12-2014

Avatar captcha : telling computers and humans apart via face classification and mouse dynamics.

Darryl Felix D'Souza
University of Louisville

AVATAR CAPTCHA: TELLING COMPUTERS AND HUMANS APART VIA FACE CLASSIFICATION AND MOUSE DYNAMICS

By

Darryl Felix D'Souza
M.S. in Computer Science, University of Louisville, USA, 2009
B. Eng. in Computer Engineering, University of Mumbai, India, 2006

A Dissertation
Submitted to the Faculty of the
J.B.Speed School of Engineering of the University of Louisville
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy

Department of Computer Engineering and Computer Science
University of Louisville
Louisville, Kentucky

December 2014

AVATAR CAPTCHA: TELLING COMPUTERS AND HUMANS APART VIA FACE CLASSIFICATION AND MOUSE DYNAMICS

By

Darryl Felix D'Souza
M.S. in Computer Science, University of Louisville, USA, 2009
B. Eng. in Computer Engineering, University of Mumbai, India, 2006

A Dissertation Approved

on October 13th, 2014

by the following Dissertation Committee:

Dr. Roman V. Yampolskiy, CECS Department,

Dissertation Director

Dr. Dar-jen Chang, CECS Department

Dr. Ibrahim Imam, CECS Department

Dr. Charles Timothy Hardin,

Industrial Engineering Department

DEDICATION

This dissertation is dedicated to my dear parents Felix and Philomena, my late grandpa Joseph, my grandma Lily, my dear family members and friends who have been my moral support through their incessant love and patience. I owe this one to you all as I could never achieve this pinnacle in my career without your blessings and friendship.

ACKNOWLEDGMENT

The completion and writing of my Ph.D. dissertation has been a long and wonderful journey. It has been one of the most significant academic challenges with an immensely rewarding experience and a learning curve that has taught me the l of research. Without the support, patience and guidance of the following people I would not have achieved this milestone. I owe them my deepest gratitude.

With great pleasure and gratitude I thank my mentor Dr. Roman V. Yampolskiy who has been inspirational and motivating through his wisdom, knowledge and commitment. He has been a great source of guidance complemented with a down to earth attitude and an innovative research vision.

My committee members, who have bestowed upon me the knowledge through their courses that have been extremely helpful and instrumental not only in completing this dissertation but also in my academic career.

My friend and colleague, Jacob Matchuny who deserves a special vote of thanks for assisting me in this work with patience, interest and enthusiasm.

The volunteer subjects and online human users who contributed their invaluable time and feedback towards solving the CAPTCHA challenges and helping me gain some resourceful insights into my work as well as validate it.

My lab colleagues Abdallah Mohamed, Nawaf Ali, Marc Beck and Roger Ouch together with my department colleagues and friends for helping me during different phases of my work. They have supported and motivated me as fellow peers. We have all been a close-knit, diverse group that enjoyed research, travel, sports and food.

The two-year Grosscurth fellowship and the Department of Computer Engineering and Computer Science for two and a half additional years of financial support that aided the pursuit of my Ph.D.

Felix and Philomena, my dear parents, Delna, my sister and my family members without whom this effort would have amounted to nothing. Your unconditional love, support, prayers and profound understanding have been inspirational and encouraging supplements to fuel my motivation to help me achieve this goal.

ABSTRACT

AVATAR CAPTCHA: TELLING COMPUTERS AND HUMANS APART VIA FACE CLASSIFICATION AND MOUSE DYNAMICS

Darryl Felix D'Souza

October 13th, 2014

Bots are malicious, automated computer programs that execute malicious scripts and predefined functions on an affected computer. They pose cybersecurity threats and are one of the most sophisticated and common types of cybercrime tools today. They spread viruses, generate spam, steal personal sensitive information, rig online polls and commit other types of online crime and fraud. They sneak into unprotected systems through the Internet by seeking vulnerable entry points. They access the system's resources like a human user does. Now the question arises how do we counter this? How do we prevent bots and on the other hand allow human users to access the system resources? One solution is by designing a CAPTCHA (Completely Automated Public Turing Tests to tell Computers and Humans Apart), a program that can generate and grade tests that most humans can pass but computers cannot. It is used as a tool to distinguish humans from malicious bots. They are a class of Human Interactive Proofs (HIPs) meant to be easily solvable by humans and economically infeasible for computers. Text CAPTCHAs are very popular and commonly used. For each challenge, they generate a sequence of alphabets by distorting standard fonts, requesting users to identify them and type them out. However, they are vulnerable to character segmentation attacks by bots, English language dependent and are increasingly becoming too complex for people to solve. A solution to this is to design Image CAPTCHAs that use images instead of text and require users to identify certain images to solve the challenges. They are user-friendly and convenient for human users and a much more challenging problem for bots to solve.

In today's Internet world the role of user profiling or user identification has gained a lot of significance. Identity thefts, etc. can be prevented by providing authorized access to resources. To achieve timely response to a security breach frequent user verification is needed. However, this process must be passive, transparent and non-obtrusive. In order for such a system to be practical it must be accurate, efficient and difficult to forge. Behavioral biometric systems are usually less prominent however, they provide numerous and significant advantages over traditional biometric systems. Collection of behavior data is non-obtrusive and cost-effective as it requires no special hardware. While these systems are not unique enough to provide reliable human identification, they have shown to be highly accurate in identity verification. In accomplishing everyday tasks, human beings use different styles, strategies, apply unique skills and knowledge, etc. These define the behavioral traits of the user. Behavioral biometrics attempts to quantify these traits to profile users and establish their identity. Human computer interaction (HCI)-based biometrics comprise of interaction strategies and styles between a human and a computer. These unique user traits are quantified to build profiles for identification. A specific category of HCI-based biometrics is based on recording human interactions with mouse as the input device and is known as Mouse Dynamics. By monitoring the mouse usage activities produced by a user during interaction with the GUI, a unique profile can be created for that user that can help identify him/her. Mouse-based verification approaches do not record sensitive user credentials like usernames and passwords. Thus, they avoid privacy issues.

An image CAPTCHA is proposed that incorporates Mouse Dynamics to help fortify it. It displays random images obtained from Yahoo's Flickr. To solve the challenge the user must identify and select a certain class of images. Two theme-based challenges have been designed. They are Avatar CAPTCHA and Zoo CAPTCHA. The former displays human and avatar faces whereas the latter displays different animal species. In addition to the dynamically selected images, while attempting to solve the CAPTCHA, the way each user interacts with the mouse i.e. mouse clicks, mouse movements, mouse cursor screen co-ordinates, etc. are recorded non-obtrusively at regular time intervals. These recorded mouse movements constitute the Mouse Dynamics Signature (MDS) of the user. This MDS provides an additional secure technique to segregate humans from bots. The security of the CAPTCHA is tested by an adversary executing a mouse bot attempting to solve the CAPTCHA challenges.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I
INTRODUCTION

1.1 Overview

"Why spend time and efforts in solving a problem manually when I can write an automated computer program (bot) to perform the task and achieve the goal ?" Users over the Internet are resorting to writing computer programs, known as bots, to perform automated tasks. In today's technologically advanced era there are numerous tools available to write bots. Bots are chiefly used to perform repetitive tasks automatically thus, saving the user a considerable amount of time and efforts to perform the same task manually. They can be classified as good and bad bots. The good bots are used for constructive purposes such as to find and index new pages for a search engine. E.g. Google, Bing, Yahoo, etc. However, cyber criminals use bots to send out spam emails, rig online polls, distribute malware, steal personal sensitive information such as credit card numbers, bank account numbers and flood servers causing Denial of Service (DoS) attacks creating security issues in the online world. These bots are malicious and have a substantial presence in today's online Internet traffic. So how do we prevent bots from gaining access to resources on computer systems and running the show? How do we segregate the human user activities from bot activities over the Internet ? One potential solution is to use CAPTCHAs (Completely automated Public Turing tests to Tell Computers and Humans Apart). It is a program that can generate and grade tests online that most humans can pass but computers cannot. Text CAPTCHAs are very popular and commonly used. For each challenge, they generate sequence of alphabets, distort them and request users to identify and type them out. However, they are vulnerable to character segmentation attacks by bots and are English-language dependent. Image CAPTCHAs are substitute for them. They use images instead of text and require users to identify certain images to solve the challenges. They are user-friendly and convenient to use.

Our work involves building an image CAPTCHA that displays random images obtained from Yahoo's Flickr. To solve the challenge the user must identify and select a certain class of images. Two theme-based challenges have been designed. They are *Avatar CAPTCHA* and *Zoo CAPTCHA*. The former displays human and avatar faces whereas the latter displays different animal species. In addition to the dynamically selected images, while attempting to solve the CAPTCHA, the way each user interacts with the mouse i.e. mouse clicks, mouse movements, mouse cursor screen co-ordinates, etc. are recorded non-obtrusively at regular time intervals. These recorded mouse movements constitute the Mouse Dynamics Signature (MDS) of the user. This MDS provides an additional secure technique to segregate humans from bots as they have different styles of mouse usage activities. This concept of monitoring the human user mouse interaction is a branch of Behavioral Biometrics, known as Mouse Dynamics. The security of our CAPTCHA is tested by an adversary executing a mouse bot attempting to solve the CAPTCHA challenges.

1.2 Bots and their threats in the online world

A "bot" is an automated computer program that executes malicious scripts and predefined functions on an affected computer. Bots pose cybersecurity threats and are one of the most sophisticated and popular types of cybercrime today. They allow hackers/attackers (bot creators) to gain control of many computers at one time, turning them into "zombie" computers, which operate as part of a powerful "botnet" to spread viruses, generate spam and commit other types of online crime and fraud. A bot might cause your computer to slow down, display mysterious messages or even crash. They sneak into a computer system in many different ways usually through the Internet by seeking vulnerable, unprotected computers to infect [1]. After taking over a computer, it carries out a variety of automated tasks such as:

- **Sending:** Spam emails to users containing web links that download spyware, malware and viruses on clicking it.
- **Stealing:** Personal, sensitive information such as credit card numbers, bank credentials, passwords are stolen.
- **Denial of Service (DoS)**: Cybercriminals use bots to launch DoS attacks to extort money from Web site owners in exchange of regaining control of the compromised sites.
- **Clickfraud:** Fraudsters use bots to boost Web advertising billings by automatically clicking on Internet ads [1].

