Classification Approaches in Off-Line Handwritten Signature Verification

BENCE KOVARI, BENEDEK TOTH, HASSAN CHARAF
Department of Automation and Applied Informatics
Budapest University of Technology and Economics
1111. Budapest, Goldman György tér 3.
HUNGARY
beny@aut.bme.hu http://www.aut.bme.hu/signature

Abstract: - The aim of off-line signature verification is to decide, whether a signature originates from a given signer based on the scanned image of the signature and a few images of the original signatures of the signer. Although the verification process can be thought to as a monolith component, it is recommended to divide it into loosely coupled phases (like preprocessing, feature extraction, feature matching, feature comparison and classification) allowing us to gain a better control over the precision of different components. This paper focuses on classification, the last phase in the process, covering some of the most important general approaches in the field. Each approach is evaluated for applicability in signature verification, identifying their strength and weaknesses. It is shown, that some of these weak points are common between the different approaches and can partially be eliminated with our proposed solutions. To demonstrate this, several local features are introduced and compared using different classification approaches. Results are evaluated on the database of the Signature Verification Competition 2004.

Key-Words: - signature verification; off-line; classification, shape descriptor, neural network

1 Introduction

The aim of off-line signature verification is to decide, whether a signature originates from a given signer based on the scanned image of the signature and a few images of the original signatures of the signer. Unlike on-line signature verification, which requires special acquisition hardware and setup, off-line signature verification can be performed after the normal signing process, and is thereby less intrusive and more user friendly. On the other hand, important information like velocity, pressure, up- and down strokes is partially lost.

In the past decade a bunch of solutions has been introduced, to overcome the limitations of off-line signature verification and to compensate for the loss of accuracy. However when tested against skilled forgeries, even the best systems deliver worse equal error rates than 5%, in contrast with a human expert, who is able to do the distinction with an error rate of 1%. To break this barrier it is essential to identify, understand and compensate for the different sources of error in the algorithms. This paper presents a solution to address the problem of improvement and thereby possibly break the 5% barrier. Typical signature verification approaches consist of 3 main phases. First they extract some features from the images of signatures, then they compare them and finally, they use some kind of classifier to decide, whether a given signature is an original or a forgery. This paper concentrates on the final phase of signature verification. In the following section several existing signature verifiers are introduced, with a special emphasis on neural network based classification. Then we summarize the classification problems, occurring when dealing with signatures, and propose solutions for them. In the second part of this paper a complete neural network based classification method is introduced to demonstrate, how some of the limitations of off-line signature verification can be overcome. Finally experimental results are presented and used to evaluate the goodness of several different features.

2 Related work

Typically signature verifiers take advantage of different general properties (global features) of the signature and use them as an input for different simple classifiers [1], [2], [3]. In [4] a more complex approach can be seen, by creating a two-stage neural network classifier. Different groups of features are defined and separate MLP (multilayer perceptron) classifiers are applied to them. These MLPs are relatively simple, containing only one hidden layer. Learning is not done through backpropagation, but through the ALOPEX algorithm, which allows the network not to get "stuck" in local minima or maxima of the response function. The MLPs have a relative wide range of input parameters, in order: 16, 96, and 48 variables. The inputs of the first network are the global features of the signature. The second takes a simplified representation of the signature as an input, by creating a 12*8 grid and measuring the intensity values in each grid cell. The third network processes texture information. The output layer contains a single neutron, delivering a response value between 0 and 1 representing the similarity between the actually measured signature, and the training set. These output

values are then processed by an RBF to make the final decision.

A similar approach is taken in [5]. They use global features (height-width proportion, middle point, corner points, etc.), and grid features as inputs. Tests are performed both by using simple MLP classifiers and by using SVMs. SVMs were tested with kernels with linear, polynomial, and radial basis function. The latter seemed to deliver the best results with an average error rate of 7-8% compared to the 16-22% error rates measured when using MLPs.

Another interesting approach can be found in [6]. It utilizes CGS vectors (originally developed for character recognition) to extract global features. The main idea here is, to assign a 1024 bit long binary vector to each image and compare these vectors in the later phases. Images are divided into 4x8=32 segments, and information (like concavity, gradient, structural properties) is encoded into the vector for each segment

These vectors are then compared by several algorithms operating with vector distances. In this scenario, the SVM based solution performs poorly, with an average error rate of 46% while a Naïve Bayes classifier achieved error rates between 20% and 25%

3 Processing model

Since the first survey paper [7] (dating back to 1989) several surveys like[8][9][10] and journal special issues like[11] have been dedicated to the comparison and evaluation of signature verification methods. While trying to give a balanced overview of the field they all face the same problem. Namely that the evaluation of signature verification algorithms, as for many pattern recognition problems, raises several difficulties, making any objective comparison between different methods rather delicate and in many cases impossible [9]. In the following sections we are going to use a generalized model of signature verifiers (Fig. 1.) as a base of our discussion.

