A Sequential Monte Carlo Algorithm for Adaptation to Intersession Variability in On-line Signature Verification

Yudai Kato, Daigo Muramatsu, Takashi Matsumoto

A Sequential Monte Carlo Algorithm for Adaptation to Intersession Variability in On-line Signature Verification

Yudai Kato *Daigo Muramatsu* *Takashi Matsumoto*
Department of Electrical Engineering and Bioscience,
Waseda University, 3-4-1 Okubo, Shinjuku, Tokyo 169-8555, Japan
{kato04,daigo}@matsumoto.elec.waseda.ac.jp takashi@mse.waseda.ac.jp

Abstract

Personal authentication is becoming increasingly important, and on-line signature verification is one of the most promising approaches to the authentication problem. A factor known as intersession variability in signatures causes deterioration of authentication performance. This paper proposes a new algorithm for overcoming this problem. We propose an algorithm that integrates a model parameter updating scheme in order to suppress deterioration in the authentication system. A model is provided for each user to calculate the score using fused multiple distance measures with respect to previous work. The algorithm consists of an updating phase in addition to a training phase and a testing phase. In the training phase, a model is generated via Markov Chain Monte Carlo for each individual. In the testing phase, the generated model determines whether a test signature is genuine. Finally, in the updating phase, the parameters are updated with test data by using a Sequential Monte Carlo algorithm. Several experiments were performed on a public database. The proposed algorithm achieved an EER of 6.39%, using random forgery for training and skilled forgery for testing.

Keywords: Online Signature Verification, Biometrics, Intersession Variability, Sequential Monte Carlo, Markov Chain Monte Carlo

1. Introduction

Personal identity verification has a variety of applications including electronic commerce, access control for buildings and computer terminals, and credit card verification. The algorithms used to verify personal identity can be classified into four groups depending on whether they are static or dynamic and whether they are biometric, or physical/knowledge-based (Figure 1). For example, fingerprints, retina, iris, blood vessels, DNA, face and ear shape are static and biometric. Algorithms classified as dynamic and biometric include lip movements, voiceprint, and on-line signatures. Schemes that use passwords are static and knowledge-based, whereas methods using IC cards, magnetic cards, and keys are static and physical. With rapidly increasing use of tablet PCs and PDAs, on-line signature verification is a promising method. Discriminating between a genuine signature (written by the originator) and a forgery (written by a forger) has presented difficulties because an on-line signature is a behavioral biometric that lacks stability. Signers cannot produce precisely the same signature each time. Fluctuations in a signature over a series of signing sessions, called intersession variability, causes a degradation of the authentication performance [1][2][3]. Our proposed algorithm integrates an updating phase to prevent this deterioration.

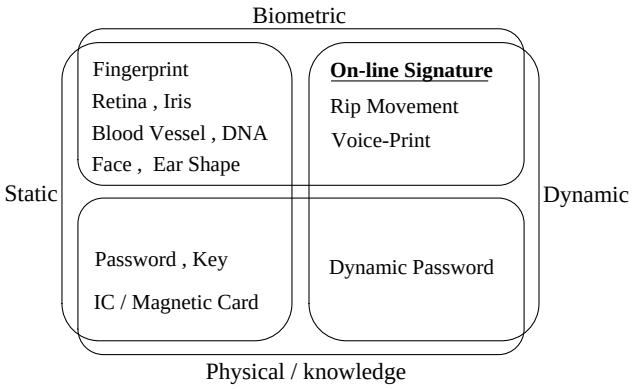


Figure 1. Authentication method

2. Algorithm

2.1. Overall Algorithm

Our proposed algorithm consists of a ”training phase,” a ”testing phase,” and an ”updating phase.” Figure 2 depicts the overall algorithm. In the training phase, template data is selected from the training signatures after preprocessing and feature extraction. Four different distance measures between the template data and the training data are computed. Using these distance measures, we estimate the posterior distribution of the parameters of a perceptron in a Bayesian framework using Markov Chain Monte Carlo (MCMC). Though it is often impossible to express the posterior distribution analytically, the Monte Carlo scheme helps to solve this problem. In the testing phase, parameter samples obtained via MCMC in the training phase are used to compute the probability that a test signature is genuine. This probability value is used for decision making. The test data is also used for updating

the parameter during the updating phase, though test data has no supervised label. If the threshold for parameter updating is satisfied, the model is updated by modifying the parameter via Sequential Monte Carlo (SMC).

