User Authentication Using Human Cognitive Abilities

Asadullah Al Galib and Reihaneh Safavi-Naini

University of Calgary
{aagalib, rei}@ucalgary.ca

Abstract. We present a novel approach to user authentication in which biometric data related to human cognitive processes, in particular visual search, working memory and priming effect on automatic processing, are captured and used to identify users. Our proposed system uses a carefully designed Cognitive Task (CT) that is presented to the user as a game, in order to capture a "cognitive signature" of the user. Our empirical results support the hypothesis that the captured cognitive signatures can identify users across different platforms. Our system provides a proof-of-concept for cognitive-based biometric authentication. We validate the robustness of our system against impersonation attack by experienced users, and show that it is hard to reproduce the cognitive signature by mimicking users' gameplay.

1 Introduction

The most widely used form of authentication is password system; that is, what we can remember. Password systems are attractive because they do not require any special hardware, but they are vulnerable to guessing attacks and passwords have the risk of being forgotten. Biometric authentication systems are based on *what we are* (fingerprints, iris pattern), and are immune to being lost or forgotten. However, traditional biometric systems require the use of special hardwares such as scanners or cameras. More recent biometric authentication systems are behavioral and based on *what we do*, including keyboard typing rhythm, mouse dynamics or walking gait. Behavioral biometric systems measure behavioral traits of a user to build a profile for him that will later be used to identify the user.

We present an authentication system which captures biometric data related to human cognitive processes and use that to build a profile for the user. Cognition refers to higher level brain functions (or mental processes) such as perception, learning, problem solving [1, 2]. Cognitive abilities of individuals are their capacities to carry out cognitive tasks that require mental processes. Basing authentication on these processes makes our approach different from behavioral biometrics which do not attempt to invoke particular mental processes.

Our work is inspired by the reported studies on individuals differences [1] in performing cognitive tasks. We present a *cognitive task* (CT) to the user in the form of a game that will be performed by the user by interacting with

the computer, and using a mouse or a touchpad. The collected data during the execution of the CT will be used to extract cognitive features related to visual search ability, working memory and the effect of priming on automatic processing of the user. Visual search refers to finding a target object in a set of objects and is measured by the search time. Working memory allows individuals to hold information in their memory for later processing. Features derived from these cognitive processes in combination with other basic stimulus-response features are used to build profiles for the users that can later identify them.

The CT is presented to the user in the form of an interactive visual search game. The game starts by presenting a set of 25 different objects arranged in a 5×5 grid to the user. The task of the user is to find a particular target object in the set. The user has to drag and drop the challenge object onto the matching target object in the set. This is equivalent to performing a visual search task by the user. On performing a correct search task (or correct match), the user is rewarded with a gold coin. The user is instructed to deposit the gold coin in a "bank". On a correct deposit, the user is presented with another challenge object and a similar interaction follows. The features derived from these interactions are used to construct a cognitive signature. The signatures are then used to develop an authentication system with accuracy comparable to other established biometric approaches. A typical behavioral biometric system such as those based on mouse dynamics, measure behavioral traits that are inadvertent. Systems designed to estimate cognitive features such as ours, can be augmented to use behavioral features related to mouse dynamics to improve authentication accuracy.

Our system is based on experiments in experimental cognitive psychology and has been carefully designed to preserve the essential elements of the corresponding experiments. Attempts have also been made to ensure that the cognitive task presented to the user is intuitive and interesting. We performed experiments in controlled and non-controlled (Amazon Mechanical Turk [3]) environments. The accuracies obtained in both cases are comparable to other state-of-the-art behavioural biometric systems [4–7]. To evaluate security of the system we considered impersonation attack where an attacker attempts to mimic a target user's gameplay. Using the data collected during the target user's gameplay, we developed a simulation of their gameplay that was later provided to the attacker for the purpose of training. After that training phase, the attacker had to impersonate the target user. We considered the attack to be successful when the attacker was able to successfully authenticate himself as the target user.

Section 2 discusses the mental processes and the design of the game. Section 3 describes how the design invokes cognitive processes. We discuss feature collection and the user classification technique in Section 4 and 5, respectively. Section 6 provides details on experimental results and analysis. Finally, we conclude in Section 7.

2 Cognitive Task

We present the CT to the user in the form of a web-based game. In this Section, we first discuss cognitive processes and then the design of the CT(game). In Section 3 we examine how the design of the game invokes these mental processes.

2.1 Mental Processes

Visual Search. Visual search is a type of cognitive task in which the user searches through a visual field for a target [8]. Performance is generally measured by the search time. The search time depends on multiple factors such as the rate at which the user scans the alternatives. In a self-terminating search, the user stops the searching process as soon as he finds the alternative he thinks is appropriate [9, 2].

Working Memory & Information Processing Speed. Working memory describes the ability of a human to hold and manipulate information in their mind over short periods of time for a cognitive task such as learning or reasoning [8]. The working memory capacity varies between individuals. The ability to reason and solve problems requires the use of information stored in working memory. However, this information is vulnerable to interruption and decay. Due to this volatility, faster processing of this information is necessary for successful completion of a cognitive task.

Automatic Processing & Priming Effect. Automatic processing, is the processing of information that guides behavior, but without being conscious of the process, and without interfering with other conscious activity that may be underway at the same time [2]. Automatic processes can be invoked by a technique called priming [2, 10]. A prime is a stimulus or event that influences an ongoing action or process. Bargh et al. [10] carried out an experiment where a group of participants were exposed to words related to the concept of elderly. The participants who were primed with the elderly concept were found walking slower than the others. However, participants had no conscious awareness of the concept of the elderly or of their reaction to it.

2.2 Design of the Game.

Our game provides a simple challenge-response task. In each *instance* of the challenge-response, the user is given a challenge, which is an object. The user responds by dragging the challenge object onto the matching object inside the search set. The user then receives a gold coin as a reward and deposits it in a bank. On a correct deposit, the user is challenged with a new object and the game continues as before. Our goal was to invoke the three mental processes within a minimal design space.