The majority of signature verification methods can be divided into five main phases: acquisition, preprocessing and feature extraction, processing and classification (although these steps are not always separable). In the off-line case data acquisition means simply the scanning of a signature. This is followed by preprocessing whereby the images of signatures are altered (cropped, stretched, resized, normalized etc.) to create a suitable input for the next phase. The next step is feature extraction, the process of identifying characteristics, which are inherent to the particular person. The processing phase is mainly based on a single comparison algorithm, which is able to calculate the distance function between signature pairs. Using these results, the classification phase is able to make a decision, whether to accept or reject the tested signature. This coarse separation of processing phases is already an extension to [12] which does not separate feature extraction from processing and classification. In our model even further extensions will be necessary to allow a better control of the dataflow. In the following subsections these 5 steps will be explained in detail and matched to the steps of several other signature verifiers ([13] [1] [14][15] [16] [17] [18]). Fig. 5. summarizes this processing model and shows the data flow between components.

.

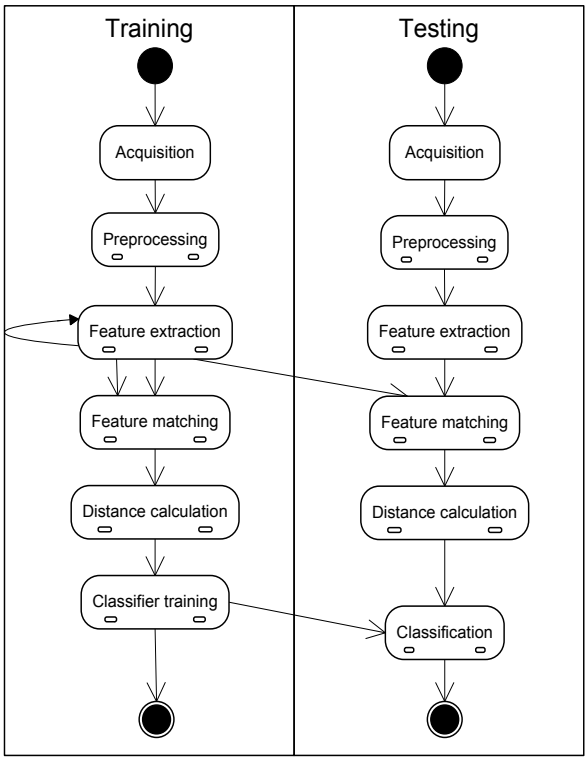

Fig. 1 Generalized view of an off-line signature verification system

The model combines the advantages of most previously introduced systems in the literature. Similarly to [16] and [18] it clearly isolates the path of the training set (original signatures) from the path of test signatures while traversing the same modules of the system. It incorporates the 5 phases of verification, where processing was further broken into feature matching and distance calculation steps to improve support for modularity. It is also interesting to note, that the two steps of the classifier were also partitioned: the classifier is trained during the training phase, only using the original signatures from the reference database, and classification decisions about test signatures will be made based on the training during the testing phase.

The model allows a direct top down data flow without ever referencing a previous module. This makes a loose coupling and individual testing of the

components possible.

As noted in the previous section, individual phases can consist of multiple sequentially or parallel coupled subcomponents, therefore five of the six states are marked as composite states.

4 Feature extraction

Feature extraction is with great certainty the most ambiguous processing phase. First let us make some definitions clear. "An image feature is a distinguishing primitive characteristic or attribute of an image. Some features are natural in the sense that such features are defined by the visual appearance of an image, while other, artificial features result from specific manipulations of an image […] Image features are of major importance in the isolation of regions of common property within an image (image segmentation) and subsequent identification or labeling of such regions (image classification)." [18]. Therefore feature extraction is the location and characterization of features, and generally it should not be confused with the later processing phases. Contrary to preprocessing which is defined sequence of transformation steps altering the original images, feature extraction is a set of (usually) independent functions returning a characteristic feature set for their input image. Several systems take advantage of multiple features to improve the quality of the input provided for distance calculations and classifiers.

In this section three different features (baseline, skew and loops) are introduced. Although some of them may seem quiet intuitive, their exact definition and extraction is essential for later processing phases. It should also be mentioned that the choice of feature types here is arbitrary. Any feature type, like those introduced in [19] or [20] could be used for comparison purposes.

4.1 Baseline

Based on the algorithm described in [19], the upper and lower bounding envelopes (baselines) and vertical and horizontal projections are compared.

Upper (/lower) Baselines are defined as a curve consisting of the first black pixels from the top (/bottom) in each column of the image. In some papers they are also referred to as parts of the "enclosing envelope". Horizontal (/vertical) projections are defined by the number of black pixels in each column (/row) of the image. Baselines and horizontal projections can be thought of as functions of x while vertical projection is a function of y. Thereby we defined 4 different functions, which can be later compared to analyze their similarity.