Some other threats involve:

- **Online polls:** Rig online polls by falsely and recurrently voting in online polls.
- **Free email accounts:** Signing up for several email accounts to dish out spam emails.
- **Gold farming:** Affects virtual worlds (Second Life, Sims Online, etc.) and Massively Multiplayer Online Role Playing Games (MMORPG) such as World of Warcraft, etc. by acquiring in-game virtual currency to be exchanged for real currency and reaching higher game levels over a short time span.

1.3 CAPTCHAs (Completely Automated Public Turing Tests to tell Computers and Humans Apart)

How do we prevent bots and allow human users from getting access to system resources? One solution is by designing a CAPTCHA. A CAPTCHA is a program that can generate and grade tests that most humans can pass but computers cannot. It is used as a tool to distinguish humans from malicious bots. They are a class of Human Interactive Proofs (HIPs) meant to be easily solvable by humans and economically infeasible for computers. The work on distinguishing computers from humans traces back to the original Turing Test [2]. Since the concept of CAPTCHA was widely introduced by von Ahn [3], many design variations have appeared. Text CAPTCHAs are popular and very commonly used. The computer generates challenges by selecting a sequence of letters and distorting them. Figure 1 shows an example from the popular *Captcha Project* undertaken at Carnegie Mellon University (CMU) [4].

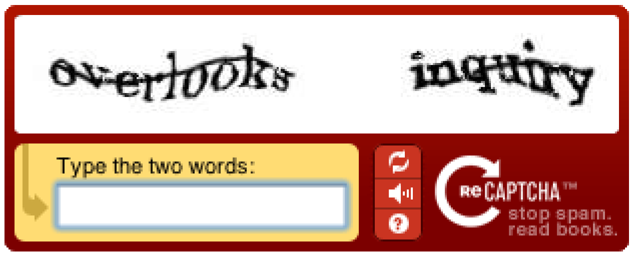


Figure 1: A text CAPTCHA by CMU [4].

It has been shown that text CAPTCHAs are vulnerable to segmentation attacks from Optical Character Recognition (OCR) systems capable of achieving human-like accuracy even when the letters are distorted, as long as the image can be segmented into its constituent letters [5]. Moreover, most of the text-CAPTCHAs are fixed length English word based challenges. The English words based CAPTCHAs, irrespective of their usage of dictionary words, are developed under the assumption that the user is either a native English speaker or fluent with the English vocabulary. These word based CAPTCHAs present distorted images of English letters to the user [6]. They make segmentation difficult by introducing noise in it to thwart OCR attacks. However, it has usability concerns as it adds to the woes of readability and tolerability for the users. Thus, text-based CAPTCHAs seem to be hard not only for a computer but also for humans as well. How do we overcome this problem? One solution is to use images in CAPTCHA challenges. We as human users can easily identify and recognize images however, it is a much more challenging problem for a computer program to solve. Image recognition, as a general machine vision problem, is much harder than character recognition. It helps exploit the gap between human and computer capabilities. However, the database constructed to support an image CAPTCHA must be large and dynamic enough to prevent brute-force attacks on it as a small database is always under the threat of brute force attacks by the attacker to break it. Figure 2 shows an example of an image CAPTCHA.

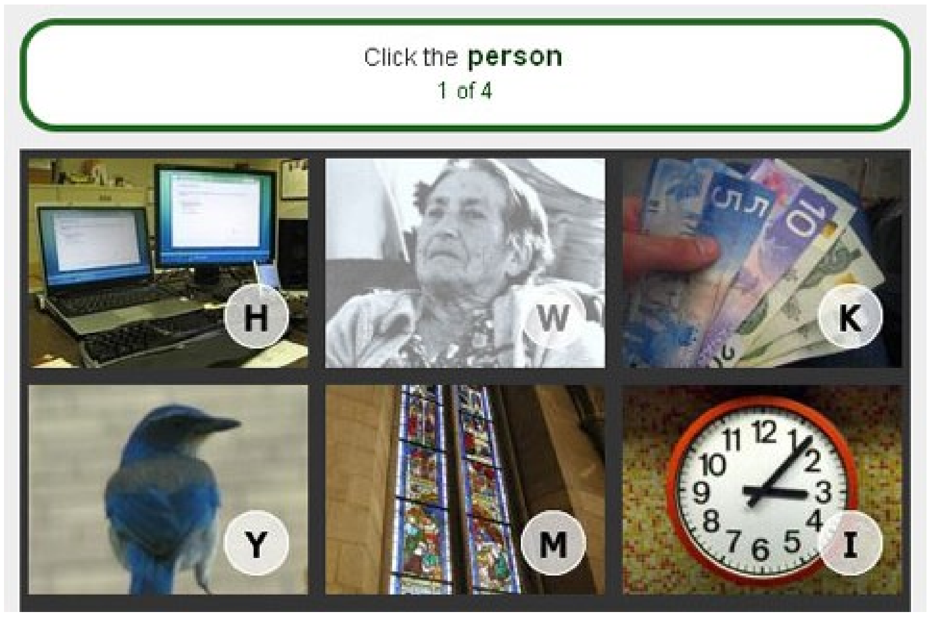


Figure 2: Example of an image CAPTCHA [7].

1.4 Virtual Worlds

Here, we briefly introduce Virtual Worlds. This will help comprehend the concept of avatars used in the CAPTCHA.

Three-dimensional virtual worlds (VW) are gaining popularity over the Internet. They use computer graphics to provide real-time, interactive, three-dimensional environments which seem realistic to the user [8]. VW have the potential to transform the operational ways of society. VW provide a digital space for the user and "mirrors" the real world activities. VW help form social groups and communities revolving around common interests amongst individuals across the world [9]. In today's advanced era of technology it becomes easier to build VW. These worlds are conveniently accessible and run on a conventional browser or a computer or a smartphone. There are several popular virtual worlds. For e.g. Second Life [10], Entropia Universe [11], Sims Online [12], etc. VW fosters socializing, shopping, venturing, being creative, educating, setting up enterprises as well as making money [10]. Examples of some of these popular VWs are shown in Figure 3.

Figure 3: Examples of popular VWs.

Virtual businesses exist within virtual worlds. Reuters has established a virtual headquarters in Second Life to broadcast news to both the real as well as the virtual world. National Public Radio has held broadcast sessions within Second Life as well.A few real-world universities are establishing islands in virtual worlds to offer classes. E.g. Kelley School of Business, Indiana University, Bloomington has its presence in Second Life too [13] as seen in Figure 4.

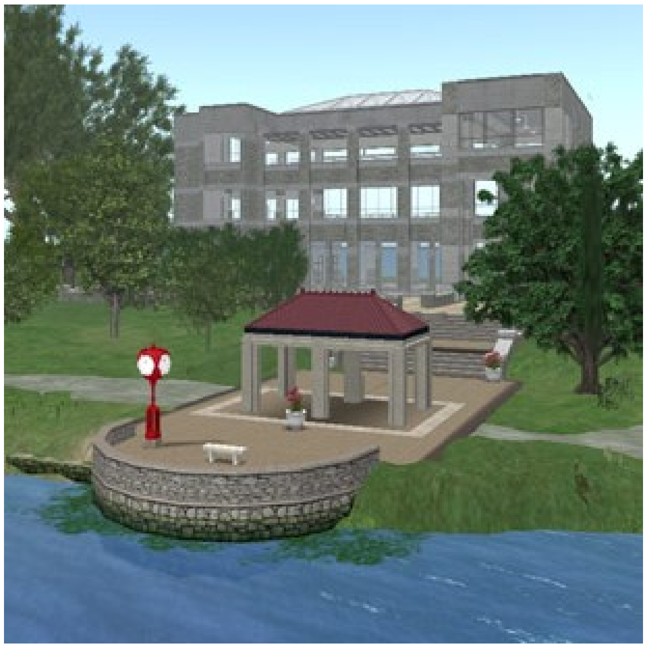


Figure 4: Kelley School of Business in Second Life [13].

1.5 Avatars

They represent the digital identity of a user within VW. Users interact with the VW environment through their respective avatars. Users often wish to create distinct and diverse avatars that reflect their personality pertaining to an imaginary identity. Though it represents a user's identity within VWs, it is not an authentic description of their identity. Within VW, users can choose how and in what way they wish to express themselves by altering the appearance of their avatars adding flexibility to their role. Some users make conscious choices in revealing facts about themselves through their avatars. Some project a popular image as their avatar. Some online services permit only one avatar identity per user account. This helps avoid trust issues and non-reliance of the user on their alternative identities. Some users give their avatars a realistic look that resembles a human being. They believe this helps them relate personally to their avatars. Avatars can also extend to other aspects such as emotions, gestures, animations, speech, voice, etc. They can also be used to express the reputation or status of a user [14]. Figure 5 shows some examples of avatars from the popular VWs Entropia Universe and Second Life respectively.

