PRIVACY PRESERVING EEG-BASED AUTHENTICATION USING PERCEPTUAL HASHING

Samir Dilip Koppikar

Thesis Prepared for the Degree of

MASTER OF SCIENCE

UNIVERSITY OF NORTH TEXAS

December 2016

APPROVED:

Hassan Takabi, Major Professor
Cornelia Caragea, Committee Member
Xiaohui Yuan, Committee Member
Barrett Bryant, Chair of the Department of Computer Science and Engineering
Coastas Tsatsoulis, Dean of the College of Engineering
Victor Prybutok, Vice Provost of the Toulouse Graduate School

Koppikar, Samir Dilip. *Privacy Preserving EEG-Based Authentication using Perceptual Hashing*. Master of Science (Computer Science), December 2016, 61 pp., 20 tables, 10 figures, bibliography, 65 titles.

The use of electroencephalogram (EEG), an electrophysiological monitoring method for recording the brain activity, for authentication has attracted the interest of researchers for over a decade. In addition to exhibiting qualities of biometric-based authentication, they are revocable, impossible to mimic, and resistant to coercion attacks. However, EEG signals carry a wealth of information about an individual and can reveal private information about the user. This brings significant privacy issues to EEG-based authentication systems as they have access to raw EEG signals. This thesis proposes a privacy-preserving EEG-based authentication system that preserves the privacy of the user by not revealing the raw EEG signals while allowing the system to authenticate the user accurately. In that, perceptual hashing is utilized and instead of raw EEG signals, their perceptually hashed values are used in the authentication process. In addition to describing the authentication process, algorithms to compute the perceptual hash are developed based on two feature extraction techniques. Experimental results show that an authentication system using perceptual hashing can achieve performance comparable to a system that has access to raw EEG signals if enough EEG channels are used in the process. This thesis also presents a security analysis to show that perceptual hashing can prevent information leakage.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1. Motivation and Objective

Authentication is the process of verifying whether the claim made by an entity about his or her identity is true or false. Authentication is necessary in developing access controls, which either grant or deny an entity, access to a system based on whether the verification has succeeded or failed. Examples could be accessing your laptop, mobile or email or connecting to a private network of an organization. Users have an identity and an associated password which the user can use to access a system. Typically, such passwords are text-based and can be easily forgotten or stolen. An alternative to text-based passwords is the use of biometrics, including, but not limited to, fingerprints, iris, speech, face and gait. Another biometric that has attracted the interest of researchers in the area of biometric authentication is electroencephalogram (EEG) which records the electrical activity of the brain and also exhibits biometric qualities. Additionally, clinicians have seen patterns and characteristics in these recordings that can act as identifiers and are unique to patients [48].

In addition to uniqueness, using EEG as a biometric has several advantages which make it a good candidate for authentication:

(1) As it results from some mental activity, it is confidential.

(2) Even for similar mental tasks done by different people they are different. Thus, they are hard to mimic [63].

(3) Stealing them through coercion is also difficult as that could affect the resulting EEG recordings [21] [41].

(4) Non-invasive EEG based Brain Computer Interfaces (BCI) that can be used to monitor the brain activity are inexpensive and available off-the-shelf.

Different works, over the past decade, have shown that EEGs can be used for authentication [51, 48, 60, 34, 14, 40, 28, 65]. However, they are prone to serious privacy issues. Primarily, the EEG signal of a user is private and can reveal information about him or her. Researchers found

that the EEG signals captured using consumer-grade EEG-based BCI devices can reveal private information such as bank cards, PIN numbers, the area of living and the user's knowledge of the known persons [35]. Furthermore, researchers were able to perform a more covert attack called a subliminal attack [22]. In that, the duration of the visual stimulus is so short that the user is unable to perceive the stimulus consciously. However, the resulting EEG signals can be analyzed by an attacker to learn private information about him or her. Moreover, with some feature extraction techniques like Fast Fourier Transform [19] and Wavelet Transform [61], one can reconstruct the original signal from the extracted features [1]. Thus, sharing the raw EEG signal or the feature vector can also result in leakage of private information. Additionally, in any biometric authentication system, there are two phases: Enrollment and Authentication. In the enrollment phase, the user enrolls his or her biometric. We extract a template from this biometric that is unique to the user, store it in the database and discard the biometric. During the authentication phase, the user again presents his or her biometric. Again, a template is extracted and matched against the template in the database. Authentication succeeds if the match is successful and it fails otherwise. Unlike in the case of a text-based password, where one can easily replace a stolen password, a stolen biometric cannot be revoked. Thus, if an attacker can compromise the database that stores the biometric template the users may permanently lose their biometric identity.

A popular way to protect text-based passwords is to use hash functions. A hash function is a function that maps an input of size m (the data that you are trying to protect) to an output of size n (the hash). The field of cryptography widely utilizes hash functions [55] and they are often used to check the integrity of the original data. These are also known to be one-way, i.e., going back to original data from the hash is computationally hard. However, these functions are very sensitive to the input, i.e., even a small change in the input (a single bit flip) results in a significant change in the output. Hence, using a cryptographic hash function for our approach is unsuitable as two EEG signals of the same person will not be identical. As an alternate method to this, we propose the use of content based hashing or perceptual hashing, i.e., to compute the hash based on the data or content, which in our case is the user's EEG signal [27]. So as long as two inputs have similar content, they generate similar hashes. Although two EEG signals of the same person

are not identical, we hypothesize that they would have similar content and would generate similar perceptual hashes. Hence, we use these perceptual hashes to perform authentication. However, this alone does not solve all the privacy issues. A compromised perceptual hash will not reveal private information about the user, but it can still result in a loss of the user's biometric identity. One way of solving this problem is to create a revocable biometric identifier (BIDs) [18] which is a repeatable binary sequence derived from the user's biometric. Similarly, we can also use a biometric key (BK) which is a cryptographic key derived from the biometric. A BID or a BK should possess the following essential characteristics:

(1) It should not leak any information about the original biometric.

(2) For a biometric captured from the same user, one must be able to generate the same BID, i.e. the BID should be repeatable. This is imperative in our case as EEG signals of the same person captured at different times will not be identical.

(3) It should not be same for two different people.

In this work, we aim to develop a privacy-preserving authentication system using EEG and perceptual hashing while addressing the above challenges. We adopt the approach proposed in [24] and develop our proposed authentication system based on perceptual hashing with this framework.The contributions of our work are listed as below:

(1) To the best of our knowledge, this is the first work on privacy preserving EEG-based authentication.

(2) To the best of our knowledge, this is the first work that explores perceptual hashing of EEGs to perform authentication and preserve the user's privacy.

(3) We compute the perceptual hash using two feature extraction techniques and explore two methods for generating the hash.

(4) We use support vector machine (SVM) and k-Nearest Neighbor (k-NN) classifiers to perform authentication. Additionally. we perform extensive tests to find the best hyper-parameters to train the SVM classifier and evaluate our approach on three data sets.

(5) We compare the performance of the classifiers and also study the impact of perceptual hashing on authentication, classifier training time, protocol execution time, and bandwidth

consumption.

1.2. Thesis Organization

The organization of this thesis is as follows: In Chapter 2, we discuss in detail about EEG, and then talk about existing work in the area of EEG-based authentication. In Chapter 3, we first give an overview of the proposed approach and then discuss its components in detail. Later, in that chapter we describe in detail protocols for enrollment and authentication in two types of authentication models: IDP-centric and User-centric. In Chapter 4, we discuss the datasets, experimental design and our results. In Chapter 5, we perform a security analysis of our work and finally, we conclude our work and discuss future directions in Chapter 6.

CHAPTER 2

BACKGROUND AND RELATED WORK

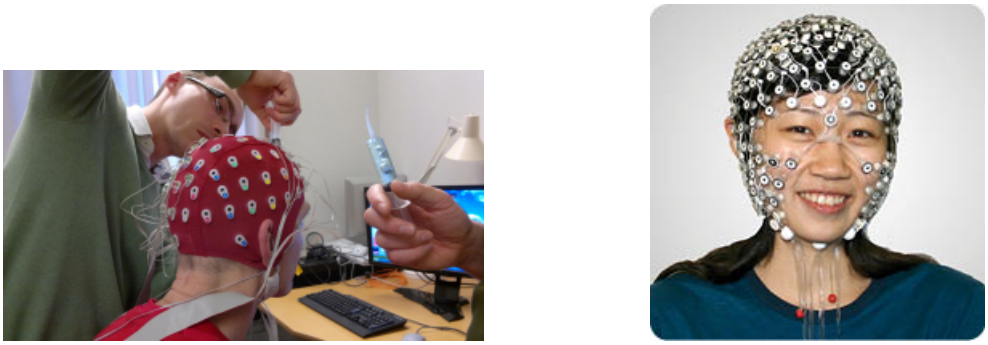
2.1. Electroencephalograms (EEG)

EEG signals record the electrical activity of the brain, and we can divide them into different bands based on frequency as: (1) Delta (δ) range $< 4Hz$, (2) Theta (θ) range $4 - 7Hz$, (3) Alpha (α) range $8 - 15Hz$, (4) Beta (β) range $16 - 31Hz$, (5) Gamma (γ) range $\geq 32Hz$.

(a) NeuroSky MindWave Mobile single-channel headset [11]

(b) EMOTIV EPOC+ 14-channel headset [7]

FIGURE 2.1. Consumer grade BCI devices



(a) Full 64 electrode setup [8]

(b) Geodesic sensor net [9]

FIGURE 2.2. BCI devices with more EEG electrodes

Typically, these are recorded using non-invasive EEG-based brain computer interface (BCI) devices having dry or wet electrodes placed on the scalp. Also, EEG signals have small signal amplitudes (μV). BCI devices with a different number of EEG channels are available and can

have a single EEG channel to 256 EEG channels. An EEG channel is an electrode that captures brain signals. Today, many inexpensive and off-the-shelf BCI devices are available. Figure 2.1 shows NeuroSky [10] MindWave Mobile headset, which has a single EEG channel and EMOTIV [6] EPOC+ headset, with 14 EEG channels. Similarly, Figure 2.2 shows BCI devices with 64 and 256 electrodes. Figure 2.3, shows the 64 electrodes map as per the international 10-10 system [5]. In our work, we make use of Visually Evoked Potentials (VEP) signals to perform authentication. These are a particular type of EEG signals that are evoked by some visual stimulus.

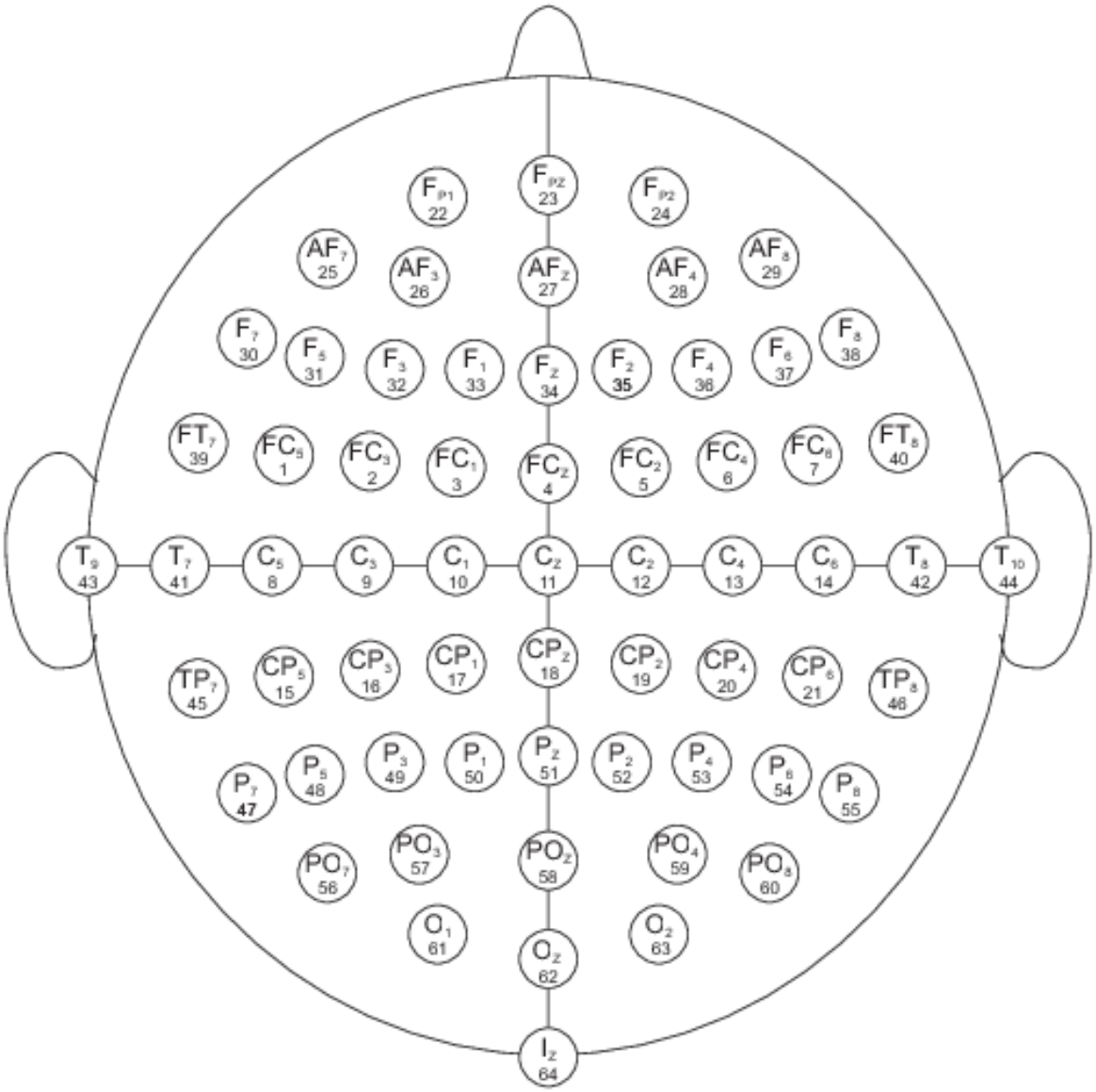


FIGURE 2.3. 64 electrodes map as per the international 10-10 system

2.2. Related Work

Some of the early works that explore EEG as a biometric include [51], [48] and [45]. Poulos et al. [51] using Autoregressive (AR) models, estimate AR parameters from the α band in the EEG signals. They use these estimates as features and perform classification using Linear Vector Quantizer network with 72-80% success. Paranjape et al. [48] work with a single EEG channel. They first examine the EEG signals using AR models and then apply Discriminant Analysis to the AR coefficients. They were able to achieve a success of 80%. Palaniappan et al. [45] study EEG signals (VEP signals) from 61 electrodes for 20 subjects with an average classification of approximately 94%. They extract the gamma band power features and employ Simplified Fuzzy ARTMAP neural networks to identify different individuals. These works were followed by many others in the following years [60], [47], [34], [40], [53], [13], [28], [62]. Ashby et al. [14] in their work propose a low-cost EEG-based authentication system using AR models for feature extraction and linear Support Vector Machine for classification. They make use of an inexpensive EEG Headset (EMOTIV EPOC) with 14 channels. Zúquete et al. [65] in their work use VEP signals to perform biometric authentication. They compute differential signals of 8 EEG channels and use the energy of these signals as biometric features. They perform authentication using k-NN, Support Vector Data Description (SVDD), and two combinations of k-NN and SVDD. However, all these works focus on improving the accuracy of the system. We aim to perform the authentication using EEG while preserving the privacy of the user and the biometric. We leverage the work of [65] on differentials signals and apply Perceptual Hashing to protect the privacy of the signals. Additionally, we incorporate a privacy preserving protocol to perform authentication.

