders our feature set invariant with respect to non signal areas (white).

D. The Classifier

Once we transform a given set of images into feature vectors, we can either use them as a nearest neighbor classifier or perform clustering on them. While clustering seems to be a more generic approach, it is constrained by the need to process all samples at the same time. Such a constraint makes the clustering approach very well suited for research purposes but hard to match any real world scenarios. The construction of the classifier consists of four steps. In the first step, we extract the image features. In the second step, we rebase the features along the principal components of a given dataset by performing principal components analysis. This step might, in a very broad sense of the term, be considered training because our method acquires information from a given dataset. In the third step we scale the rebased features by a scaling vector which was defined by evolutionary optimization on the train-set. The optimization process is also performed on a given dataset and should also be considered as a training stage. While it is not required, it makes more sense that both training steps are performed on the same dataset. The fourth and last step is to calculate the L1 norm on the scaled and rebased feature vectors. Steps two and three can be combined in to a linear operation on the feature space and in many aspects should be viewed as a statistically derived heuristic matrix. Our classifier, as was implemented, has two inputs, a subject dataset and a reference dataset. The output consists of a table where each row refers to a sample in the subject dataset and contains all samples in the reference dataset ranked by similarity to the specific sample. When benchmarking classification rates of our method, we can simply run our classifier with an annotated dataset as both object dataset and reference dataset. In this case, the first column contains the object sample and the second column contains the most similar sample in the dataset other than its self. The rate at which the classes in the first column agree to the classes in second column, is the nearest neighbor classification rate.

E. Scale Vector Optimisation

Describing in detail the optimization process of the scaling vector would go beyond the scope of this paper. In brief we optimized using an evolutionary algorithm. We used as input the 125 most prominent components of the features and the id of the writer of each sample. We optimized using the ICHFR 2012 writer identification competition dataset [13] which contains 100 writers contributing 4 writing samples each. Individuals of the algorithm were modeled as vector of continuous scaling factors for each feature in the feature space. The fitness function was based on the classification rate a nearest neighbor classifier obtains when the feature space is scaled by each individual. The stoping criteria was set to 2000 generations, and each generation had 20 individuals. Suitable parents were determined by the rank they obtained in the generation.

III. Experimental Procedure

In order to have a proper understanding of our methods performance, its robustness, and its limitations, we conducted a series of experiments. We used two datasets for our experiments: the dataset from the ICDAR 2011 writer identification contest [12], hereafter 2011 dataset and the dataset from the ICHFR 2012 writer identification challenge [13], hereafter 2012 dataset. The 2011 dataset has 26 writers contributing samples in Greek, English, German, and French with 8 samples per writer. The 2012 dataset has 100 writers, contributing samples in Greek and English with 4 samples per writer. While the 2011 dataset was given as the train set for the 2012 contest, we used them in the opposite manner. In order to avoid overfitting during the optimization step, we deemed the ”harder” dataset, containing more classes and less samples per class, was better suited for training.

A. Performance

As previously described, our method consist of four stages: feature extraction, principal components analysis, scaling vector optimization, and L1 distance estimation. Steps two and

TABLE I: Performance Results. Various modalities of our method's results on the 2011 dataset [12] and state of the art methods performance for reference

NAME	Nearest Neighbor	Hard Top-2	Hard Top-3	Hard Top-5	Hard Top-7	Soft Top-5	Soft Top-10
Tsinghua	99.5%	97,1%	NA	84.1%	44.1%	100%	100%
MCS-NUST	99.0%	93.3%	NA	78.9%	38.9%	99.5%	99.5%
Tebessa	98.6%	97.1%	NA	81.3%	50.0%	100%	100%
No PC, No train	**96.63%**	87.02%	79.33%	63.94%	28.84%	98.56%	99.04%
PC, No train	98.56%	91.35%	84.62%	68.27%	34.62%	98.56%	98.56%
PC, Train	**98.56%**	95.19%	91.83%	84.13%	50.48%	99.04%	99.04%

