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.

.

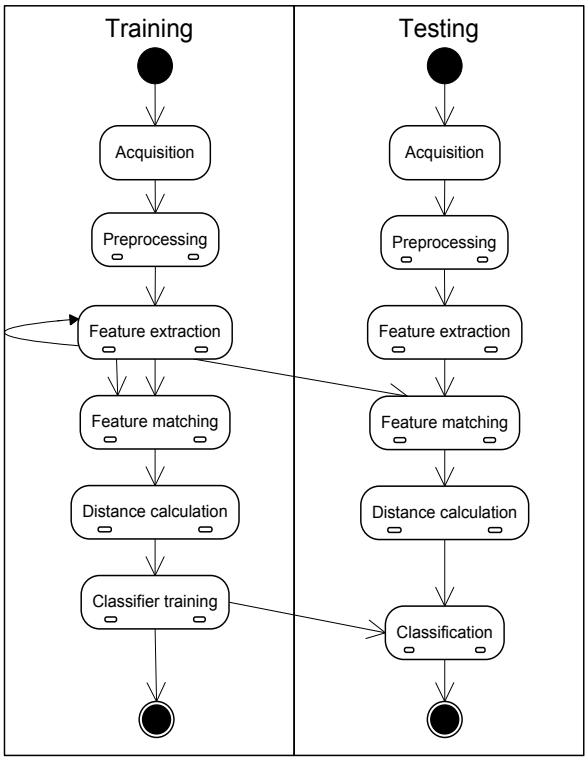

Fig. 1 Generalized view of an off-line signature verification system

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.

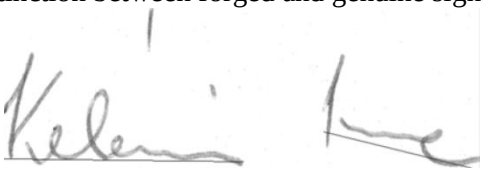

Fig.3. Three skew lines obtained by our algorithm

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,