Perceptual Hashing has been around for a while and can apply to any multimedia like image, audio or video. It is a content-based hashing technique and has found applications in areas like content identification and content retrieval. Hashing algorithms for audio content identification have been discussed in [25], [37], [44] and for database searching in [20], [30], [58]. It can also been used for image retrieval, image authentication and copy detection in [29], [32], [43]. Similarly, Perceptual Hashing algorithms for video identification have been discussed in [42] and [64]. In our approach, we compute a hash of the features extracted from the EEG signals. Gunasinghe et

al. [24] in their work develop a biometric authentication system using iris. They compute a perceptual hash of iris images using a type-II Discrete Cosine Transform (DCT) based hashing technique. After preprocessing and resizing the iris image, they compute the DCT coefficients. Then, they extract 64 low-frequency coefficients in the form of an 8x8 matrix (the first eight columns and first eight rows). They convert this matrix to a row vector by concatenating each row. Finally, they compute the hash by comparing each element to the median of the vector. The Perceptual Hashing technique that we use in our work is very similar to this but with two important differences: (1) We compute the hash based on the features extracted using Parseval's Spectral Power Ratio Theorem and Power Spectral Density instead of DCT; (2) We additionally support dividing the feature vector into segments for calculating the perceptual hash.

We need to create a BID or a BK from the EEG signal of the user that is repeatable and revocable. Palaniappan et al. [46] investigate a thought based system for generating a personal identification number. They use a single electrode from a P300 based BCI device for this purpose. Another important requirement is that the identifier should be repeatable and revocable. Ratha et al. [52] in their work present a system to generate cancelable biometrics. They vary the distortions on the biometric features to provide various versions of a biometric template. Recently, Bajwa et al. [15] explored the use of EEG for generating cancelable biometric keys. They extract feature vectors from the user's EEG signals captured while performing a chosen mental activity. These feature vectors are then binarized using the authentic regions of the user to generate the key. The authentic regions of the EEG features are established by the user for the chosen activity during the enrollment phase. To change the key, the user has to choose a different mental activity. In our work, we adopt the BID generation technique used in [24]. In that, an identity provider (IDP) generates a repeatable BID using the output of a customized SVM classifier and a secret derived from the user's password. The IDP hides the BID in a token which one can revoke by changing the password. However, they provide the biometric (iris) to the IDP. In our approach, we present only the perceptual hash of the EEG signals to the IDP.

CHAPTER 3

METHODS

3.1. Architecture Overview

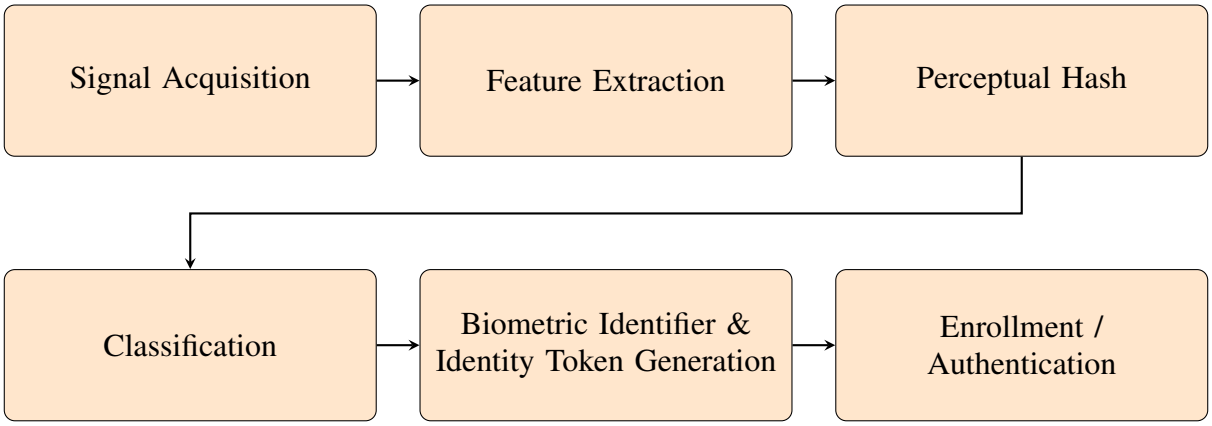


FIGURE 3.1. Architecture overview

Figure 3.1 shows various components in our approach, and we will discuss them in detail in the following sections. The architecture involves three entities or parties viz., the Identity Provider (IDP), the User (U) and the Server (S). The goal of the system is to authenticate U so that he or she can access the services provided by S. The IDP is a trusted party and is responsible for enrolling the user and generating a BID and identity token for the user. Depending upon the authentication model, the IDP may or may not be involved in the authentication process.

- Signal Acquisition: The first step in any EEG-based authentication system is acquiring the EEG signals and is done using BCI devices. Today, many commercial BCI devices like NeuroSky and Emotiv are available that can be used to acquire the signals. However, instead of collecting our data (EEG Signals), we make use of publicly available datasets (see Section 4.1). We then select appropriate channels and compose the required signal. We discuss these in Section 3.2 and Section 3.3 respectively.
- Feature Extraction: The next step in our approach is to extract features from the EEG signals. We employ two techniques for this purpose viz., Parseval's spectral power ratio theorem and Power Spectral Density. We discuss these techniques in Sections 3.4.1 and

3.4.2 respectively.

- Perceptual Hash: We next apply Perceptual Hashing to the extracted feature vectors. This step transforms the feature vectors to a sequence of 1s and 0s. We discuss this in detail in Section 3.5.
- Classification: Two EEG signals of even the same person are not identical. Hence, to identify people using the EEG signals, we employ machine learning techniques, SVM and k-NN, for this purpose. We discuss these techniques in Sections 3.6.2 and 3.6.1.
- Biometric Identifier and Identity Token Generation: The output of the classification step gives us a class label for the perceptual hash. We use this class label to generate a Biometric Identifier (BID) and discuss that process in Section 3.7.4. We generate an identity token for the user and hide the BID in it. We discuss the process of generating the token in Section 3.7.5
- Enrollment/Authentication: We adopt the proposed approach in [24] to implement the enrollment and authentication phases. The BID and Identity Token generation component is also a part of the adopted approach. During enrollment, we generate a token for the user who can use the token to sign-up with a server. Later, the user uses the token to complete authentication. There are two models in our approach: IDP-centric model and User-centric model. We discuss both enrollment and authentication phases in each model in Sections 3.8.1 and 3.8.2

3.2. EEG Channel Selection

In our approach, we make use of VEP signals, i.e., the EEG signals generated in response to some visual stimulus. Throughout our discussion in this thesis, we use these terms interchangeably. The gamma band (around 40 Hz) is related to attention and processing visual information [39]. Thus, we choose only the γ band in our approach and look at the 30-50 Hz frequency band. Another important step in using EEG signals for performing authentication is selecting the appropriate EEG channels, which by itself is an entirely different research topic. Hence, we leverage existing work to achieve this task. We primarily consider 8 EEG channels in the occipital region as they record stronger electrical activity in the γ band in response to visual stimuli [33] [59] [23].

In addition to these EEG channels, we also consider other channels from other regions for performing some experiments. Table 3.1 gives information about the various EEG channels we have considered in our approach.

#	EEG Channels
8	PO3, PO4, POz, PO7, PO8, O1, O2, Oz
14	AF3, F7, F3, FC5, T7, P7, O1, O2, P8, T8, FC6, F4, F8, AF4
17	PO3, PO4, POz, PO7, PO8, O1, O2, Oz, P1, P2, P3, P4, P5, P6, P7, P8, Pz
26	PO3, PO4, POz, PO7, PO8, O1, O2, Oz, CP1, CP2, CP3, CP4, CP5, CP6, CPz, TP7, TP8, P1, P2, P3, P4, P5, P6, P7, P8, Pz

TABLE 3.1. Selected EEG channels

3.3. Signal Composition

We consider multiple EEG channels to perform authentication and hence, represent them as a two-dimensional (2D) array where each channel is a column in the array. Let C denote all these channels. Note that we have an EEG Signal from each EEG channel. Additionally, we include differential signals as used in [65] and obtain them by subtracting corresponding elements in each pair of columns. Thus, total number of columns $C_T = C + \binom{C}{2}$. The number of rows in the array will depend on the sampling rate.

EEG Signals			*Differential Signals*			
$C1$	$C2$	$C3$	$C1 - C2$	$C1 - C3$	$C2 - C3$	
$C1_1$	$C2_1$	$C3_1$	$C1_1 - C2_1$	$C1_1 - C3_1$	$C2_1 - C3_1$	} *N samples*
$C1_2$	$C2_2$	$C3_2$	$C1_2 - C2_2$	$C1_2 - C3_2$	$C2_2 - C3_2$	
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	
$C1_N$	$C2_N$	$C3_N$	$C1_N - C2_N$	$C1_N - C3_N$	$C2_N - C3_N$	

TABLE 3.2. Signal Composition

The purpose of using differential signals is to provide to the classifiers information about the phase of EEG signals instead of just providing information about their amplitudes (energies). Phase shifts between subtracted EEG signals with equal frequency and amplitude generate non-null signals with the energy as a function of the phase shift. Thus, the energy of differential channels can denote phase shifts between the EEG channels and we can provide more information to the

classifiers about each subject [65]. Table 3.2 shows the signal composition for three EEG channels having N samples. As you can see from the table, for three EEG channels, there are $\binom{3}{2} = 3$ differential columns, i.e., a total of six columns and N rows. Based on our discussion in Section 3.2, the number of columns in the signal will vary upon the number of channels selected. Refer to Table 3.3 for more details.

EEG Signals (C)	Differential Signals ($\binom{C}{2}$)	Total Signals ($C_T = C + \binom{C}{2}$)
8	28	36
14	91	105
17	136	153
26	325	351

TABLE 3.3. Calculation of Total EEG Signals

3.4. Feature Extraction

We employ two techniques to extract features from the raw EEG signals viz., Parseval's spectral power ratio theorem and Power Spectral Density.

3.4.1. Parseval's Spectral Power Ratio Theorem (PSPRT)

This technique computes the energy of the signal using the following equation:

$$E(S) = \frac{1}{N} \sum_{n=1}^{N} s_n^2 \tag{1}$$

Here, $E(S)$ denotes the energy of signal S. The signal has N samples and s_n is the n^{th} sample. We apply Equation 1 to each column, i.e., all EEG signals and differential signals (see Section 3.3). This results in a feature array (feature vector) F with a single row. To normalize the feature array, we divide each element by the maximum element in the array.

$$F(i) = \frac{F(i)}{max(F)} \;\; i = 1, 2, \ldots, C_T$$

The total number of columns in the feature array after feature extraction is still C_T. Table 3.4 shows how many features PSPRT yields for the set of selected channels. You can refer to Table 3.3 to see how we calculate the total number of signals.

EEG Channels (C)	Total Signals (C_T)	Total Features
8	36	36
14	105	105
17	153	153
26	351	351

TABLE 3.4. Number of features yielded by PSPRT

3.4.2. Power Spectral Density (PSD)

We compute the PSD estimate $PSD(S)$ of the signal S (refer to Table 3.2) using $pwelch$ in MATLAB [36]:

$$PSD(S) = pwelch(S, window, noverlap, nfft) \tag{2}$$

$pwelch$ computes the PSD estimate of each EEG signal, i.e., column in S by dividing the signal into segments. Each segment has $window$ number of samples. Two consecutive segments will have $noverlap$ number of overlapping samples. $nfft$ gives the number of Discrete Fourier Transform points to use in the computing the PSD estimate. $PSD(S)$ has the same number of columns as in S. We then extract the γ band between 30-50 Hz and finally normalize each row of $PSD(S)$ using $normr$ in MATLAB. Since we extract only the γ band, the number of rows in the feature array reduces to 20. Then we transpose each column in the feature array into a row vector and concatenate all these row vectors. Finally, we have a feature array with a single row and $20 \times C_T$ columns. Table 3.5 shows how many features are present in the final feature vector for the set of selected channels using PSD as a feature extraction technique.

EEG Channels (C)	Total Signals (C_T)	Total Features ($20 \times C_T$)
8	36	720
14	105	2100
17	153	3060
26	351	7020

TABLE 3.5. Number of features yielded by PSD

3.5. Perceptual Hash

A perceptual hash of any multimedia file is a representation that is generated based on the content of that file [27]. In the context of any image, its gray-scale and color representation will have similar perceptual hashes because both of them have similar content. The same applies to audio and video files as long as they have similar content.

We first extract a feature vector from the raw EEG signal as discussed in Sections 3.4.1 and 3.4.2 and then, compute a perceptual hash of this feature vector. EEG signals of the same person, captured at different times, are not identical to each other. However, they should be similar or else using them for authentication will not be possible. Hence, we argue that they should generate similar perceptual hashes.

The raw EEG signal of an individual is private and sharing it as is can result in information leakage, and sometimes it is possible to reconstruct the original signal using the extracted features. Achieving this would be hard with our choice of features extraction techniques, but with techniques like Fast Fourier Transform and Discreet Wavelet Transform, it is feasible to reconstruct the original signal. Hence, we consider the feature vector to be private as well. Additionally, the perceptual hash is a sequence containing just 0s and 1s instead of having the actual feature values. Thus, by using the hash instead of the feature vector, we expect to protect the privacy of the biometric.

Algorithm 1 Compute Perceptual Hash

Input: F {Feature Vector}, $segmentSize$ {Size of each segment}
Output: PH {Perceptual Hash of the Feature Vector}

1: $segments \Leftarrow$ Divide F into segments of size $segmentSize$
   //Compute Perceptual Hash H for each segment
2: **for** each $segment$ in $segments$ **do**
3: $\quad median \Leftarrow$ Compute median of $segment$
4: $\quad$ **for** $i = 0$ **to** $segment.length$ **do**
5: $\quad\quad H(i) \Leftarrow \begin{cases} 1, & segment[i] \geq median \\ 0, & segment[i] < median \end{cases}$
6: $\quad$ **end for**
7: $\quad PH \Leftarrow PH \mid H$ for $segment$ {Append H for $segment$}
8: **end for**
9: **return** Perceptual Hash PH

Refer to Algorithm 1, which shows the steps taken to compute the Perceptual Hash for a

given feature vector. We first divide the feature vector into segments having $segmentSize$ number of features (Line 1). However, depending on the total number of features, the last segment could have fewer features. For each segment, we compute the $median$. We then compare the value of each feature in the segment with the $median$ to compute the perceptual hash (Line 2 - Line 6). Finally, we append the perceptual hash of each segment to get the hash of the feature vector (Line 7). We choose two values of $segmentSize$:

(1) 100% features i.e., consider the entire feature vector as a single segment.

(2) At least 50% features i.e., divide the feature vector into two segments with the first segment having at least 50% of the features.

3.6. Classification

To perform authentication, we need to predict the class label (subject) based on the Perceptual Hash. For this purpose, we employ two classifiers viz., k-Nearest Neighbor (k-NN) and Support Vector Machine, which we discuss next.

3.6.1. k-Nearest Neighbor

k-Nearest Neighbor (k-NN) classification is a technique that looks at existing data records with known class labels to predict the class label of a new data record with an unknown class label. The k-NN classifier computes the distance between the new data record and all existing data records to find out k nearest neighbors or closest matching data records. It then decides the class label of the new record based on a majority vote. The value of k is usually chosen to be an odd number to avoid ties. We use the IBk classifier provided by Weka [26] in our approach.