An image is first broken into a grid of $5 \times 5 = 25$ square cells. We refer to this partitioned image as the search set θ, $|\theta| = 25$. Each square is called a *tile*. The game starts by presenting a random challenge tile t_c at a location P_{t_c} outside the partitioned image. The tile, t_c, is a copy of a tile $t_r \in \theta$. We have divided

the user's response into two actions. (1) The user drags t_c and drops it onto t_r located at position P_{t_r} within the search set, in which case t_c rests on t_r and becomes *unmovable*. We refer to this action as A_{resp}. On an incorrect match, t_c automatically moves back to position P_{t_c} signifying a *mismatch* and allowing the user to retry. (2) On a correct placement of t_c on t_r, the user is rewarded with a gold coin, g_c, which appears exactly at P_{t_r} (superimposed). The user is then instructed to deposit (*drag* and *drop*) g_c in a bank (a bounded box with the same dimensions as that of a tile) appearing at P_{t_c}. We refer to this action as A_{rew}. On depositing the coin, the user is challenged with the next tile, and the game continues as before. Therefore, one *instance* of this game is comprised of the correct placement of the target tile, A_{resp} and correct deposit of the gold coin, A_{rew} (Appendix A, Figure 4).

From Conceptual Modeling to Implementation. In order to guarantee the invocation of the aforementioned mental processes certain constraints have been used throughout the game.

C1: At the beginning of each *instance*, a copy of a randomly chosen original tile $t_r \in \theta$ appears at P_{t_c} as the challenge tile t_c. Challenge tiles cannot be repeated. Therefore, each of the 25 partitioned portions (original tiles) of the image must appear only once as the challenge tile.

C2: On completing the action A_{resp}, t_c is superimposed on t_r and becomes *unmovable*. At this point all the *loose* tiles (tiles that have not appeared in the challenge phase till now) disappear from the grid leaving only the *unmovable* ones. This allows the user to observe the current status of the game. The user can observe the tiles that have been placed correctly till now and the remaining empty square cells on the grid. Refer to Appendix A Figure 4(b).

C3: At the beginning of A_{resp} all current *loose* tiles in the grid are shuffled. They randomly change their positions on the grid except for the target tile t_r and the *unmovable* ones. All 25 tiles are visible during A_{resp}. Therefore the actual positions of the *loose* tiles remain unknown to the user. Refer to Appendix A Figure 4(d).

C4: Two straight lines from P_{t_r} to P_{t_c} appear during the A_{rew} action. During this action if the gold coin touches any one of the straight lines, its color changes from green to red, without hampering the current movement (Appendix A Figure 4(c)).

C5: The tiles consist of random-shaped black symbols on a white background. All tiles have the same opacity throughout the game. The symbols being of random shapes do not *necessarily* represent or convey any meaning to the user.

C6: We allow some tolerance on the placement of the tile and the gold. This means that the user does not need perfect accuracy when dropping t_c onto t_r or when depositing g_c at P_{t_c}. A *drop* is considered a match if t_c or g_c covers 60% of the area of the underlying tile t_r or bank, respectively. On releasing they are automatically superimposed over their destinations. However, there are exceptions, on K random *instances* the *drop* accuracy is increased for g_c *only*, requiring 90% overlapping area. We choose $K = k_1 \dots k_5$ randomly from consecutive *instance* intervals $\{i_1 \dots i_5\}$, $\{i_6 \dots i_{10}\}$, $\dots$, $\{i_{21} \dots i_{25}\}$. Each time

the *drop* accuracy is not met, g_c automatically moves back to P_{t_r}. The user is then allowed to re-try.
C7: Each time there is a *mismatch* t_c moves back to its original location P_{t_c}. The user is then allowed to re-try. However, as soon as the user hovers over t_c, the grid is again shuffled according to Constraint $C3$.

3 The CT Constraints & Mental Processes

Here, we illustrate the importance of the aforementioned constraints and how they aid in triggering the mental processes.

3.1 Revisiting Visual Search

Our game (CT) has been designed to invoke self-terminating searches. As soon as the user finds the target tile, further searches are not required. Due to Constraint *C1*, we refer to our search set as a (+ve) search set, meaning that the target must appear within the set. This ensures that a match always occurs and the search terminates. Constraint *C3* aids in invoking serial search and *C5* reduces the conspicuity of the target. If a target location is known beforehand and if a target is too conspicuous they can affect the search process [8]. If the grid was not shuffled at each instance according to *C3 & C7*, the user may remember certain target positions. This bias the visual search process.

3.2 Revisiting Working Memory & Information Processing Speed

Due to Constraint $C2$, the user can observe the current game status. The user can observe the empty grid cells and the already placed tiles. He can hold this information in his working memory for a short interval of time while he completes action A_{rew}. If the information is not lost, he will be able to decrease the size of the search set $|\theta|$ for the next challenge. For example, for the 11th *instance* of the game he will be able to shrink $|\theta|$ to 15, thus skipping over the already placed 10 tiles. This design concept is similar to Visual Pattern Test [11] used for measuring *pure* visual working memory. Recall that according to Constraint $C3$, after the user completes A_{rew}, all 25 tiles are visible. Therefore, if the user fails to hold the information (status of the game) in his working memory, his $|\theta|$ must be lower bounded by 15. In such case, the time elapsed on placing the target tile correctly is relatively longer. Therefore, it is necessary to investigate the working memory capacity in terms of information processing speed for each individual user.

3.3 Revisiting Automatic Processing & Priming Effect

Recall, that during A_{rew} two straight lines are drawn from P_{t_c} to P_{t_r} to guide the movement of the gold coin. Constraint *C4* provides priming for invoking the automatic processing. We conjecture that a user who is primed with the color

change of the straight lines, will drag g_c on a straighter trajectory compared to an unprimed user. We measure the effectiveness of the prime, EOP, as the ratio of the length of the line *not* overlapped by the gold coin to the length of the guiding line. Constraint $C6$ also has priming effect on the user, particularly on the way the gold coins are deposited in the bank, and the tiles are dropped on the grid for the subsequent *instances*. We measure the effectiveness of this priming as the ratio of the two areas. The primes are considered effective if the ratios tend to 1.

