CSIT (June 2014) 2(2):129–140
DOI 10.1007/s40012-014-0054-4

ORIGINAL RESEARCH

Continuous multimodal user authentication: coupling hard and soft biometrics with support vector machines to attenuate noise

K. G. Srinivasa · Soumya Gosukonda

Received: 22 December 2013 / Accepted: 5 August 2014 / Published online: 2 September 2014

Abstract Static Authentication provides a secure framework for a one-time authentication session, but fails to authenticate the user throughout the session. This presents the possibility of an imposter gaining access when a user session is active and the user moves away from the system. The goal of continuous authentication is to authenticate the user right from the initial stages of log-in till log-out. Intuitively, this can be implemented by extrapolating the tried-and-tested static authentication techniques throughout the session. However, extrapolating one-time authentication techniques poses new challenges of being computationally expensive, restricting the user's movement and postures in front of the system, depending on extra expensive hardware and deviating the user from his workflow. In these situations, the user no longer remains uninterrupted by the authentication process in the background. The proposed framework provides unobtrusive Continuous Authentication, by alternating between two modes which utilize hard and soft biometrics respectively, depending on certain confidence parameters. We use facial features as the hard biometric trait for recognizing the user. Employing face recognition for extended periods of time produces noise, which is dampened by using a supervised machine learning algorithm. The color of user's clothing as the soft biometric trait relieves the CPU of comparatively high computation and relaxes constraints on the user's upper body movement.

K. G. Srinivasa (✉) · S. Gosukonda
Department of Computer Science and Engineering, M. S. Ramaiah Institute of Technology, Bangalore 560054, India
e-mail: srinivasa.kg@gmail.com; kgsrinivas@msrit.edu

S. Gosukonda
e-mail: soumya.gk@gmail.com

1 Introduction

Authentication in the context of computer security is a process which verifies the claimed identity of the user. Upon successful authentication, the user may be granted privileges enabled by a higher authority. A number of elements together can decide the authenticity of the user. These elements can be classified based on three factors called *authentication factors*. Security research [1] has determined that for a positive identification, atleast two of the three authentication factors need to be satisfied. These factors are:

- *Knowledge factors* Something the user *knows* (e.g., Username-password pair)
- *Ownership factors* Something the user *has* (e.g., ID card, cell phone, security token)
- *Inherence factors* Something the user *is* (e.g., Fingerprint, retinal patterns, facial features)

The purpose of this work is to provide unobtrusive continuous user authentication techniques by taking into account these authentication factors.

The process of authentication can be either *static* or *continuous*. *Static Authentication* refers to the method of authenticating a user at the time of log-in. In most cases, knowledge-based methods such as passwords are used since every user can verify his/her claim in a very convenient manner. But passwords and knowledge factors lose their credibility when shared, forgotten or stolen. Similarly, ownership-based methods too can duplicated, stolen or lost. When dealing with sensitive content, one might resort to using additional equipment which verifies the user based on his unique traits such as fingerprints or retinal patterns. In this case, the only period during which the system is very confident and fully aware of user's identity claim is during the authentication period. Some computer systems use a 2-factor authentication technique which verifies the user's password and facial features at the time of log-in and assumes it is the same authenticated user till logout. But when the user moves away from the system to take a break without logging-out, it is susceptible to tailgating - where an imposter takes the authenticated user's place. This could prove to be a critical security weakness in high-security systems. In order to address this issue, the system needs to continuously verify the user's identity claim. This is the goal of *Continuous Authentication*.

Biometrics is emerging as a favorable choice around which Continuous Authentication systems can be modeled. *Biometric traits* refer to the physiological or behavioral traits of a user that can identify a user for a session. These traits can be divided into the following two categories [2]:

- *Hard Biometric traits* These are physical traits of a user that are assumed to be present universally and can uniquely identify an individual. For example, fingerprints, facial features, DNA and so on.
- *Soft Biometric traits* These are characteristics of a user that "provide some information about the individual, but lack the distinctiveness and permanence to sufficiently differentiate any two individuals"[3]. For example, color of clothing/skin/eye/hair, gender and other such factors.

The motivation for this work arises from the flaws in the present-day authentication systems which solely rely on static techniques. Consider the online marketplaces or stores on mobile devices and personal computers. They are enabled with a pre-authorized account in which the user is not prompted for any credentials before making purchases. Using this type of account circumvents the need to enter account credentials before making a purchase. Thus, convenience for the user is achieved at the risk of an unauthorized person making fraudulent purchases. Another instance would be web services such as email, file-sharing, etc. that are accessible on public systems. Once again, static authentication techniques are employed which do not address the possibility of tailgating-a situation where a person forgets to log out allowing unauthorized access to their data.

A possible solution to this problem is verifying the biometrics that characterize a user throughout a session. But, most conventional biometric techniques have low availability, i.e., they are hard to capture continuously since the user cannot be assumed to remain in one particular position for the duration of the session. For example, methods such as fingerprint and retinal pattern recognition prove to be inconvenient to the user since posture would be restricted. Even facial features restrict the user to face the camera and features not captured in the training data surface as noise in practice. Continuous authentication using biometrics forces log-in multiple times when the user moves away from the data access point. This can occur in the case of a medical practitioner accessing patient records and moving away to treat the patient. Hence, solely using conventional biometrics disrupts the user's natural workflow and can be considered obstrusive.

This paper presents an approach towards implementing continuous authentication on a personal system in an unobtrusive manner. By unobtrusive, we mean the user need not incur the cost of deviating from his normal workflow to enjoy the benefit of continuous authentication. Integrated in this implementation are modules—which utilize hard and soft biometrics separately—to continuously verify the user throughout the session. We use the facial features of the user as the hard biometric trait, which is captured using an image-based face recognition algorithm—Eigenfaces. For the soft biometric measure, our choice was to use the color of user's clothing since it is more tolerant towards the user's posture. The control flow of the system alternates between the hard and soft biometrics module depending on the need to reinforce the system's belief.

We view this work as having three contributions.

(1) We provide an overview of the framework for continuous authentication implemented as separate modules. These modules emit individual confidence as a function of one of the biometric trait being captured at a given point in time.
(2) In order to use facial features, we augmented the face recognition algorithm with a Support Vector Machine to dampen noise. This is necessary primarily for two reasons. First, face recognition algorithms, including modern ones arent highly accurate. Hence, the features not captured during training surface as false recognitions, i.e., noise. Second, when the face recognition database is biased—such as when there exists only a single user or contains face images of only one gender—falsely recognizing an imposter becomes highly likely in some cases.
(3) We identify that the Support Vector Machine when trained on a fixed number of features captured in the past N frames, benefits from decaying old data and learning to make more confident predictions based only on the recent ones.