3.6.2. Support Vector Machine

For any given training set in which records belong to one of two categories, a Support Vector Machine (SVM) classification algorithm builds a model that classifies new data records with an unknown class label as one of those two categories. An SVM classifier does this by constructing an optimal hyperplane in a high-dimensional space. Since data records in the training set may not be linearly separable in their original finite-dimensional space, the SVM classifier maps the original space to a higher dimensional space where linearly separating these data records

could be easier. The mapping from the original dimensional space to a higher dimensional space is done using kernel functions. In our approach, we use Poly Kernel and Radial Basis Function (RBF) Kernel provided by Weka [26].

3.7. Components in Enrollment/Authentication

The approach that we adopt from [24] has the following components:

3.7.1. Pedersen Commitment

We make use of the Pedersen Commitment scheme [49] to hide the BID of a user in the identity token, and this token is used by the user to complete the authentication process. The fact that solving discrete logarithms is a hard problem makes this commitment scheme secure. We adopt this technique from [24]. In such a scheme, there are two entities: Committer and Verifier, and the following steps:

(1) Let p and q be two large prime numbers such that q divides $p-1$. We choose p as a 1024 bit integer and q as 160 bit integer. Z_p^* is the multiplicative group of order p and G_q is a unique subgroup of Z_p^* of order q. Let g denote the generator of G_q. During system initialization, a trusted party chooses this generator. It also chooses h such that $log_g h$ is a secret. This can be done by choosing a secret x and computing $h = g^x \bmod p$. Note that both g and h are elements of G_q. Moreover, the parameters $< p, q, g, h >$ are publicly available.

(2) A Committer who wants to commit to a secret value $s \in Z_q$ chooses another random secret $r \in Z_q$ and computes the commitment $C(s, r)$ as follows:

$$C(s, r) = g^s h^r \bmod p$$

(3) Finally, and if required, the Committer can reveal s and r. Using these values, the Verifier can open the commitment to verify if it is authentic.

It is important to note that $C(s, r)$ does not reveal any information about s and that unless one can find $log_g h$, the commitment cannot be opened with any $s' \neq s$ [49]. In our case, BID of

the user is one of the secrets which we need to hide. We achieve this by using these properties of this commitment scheme.

3.7.2. Zero Knowledge Proof of Knowledge Protocol

A Zero Knowledge Proof of Knowledge (ZKPK) Protocol is a protocol that lets one entity (the Prover), who has some secret, prove to another entity (the Verifier) he or she has knowledge of the secret. However, while doing so, the Verifier should not gain any information about the Prover's secret. We adopt the approach for implementing the ZKPK protocol from [24]. We saw in Section 3.7.1 how we can hide the secrets s and r in a commitment. In our approach, we use Protocol 1 that lets the user prove he or she is the owner of the identity token without actually revealing the values of secrets hidden in it. The protocol has the following three important properties:

(1) Completeness: The protocol succeeds with a very high probability if the Committer and the Verifier are honest.

(2) Soundness: The protocol prevents the Committer from proving a false statement.

(3) Zero-Knowledge: The proof does not leak any information about the secrets hidden in the commitment.

Let P denote the user and V denote the Verifier.

Protocol 1 Zero Knowledge Proof of Knowledge Protocol

1: $P \Rightarrow V : P$ randomly chooses $y, t \in Z_q$, computes $d = g^y h^t$ and sends it to V
2: $P \Leftarrow V : V$ randomly chooses $e \in Z_q$ and sends it to P
3: $P \Rightarrow V : P$ computes $u = y + es$ and $v = t + er$ and sends them to V
4: $V : V$ accepts if $g^u h^v = dC^e$

3.7.3. Password-based Key Derivation

We hide two secrets s and r in the Pedersen Commitment. However, even in practical scenarios, users find it difficult to remember a single password and to expect the users to remember two passwords can affect the usability of the system. Hence, we derive the required secrets from a single password as done in [24]. It is worthwhile to mention that generating the necessary secrets also addresses security issues related to storing the secrets and secrets not being uniformly

distributed in the keyspace.

$$secret = PBKDF2(PKCS\#5, Password, Salt, keyLength)$$

$PBKDF2$ is password-based key derivation function 2 that we use to derive the required secrets. It uses $PKCS\#5$ as the pseudo-random function. $Password$ is the user-chosen password and $Salt$ is the salt value. $keyLength$ controls the length of the $secret$ generated. In our case, it is 288 bits. We extract the first 128 bits from this secret as $secret_1$, and the remaining 160 bits is the second secret $secret_2$. We append $secret_1$ to the classification output, a 32-bit integer, to form the 160-bit as secret s. This secret is the BID of the user. $secret_2$, which already has 160 bits, as the second required secret r. Thus, $s, r \in Z_q$ and can be used in the Pedersen commitment.

3.7.4. BID Generation

Algorithm 2 shows the steps involved in generating the BID, which we adopt from [24]. We first predict the class label $class$, a 32-bit integer, for the perceptual hash provided by the user (Line 1). Then, using the password chosen by the user, we derive two secrets (Section 3.7.3). Finally, the BID is generated by appending the secret $secret_1$ to $class$ (Line 2). $secret_1$ is a 128-bit integer and hence, the resulting BID is a 160-bit integer. Thus, the BID $\in Z_q$ as discussed in Section 3.7.1 and is used to generate the identity token.

Algorithm 2 Generate BID

Input: PH {Perceptual Hash of the Feature Vector}, $secret_1$ {128-bit secret derived from user's password. Refer to Section 3.7.3}
Output: BID {generated BID}
1: $class \Leftarrow$ Predict the class label for the given PH {$class$ is a 32-bit integer}
2: $BID \Leftarrow class + secret_1$ {Append $secret_1$ to $class$}
3: **return** BID

3.7.5. Identity Token Generation

Algorithm 3 shows the steps involved in generating the identity token $IDToken$ for the user. We adopt it from [24]. It takes BID, $secret_2$, $expires$, $from$ and to parameters as input. The BID is the user's identity and is generated using algorithm 2. $secret_2$ is the 160-bit secret

we derive from the user's password (see Section 3.7.3). $expires$ indicates when the identity token expires. The $from$ and to fields indicate the sender and the receiver respectively.

Algorithm 3 Generate Identity Token

Input: BID {User's BID}, $secret_2$ {160-bit secret derived from user's password. Refer to Section 3.7.3}, $expires$ {Indicating when the token should expire}, $from$, to
Output: $IDToken$ {generated Identity Token}
1: $s \Leftarrow BID$
2: $r \Leftarrow secret_2$
3: $IDToken \Leftarrow$ Generate a blank token
//Set the fields in the token
4: $IDToken.commitment \Leftarrow g^s h^r \bmod p$ {Compute and set the commitment in the token}
5: $IDToken.expires \Leftarrow expires$ {Set when the token expires}
6: $IDToken.from \Leftarrow from$
7: $IDToken.to \Leftarrow to$
8: IDP digitally signs $IDToken$
9: **return** $IDToken$

First, we create the Pedersen Commitment on $BID, secret_2 \in Z_q$ (Section 3.7.1) using the publicly available parameters $< p, q, g, h >$ and set in the token (Line 4). Then the remaining fields in the token are set (Line 5 - Line 7). Finally, the IDP signs the token (Line 8). The $from$ and to fields have been added to the identity token to prevent Mafia fraud attacks on the ZKPK identity verification protocols [24]. Please note that the $from$ field in the token can be a random value or a pseudonym of the user and prevents an adversary from identifying the user. The to field can also be misused by a malicious Server to impersonate the user. To avoid this, we can create a commitment on the Server's name and set this commitment in the to field. This method is secure as only the user has the secrets required to open the commitment.

3.8. Authentication Models

There are two types of biometrics-based identity management architectures viz., IDP-centric and User-centric. In the IDP-centric model, the user requires the IDP each time it wants to authenticate to a server. Whereas, the User-centric model eliminates the need of the IDP during authentication. In this section, we discuss the enrollment and authentication phases in both these models.

3.8.1. IDP-centric Authentication Model

The protocols we this use in this model are similar to the protocols utilized in the user-centric model that we adopt from [24]. However, we tweak these protocols to suit the IDP-centric model.

Protocol 2 shows the Enrollment phase in this model. The user acquires the EEG signal, extracts the feature vector and computes the perceptual hash (Line 1 - Line 2). The user also chooses a password (Line 3) and provides the perceptual hash along with the password, $from$ and to fields to the IDP (Line 4). The IDP generates a salt for this user and stores it (Line 5). Using the password and the salt, the IDP derives the required secrets: $secret_1$ and $secret_2$ (Line 6). It then predicts a class label for the perceptual hash and generates the BID (Line 7). It creates a commitment on the BID and $secret_2$. Then, it sets the commitment in the identity token $IDToken$ and stores $IDToken$ (Line 8). Finally, it sends this token to the user (Line 9).

Protocol 2 Enrollment with IDP (IDP-centric Model)

1: U : Captures the EEG signal and extracts the feature vector
2: Computes the perceptual hash for the feature vector (Section 3.5)
3: Chooses a password
4: $U \Rightarrow IDP$: Sends the perceptual hash, the chosen password, $from$ and to fields to the IDP
5: IDP : Generates a Salt and stores it
6: Derives $secret_1$ and $secret_2$ using the password and salt (Section 3.7.3)
7: Predicts the class for the given perceptual hash and generates BID (Section 3.7.4)
8: Creates the commitment and identity token $IDToken$ (Section 3.7.5). Store token.
9: $IDP \Rightarrow U$: IDP sends $IDToken$ to the user

Once the user receives the identity token, he or she can use it to sign-up with the Server using Protocol 3. The user completes the required formalities based on the Server's policies for sign-up, and finally, presents the identity token to the Server (Line 1). The Server first verifies the signature of the IDP on the token to make sure that nobody has tampered with the token. Then, it verifies that the token is still valid by checking the $expires$ field (Line 2). If both of these conditions are satisfied, the Server completes the sign-up process and stores the token for future use (Line 3).

Protocol 4 shows steps involved in the Authentication phase. Again, the User acquires the EEG signals, extracts the feature vector and computes the perceptual hash (Line 1 - Line 2). It

Protocol 3 User sign-up with S

1: $U \Rightarrow S$: User presents the identity token to the S
2: S : Verifies IDP signature and $expires$ field in the token to ensure the token is valid
3: Stores the token

then sends this perceptual hash, password and the other required fields to the IDP and requests for the identity token (Line 3). The IDP retrieves the user's salt and derives the required secrets (Line 4 - Line 5). It then predicts the class for the perceptual hash, generates the BID, creates the commitment and stores the commitment in the token (Line 6 - Line 7). Before sending the token to the user, the IDP needs to verify if the user is authentic. To do that, it compares the token with the token generated during enrollment. As the commitment created depends on the perceptual hash and the password, the IDP can directly compare the commitment in both the tokens to decide if the user is authentic or not. Upon successful verification, the IDP sends the generated token to the user. The user then initiates the authentication request and sends the token to the Server (Line 9). Like before, the Server verifies the IDP signature and $expires$ field to ensure that the token is valid (Line 11). Upon successful validation, the User and the Server run the ZKPK Protocol. If the ZKPK Protocol succeeds, then so does the authentication process (Line 12 - Line 15).

Protocol 4 Authenticating the User (IDP-centric Model)

1: U : Captures the EEG signal and extracts the feature vector
2: Computes the perceptual hash for the feature vector (Section 3.5)
3: $U \Rightarrow IDP$: Sends the perceptual hash, the password, $from$ and to fields to the IDP
4: IDP : Retrieve's the User's Salt
5: Derives $secret_1$ and $secret_2$ using the password and salt (Section 3.7.3)
6: Predicts the class for the given perceptual hash and generates BID (Section 3.7.4)
7: Generates the commitment and identity token $IDToken$ (Section 3.7.5)
8: **if** $IDToken$ matches token generated during enrollment **then**
9: $IDP \Rightarrow U$: IDP sends $IDToken$ to the user
10: $U \Rightarrow S$: User initiates the authentication request and sends $IDToken$ to S
11: S : Verifies IDP signature and $expires$ field in the token to ensure the token is valid
12: **if** IDP signature and $expires$ verified successfully **then**
13: S and U execute ZKPK Protocol (Section 3.7.2)
14: Authentication successful if ZKPK Procotol succeeds
15: **end if**
16: **end if**

3.8.2. User-centric Authentication Model

The IDP-centric model that we saw in the previous section has an important privacy issue. As it is evident from the discussion therein, the user will have to request for an identity token from the IDP every time it wants to authenticate a Server. Thus, the IDP will be aware of all the user's transactions and can derive private information about him or her. The User-centric model overcomes this problem by eliminating the need of the IDP during authentication. We adopt one such approach from [24] and tweak it so that the user gives only the perceptual hash of the feature vectors to the IDP instead of the biometric or the feature vector.

In this model when the user enrolls with the IDP, in addition to the identity token, he or she gets the value of the salt, a customized classification model, and a client authentication application. It would be worthwhile to discuss the need for these three things before looking at the protocols.

- Client authentication application: This application is a standalone application which takes care of generating the token and completing authentication phase without the intervention of the IDP. Thus, the user can complete the authentication phase without involving the IDP. Now, the IDP is no longer aware of the User's transactions and will be unable to learn any private information about the user from them.
- Salt: We discussed in Section 3.7.3 how we derive secrets from the user's password. These secrets are required to generate the BID and identity token (see Sections 3.7.4 and 3.7.5) during both the enrollment and authentication phases. However, in this model, the client authentication application handles the authentication phase and requires the salt to generate BID by deriving the secrets.
- Customized classification model: The IDP trains a classification model, and for each user, it customizes this model before predicting a class for the perceptual hash. The client authentication application requires this classification model to generate the BID (see Section 3.7.4) and identity token during the authentication phase. However, giving the original model as is could compromise the information about the other classes (users) of IDP. A malicious user who somehow manages to get access to the classification model on the client can try to learn information about other classes (users). To prevent this, the

IDP customizes the classification model by replacing the original classes with random 32-bit integers. Since the customization is different for each user, a malicious user will be unable to learn any information about other users by analyzing the model.

Protocol 5 Enrollment with IDP (User-centric Model)

1: U : Captures the EEG signal and extracts the feature vector
2: Computes the perceptual hash for the feature vector (Section 3.5)
3: Chooses a password
4: $U \Rightarrow IDP$: Sends the perceptual hash, the chosen password, $from$ and to fields to the IDP
5: IDP : Generates a Salt
6: Derives $secret_1$ and $secret_2$ using the password and salt (Section 3.7.3)
7: Predicts the class for the given perceptual hash and generates BID (Section 3.7.4)
8: Creates the commitment and identity token $IDToken$ (Section 3.7.5)
9: $IDP \Rightarrow U$: IDP sends $IDToken$, Salt, Customized classification model and Client authentication application to the user

Protocol 5 shows the Enrollment phase in this model. As can be seen, it is almost identical to Enrollment phase in the IDP-centric model (see Protocol 2). In the IDP-centric model, we use both k-NN and SVM for predicting the class. However, in this authentication model, we can only use SVM. k-NN requires all the instances (records of the other users) to predict the class label. For a customized k-NN classifier to work on the user's device or computer, we will need to send records of the each user to every other user. This is not feasible as the IDP cannot risk compromising the privacy of the other class labels, and also the user might have limited processing and storage resources. At the end of the enrollment phase, in addition to the identity token, the IDP sends the generated salt, customized classification model and the client authentication application to the user. Since the client authentication application generates the token for authentication, the IDP no longer needs to store this token. Upon receiving the identity token, the user follows the Protocol 3 discussed in Section 3.8.1 to complete the sign-up at the Server.