3.4 Evidence of Working Memory and Priming Effect from Experimental Data.

We analyze our data to provide evidence for the underlying mental processes. Recall that after a successful match, the user can observe the game status ($C2$). If this information is not lost from the working memory, then $|\theta|$ must decrease with each *instance*, resulting in a descending series of **V**isual **S**earch **T**ime, VST. That is, VSTs must decrease with decreasing $|\theta|$'s as the game progresses. Considering information storage is likely to occur near the end of the game, since it is easiest to recall the last remaining empty cell, we find the length of sub-sequence $l = n-k+1$ of n *instances* such that $VST_k > VST_{k+1} > \ldots > VST_n$ for $|\theta|_k > |\theta|_{k+1} > \ldots > |\theta|_n = 1$. Since the user might store partial information as well, we allow some tolerance such that v number of violations (sign changes) can happen in the sequence. Figure 1(a) shows the average sub-sequence length l (for 5 games) when v is varied from 0-3. We can observe the variations in working memory capacities among a group of users.

On the other hand, after triggering a prime, a user might, (1) receive it and get influenced (invocation of automatic processing), (2) receive it but *not* get influenced by it (no invocation of automatic processing) or, (3) *not* receive it at all; e.g. a *cautious* user dropping a gold with overlapping area $\geq 90\%$. Considering that the prime has been triggered in the i^{th} *instance*, the following is a possible explanation for the three cases in the $i+1^{th}$ *instance*, for Constraint $C6$. (1) $EOP^i_{C6} < EOP^{i+1}_{C6}$ and g_c was misplaced at the i^{th} *instance*. In other words, the user *drops* g_c with higher accuracy in the $i+1^{th}$ *instance*, i.e. prime was received and was effective. (2) $EOP^i_{C6} \geq EOP^{i+1}_{C6}$ and g_c was misplaced in the i^{th} *instance*, i.e. prime was received but was not effective. (3) g_c was never misplaced in the i^{th} *instance*, i.e. prime was not received. Figure 1(b) shows the average percentage of each of these cases (for 5 games) when primes are triggered. Notice that due to slight inaccuracies in placement, all users received at least some prime. Around 30% of the users never *missed* getting influenced (without being aware) by the prime whenever they received it.

4 Feature Estimation Process

Raw Data. The interaction data during performing the task is collected for each user: (1) The x_{ce} and y_{ce} co-ordinates of the click event e and the corresponding timestamp t_{ce}. (2) The x_{re} and y_{re} co-ordinates of the release event e and the

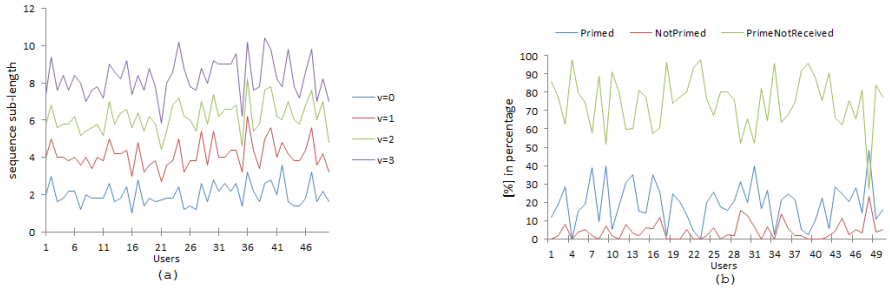

Fig. 1. We chose 50 users randomly from Experiment-1(b). (a) Average sub-sequence length with a linear relationship between VSTs and search set sizes indicating differences in working memory capacity of different users. Different curves represent varying number of violations from linear relationship. (b) Percentage of the three cases when prime is triggered (Section 3.4). Users are influenced in different ways.

corresponding timestamp t_{re}. (3) The horizontal co-ordinate x_{de}, $de = 1 \ldots n$ and the vertical co-ordinates y_{de}, $de = 1 \ldots n$ of the pointing device sampled at 100 ms intervals.

4.1 Cognitive Feature Estimation

We estimate features that capture cognitive abilities from the aforementioned raw data. We also discuss other important features that are based on the users' responses to certain stimuli during the execution of an *instance*.

Drop and Pick Reaction Time, DPT (f_1, f_2). At the end of the A_{resp} action, the user picks up the gold coin, g_c, appearing at P_{t_r}. The time elapsed between the stimulus (g_c) and the user picking it up (response) is denoted by t^g_{DPT} (f_1). At the end of A_{rew}, after depositing the gold coin at P_{t_c}, the user picks t_c from location P_{t_c}. The time elapsed between the appearance of the stimulus (t_c) and the user picking it up (response) is referred to as the t^t_{DPT} (f_2). DPT might seem similar to traditional pause-and-click. However, traditional pause-and-click is highly dependent on what the user is currently reading or exploring [6]. DPT is the result of a controlled stimulus and therefore, is not content-specific.

Visual Search Time, VST & ratio(f_3, f_4). VST is the time required for the user to visually search and detect the target tile. VST is calculated by the subtraction method [2]. The subtraction method involves subtracting the amount of time information processing takes with the process from the time it takes without the process. That is the time difference between actions A_{resp} ($A^t_{resp} + t^t_{DPT}$) and A_{rew} (A^t_{rew}),

$$VST = (A^t_{resp} + t^t_{DPT}) - A^t_{rew}. \quad (1)$$

It is important to consider t^t_{DPT} in the above equation, since a t_c is exposed as soon as a g_c is deposited. So the minuend of equation (1) refers to the time elapsed between the exposure of t_c and its correct placement inside the grid.