three require a training dataset, while steps one and four are totally independent of any data. In TABLE I analytical scores of our method in various modalities can be seen. Apart from the nearest neighbor accuracy we also add the hard TOP-N and soft TOP-N criteria [12], [13]. The soft TOP-N criterium is calculated by estimating the percentage of samples in the test set that have at least one sample of the same class in their N nearest neighbors. The hard TOP-N criterium is calculated by estimating the percentage of samples in the test set that have only samples of the same class in their N nearest neighbors. More in detail, in TABLE I we can see various versions of our method and their performance as well as some state of the art methods for reference. Methods Tsinghua, MCS-NUST and Tebessa [14] are the top performing methods from the ICDAR 2011 writer identification contest. We must point out that our method had a vastly superior train set, consisting of 400 samples, and we had access to the test set while working. Our method has two parts that were optimized on our train set, the 2012 dataset. The first is the principal components of the train set and the second is the scaling of the feature space. No PC, No train is the raw feature space without any training, just the features in an L1 nearest neighbor setup. PC, No train is the feature space rebased along the principal components of the the train set in a L1 nearest neighbor setup. PC, Train is the feature space rebased along the principal components of the the train-set and scaled along the optimized vector in a L1 nearest neighbor setup. As we can see our method almost reaches the overall performance of the state of the art when it incorporates the full trained heuristics but it also provides very good results in its untrained form.

B. Qualitative Experiments

Apart from providing a comprehensive accuracy score that is comparable to other methods, in order to describe the strength and limitations of our method, we performed a series of experiments that simulate frequently appearing distortions to the data.

1) Rotation: Text orientation, is a text image characteristic that is definitely affected by the writer. Under controlled homogeneous experimental conditions of data acquisition, text orientation should depend only on the writer. Quite often in real life scenarios we have no way of knowing whether an image has been rotated or not and to which extent. One of the important characteristics of a writer identification system is the robustness against moderate rotation. We address this issue by an experiment where we try to recognize samples of a dataset with rotated versions of the database. More specifically we took the 2012 dataset and we rotated its samples by 1° from -20° to 20°. We obtained our measurements by classifying the original 2012 dataset with the the rotated versions. In Fig. 3 the rotation sensitivity of our method can be demonstrated. Two different measurements can be seen. The first one, noted as Sample Self Recognition, is the the nearest neighbor including the test sample. Sample Self Recognition rate will be by definition 100% when no rotation occurs. The second measurement, marked as Nearest Neighbor is the accuracy of nearest neighbor excluding the first occurrence. Nearest Neighbor is by definition the accuracy when no rotation occurs. As can be seen in Fig. 3 our method demonstrates some rotation tolerance from -5° to $+5^\circ$ with sustainable accuracy rates, but performance drops significantly beyond this limit[1]. It is also worth noticing the fact that -1° and $+1^\circ$ rotations perform slightly worst than -2° to $+2^\circ$; a possible explanation for this could be aliasing phenomena.

2) Downsampling: As we stated previously, in most real world scenarios, the sampling resolution will be known to a writer identification system, but not always controlled as sometimes the data are acquired by external sources or at different times. We devised an experiment that demonstrate the behavior and limitations of our method in what concerns the resolution. We took the ICDAR 2011 Writer Identification dataset and rescaled it to various scales from 100% down to 10%. As can be seen in Fig. 4 we obtained three measurements. The first, marked as Self Recognition Unscaled Sample, is the nearest neighbor when classifying the initial dataset with the subsampled dataset as a database. The second, marked as Nearest Neighbor Unscaled Sample, is the second nearest neighbor when classifying the initial dataset with the subsampled dataset as a database. We presume that the first nearest neighbor will always be the same sample in different scales and therefore disregard it for this measurement. The third measurement, named Nearest Neighbor Scaled Sample, is the accuracy of the second nearest neighbor when classifying the scaled dataset with the scaled dataset a database. The first two measurements describe the sensitivity our method has in comparing samples of different sampling resolution and therefore scale as well, while the third measurement demonstrates how well our method would work on datasets of lower resolution. We should also point out that the optimization process was performed on the original resolution. As we expected and can be seen in Fig. 4, we find that our method has no tolerance in comparing samples from different sampling rates. We can also

[1]samples rotated by more than 5° could be manually corrected during sample aquisition