We readily acknowledge that this paper does not present any fundamental contributions to the field of machine learning or computer vision. Rather, we focus on integrating the end-to-end machine learning and computer vision algorithms on a personal system to provide continuous unobtrusive user authentication. This is achieved by writing the program in a high-level language which provides wrappers to various user interfaces or with the operating system itself.

2 Related work

The study conducted by Klosterman et al. [4] in the design of biometric-enhanced authentication system describes the challenges posed by such a system. It points out that for unobtrusive continuous monitoring of a user's biometric traits, the choice of trait should be such that it does not hinder the user from working. A biometric measure such as facial features of a user, hence make for an acceptable choice for continuous unobtrusive monitoring, as compared to fingerprints or iris patterns. Another important observation is that biometrics are expensive to compute. In case of facial features, the image processing and recognition algorithms can be computationally much more expensive as compared to password verification. This serves as a compelling reason to interleave the authentication process using an alternative measure which is: inexpensive to compute and allows user more flexibility in posture. In our case, the alternative measure is the soft biometric information of the user.

Incorporating Soft Biometric traits for improved accuracy of recognition, for static authentication, was introduced by Jain et al. [3]. It proposed a framework for integrating the soft biometric information along with the output of the primary biometric system, with fingerprint as the primary biometric identifier and gender, ethnicity, and height as the soft biometric variables.

Using Soft Biometrics for Continuous Authentication has been studied using different types of Soft Biometric traits such as color of clothing and skin [2], keystroke dynamics [5] and electrocardiogram data [6]. Of these the most relevant is the work by Niinuma et al. where they implemented checking the color of shirt and skin continually to match the template created at the start of the session [2].

They conducted experiments where the user was asked to enact 6 typical scenarios a user may normally exhibit such as turning head in different directions, stretching arms or walking away. A measure of the system's performance for soft biometrics was made by evaluating two main parameters-False Reject and False Accept. Their experimental results indicated an overall False Rejection rate of 4.16% and a False Accept rate of 0% over the considered scenarios. While this model yields fairly good results, it leaves scope for improvement of recognition by considering temporal information. This implies a decay of data beyond a certain time in the past. We exploit this temporal information by using a supervised machine learning algorithm, to improve the accuracy of recognition of user in the Hard Biometrics phase.

3 Background

In this section we briefly go over the algorithms implemented in the system. This is essential to understanding the individual roles played by the components of the proposed continuous authentication system.

3.1 Face detection using Viola–Jones algorithm

The face detector proposed by Viola and Jones [8] combines four key concepts [9]:

- Simple rectangular features, called Haar features
- An Integral Image for rapid feature detection
- A variant of the learning algorithm AdaBoost
- Cascaded architecture

The features that Viola and Jones used are based on Haar wavelets. Haar wavelets are single wavelength square waves (one high interval and one low interval). In two dimensions, a square wave is a pair of adjacent rectangles—one light and one dark. The actual rectangle combinations used for visual object detection are not true Haar wavlets. Instead, they contain rectangle combinations better suited to visual recognition tasks. Because of that difference, these features are called Haar features, or *Haarlike features*, rather than Haar wavelets. The presence of a Haar feature is determined by subtracting the average dark-region pixel value from the average light-region pixel value. If the difference is above a threshold (set during learning), that feature is said to be present.

To determine the presence or absence of hundreds of Haar features at every image location and at several scales efficiently, Viola and Jones used a technique called an *Integral Image*. Using this technique, rectangular features can be evaluated in constant time, which gives them a considerable speed advantage over their more sophisticated relatives.

Viola and Jones combined a series of *AdaBoost classifiers* as a filter chain, that is especially efficient for classifying image regions. Each filter is a separate AdaBoost classifier with a fairly small number of weak classifiers. The cascade architecture has interesting implications for the performance of the individual classifiers. Because the activation of each classifier depends entirely on the behavior of its predecessor, the false positive rate for an entire cascade is:

$$F = \prod_{i=1}^{K} f_i \quad (1)$$

Similarly, the detection rate is:

$$D = \prod_{i=1}^{K} d_i \quad (2)$$

Thus, to match the false positive rates typically achieved by other detectors, each classifier can get away with having surprisingly poor performance. At the same time, however, each classifier needs to be exceptionally capable if it is to achieve adequate detection rates.

3.2 Face recognition using Eigenfaces

3.2.1 *Face recognition using Eigenfaces*

Eigenfaces is a face recognition algorithm that was first described by Turk and Pentland [7]. It works to capture the variations present among the images of faces, that form the training set. It uses this information to create a face model where each image is represented as an eigenvector in what is called a PCA subspace. Recognition of a test face is carried out by converting the test image into a similar eigenvector and then comparing its distance from all others in the subspace. The person corresponding to closest image is said to be the person recognized in the test image. It encodes the complete face characteristics, as opposed to capturing features of the face separately.

3.2.2 *Eigenface generation*

Eigenfaces are generated in the following manner, as described in [7]:

1) During the account creation phase, a series of face images are captured and preprocessed. Hence a training dataset **S** of preprocessed face images containing $\tau_1, \tau_2, ...\tau_M$ is prepared. Each image is then converted to a vector of size **N** by concatenating all the pixels row by row. These vectors are put into a matrix **T** with each row representing an image.
2) The average of all vectors ψ is calculated and subtracted from each of the vectors in **T** to obtain vectors $\phi_i, i = 1, 2, ..., n$.
3) The eigenvectors u_k and eigenvalues $\lambda_k, k = 1, ..., M$ of the co-variance matrix **C** are calculated. The covariance matrix itself is found by:

$$\mathbf{C} = \frac{1}{M}\sum_{n=1}^{M} \phi_n \phi_n^T \tag{3}$$

4) Since the dimension of C is very high (of the order of the number of pixels in the image), another matrix **L** as analyzed in [7] with the dimensions $M \times M$ is constructed,

$$\mathbf{L} = \mathbf{A}^T\mathbf{A} \tag{4}$$

where

$$\mathbf{A} = \{\phi_1, \phi_2, ..., \phi_M\} \tag{5}$$

5) The eigenvectors v_l of the matrix **L** are determined such that,

$$u_l = \sum_{k=1}^{M} v_{lk}\phi_k \tag{6}$$

where $l = 1, ..., M$.

To recognize a face, the face image is transformed into its eigenface components. The input image τ_{new} is compared with the mean image and their difference is multiplied with each eigenvector of the **L** matrix. Each value represents a weight and would be saved on a vector Ω.

$$\omega_k = u_k^T(\tau_{new} - \psi)\Omega^T = [\omega_1, \omega_2, .., \omega_k] \tag{7}$$

The Euclidiean distance ε is minimized to determine which face class the new face belongs to. It is computed as follows [10]:

$$\varepsilon_k = \parallel \Omega - \Omega_k \parallel \tag{8}$$

If ε_k is below an established threshold θ_ε, then the input face is considered to belong to that respective class.