The time elapsed during A_{rew} is simply the movement time and does not involve user's thinking or search time. Therefore, we are able to distill the plain visual search time for each *instance*. Moreover, note that the subtraction method allows VST to self-adjust to the user's specific environment by remaining immune to differing mouse speed or acceleration. We also consider the ratio of A^t_{resp} to A^t_{resp} + t^t_{DPT}, (f_4) to capture the phenomena where user searches while dragging, or searches and then drag.

Information Processing Speed, IPS (f_5). If information (game status) is not lost from the user's working memory then in the i^{th} instance the user is left with 25 - i +1 alternatives for the search operation. We can derive the following equation for IPS from Hick-Hymen law [12, 13]

$$IPS = \frac{H_i}{VST}. \tag{2}$$

The amount of information in the i^{th} instance can be expressed as $H_i = \sum_{k=1}^{|\theta|} P_k \left(log_2 \left[\frac{1}{P_k} \right] \right)$ where P_k is the probability of the k^{th} alternative in the i^{th} instance with $|\theta| = 25 - i + 1$ alternatives. Due to the Constraints *C3 & C7* all these alternatives are equally probable.

Pause and Search, P&S (f_6, f_7). While dragging the target tile we noticed that the user sometimes pauses and searches for the target inside the grid. If the user remains on the same pixel for more than $\alpha = 0.1$ seconds while dragging the tile or gold, we refer to it as a pause. We measure the ratio of tile paused time to A^t_{resp} during A_{resp} (f_6), and the ratio of gold paused time to A^t_{rew}, (f_7) during A_{rew}.

Effectiveness Of Priming, EOP (f_{8-19}). Recall that Constraint *C6* provides the priming effect necessary to invoke an automatic processing. We measure the effectiveness of this priming through EOP_{C6}

$$EOP_{C6} \ = \frac{Area\ overlapped\ between\ source\ and\ destination}{Area\ of\ source\ or\ destination}. \tag{3}$$

EOP^g_{C6} (f_8) refers to the effectiveness of priming while depositing g_c (source) in the bank (destination). EOP^t_{C6} (f_9) refers to the effectiveness of priming while placing the t_c (source) on the matching tile t_r (destination). We consider related features that might capture the effectiveness of priming as well. We consider the *Drop Error Distance* for tile, $\triangle^{E_t}_{x_{re},\ y_{re}}$ (f_{10}) and gold, $\triangle^{E_g}_{x_{re},\ y_{re}}$ (f_{11}), defined as the distance from the drop point to the center of their destination. We measure the *Click Error Distance* which is the distance of the click point to the center of the tile $\triangle^{E_t}_{x_{ce},\ y_{ce}}$ (f_{12}) and the gold $\triangle^{E_g}_{x_{ce},\ y_{ce}}$ (f_{13}). We also consider the *Drop Error Angle* for tile/gold $\angle^{E_{t/g}}_{(x_{re},\ y_{re})}$ ($f_{14},\ f_{15}$) which is the angle made from the drop point to the (+ve) x-axis with the center of the destination being the vertex, and *Click Error Angle* which is the angle made from the click point to the (+ve) x-axis with center of the tile/gold being the vertex, $\angle^{E_{t/g}}_{(x_{ce},\ y_{ce})}$ ($f_{16},\ f_{17}$).

On the other hand, we measure the effectiveness of priming due to $C4$ as the ratio of two lengths. EOP_{C4} is the ratio of the length of the lines *not* overlapped

by the gold coin to the total length of the guiding lines. EOP_{C4}^{L1} (f_{18}) and EOP_{C4}^{L2} (f_{19}) are the effectiveness of priming on the top and bottom guiding lines respectively.

5 System Design

In this Section we provide details of the classification technique used to identify the users. We then discuss the security of our system against impersonation attack. Details on how we measure the error metrics in our system are also provided.

5.1 Classification Technique

We use a statistical approach for classifying the users. We model the features as random variables F_1, F_2, ..., F_n and assume class-conditional independence between them. In the i^{th} *instance* of the game a row of feature values $F^i = (f_{1,i}, f_{2,i}, \ldots, f_{n,i})$ is generated. Therefore, for a sequence of k *instances* denoted by $F = (F^1, \ldots, F^k)$, the interaction information can be denoted using a matrix of size $k \times n$. During the learning stage the probability density functions of the features are estimated using a non-parametric approach. In the classification stage posterior probabilities are used to estimate the probability of a classification being correct.

Learning. The learning phase consists of the estimation of the probability density function for each of the feature vectors. A parametric approach to estimating a density, f, involves assuming that f belongs to a parametric family of distributions. We resort to using non-parametric approach, in particular, the kernel density estimator [14] to avoid making any assumption on the distribution of the underlying population and to better understand the structure of the data. The easiest non-parametric estimation of a probability distribution is the use of histogram. It is simple but has disadvantages such as discontinuity and high sensitivity to bin edges. Kernel density estimators are superior to histogram and are quite intuitive [14, 15].

We estimate the unknown density function $f_j(x)$ of the j^{th} feature vector, represented by a random variable F_j, based on its m samples (or training data) x_1, .., x_m. Assuming that the observations are independent realizations of F_j, the estimation of the density function, $\hat{f}_j(x)$, using a kernel density estimator, for univariate case is $\hat{f}_j(x) = \frac{1}{mh} \sum_{i=1}^{m} K\left(\frac{x-x_i}{h}\right)$. At this point, the estimation of $f_j(x)$ reduces to (1) choosing a kernel function K and, (2) selecting an appropriate bandwidth selection algorithm to determine h. Although the choice of kernel functions is not of particular importance for an experiment [14], according to our empirical results, we used Gaussian kernel among others as the kernel function K. The kernel estimate is constructed by centering the Gaussian kernel at each observation. Therefore, the value of the kernel estimated at a point x_i is simply the average of the m normal kernel ordinates at that point. Therefore,

the width of the chosen kernel function determines the smoothness of the resulting density function. Oversmoothing can happen as a result of larger width whereas undersmoothing can happen due to smaller width. Therefore, selecting the appropriate width h is a crucial task while estimating the density function.