Protocol 6 shows steps involved in the Authentication phase. Again, the User acquires the EEG signals, extracts the feature vector and computes the perceptual hash (Line 1 - Line 2). The client authentication application derives the required secrets using the user's password and salt (Line 3). It then uses the customized classification model to predict the class for the given perceptual hash. Next, it generates the BID, creates the commitment and stores the commitment in

Protocol 6 Authenticating the User (User-centric Model)

1: U : Captures the EEG signal and extracts the feature vector
2: Computes the perceptual hash for the feature vector (Section 3.5)
3: Using password and salt (received from IDP), derives $secret_1$ and $secret_2$ (Section 3.7.3)
4: Predicts the class for the given perceptual hash and generates BID (Section 3.7.4)
5: Generates the commitment and identity token $IDToken$ (Section 3.7.5)
6: $U \Rightarrow S$: User initiates the authentication request and sends $IDToken$ to S
7: S : Verifies IDP signature and $expires$ field in the token to ensure the token is valid
8: **if** IDP signature and $expires$ verified successfully **then**
9: S and U execute ZKPK Protocol (Section 3.7.2)
10: Authentication successful if ZKPK Procotol succeeds
11: **end if**

the token (Line 4 - Line 5). Afterwards, it initiates an authentication request and sends the token to the Server (Line 9). The Server verifies the IDP signature and $expires$ field to ensure that the token is valid (Line 11). If the token is valid, the User and the Server run the ZKPK Protocol to complete the authentication process. The authentication is successful if the ZKPK Protocol succeeds (Line 12 - Line 15).

CHAPTER 4

EXPERIMENTS AND RESULTS

In this chapter, we first discuss the datasets used to evaluate our approach followed by the experimental design, and finally, the results of the experiments.

4.1. Datasets

To evaluate our approach for authentication using EEG signals, we use three publicly available datasets: EEG Alcohol, EEG Motor/Imagery, and EEG SSVEP Dataset III. All the signal processing is done using MATLAB [36].

4.1.1. EEG Alcohol Dataset

The EEG Alcohol [17] dataset, available at the UCI Machine Learning Repository [31], was the result of an extensive study to examine genetic predisposition to alcoholism using EEG. The EEG signals were collected using a device with 64 electrodes placed on the subject's scalp and sampled at 256 Hz (3.9 msec epoch) for 1 second. There are two categories of subjects viz., $alcoholic$ and $control$. For collecting the data, each subject was exposed to a single stimulus (S1) or two stimuli (S1 and S2) using pictures of objects from the 1980 Snodgrass and Vanderwart picture set [57]. In the case of two stimuli, the pictures shown are in a matched condition (S1 $=$ S2) or a non-matched condition (S1 $\neq$ S2). The dataset has 122 subjects and 120 trials for each subject based on different stimuli. To evaluate our approach we consider a subset of this dataset consisting of 70 subjects. Table A.1 lists the subjects present in our subset.

4.1.2. EEG Motor/Imagery Dataset

The EEG Motor/Imagery [54] dataset is available at PhysioNet [50] and has over 1500 one and two minute EEG recordings from 109 subjects. The recording of EEG signals is from 64-channels using the BCI2000 system [12] while each subject performs different motor/imagery tasks. Each subject performs two one-minute baseline runs (one with eyes open and one with eyes closed) and three two-minute runs of each of four tasks described below.

(1) In task 1, a target appears on either the left or the right side of the screen. The subject responds by opening and closing the corresponding fist until the target disappears. Then the subject relaxes.

(2) In task 2, a target appears on either the left or the right side of the screen. The subject responds by imagining opening and closing the corresponding fist until the target disappears. Then the subject relaxes.

(3) In task 3, a target appears on either the top or the bottom of the screen. If the target is at the top of the screen, the subject responds by opening and closing both fists. If the target is at the bottom, the subject responds by opening and closing both feet until the target disappears. Then the subject relaxes.

(4) In task 4, a target appears on either the top or the bottom of the screen. If the target is at the top of the screen, the subject responds by imagining opening and closing both fists. If the target is at the bottom, the subject responds by imagining opening and closing both feet until the target disappears. Then the subject relaxes.

In all, each subject performs 14 experimental runs. The EEG signals are sampled at 160 Hz.

We can summarize the 14 experimental runs as below:

(1) Baseline - with eyes open

(2) Baseline - with eyes closed

(3) Task 1 (opening and closing left or right fist)

(4) Task 2 (imagining opening and closing left or right fist)

(5) Task 3 (opening and closing both fists or both feet)

(6) Task 4 (imagining opening and closing both fists or both feet)

(7) Task 1

(8) Task 2

(9) Task 3

(10) Task 4

(11) Task 1

(12) Task 2

(13) Task 3

(14) Task 4

The data is available in EDF+ format, and we use $rdsamp$ in the PhysioToolkit [50] software to extract the data in text format. Each original data file has a corresponding annotation file with the extension *.event* and each annotation has one of three codes viz., $T0$, $T1$ or $T2$:

- $T0$ corresponds to rest.
- $T1$ corresponds to onset of motion (real or imagined) of left fist (in runs 3, 4, 7, 8, 11, and 12) and both fists (in runs 5, 6, 9, 10, 13 and 14).
- $T2$ corresponds to onset of motion (real or imagined) of right fist (in runs 3, 4, 7, 8, 11, and 12) and both feet (in runs 5, 6, 9, 10, 13 and 14).

The annotation file has useful information like the event code, the time and sample when the event started and the event's duration. We convert its data into text format using $rdedfann$. Moreover, we extract each event $T0$, $T1$ and $T0$ as a separate record. Events $T1$ and $T2$ relate to a real or imagined motion which is in response to some visual stimulus whereas $T0$ relates to the rest state. Since, we are interested in VEP signals (EEG signals in response to a visual stimulus), we only consider events $T1$ and $T2$. Also, to evaluate our approach, we randomly select a subset of the dataset consisting of 27 subjects. Table A.2 lists the subjects present in our subset.

4.1.3. EEG SSVEP Dataset III

The MAMEM SSVEP Dataset III [2] has EEG signals acquired from 11 subjects using 14 channels while the subjects were executing a Steady State VEP (SSVEP)-based experimental protocol. The EEG signals are captured using the Emotiv [6] EPOC deadset with a sampling rate of 128 Hz. The stimuli of the experiment are five simultaneously flickering violet boxes on a computer screen with each box flickering at a different frequency (6.66, 7.5, 8.57, 10.0, 12.0 Hz). In a trial, the subject focuses on a flickering box indicated by a yellow arrow for 5 seconds and then rests for next 5 seconds. The experiment for each subject is initiated by an adaptation period of 100 seconds which has 10 trials. This is followed by 5 identical sessions having two parts and overall 25 trials. The first part has 12 trials whereas the second part has 13 trials and both parts

are separated by a 30 second interval. From this data, we extract the 5 seconds intervals where the user focuses on the stimulus and consider this a one signal. Thus, we have 135 signals for each subject (10 signals from adaptation + 5 sessions * 25 signals from each session).

Dataset	Subjects	Total Signals	Signals per Subject
EEG Alcohol	70	4827	≈69 (largely vary per subject)
EEG Motor/Imagery	27	4881	≈180
EEG SSVEP III	11	1485	135

TABLE 4.1. Summary of EEG datasets

4.1.4. Preprocessing

EEG signals may contain artifacts, i.e., the electrical activity resulting from activities such as eye motion, eye blinking, etc. To detect and eliminate these artifacts, we employ an amplitude threshold filtering technique. We discard VEP signals having magnitude over 50 μV as they are assumed to be contaminated [56] [65]. We remove artifacts only from the EEG Alcohol dataset. Next, to extract the γ band from the EEG signals, we further process the artifact-free signals following the steps in [65]. We use a 10^{th} order Butterworth digital filter to filter the signals with a 30-50 Hz passband. To cancel the non-linearity of this filter, we use forward and reverse filtering. Finally, we discard the first and last 20 samples as they do not represent an appropriately filtered signal. In the EEG Alcohol dataset, after preprocessing and dropping 40 samples (first and last 20), each signal has 216 samples. Also, depending on the number of EEG channels selected initially, we will have C_T signals (see Table 3.3). We have a total of 4827 signals for 70 subjects in this dataset. The number of signals per subject varies largely. In the EEG Motor/Imagery dataset, as per the annotation file, each event lasts for 4.1 seconds. Since the sampling rate is 160, we consider 640 samples (i.e., samples for 4 seconds). After preprocessing, each signal has 600 samples as we discard 40 samples (first and last 20). Also, depending upon the number of EEG channels selected initially, we will have C_T signals (see Table 3.3 or section 3.3). In this dataset, we have a total of 4881 signals from 27 subjects. Unlike the EEG Alcohol dataset, the number of signals per subject is rather uniform. All but three subjects have 180 signals each, and the average is still ≈180 signals per subject. Similarly, in the SSVEP Dataset III, each signal has 640 samples (5 seconds

* 128 samples/second) that are reduced to 600 after preprocessing. Also, since each subject has 135 signals, and there are 11 subjects, we have a total of 1485 signals in this dataset. Table 4.1 summarizes the details about the EEG Alcohol, EEG Motor/Imagery and SSVEP Dataset III.

4.2. Experimental Design

We implement the User, IDP and Server programs in JAVA SE 1.8 using an open source crypto library Qilin [38] and assume all communication happens over secure communication channels. We conduct the experiments on a desktop machine with Intel Core(TM) i7-4770 CPU @ 3.40 GHz processor, 16 GB RAM and Windows 7 Professional 64 bit. We use Weka 3.6 [26] implementations of SMO-SVM and IBk (k-NN) for training and 10-fold cross validation (stratified sampling) for evaluation. In each iteration, we use 8 folds as train set and 1 fold each as validation and test set. For k-NN, we experiment with $k = 1$ and use the default settings in Weka. For SVM, we experiment with Poly kernel and RBF kernel. For Poly kernel, we vary the exponent $E = [1, 2]$. For RBF kernel, we vary $\gamma = [0.01, 0.1, 0.05, 1.0]$. For both kernels, we vary $C = [0.01, 0.1, 1.0, 5.0, 10.0]$. We choose the hyper-parameters that give the best performance on the validation set and report their results on the test set. We present F1 score (F1) and Accuracy (Acc.) to measure the performance of the classifiers and False Acceptance Rate (FAR) and False Rejection Rate (FRR) to gauge the performance of the authentication system.

Recall gives the proportion of samples or records that were classified as class x, among all samples or records that truly belong to class x. It is calculated as $\frac{TP}{TP+FN}$ where TP is True Positive, and FN is False Negative. Precision is the proportion of the records which actually have class x among all those records that were classified as class x. The F-Measure or F1 score is a measure that takes into account both precision and recall and is computed as $\frac{2\times(Prec.)\times(Rec.)}{(Prec.)+(Rec.)}$ [4]. FAR and FRR are commonly used metrics to compare the performance of a biometric authentication system. FAR is the measure of the likelihood that the biometric authentication system will incorrectly accept an unauthorized user's access attempt [16]. We calculate FAR as $\frac{Impostors\ Accepted}{Total\ Impostors}$. FRR is the measure of the likelihood that the biometric authentication system will incorrectly reject an authorized user's access attempt [3]. We calculate FRR as $\frac{Genuines\ Rejected}{Genuine\ Users}$.

4.2.1. FAR and FRR Calculation

We compute the values for FAR and FRR for each class and then calculate the weighted average to get the FAR and FRR values for the authentication system, from the confusion matrix generated by Weka. We show these calculations for a small dataset having four classes (a, b, c and d) and 248 instances in Table 4.2. In the table, the first five rows and five columns represent the confusion matrix. For each class:

(1) The sum of the row elements gives the total instances of that class $Total$. These are also the genuine users (G) of that class. Ex. For class a, $Total = G = (24+12+0+0) = 36$.

(2) The diagonal elements denote the number of instances that are genuine and accepted (GA). Ex. For class a, $GA = 24$.

(3) The sum of row elements except the diagonal element indicates the instances that are genuine but were rejected (GR). Ex. For class a, $GR = (12 + 0 + 0) = 12$.

(4) The sum of column elements except the diagonal element denotes the instances that are impostors but were accepted (IA). Ex. For class a, $IA = (8 + 1 + 0) = 9$.

(5) Instances of all the other classes will be impostors and their sum will be $Total\ Impostors$ (I). Ex. For class a, impostors will be the sum of all instances of class b, c and d i.e., $I = (55 + 55 + 102) = 212$.

(6) We calculate the FAR as $\frac{IA}{I}$ and FRR as $\frac{GR}{G}$. Ex. For class a, $FAR = \frac{9}{212} = 0.0425$, and $FRR = \frac{12}{36} = 0.3333$.

(7) We then calculate the weighted FAR, $Wt\ FAR = FAR \times (\frac{Total\ Instances\ of\ Class}{Total\ Instances\ in\ Dataset}) = 0.0425 \times (\frac{36}{248}) = 0.0062$ and weighted FRR, $Wt\ FRR = FRR \times (\frac{Total\ Instances\ of\ Class}{Total\ Instances\ in\ Dataset}) = 0.3333 \times (\frac{36}{248}) = 0.0484$.

Once we have the weighted FAR and FRR values for each class, we sum the respective values to compute the FAR and FRR for the authentication system. For the authentication system, $FAR = (0.0062 + 0.0391 + 0.0057 + 0) = 0.0509$, and $FRR = (0.0484 + 0.0524 + 0.0887 + 0.0040) = 0.1935$.

a	b	c	d	← Classified as	Total	I	GA	GR	IA	FAR	FRR	Wt FAR	Wt FRR
24	12	0	0	a	36	212	24	12	9	0.0425	0.3333	0.0062	0.0484
8	42	5	0	b	55	193	42	13	34	0.1762	0.2364	0.0391	0.0524
1	21	33	0	c	55	193	33	22	5	0.0259	0.4	0.0057	0.0887
0	1	0	101	d	102	146	101	1	0	0	0.0098	0	0.0040

TABLE 4.2. Calculation of FAR and FRR

4.2.2. Experiments on Feature Extraction Techniques

We carry out different experiments for each feature extraction technique to improve the overall accuracy of our approach. For both the features extraction techniques, i.e., PSPRT and PSD, we experiment by selecting 8 and 17 EEG channels. In some cases, we also experiment with 26 EEG channels when extracting features using PSPRT only. As using 26 EEG channels, PSD would yield a feature vector with 7020 features and would increase the complexity of training the classifiers. However, as we have EEG signals from 14 channels in the SSVEP Dataset III, we experiment only with 14 channels. As discussed in Section 3.4.2, we can vary the $window$ and $overlap$ while calculating the PSD's features using $pwelch$. In the Table 4.3, we show the different combinations we experiment with on both the datasets. The entries are in the form [$window$, $overlap$] and samples column indicates the number of samples present in the EEG signal (see preprocessing in Section 4.1.4).

Dataset	Samples	Combinations
EEG Alcohol	216	[54, 27], [54, 0], [108, 54], [108, 0]
EEG Motor/Imagery	600	[300, 150], [300, 75], [300, 0], [150, 75], [150, 0], [75, 40], [75, 0]
EEG SSVEP III	600	[300, 150], [300, 75], [300, 0], [150, 75], [150, 0], [75, 40], [75, 0]

TABLE 4.3. Combinations of window and overlap for calculating PSD's features

4.2.3. Experiments on Perceptual Hashing

Once we have the feature vector, we compute the perceptual hash. Now, as discussed in Section 3.5, we compute the perceptual hashes in two ways:

(1) Using the entire feature vector as a segment

(2) Dividing the feature vector into two segments with at least 50% features in the first segment.