To improve the extraction speed and precision, instead of the original image a thinned, vectorized representation of the signature is considered, as described in [20].

For example, to locate the lower baseline, instead of scanning upwards for each column of the picture the lowest strokes are considered as a starting point for the scanning process and scanning is done downwards. In a typical 300dpi image a line width is about 8-10 pixels, thereby using this algorithm it is usually sufficient to scan 5-6 pixels to locate the lowest valuable pixel of a signature in a given column. An example of such a baseline can be seen in Fig.2.

Fig.2. Baseline of a signature

4.2 Skew

The very first step of acquiring the skew of a signature is to define what skew stands for. Using the knowledge gained in a consultation with a handwriting expert a definition was created which presumably would be helpful at the comparison: the skew information consist of a set of straight lines, where each line represents an imaginary foundation of a component, which can be regarded as an autonomous element of the signature. This definition allows us to assign skew information to the gaps between signature elements, and according to [21] those spaces are just as peculiar as any other feature of a signature.

Our first approach was a naive algorithm, where the goal was to obtain the lower contour of the signature, which was done by starting vertical scan lines from the bottom left corner of the image and store the lowest pixel which was part of the signature. To determine whether a pixel is representing paper or ink a function was created, which not only used the pixel itself but its environment as well to give the best result. This was necessary because of the different kind of images with different amount of noise on them.

After obtaining the lower contour a line to each separable segment was fitted using linear regression. Separable segments are parts divided by a horizontal gap. The resulting lines were often convincing enough, but of course this algorithm has its drawbacks: the most important problem was that it frequently had trouble recognizing the separate parts of a signature, in fact it could only distinguish two segments when a significant horizontal gap existed between them, but unfortunately this was not true for most of the signatures in our database. Another remarkable disadvantage was that it

used information only from the image itself, while by the time there was additional information available [22] that could definitely increase the reliability of the algorithm, like stroke positions.

Using the experience gained so far, we came up with a new approach, whose fundamental element became components. Components are parts of the signature that could and should be treated as an independent part, having their own skew information. Typically a component is a part of the name (first name, last name) or an accent. Experiments showed that accents should have their own skew information, as they are a distinctive feature of signatures.

The resulting lines seemed to almost perfectly represent our definition of skew (Fig.3.); hence it was time to examine whether they could be used to make a distinction between forged and genuine signatures.

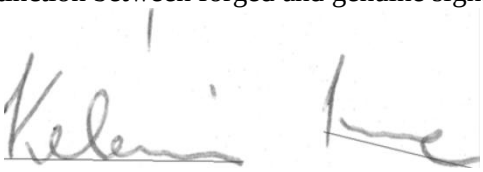

Fig.3. Three skew lines obtained by our algorithm

It is important to note, that in some cases more separable segments were found than expected, but since those extra segments were found on almost all genuine signatures of the given signer, they should be considered as a feature of the signature and not as an error.
Representing the skew information with numeric values (angle, length, position) and examining these values for both forged and genuine signatures has shown that our skew lines were adequate features to verify signatures, however alone they are not sufficient to unquestionably separate valid and forged ones.

4.3 Loops

Loops in our definition are connected regions in the image which are fully enclosed by "signature" pixels.

This definition implies the following 3 important properties:

First: pixels should be unambiguously classified as some belonging to the background ("paper" pixels) or belonging to the signature ("signature" pixels). This is currently done by testing the color components of a pixel against some thresholds.

Second: The region must be connected. Although it sounds logical at the first glance, this is against the traditional definition of a loop, which can be interrupted by other lines. This simplification however allows us a much faster processing of the image.

Third: using fully enclosed regions showed to be an unrealistic target. Because of errors of the pen, and sometimes because of errors of the scanning process, there are often 1-2 pixel wide interrupts in the pen strokes, which would break our definition of loops. To eliminate them, a morphological closing is applied to each image before loop extraction.

Shape descriptors are used to describe the different aspects of loops, thereby allowing an easy comparison. There are several promising formulas described in the literature for calculating shape descriptor values. Instead of choosing one of them, we used as many of them as possible. This will allow us to identify the most significant shape factors in later phases. Our hypothesis was that there will be at most 2-3 significant shape descriptors, and all the others will be redundant and can be ignored in the future. However, this was not the case, as you will see in section 5.

The following shape descriptors were used during feature extraction: Perimeter, area, formfactor, maximum diameter, maximum diameter angle, roundness, centroid, bounding box, inscribed diameter, extent, modification ratio, compactness, bounding circle, moment axis angle, convexity, solidity, aspect ratio

The nontrivial descriptors are defined as follows:

$$Modification\ Ratio = \frac{Inscribed\ Diameter}{Maximum\ Diameter} \quad (1)$$

$$\text{Formfactor} = \frac{4\pi \cdot \text{Area}}{\text{Perimeter}^2} \quad (2)$$

$$Extent = \frac{\text{Area}}{\text{Bounding Rectangle Area}} \quad (3)$$

$$\text{Compactness} = \frac{\sqrt{\left(\frac{4}{\pi}\right)\text{Area}}}{\text{Maximum Diameter}} \quad (4)$$

$$\text{Convexity} = \frac{\text{Convex Perimeter}}{\text{Perimeter}} \quad (5)$$

$$\text{Solidity} = \frac{\text{Area}}{\text{Convex Area}} \quad (6)$$

$$\text{Aspect Ratio} = \frac{\text{Maximum Diameter}}{\text{Minimum Diameter}} \quad (7)$$

A detailed introduction of shape descriptors can be found in [23].