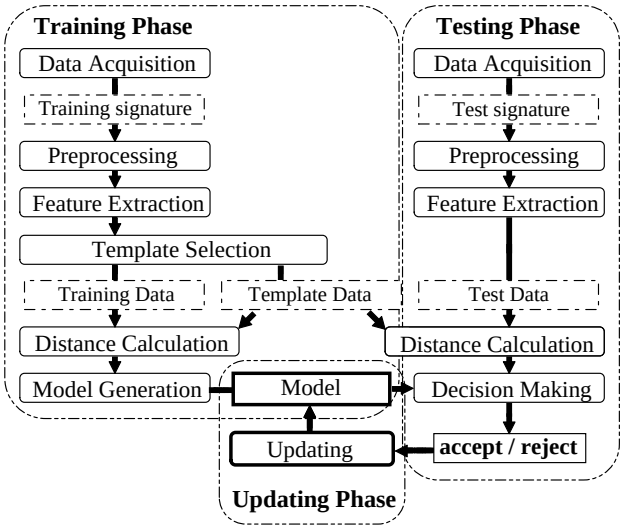


Figure 2. Authentication flow

2.2. Data Acquisition

An electric tablet and a pen are used for data acquisition. The raw data consists of a five-dimensional time series data set:

$$(x(j), y(j), p(j), \phi(j), \psi(j)) \quad j = 1, ..., J$$

where J is the size of the data. $x(j), y(j)$ is the coordinates of the pen position at time j , $p(j)$ represents the pen pressure, and $\phi(j)$,$\psi(j)$ is the azimuth and altitude angle of the pen (Figure 3).

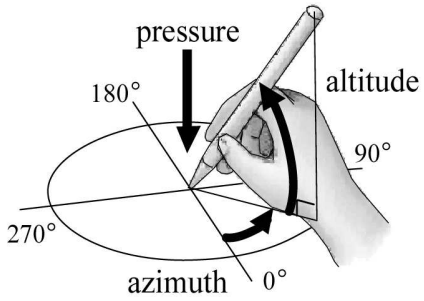


Figure 3. Data from tablet

2.3. Feature Extraction

During preprocessing, the signature data is normalized as follows.

$$dx(j) = \frac{x(j) - X_g}{x_{max} - x_{min}} \cdot N \ , \ dy(j) = \frac{y(j) - Y_g}{y_{max} - y_{min}} \cdot N \quad (1)$$

$$X_g = \frac{\sum_{j=1}^{J} x(j)}{J} \ , \ Y_g = \frac{\sum_{j=1}^{J} y(j)}{J} \quad (2)$$

Here, N is a scaling parameter of the normalization that equalizes the size of the signatures. After the x and y coordinates are normalized, the coordinates are transformed to polar coordinates $\theta(j)$ (vector angle) and $f(j)$ (vector length), where the origin is the gravity point (X_g, Y_g) of the signature.

$$f(j) = \sqrt{dx(j)^2 + dy(j)^2} \quad (3)$$

$$\theta(j) = \begin{cases} \tan^{-1} \frac{dy(j)}{dx(j)} & (dx(j) > 0) \\ sign(dy(j)) \cdot \frac{\pi}{2} & (dx(j) = 0) \\ \tan^{-1} \frac{dy(j)}{dx(j)} + \pi & (dx(j) < 0, dy(j) \geq 0) \\ \tan^{-1} \frac{dy(j)}{dx(j)} - \pi & (dx(j) < 0, dy(j) < 0) \end{cases} \quad (4)$$

The transformed feature vector is defined by

$$(\theta(j), f(j), p(j), \phi(j), \psi(j)) \quad j = 1, ..., J.$$

With this change of coordinates, the original signature trajectory (Figure 4a) is converted into a relative trajectory (Figure 4b).

(a) Original signature (b) Relative trajectory

Figure 4. Trajectories

2.4. Enrollment

The distance or similarity between the test signature and the template signature referenced in advance is calculated in order to verify that a test signature is genuine. In general, template signatures are selected from the genuine signatures for training. In this paper, however, we use all the genuine training signatures as template data because only a few signatures are available for training.