The performance of the kernel density estimator depends on how closely the estimated $\hat{f}_j(x)$ resembles the true $f_j(x)$ of the j^{th} feature. This performance can be measured in terms of the MISE (Mean Integrated Square Error), which globally measures the distance between $\hat{f}_j(\cdot; h)$ and $f_j(x)$,

$$E[ISE(\hat{f}_j(.;h))] = E\int \hat{f}_j(x)^2\, dx - 2E\int \hat{f}_j(x)\, f_j(x)\, dx + E\int f_j(x)^2\, dx \quad (4)$$

We use least square cross validation (LSCV) [16, 17] which is a data-driven bandwidth selector. The third term on the right hand side of equation (4) does not depend on h and can be ignored. The first term can be calculated from the observations. The middle term depends on h and contains an unknown quantity $f_j(x)$. In order to solve this issue, we resort to a leave-one-out LSCV. Although there are more complex bandwidth selection algorithms [18], we chose LSCV since it is simple and intuitive.

Classifying. We assume class-conditional independence between the features, modeled as random variables F_1, F_2, ..., F_n, for a user u_w, $w \in \{1, \ldots, m\}$. So for the i^{th} *instance*, $Pr\left(F^i|u_w\right) = \prod_{j=1}^{n} Pr(f_{j,i}|u_w)$. Although the class-conditional independence between features is not true in general, the assumption works well in many complex real life systems. The posterior probability of a user u_w for an *instance*,

$$Pr\left(u_w|F^i\right) = \frac{Pr\left(u_w\right)\prod_{j=1}^{n} Pr(f_{j,i}|u_w)}{Pr(f_{1,i},\ \ldots,\ f_{n,i})}. \quad (5)$$

We then classify a test *instance* according to the largest posterior probability. We accept a sequence of *instances* as genuine if the number of accepted *instances* exceeds some decision threshold α. The value of threshold α is set in such a way such that the false acceptance rate is close to the false rejection rate.

5.2 Security Model

Correctness and security of a biometric system is measured using False Acceptance Rate and False Rejection Rate. A biometric system should not accept a user without genuine biometric (FA), and should not reject a genuine user (FR). Therefore, both these metrics should remain close to zero. We evaluate our system performance using user u's own test sessions and other test sessions from $n-1$ users. A positive test *session* of length l *instances* is considered misclassified for a user u, if the classifier outputs a score below the threshold α. This is referred to as a False Rejection. On the other hand, a negative test *session* is considered classified if the classifier's output score is above the threshold α. This is referred to as a False Acceptance. We calculate the FAR as the ratio of FA to TN, where FA is the number of false acceptance and TN is the number of

test *sessions* belonging to the $n-1$ other users. The FRR is calculated as the ratio between FR and TP where FR is the number of false rejection and TP is the number of test *sessions* belonging to the user u.

Impersonation Attack. Impersonation attack in biometric systems involves an attacker generating the biometric information of a legitimate owner *somehow* without the owner being present at the scenario [19], for example by lifting latent fingerprints from objects and presenting it to the system. In our system the attacker is allowed to observe a target user's gameplay. Later on, the attacker tries to mimic that user's gameplay in order to get authenticated as the victim user. The success of the attacker is measured as the probability of success in being authenticated as the claimed user. We build a web-based program capable of simulating any user's game playing activities once fed with data collected during the data acquisition period. We then select experienced users and instruct them to observe and imitate other users' simulations (Section 6.3). We assume that the user's interaction data are not available to any computer programs such as spyware.

6 Experiments & Results

We formulated two questions. (1) Is it possible to verify a user based solely on the derived cognitive features with high accuracy under, (a) controlled condition and under, (b) non-laboratory condition? (2) How effective are impersonation attacks against our system when carried out by trained users?

We devised three separate experiments to answer the above questions. Since our experiments required human volunteers, we obtained approval from the Research Ethics Board of our University. The 1^{st} experiment was carried out in a controlled environment with 23 graduate students. The 2^{nd} experiment consisted of 129 workers from Amazon Mechanical Turk. And the 3^{rd} one was carried out with 5 graduate students in a controlled environment and 10 Amazon Mechanical Turk workers. All experiments were divided into three phases (1) Phase-I, where participants agreed to the consent information. (2) In Phase-II, a short video displayed how the game is played for a few *instances*. (3) In Phase-III, participants were required to fill up an exit survey consisting of the standard SUS [20] and a few other questions S_{Fun}. In all the experiments, interaction data were recorded using JavaScript and submitted passively via AJAX requests to the web server. We randomly picked a set of distinct images for each user in a *session*.

6.1 Experiment 1(a): Accuracy & Efficiency (Controlled Condition)

The goals of the 1^{st} experiment were to figure out, (1) Accuracy and verification time of our system when users are trained in a non-distracting environment using a single platform and, (2) perform an analysis on the derived features.

Setup. Each user in a *session* is required to play the game 7 times, every time with a new random partitioned image, resulting into a dataset of $25 \times 7 = 175$ *instances*. Afterwards, they completed the exit survey. All of them used a PC

Works	FAR	FRR	Session Size	System	Notes
[5]	2.46%	2.46%	2000 Mouse Actions	MDS	Free mouse movement
[4]	6.3%	6.3%	20 Strokes	MDS	Confined within a task.
[6]	1.3%	1.3%	20 Mouse clicks	MDS	Free mouse movement
[21]	2.11%	2.11%	25 Text Characters	HBS	Confined within a task
[22]	0.01%	4%	683 Characters	KDS	Fixed-text input
[23]	Accuracy 93.3% - 99.5 %		200 Characters	KDS	Free-text input
[Ours]	0%, 2.3%	0%, 7.8%	25 Instances	CBS	Confined within a task

Table 1. Comparison with other approaches. Mouse Dynamics System(MDS), Keystroke Dyanmics System (KDS), Homogeneous Physio-Behavioral System(HBS), CBS (Cognitive based Biometric System). Session size refers to the amount of interactions during the authentication phase.

with 2.10 GHz Intel i3, 4GB RAM and an wireless optical USB mouse. They used Google Chrome on a screen of resolution 1366×768 (96 DPI) in Windows 7 SP1 OS.

