Off-line Signature Verification Incorporating the Prior Model

LIANG WAN[(1)], ZHOU-CHEN LIN[(2)], RONG-CHUN ZHAO[(1)]

[(1)] Dept. Computer Science & Engineering, Northwestern Polytechnical University, Xi'an, Shaanxi 710072, China

[(2)] Multimodal User Interface Group, Microsoft Research, Asia, Beijing 100080, China

E-mails: liangwan@mail.nwpu.edu.cn, zhoulin@microsoft.com, rczhao@nwpu.edu.cn

Abstract

The existing signature verification systems usually train classifiers for a new user by both his/her genuine and forgery signatures. Obviously, the requirement of forgery signatures is impractical. This paper presents an off-line signature verification system that only requires the genuine signatures of a new user. At the training stage the system learns the mapping between the parameters of classifiers without simple forgeries and those with simple forgeries. In the application stage, a primary classifier is trained for a new user without his/her simple forgeries. The final classifier is obtained by transforming the primary classifier via the mapping learnt in the training stage. Experimental results confirm the effectiveness of the proposed system.

Keywords

Signature verification; Simple forgery; Boosting algorithm; The prior model

1. Introduction

Financial and legal transactions demand several types of handwritten documents to be validated. The process involves two independent but highly related tasks: a) writer identification, and b) signature verification, i.e. decision whether the suspected signature is true or not. Handwritten signature verification (HSV) can be classified into two categories: on-line HSV and off-line HSV. The former relies on dynamic attributes, such as pressure, velocity and acceleration. The latter analyzes the digitized signature images, in which dynamic features are lost.

Usually three kinds of forgery can happen in signature verification. Random forgery is taking the genuine signature of others for that of the current user. Skilled forgery is produced with close imitations. It is hard to be differentiated from the genuine one only by shape variations. Simple forgery is produced with the knowledge of content but without close imitations. For example, the forger signs out of his/her memory on the genuine signature.

There have been many systems for signature verification. Sabourin and Drouhard[8] proposed an approach based on directional probability density function in combination with BP neural networks to detect random forgery. Qi and Hung[7] used global and grid features with a simple Euclidean distance classifier. Bajaj and Chaudhury[4] proposed a system consisting of sub-classifiers based on three sets of global features. Sansone and Vento[2] proposed a sequential three-stage multi-expert system, in which the first expert eliminates random and simple forgeries, the second isolates skilled forgeries, and the third gives the final decision by combining decisions of the previous stages together with reliability estimations. However, its performance relies greatly on the rejection criteria chosen for each expert. Baltzakis and Papamarkos[1] developed a two-stage neural network, in which the first stage gets the decisions from neural networks and Euclidean distance classifier supplied by the global, grid and texture features, and the second stage combines four decisions using a radial-base function (RBF) neural network.

Most existing systems use both positive (genuine) samples and negative (forgeries) samples in the training process. Actually, it is hard to collect enough forgeries (of simple and skilled types) for each new user, especially when the number of users is large.

In this paper, a novel system for signature verification is proposed. It can detect both random and simple forgeries. In Section 2, the system model is introduced in detail. Section 3 gives the experimental results. Section 4 concludes this paper.

2. The Proposed System

The proposed Handwritten Signature Verification System (HSVS) involves two parts, as illustrated in Figure 1.

2.1. The System Model

We build one individual classifier for each signature owner. His/her dataset consists of genuine signatures as positive samples, random (and simple) forgeries as negative samples. Note that genuine signatures of an owner are random forgeries of other owners. Supposing the selected features are effective for most users, the mapping between the classifiers trained with and without simple forgeries will be similar among users.

Referring to Figure1, our system consists of two stages: the training stage and the application stage, where:

1. O^1 and O^2 are two non-overlapped user sets.
2. θ_{rk} and τ_{rk} denote the parameters of the *k*-th sub-classifier used in the training stage and the application stage.
3. α_k and β_k denote the weights of the *k*-th sub-classifier used in the training stage and the application stage.
4. $D(i)$ and $D'(i)$ denote training datasets for *i*-th user without and with simple forgery.

The system training stage is implemented on O^1:

Step 1: For user i in O^1, his/her first training dataset $D_1(i)$ is composed of genuine signatures and random forgeries $D_1(i) = D_{g1}^1(i) \bigcup D_{rf1}^1(i)$. We can obtain a classifier $H_1(i) = (\theta_{i,rk}, \alpha_{i,k}, L_{i,k})$, in which $\{L_{i,k}\}$ are label sets of genuine signatures which are randomly sampled for training.

Step 2: For user i, his/her second training dataset has simple forgeries, i.e. $D_1'(i) = D_{g1}^1(i) \bigcup D_{rf1}^1(i) \bigcup D_{sf1}^1(i)$. To make parameters of two classifiers comparable, we keep using $\{L_{i,k}\}$ as the positive training samples, and train another classifier $H_1' = (\theta_{i,rk}', \alpha_{i,k}')$.