2.5. Distance Calculation

In this paper, we compute four different distance measures using a Dynamic Programming algorithm. The first one combines three features, vector angle θ, vector length f between pen position and the gravity point, and pen pressure p, instead of treating them separately. The second and third ones are pen inclination distance on azimuth angle and altitude angle. The fourth measure is data length. We calculate the four kinds of distance defined below.

$$\boldsymbol{d} = (d_1, d_2, d_3, d_4) \quad (5)$$

$$d_1 := \min_{\substack{j_s \leq j_{s+1} \leq j_s+1 \\ k_s \leq k_{s+1} \leq k_s+1}} \sum_{s=1}^{S} |\theta_I(j_s) - \theta_T(k_s)| \cdot S_i(p_I(j_s), p_T(k_s)) S_d(f_I(j_s), f_T(k_s)) \quad (6)$$

$$d_2 := \min_{\substack{j_s \leq j_{s+1} \leq j_s+1 \\ k_s \leq k_{s+1} \leq k_s+1}} \sum_{s=1}^{S} |\phi_I(j_s) - \phi_T(k_s)| \quad (7)$$

$$d_3 := \min_{\substack{j_s \leq j_{s+1} \leq j_s+1 \\ k_s \leq k_{s+1} \leq k_s+1}} \sum_{s=1}^{S} |\psi_I(j_s) - \psi_T(k_s)| \quad (8)$$

$$d_4 := |J - K| \quad (9)$$

where $j_1 = k_1 = 1$, $j_S = J$, $k_S = K$, J and K are data length, subscript I means input data, T means template

data and the functions S_i and S_d prevent information impactedness and are defined as

$$S_i(u,v) = \begin{cases} 1 & (u = v) \\ |u-v| & (u \neq v) \end{cases} \quad (10)$$

$$S_d(u,v) = \begin{cases} \nu & (|u-v| \leq \nu) \\ |u-v| & (|u-v| > \nu) \end{cases} . \quad (11)$$

The Dynamic Programming can be used for computing d_1, d_2 and d_3 because of the sequential nature of the distance function. Let

$$\boldsymbol{D} = (\boldsymbol{d}_m, t_m)_{m=1}^{M} \quad (12)$$

$$\boldsymbol{d}_m = (d_{1m}, d_{2m}, d_{3m}, d_{4m}) \quad (13)$$

$$t_m = \begin{cases} 1 & if\ genuine \\ 0 & if\ forgery \end{cases} \quad (14)$$

be the training data set. Template data and training data have a label t which is $t = 1$ if the signature is genuine, $t = 0$ if forgery. Figure 6(a)-(d) depict 2-D scatter plots of the distance measures associated with the generated templates.

2.6. Model and Fusion Strategy

We adopt the model as a criterion function that considers each individual characteristic, instead of discriminating by simply averaging or summing the distance measures. The model for verification is defined in terms of a semi-parametric probability distribution induced by a neural network, and is formulated within a Bayesian framework. We prepare a three-layer perceptron for each individual as a model for the fusion method of distance measures (d_1, d_2, d_3, d_4).

$$f(\boldsymbol{d}; \boldsymbol{w}) := \sigma(\sum_{i=1}^{h} a_i \sigma(\sum_{j=1}^{4} b_{ij} d_j + c_i) + c_0) \quad (15)$$

$$\sigma(u) = \frac{1}{1+e^{-u}} \ , \ \boldsymbol{w} = (a_i, b_{ij}, c_i) \quad (16)$$

Here, h is the number of hidden units.