In the latter case, the number of samples depends on the number of EEG channels chosen and the feature extraction technique (see Table 3.3). Following are the sets of segment values we choose for our experiments. Note that (x, y) indicates x samples in the first segment and y samples in the second segment:

- 8 EEG Channels: PSPRT - (18, 18), PSD - (360, 360)
- 14 EEG Channels: PSPRT - (50, 55), PSD - (1050, 1050)
- 17 EEG Channels: PSPRT - (80, 73), PSD - (1530, 1530)
- 26 EEG Channels: PSPRT - (180, 171)

4.2.4. Experiments on the Overall Architecture

We first look at how perceptual hashing impacts the classifier training time and later look at how perceptual hashing affects the protocol execution time and bandwidth consumption in both the models, i.e., IDP-centric model and User-centric model.

4.2.5. Baseline

We perform different experiments to see the impact of perceptual hashing on the performance of the system, training time of the classifiers, protocol execution time and bandwidth consumption. To be able to study the effect of perceptual hashing on these, we run each experiment without perceptually hashing the feature vectors and use those results as baselines.

4.2.6. Terminology

We use the following terms in the tables and discussions of our results:

- *FV*: Feature vector extracted from EEG signals.
- *HFV100*: Hash of the FV computed by considering the entire FV as a single segment.
- *HFV50*: Hash of the FV computed by dividing FV into two segments with the first segment having at least 50% of the features.
- *Ch.*: Number of EEG channels.
- *H-Params*: Hyper-parameters.

- *P, E* e*, C* c: Poly kernel, $Exponent$ and C-parameter.
- *R, C* c*,* γ g: RBF kernel, C-parameter and γ.

4.3. Results

4.3.1. Impact of Perceptual Hashing on Authentication

First, we discuss the effect of perceptual hashing PSPRT's features that we use for training the SVM and k-NN classifiers on both the datasets. Followed by this we study the effect of perceptual hashing PSD's features that we use for training both classifier on both the datasets.

Ch.	Features	H-Params	FAR	FRR	F1	Acc. (%)
EEG Alcohol Dataset						
8	FV	P, E 1, C 10.0	0.001	0.0723	0.928	92.77
	HFV100	P, E 2, C 0.01	**0.0061**	**0.376**	**0.614**	**62.40**
	HFV50	R, C 1.0, γ 0.1	0.0068	0.4052	0.583	59.48
17	FV	P, E 1, C 10.0	0.0005	0.0342	0.966	96.58
	HFV100	R, C 1.0, γ 0.05	**0.0011**	**0.0849**	**0.917**	**91.51**
	HFV50	R, C 1.0, γ 0.01	0.0013	0.0885	0.912	91.15
26	FV	R, C 1.0, γ 0.1	0.0003	0.0186	0.981	98.14
	HFV100	P, E 1, C 0.01	0.0004	0.0298	0.97	97.02
	HFV50	P, E 1, C 0.01	**0.0004**	**0.0271**	**0.973**	**97.29**
EEG Motor/Imagery Dataset						
8	FV	R, C 10.0, γ 1.0	0.0008	0.0213	0.979	97.87
	HFV100	R, C 5.0, γ 0.1	0.0152	0.393	0.6	60.7
	HFV50	R, C 0.1, γ 0.01	**0.0153**	**0.3915**	**0.598**	**60.85**
17	FV	R, C 10.0, γ 0.01	0.0005	0.0117	0.988	98.67
	HFV100	R, C 10.0, γ 0.01	0.0013	0.0332	0.967	96.68
	HF50	R, C 10.0, γ 0.01	**0.0013**	**0.0328**	**0.967**	**96.72**
26	FV	R, C 10.0, γ 0.05	0.0003	0.0084	0.992	99.16
	HFV100	R, C 10.0, γ 0.05	0.0013	0.0332	0.967	96.68
	HFV50	R, C 10.0, γ 0.05	**0.0013**	**0.0322**	**0.968**	**96.78**
EEG SSVEP Dataset III						
14	FV	R, C 10.0, γ 0.05	0.0037	0.037	0.963	96.30
	HFV100	R, C 10.0, γ 0.05	0.0098	0.0976	0.903	90.24
	HF50	R, C 10.0, γ 0.05	**0.009**	**0.0896**	**0.911**	**91.04**

TABLE 4.4. Impact of perceptual hashing PSPRT's features on authentication (SVM)

Impact of perceptually hashing PSPRT's features: Table 4.4 shows the impact of perceptual hashing on authentication using SVM. On the EEG Alcohol dataset, for 8 EEG channels, the accuracy drops significantly (by about 30%) when we train the classifier on the perceptual hashes of the feature vectors (HFV100 and HFV50), and on the EEG Motor/Imagery dataset, the accuracy drops by around 37%. Note that PSPRT yields a feature vector having just 36 features. Perhaps, perceptually hashing this feature vector results in the loss of the pattern that distinguishes one subject from the other. Accuracy on HFV100 is better than HFV50 for the EEG alcohol dataset, but the accuracy on them is more or less similar for the EEG Motor/Imagery dataset. To verify that our reasoning for the drop in the accuracy was in fact due to fewer features, we ran some experiments with 17 EEG channels, and in this case, we have 153 features (see Table 3.4). As you can see from the Table 4.4, the accuracy now drops by about 5% when we perceptually hash the signal. On the EEG Motor/Imagery dataset, the drop in accuracy is almost 2%. With 26 EEG channels, this drop reduces to around 1% on the EEG Alcohol dataset. However, on the EEG Motor/Imagery dataset, we do not observe any significant improvement as compared to the accuracy on HFV of 17 EEG channels. On the SSVEP Dataset III, with just 14 channels, the accuracy drops only by about 6% when we perceptually hash the EEG signals. Additionally, it can be seen that increasing the channels improves the accuracy of the classifier on both FV and HFV. Thus, it can be said that our reasoning for the drop in accuracy is plausible. In the following discussion, as will be seen, this applies to k-NN as well. With more EEG channels (14, 17 and 26), performance on HFV100 and HFV50 is more or less similar. Additionally, as we increase the channels, FAR and FRR decrease, which indicates the authentication system is performing better.

We next compare the performance of k-NN on both the datasets. Please refer to Table 4.5. Just like SVM, even k-NN performs poorly when it is trained on HFV from 8 EEG channels. With 14 EEG channels, on the EEG SSVEP Dataset III, we see an improvement in the performance of the classifiers when trained on perceptual hashes. Additionally, as we increase the EEG channels, the drop in the accuracy reduces. This observation is constant over both the datasets. However, on the EEG Motor/Imagery Dataset, the accuracy of the classifier trained on HFV reduces as we increase the channels from 17 to 26. The reason behind this is not clear, but we can say that with

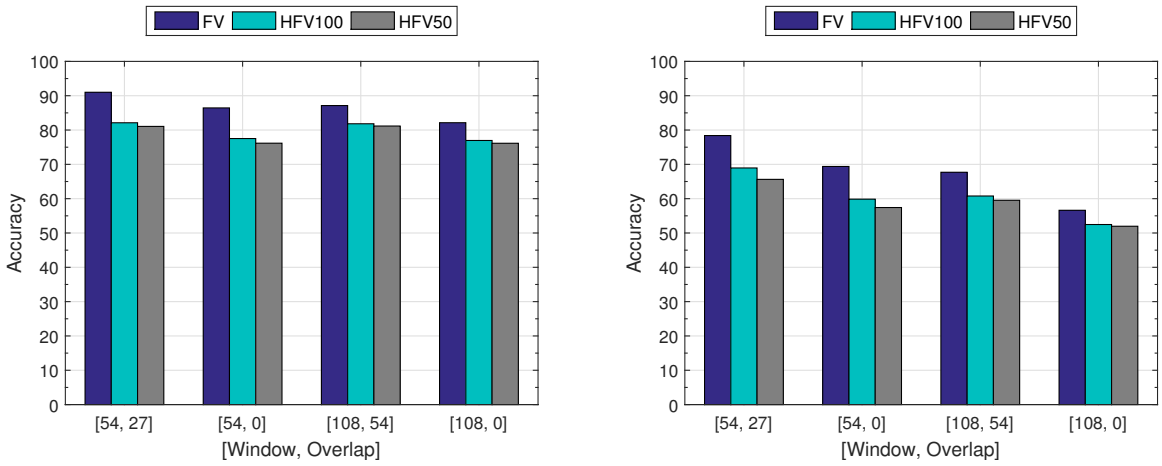
Ch.	Features	FAR	FRR	F1	Acc. (%)
EEG Alcohol Dataset					
8	FV	0.0029	0.186	0.811	81.4
	HFV100	0.0072	0.4313	0.524	54.07
	HFV50	**0.0076**	**0.4485**	**0.54**	**55.15**
17	FV	0.0012	0.0704	0.929	92.96
	HFV100	**0.0023**	**0.1452**	**0.853**	**85.48**
	HFV50	0.0024	0.1496	0.848	85.04
26	FV	0.0006	0.0358	0.964	96.42
	HFV100	0.0011	0.0671	0.932	93.29
	HFV50	**0.001**	**0.0634**	**0.936**	**93.66**
EEG Motor/Imagery Dataset					
8	FV	0.0028	0.0711	0.929	92.89
	HFV100	0.0158	0.4067	0.59	59.33
	HFV50	**0.0156**	**0.4001**	**0.595**	**59.99**
17	FV	0.001	0.0256	0.974	97.44
	HFV100	**0.0022**	**0.0574**	**0.943**	**94.26**
	HFV50	0.0023	0.0598	0.94	94.02
26	FV	0.0009	0.0236	0.976	97.64
	HFV100	0.0025	0.0645	0.935	93.55
	HFV50	**0.0024**	**0.0615**	**0.938**	**93.85**
EEG SSVEP Dataset III					
14	FV	0.0085	0.0848	0.916	91.52
	HFV100	0.0178	0.1785	0.817	82.15
	HF50	**0.0156**	**0.1562**	**0.84**	**84.38**

TABLE 4.5. Impact of perceptual hashing PSPRT's features on authentication (k-NN)

PSPRT as a feature extraction technique, we do not see a significant improvement in the performance of the classifier and the authentication system as we increase the channels from 17 to 26. Also, for k-NN, we do not see any significant difference between accuracy of the classifiers trained on HFV100 and HFV50 on the EEG Alcohol and EEG Motor/Imagery datasets. However, on the EEG SSVEP Dataset III, the performance on HFV50 is better than HFV100.

Impact of perceptually hashing PSD features: Now let us examine the effect of perceptual hashing PSD's features to train the classifiers on both the datasets. As already discussed, we experiment with 8, 14 and 17 EEG channels only. To compute the PSD features we can vary the $window$ and

overlap parameters passed to *pwelch* (see Section 3.4.2). We looked at the performance of the classifiers on all the datasets by varying these parameters and chose the one that gives us the best performance. Refer to Table 4.3 for these combinations.



(a) SVM (b) k-NN

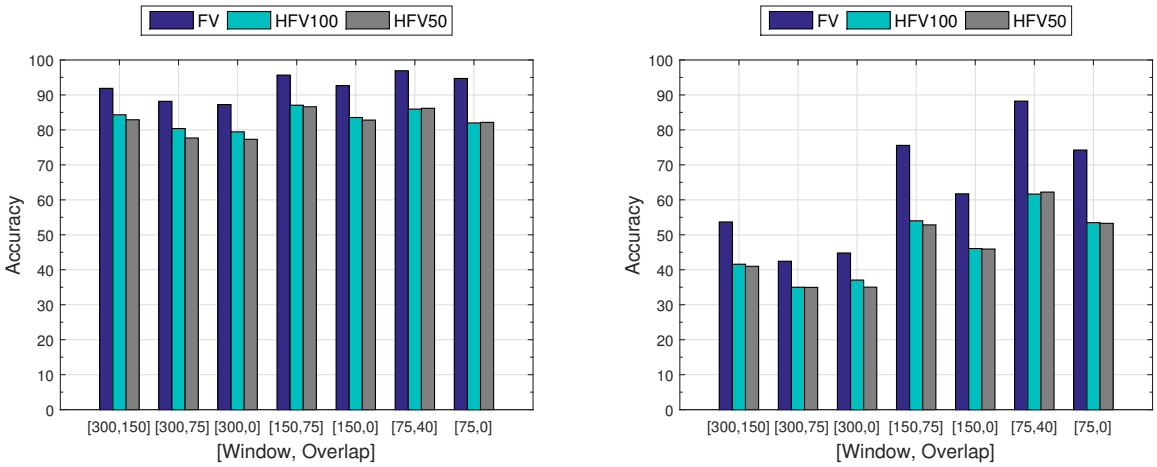
FIGURE 4.1. Accuracy of classifiers trained on PSD's features computed from different window and overlap - EEG Alcohol

Classifier	Measures	[54, 27]	[54, 0]	[108, 54]	[108, 0]
SVM	Acc. (%)	82.12	77.52	81.83	76.96
	F1	0.822	0.776	0.822	0.775
k-NN	Acc. (%)	68.95	59.86	60.78	52.47
	F1	0.683	0.59	0.594	0.511

TABLE 4.6. Performance of PSD's features on different combinations of window and overlap - EEG Alcohol

On the EEG Alcohol dataset, both the classifiers perform the best when trained on the HFV of PSD features computed with a *window* of 54 and *overlap* of 27. Figure 4.1(a)-(b) shows the performance of SVM and k-NN, respectively, on different combinations of *window* and *overlap*. Even the F1 score for this combination was the best among other combinations (Refer to Table 4.6).

Choosing the best combination on the EEG Motor/Imagery dataset was not that straight forward. Figure 4.2(a)-(b) show the accuracy of SVM and k-NN, respectively, on different combinations. Whereas Table 4.8 compares the F1 scores of the classifiers trained on HFV of PSD's



(a) SVM (b) k-NN

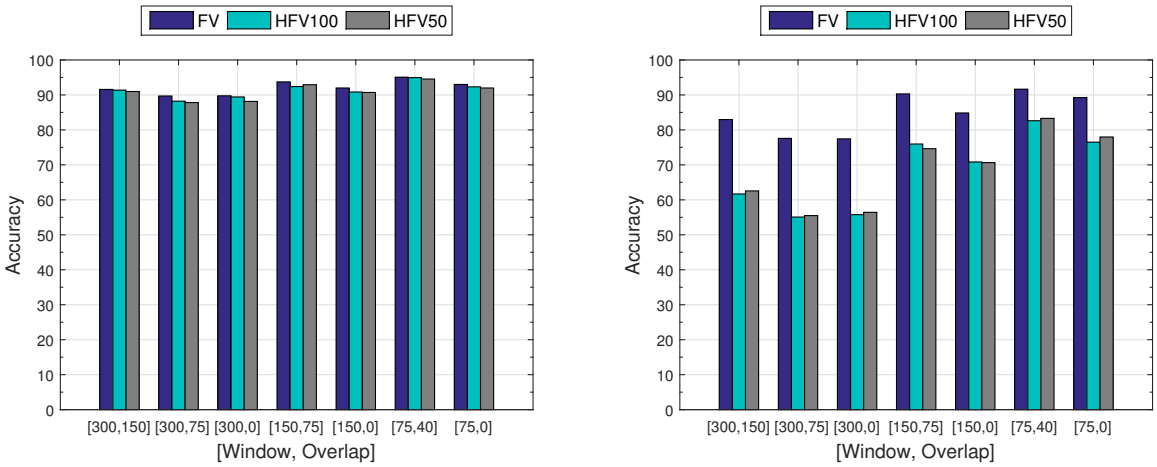
FIGURE 4.2. Accuracy of classifiers trained on PSD's features computed from different window and overlap - EEG Motor/Imagery

Classifier	Measures	[300, 150]	[300, 75]	[300, 0]	[150, 75]	[150, 0]	[75, 40]	[75, 0]
SVM	Acc. (%)	84.33	80.39	79.45	87.07	83.55	86.19	82.16
	F1	0.843	0.804	0.79	0.871	0.836	0.863	0.822
k-NN	Acc. (%)	41.59	35.01	37.06	54.01	46.08	62.22	53.47
	F1	0.416	0.351	0.371	0.539	0.461	0.624	0.537

TABLE 4.7. Performance of PSD's features on different combinations of window and overlap - EEG Motor/Imagery

features of these combinations. As can be seen, the F1 score of SVM trained on the HFV of the combination [150, 75] is slightly better than [75, 40]. However, we chose [75, 40] for two reasons: (1) Accuracy of the SVM when trained on the FV i.e., PSD's features of this combinations has an accuracy of 96.93% and an F1 score of 0.969, which is better than all other combinations we experimented with; (2) F1 score of k-NN when trained on the HFV of this combination's PSD's features is the best. On the SSVEP Dataset III, it can be clearly seen from Figure 4.3(a)-(b) that the HFV of the combination of [75, 40] outperforms the others. Next, we will discuss in detail the results of these combinations i.e., $window$ of 54 and $overlap$ of 27 on the EEG Alcohol dataset, and $window$ of 75 and $overlap$ of 40 on the EEG Motor/Imagery dataset and SSVEP Datatset III. Also, since these combinations performed the best on 8 EEG channels, we use them for our experiments on 17 EEG channels on the EEG Alcohol and EEG Motor/Imagery datasets.