3.3 Noise dampening using support vector machines

Using the output produced by the face recognition module, the goal is to dampen the noise, by taking advantage of certain features like

- Temporal information
- Confidence in prediction of recognized face, which in case of Eigenfaces is the Mahalanobis distance of the projected point from its nearest neighbour
- Patterns inherent in the noise

The following theory on Support Vector Machines is as discussed in [11] Noise attenuation is achieved by training a classifier on existing data represented as:

$$\mathcal{D} = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\} \tag{9}$$

where $(x^{(i)}, y^{(i)})$ represents the *ith* training example in a set of m training examples and $x^{(i)} \in \mathbb{R}^n, y^{(i)} \in \{+1, -1\}$. We generate a hyperplane represented as:

$$h_{w,b}(x) = g(w^Tx + b) \tag{10}$$

where

$$g(z) = \begin{cases} +1 & if\ z \geq 0 \\ -1 & if\ z < 0 \end{cases} \tag{11}$$

The primal optimization problem for finding the optimal margin classifier can be stated as:

$$\min_{\gamma,w,b} \frac{1}{2} \parallel w \parallel^2 \tag{12}$$

subject to the constraint

$$y^{(i)}(w^Tx + b) \geq i = 1, ..., m \tag{13}$$

When we construct the Lagrangian for this optimization problem, we have:

$$\mathcal{L}(w,b,\alpha) = \frac{1}{2} \parallel w \parallel^2 - \sum_{i=1}^{m} \alpha_i [y^{(i)}(w^T x^{(i)} + b) - 1] \quad (14)$$

where α_i is a Lagrange multiplier. To find the dual form, we need to minimize $\mathcal{L}$ which is obtained by differentiating this equation with respect to w and b. Therefore, by taking the derivative with respect to w and setting it to zero:

$$\nabla_w \mathcal{L}(w,b,\alpha) = w - \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} = 0 \quad (15)$$

$$\Rightarrow w = \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} \quad (16)$$

Similarly, the derivative with respect to b:

$$\frac{\partial}{\partial b} \mathcal{L}(w,b,\alpha) = \sum_{i=1}^{m} \alpha_i y^{(i)} = 0 \quad (17)$$

Plugging this back into Eq. 14, we get

$$\mathcal{L}(w,b,\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j (x^{(i)})^T x^{(j)} \quad (18)$$

Putting this together with the constraint $\alpha_i \geq 0$, we obtain the following the dual optimization problem:

$$\max_{\alpha} W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle \quad (19)$$

subject to the constraints

$$\alpha_i \geq 0, i = 1, 2, \ldots, m \quad (20)$$

and

$$\sum_{i=1}^{m} \alpha_i y^{(i)} = 0 \quad (21)$$

It can be verified that the conditions required for $p^* = d^*$ and the KKT conditions to hold are satisfied in this optimization problem. By finding the α's given in Eq. 19, which maximizes $W(\alpha)$, the optimal w's can be represented as a function of α's. Having found w^*, by considering the primal problem, we obtain the orientation of the hyperplane. The optimal value of intercept term b can be calculated as:

$$b^* = -\frac{max_{i:y^{(i)}=-1} w^{*T} x^{(i)} + min_{i:y^{(i)}=1} w^{*T} x^{(i)}}{2} \quad (22)$$

Suppose the model's parameters w and b are fit to the training set, a prediction would require calculate $w^T x + b$ for a new point x. This quantity can be written as:

$$w^T x + b = \left(\sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} \right)^T x + b \quad (23)$$

$$= \sum_{i=1}^{m} \alpha_i y^{(i)} \langle x^{(i)}, x \rangle + b \quad (24)$$

Thus, if the value of α's have been calculated, in order to make a prediction, a quantity that only depends on the inner product of the new point and training data needs to be calculated. Let this quantity be represented as $\langle x, z \rangle$. Given a feature mapping ϕ, this inner product can be entire replace by $\langle \phi(x), \phi(z) \rangle$. A Kernel can now be defined as:

$$K(x,z) = \phi(x)^T \phi(z) \quad (25)$$

In our proposed solution, we use radial basis function as the kernel:

$$K(x,z) = exp\left(\frac{\parallel x - z \parallel^2}{2\sigma^2} \right) \quad (26)$$

In the next section we show how the model obtained is used to solve the learning problem.

4 Design and implementation of the continuous authentication system

In this section, we first look at the high-level overview in Sect. 4.1, in terms of the modules used and design choices. In Sect. 4.2 we discuss the implementation details of the proposed system.

4.1 Architecture

The control flow in the proposed work exists in the following three states:

(1) Conventional password log-in
(2) Hard biometrics mode
(3) Soft biometrics mode

These states are implemented by designing the system as shown in Fig. 1.

Upon successful password log-in, the control alternates between the hard and soft biometrics mode. These transitions are made depending on confidence parameters θ_H and θ_S respectively.

4.1.1 Hard biometrics

Systems solely relying on knowledge-based authentication factors such as passwords are vulnerable when the account credentials are stolen. This is circumvented by capturing

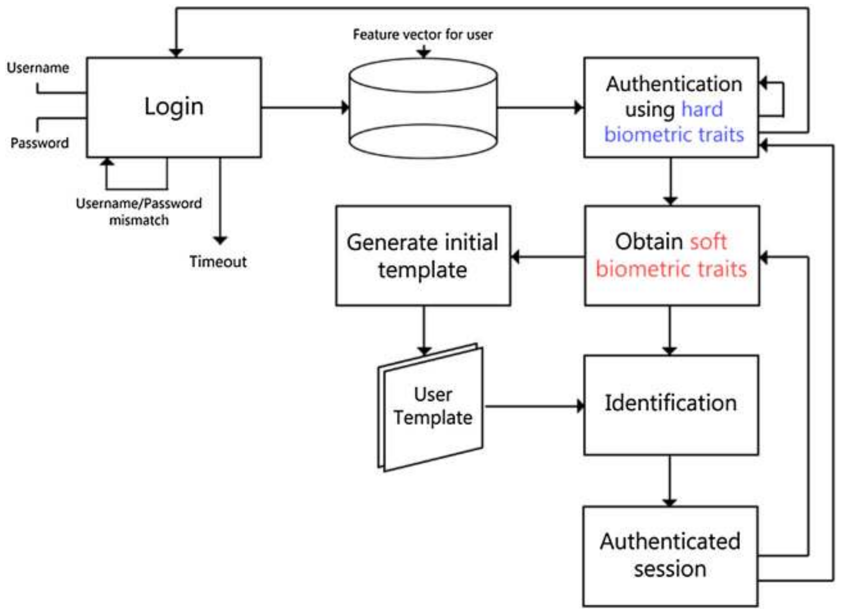
Feature vector for user
Username
Password
Login
Username/Password mismatch
Timeout
Authentication using hard biometric traits
Generate initial template
Obtain soft biometric traits
User Template
Identification
Authenticated session

Fig. 1 Continuous Authentication system overview

hard biometrics traits after a conventional password log-in. Use of these traits allows robust authentication based on features that are unique per user.