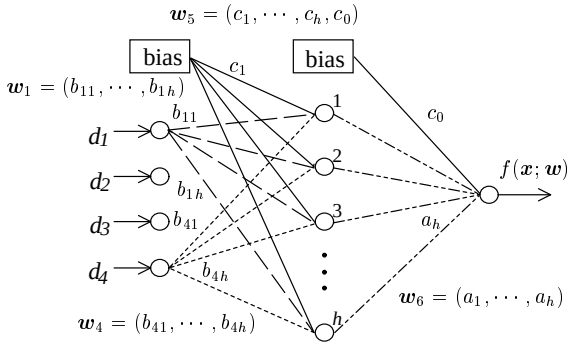
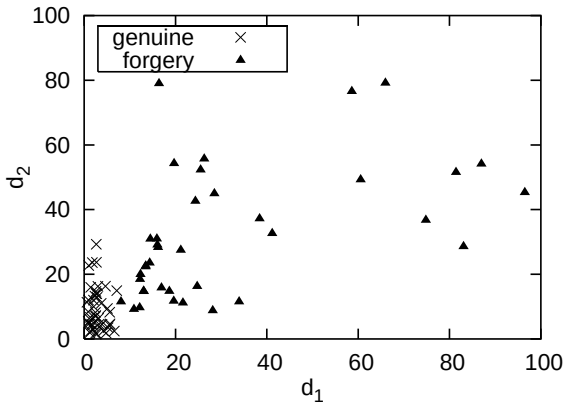


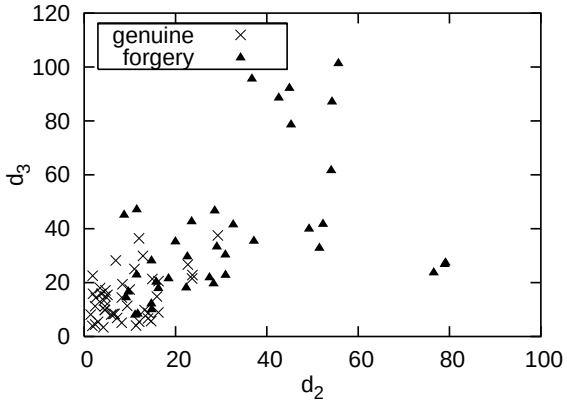
Figure 5. 3-layer perceptron

The parameter $\boldsymbol{w}$ needs to be estimated from training data. We define the following likelihood function induced from (15):

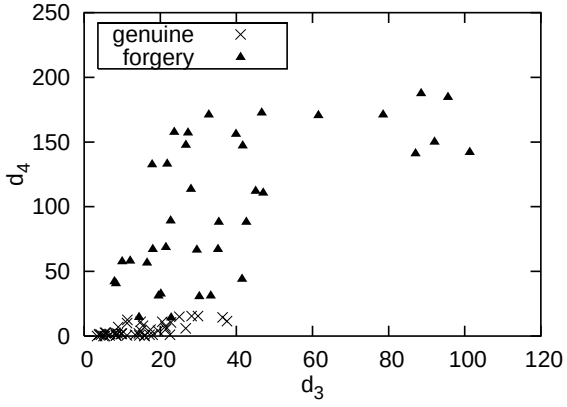
$$P(\boldsymbol{D}|\boldsymbol{w}) = \prod_{m=1}^{M} f(\boldsymbol{d}_m; \boldsymbol{w})^{t_m} (1 - f(\boldsymbol{d}_m; \boldsymbol{w}))^{1-t_m}. \quad (17)$$



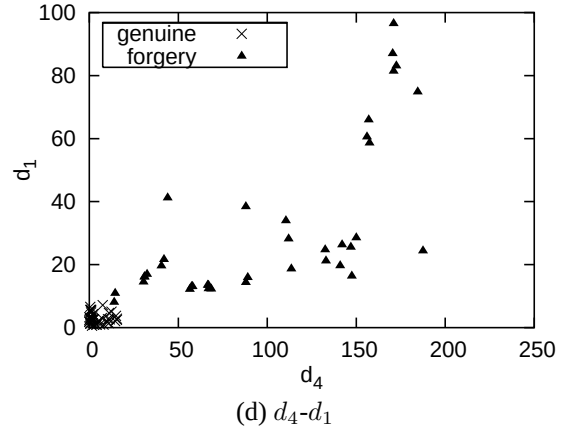
(a) d_1-d_2



(b) d_2-d_3



(c) d_3-d_4



(d) d_4-d_1

Figure 6. The 2-D scatter plots (a) in the (d_1;d_2); (b) in the (d_2;d_3); (c) in the (d_3;d_4); (d) in the (d_4;d_1)-space.