Step 3: Repeat Step 1 and Step 2 for several owners. Establish the mapping between classifier parameters under two training modes,

$$f : (\theta_{rk}, \alpha_k) \to (\theta_{rk}', \alpha_k').$$

The system application stage is tested on O^2:

Step 4: For the *j*-th new user in O^2, we select genuine samples from O^1 as random forgeries, then his training dataset consists of genuine samples and random forgeries: $D_2(j) = D_{g2}^2(j) \bigcup D_{rf1}^2(j)$, where $D_{rf1}^2(j) = D_{g1}^1$. Train the primary classifier $H_2(i) = (\tau_{j,rk}, \beta_{j,k}, L_{j,k})$.

Step 5: Based on the previous assumption, we apply the mapping f to get another set of parameters that should correspond to the classifier trained with simple forgeries: $(\tau_{j,rk}', \beta_{j,k}') = f(\tau_{j,rk}, \beta_{j,k})$.

Step 6: Build the classifier H_2' with $(\tau_{j,rk}', \beta_{j,k}', L_{j,k})$.

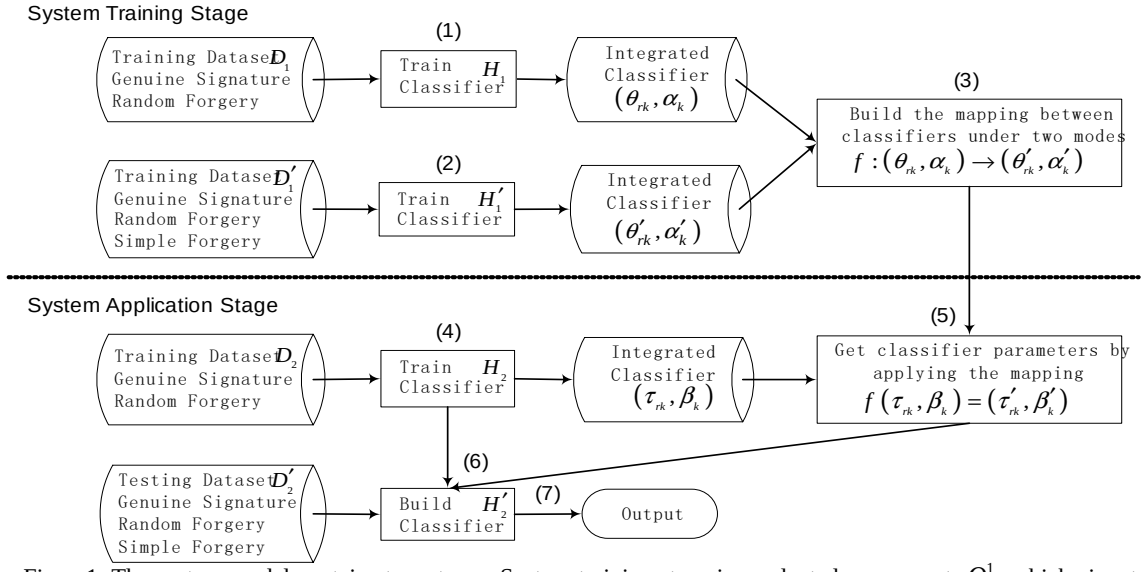

Figure1. The system model contains two stages. System training stage is conducted on user set O^1, which aims to get the mapping. System application stage is performed for each new owner, which is tested on user set O^2.

Step 7: Make verification using H_2'.

This system architecture can train a classifier for a new owner without simple forgeries, which makes the system practical. In the following we introduce more details.

2.2. Feature Extraction

Signature images are scanned with a resolution of 300 dpi. To eliminate homogenous background[13] produced in the scanning process, local contrast enhancement is first employed and local binarization proceeds. Since the binarized signature shrinks along edges, a dilation morphological operator is used to make the stroke wider, and a bridge operator to connect two areas that are one pixel apart. The resulting binarized image is taken as a mask image to extract the gray-level signature trace from the scanned image.

In this system, four groups of features are extracted on global or local scales. As a shape descriptor, global feature provide information about the whole structure of the signature. We select a subset of global features proposed by Baltzakis[1], which include pure width, image area, centers of the signature, maximum vertical and horizontal projection, vertical and horizontal projection peaks, local slant angle, number of edge points, and number of cross points.

The grid gray feature is defined as a group of average gray value in each grid overlapped on the preprocessed image, which describe local shape characteristic. Here grid coordinates are determined adaptively by dividing pixel projection histograms into successive parts with almost equal pixel counts.

As a pseudo-dynamic descriptor, ink distribution feature[7] give out gradual changes of gray values on the local scale.