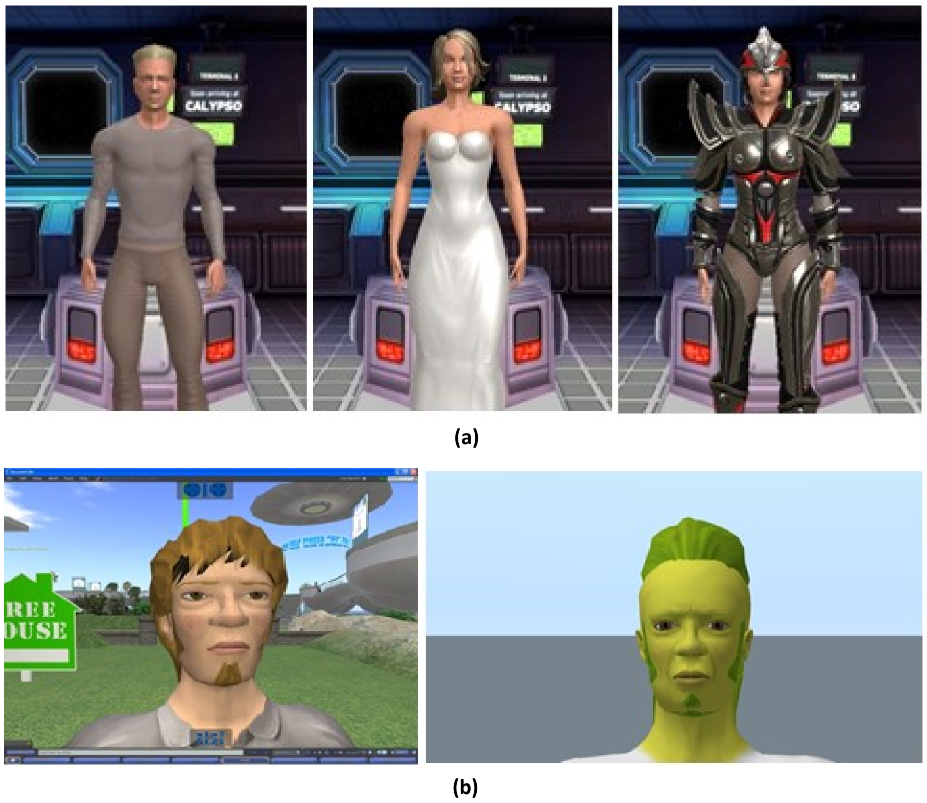


Figure 5: Avatar images from popular VW's (a) Entropia Universe (b) Second Life.

1.6 Local Directional Pattern (LDP)

In this section we introduce LDP which will help comprehend the technique used to classify human and avatar faces to test the security of the CAPTCHA.

LDP is a robust local facial feature descriptor introduced by Jabid et al. It is sensitive to non-monotonic illuminations and performs well in the presence of random noise. LDP computes the edge response values in different directions and uses these to encode the image texture. It assigns an 8-bit binary code to each pixel based on its neighboring edge response values in different directions. Kirsch masks, known to detect different directional edge responses, are used [15]. Its edge response masks in eight directions are shown in Figure 6.

$$\begin{bmatrix} -3 & -3 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & 5 \end{bmatrix} \begin{bmatrix} -3 & 5 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & -3 \end{bmatrix} \begin{bmatrix} 5 & 5 & 5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix} \begin{bmatrix} 5 & 5 & -3 \\ 5 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix}$$

East **(Mask M$_0$)** **North East** **(Mask M$_1$)** **North** **(Mask M$_2$)** **North West** **(Mask M$_3$)**

$$\begin{bmatrix} 5 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & -3 & -3 \end{bmatrix} \begin{bmatrix} -3 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & 5 & -3 \end{bmatrix} \begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & -3 \\ 5 & 5 & 5 \end{bmatrix} \begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & 5 \\ -3 & 5 & 5 \end{bmatrix}$$

West **(Mask M$_4$)** **South West** **(Mask M$_5$)** **South** **(Mask M$_6$)** **South East** **(Mask M$_7$)**

Figure 6: Kirsch edge response masks in eight directions [15].

For a central pixel the eight directional edge response values {m$_i$} are evaluated using the Kirsch masks in eight different orientations centered on its position. This helps obtain varying edge response values of which not all are significant. High response values indicate the presence of a corner or edge in a particular direction. These values indicating prominent directions are captured to generate the LDP code. A threshold value of top prominent directions are chosen to generate the LDP code. These values are set to 1 and the remaining (8-*k*) bits are set to 0. The equation below depicts the LDP code generation process [15].

$$LDP_k = \sum_{i=0}^{7} bit_i(m_i - m_h)2^i \tag{1}$$

$$bit_i(z) = \begin{cases} 1, & z \geq 0 \\ 0, & z < 0 \end{cases} \tag{2}$$

where, m_h is the k^{th} most prominent directional response.

Figure 7 depicts an outline of the eight directional responses positions with the LDP binary bit positions.

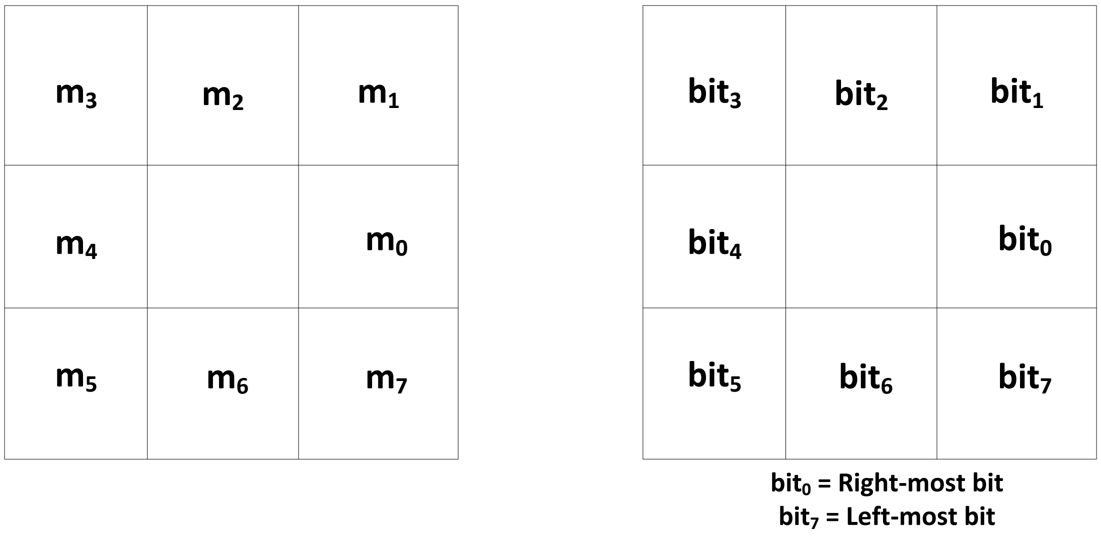


Figure 7: The eight directional responses positions $m_0 - m_7$ and the LDP binary bit positions $bit_0 - bit_7$.

Figure 8 illustrates an example of an LDP code with its decimal equivalent for threshold k=3

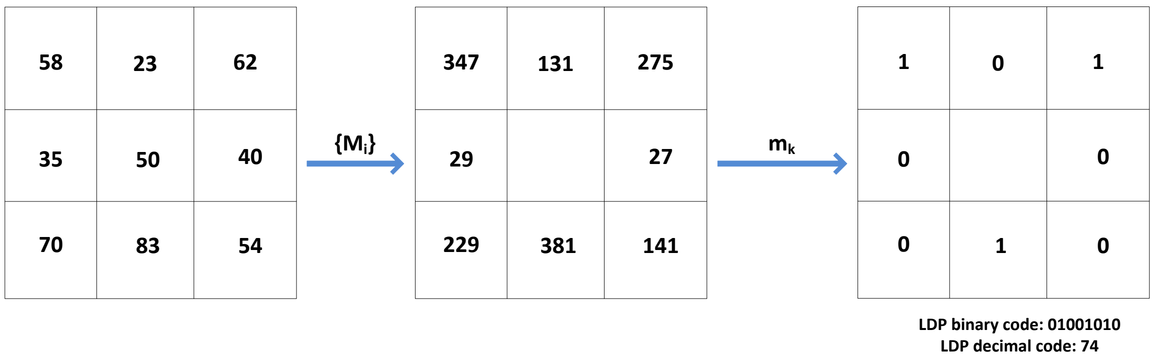
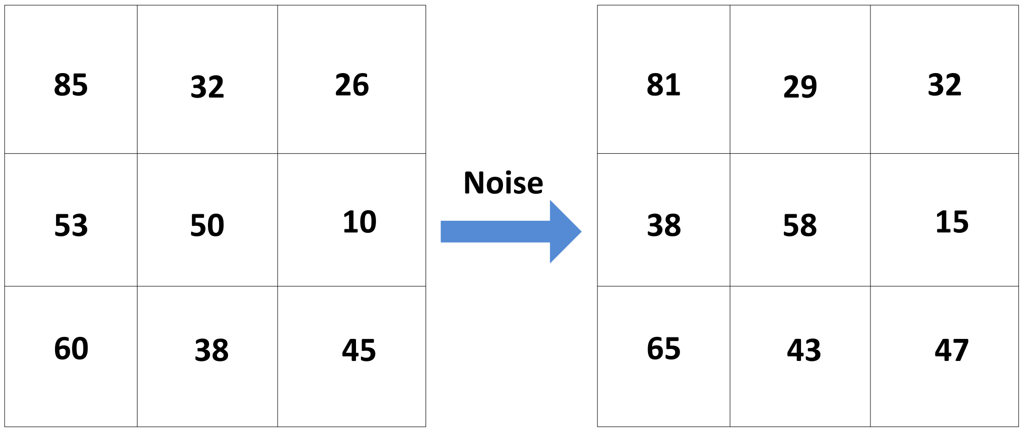


Figure 8: LDP code with k=3.

Edge response values are more stable than image intensity values. LDP provides the same pattern in presence of noise and non-monotonic illumination changes. Bit changes due to noise additions do not affect the LDP code patterns thus making it a robust and stable descriptor. An example is shown in Figure 9.