In this paper, the weight parameter vector $\boldsymbol{w}$ is decomposed into following groups as was done in [4]. $\{\boldsymbol{w}_c\}_{c=1}^4$ is weight parameter between each input unit and hidden units, $\boldsymbol{w}_5$ is weight parameter of bias to hidden units and output unit, and $\boldsymbol{w}_6$ is weight parameter between hidden units and output unit. We assign a hyper-parameter α_c to each parameter $\{\boldsymbol{w}_c\}_{c=1}^6$. The prior distribution of $\boldsymbol{w}$ is defined as

$$P(\boldsymbol{w}|\boldsymbol{\alpha}) = \prod_{c=1}^{6} \frac{1}{Z_W(\alpha_c)} \exp(-\frac{\alpha_c}{2}||\boldsymbol{w}_c||^2). \quad (18)$$

This is called the "weight-decay" prior, which favors small magnitudes of the weight parameters and decreases the tendency to over-fitting. Each hyper-parameter α_c controls the degree of weight decay. The weight parameter vector associated with one feature may have different contributions to the predictions than that of the other feature. The degree of contribution of each feature can be controlled by each hyper-parameter α_c. These observations leads to the decomposition shown in Figure 5. This method is called the Automatic Relevance Determination (ARD) [5], and such decomposition has been successfully used in difficult problems. It follows from the Bayes formula that the posterior distributions of $\boldsymbol{w}$ and $\boldsymbol{\alpha}$ are given by

$$P(\boldsymbol{w}|\boldsymbol{D},\boldsymbol{\alpha}) = \frac{P(\boldsymbol{D}|\boldsymbol{w})P(\boldsymbol{w}|\boldsymbol{\alpha})}{P(\boldsymbol{D}|\boldsymbol{\alpha})}$$

$$= \frac{1}{Z(\boldsymbol{\alpha})} \exp(-(-G(\boldsymbol{w}) + \sum_{c=1}^{6} \frac{\alpha_c}{2}||\boldsymbol{w}_c||^2)) \quad (19)$$

$$G(\boldsymbol{w}) = \log P(\boldsymbol{D}|\boldsymbol{w}) \quad (20)$$

$$P(\alpha|\boldsymbol{D}) = \frac{P(\boldsymbol{D}|\boldsymbol{\alpha})P(\boldsymbol{\alpha})}{P(\boldsymbol{D})}. \quad (21)$$

We assume that the prior distribution of $\boldsymbol{\alpha}$ is a Gamma distribution as was done in [4]. Using the posterior distribution of $\boldsymbol{w}$ and $\boldsymbol{\alpha}$, we draw samples from the predictive distribution of label t.

$$P(signature\ is\ genuine) = P(t=1|\boldsymbol{d})$$

$$= \iint f(\boldsymbol{d};\boldsymbol{w})P(\boldsymbol{w}|\boldsymbol{D},\boldsymbol{\alpha})P(\boldsymbol{\alpha}|\boldsymbol{D})d\boldsymbol{w}d\boldsymbol{\alpha} \quad (22)$$

It is often impossible to express the posterior distribution of $\boldsymbol{w}$ and $\boldsymbol{\alpha}$ analytically because of the complexity of the posterior distribution landscape. We use MCMC to draw samples from the posterior distribution.

2.6.1. Markov Chain Monte Carlo

MCMC is used to draw samples from the posterior distribution of $\boldsymbol{w}$. The MCMC scheme consists of two different operations for obtaining samples (Figure 7). The Metropolis scheme and Gibbs Sampling are alternated to obtain numerous L samples of $\boldsymbol{w}$ and $\boldsymbol{\alpha}$. The samples from the first K iterations are discarded because of their initial value dependency. Samples of the last $S=(L-K)$ iterations of samples $\{\boldsymbol{w}^{(l)}\}_{l=1}^S$ are used for verification.

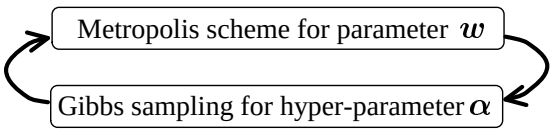


Figure 7. Schematic diagram of the MCMC

2.6.2. Metropolis Algorithm

Let $\boldsymbol{w}^{(l)}$ and $\alpha^{(l)}$ be a sample drawn in the previous stage l. We want to draw samples $\boldsymbol{w}^{(l+1)}$ from $P(\boldsymbol{w}|D,\alpha^{(l)})$, based on the assumption that evaluation of the target distribution up to normalizing a constant is analytically possible, even though a full evaluation is impossible. The transition from $\boldsymbol{w}^{(l)}$ to $\boldsymbol{w}^{(l+1)}$ proceeds as follows.