Intra-Session Evaluation: The dataset was divided into two parts. The first part consisting of 5 games (g_1, ..., g_5) each of 25 *instances*, is used for training purpose and the last two games g_6, g_7 are separately used for testing purpose. The average EER is 0% suggesting that all users have consistent game playing activities in a continuous *session*. Figure 2(a) shows the variations in FAR and FRR, at $\alpha = 0.5$, as the number of *instances* are varied. Less number of *instances* e.g. 16 would have significantly decreased the enrollment and verification time but with an FRR of 8.7%. Table 1 provides a comparison of our system with others. The verification time is the time taken to collect the verifiable biometric data and the time taken to complete the classification task [24]. It took an average 76.7 seconds to complete one game (25 *instances*) and an average of around 44.7 seconds to complete part of the game (16 *instances*), considering time intervals, $(A^t_{resp}+A^t_{rew})$. These are comparable to several recently proposed authentication technique (Table 2). Our classification time does not have any particular impact on the verification time.

Feature Analysis: We use data from the 23 users for analyzing the features. Figure 3(a) shows the correlation coefficients of pairs of features in a color-coded plot. There are a few highly correlated features. According to our observation these pairs do not have any effect on classification accuracy. We consider all 19 features, since they do not add any significant burden on the training time. We also considered the strength of each of the features in identifying the 23 users. Each feature was in turn used for learning and classifying. We found that features related to DPT, EOP_{C4}, VST and $P\&S$ are in the top 7 based on average EER.

Inter-session Evaluation: Participants were emailed to play in three other occasions, each separated by 1-day, 2-day and 3-day intervals respectively. We consider these intervals to be congruous with real account login intervals. The training data came from the original data acquisition session (g_1, ..., g_5). On each occasion participants were required to complete one game using any machine

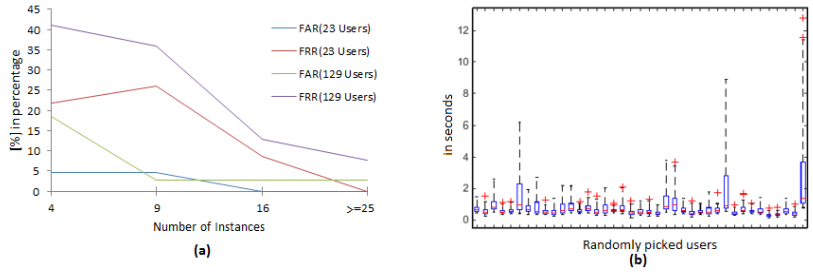

Fig. 2. (a) Avg. FAR and FRR at $\alpha = 0.5$ (Section 5.1) with varying number of *instances*. The first 5 games are used for training and the last 2 for testing purpose (Intra-session). (b) Box plot of Gold Pick Reaction Time (in seconds) before noise removal (whiskers set to 3) of a group of Amazon workers showing outliers.

(except cellular device) and browser at their most suitable time. At $\alpha = 0.5$, FAR remained 0% through all *sessions* with FRR being 0%, 8.70% and 4.35% on the 1, 2 and 3-day *sessions* respectively.

6.2 Experiment 1(b): Accuracy & Efficiency (Non-Laboratory Condition)

The goal of this experiment was to evaluate the system with random users from different parts of the world who self train and later remotely authenticate. We created 4 HITs altogether. The 1^{st} HIT was created with 130 assignments to have 130 unique workers. We gathered 129 *valid* submissions until the HIT expired. Each assignment was worth \$0.7. We refer to each HIT as a *session*. The workers were directed to the website hosting the game. After watching the video, they were required to play 7 games and then complete an exit survey. At the end workers copy-pasted a code generated on our website back to Amazon.

Intra-Session Evaluation: Similar to Experiment-I, $g_1, \ldots, g_5$ were used for training and g_6, g_7 for testing purpose. We noticed abnormal VSTs in the dataset for few cases. On closer scrutiny and observing the simulations for such cases we noticed very large *Drop and Pick Reaction Time*, t^t_{DPT} and t^g_{DPT}. This shows that users are more likely to get distracted at the end of the actions (A_{resp}, A_{rew}) rather than while performing them. We detect these *extreme* outliers using interquartile range for each user u, with the upper fence $UF = f_u^{Q3} + (3 \times f_u^{IQR})$, $f_u \in \{t^t_{DPT_u}, t^g_{DPT_u}\}$ and replacing them with UF (Figure 2(b)). Noise removal was done separately for the training and test dataset. Figure 2 (a) shows the avg. error rates for varying number of *instances* for the 129 workers. Our classifier reaches an FAR of 2.3% and FRR of 7.8% with ≥ 25 *instances*. The average time it took to complete one assignment of the HIT is 24.3 minutes. Mouse type statistics included wireless/wired mouse (61.3%), laptop touchpad (38.7%) (user claimed).

Inter-Session Evaluation: We created another 3 HITs, each separated by 1, 2 and 3-day intervals respectively. We made sure participants completing the

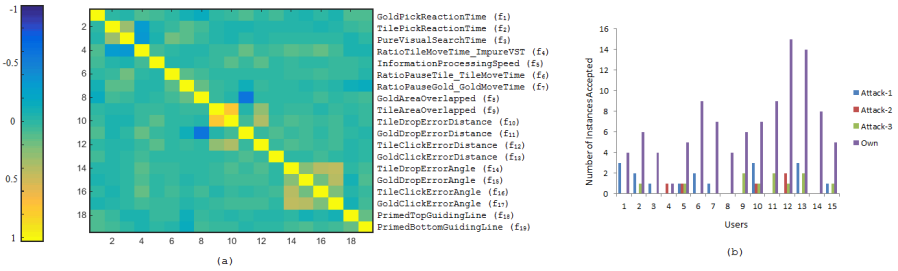

Fig. 3. (a) Correlation coefficients [-1,1] of pairs of features in a color-coded plot. (b) Results from Impersonation attack. Most *instances* are accepted as "own" rather than as the victims' (attack-1, 2 and 3).