The proposed work uses facial features of the user as the hard biometric trait. By using facial features, we are able to authenticate the user in an unobtrusive manner. Alternatives such as retinal patterns and finger-print recognition prove inconvenient since they distract the user when employed continually. These alternatives also require additional expensive hardware and increase the cost of the system.

Face recognition using Eigenfaces [7], an image-based face recognition algorithm is used to determine the authentication state by capturing the facial features of the user from a live video stream. Eigenfaces works by learning facial parameters of the user during training. It finds a PCA subspace whose basis vectors correspond to the variance in the face images. During testing, the face image captured from the web-cam is projected into the PCA subspace and components of the eigenvectors that represent the face image are calculated. The nearest neighbor of this projected point is then presented as the recognized user.

We are however limited by the drawbacks of Eigenfaces, such as its low accuracy when lighting conditions differ or the user is viewed at different angles. Although more advanced algorithms for image-based and video-based face recognition are available, we attempt to overcome the drawbacks presented by Eigenfaces as explained in the next section. Moreover, since we have laid out a generic framework which interfaces with a face recognition module (Eigenfaces in this case), it can be replaced with a better performing module without causing side-effects.

4.1.2 Noise dampening

Image-based face recognition algorithms do not take into account recognition over a video stream and are not designed to meet real-time constraints. Advances in video-based face recognition algorithms are designed to recognize faces after processing the recorded video sequence. But, the need to continuously authenticate the user requires face recognition to meet real-time constraints on a live video stream.

Eigenfaces recognizes the user by predicting $user_{recognized}$ for each frame in the video stream. If the output over an extended period of time is represented by

$$1\{user_{recognized} = user_{authenticated}\} \in \{0, 1\} \tag{27}$$

for each prediction, the output appears as shown in Fig. 2. The data used to produce the figure was obtained with the authenticated user in front of the system under conditions deviating from normal. Since Eigenfaces is an image-based face recognition algorithm, the changes in user's postures were exhibited as false negatives. False positives also easily surface, such as in the scenario where in only a single user is registered. When the Eigenfaces algorithm projects any user's facial traits on to the PCA subspace, this single registered user is always the nearest neighbour to the projected point. Hence, all predictions made by the Eigenface algorithm for any user other than the registered user are false positives, and we say that the database is biased in such a case. The noise - false positives and false negatives leads to the authenticated user being recognized as an imposter or vice-versa.

We show that the factors that help dampen this noise are:

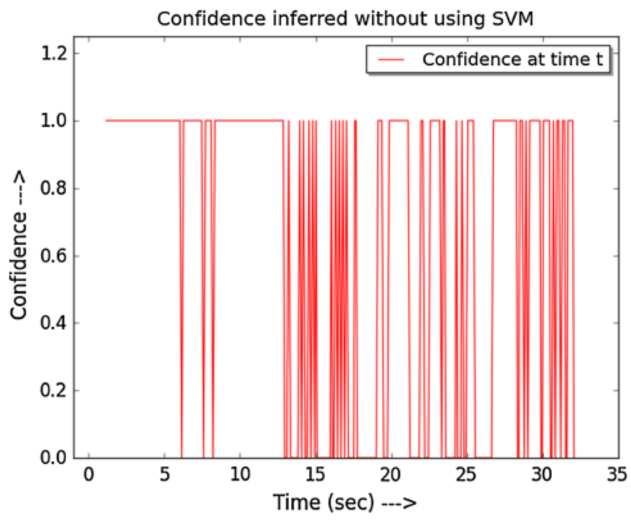

Fig. 2 Output of Eigenfaces versus time as described by the indicator function

- Confidence in prediction by the face recognition algorithm
- Temporal patterns exhibited by authentication predictions
- Time since user's last confident authentication

Our proposed technique overcomes these drawbacks using a supervised learning algorithm by training over features numerically expressing these factors.

4.1.3 Soft biometrics

Hard biometric traits provide confident predictions using unique-per-user features at the cost of:

- Consuming more time for processing each frame
- Restricting the user's postures and movements
- Falsely rejecting user due to occlusion and contrasting changes in facial expression.

Using Soft biometrics, we are able to overcome these drawbacks and provide unobtrusive continuous authentication.

When the confidence of the system in the hard biometrics mode exceeds a given threshold, the control flow transitions into this mode. Just before entering this phase, the soft biometric traits of the user are enrolled in a template [2]. This allows the system to authenticate the user using relaxed unique-per-session traits.

The proposed solution uses color of the user's clothing as a soft biometric trait. During template enrollment, the Hue-Saturation-Value(HSV) of the clothing is recorded. The soft biometric confidence is then calculated as the similarity between this enrolled template and the HSV of the user's clothing in every subsequent frame. When this confidence falls below a certain threshold, the control flow enters the hard biometrics mode.

4.2 Implementation

The proposed continuous authentication system is implemented on GNU/Linux, with majority of the code written in C++ and house-keeping tasks for creating and reorganizing training data written in Python 2.7. We use OpenCV [12] for image processing tasks. Using its support for Viola-Jones object detection algorithm and Eigenfaces, we were able to implement face detection and recognition, respectively. Our choice of the supervised learning algorithm is Support Vector Machine [13], implemented using libSVM [14].

Our solution involves an account creation phase(Training) and a continuous authentication phase(Predicting). In the account creation phase, a username-password combination is registered and the facial features of the user are captured for Eigenfaces training. Since Eigenfaces is an image-based face recognition algorithm, the images used are obtained as a sequence of frames captured at regular intervals from a video device. These images are converted to grayscale, equalized and the detected face-image in the frame is cropped out and resized to fixed dimensions. The parameters learnt from Eigenfaces training over these images and the username-password combination are stored in an XML file.

The control flow in the continuous authentication phase can be viewed to exist in Hard or Soft biometrics mode. As seen in Fig. 3, the user begins a session by entering the right username-password combination. This initiates the Hard Biometrics phase where T frames are used to construct a feature vector X. The Support Vector Machine represented as $y = g(w^T x + b)$ predicts $y \in \{-1, +1\}$. These series of predictions are represented in the form of a bit-vector $\langle b_0 b_1 b_2 \ldots b_N \rangle$. Confidence of the system in hard biometrics mode θ_H is formulated as:

$$\theta_H = \frac{No.\ of\ bits\ enabled}{Length\ of\ bit - vector} \tag{28}$$

$$= \frac{\sum_{i=0}^{N} 1\{b_i = 1\}}{N} \tag{29}$$

When this hard-biometrics mode confidence θ_H exceeds a given threshold τ_H, the control flow transitions to the soft biometrics mode.