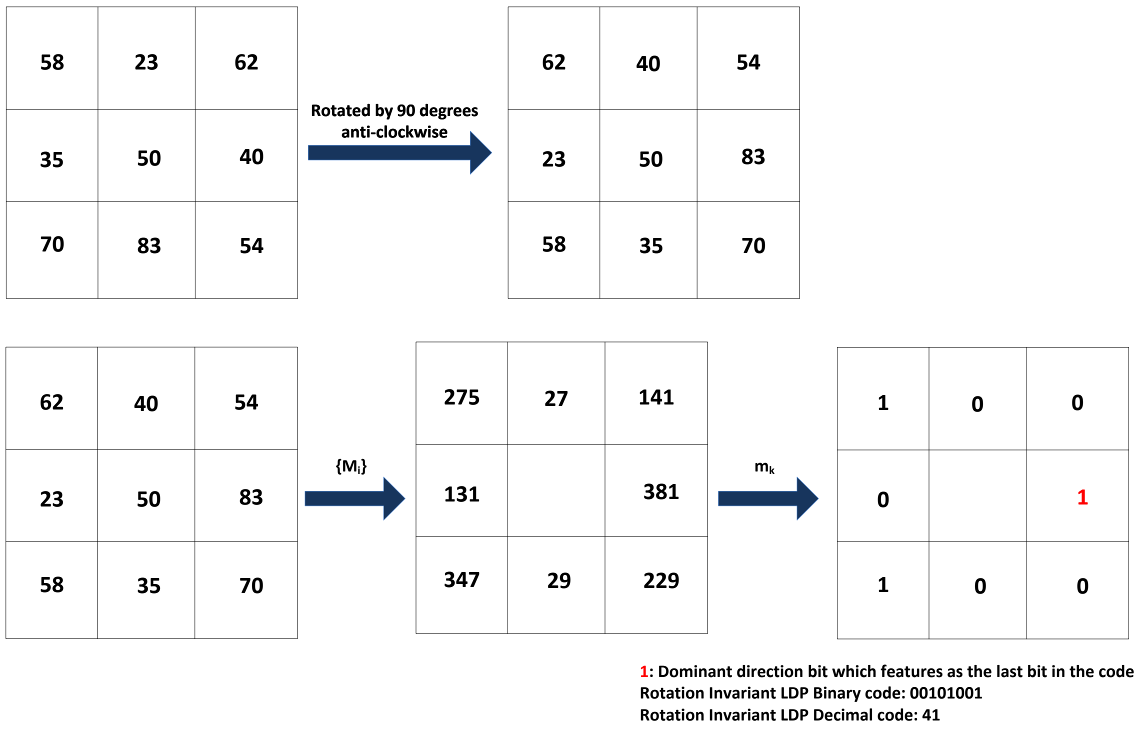
85	32	26
53	50	10
60	38	45

81	29	32
38	58	15
65	43	47

LDP binary code unaltered before and after noise: 00010011

Figure 9: Robustness of LDP code in presence of noise.

Another interesting property of LDP is rotation invariance. Rotational change of any image leads to alter its spatial intensity distribution. This results in a change in the edge response values of each direction, which in turn will consequently generate a completely different LDP code. However, on minutely observing the edge response values we can learn that the strongest edge response will be unaffected by the image rotation. Figure 10 illustrates a good example to explain this with a 90 degree anti-clockwise rotation of an image. Thus, rotation invariant LDP feature is obtained by applying circular shift operation on the original binary LDP code value. The highest edge response becomes the dominant direction of the code. The bit value associated with the dominant direction is shifted to the rightmost position. For example, if the bit positions in the original code are ‘pqrstuvw’, where each character signifies a bit location in the 8-bit code, and t is the dominant directional bit then t is shifted 3 bit positions to the right. Subsequently the rest of the bits are circularly shifted to the right by 3 places as well to generate the final code ‘uvwpqrst’. The LDP operator produces C_k^8 different LDP code values corresponding to the 8-bit binary patterns with k-bits equal to 1. However, after applying circular shift to achieve rotation invariance the LDP code reduces to C_{k-1}^7 . Thus, the histogram bin count reduces from C_k^8 to C_{k-1}^7 [15].



58
23
62
35
50
40
70
83
54
Rotated by 90 degrees anti-clockwise
62
40
54
23
50
83
58
35
70
62
40
54
23
50
83
58
35
70
{Mi}
275
27
141
131
381
347
29
229
mk
1
0
0
0
1
1
0
0
1: Dominant direction bit which features as the last bit in the code
Rotation Invariant LDP Binary code: 00101001
Rotation Invariant LDP Decimal code: 41

Figure 10: Rotation invariant LDP calculation.

The LDP image descriptor used is a histogram. The histogram for the entire image is obtained by concatenating the individual histograms extracted from image regions. Figure 11 shows an example. It is based on the occurrence of LDP features. After computing the LDP code for every pixel of an image *I(x,y),* the image is subdivided into blocks of regions. Histograms are extracted from these individual, local regions. These regional histograms are finally concatenated into one single histogram. This histogram H serves as a global feature LDP descriptor of the entire image *I* of size *M* x *N* with τ as the LDP code value.

$$H(\tau) = \sum_{r=1}^{M} \sum_{c=1}^{N} f(LDP_k(r,c), \tau) \tag{3}$$

$$f(a,\tau) = \begin{cases} 1, & a = \tau \\ 0, & otherwise \end{cases} \tag{4}$$

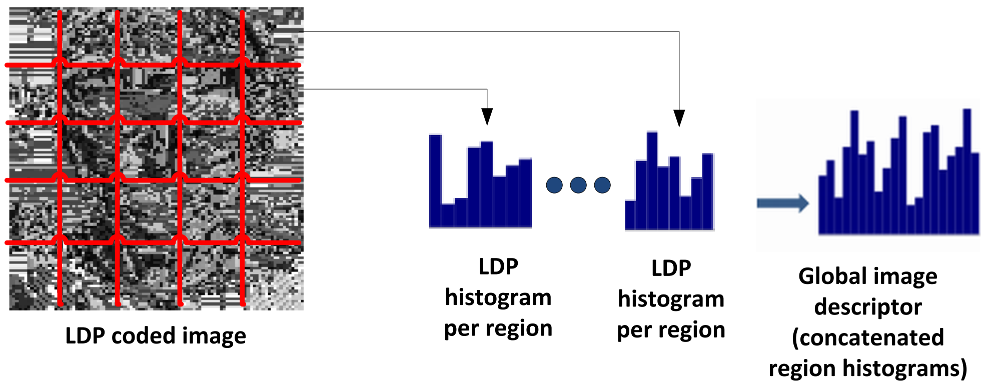


Figure 11: LDP image descriptor.

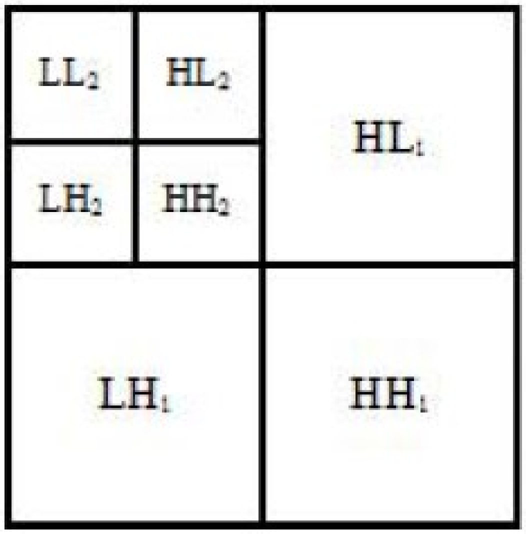
1.7 Discrete Wavelet Transform (DWT)

Here we briefly introduce the Discrete Wavelet Transform (DWT) that will help comprehend the technique used to classify human and avatar faces to test the security of the CAPTCHA.

DWT is a very popular tool in the field of image processing. It helps to view and process digital images at multiple resolutions. Its mathematical background and advantages have been discussed in many research articles [16]. The chief advantages of using WT are:

- It decomposes an image by reducing the sub-image resolutions thus reducing the computational complexity of the system.
- It decomposes images into sub-bands corresponding to different frequency ranges. They easily meet the requirements for the next major step and thereby reduces the computational overhead of the system.
- It provides local information in both spatial and frequency domains which helps obtain the spatial and frequency characteristics of an image at the same time.

The main characteristic of wavelets is they provide multi resolution analysis of the image in the form of co-efficient matrices. Strong arguments for multi-resolution decomposition in psychovisual research support evidence that humans process images in a multi-scale way . Figure 12 shows an example of an avatar image with its corresponding structure of the wavelet coefficient [17].

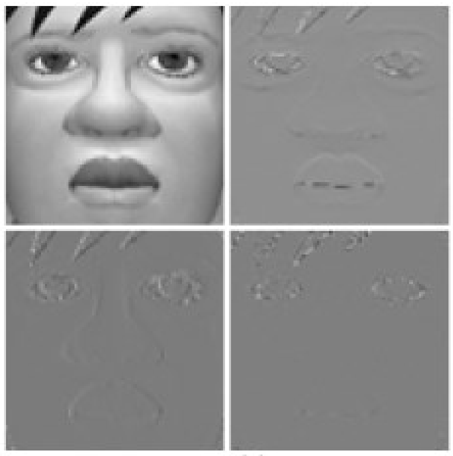


(a)

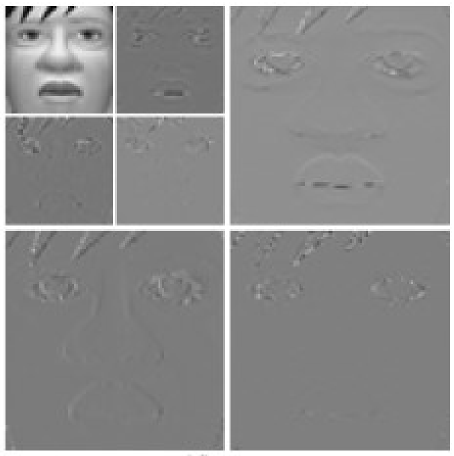
(b)

Figure 12: (a) Wavelet coefficient structure (b) A sample avatar image.

Figure 13. depicts the first and second level of wavelet decomposition on a sample avatar face image [17].



(a)



(b)

Figure 13: (a) One level wavelet decomposition (b) Two levels wavelet decomposition.

The two-dimensional WT is obtained by applying a one-dimensional WT to the rows and columns of the two-dimensional image data. The one-level wavelet decomposition of an image results in an approximation image (LL1) and three detail images in the horizontal (HL1), vertical (LH1) and diagonal (HH1) directions respectively. The approximation image contains the low frequency information of the facial image which is subsequently used for the next level of decomposition. The detail images contain most of the high frequency information of the facial image. Thus, the original image is decomposed and represented by a set of sub-images at several scales [17].

1.8 Behavioral Biometrics: Mouse Dynamics

Here we briefly introduce the concept of Behavioral Biometrics along with its sub-category of human computer interaction with the mouse as the input device known as Mouse Dynamics.