4^{th} HIT had already participated in the previous 3 HITs. Each assignment was worth \$0.2. We received 49, 37 and 37 valid submissions until the HIT expired. The assignment required completing only one game. As before the classifier used the initial acquired data $g_1, \ldots, g_5$ for learning. The FAR were 2.08%, 0%, 2.70%, and FRR 8.16%, 8.11%, 5.50% with $\alpha = 0.5$ on the 1-day, 2-days, 3-day *sessions* respectively. This suggests that even after small periods of inactivity and using the original training data, the classifier can still distinguish the users.

6.3 Experiment 2: Impersonation Attack

Impersonation attack demands trained users capable of mimicking a victim's gameplay. Therefore, participants were selected based on how fast they completed the previous *sessions*. We selected 5 participants from the 1^{st} pool and 10 Turkers from the 2^{nd} pool. Each assignment was worth \$0.5 (Amazon). Each participant was required to watch and mimic the simulations of 3 victims. We considered a strong attack scenario. We, (1) displayed a clock while the simulation was playing, (2) provided the same image in the attack phase, (3) declared a bonus of \$0.5 if the Turkers could mimic accurately and, (4) allowed to repeat the attacks as many times as desired. All attackers were given instructions to observe when and how tiles and gold are picked, dragged and dropped.

A successful attack would require reproducing the cognitive features of a victim. Figure 3(b) depicts the *maximum* number of *instances* (out of all attempts) that have been correctly classified to the corresponding victims. A maximum of 3 *instances* were correctly mimicked by user-1 and worker-10 and 13. None of the attackers would have successfully authenticated with the threshold set at $\alpha = 0.5$. In fact, workers 12 and 13 were identified as "themselves" in one of the games (attacks). A successful attack in this case would require mimicking almost all 19 cognitive features, which appeared to be a hard task. The challenge tiles appeared randomly, and the grid was shuffled at each instance (*C3*), and so the sequence of challenges in the simulation and actual attack differed, making it harder to recall the corresponding VSTs and other reaction times.

	[Ours]	[21]	[4]	[25]
Verification Time, VT(Approx.)	76.7s, 2.5min	39s	25s	5-6min
Test Session Size	25 instances	25 characters	25 strokes	540 items
Enrollment Time, ET(Approx.)	9.8min, 24.3min	6.5min	6.7min	30-40min
Enrollment Session Size	$\geq$175 instances	250 characters	400 strokes	3780 items

Table 2. Comparison of VT, ET, test and enrollment (train+test) session size. Session size refers to the amount of interactions made during the enrollment or verification phase.

7 Discussion and Conclusions

Our cognitive based authentication system assumes that users play consistently and use his cognitive abilities appropriately. This is arguably a desirable property and careless treatment of security should be punished by denying access. Our proposed system, like other biometric systems cannot authenticate a user if their biometric data are damaged (e.g. a severe burn to one's finger). In cognitive based systems the damage may be long term and caused by cognitive and mental disorders, or short term when under the influence of substance. To provide user access in such cases depending on the type of the damage and the organizational policy, a different type of authentication system such as a password system, should be used as backup. Cognitive abilities can change slowly over time due to age and experience. In such cases, an adaptive enrollment mechanism is necessary to capture and represent the most current features of the user.

Analysis of user surveys shows that the game is user friendly and easy to play: 71.8% and 85.5% of the users agreed that the game is fun and easy respectively in S_{Fun} questions. The average SUS (System Usability Scale) [20] scores are within the user-friendly software ratings of 60-70 [26]. Our system can be used as a stand-alone system, or can be used in a multi-factor authentication system. Since well selected cognitive features cannot easily be mimicked the authentication system will be secure. Our future work will include designing systems that invoke other mental processes and extract a wider range of cognitive features.

Acknowledgments. This research is in part supported by Alberta Innovates Technology Futures and Telus Mobility Canada.

References

1. Galotti, K.M.: Cognitive Psychology In and Out of the Laboratory. SAGE Publications, Inc (2013)
2. Sternberg, R.J.: Cognitive Psychology. Cengage Learning (2011)
3. Amazon mechanical turk. last accessed on 12/12/2014. https://www.mturk.com/mturk/welcome
4. Gamboa, H., Fred, A.: A behavioral biometric system based on human-computer interaction. Proc. SPIE 5404, 381–392 (2004)