The Soft biometrics mode as seen in Fig. 4 begins by creating an enrollment template ξ based on N frames. This template contains the Hue-Saturation-Values (HSVs) of color of the user's clothing. Every subsequent N frames are then used to create an average template μ. The templates ξ and μ are represented as vectors

$$\xi = [c_{\xi,0} \quad c_{\xi,1} \quad \ldots \quad c_{\xi,C}] \tag{30}$$

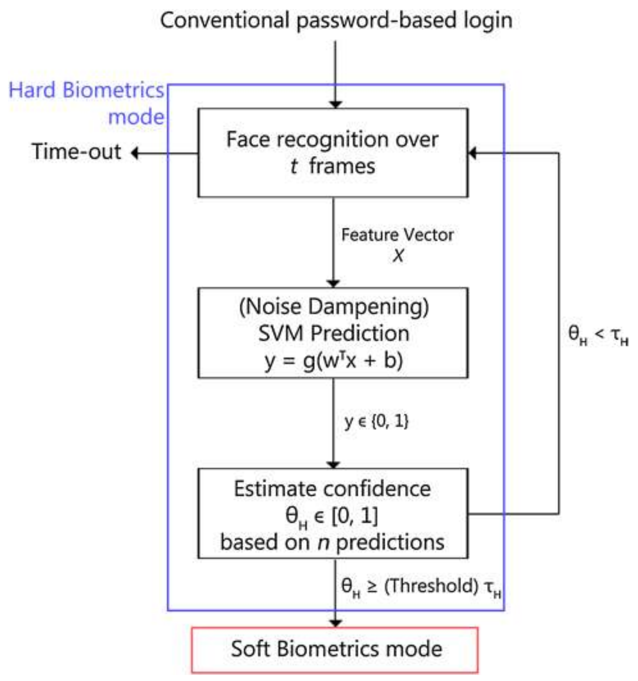
Conventional password-based login
Hard Biometrics mode
Time-out
Face recognition over t frames
Feature Vector X
(Noise Dampening) SVM Prediction y = g(wᵀx + b)
y ∈ {0, 1}
Estimate confidence θH ∈ [0, 1] based on n predictions
θH < τH
θH ≥ (Threshold) τH
Soft Biometrics mode

Fig. 3 Control flow in Hard Biometrics mode

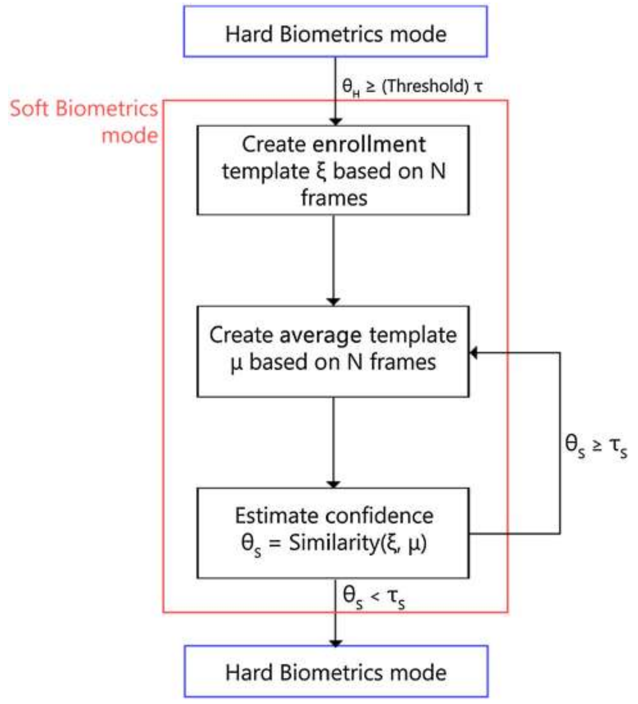
Hard Biometrics mode
θH ≥ (Threshold) τ
Soft Biometrics mode
Create enrollment template ξ based on N frames
Create average template μ based on N frames
θS ≥ τS
Estimate confidence θS = Similarity(ξ, μ)
θS < τS
Hard Biometrics mode

Fig. 4 Control flow in soft biometrics mode

$$\mu = [c_{\mu,0} \quad c_{\mu,1} \quad \ldots \quad c_{\mu,C}] \tag{31}$$

where $c_{i,j}$ represents the HSV for color j in template i. The confidence of the Soft Biometrics mode θ_S is calculated

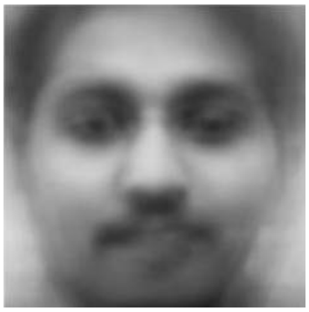

Fig. 5 Average image generated by Eigenfaces

based on the similarity between ξ and μ. The similarity is formulated as the normalized root mean square difference between ξ and μ as shown in Eq. 32.

$$\theta_S = \sqrt{\frac{\sum_{i=1}^{C}(c_{\xi,i} - c_{\mu,i})^2}{n}} \tag{32}$$

When this confidence θ_S falls below the specified threshold τ_S, the control moves back to Hard biometrics mode where the decision to log-out is made.

5 Results and discussion

In this section, we study the performance and discuss the design choices made for each module described previously in the Sect. 4.1. For the rest of the section, the face detection and face recognition algorithm implies Viola-Jones' method and Eigenfaces respectively.

5.1 Hard biometrics

In the Eigenface training phase, for the dataset to be centered during PCA, an average image as shown in Fig. 5 is computed by calculating the mean of the pixels of all the images in the training data set. The faces are then represented as a composition of the average face and a weighted average of the eigenface features as seen in 6. For example, a person might be characterized as the average image plus 20% from eigenface 1, 12% from eigenface 2 and so on.

Figure 5 and 6 were generated using 250 images, with 10 users contributing 25 images each. It can be seen that the average image shows a smooth face structure, the first few eigenfaces shows some of the dominant traits and later on mostly noise is captured and hence the contributions from these are negligible.