Table 4.9 presents the performance of SVM trained on PSD's features (FV) and their per-



(a) SVM (b) k-NN

FIGURE 4.3. Accuracy of classifiers trained on PSD's features computed from different window and overlap - EEG SSVEP Dataset III

Classifier	Measures	[300, 150]	[300, 75]	[300, 0]	[150, 75]	[150, 0]	[75, 40]	[75, 0]
SVM	Acc. (%)	91.38	88.22	89.43	92.93	90.84	94.95	92.32
	F1	0.915	0.884	0.896	0.93	0.909	0.95	0.924
k-NN	Acc. (%)	62.56	55.49	56.43	75.96	70.84	83.30	77.98
	F1	0.598	0.523	0.536	0.743	0.689	0.824	0.765

TABLE 4.8. Performance of PSD's features on different combinations of window and overlap - EEG SSVEP Dataset III

ceptual hashes (HFV) on both the datasets. For 8 EEG channels, even here the accuracy drops when SVM is trained on HFVs. On the EEG Alcohol dataset, the accuracy drops by about 10%, and on the EEG Motor/Imagery dataset, the accuracy drops by around 9%. However, the drop in the accuracy is not as severe as the drop we saw in the case of HFVs of PSPRT's features. With PSD, there are 720 features after feature extraction, which is far greater than 36 features in the case of PSPRT (see Table 3.3). Thus, even after perceptually hashing them, the hashes still retain some unique pattern that helps in identifying a subject. With 17 channels the accuracy of the classifier when they are trained on HFV again improves. As can be seen from Table 4.9, on the EEG Alcohol dataset, the drop in accuracy is about 2.5% (95.83% on FV and 93.47% on HFV). Interestingly, on the EEG Motor/Imagery dataset, the accuracy on the FV (98.98%) and the HFV50 (98.46%) is very close at it drops by 0.52%. Similarly, on the EEG SSVEP Dataset III, with just 14 channels,

the accuracy on HFV and FV is more or less similar. Furthermore, except in the case of 8 EEG channels on the EEG Alcohol dataset, where performance on HFV100 is better than HFV50, in all other cases their performance is comparable.

Ch.	Features	H-Params	FAR	FRR	F1	Acc. (%)
EEG Alcohol Dataset ($window = 54, overlap = 27$)						
8	FV	P, E 1, C 0.1	0.0013	0.0899	0.91	91.01
	HFV100	P, E 1, C 0.01	**0.0026**	**0.1788**	**0.822**	**82.12**
	HFV50	P, E 1, C 0.01	0.0027	0.1894	0.812	81.06
17	FV	R, C 10.0, γ 0.01	0.0006	0.0417	0.959	95.83
	HFV100	P, E 2, C 0.1	0.0007	0.0663	0.938	93.37
	HFV50	P, E 2, C 10.0	**0.0007**	**0.0653**	**0.939**	**93.47**
EEG Motor/Imagery Dataset ($window = 75, overlap = 40$)						
8	FV	R, C 5.0, γ 0.05	0.0012	0.0307	0.969	96.93
	HFV100	R, C 10.0, γ 0.01	0.0054	0.1405	0.86	85.95
	HFV50	R, C 10.0, γ 0.01	**0.0053**	**0.1381**	**0.863**	**86.19**
17	FV	R, C 10.0, γ 0.01	0.0004	0.0102	0.99	98.98
	HFV100	P, E 2, C 10.0	0.0007	0.017	0.983	98.3
	HFV50	P, E 2, C 10.0	**0.0006**	**0.0154**	**0.985**	**98.46**
EEG SSVEP Dataset III ($window = 75, overlap = 40$)						
14	FV	P, E 2, C 0.01	0.0049	0.0492	0.95	95.08
	HFV100	P, E 2, C 10.0	**0.0051**	**0.0505**	**0.95**	**94.95**
	HFV50	P, E 2, C 10.0	0.0055	0.0545	0.946	94.55

TABLE 4.9. Impact of perceptual hashing PSD's features on authentication (SVM)

Finally, we discuss the performance of k-NN when it is trained on the HFVs of PSD's features. Please refer to Table 4.10. For 8 EEG channels, on the EEG Alcohol dataset, we get an accuracy of 78.39% when the classifier is trained on FV. Although this combination of $window$ and $overlap$ gives the best result amongst the combination we experimented with, it is still poor. The accuracy further worsens (to 68.95%) when we train k-NN on the HFVs. Similarly, on the EEG Motor/Imagery dataset, the accuracy is 88.24% on FV, but drops to 62.22% on HFVs. Again, with 17 EEG channels, the performance of this classifier improves on both the datasets. However, the drop in the accuracy after perceptually hashing the signals is still significant as compared to the drop in accuracy of SVM when it is trained on HFV of 17 EEG channels. Also, the performance on HFV100 is better than HFV50 in two cases viz., 8 EEG channels on EEG Alcohol dataset and

Ch.	Features	FAR	FRR	F1	Acc. (%)
EEG Alcohol Dataset ($window = 54, overlap = 27$)					
8	FV	0.0034	0.2161	0.781	78.39
	HFV100	**0.0047**	**0.3105**	**0.683**	**68.95**
	HFV50	0.0052	0.3437	0.647	65.63
17	FV	0.0014	0.0897	0.91	91.03
	HFV100	0.002	0.1298	0.868	87.02
	HFV50	**0.0019**	**0.1277**	**0.869**	**87.23**
EEG Motor/Imagery Dataset ($window = 75, overlap = 40$)					
8	FV	0.0046	0.1176	0.883	88.24
	HFV100	0.0149	0.3835	0.62	61.65
	HFV50	**0.0146**	**0.3778**	**0.624**	**62.22**
17	FV	0.0015	0.0389	0.961	96.11
	HFV100	**0.0039**	**0.1024**	**0.889**	**89.76**
	HFV50	0.0043	0.1115	0.899	88.85
EEG SSVEP Dataset III ($window = 75, overlap = 40$)					
14	FV	0.0084	0.0835	0.916	91.65
	HFV100	0.0174	0.1737	0.814	82.63
	HFV50	**0.0167**	**0.167**	**0.824**	**83.3**

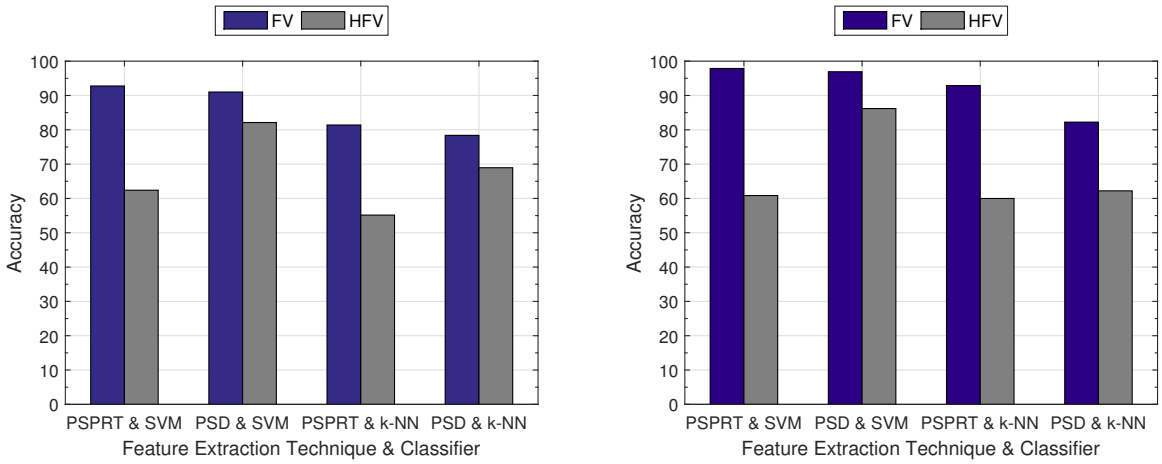
TABLE 4.10. Impact of perceptual hashing PSD's features on authentication (k-NN)

17 EEG channels on the EEG Motor/Imagery dataset. In the remaining two cases the performance on them is more or less similar.

Thus, we see that as we increase the EEG channels, the performance of the system also increases. Both the accuracy and the F1 score of the classifiers trained on FV and HFV become comparable. Moreover, FAR and FRR also reduce as we increase the channels. Note that the increase in the EEG channels results in an increase in the number of features. Ex. For 17 EEG channels, PSPRT yields just 153 features whereas PSD yields 3060 features. Although the increase in the number of features does not affect the authentication phase, it does impact the classifier training time, which we discuss in Section 4.3.2. Also, the performance of the classifiers on HFV100 and HFV50 is comparable in most of the cases.

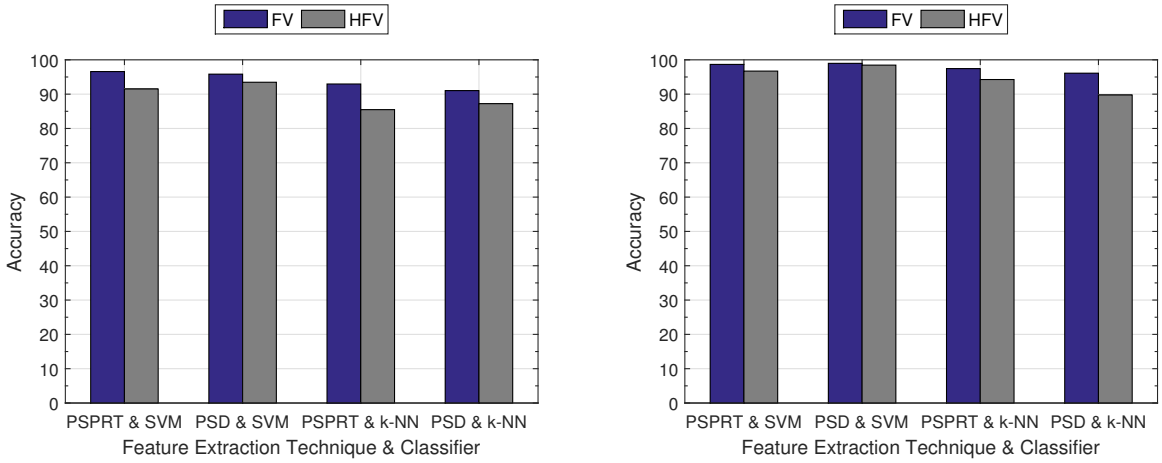
Comparison of SVM and k-NN on PSPRT and PSD: It would be interesting to see how the classifiers perform when trained on our feature extraction techniques. This comparison also helps us

compare the feature extraction techniques. We first compare the performance of the classifiers and later compare the feature extraction techniques based on the performance of the classifiers. Also, we look at the accuracy and F1 score of the classifiers trained on FV and HFV. Since, we have two hashes, i.e., HFV100 and HFV50, we choose the one that gives the best performance.



(a) EEG Alcohol (b) EEG Motor/Imagery

FIGURE 4.4. Comparison of SVM and k-NN on PSPRT and PSD (8 channels)

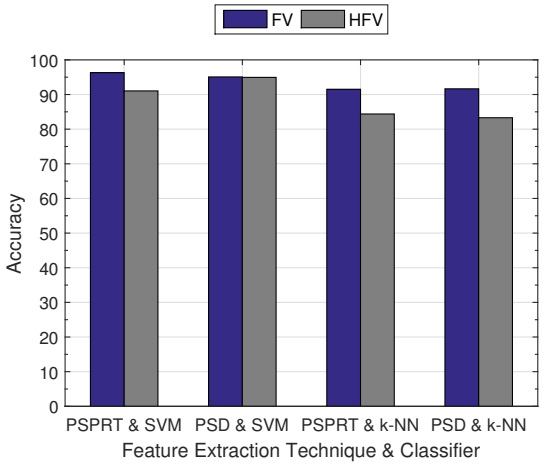


(a) EEG Alcohol (b) EEG Motor/Imagery

FIGURE 4.5. Comparison of SVM and k-NN on PSPRT and PSD (17 channels)

When we consider 8 EEG channels, the accuracy of SVM on both the datasets is far superior to k-NN on both the datasets (see Figure 4.4(a)-(b)). However, when we consider 17 EEG

channels, although the accuracy achieved by SVM is greater that k-NN on both the datasets, the difference is not as large as the difference we saw with 8 EEG channels (see Figures 4.5(a)-(b)). Similarly, on the SSVEP Dataset III, SVM clearly outperforms k-NN (see Figure 4.6(a)). Note that, we are more concerned with the performance of the classifiers when they are trained on the perceptual hashes, and hence, we conclude that SVM is more suitable for our approach.



(a) EEG SSVEP Dataset III

FIGURE 4.6. Comparison of SVM and k-NN on PSPRT and PSD (14 channels)

Next, we compare the feature extraction techniques using the performance of the classifiers. Figure 4.4(a)-(b) shows this comparison, for 8 EEG channel data, on the EEG Alcohol and EEG Motor/Imagery dataset, respectively. As can be seen, for both the datasets, the classifiers perform better when they are trained on the FV extracted using PSPRT. However, the performance of the classifiers deteriorates when they are trained on the HFV of PSPRT's features. However, the decline in the accuracy of classifiers trained on HFVs of PSD's features is lower as compared to HFV of PSPRT's features. Since we are concerned about the accuracy when the classifiers are trained on perceptual hashes, we conclude that PSD performs better than PSPRT for 8 EEG channels. Figure 4.5(a)-(b) show the comparison on the EEG Alcohol and EEG Motor/Imagery dataset, respectively, when we consider data from 17 EEG channels. As can be seen from the figures, the performance of the classifier on HFV of PSPRT's features and HFV of PSD's features is more or less similar. However, the performance of SVM is slightly better on both the datasets

when it is trained on the HFV of PSD's features. Hence, yet again, PSD is better suited for our approach. Refer Figure 4.6(a) for a similar comparison on the EEG SSVEP Dataset III. As it can be seen, SVM trained on HFV of PSD's features clearly outperforms the SVM trained on HFV of PSPRT's features. However, although the performance of k-NN trained on the PSPRT's features performs marginally better than the k-NN trained on PSD's features, k-NN still performs poorly as compared to SVM. Therefore, on the basis of these experiments, SVM trained on the perceptual hashes of PSD features gives the best performance.