In today's Internet world user profiling or user identification has gained a lot of significance. Identity thefts, caused by misusing someone's personal information, can be prevented by providing authorized access to resources. To achieve timely response to a security breach frequent user verification is needed. However, this process must be passive, transparent and non-obtrusive. In order for such a system to be practical, it must possess the following features:

- Accuracy: Accurately identify an imposter by rejecting access to resources as well as never reject a genuine user to get through.
- Efficiency: Quick decisions to distinguish users in a timely manner.
- Difficult to forge: Nearly impossible for an impostor to mimic a user's biometric profile and defeat the verification system [18].

Behavioral biometric systems are usually less prominent and only muscle based controls such as keystrokes, signature and gait are well analysed. These systems provide numerous and significant advantages over traditional biometric systems. They can be collected non-obtrusively. Collection of behavior data is cost-effective as it requires no special hardware. While these systems are not unique enough to provide reliable human identification, they have shown to be highly accurate in identity verification. In accomplishing everyday tasks, human beings use different styles, strategies, apply unique skills and knowledge, etc. These define the behavioral traits of the user. Behavioral biometrics attempts to quantify these traits to profile users and establish their identity. Human computer interaction (HCI)-based biometrics comprise of interaction strategies and styles between a human and a computer. These unique user traits are quantified to build profiles for identification. A specific category of HCI-based biometrics is based on recording human interactions with mouse as the input device and is known as Mouse Dynamics [19].

By monitoring the mouse usage activities produced by a user during interaction with the GUI, a unique profile can be created for that user that can help identify him/her. These mouse actions include drag and drop, point and click and silence (absence of movement). From these a set of features such as average speed, velocity, distance travelled, etc. can be extracted. Mouse event data can be categorized into mouse wheel movements, clicks, menu and toolbar clicks, single click, double click, etc. This mouse dynamics based biometric scheme has been extended to online games as well. The extracted features involve x and y co-ordinates, mouse velocity, tangential velocity, tangential acceleration and angular velocity [19]. Mouse-based verification approaches do not record sensitive user credentials like usernames and passwords. Thus, they avoid privacy issues. However, these approaches have resulted in either unacceptably lower accuracy rates or longer verification times. Environmental factors such as different machines, operating environments, mice, screen resolution, user's physical and mental state, etc. play a significant impact in accurate feature extraction for accurate identification. Thus, mouse-based biometric approaches have to overcome these limitations and be more robust across different platforms and environments [18].

Mouse dynamics differ by how users move and click as well as where they click on the screen. Mouse metrics such as speed, acceleration, velocity, pause and click, time between clicks, mouse wheel movements, etc. are all hardware dependent i.e. they depend on the nature of the mouse. These metrics are less reliable and stable towards accurately determining the identity of the user. Fine-grained angle-based metrics are devised to compensate for these limitations. These newly defined metrics can accurately characterize a user based on their unique mouse moving behaviors, independent of the operating platform. These metrics are:

- Direction: For any two consecutive recorded points A and B, the direction traveled along AB from the first point to the second is recorded. The direction is defined as the angle between the line AB and the horizontal line.

- Angle of Curvature: For any three consecutive recorded points A, B and C, the angle of curvature is the angle between the lines from AB and BC.

- Curvature Distance: For any three consecutive recorded points A, B and C, consider the length of the line AC. The curvature distance is the ratio of the length of AC to the perpendicular distance from point B to AC. It has no units since it is a ratio of two similar quantities [18].

Figure 14 illustrates the angle-based metrics which are unique across users. Not only does the same user have very similar angle-based results on different platforms, different users have different angle-based results even on similar platforms. Moreover, these metrics are platform independent.

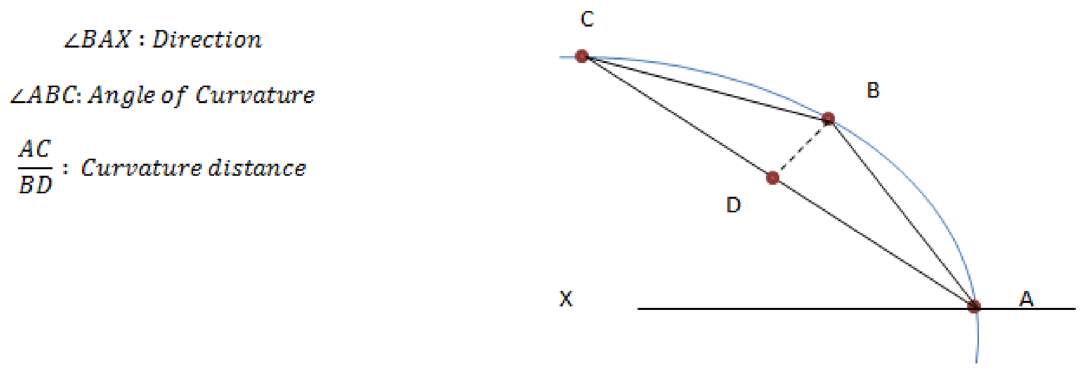

$\angle BAX$: Direction
$\angle ABC$: Angle of Curvature
$\frac{AC}{BD}$: Curvature distance
C
B
D
X
A

Figure 14: Illustration of angle-based metrics [18].

1.9 Support Vector Machines (SVM)

Support Vector Machines (SVM) is a popular, two-class, data classification technique in the machine learning domain. It is a supervised learning model i.e. it infers a function that maps unseen data samples based on its learning from a labeled training dataset. It analyzes data and recognizes patterns. Given a set of labeled training samples, each belonging to one of two classes, an SVM training algorithm builds a model that assigns a new samples into one class or the other. Thus, SVM serves as a binary classifier that implements the following idea: if the data are not linearly separable in that space, it maps the input training feature vectors into some high dimensional feature space through some non-linear mapping chosen apriori. In this space, a linear decision surface (hyperplane) is constructed that ensure high generalization of the network. A hyperplane achieves a good separation if it has the largest distance to the nearest training data point of any class, since the larger the margin the lower the generalization error i.e. the distance between the error on the training and the test set. Such an hyperplane is known as maximum-margin hyperplane and is ideal for classification [20, 21]. Figure 15 shows an example of two hyperplanes separating data points from two classes of which one is the maximal-margin hyperplane.

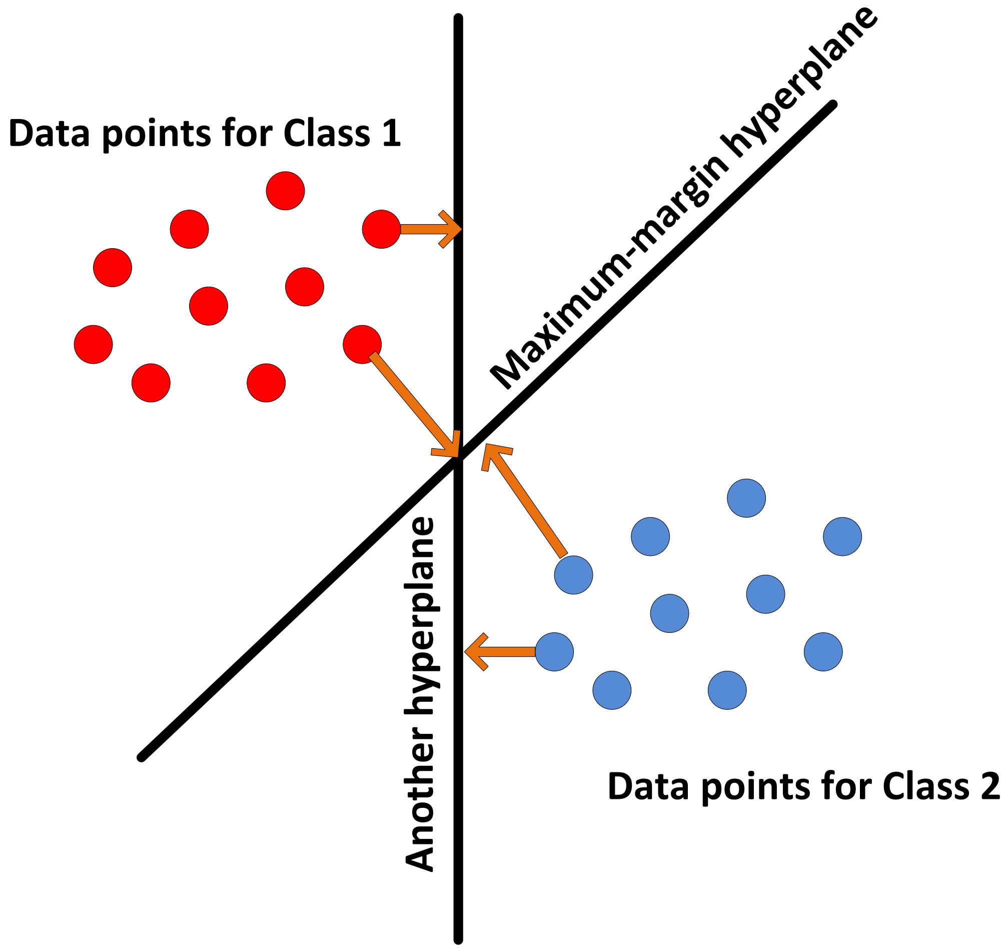


Figure 15: Hyperplanes separating data points using SVM.

Given some training data D, a set of n points of the form

$$D = \{(x_i, y_i) \mid x_i \in \mathbb{R}^{'},\ y_i \in \{-1, 1\}\}_{i=1}^{n} \tag{5}$$

where each x$_i$ is a p-dimensional vector and y$_i$ is either 1 or -1, indicating the class to which the point x$_i$ belongs. Any hyperplane can be written as

$$w \cdot x - b = 0 \tag{6}$$

where, $\cdot$ denotes the dot product and w denotes the normal vector to the hyperplane. The parameter $\frac{b}{\|w\|}$ determines the offset of the hyperplane from the origin along w.

If the training data are linearly separable, we can select two hyperplanes that separate the data and there are no points between them, and then try to maximize their distance. The region bounded by them is called “the margin”. These hyperplanes can be described by the equations:

$$w \cdot x - b = 1 \tag{7}$$

$$w \cdot x - b = -1 \tag{8}$$

By geometry, the distance between these two hyperplanes is $\frac{2}{\|w\|}$. Thus, we want to minimize $\| w \|$ as well as prevent data points from falling into the margin, the following constraint for each i can be written as

$$y_i(w \cdot x_i - b) \geq 1, \; for \; all \; 1 \leq i \leq n \tag{9}$$

Putting this together we get the optimization problem: Minimize $\| w \|$ in (w, b) subject to $y_i(w \cdot x_i - b) \geq 1, \; for \; all \; 1 \leq i \leq n$ [21]

Figure 16 shows an example of the maximum-margin hyperplane and margins for an SVM, trained with samples with two classes.