The co-occurrence matrix is a widely used approach to texture analysis. Each element in the matrix represents the probability of the combination of gray values at pairs of points separated by a vector $\delta = (d, \theta)$, whereby d is the distance and θ the angle. Here the corresponding matrix is denoted as $M(d, \theta)$. In our approach, four matrices $M(1, 0°), M(1, 45°), M(1, 90°), M(1, 135°)$ are calculated separately. And four second-order statistic measurements[9] (matrix energy, difference matrix energy, difference matrix mean, and relevance variance) are evaluated for each matrix. Finally all those features are grouped into a 16-dimension feature vector.

2.3. Integrated Multiple Classifiers

Referring to Fig1, H can be any classifier in theory. The parameters of two classifiers must be comparable. More importantly, it should be easy to build a classifier out of parameters. Here we choose MED (Minimum Euclidean Distance) classifiers as sub-classifiers and make classifier fusion based on Boosting algorithm.

It is crucial to choose a suitable distance threshold for an MED classifier. We define the relevance threshold T^k as

$$T^k = \frac{D^k_{gmax} + D^k_{gmin}}{2 * \sigma}, k = 1, ..., 4 \qquad (1)$$

where σ is the standard deviation of genuine training samples, D_{gmax} denotes the maximum distance between genuine testing samples and their mean, and D_{fmin} is the minimum distance between forgeries and the mean of genuine samples. Relevance threshold T^k provides a feasible basis for the mapping.

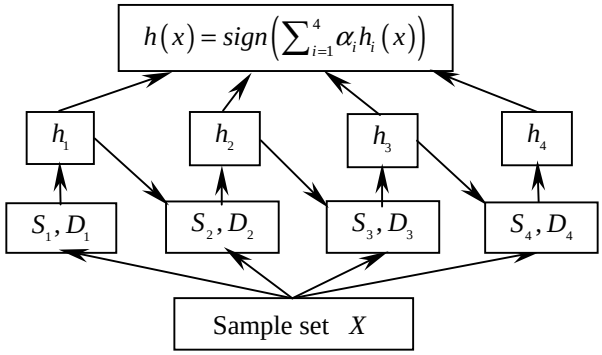

Figure 2. Integrate multiple classifiers by Adaboost algorithm

Boosting algorithm can change many weak learners into a strong learner. An MED sub-classifier is established respectively for each group of features. Adaboost[10] approach is employed to combine sub-classifiers, as shown in Fig 2, where h_i refers to an individual classifier, S_k denotes the feature set, and D_k is the sample distribution. Tests show the order of sub-classifiers has little effects on the performance of the integrated classifier.

3. Experimental Results

3.1. Data collection

Signatures in Chinese for two non-overlapped user sets are collected respectively. O^1 consists of 8 owners, each one providing 12 genuine signatures, and 1 simple forgery each for the rest 7 owners. O^2 consists of 20

owners, each providing 24 genuine signatures, and 1 simple forgery each for the rest 19 owners. Signature may be affected by the mood of the owner, paper quality, and stencil type. Here subjects work with gel pens of the same type, writing on the same paper printed with 3*4 tables, each cell of which contains a signature.

3.2. System Training Stage

This stage is conducted on the user set O^1. For *i*-th user, his/her first training dataset $D_1(i)$ is composed of 7 genuine signatures and 3*8 random forgeries randomly selected from sample sets of the 8 owners in O^2. Then train the first classifier H_1. The second training dataset $D_1'(i)$ includes 7 more simple forgeries. We can train the second classifier H_1'.

In each iteration step, Adaboost algorithm will randomly select samples from the dataset according to the current sample distribution. So each training process may get different classifier. By repeating ten times of the training process for every owner in O^1, we get 80 groups of parameter pairs. Parameter data are feed into a BP neural network N_F with $8\times4\times8$ structure.

3.3. System Application Stage

This stage is proposed to imitate the practical application. For *j*-th user in O^2, the training dataset $D_2(j)$ consists of 12 genuine signatures, and 3*8 random forgeries randomly selected from the sample sets of the 8 owners in O^1. Train the primary classifier as H_2. Take parameters of H_2 as the input of the neural network N_F. Combine its output with the structure of H_2 to build the final classifier H_2'. Considering the process in system training stage, we can see that H_2' has the same role as the classifier which is trained with simple forgeries included.

To evaluate the efficiency of H_2', we use the remaining 12 genuine signatures, 9 simple forgeries, and 12*19 random forgeries, randomly selected from the sample sets of the rest 19 owners in O^2, as the testing set for user j.

Type I (false rejection rate, or FRR) and Type II (false acceptance rate, or FAR) errors are calculated for our testing purpose. We record error rates for sub-classifiers, classifier H_2 and the final classifier H_2'.