4.3.2. Impact of Perceptual Hashing on Classifier Training Time

Table 4.11 shows the time it takes to train the SVM classifier on features of PSPRT and PSD on all the datasets. Since, we have two hashes, i.e., HFV100 and HFV50, we present the training time for the classifier and the corresponding kernel that has the best performance. As can be seen from the table, for both PSPRT and PSD, training the classifier on HFV takes more time than training it on FV. However, on the EEG Motor/Imagery dataset, the training time on FV of PSD's features from 17 EEG channels is 53.51 seconds whereas the training time on HFV is 53.94 seconds. The SVM classifier tries to achieve maximum separation between the hyperplanes. In all previous cases, making this separation might have been difficult, which directly impacts the training time. However, in this case, making the separation might not have been as difficult. Also, it takes less time to train SVM on features extracted using PSPRT than it takes to train on features extracted using PSD. For 8 EEG channels, note that PSPRT yields only 36 features as opposed to 720 features yielded by PSD. Similarly, when we consider 17 EEG channels, with PSPRT we just have 153 features while PSD yields 3060 features. Since PSD has more features as compared to PSPRT, training the SVM on the features of PSD takes longer. Finally, note that the time taken to train the SVM classifier on the EEG SSVEP Dataset III is the least as it is a small dataset as compared to the other datasets.

For k-NN, however, the training time was always under 1 second irrespective of the feature extraction technique. Moreover, training the classifier directly on the feature vectors or their perceptual hashes also did not impact the training time as it was still under 1 second in both these cases. A possible reason for this could be the relatively small size of the datasets.

Ch.	Features	Feature Extraction Technique	
		PSPRT	PSD
EEG Alcohol Dataset			
8	FV	4.33 sec (Poly)	8.1 sec (Poly)
	HFV	6.29 sec (Poly)	10.06 sec (Poly)
17	FV	4.62 sec (Poly)	56.06 sec (RBF)
	HFV	12.75 sec (RBF)	67.17 sec (Poly)
EEG Motor/Imagery Dataset			
8	FV	2.84 sec (RBF)	14.91 sec (RBF)
	HFV	5.8 sec (RBF)	28.18 sec (RBF)
17	FV	3.37 sec (RBF)	53.51 sec (RBF)
	HFV	9.37 sec (RBF)	53.94 sec (Poly)
EEG SSVEP Dataset III			
14	FV	0.49 sec (RBF)	2.79 sec (Poly)
	HFV	0.79 sec (RBF)	4.38 sec (Poly)

TABLE 4.11. Impact of perceptual hashing on SVM training time

Overall, it can be seen that perceptually hashing the feature vectors increases the classifier training time for SVM, but the training time of k-NN is unaffected. Although the training time increases (for SVM), as will be seen in the following section, this increase does not impact our overall system as the training phase of the classifier completes before the user enrolls with the IDP.

4.3.3. Impact of Perceptual Hashing on Protocol Execution and Bandwidth Consumption

In this section, we study the impact of perceptual hashing on the protocol execution time and the amount of data transferred. We look at these for both models in our approach viz., IDP-centric model and User-centric model (see Sections 3.8.1 and 3.8.2). In both the models, the enrollment phase is like a set-up phase which takes place before the authentication phase and hence, we do not discuss the execution time or data transferred during that phase. The execution times are in milliseconds (ms) and data transferred in Mega Bytes (MB).

Table 4.12 summarizes the execution times for and data transferred during the authentication protocol 4 in the IDP-centric authentication model. When we train both the classifiers, SVM and k-NN, on features extracted using PSPRT, the execution time varies between 146 ms and 213 ms on the EEG Alcohol dataset. Similarly, it ranges from 149 ms and 178 ms for the

EEG Motor/Imagery dataset and between 151 ms and 156 ms on the EEG SSVEP Dataset III. These numbers are very close. Also, irrespective of whether we train the classifiers directly on the features (FV), their perceptual hashes (HFV) or the number of EEG channels we consider, the protocol execution time is more or less stable. We have the same observation when features are extracted using PSD. The execution time varies between 175 ms and 242 ms for the EEG Alcohol dataset and 172 ms, 223 ms for the EEG Motor/Imagery dataset, and between 166 ms and 200 ms for the EEG SSVEP Dataset III. Thus, the execution time of the authentication protocol in this model is not affected by perceptual hashing.

Ch.	Classifier	Features	PSPRT		PSD	
			Time (ms)	Data (MB)	Time (ms)	Data (MB)
EEG Alcohol Dataset						
8	SVM	FV	150	11.15	242	73.42
		HFV	160	11.24	195	73.5
	k-NN	FV	146	11.04	186	73.51
		HFV	153	11.06	175	73.73
17	SVM	FV	167	21.85	190	288.70
		HFV	213	21.71	190	288.69
	k-NN	FV	163	21.75	189	288.73
		HFV	164	21.71	188	288.67
EEG Motor/Imagery Dataset						
8	SVM	FV	149	9.84	206	72.02
		HFV	153	9.96	227	72.03
	k-NN	FV	156	9.91	172	72.24
		HFV	157	9.76	175	72.03
17	SVM	FV	169	20.33	176	287.28
		HFV	172	20.42	218	287.27
	k-NN	FV	163	20.41	183	287.32
		HFV	178	20.48	223	287.40
EEG SSVEP Dataset III						
14	SVM	FV	156	15.74	200	199.84
		HFV	154	15.78	180	199.91
	k-NN	FV	151	15.75	166	199.81
		HFV	148	15.83	172	199.79

TABLE 4.12. Impact of perceptual hashing on protocol execution time and data transferred during the authentication phase (IDP-centric Model)

Now, let us look at the impact of perceptual hashing on the amount of data transferred during the authentication protocol in the IDP-centric model. Following are some important observations regarding the data transferred from Table 4.12:

(1) *The amount of data transferred changes as the number of EEG channels change*. The user sends the perceptual hash of the EEG signal to the IDP to get the token. Now, as the EEG channels increase so do the features that are sent to the IDP, and this explains why there is an increase in data transferred. On the EEG Alcohol dataset, when we extract features using PSPRT, about 11 MB of data is transferred for data from 8 EEG channels. However, this changes to around 22 MB when we consider data from 17 channels. Similarly, on the EEG Motor/Imagery dataset, data transferred is, approximately, 10 MB for 8 EEG channels and 20 MB for 17 channels. On the EEG SSVEP Dataset III, the data transferred is around 16 MB for 14 EEG channels which is between the data transferred for 8 EEG channels and 17 EEG channels. We have similar observations when PSD is used for feature extraction.

(2) *The amount of data transferred changes as the feature extraction technique changes*. On the EEG Alcohol dataset, using PSPRT for feature extraction, data transfer of around 11 MB takes place when we consider 8 EEG channels. However, that changes to about 73 MB when we use PSD for feature extraction for the same number of channels. This increase is because PSPRT results in just 36 features for 8 EEG channels where are PSD results in 720 features. We have a similar observation when we consider data from 17 EEG channels, i.e., approximately 22 MB (for PSPRT) and 289 MB (for PSD) for data is transferred. We have a similar observation on the EEG Motor/Imagery dataset.

(3) *Perceptual hashing a feature vector should reduce the amount of data transferred*: Perceptual hashing a feature vector transforms it into a binary sequence (see Algorithm 1). Thus, perceptually hashing a feature vector should reduce the amount of data transferred. However, irrespective of whether the feature vectors are perceptually hashed or not, the amount of data transferred is similar as long as we use the same number of channels and employ the same feature extraction technique. The reason for this is that we leverage

existing classes in Weka [26] to represent a feature vector or an instance. Weka considers any feature that has numeric value to be of type numeric. After feature extraction, the feature vectors have real values and perceptually hashing these feature vectors converts them into binary. But, these are still numbers and are considered to be of type numeric by the Weka classes. Having a custom implementation to represent the perceptually hashed feature vectors will certainly reduce the amount of data transferred.

Ch.	Features	PSPRT		PSD	
		Time (ms)	Data (MB)	Time (ms)	Data (MB)
EEG Alcohol Dataset					
8	FV	127	3.20	185	3.35
	HFV	125	3.09	130	3.13
17	FV	112	3.12	144	3.14
	HFV	167	3.17	139	3.33
EEG Motor/Imagery Dataset					
8	FV	113	3.11	168	3.09
	HFV	124	3.22	204	3.32
17	FV	139	3.16	132	3.20
	HFV	137	3.19	125	3.23
EEG SSVEP Dataset III					
14	FV	104	3.11	122	3.17
	HFV	116	3.1	127	3.12

TABLE 4.13. Impact of perceptual hashing on protocol execution time and data transferred during the authentication phase (User-centric Model)

Next, we discuss the execution time for and data transferred during the authentication protocol in the User-centric authentication model (see Table 4.13). These details are for the SVM classifier as we can only employ SVM and not k-NN in this model (see section 3.8.2). Even in this model, whether the feature vectors are perceptually hashed or not, does not affect the protocol execution time. Additionally, neither the feature extraction technique nor the number of EEG channels we consider has an impact on the execution time. Also, the execution time for the authentication protocol in this model is consistently lower than the execution time in the IDP-centric model. Similarly, the data transferred in this model is also lower than the data transferred in the IDP-centric model. We expected both these things because, unlike in the IDP-centric model, the

client authentication application generates the token required for authentication without the intervention of the IDP. So, the user does not need to send the perceptual hash to the IDP, which reduces the protocol execution time and amount of data transferred (see Protocol 6).

Thus, based on our experiments, it can be seen that perceptual hashing does not affect the protocol execution of or the data transferred during the authentication phase, and a better implementation of representing the perceptual hash can certainly reduce the bytes transferred during the protocol execution.

CHAPTER 5

SECURITY ANALYSIS

5.1. Threat Model

We consider the semi-honest or honest-but-curious threat model. In this threat model, the involved parties follow the established protocol but may try to learn from the data. Ex. The IDP is a trusted entity and will never deviate from the protocol. However, given the raw EEG signals, it might want to learn some private information about the user.

5.2. Privacy of EEG Signals

Since we adopt the authentication model proposed in [24], we also inherit its benefits like confidentiality of the perceptual hash, BID and $IDToken$, repeatability of BID and revocability of $IDToken$. We will discuss these and a few others in Sections 5.3, 5.4, and 5.5. However, we are also concerned about the privacy of the EEG signal and the user. In our approach, the IDP receives only the perceptual hash and not the EEG signal or the feature vector. This way we ensure the confidentiality of the EEG signals and its feature vectors. However, the perceptual hash can also leak private information. Now, each sample in the perceptual hash is either 1 or 0 and is based on the median of the feature vector. After analyzing the perceptual hashes for the proportion of 1s and 0s, we observe that the perceptual hash of each EEG signal has about 50% of 1s and 0s. Thus, the perceptual hash can be considered as any random binary sequence. Moreover, the feature extraction technique or the number of segments (in our case a single segment or two segments) we consider to compute the perceptual hash does not affect the proportion of 1s and 0s in the hash. Thus, we believe that the perceptual hash will prevent information leakage.

5.3. Confidentiality of Sensitive Information

The authentication system involves other sensitive information like Perceptual Hashes, BID, and the IDToken. We use the perceptual hash to predict the class label and then use this to generate the BID. Once we have the BID, the hash is no longer required and hence, we discard it. Similarly, BID is only required to generate the IDToken, and after we have the token, we even

discard the BID. This way we maintain the confidentiality of the perceptual hash and BID. We derive the secrets required for generating the BID and the Pedersen commitment on-the-fly using the user's password and salt. Thus, as long as the password is kept secret, these secrets remain hidden. Additionally, the use of the ZKPK protocol also helps keep the secrets hidden during the authentication phase. Moreover, a replay attack on the token does not affect the system as new values of y and t are generated to complete the proof of knowledge (see Section 3.7.2).

In the IDP-centric model, as we also employ the $k-$NN classifier to predict the class label, we need to store the perceptual hashes in a database. The storage of perceptual hashes poses a risk of permanent loss of the user's biometric identity in case an attacker can compromise the database. However, in that event, only the perceptual hash of the features extracted from an EEG signal will be lost. Since this hash is not invertible, the feature vector and the original EEG signals remain confidential. Additionally, the loss of identity can be overcome changing the algorithm to compute the perceptual hash. Furthermore, for the authentication to be successful, an impostor will also need to know the password. Nevertheless, this is an undesirable situation as it involves a compromise of the perceptual hash and can be considered to be a weakness in this model. Another flaw in this model is that each time the user wants to access a server it needs the IDToken from the IDP to complete the authentication phase. Thus, the IDP gets an opportunity to learn about the user from the transaction pattern. We overcome these weaknesses in the User-centric model.

In the User-centric model, we do not employ the k-NN classifier as we need to send a customized model to the user. k-NN is an instance-based classifier and requires the other instances or data records to predict a class label of an unlabeled record. Sending the other instances to the user is not feasible due to extremely high security and privacy issues and limited computing resources available to the user. Since the user has a customized model and the client authentication application, it does not need the IDP to generate the IDToken. This way the IDP does not have access to the user's transaction pattern and cannot learn anything from them.

5.4. Repeatability of BID and Revocability of IDToken

The BID is generated using the predicted class and the secret derived from the user's password. Thus, as long as the predicted class and the user's password are correct, the same BID will

be generated i.e., the BID will be repeatable.

In the case an attacker compromises the user's IDToken, the user can simply have the IDToken revoked by informing the appropriate Server. Then, with the help of the IDP, he or she can have a new IDT issued by generating a new BID and choosing a new password.

5.5. Protection against Malicious Users and Servers

An external attacker or a malicious Server may try to perform a man-in-the-middle attack to impersonate the user. In the former case, the attack can be prevented by making use of secure communication channels. However, secure communication channels are ineffective when a Server is malicious and is trying to impersonate the user by attacking the zero knowledge proof of the user's identity commitment. This is also known as a Mafia Fraud attack, i.e., where the prover is honest but the verifier is malicious. To prevent this, the user can create a commitment on the Server's identity and have the IDP include this commitment in the to field of the IDToken. The advantage of this is twofold. First, only the user can successfully open the commitment as only he or she has the required secrets. Second, the IDP is unaware of which Server the user is communicating with and will be unable to learn anything about the user's usage pattern. Moreover, the IDToken has an expires field and is signed by the IDP. A Server first verifies these to ascertain the validity of the token and only after successful validation proceeds with the ZKPK protocol.

As we already know that in the User-centric model, a customized SVM model is sent to the user, i.e., original labels are replaced by different random numbers for different users. Although this model will not be directly available to the users, a malicious user might be successful in finding it. He or she may then analyze the model and try to learn information about the other class labels, which can compromise the privacy of the other users of the IDP. However, as this model is customized, that attacker will not learn any information about the other users.

CHAPTER 6

CONCLUSION AND FUTURE WORK

In this work, we have shown that it is possible to apply perceptual hashing to feature vectors extracted from EEG signals and use them to perform authentication. We can achieve this without having a significant loss in the accuracy, and by considering data from more EEG channels, we can further reduce this loss. Additionally, the use of perceptual hashing also maintains the confidentiality of EEG signals and the privacy of the user. Additionally, the performance on the two types of perceptual hashes that we computed (HFV100 and HFV50) had more or less similar performance. We also observed that perceptual hashing does increase the classifier time, but it is not a significant increase and is a one-time activity that does not affect the authentication protocol execution time. Since the perceptual hash is a binary sequence, one can expect a reduction in the amount of data transferred during the enrollment and authentication phases. We also incorporated an approach to perform authentication in a privacy-preserving manner using a biometric identifier that is repeatable, revocable, and unique to the user based on his or her EEG signal.