Metropolis algorithm for $\boldsymbol{w}^{(l+1)}$
Step 1.
Generate a candidate state $\boldsymbol{w}^*$ from a proposal distribution $Q(\boldsymbol{w}|\boldsymbol{w}^{(l)})$ that may depend on the current state.

Step 2.
If $P(\boldsymbol{w}^*|\boldsymbol{D},\alpha^{(l)}) \geq P(\boldsymbol{w}^{(l)}|\boldsymbol{D},\alpha^{(l)})$, accept the candidate state $\boldsymbol{w}^*$.
Else, accept the candidate state with probability $P(\boldsymbol{w}^*|\boldsymbol{D},\alpha^{(l)})/P(\boldsymbol{w}^{(l)}|\boldsymbol{D},\alpha^{(l)})$.

Step 3.
If the candidate state is accepted, let $\boldsymbol{w}^{(l+1)} = \boldsymbol{w}^*$.
Else, let $\boldsymbol{w}^{(l+1)} = \boldsymbol{w}^{(l)}$.

2.6.3. Gibbs Sampling

Let $\boldsymbol{w}^{(l+1)}$ be the l+1th sample obtained in the Metropolis step described above. Suppose we wish to draw samples from the joint distribution for $\boldsymbol{\alpha}$=$\{\alpha_1,\cdots,\alpha_C\}$. The Gibbs sampler does this by repeatedly replacing each component with a value drawn from its distribution conditional on the current values of all other components.

Gibbs Sampling for $\alpha^{(l+1)}$
Step 1.
$\alpha_1^{(l+1)} \sim P(\alpha_1|\alpha_2^{(l)},\alpha_3^{(l)},\cdots,\alpha_C^{(l)},\boldsymbol{w}^{(l+1)})$
Step 2.
$\alpha_2^{(l+1)} \sim P(\alpha_2|\alpha_1^{(l+1)},\alpha_3^{(l)},\cdots,\alpha_C^{(l)},\boldsymbol{w}^{(l+1)})$
$\vdots$
Step C.
$\alpha_C^{(l+1)} \sim P(\alpha_C|\alpha_1^{(l+1)},\alpha_2^{(l+1)},\cdots,\alpha_{C-1}^{(l+1)},\boldsymbol{w}^{(l+1)})$

2.7. Decision Making

Probability $P(signature\ is\ genuine)$ is generated with the input test data $\boldsymbol{d}$ and samples $\{\boldsymbol{w}^{(l)}\}_{l=1}^S$,

$$P(signature\ is\ genuine) = P(t=1|\boldsymbol{d}) \quad (23)$$

$$\approx \frac{1}{S}\sum_{l=1}^{S} f(\boldsymbol{d};\boldsymbol{w}^{(l)}) \quad (24)$$

$$\boldsymbol{w}^{(l)} \sim P(\boldsymbol{w}|\boldsymbol{D}). \quad (25)$$

Compute $P(t=1|\boldsymbol{d})$ and decide

$$Signature\ is \begin{cases} genuine & if\ P(t=1|\boldsymbol{d}) \geq threshold \\ forgery & if\ P(t=1|\boldsymbol{d}) < threshold \end{cases} \quad (26)$$

2.8. Parameter Updating

The method of updating templates and model parameters is a promising approach to adapting to deterioration of authentication performance resulting from intersession variability in signatures. This paper proposes an algorithm with an integrated scheme for updating model parameters. We need to use test data to update the parameter for adapting to intersession variability. However, test data has no supervised label t. Therefore, we assign a label to the test data using the Decision Directed method [6]. The predicted probability of label t_n produced by the n–1th model and the thresholds for learning are used to assign a label to the nth test data according to the following rule.

$if\ P(t_n = 1) \geq threshold_{genuine}$
assign t_n with 1 and use for updating
$if\ P(t_n = 1) \leq threshold_{forgery}$
assign t_n with 0 and use for updating
$otherwise$
not use for updating

In the updating phase, we use Sequential Monte Carlo (SMC) [7].