5. Ahmed, A., Traore, I.: A new biometric technology based on mouse dynamics. Dependable and Secure Computing, IEEE Transactions on 4(3), 165–179 (July 2007)
6. Zheng, N., Paloski, A., Wang, H.: An efficient user verification system via mouse movements. In: Proceedings of the 18th ACM Conference on Computer and Communications Security. pp. 139–150. CCS '11, ACM, New York, NY, USA (2011)
7. Frank, M., Biedert, R., Ma, E., Martinovic, I., Song, D. (2013). Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication. Information Forensics and Security, IEEE Transactions on, 8(1), 136-148.
8. Wickens, C.D., Lee, J.D., Liu, Y., Gordon-Becker, S.: Introduction to Human Factors Engineering (2nd Edition). Pearson (2003)
9. Van Zandt, T., Townsend, J.T.: Self-terminating versus exhaustive processes in rapid visual and memory search: An evaluative review. Perception & Psychophysics 53(5), 563–580 (1993)
10. Bargh, J.A., Chen, M., Burrows, L.: Automaticity of social behavior: Direct effects of trait construct and stereotype activation on action. Journal of personality and social psychology 71(2), 230 (1996)
11. Della Sala, S., Gray, C., Baddeley, A., Allamano, N., Wilson, L.: Pattern span: a tool for unwelding visuo–spatial memory. Neuropsychologia 37(10), 1189–1199 (1999)
12. Adams, J.A.: Human factors engineering. Macmillan Publishing Co, Inc (1989)
13. Hick, W.E.: On the rate of gain of information. Quarterly Journal of Experimental Psychology 4(1), 11–26 (1952)
14. Wand, M.P., Jones, M.C.: Kernel smoothing, vol. 60. Crc Press (1994)
15. Zucchini, W., Berzel, A., Nenadic, O.: Applied smoothing techniques (2003)
16. Rudemo, M.: Empirical choice of histograms and kernel density estimators. Scandinavian Journal of Statistics pp. 65–78 (1982)
17. Bowman, A.W.: An alternative method of cross-validation for the smoothing of density estimates. Biometrika 71(2), 353–360 (1984)
18. Jones, M.C., Marron, J.S., Sheather, S.J.: A brief survey of bandwidth selection for density estimation. Journal of the American Statistical Association 91(433), 401–407 (1996)
19. Bolle, R.: Guide to biometrics. Springer (2004)
20. Brooke, J.: Sus-a quick and dirty usability scale. Usability evaluation in industry 189, 194 (1996)
21. Hamdy, O., Traoré, I.: Homogeneous physio-behavioral visual and mouse-based biometric. ACM Transactions on Computer-Human Interaction (TOCHI) 18(3), 12 (2011)
22. Gaines, R.S., Lisowski, W., Press, S.J., Shapiro, N.: Authentication by keystroke timing: Some preliminary results. Tech. rep., DTIC Document (1980)
23. Villani, M., Tappert, C., Ngo, G., Simone, J., Fort, H.S., Cha, S.H.: Keystroke biometric recognition studies on long-text input under ideal and application-oriented conditions. In: Computer Vision and Pattern Recognition Workshop, 2006. CVPRW'06. Conference on. pp. 39–39. IEEE (2006)
24. Kung, S.Y., Mak, M.W., Lin, S.H.: Biometric authentication: a machine learning approach. Prentice Hall Professional Technical Reference (2005)
25. Bojinov, H., Sanchez, D., Reber, P., Boneh, D., Lincoln, P.: Neuroscience meets cryptography: designing crypto primitives secure against rubber hose attacks. In: Proceedings of the 21st USENIX Security Symposium (2012)

26. Lewis, J. R., Sauro, J. (2009). The factor structure of the system usability scale. In Human Centered Design (pp. 94-103). Springer Berlin Heidelberg.
27. Chiang, A., Atkinson, R.C.: Individual differences and interrelationships among a select set of cognitive skills. Memory & Cognition 4(6), 661–672 (1976)
28. Jensen, A.R.: Individual differences in the Hick paradigm. Ablex Publishing (1987)
29. Dovidio, J.F., Gaertner, S.L.: Stereotyping, prejudice, and discrimination: Spontaneous and deliberative processes. Paper presented at the meeting of the Society of Experimental Social Psychology, Washington, DC (1995, October)

A Related Work

The work closest to ours, although it is a combination of mouse dynamics and cognitive factors, is that of Hamdy and Traore [21]. The authors combine visual search and short-term memory effect with mouse dynamics. Their system requires the user to search for letters on a shuffled *virtual* keyboard. However, it is highly likely that the exposure of the same *virtual* keyboard and the string of letters have affected the visual search process. The work in [25] uses the concept of implicit learning from cognitive psychology whereby the user is trained on a fixed sequence which can later be used during authentication. Our system does not rely on implicit learning and uses a random challenge sequence and so the user does not repeat the same sequence of activities. Individual differences in visual search task and information processing speed are evident from recent works [27, 28]. Individual differences in automatic processing due to priming are evident from [29].

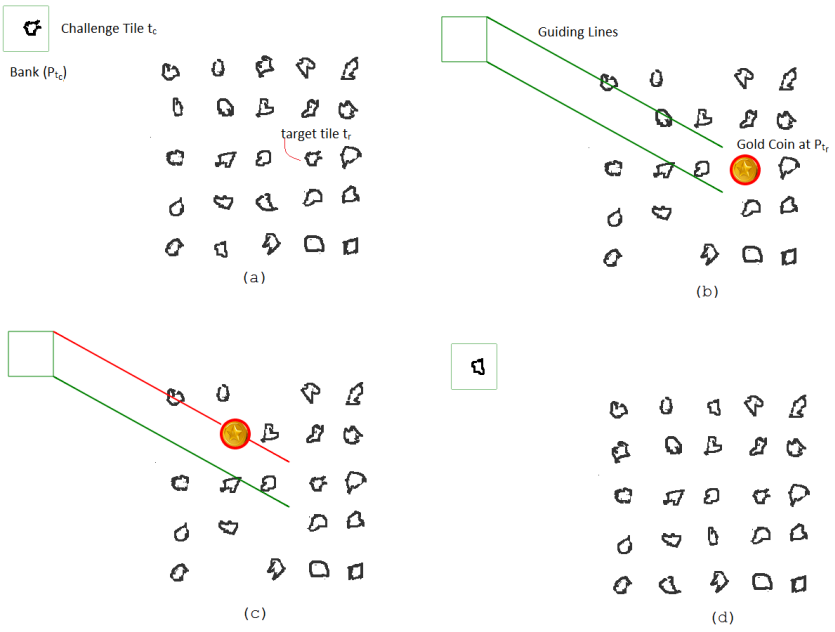

Fig. 4. (a) User is presented with a challenge tile, t_c, at the beginning of the 21^{st} *instance*. (b) User performs A_{resp}, i.e. drags and drops t_c onto t_r inside the grid. On a correct match the *loose* tiles disappear showing the current game status (at 21^{st} *instance*). (c) User performs A_{rew}, i.e. drags and drops gold coin, g_c, onto P_{t_c}. Top guiding line color changes from green to red as the gold coin touches it (Constraint $C4$). (d) User successfully deposits g_c and gets the next challenge tile. All 25 tiles are visible at this point. Notice that the 21 *unmovable* tiles in b and d have not changed their positions. All the *loose* tiles have changed their position (compare a and d). The target tile t_r appears at its original position in the image.