5 Classification

5.1 General problems of classification

After the extraction process features are matched with an algorithm, like [26] and the similarity between feature pairs is computed. These similarity values are used to make decisions about the acceptance of a given signature in the classification phase. A single classifier is trained with a set of original signatures. Based on the training,

the classifier can make decisions about the acceptance or rejection of a single test signature.

It should be noted, that in Fig. 1. several simplifications were used to get a uniform view of the different approaches. This "single classifier" can of course represent a composite system consisting of different local and global experts allowing the decision to be made with a deep understanding of the context. In some other cases there are classifiers used, to improve the feature extraction or distance calculation phase. These should not be confused with the classification phase which in that scenario is a single threshold decision.

While it is uncommon in literature, the decision of the classifier is not limited to a binary decision. Beside the values "accept" and "reject" a third value "uncertain" is introduced in some works, usually combined with a confidence value.

Although in many fields, the application of a classifier (eg. the application of a neural network) is a routine operation, feature based off-line signature verification has some specialties, which we had to address. The following subsections introduce some of the problems and describe our solutions for them.

5.1.1 Distance measurements

Input values are not always simple values. The baseline of a signature for example is a set of points, which could easily form a large, hard to interpret input for the classifier. To overcome this, the feature values are not directly used as an input. Instead, each signature is compared with every other signature, delivering a wide range of comparison results which are then stored and used as input for classification.

Each feature can define its own distance function. For example when calculating the distance of loop centroids, this distance function is a Euclidean distance function, but when calculating the distance of two baselines this function is a DTW (Dynamic Tiem Warping) function.

5.1.2 Normalization

Input variables can, and should be interpreted on different scale. Some features (like moment axis angle) are stored in degrees, some of them represent Euclidean distances in pixels, and again others (like solidity) are simple proportions, with no direct meaning. To allow a general processing, all input values are rescaled to fit in the [0…1] interval.

5.1.3 Missing values

A neural network has a well defined number of input variables. When working with global features (like width or height of a signature) this number can be easily defined. On the other side local features tend to be much more instable. Some signatures of a signer may have 3 loops, while all the others contain 4 loops. Without further preprocessing, this could result in input vectors of different lengths for the classifier. To eliminate this problem, the feature extraction phase is extended to consider not only a single signature, but to examine all original signatures of a signer available for training purposes. Thereby, the feature extractor is able to identify the stable features of a signer, thus setting their count to a fixed number. Later, when testing new signatures, features located in the test signature can be matched to this fixed model. If a feature has no match in the model, it is ignored. On the other hand, if there is a feature in the model, which has no matching feature in the test signature, the distance of the features is set to infinite.

This allows us to lock the length of the input vector used by the neural network.

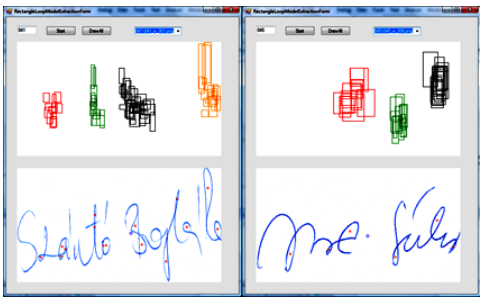

Fig.4. Matching loops: bounding rectangles of matching loops are projected over each other (top)

5.1.4 Incomplete training data

Most of the previous problems addressed were specific problems to the feature based approach. However there is one important problem, which every classifier in the field must address: in the real world, there are no negative samples available for training. Without negative samples, a neural network (or any other classifier) can easily get overtrained, which is of course not desirable.

It is acceptable to ignore this problem, and provide some negative samples to the system [24][25], however this kind of benchmarking allows only a theoretical evaluation of the goodness of given feature extraction and classification algorithms, with respect to the quality of the provided forgeries.

To prepare the system for real world scenarios, another approach has to be taken, which is the artificial generation of negative samples. This can be done in any phase of the signature processing

- *the original signature can be morphed*, however, there is no guarantee, that the morphing will realistically model real forgeries
- *feature values can be altered*: this seems to be a promising way, because alterations could be

performed with respect to the current features
- *normalized feature distances can be altered*: this is the easiest way, because it does not require feature specific implementation. This is the approach we have taken.