Figure 7 was obtained by running the continuous authentication system in hard biometrics mode under normal conditions, where the user is present in front of the

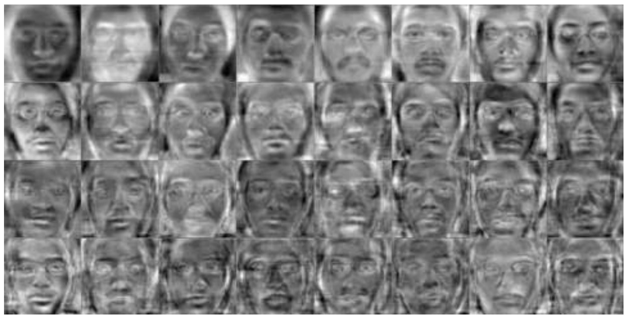
Fig. 6 The first few dominant Eigenfaces

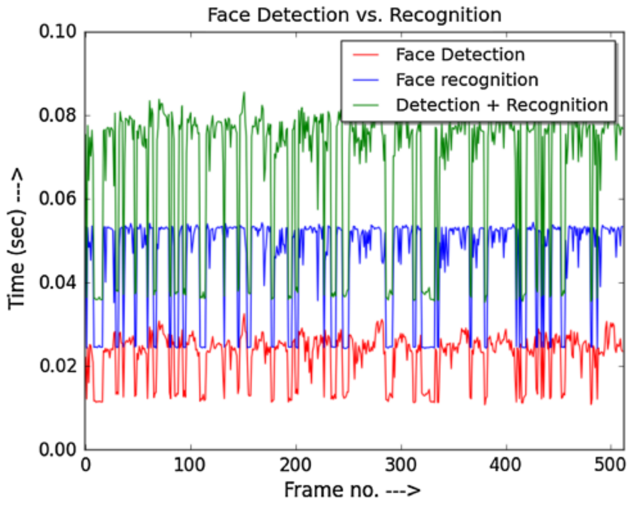

Fig. 7 Comparison of the facial features processing tasks

system. It shows the graphs for time taken to detect a face, recognize a face and to perform both over time (represented as frame numbers). The average time taken to process each frame for the task of face detection on an entire frame and face recognition for a face image are shown in Table 1.

In practice, all operations (including face recognition using Eigenfaces) on the image are performed by retrieving the face resized to a fixed height and width.

As mentioned earlier, the accuracy of Eigenfaces is affected by contrasting changes in posture and poor illumination. This can be seen in Fig. 8 where the y-axis represents the function $1\{user_{recognized} = user_{authorized}\}$ and UID refers to the User-ID of the respective users in the database. In the experiment, the user frequently changed the position and orientation of the face, causing a lot of noise to appear in the recognition. A similar result was observed when running the face recognition module in poor lighting conditions.

Table 1 Facial features processing time

	Average time taken
Face detection	0.0224
Face recognition	0.0454

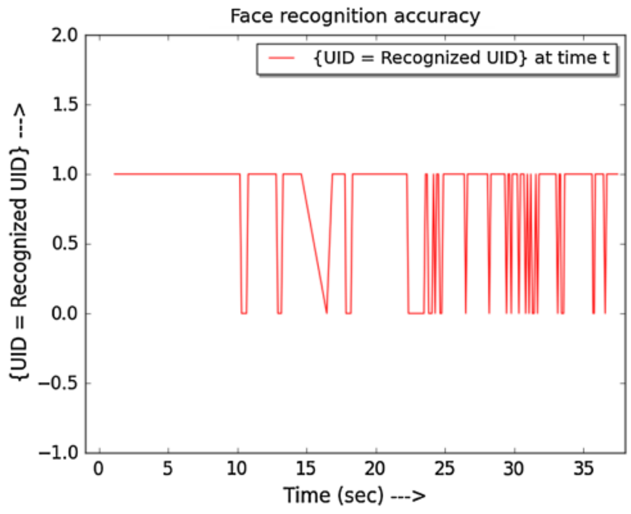

Fig. 8 Accuracy of face recognition achieved using Eigenfaces

5.2 Soft biometrics

Figure 9 empirically justifies our reason to transition into soft biometrics mode whenever possible. We observed that after retrieving the face image of the person in front of the system, the soft biometrics consumes half as much time as face recognition. Note that the time represented for both the observations in Fig. 9 includes the time taken to preprocess the frame and retrieve the face image. There are instances when face recognition takes relatively significant time to predict a face, such as in frame 155 in Fig. 9. This occured because the detected face was a false positive. While the soft biometrics module only retrieves a noisy data point for such a frame, face recognition additionally incurs an overhead cost in such a case.

The average time consumed per frame as observed can be seen in Table 2. As shown, an accuracy of 80% was achieved during the Soft biometrics mode, which is higher than that achieved in the Hard biometrics mode. This is because, in the Soft biometric mode, only the color composition of the rectangle that captures the shirt, and the stored template are compared. This computation puts relatively less load on the processor as compared to face recognition.

In Fig. 10a, b, the confidence over time was observed in two cases - when the authenticated user was in front of the

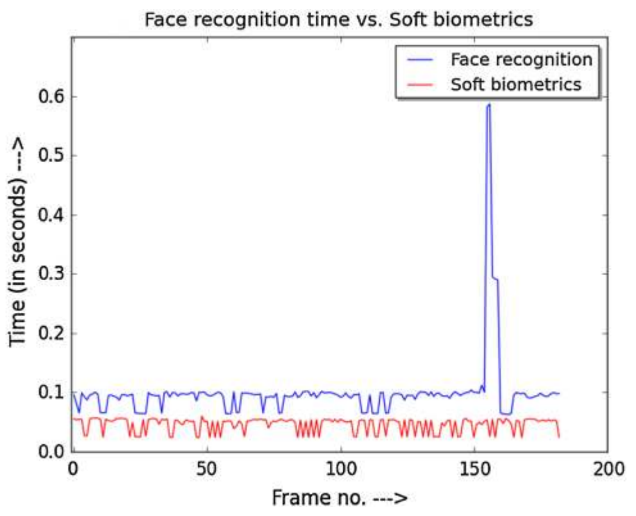

Fig. 9 Comparison of face recognition and Soft biometrics

Table 2 Face recognition vs. Soft biometrics

–	Average time	Accuracy
Face recognition	0.12	60–80%
Soft biometrics	0.045	80% ($\theta_S = 0.75$)

system and when he is tailgated by an imposter. By increasing T, the comparison of the template at time t with the enrolled template, can be smoothened out. This can be seen in both the figures where the peaks are dampened out as a result of increasing T.

Our solution overcomes this by extracting temporal information and the confidence estimated by the face recognition algorithm. Various situations were modeled and the data extracted was used in training to develop a classifier. The output produced by this classifier is then represented as confidence, which can be seen in Fig. 11b.

5.3 Noise dampening using SVM

Figure 11a captures the output of indicator function as described earlier. As seen from this figure, the immense noise in the data cannot solely be the basis for predicting if the user in front of the system is the authenticated user. This noise exists as a result of false positives and false negatives from the prediction.

Our solution overcomes this by extracting temporal information and the confidence estimated by the face recognition algorithm. Various situations were modeled and the data extracted was used in training to develop a classifier. The output produced by this classifier is then represented as confidence, which can be seen in Fig. 11b.