The use of integrated classifier can improve the performance on random forgeries, as is evident from the fourth column in Table 1. Comparing the resulting error rates for H_2 and H_2', it's obvious that the latter has better performance on two types of forgery. However, the FRR in this case is higher than that of H_2. This is due to the fact that in the system training stage, the second training process takes simple forgeries into account. Since the distances between simple forgeries and genuine signatures are usually less than those between random forgeries and genuine signatures, the introduction of simple forgeries may decrease the relevance threshold. Therefore, FRR will increase a bit accordingly. The calculated results also demonstrate the mapping between the parameters with or without simple forgeries is effective in our application environment.

Table 1. System verification results

Error rate	FRR (%)	FAR(%) Simple Forgery	FAR(%) Random Forgery	Total Error (%)
Texture features	13.3	15.0	1.91	2.93
Grid gray features	5.42	50.56	7.83	9.26
Ink distribution features	8.75	37.22	6.89	8.07
Global features	10.83	52.78	16.47	17.51
H_2	4.58	34.44	1.16	2.53
H_2'	7.92	13.89	0.35	1.2

3.4. Comparison test

The results obtained with the proposed system are also compared with those achieved by Baltzakis and Papamarkos' method. See Table 2 for details.

Table 2. Results of Baltzakis and Papamarkos' method based on user set O^2

Error rate	FRR (%)	FAR(%) Simple forgery	FAR(%) Random forgery	Total error (%)
Texture feature NN	25.00	30.56	0.86	3.09
Grid skeleton NN	25.42	22.78	1.34	3.27
Global feature NN	42.08	27.22	2.26	5.08
RBF	77.92	7.78	0.81	4.78
BP	14.58	33.33	0.24	2.13

Table2 records error rates for individual sub-classifiers based on each group of features, those of the system with RBF used as the second stage classifier (proposed by Baltzakis), and those of the system with BP used in the second stage (tested for comparison). It is clear that using different classifiers in the second stage results in quite different performance. And the FRR when using RBF are

significantly low. The possible reason is that the second classifier is trained only with the decisions from classifiers in the first stage. In other words no knowledge from sample distribution is considered.

4. Conclusions

This paper has presented a Handwritten Signature Verification system incorporating a prior model, which trains the classifier for any new owner without simple forgeries. Experimental results of the system have been provided with reference to two non-overlapped user sets. The performance has testified to the effectiveness of the proposed approach. The final results obtained by using the integrated classifier and the mapping are better than those obtained by the classifier working separately. And they are also much better than those by Baltzakis' method.

The future work lies in three aspects. Considering the size of user sets used in the current experiment, the system should be tested on a larger dataset. It should also include skilled forgery into account. Taking the random characteristics of the Adaboost algorithm into account, an integrated classifier of better generalization performance may be selected based on verification results.

References

[1] Baltzkis H, Papamarkos N, A new signature verification technique based on a two stage neural network classifier, Engineering Application of Artificial Interlligence,14: 95-103,2001

[2] Sansone C, Vento M, Signature verification: increasing performance by a multistage system, Pattern Analysis & Application, 3:169-181,2000

[3] Kashi R, Nelson W, Signature verification: benefits of multiple tries, Proceedings of the 8th international workshop on frontiers in handwriting recognition, 424-427, 2002

[4] Bajaj R, Chaudhury S, Signature verification using multiple neural classifiers, Pattern Recognition, 30(1):1-7, 1997

[5] Sabourin R, Off-line signature verification: Recent advances and perspectives, BSDIA'97, Curtiba, 84-98, Nov.1997

[6] Huang K, Yan H, Off-line signature verification based on geometric feature extraction and neural network classification, Pattern Recognition, 30(1): 9-17,1997

[7] Yingyong Q, Hunt B R, Signature verification using global and grid features, Pattern Recognition, 22(12): 1621-1629, 1994

[8] Sabourin R, Drouhard J P, Offline signature verification using directional PDF and neural networks, Proceedings 11th international conference on pattern recognition, 2:321-325, 1992

[9] Franke K, Bunnemyer O, Sy T, Ink texture analysis for writer identification, Proceedings of 8th international workshop on frontiers in handwriting recognition, 268-273, 2002

[10] Freund Y, Schapire R E, A short introduction to boosting, Journal of Japanese Society for Artificial Intelligence, 14(5):771-780, Sep. 1999

[11] Srihari S N, Cha S H, Arora H, Lee S, Individuality of handwriting, Journal of Forensic sciences, 47(4):1-7, July 2002

[12] Said H E S, Tan T N, Baker K D, Writer identification based on handwriting, IEE 3rd european workshop on handwriting analysis and recognition, 4:1-6, July 1998

[13] Franke K, Koppen M, A computer-based system to support forensic studies on handwriting documents, International Journal on Document Analysis and Recognition, 3:218-231, 2001