5.2 Statistical evaluation of features

5.2.1 Correlation of features and originality

The fact, how much a specific feature can imply originality of a signature can be measured by the correlation of the coordinates, representing the feature value, and a value, representing, whether the signature is original or a forgery. Thus the correlation coefficients can be calculated.

$$r = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2 \sum_{i=1}^{n}(y_i-\bar{y})^2}} \quad (8)$$

Talking about geometric features, especially in the case of shape descriptors it is probable, that there will be a strong correlation between some of them. During our experiments, we performed the independence analysis for each feature pair. When experimenting with new feature types these results helped us to evaluate the effective value of a new method.

Table 1 showed that there is no specific feature type with a strong correlation between the originality of a given signature. This means, that there probably none of the features examined here can be used in itself, to differentiate between original and forged signatures.

Motivated by the above results, and real life experiences, we also performed a cluster analysis for feature values of signatures from a typical signer. Results can be seen in table 5 and 6.

5.3 Decision tree

Considering the final decision as a logical function with multiple values, we can apply the method of decision trees. A decision tree is a tree, with all inner nodes representing input parameters. At a given node the choice is always made based on the value of the associated parameter. The training phase in this case means the building of the decision tree, while the classification would be a simple series of choices made based on the tree. It would be also possible to alter the tree based on some feedback. A possible building algorithm would look like this: the parameter with the best separation power is taken as the root node of the tree, and the training set is divided into two sets, based on the given parameter. If one of these sets includes only positive samples and the other consists only of negative samples, the tree is ready, otherwise, the algorithm is repeated recursively, and sets are divided into new and new sets based on the next parameter until the best possible separation is achieved.

There are two main difficulties, when applying this method. First and most important is the lack of negative samples, and second is that our input parameters (which represent feature distances) have non discrete values.

5.4 Nearest neighbor

Considering the fact, that signatures can differ on a wide scale it may be useful to ignore the original signatures which have the less in common with a tested signature and only use the most similar signatures as references [2]. The need for such a distinction was also confirmed by the cluster analysis of feature distances.

The nearest neighbor algorithm realizes the above idea. The main advantage of its application is that because of the reduced number of samples, testing can be performed by the direct comparison of feature pairs in contrast to other methods, which have to calculate cumulated values, like averages of feature distances.
It may be possible, that results can be improved, when a few more samples (not only the nearest neighbor) are taken into account.

5.5 Quartile

Another simple solution for grouping the signatures into separate sets is based on a simple outlier detection principle. The algorithm is based on the quartile based outlier detection algorithm, which is widely used in mathematical statistics.

In this case, all the feature distances of the original samples are grouped into 4 quartiles based on their distribution. During the training the middle quartiles are used as reference intervals for each feature type. When testing a signature, features are examined in each of the corresponding dimensions, whether they are within the given reference interval. A positive answer means a positive vote and a negative answer means a negative vote for the given feature. These votes can then be summarized (and may be weighted) to make the final decision. The precision of this algorithm depends highly on the fine tuning of the voting algorithm. It can also be seen that the proportion of the FAR and FRR values can be easily controlled, by setting the number of positive votes required for a positive decision. Thereby the strictness of the system can be parameterized.

.

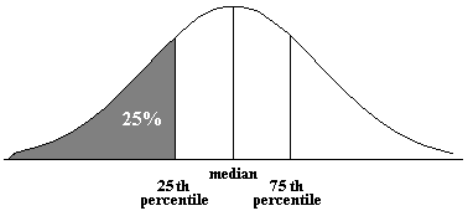

Fig.5. quartiles in a typical distribution

A main advantage of the algorithm is that it only requires positive samples in the learning phase. Drawbacks are the need for manual parameterization and the fact, that it is only able to consider feature pairs and ignores any higher relationships between them.

5.6 Neural network classifier

Basically there are two different applications of neural networks in our system. First, a neural network is used, to evaluate the "optimal" solution, the theoretical limits of the whole approach, by using the whole signature database as an input. Second, a classifier is used, to learn the specifics of each signers signatures. Training is done with generated negative samples. Testing is done with real originals and forgeries.

5.6.1 Optimal solution

Supposed, that there is some kind of nonlinear relationship between the input vectors and the originality of a signature, neural networks can be effectively used to identify the best theoretical results which could be achieved by a classification algorithm.

Of course, a neural network can easily get overtrained, therefore we separate the samples in three disjoint sets, one for training, one for measurement and one for testing[26] purposes. The *training* set is used for training the neural network until it reaches a given precision measured on the *measurement* test. Finally the effective performance of the network is tested on the *test* set. The results of the later are then used to calculate the error rates.