2.8.1. Sequential Monte Carlo

Let $\boldsymbol{w}^{(l)}$ be samples used in a previous decision-making step, and $\boldsymbol{w}^{(l)}$ is rewritten as $\boldsymbol{w}_{n-1}^{(l)}$.

$$\boldsymbol{w}_{n-1}^{(l)} \leftarrow \boldsymbol{w}^{(l)} \quad (27)$$

The procedure to obtain the sample $\boldsymbol{w}_n^{(l)}$ with the nth test signature is shown below.

Step 1 Updating $\boldsymbol{w}$
Define the normalized importance weight $\overline{\Omega}$ for each sample $\boldsymbol{w}_{n-1}^{(l)}$.

$$\overline{\Omega}(\boldsymbol{w}_{n-1}^{(l)}) = \frac{P(D_n|\boldsymbol{w}_{n-1}^{(l)})}{\sum_{m=1}^{S} P(D_n|\boldsymbol{w}_{n-1}^{(m)})} \quad (28)$$

Resample $\boldsymbol{w}$ according to (28).

$$\boldsymbol{w}^{*(l)} \sim \sum_{l=1}^{S} \overline{\Omega}(\boldsymbol{w}_{n-1}^{(l)}) \delta(\boldsymbol{w} - \boldsymbol{w}_{n-1}^{(l)}) \quad (29)$$

Step 2 Prediction
Using samples $\{\boldsymbol{w}^{*(l)}\}_{l=1}^{S}$ drawn from the posterior distribution of parameter $\boldsymbol{w}$, draw samples of parameter $\{\boldsymbol{w}_n^{(l)}\}_{l=1}^{S}$ from the following distribution.

$$\boldsymbol{w}_n^{(l)} \sim P(\boldsymbol{w}_n|\boldsymbol{w}^{*(l)}) \quad (l = 1, ..., S) \quad (30)$$

The transition distribution of the weight parameter is assumed to be Gaussian $\mathcal{N}(0, \gamma^{-1})$.

$$\boldsymbol{w}_n = \boldsymbol{w}^* + \mu \quad \mu \sim \mathcal{N}(0, \gamma^{-1}) \quad (31)$$

Set

$$\boldsymbol{w}^{(l)} \leftarrow \boldsymbol{w}_n^{(l)} \quad (32)$$

and verify the next test signature in (24).

3. Experiment

3.1. Database

We used BIOMET database's signatures acquired using WACOM intuos2 A6, with an ink pen [8]. The sampling frequency of the acquired signal was 100Hz. We obtained the subset of BIOMET, which includes the signatures from 84 individuals. The signatures were captured in two sessions with a five-month interval between sessions. Five genuine signatures and six forgeries were collected in the first session, and ten genuine signatures and six forgeries were collected in the second session. In this study, we used a subset of 61 individuals from the 84 individual BIOMET database because signatures for some of the 23 total individuals were missing.

Table 1. Data set for experiment

	Training data		Test data	
	Genuine	Forgery (Random)	Genuine	Forgery (Skilled)
Each individual	5	10	10	12
Total	305	610	610	732

3.2. Forgery

A signature consists of a genuine signature written by a user, while a forgery is maliciously written by a forger. A forgery is categorized into one of four types.

1. Random Forgery
   A forged signature written when the forger does not have access to the genuine signature.
   Other genuine signatures were used as random forgeries in this case because of the ease of acquisition.
2. Simple Forgery
   A forged signature written by a forger who knows only the name of the person who wrote the genuine signature.
3. Simulated Forgery
   A forged signature written by a forger who receives off-line data and can trace the genuine signature.
4. Skilled Forgery
   A forged signature written by a forger who can view and practice the genuine signature. The skilled forgery is generally considered to be the most difficult to identify.