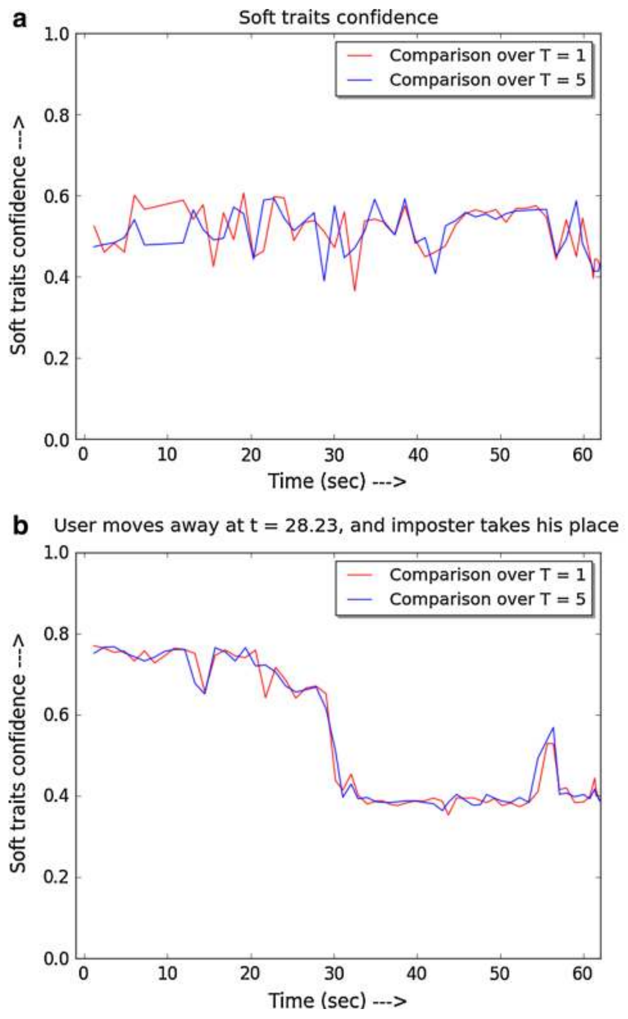

Fig. 10 Soft biometrics authentication **a** authenticated user is in front of the system **b** authenticated user is tailgated

In this work, the temporal information is extracted based on a fixed number of previous frames rather than a time period of fixed length. This is not only because the number of frames processed varies among these time periods, as seen in Fig. 12, but also because it is dependent on the system's resources and other conditions.

5.4 Continuous authentication mode

In this subsection, we briefly look into how the control flow jumps between the hard and soft biometrics modules. Figure was plotted by alternating the authenticated user and an imposter in front of the system. This resulted in a transition from hard to soft biometrics when the system was confident, and the other way round when the system needed to re-enforce its belief. Figure models the real-world condition wherein the authenticated user, after log-in, takes a break and moves away from the system and an

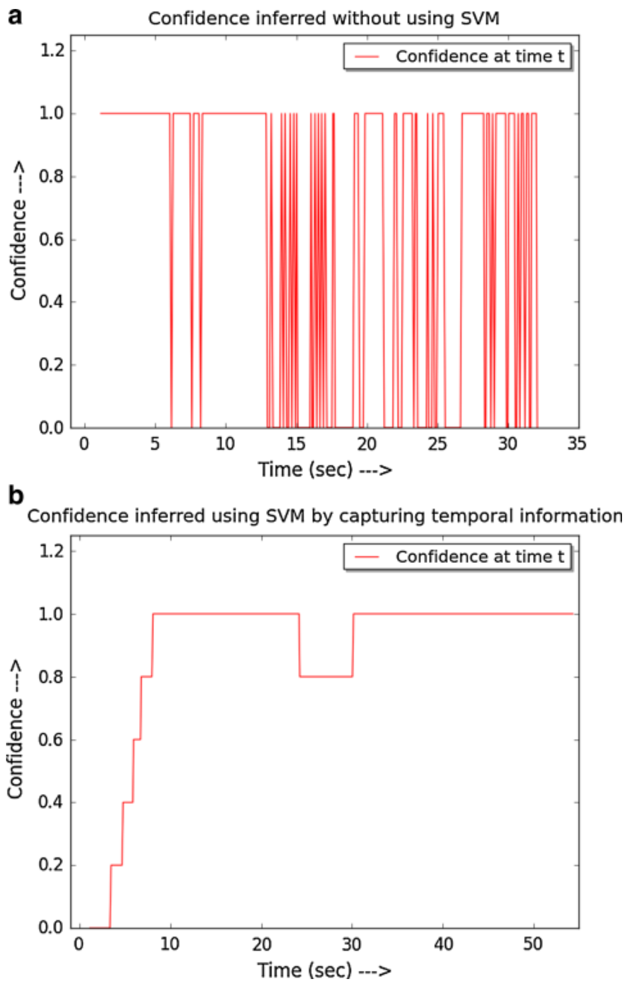

Fig. 11 Comparison of system's output with and without SVM **a** output received from the Face recognition module **b** confidence estimated by SVM using Face recognition data