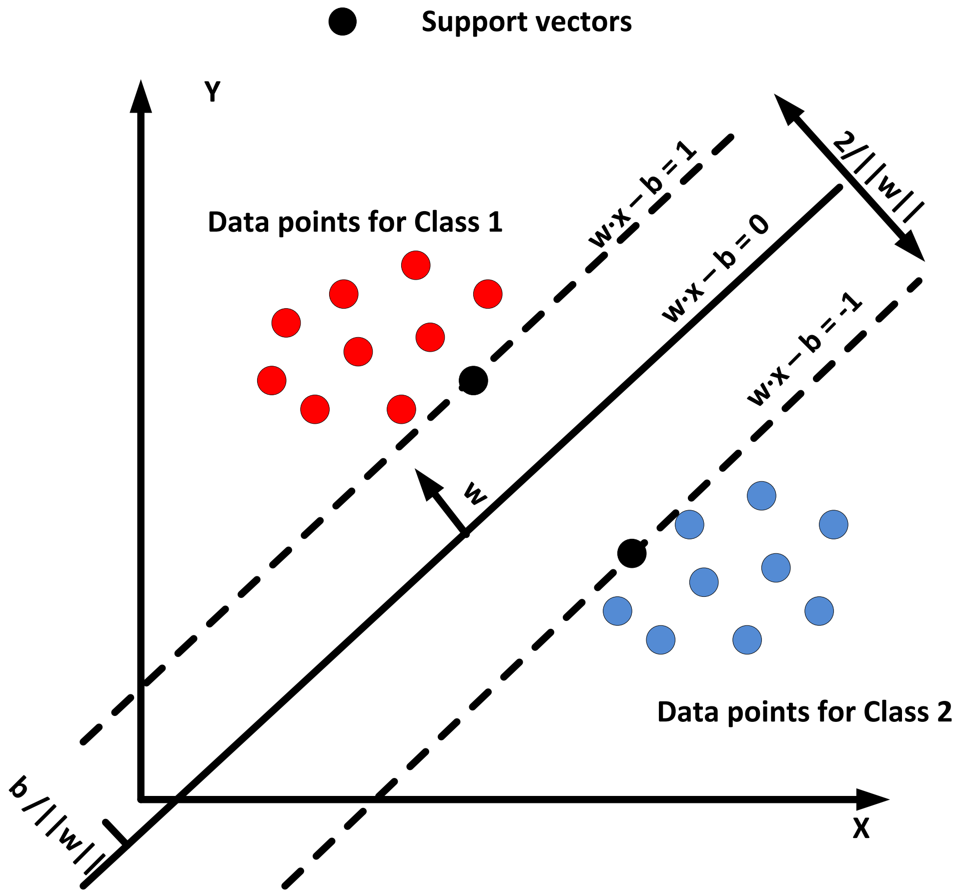


Figure 16: Maximum-margin hyperplane and margins for SVM trained with samples from two classes.

To keep the computational load reasonable, the mappings used by SVM are designed to ensure that dot products may be computed easily in terms of the variables in the original space by defining them in terms of a kernel function $K(x, y)$. There are four basic kernels used in SVM. For linear classification a linear kernel is used. However, it is not always possible to linearly classify the data. For this a kernel trick is applied to the maximum-margin hyperplane to achieve non-linear classification. This allows the algorithm to fit the maximum-margin hyperplane in a transformed feature space. An example is shown in Figure 17. The kernels used to achieve this are the Polynomial kernel, Radial Basis Function (RBF) kernel and the Sigmoid kernel. The mathematical equations of each kernel is stated below:

- Linear kernel: $K(x, y) = x^T y$
- Polynomial kernel: $K(x, y) = (\gamma x^T y + r)^d$, $\gamma > 0$

- Radial Basis Function kernel: $K(x, y) = e^{(-\gamma\|x-y\|^2)}$, $\gamma > 0$
- Sigmoid kernel: $K(x, y) = tanh(\gamma x^T y + r)$

Here, $\gamma, r\ and\ d$ are kernel parameters [22].

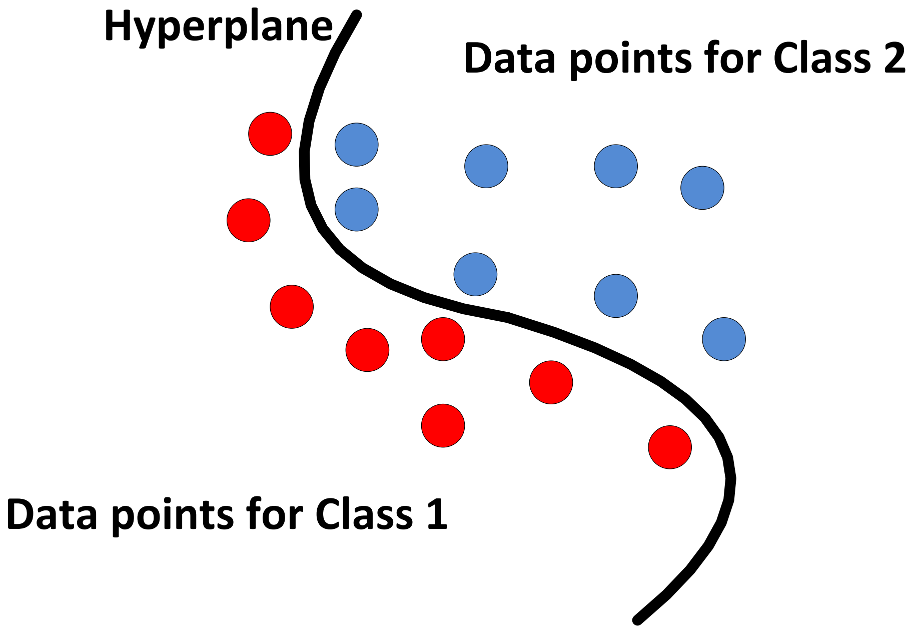


Figure 17: Non-linear classification using SVM.

CHAPTER II
RESEARCH TRENDS

In this section we highlight the significant contributions in the domains of CAPTCHAs and their security, Local Directional Pattern (LDP), Behavioral Biometrics and Mouse Dynamics.

2.1 CAPTCHAs

CAPTCHA is an acronym for Completely Automated Public Turing Tests to tell Computers and Humans Apart. It is used to verify human users from malicious bots. The concept of the Turing test stems from [2] and the concept "Can machines think?". A collection of published research works in the domains of HIP and CAPTCHA systems together with CAPTCHA schemes, attacks and analysis, applications and their usability issues are highlighted in [23]. A game is described in which a human interrogator has to distinguish between responses to certain queries from a human user and a machine located in separate rooms apart from that of the interrogator. The Turing test was first utilized by Moni Naor to verify the presence of a human user, and not a bot [24] . Alta Vista patented a similar idea in 1997 as one of the first CAPTCHA techniques. A formal definition of CAPTCHA was coined in 2000 by Manuel Blum and Luis Von Ahn at Carnegie Mellon University [3]. They apply a reverse Turing Test principle to design CAPTCHAs that are tests generated and graded by computers to identify the human user [25]. Baird has generated some CAPTCHA techniques together with describing the evolution of Human Interactive Proofs (HIPs) Research and Development as challenge/response protocols to allow humans to authenticate themselves [26]. He has also discussed using HIPs to defend web services against abuse by bots by using them to distinguish between human users and bots [27].

2.1.1 Text CAPTCHAs

Text CAPTCHAs are the most popular and widely deployed. Renowned organizations such as Microsoft, Yahoo, Google, etc. have their own versions of this form of CAPTCHA.

Pessimal Print, a variant of the Turing test, uses low-quality images of machine-printed text synthesized pseudo-randomly over certain ranges of words and shown to be illegible to Optical Character Recognition (OCR) attacks [28]. BaffleText uses non-English pronounceable words and Gestalt-motivated image-masking degradations to defend against dictionary and image restoration attacks respectively [29]. ScatterType is designed to resist character-segmentation attacks by generating pseudo-random synthesized images of text strings in which the characters are fragmented using horizontal and vertical cuts and the fragments are scattered by vertical and horizontal displacements [30]. Mathematical theory of assurance is used to ensure that the probability of a correct response to a CAPTCHA is not just a lucky guess and the most common types of challenge strings are examined for weaknesses [31]. Random words and unfamiliar text have been used to generate CAPTCHA challenges with a more uniform distribution of difficulty by balancing image degradations against familiarity [32]. The masking characteristics of the Human Visual System is adequately utilized by picking random English alphabets for CAPTCHA challenges with added noise to deceive the bot by randomly changing the visibility of characters for humans [33]. Human friendly HIPs have been designed to find out the visual distortions most effective at foiling computer attacks without hindering humans [34] as well as introducing segmentation-based reading challenges to help build stronger human-friendly HIPs [35]. There are other CAPTCHAs with non-English word based challenges which aid the non-native English users [36, 37, 38, 39]. Semantic CAPTCHAs are designed to incorporate perception or cognition of human users to understand the semantic differences in the challenges before solving them thus, giving it an advantage over machines [40]. Visual word-based CAPTCHA using 3D characters with 3D boundaries delimited by shadows have been designed as well. The shadows are obtained from different light effects to further enhance security against automatic character recognition tools [41]. Figure 18 shows samples of few of these text CAPTCHAs.