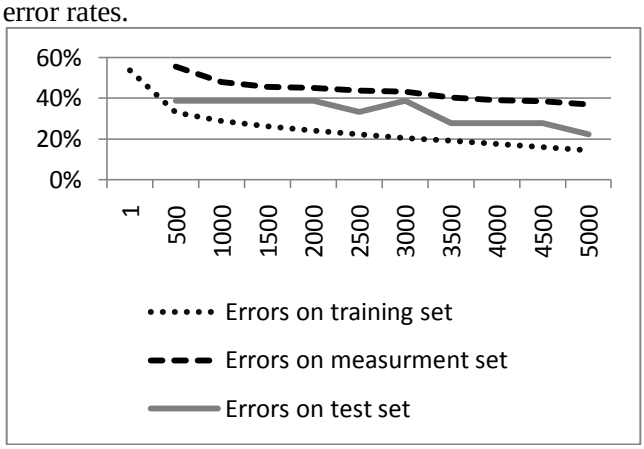

Fig.5. Learning curve of the neural network

5.6.2 Real world scenario

In a real world scenario, a classifier can only be trained with artificial forgeries. Although we have experienced with several alternative configurations, best results were achieved by a network with 13 preprocessing neurons, two hidden layers (with 2 and 8 neurons) and with a transformed sigmoid activation function. Further results are presented in the next section.

6 Experimental Results

In our experiments the database of the Signature Verification Competition 2004 [15] was used. This is an on-line signature database therefore it contains the stroke information, but no images are provided. The stroke information was used to synthesize signatures similar to the original ones. Stroke points were connected with straight lines, fading out on the line borders. Bicubic interpolation and anti-aliasing were used to make the final image smoother. An example of reproduced signature can be seen on Fig. 2. 1600 signatures from 40 signers (20 originals and 20 forgeries from each) ensure a sample large enough for testing our feature extraction and classification algorithms.

The experimental setup uses 10 original signatures from each signer and a set of generated forgeries for training. Afterwards the network is tested with 10 other original and 10 forged signatures. The resulting average error rates are summarized in table 4. It can be seen, that the values vary largely between different signers. It is however really promising, that this error rate is under 10% at two of the signers.

Another application of our setup is the evaluation of the differentiating power of the different shape descriptors. Experiments have shown that in contrary to our hypothesis, none of the shape descriptors is significant for *all* the signers. However, 12 out of 16 signers, who have loops in their signatures, have significant differences at some of their descriptors.

7 Conclusion

In this paper we discussed problems occurring during feature based off-line signature verification and delivered solutions for the special questions of this problem class. Although our achieved error rates are not yet ready for real world scenarios, we have demonstrated that local features can successfully be used with neural network classification systems, to distinguish original signatures from forgeries.

We have also demonstrated that by changing the testing configuration our system can be used to estimate its own theoretical boundaries, and to evaluate the discriminative power of different features.

Feature	**Correlation to original**
DTW	-0,340
Loop Area1 Difference 1	-0,096
Loop Area2 Difference 1	-0,097
Loop Area3 Difference 1	-0,097
Loop Perimeter Difference 1	-0,111
Loop FormFactor1 Difference 1	-0,148
Loop FormFactor2 Difference 1	-0,159
Loop FormFactor3 Difference 1	-0,158
Loop Maximum Diameter Difference 1	-0,096
Loop Roundness1 Difference 1	-0,174
Loop Roundness2 Difference 1	-0,172
Loop Roundness3 Difference 1	-0,170
Loop Compactness1 Difference 1	-0,158
Loop Compactness2 Difference 1	-0,173

Table 1. Correlation of feature values and originality

Feature	Average error rate
Baseline angle difference	0,412108726
Baseline length difference	0,425197368
DTW	0,220716759
Gap angle difference	0,483051619
Gap length difference	0,458522267
Loop Area	0,45683366
Loop Bounding Circle Radius	0,432501408
Loop Compactness	0,442280517
Loop count difference	0,44958795
Loop Extent	0,440355355
Loop FormFactor	0,449571648
Loop Inscribed Diameter	0,444257183
Loop Maximum Diameter	0,432918075
Loop Maximum Diameter Angle	0,382043038

Table 2. Discriminative power of features

SVC signature identifier	**EER**
002.csv	50%
006.csv	43%
013.csv	27%
015.csv	37%
018.csv	20%
020.csv	37%
032.csv	33%
033.csv	37%
035.csv	3%
038.csv	47%
040.csv	7%
Average	**31%**

Table 3. Achieved equal error rates with a neural network, using artificial negative samples