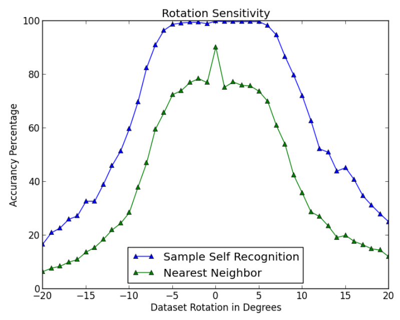

Fig. 3: Rotation Sensitivity

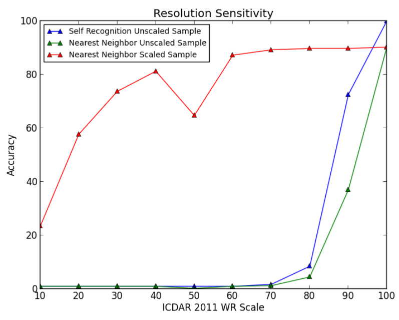

Fig. 4: Resolution/scale sensitivity

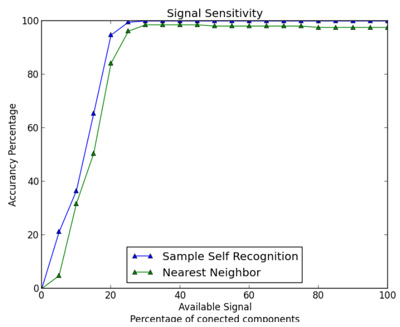

Fig. 5: Grapheme quantity sensitivity

conclude that our method has tolerance to lower than standard resolutions, but benefits mostly from higher resolutions. The out of the norm measurement in Nearest Neighbor Scaled Sample posed us with a puzzle. The most probable explanation is that it is related to aliasing but is worth investigating more.

3) Removing Graphemes: A very important characteristic of writer identification methods is how much text is required to in order to reach the claimed accuracy. We conducted an experiment to answer specifically this question. Our strategy was to create group datasets that vary only on the amount of signal (text) and then compare results on these datasets. As the primary dataset we took the ICDAR 2011 writer identification dataset, because it provides us with relatively large text samples. In order to quantify the available signal, we took the 2011 dataset and for each image in the dataset, we produced 20 images with different amounts of connected components from the original image. Due to the very high locality of our feature set, the fact that we removed connected components instead of text lines should be negligible and at the same time it gave us quite higher control over the signal quantity. As can be seen in Fig. 5 the results are quite surprising. Instead of having a gradual drop in performance, the performance is unaffected down to 30% of the graphemes, bellow that point, performance drops linearly.

4) Writer vs Writing Style: We submitted an earlier version of our method to the SigWiComp2013 competition. The goal of the writer identification part of the competition, is to measure the performance of writer identification systems, when the handwriting style has been altered. A sample dataset was made available by the organizers of the competition. The dataset contained 55 writers contributing 3 text samples each and each sample written a different writing style. Having access to the sample dataset, we performed a simple experiment to determine whether our features encapsulate writer biometrical information or simply the writing style. We separated the dataset of 165 samples in to left and right halves. We then performed a pair matching of the left halves to the right halves based on the nearest neighbor classification. We obtained two measurements, first the percentage of left-samples having an assigned right-sample written by the same writer (55 classes), and second the percentage of left-samples having the specific sample's complementary right-half as the nearest neighbor (165 classes). The writer identification rate was **87.27%**, while the specific sample recognition rate was **86.06%**. By definition the writer identification rate is greater or equal to the sample recognition rate. We performed a one tail t-test on the results on 165 sample-classifications and obtained a p-value of 0.3734, which by all standards make the recognition-rates indistinguishable. This experiment indicates that for our method any two samples written in different writing styles are as different regardless of whether they were written by the same individual or not. From a forensic perspective, these measurements imply that our method does not distinguish between disguised writing style and natural writing style.