There have been attacks carried out to break text CAPTCHAs. One of the first renowned attack was on the EZ-Gimpy CAPTCHA using shape context matching that identified the challenge word with an accuracy of 92% and the requisite 3 words in the Gimpy CAPTCHA image 33% of the time [42]. Some other attacks on the Gimpy CAPTCHA used representative shape contexts and shapemes to determine similar shapes [43], using correlation with an accuracy of 99% on the EZ-Gimpy challenge and direct distortion estimation to identify the four letters in the Gimpy-r challenge image 78% of the time [44] and using shape matching and chamfer matching for correlation [45]. Two low cost-methods using Affine Moment Invariants and a naive, trivial technique have been employed to break CAPTCHAs used to protect SMS gateways of two mobile operators [46]. Simple pattern recognition algorithms were used to exploit fatal design errors in the CAPTCHA generation schemes, believed to be resistant to OCR attacks, with a near 100% success rate [47]. HIPs were identified to be pure recognition tasks and were found to be easily broken using machine learning algorithms, thus enabling to build effective HIPs by designing challenging segmentation tasks to confuse the algorithms [48]. A simple, low-cost segmentation attack was carried out on the Microsoft CAPTCHA with an overall success rate of about 60% [49].

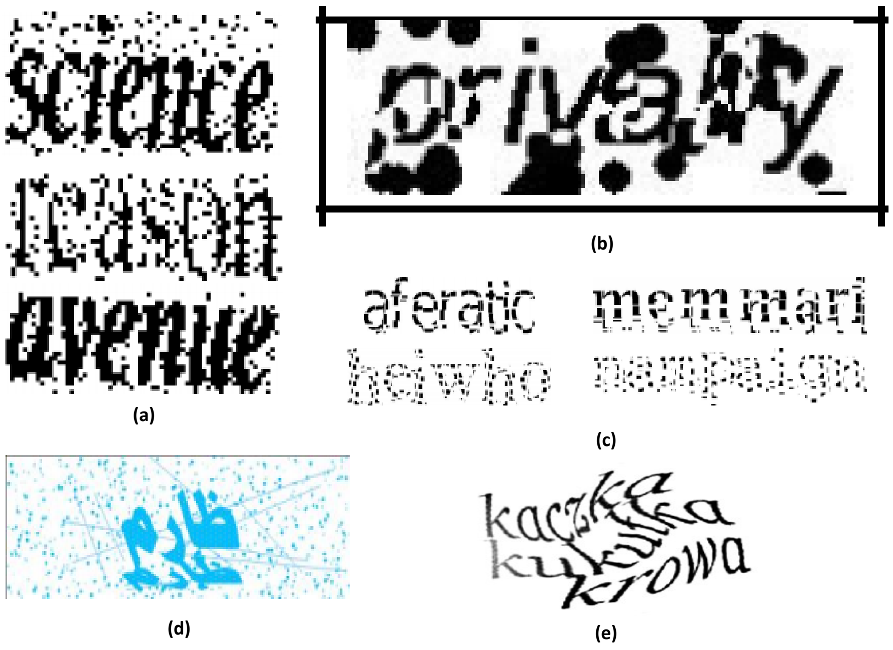


Figure 18: Samples of Text CAPTCHAs (a) Pessimal Print [28] (b) BaffleText [29] (c) ScatterType [30] (d) Arabic CAPTCHA [36] (e) Semantic CAPTCHA [40].

2.1.2 Image CAPTCHAs

Image CAPTCHAs have been used as an alternative to text CAPTCHAs. Implicit CAPTCHAs extend the traditional Text CAPTCHA usage domain wherein challenges can be answered with a single click, can be answered through contextual experiences and easy enough for humans to pass and detect failure attempt by a bot [50]. CAPTCHA challenges can also be designed based on naming, distinguishing and identifying anomalies between images [51]. ESP, Peekaboom, Phetch and Verbosity are some online computer games that utilize and constructively channel human processing power to solve problems that computers cannot [52]. Microsoft's ASIRRA (Animal Species Image Recognition for Restricting Access) is an HIP that request users to identify images of cats and dogs [53]. Google's What's Up CAPTCHA, based on image orientation, requests users to set a bunch of rotated images in their upright positions [54]. More robust and user-friendly CAPTCHA systems have been developed. Images in challenges are randomly distorted before presenting them to the user. IMAGINATION (Image Generation for Internet Authentication) CAPTCHA is attack-resistant and user-friendly in which controlled distortions are produced on randomly chosen images and requests users to annotate them. The distortions applied have low perceptual degradation and high resistance to attacks by content-based image retrieval systems [55]. CAPTCHA challenges are also designed using human faces [56], matching distorted faces of several different human subjects [6], finding human face image pairs [57], detection of visually distorted human faces embedded in a complex background [58], distinguishing between human and avatar faces [59] as well as identifying gender of face images [60]. Three-dimensional (3D) character images [61], 3D animation [62] and interactive 3D Flash-based [63] CAPTCHA challenges have been designed to extend the two-dimensional (2D) domain in an effort to thwart OCR attacks. Most visual based HIPs require a database of labeled images whose creation is expensive and time consuming. An approach to solve this is to create games [52] and use image search engines [64]. Image CAPTCHAs have been also combined with graphical passwords for user authentication [65]. Some other forms and variants of CAPTCHAs include themes based on humorous cartoons to make it an enjoyable experience to the user by focusing on the human ability to understand humor [66], solving jigsaw puzzles as challenges [67] , determine the logical sequence of images based on their tags [68], etc. There have been image CAPTCHAs designed for touchscreen devices such as PDA's and mobile phones too [69, 70, 71]. Figure 19 shows a few samples of Image CAPTCHAs.

There have been attacks carried out against Image CAPTCHAs as well. The semantic gap i.e. limitations of automatic methods for image retrieval as compared to humans is narrowed by motivating users to provide semantic image annotations [72]. The IMAGINATION CAPTCHA is attacked by detecting candidate rectangles to detect the object boundaries and by examining the consistency of the interface of each rectangle with its neighboring rectangles to determine the highest likelihood. Besides this, computers are typically good at extracting low-level features such as color, shape, texture, color layout, etc. and high-level semantics associated with perception or interpretation of semantically meaningful objects contained in an image and their relationships [73]. The ESP-PIX CAPTCHA is attacked by first scanning the images through an OCR system for tags and then compare the challenge images with the images associated with the four given tags for correlation probabilities. Then it combines the results of the tag and image correlation probabilities to compute the highest possible probability for the correct response challenge [74]. A combination of support-vector machine (SVM) classifiers are trained on color and texture features extracted from images of cats and dogs of the ASIRRA CAPTCHA to solve it with 80.6% accuracy [75]. Moreover, another attack on ASIRRA using Hierarchical Temporary Model (HTM) (a form of neural networks) yielded an accuracy of 74.7% [74]. A machine learning attack was carried out on ARTiFACIAL which comprised of detecting the face in the challenge and then locating the six facial corner points on the face [73]. Image recognition and machine learning algorithms are used to break the HumanAuth CAPTCHA system [76].

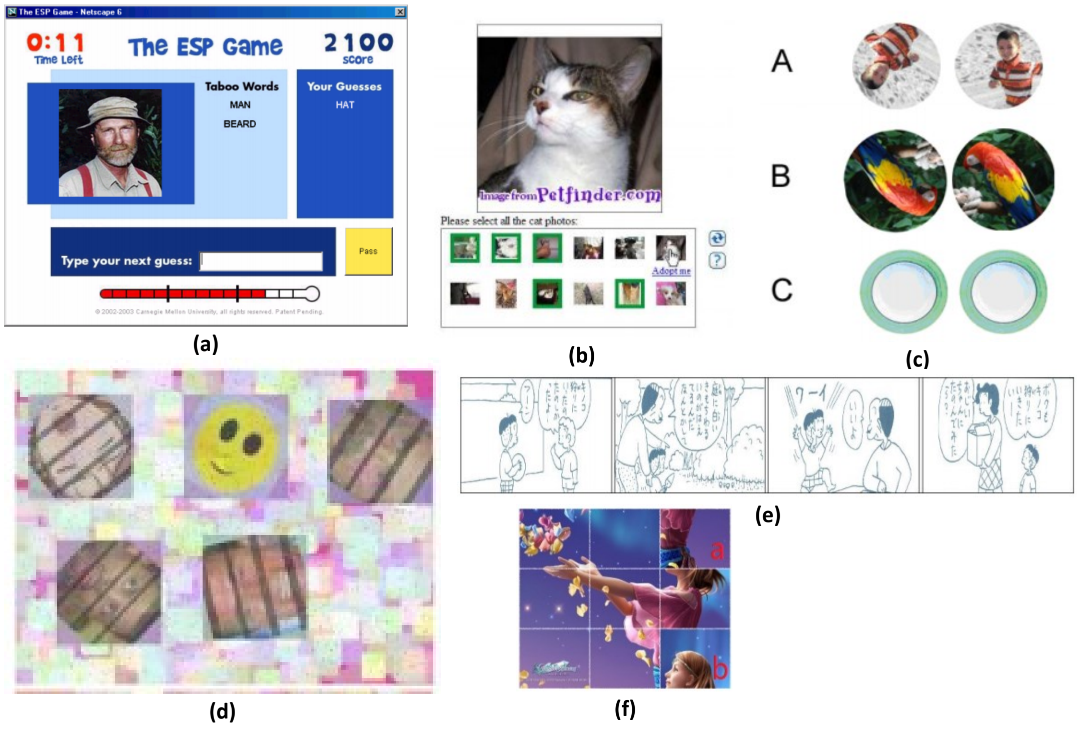


Figure 19: Samples of Image CAPTCHAs. (a) ESP [52] (b) ASIRRA [53] (c) What's Up [54] (d) FaceD [58] (e) Cartoon [66] (f) Jigsaw puzzle [67].

2.2 Local Directional Patterns

Local Directional Patterns (LDP) describes the local features of an image. Experimental results on the Brodatz texture database show that it impressively outperforms other dense descriptors such as Gabor-Wavelet and Local Binary Patterns (LBP) [15]. Few applications of this technique involve local feature extraction from depth silhouettes for human gait recognition by processing depth videos from a depth camera [77], fingerprint image texture extraction for matching [78]. LDP has also been applied towards analyzing facial images. Some examples include classifying natural and artificial faces in combination with wavelets [79], represent facial geometry and analyze its performance in facial expression recognition [80], etc. An enhanced version of Local Directional Pattern (ELDP) adopts local edge gradient information of facial images towards face recognition [81]. LDP has also been applied towards gender classification as well [82].