SVC signature identifier	Area Difference	Perimeter Difference	Maximum Diameter Difference	Compactness Difference	Inscribed Diameter	Modification Ratio	Extent	Bounding Circle Radius	Maximum Diameter Angle	Moment Axis Angle
002	0 ,49	0,48	0,51	0,43	0,48	0,42	**0,18**	0,51	**0,13**	0
004										
006	0,5	0,48	0,49	0,5	0,49	0,5	0,46	0,49	0,4	0,42
008	0,43	0,39	0,38	0,36	0,43	0,37	**0,2**	0,38	0,48	**0,13**
010	0,49	0,5	0,5	0,45	0,48	0,42	0,45	0,5	0,41	0,42
013	0,5	**0,34**	**0,35**	0,41	0,51	0,48	0,5	**0,35**	0,5	0,5
015	0,48	0,48	0,48	0,48	0,47	0,5	0,38	0,48	0,5	0,43
018	0,41	0,48	0,5	0,37	0,41	0,4	0,42	0,5	0,46	0,46
020	**0,1**	**0,19**	**0,26**	**0,25**	**0,09**	**0,2**	0,37	**0,25**	0,49	**0,25**
022										
024										
025	0,44	0,42	0,45	0,49	0,43	0,49	0,42	0,45	0,44	0,44
028	**0,32**	**0,32**	0,41	0,46	**0,27**	0,48	0,42	0,4	0,4	0,5
032	0,38	**0,31**	**0,27**	0,35	0,48	0,43	0,41	**0,26**	0,49	0,49
033	0,5	0,49	0,48	0,44	0,48	0,4	0,48	0,48	0,45	0,45
034										
035	**0,29**	**0,31**	**0,33**	0,46	**0,35**	0,44	0,45	**0,33**	**0,23**	**0,23**
037	0,46	0,44	0,38	0,46	0,48	0,47	0,46	0,38	0,5	0,5
038	0,36	0,38	**0,32**	**0,34**	**0,33**	**0,31**	0,41	**0,33**	0,37	0,4
040	0,5	0,5	0,5	0,44	0,5	0,45	0,41	0,5	**0,25**	**0,27**

Table 4. Achiveable equal error rates by considering single shape descriptors in signatures. Error rates below 35% are highlighted

	Cluster		
	1	2	3
VAR00001	,42	,41	,33
VAR00002	,39	,15	,40
VAR00003	,40	,15	,40
VAR00004	,40	,15	,40
VAR00005	,47	,19	,45
VAR00006	,53	,36	,21
VAR00007	,63	,39	,24
VAR00008	,64	,39	,25
VAR00009	,58	,25	,39
VAR00010	,65	,38	,23
VAR00011	,71	,42	,23
VAR00012	,72	,41	,23
VAR00013	,65	,38	,22
VAR00014	,71	,41	,22
VAR00015	,71	,40	,22
VAR00016	,36	,18	,33
VAR00017	,66	,36	,22
VAR00018	,45	,35	,31
VAR00019	,39	,27	,25
VAR00020	,38	,27	,25
VAR00021	,58	,24	,41
VAR00022	,47	,55	,49
VAR00023	,27	,49	,29
VAR00024	,25	,23	,25
VAR00025	,31	,48	,30
VAR00026	,39	,37	,40
VAR00027	,20	,16	,30
VAR00028	,21	,15	,31
VAR00029	,36	,20	,36
VAR00030	,21	,45	,32
VAR00031	,22	,45	,32
VAR00032	,21	,51	,31
VAR00033	,33	,36	,33
VAR00034	,39	,60	,32
VAR00035	,38	,60	,32
VAR00036	,26	,45	,27
VAR00037	,54	,23	,34
VAR00038	,61	,24	,32
VAR00039	,62	,24	,32
VAR00040	,54	,23	,34
VAR00041	,62	,24	,32
VAR00042	,63	,24	,32
VAR00043	,20	,38	,32
VAR00044	,64	,22	,33
VAR00045	,44	,37	,37
VAR00046	,43	,45	,35
VAR00047	,43	,46	,35
VAR00048	,24	,45	,27
VAR00049	,49	,39	,52
VAR00050	,32	,49	,33
VAR00051	,28	,19	,34
VAR00052	,34	,46	,33
VAR00053	,27	,42	,36
VAR00054	,21	,53	,28
VAR00055	,20	,51	,28
VAR00056	,26	,48	,26

Table 5. Final cluster centers

Cluster	1	25,000
	2	40,000
	3	71,000
Valid		136,000

Table 6. Number of cases in each cluster

References

[1] V. E. Ramesh and M. N. Murty, "Off-line signature verification using genetically optimized weighted features," Pattern Recognition, no. 32, pp. 217-233, 1999.

[2] J. Coetzer, B. M. Herbst, and J. A. d. Preez, "Off-line Signature Verification Using the Discrete Radon Transform and Hidden Markov Model," EURASIP Journal on Applied Signal Processing, vol. 4, pp. 559-571, 2004.

[3] M. A. Ismail and S. Gad, "Off-line Arabic Signature Recognition and Verification," Pattern Recognition, vol. 33, pp. 1727-1740, 2000.

[4] H. Baltzakisa and N. Papamarkos, "A new signature verification technique based on a two-stage neural network classifier," Engineering Applications of Artificial Intelligence, vol. 14, pp. 95-103, 2001.

[5] E. Ozgunduz, T. Senturk, and M. Karsligil, "Off-line Signature Verification and Recognition by Support Vector Machine," Thirteenth European Signal Processing Conference, 2005.