3.3. Experimental Setting

The training data set consisted of the five genuine signatures from session 1 and 10 random forgeries (all genuine signatures of individuals other than the target user) for each of the 61 individuals. The test dataset included a genuine signature from session 2 and skilled forgery from sessions 1 and 2. The proposed algorithm was evaluated by comparing it with the algorithm which does not have an updating phase. In this experiment, Q is normal distribution with mean $\boldsymbol{w}^{\boldsymbol{l}}$ and variance ν. The details of some parameters are indicated below.

number of hidden units h	8
number of leanings L	20000
number of samples S	1000
hyper-parameter γ (SMC)	100
variance ν in Metropolis	0.1
$threshold_{genuine}$	0.8
$threshold_{forgery}$	0.2

3.4. Result

The experiment results are presented in Table 2. The Equal Error Rate (EER) is often used for performance assessment. The EER is the error rate where the FAR (False Acceptance Rate) equals the FRR (False Rejection Rate), determined by fluctuating the threshold.

Table 2. Verification results of experiment

	Previous (Without updating)	Proposed (MCMC+SMC)
EER (%)	8.85	6.39

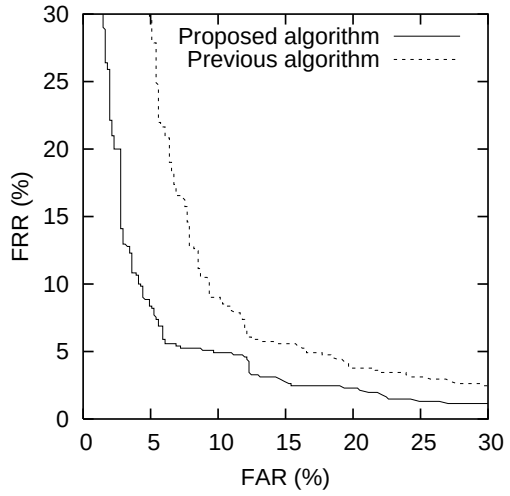


Figure 8. Error trade-off curve

4. Conclusion

We developed an parameter-updating algorithm for on-line signature verification considering deterioration of verification performance caused by intersession variability in signatures, and evaluated it using a BIOMET database, including genuine signatures with five months intersession variability. The comparison between the proposed algorithm and the previous algorithm, which does not have an updating phase, demonstrates that the proposed algorithm outperforms the previous algorithm. However, there is a problem involving the adjustment of the hyper-parameter γ during the updating phase. When hyper-parameter γ is large, the parameter is hardly updated. In contrast, when it is small, the parameter is updated depending on a single test signature, causing degradation of verification performance. Therefore, appropriate adjustment of the hyper-parameter is important. We are planning to implement an automatic adjustment of the hyper-parameter via SMC as a step toward solving this problem. Other future projects include further analysis of the correlation between the passage of time and the variability of signature.

References

[1] Rejean Plamondon and Guy Lorette, "Automatic signature verification and writer identification - the state of the art," *Pattern Recognition*, vol. 22, no. 2, pp. 101–131, 1989.

[2] S. Yamanaka, M. Kawamoto, T. Hamamoto, and S. Hangai, "Signature verification adapting to intersession variability," *International Conference on Multimedia and Expo*, pp. 89–92, 2001.

[3] B.Ly Van, S. Garcia-Salicetti, and B. Dorizzi, "Fusion of hmm's likelihood and viterbi path for on-line signature verification," *Biometrics Authentication Workshop (BIOAW)*, pp. 318–331, 2004.

[4] D. Muramatsu, M. Kondo, M Sasaki, S. Tachibana, and T. Matsumoto, "A markov chain monte carlo algorithm for bayesian dynamic signature verification," *IEEE Trans. Information Forensic and Security.*, 2006.(in press).

[5] R. M. Neal, "Bayesian learning for neural networks," Springer Verlag, 1996.

[6] F. Roli, "Semi-supervised multiple classifier systems: Background and research directions," *Multiple Classifier Systems*, pp. 1–11, 2005.

[7] A. Doucet, N. de Freitas, and N. Gordon, "An introduction to sequential monte carlo methods," in *Sequential Monte Carlo methods in practice*, A. Doucet, N. de Freitas, and N. Gordon, Eds., pp. 3–14. Springer Verlag, 2001.

[8] S. Garcia-Ssalicetti, C. Beumier, G. Chollet, B. Dorizzi, J. Leroux-Les Jardins, J. Lanter, Y. Ni, , and D. Petrovska-Delacretaz, "Biomet: a multimodal person authentication database including face, voice, fingerprint, hand and signature modalities," *4th International Conference on Audio- and Video-Based Biometric Person Authentication*, pp. 845–853, 2003.