In the future, we would like to work on improving the accuracy of the system with a lesser number of EEG channels while employing the current feature extraction techniques. We would also like to explore other feature extraction and machine learning techniques to see if we can achieve a better accuracy than our present system. Additionally, we would like to explore other perceptual hashing techniques that are not only secure but also improve the accuracy of the system. Another appealing area for future research is eliminating the need for a password to generate the BID. Instead, one should be able to compute the BID solely from the EEG signals without affecting privacy. Finally, we would like to explore other techniques for generating BIDs while meeting the requirements of repeatability, revocability, and uniqueness.

APPENDIX

SUBJECTS CHOSEN FROM DATASETS FOR EXPERIMENTS

Tables A.1 and A.2 indicate the subjects chosen from the EEG Alcohol and EEG Motor/Imagery datasets, respectively, for our experiments. Please note that these subjects were randomly chosen from the datasets.

co2a0000364	co2a0000365	co2a0000368	co2a0000369	co2a0000370
co2a0000377	co2a0000379	co2a0000384	co2a0000388	co2a0000390
co2a0000394	co2a0000395	co2a0000396	co2a0000402	co2a0000403
co2a0000404	co2a0000407	co2a0000409	co2a0000410	co2a0000414
co2a0000418	co2a0000421	co2a0000433	co2a0000437	co2a0000438
co2a0000439	co2a0000440	co2a0000443	co2a0000445	co2a0000447
co2c0000337	co2c0000338	co2c0000339	co2c0000340	co2c0000341
co2c0000342	co2c0000344	co2c0000345	co2c0000346	co2c0000352
co2c0000364	co2c0000371	co2c0000357	co2c0000378	co2c0000392
co2c0000381	co2c0000383	co2c0000359	co2c0000387	co3a0000448
co2c0000389	co2a0000430	co2a0000428	co2a0000382	co2a0000424
co2c0000394	co2a0000426	co2c0000363	co2c0000397	co2a0000398
co2a0000387	co2c0000367	co2a0000432	co3a0000454	co3a0000453
co2a0000392	co3a0000457	co3a0000458	co3a0000460	co2a0000400

TABLE A.1. Subjects chosen from EEG Alcohol dataset

S004	S011	S013	S019	S021	S022	S024	S026	S027
S028	S030	S031	S033	S038	S043	S044	S050	S034
S054	S063	S065	S069	S072	S086	S087	S089	S088

TABLE A.2. Subjects chosen from EEG Motor/Imagery dataset

BIBLIOGRAPHY

[1] *Signal reconstruction from continuous wavelet transform coefficients*, (accessed August 4, 2016), http://www.mathworks.com/help/wavelet/examples/signal-reconstruction-from-continuous-wavelet-transform-coefficients.html.

[2] *Mamem: Multimedia authoring & management using your eyes & mind*, (accessed July 28, 2016), http://www.mamem.eu/results/datasets/.

[3] *False rejection rate*, (accessed May 18, 2016), http://www.webopedia.com/TERM/F/false_rejection.html.

[4] *Tpr, fpr, precision, recall, f1 score*, (accessed May 18, 2016), https://weka.wikispaces.com/Primer.

[5] *64 electrodes as per the international 10-10 system [online image]*, (accessed May 28, 2016), https://physionet.org/pn4/eegmmidb/64_channel_sharbrough.png.

[6] *Emotiv*, (accessed May 28, 2016), http://emotiv.com/.

[7] *Emotiv epoc+ headset [online image]*, (accessed May 28, 2016), http://emotiv.com/wp-content/uploads/2016/02/emotiv_epoc_600.png.

[8] *Full 64 electrodes setup [online image]*, (accessed May 28, 2016), https://physionet.org/pn4/eegmmidb/64_channel_sharbrough.png.

[9] *The geodesic sensor net [online image]*, (accessed May 28, 2016), https://www.egi.com/images/stories/clinical/images/products/Wearing_GSN_grey_bg_210x210.jpg.

[10] *Neurosky*, (accessed May 28, 2016), http://neurosky.com/.

[11] *Neurosky mindwave [online image]*, (accessed May 28, 2016), http://neurosky.com/wp-content/uploads/2014/11/mindwave-mobile-headset1-300x300.jpg.

[12] *www.bci2000.org*, (accessed May 28, 2016), www.bci2000.org.

[13] Muhammad Kamil Abdullah, Khazaimatol S Subari, Justin Leo Cheang Loong, and Nu-

rul Nadia Ahmad, *Analysis of the eeg signal for a practical biometric system*, World Academy of Science, Engineering and Technology 68 (2010), 1123–1127.

[14] Corey Ashby, Amit Bhatia, Francesco Tenore, and Jacob Vogelstein, *Low-cost electroencephalogram (eeg) based authentication*, Neural Engineering (NER), 2011 5th International IEEE/EMBS Conference on, IEEE, 2011, pp. 442–445.

[15] Garima Bajwa and Ram Dantu, *Neurokey: towards a new paradigm of cancelable biometrics-based key generation using electroencephalograms*, Computers & Security (2016).

[16] Vangie Beal, *False acceptance rate*, (accessed May 18, 2016), `http://www.webopedia.com/TERM/F/false_acceptance.html`.

[17] Henri Begleiter, Neurodynamics Laboratory, State University of New York Health Center, Brooklyn, New York, 1999.

[18] Abhilasha Bhargav-Spantzel Elisa Bertino, Anna Squicciarini Xiangwei Kong, and Weike Zhang, *Biometrics-based identifiers for digital identity management*.

[19] E Oran Brigham and Elbert Oran Brigham, *The fast fourier transform*, vol. 7, Prentice-Hall Englewood Cliffs, NJ, 1974.

[20] Christopher JC Burges, John C Platt, and Soumya Jana, *Distortion discriminant analysis for audio fingerprinting*, IEEE Transactions on Speech and Audio Processing 11 (2003), no. 3, 165–174.

[21] Bambi L DeLaRosa, Jeffrey S Spence, Scott KM Shakal, Michael A Motes, Clifford S Calley, Virginia I Calley, John Hart, and Michael A Kraut, *Electrophysiological spatiotemporal dynamics during implicit visual threat processing*, Brain and cognition 91 (2014), 54–61.

[22] Mario Frank, Tiffany Hwu, Sakshi Jain, Robert Knight, Ivan Martinovic, Prateek Mittal, Daniele Perito, and Dawn Song, *Subliminal probing for private information via eeg-based bci devices*, arXiv preprint arXiv:1312.6052 (2013).

[23] Thomas Gruber, Matthias M Müller, and Andreas Keil, *Modulation of induced gamma band responses in a perceptual learning task in the human eeg*, Journal of cognitive neuroscience 14 (2002), no. 5, 732–744.

[24] Hasini Gunasinghe and Elisa Bertino, *Privacy preserving biometrics-based and user centric authentication protocol*, Network and System Security, Springer, 2014, pp. 389–408.

[25] Jaap Haitsma, Ton Kalker, and Job Oostveen, *Robust audio hashing for content identification*, International Workshop on Content-Based Multimedia Indexing, vol. 4, Citeseer, 2001, pp. 117–124.

[26] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten, *The weka data mining software: an update*, ACM SIGKDD explorations newsletter 11 (2009), no. 1, 10–18.

[27] E. Klinger and D. Starkweather, *phash.org: Home of phash, the open source perceptual hash library (2008-2010)*, (accessed May 18, 2016), `http://www.phash.org/`.

[28] Juris Klonovs, Christoffer Kjeldgaard Petersen, Henning Olesen, and Allan Hammershoj, *Id proof on the go: Development of a mobile eeg-based biometric authentication system*, IEEE Vehicular Technology Magazine 8 (2013), no. 1, 81–89.

[29] Yin-Hsi Kuo, Kuan-Ting Chen, Chien-Hsing Chiang, and Winston H Hsu, *Query expansion for hash-based image object retrieval*, Proceedings of the 17th ACM international conference on Multimedia, ACM, 2009, pp. 65–74.

[30] Frank Kurth and Roman Scherzer, *Robust real-time identification of pcm audio sources*, Audio Engineering Society Convention 114, Audio Engineering Society, 2003.

[31] M. Lichman, *UCI machine learning repository*, 2013 (accessed May 18, 2016), `http://archive.ics.uci.edu/ml`.

[32] Chun-Shien Lu and Chao-Yong Hsu, *Geometric distortion-resilient image hashing scheme and its applications on copy detection and authentication*, Multimedia Systems 11 (2005), no. 2, 159–173.

[33] Werner Lutzenberger, Friedemann Pulvermüller, Thomas Elbert, and Niels Birbaumer, *Visual stimulation alters local 40-hz responses in humans: an eeg-study*, Neuroscience letters 183 (1995), no. 1, 39–42.

[34] Sebastien Marcel and José del R Millán, *Person authentication using brainwaves (eeg) and*

maximum a posteriori model adaptation, IEEE transactions on pattern analysis and machine intelligence 29 (2007), no. 4, 743–752.

[35] Ivan Martinovic, Doug Davies, Mario Frank, Daniele Perito, Tomas Ros, and Dawn Song, *On the feasibility of side-channel attacks with brain-computer interfaces*, Presented as part of the 21st USENIX Security Symposium (USENIX Security 12), 2012, pp. 143–158.

[36] MATLAB, *Release 2015a*, The MathWorks Inc., Natick, Massachusetts, United States, 2015.

[37] M Kivanç Mıhçak and Ramarathnam Venkatesan, *A perceptual audio hashing algorithm: a tool for robust audio identification and information hiding*, International Workshop on Information Hiding, Springer, 2001, pp. 51–65.

[38] Tal Moran, *The qilin crypto sdk: An open-source java sdk for rapid prototyping of cryptographic protocols*, GitHub, 2015 (accessed May 18, 2016, `https://github.com/factcenter/qilin`.

[39] Matthias M Müller, Thomas Gruber, and Andreas Keil, *Modulation of induced gamma band activity in the human eeg by attention and visual information processing*, International Journal of Psychophysiology 38 (2000), no. 3, 283–299.

[40] Isao Nakanishi, Sadanao Baba, and Chisei Miyamoto, *Eeg based biometric authentication using new spectral features*, Intelligent Signal Processing and Communication Systems, 2009. ISPACS 2009. International Symposium on, IEEE, 2009, pp. 651–654.

[41] Arne Öhman, *The role of the amygdala in human fear: automatic detection of threat*, Psychoneuroendocrinology 30 (2005), no. 10, 953–958.

[42] Job C Oostveen, Ton Kalker, and Jaap Haitsma, *Visual hashing of digital video: applications and techniques*, International Symposium on Optical Science and Technology, International Society for Optics and Photonics, 2001, pp. 121–131.

[43] Junlin Ouyang, Gouenou Coatrieux, and Huazhong Shu, *Robust hashing for image authentication using quaternion discrete fourier transform and log-polar transform*, Digital Signal Processing 41 (2015), 98–109.

[44] Hamza Özer, Bülent Sankur, Nasir Memon, and Emin Anarım, *Perceptual audio hashing functions*, EURASIP Journal on Advances in Signal Processing 2005 (2005), no. 12, 1–14.

[45] R Palaniappan and KVR Ravi, *A new method to identify individuals using signals from the brain*, Information, Communications and Signal Processing, 2003 and Fourth Pacific Rim Conference on Multimedia. Proceedings of the 2003 Joint Conference of the Fourth International Conference on, vol. 3, IEEE, 2003, pp. 1442–1445.

[46] Ramaswamy Palaniappan, Jenish Gosalia, Kenneth Revett, and Andrews Samraj, *Pin generation using single channel eeg biometric*, International Conference on Advances in Computing and Communications, Springer, 2011, pp. 378–385.

[47] Ramaswamy Palaniappan and KVR Ravi, *Improving visual evoked potential feature classification for person recognition using pca and normalization*, Pattern Recognition Letters 27 (2006), no. 7, 726–733.

[48] RB Paranjape, J Mahovsky, L Benedicenti, and Z Koles, *The electroencephalogram as a biometric*, Electrical and Computer Engineering, 2001. Canadian Conference on, vol. 2, IEEE, 2001, pp. 1363–1366.

[49] Torben Pryds Pedersen, *Non-interactive and information-theoretic secure verifiable secret sharing*, Advances in CryptologyCRYPTO91, Springer, 1991, pp. 129–140.

[50] PhysioToolkit PhysioBank, *Physionet: components of a new research resource for complex physiologic signals*, Circulation. v101 i23. e215-e220.

[51] M Poulos, M Rangoussi, V Chrissikopoulos, and A Evangelou, *Person identification based on parametric processing of the eeg*, Electronics, Circuits and Systems, 1999. Proceedings of ICECS'99. The 6th IEEE International Conference on, vol. 1, IEEE, 1999, pp. 283–286.

[52] Nalini K. Ratha, Jonathan H. Connell, and Ruud M. Bolle, *Enhancing security and privacy in biometrics-based authentication systems*, IBM systems Journal 40 (2001), no. 3, 614–634.

[53] Kenneth Revett, Farzin Deravi, and Konstantinos Sirlantzis, *Biosignals for user authentication-towards cognitive biometrics?*, Emerging Security Technologies (EST), 2010 International Conference on, IEEE, 2010, pp. 71–76.

[54] Gerwin Schalk, Dennis J McFarland, Thilo Hinterberger, Niels Birbaumer, and Jonathan R Wolpaw, *Bci2000: a general-purpose brain-computer interface (bci) system*, Biomedical Engineering, IEEE Transactions on 51 (2004), no. 6, 1034–1043.

[55] Bruce Schneier, *Applied cryptography: protocols*, Algorithms, and Source Code in C 2 (1996), 216–222.

[56] R Sivakumar and G Ravindran, *Identification of intermediate latencies in transient visual evoked potentials*, Academic Open Internet J 17 (2006).

[57] Joan G Snodgrass and Mary Vanderwart, *A standardized set of 260 pictures: norms for name agreement, image agreement, familiarity, and visual complexity.*, Journal of experimental psychology: Human learning and memory 6 (1980), no. 2, 174.

[58] Somsak Sukittanon and Les E Atlas, *Modulation frequency features for audio fingerprinting*, Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on, vol. 2, IEEE, 2002, pp. II–1773.

[59] Catherine Tallon-Baudry, Olivier Bertrand, Franck Peronnet, and Jacques Pernier, *Induced γ-band activity during the delay of a visual short-term memory task in humans*, The Journal of neuroscience 18 (1998), no. 11, 4244–4254.

[60] Julie Thorpe, Paul C van Oorschot, and Anil Somayaji, *Pass-thoughts: authenticating with our minds*, Proceedings of the 2005 workshop on New security paradigms, ACM, 2005, pp. 45–56.

[61] Christopher Torrence and Gilbert P Compo, *A practical guide to wavelet analysis*, Bulletin of the American Meteorological society 79 (1998), no. 1, 61–78.

[62] Seul-Ki Yeom, Heung-Il Suk, and Seong-Whan Lee, *Person authentication from neural activity of face-specific visual self-representation*, Pattern Recognition 46 (2013), no. 4, 1159–1169.

[63] Qinglin Zhao, Hong Peng, Bin Hu, Quanying Liu, Li Liu, YanBing Qi, and Lanlan Li, *Improving individual identification in security check with an eeg based biometric solution*, International Conference on Brain Informatics, Springer, 2010, pp. 145–155.

[64] Xuebing Zhou, Martin Schmucker, and Christopher L Brown, *Perceptual hashing of video content based on differential block similarity*, International Conference on Computational and Information Science, Springer, 2005, pp. 80–85.

[65] André Zúquete, Bruno Quintela, and João Paulo da Silva Cunha, *Biometric authentication using brain responses to visual stimuli.*, BIOSIGNALS, 2010, pp. 103–112.