IV. Discussion

In this paper we presented a powerful feature set that summarizes any texture on a binary image as a vector of 327. We use our feature extraction method to produce a database from any given dataset with handwriting samples and use it as a nearest neighbor classifier. In order to improve our classifier we performed PCA on a specific dataset and linearly transformed the feature space. We also scaled the features by a scaling vector in order to increase the impact of the features that contribute to correct classifications on our test set. Both these improvements can be combined in to single matrix with which we multiply all feature vectors. This single matrix should be viewed as a heuristic matrix statistically derived from the 2012 dataset. It is also valid to think of this matrix as the result of a supervised learning process. The idea is to calculate this matrix once per type of texture we want to classify. In the context of western script handwriting, we obtained the matrix from the 2012 dataset and used it in benchmarking our method on the 2011 dataset, our qualitative experiments, and our submission to SigWiComp2013 [11]. When comparing the experimental results to the state of the art, we can not obtain a perfectly fair comparison. The state of the art methods participated in a blind competition with a very small train-set, although we could maybe assume that participants had access to larger third-party datasets as well. Since datasets of competitions are published after the competitions, the only way to have a perfectly fair comparison to the state of the art is to participate in those competitions. A comparison of the untrained classifier (96.63%) to the state of the art (99.5%) is quite unfair towards our method. On the other hand, a comparison of our trained classifier (98.56%) to the state of the art (99.5%) is a bit unfair towards the state of the art. In the authors opinion, a fair comparison would be a lot closer to the trained classifier than to the untrained. The performance of the untrained classifier demonstrates clearly the potency of our feature set. The qualitative experiments were not performed with forensics in mind, except for the last one, writer vs writing style. In writer vs writing style we tried to determine the extent to which our feature set can deal with disguising writers; the quick answer is, no our method can not deal with disguising writers. There are many subtleties in the conclusions that can be drawn from the writer vs writing style experiment about what phenomena is that our features model. One could even say that our method is more about texture similarity than about writer similarity; assuming there are biometric features in handwriting, the proposed feature set does not seem to encapsulate them. From a software engineering perspective the approach of treating writer identification as a distance metric instead of a classifier [12] seems more efficient and modular, it allows for simplification and standardization of benchmarking. The fact that the proposed features encapsulate no structural information what so ever, makes them a very good candidate for fusion with other feature sets.

Acknowledgment

The first author of this paper would like to thank Georgios Louloudis for his precious insights on the subject of writer identification and performance evaluation. The first author would also like to thank Muhammad Imran Malik for his effort and assistance in the participation of this method on the SigWiComp2013 competition.

References

[1] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 7, pp. 971–987, 2002.

[2] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face recognition with local binary patterns," in *Computer Vision-ECCV 2004*. Springer, 2004, pp. 469–481.

[3] G. Zhao and M. Pietikainen, "Dynamic texture recognition using local binary patterns with an application to facial expressions," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 6, pp. 915–928, 2007.

[4] X. Wang, T. X. Han, and S. Yan, "An hog-lbp human detector with partial occlusion handling," in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 32–39.

[5] G. A. Dean, I. W. Kelly, D. H. Saklofske, and A. Furnham, "Graphology and human judgment." 1992.

[6] A. Furnham, "Write and wrong: The validity of graphological analysis," *The Hundreth Monkey and Other Paradigms of the Paranormal*, pp. 200–205, 1991.

[7] A. Bensefia, T. Paquet, and L. Heutte, "Handwritten document analysis for automatic writer recognition," *Electronic letters on computer vision and image analysis*, vol. 5, no. 2, pp. 72–86, 2005.

[8] Z. He, X. You, and Y. Y. Tang, "Writer identification of chinese handwriting documents using hidden markov tree model," *Pattern Recognition*, vol. 41, no. 4, pp. 1295 – 1307, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320307004037

[9] L. Du, X. You, H. Xu, Z. Gao, and Y. Tang, "Wavelet domain local binary pattern features for writer identification," in *Pattern Recognition (ICPR), 2010 20th International Conference on*. IEEE, 2010, pp. 3691–3694.

[10] C. Bird, B. Found, and D. Rogers, "Forensic document examiners skill in distinguishing between natural and disguised handwriting behaviors," *Journal of forensic sciences*, vol. 55, no. 5, pp. 1291–1295, 2010.

[11] M. I. Malik, M. Liwicki, L. Alewijnse, W. Ohyama, M. Blumenstein, and B. Found, "Signature verification and writer identification competitions for on- and offline skilled forgeries (sigwicomp2013)," in *12th Int. Conf. on Document Analysis and Recognition*, Washigton, DC, USA, 2013, p. n.A.

[12] G. Louloudis, N. Stamatopoulos, and B. Gatos, "Icdar 2011 writer identification contest," in *Document Analysis and Recognition (ICDAR), 2011 International Conference on*. IEEE, 2011, pp. 1475–1479.

[13] G. Louloudis, B. Gatos, and N. Stamatopoulos, "Icfhr 2012 competition on writer identification challenge 1: Latin/greek documents," in *Frontiers in Handwriting Recognition (ICFHR), 2012 International Conference on*. IEEE, 2012, pp. 829–834.

[14] D. Chawki and S.-M. Labiba, "A texture based approach for arabic writer identification and verification," in *Machine and Web Intelligence (ICMWI), 2010 International Conference on*. IEEE, 2010, pp. 115–120.