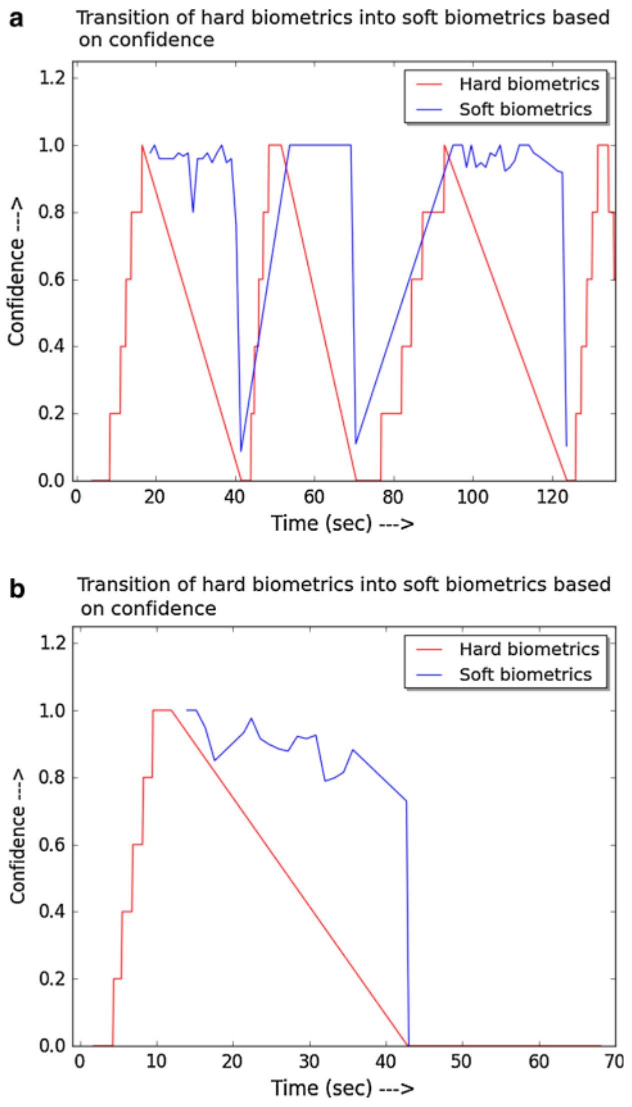

Fig. 13 Final confidence predicted by the complete system

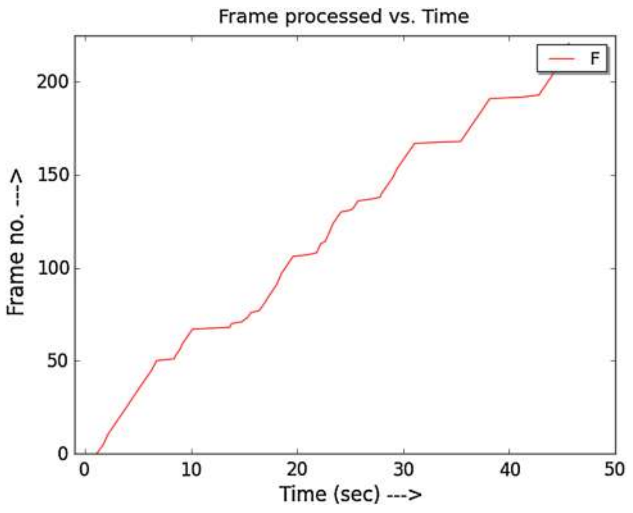

Fig. 12 Frames processed with respect to time

imposter takes his place. Thus, the soft biometrics mode confidence drops which activates the hard biometrics module. The predictions made, at this stage, prove that the authenticated user has been tailgated (Fig. 13).

Another notable result as published in [15, 16] is the performance of the Viola-Jones face detection and Eigenfaces face recognition algorithms when ported on to a GPU. As per [15], a CPU-GPU cooperative implementation of the Viola-Jones face detection algorithm on GTX280 graphics card achieved speed-ups of over 20× as compared to its implementation on the Intel Core 2 Duo CPU alone. As for Eigenfaces, it was shown by [16], that a highly parallelised implementation of Eigenfaces achieved highest speedups on GeForce GTX 480 for database of 15,000 images. A speedup of 207× was achieved for extraction of feature vectors in training process while the same for the recognition pipeline was shown to be 330×. Overall testing process yielded a speedup of 165× (testing over 40 images). The OpenCV implementations of both these algorithms can be modified to run on a GPGPU architecture to achieve a boost in their performance, thereby improving the performance of the proposed Multimodal Continuous Authentication System, if the parallelized implementations are considered.

6 Conclusion

In order to continuously authenticate the user, we presented an approach of alternating between two modes - Hard biometrics and Soft biometrics. The control flow transitions between these modes depending on the need to re-enforce the system's belief. Efforts made to increase the accuracy of face recognition algorithms fail to take into account the need to continuously recognize the user throughout a session. The result of this was seen in Sect. 5, where the predictions made become noisy due to variations in postures, since the user is unaware of the recognition process in the background. By learning from temporal data using Support Vector Machines, we attenuate this noise and make confident predictions over a stretch of time. The framework used also provides flexibility in replacing as well as extending the current setup to include other modules by transitioning based on only confidence of each mode.

Our future work in this field involves enhancements such as:

- Taking advantage of the enormous speed-ups when face detection and recognition are processed on the GPU.
- The training of the user's face model can be improved by introducing Online Training where each time the user is authenticated with a high level of confidence, a few face images are captured and the face model is retrained.
- The whole system can be implemented on a distributed architecture, with the face database in a central repository and the users logging in via different nodes associated with this central repository. The Client - Server architecture may be implemented for this.
- The Soft Biometric Traits may be expanded to include more features such as the complexion of the user, eye color or other facial features like facial hair.
- Support for multiple users sharing a certain account; this may require a biometric hand-off [4] to occur between users.
- Improve accuracy of face recognition under varied lighting conditions by implementing recognition using a different algorithm or approach since face recognition has been implemented as a separate module.
- Make provision for recognizing any kind of tampering occurring to the video feed, so as to prevent authenticating imposters. This can be done by restricting access to the webcam feed via parameters that define access to it.

References

1. Federal Financial Institutions Examination Council Authentication in an Internet Banking Environment, 2008.
2. Niinuma K, Unsang Park, Fujitsu Labs Ltd., Kawasaki Japani, AK Jain (2010) Soft biometric traits for continuous user authentication. Inf Forensics Secur IEEE Trans 5(4): 771–780
3. Jain AK, Dass SC, Nandakumar K (2004) Can soft biometric traits assist user recognition? Proc SPIE 5404:561–572
4. Klosterman AJ, Ganger GR (2000) Secure continuous biometric-enhanced authentication. Carnegie Mellon University, Tech. Rep. CMU-CS-00-134
5. Monrose F, Rubin AD (2000) Keystroke dynamics as biometrics for authentication. Futur Gener Comput Syst 16:351–359
6. Guennoun M, Abbad N, Talom J, Rahman SMM, El-Khatib K (2009) Continuous authentication by electrocardiogram data. Sci Technol Humanit (TIC-STH) IEEE Tor Int Conf
7. Turk M, Pentland A (1987) Face recognition using eigenfaces. In: Proceedings IEEE Conference on Computer Vision and Pattern Recognition. pp. 586–591
8. Viola P, Jones M (July 13, 2001) Robust real-time object detection. Second international Workshop on statistical and computational theories of vision—modeling, learning, computing, and sampling (Vancouver, Canada)
9. How face detection works, http://www.servomagazine.com/ SERVO Magazine, 2007
10. http://www.pages.drexel.edu/sis26/Eigenface%20Tutorial.htm. Eigen faces tutorial. Accessed 4 Feb 2013
11. http://www.see.stanford.edu/. Stanford SEE Machine Learning class. Accessed 4 Feb 2013
12. http://www.opencv.willowgarage.com/OpenCV. Accessed 4 Feb 2013
13. Cortes C, Vapnik VN (1995) Support-vector networks. Mach Learnin 20
14. Chang CC, Lin CJ (2011) LIBSVM: A library for support vector machines. ACM Trans Intell Syst Technol available at http://www.csie.ntu.edu.tw/~cjlin/libsvm
15. Kong J, Deng Y (2010) GPU accelerated face detection. Int Conf Intell Control Inf Process (ICICIP)
16. Ashraf N CUDA accelerated face recognition. NeST–NVIDIA Center for GPU Computing NeST, India