[6] M. K. Kalera, S. Srihari, and A. Xu, "Offline Signature Verification and Identification Using Distance Statistics," International Journal of Pattern Recognition and Artifcial Intelligence, vol. 18, no. 7, pp. 1339-1360, 2004.

[7] R. Plamondon and G. Lorette, "Automatic Signature Verification and Writer Identification - The State of the Art," Pattern Recognition, vol. 22, no. 2, pp. 107-131, 1989, Ősi áttekintés az aláírásfelismerés akkori állásáról.

[8] F. Leerle and R. Palmond, "Automatic Signature Verification - The State of the Art 1989-1993," Int'l Pattern Recognition and Artificial Intelligence, special issue signature verification, vol. 8, no. 3, pp. 643-660, 1994, Összefoglaló a korabeli eredményekről.

[9] R. Plamondon and S. N. Sargur, "On-Line and Off-

Line Handwriting REcognition: A Comprehensive Survey," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 1, pp. 63-80, 2000, Átfogó tanulmány, jó hivatkozásokkal, 2-5%-ra saccolt hibaarány, jó kritika az összehasonlításról.

[10]B. Kovari, "The development of off-line signature verification methods, comparative study," in microCAD 2007 International Scientific Conference, 2007.

[11]"Pattern Recognition, special issue on automatic signature verification," vol. 8, no. 3, Jun. 1994, különkiadás csomó cikkel a témáról.

[12]K. A. Jain. Michigan State University - Biometrics. [Online]. http://www.cse.msu.edu/~cse891/Sect601/SignatureRcg.pdf

[13]J. Fierrez and J. Ortega-Garcia, "On-Line Signature Verification," in Handbook of Biometrics. Springer US, 2007, pp. 189-209, Legfrissebb áttekintő anyag.

[14]A. Lumini and L. Nanni, "Over-complete feature generation and feature selection for biometry," Expert Systems with Applications, 2007.

[15]J. Fierrez-Aguilar, S. Krawczyk, J. Ortega-Garcia, and A. K. Jain, "Fusion of Local and Regional Approaches for On-Line Signature Verification," IWBRS 2005, LNCS 3781, p. 188–196, 2005.

[16]K. Huang and H. Yan, "Off-line signature verification using structural feature correspondence," Pattern Recognition, no. 35, p. 2467–2477, 2002.

[17]S.-H. Kim, K.-S. Oh, and H.-I. Choi, "Off-line Verification System of the Handwrite, Signature or Text using a Dynamic Programming," ICCSA 2007, LNCS 4705, no. 1, pp. 1014-1023, 2007.

[18]G. F. Russell and A. B. Jianying Hu, "Dynamic Signature Verification Using Discriminative Training," in Proceedings of the 2005 Eight International Conference on Document Analysis and Recognition (ICDAR'05), 2005.

[19]W. K. Pratt, Digital Image Processing: PIKS Scientific Inside. Wiley-Interscience, 2007.

[20]S. Akle, M. .-E. Algorri, and A. Salcedo, "Use of wavelet-based basis functions to extract rotation invariant features for automatic image recognition ," WSEAS Transactions on Information Science and Applications, vol. 5, no. 5, pp. 664-673, 2008.

[21]X. D. Zhuang and N. E. Mastorakis, "The curling vector field transform of gray-scale images: A magneto-static inspired approach," WSEAS Transactions on Computers, vol. 7, no. 3, pp. 147-153, 2008.

[22]A. A. Kholmatov, "Biometric Identity Verification Using On-Line & Off-Line Signature Verification," 2003.

[23]B. Kovari, "Time-Efficient Stroke Extraction Method for Handwritten Signatures," in ACS07, The 7th WSEAS International Conference on Applied Computer Science, 2007, pp. 157-161.

[24]R. A. Huber and A. M. Headrick, "Handwriting Identification: Facts and Fundamentals," CRC Press, LCC, 1999.

[25]B. Kovari, G. Kiss, and H. Charaf, "Stroke Extraction and Stroke Sequence Estimation for Off-line Signature Verification," The Eighth IASTED International Conference on Visualization, Imaging, and Image Processing.

[26]J. C. Rush, The Image Processing Handbook, Fifth edition. North Carolina State University, 2006.

[27]R. Nerino, "Automatic registration of point-based surfaces ," WSEAS Transactions on Computers, vol. 5, no. 12, pp. 2984-2991, 2006.

[28]J. Mahmud and C. M. Rahman, "On the power of feature analyser for signature verification," Proceedings of the Digital Image Computing, Techniques and Applications, 2005.

[29]S. N. Srihari, A. Xu, and M. K. Kalera, "Learning Strategies and Classification Methods for Off-line Signature Verification," Proceedings of the 9th Int'l Workshop on Frontiers in Handwriting Recognition (IWFHR-9 2004), 2004.

[30]G. Horváth, et al., Neural Networks. Budapest, 2006.