2.3 Discrete Wavelet Transform

The Discrete Wavelet Transform (DWT) is a very popular tool in the field of image processing. It helps analyze images at multiple resolutions. High frequency components are studied with sharper time resolutions than low frequency components [83]. An image-watermarking technique is developed with DWT and Singular Value Decomposition (SVD) [84]. Wavelet decomposition is used to diagnose fault in industrial induction machines by extracting health information of a system through signals and detecting different electrical faults [85]. Satellite image resolutions have been enhanced using DWT as well [86]. DWT has also been applied towards analyzing facial images. Face recognition is a popular application. Few such examples involve recognizing faces under varying lighting conditions [87], building a face recognition system in combination with fast Principal Component Analysis (PCA) [88], etc. Radon and Wavelet Transforms are combined to extract key facial features for gender classification from facial images [89]. The domain of face classification and recognition has also been extended beyond human faces to virtual (artificial) avatar faces. Few examples include classification of natural and artificial faces with LDP [79], recognizing avatar faces using DWT and eigenfaces [90] and using hierarchical multi-scale LBP [17].

2.4 Behavioral Biometrics: Mouse Dynamics (MD)

Several different usages and techniques for MD have been proposed. The MD analysis can be divided into two categories: static analysis and continuous analysis. The former analyzes mouse behavior at some particular moment e.g. login time, game-solving, etc. whereas the latter analyzes mouse behavior throughout the course of interaction [91].

- **Static analysis:** A user identification system, based on signature writing is presented in [92]. It utilizes new signature writing parameters and verifies the data using geometric average means and updates its database dynamically. An online, hybrid, Java-based internet biometric authentication system is described in [93]. It requires authenticity confirmation from a typing style test and a signature match and achieves a fraudulent success rate and authentic users access rate of approximately 4.4% and 99% respectively. A two-phase authentication model based on enrollment and verification is presented in [94]. It enroll users by requesting them to move the mouse and follow a sequence of dots presented on the screen, later verifying them during the login phase when they perform the same action. Biometric traits based on user's interaction with a web page is extracted in [95]. Here the online environment of entering a virtual pin in an Internet login page is simulated through a memory game-like environment by identifying matching tiles to uncover hidden patterns. A graphical authentication system, dubbed Mouse-lock, is presented in [96], which deploys the analogy of a safe and the password is entered via the mouse in a graphical equivalent of a combination lock. A login system requesting users to follow a maze is described in [97]. The task consists of navigating the mouse pointer between two lines from a start point to an end point based on which mouse features are extracted to authenticate that user. An application for authenticating users is presented in [98]. In here, the mouse features are extracted from nine paths traced by users between seven squares displayed consecutively on the screen.

- **Continuous analysis:** User interaction via the mouse is captured and the behavioral information is used for user identity authentication. Statistical pattern recognition techniques are used to classify the interactions as genuine or imposter [99]. A re-authentication model based on continuous monitoring of the user's behavior to flag anomalous user behavior is presented in [100]. This model applies a supervised learning technique and raises an alarm if the current user behavior deviates sufficiently from its learned behavior. A biometric system based on mouse curves (mouse movements with little or no pause between them) is presented in [101]. These curves are individually classified and used to develop classification algorithms to characterize the user's mouse usage activities during an entire session. Three sets of experiments are carried out to model the behavioral characteristic of users based on their mouse data. The main experiment reproduces real operating conditions and the other examine confounding factors arised from the main experiment by fixing the environment variables [102]. Fine-grained, angle-based metrics for mouse movements are used for user verification in [18]. These metrics are relatively unique for each user and platform independent. A pattern-growth based mining method to extract frequent-behavior segments to obtain stable mouse characteristics is described in [103]. Here, one-class classification is employed to achieve continuous user authentication. Users are authenticated per mouse event performed on their system in this model [104]. Here, MD data of forty nine users are used and the system performance is evaluated with six machine learning algorithms.

The domain of Behavioral Biometrics (BB) has been extended towards verifying and recognizing malicious software agents such as bots. As such programs draw closer to the abilities and intelligence of human beings the need is to segregate between human and bot activities over the Internet. This research is known as *Artimetrics* after the word "artiliect", a shortened version of "artificial intellect" [105]. BB has also been used for intrusion detection and online gaming. A system for verification of online poker players based on a behavioral profile representing the statistical model of player's strategy and its expansion to verify artificial poker playing agents has been proposed in [106]. The abuse of online games by game bots for gaining unfair advantage has plagued several online game players. To counter this a continuous game bot detection strategy has been devised differentiating bots from human players by passively monitoring input actions that are difficult for current bots to perform in a human-like manner [107]. Blog bots post comments to blog sites often including spam or other malicious links. An effective defense approach is devised using BB, primarily mouse and keystroke dynamics, to distinguish between human and bots based on passive monitoring and their behavioral differences [108]. Notable work has also been carried out to assist human users in identifying who they are interacting with on Twitter i.e. a fellow human user, a bot or a cyborg (bot-assisted human or human-assisted bot) by observing the difference in terms of their tweeting behavior, tweet content and account properties [109].

CHAPTER III
INITIAL DESIGN & SECURITY EVALUATION

3.1 Avatar CAPTCHA

Our first approach builds a prototype for an image-based CAPTCHA. The aim is to build a CAPTCHA using grayscale biological (human) faces and non-biological (avatar) faces with a challenge requesting users to identify the latter from a set of 12 images . Avatar faces resemble human faces which makes it a significantly challenging problem for bots, who are good at detecting faces, to distinguish between the two. The CAPTCHA was solved fairly successfully. Good feedback was obtained from the users as well on this prototype. Several users preferred solving image CAPTCHAs and rated this one positively as they found it easy to solve. Inspiration is drawn from Microsoft's ASIRRA CAPTCHA [53] and Luis von Ahn's art of harnessing human capabilities to address problems that computers cannot solve [52]. Faces are easily identifiable and distinguished by the human eye as demonstrated by the results below. Moreover, to the best of our knowledge, there is no work in the area of distinguishing grayscale images of human and avatar faces. Our CAPTCHA comprises of 2 rows with 6 images each. These images are randomly picked from human and avatar face datasets. Each image has a checkbox associated with it for the user to select/deselect them . There are Refresh and Submit buttons at the bottom of the page. A snapshot is shown in Figure 20. Refresh helps to randomly pick a new set of 12 images. On hitting Submit the challenge is validated. The goal is to select all the avatar faces in the challenge.

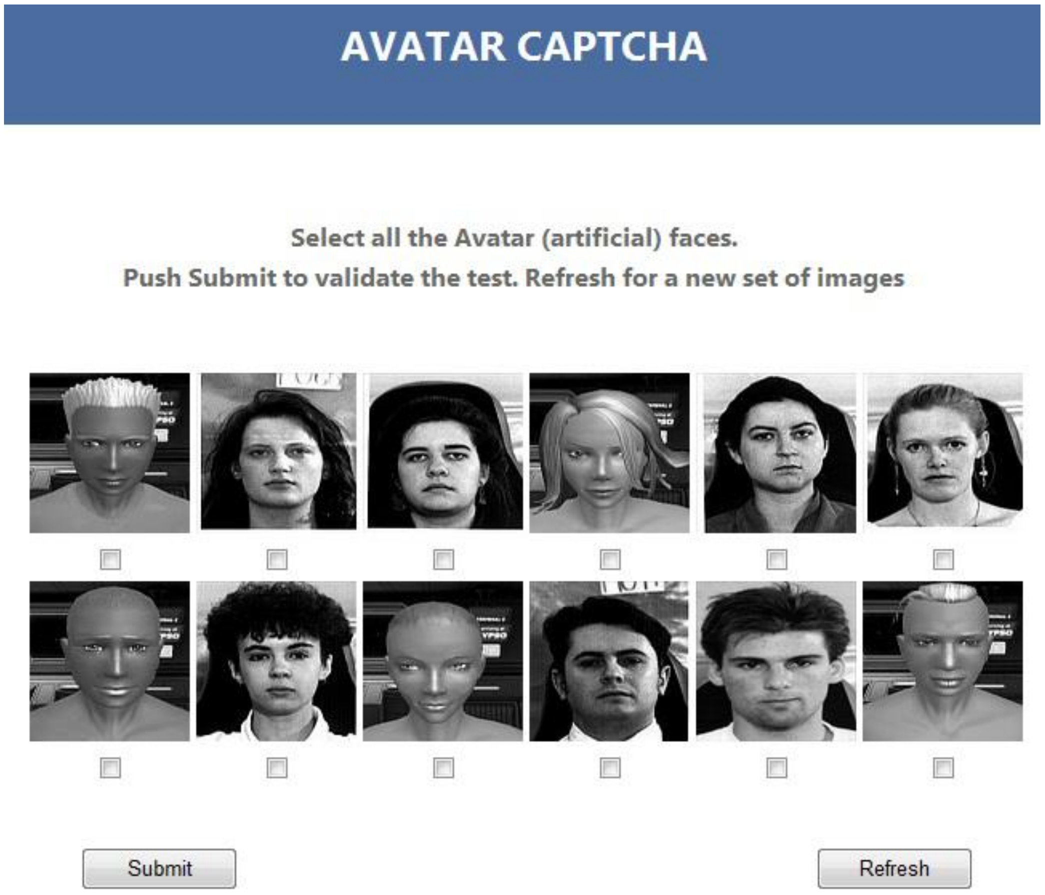

AVATAR CAPTCHA
Select all the Avatar (artificial) faces.
Push Submit to validate the test. Refresh for a new set of images
Submit
Refresh

Figure 20: Snapshot of the grayscale Avatar CAPTCHA [59].

The images are converted to grayscale before being displayed to prevent malicious computer programs (bots) to breach it by taking advantage of the varying color spectrum difference between the images. The user's choices are validated to prevent unauthorized access to bots. The architecture of the CAPTCHA is based on the popular client-server architecture as seen in Figure 21, where the client machine (browser) requests the server for an authentication service. The server randomly picks 12 images of humans and avatars from the database. Of these, 5 or 6 are avatar images. The server then transfers these images to the client. Their selection is verified by the server which in turn notifies the client to validate the user as a human or a bot.

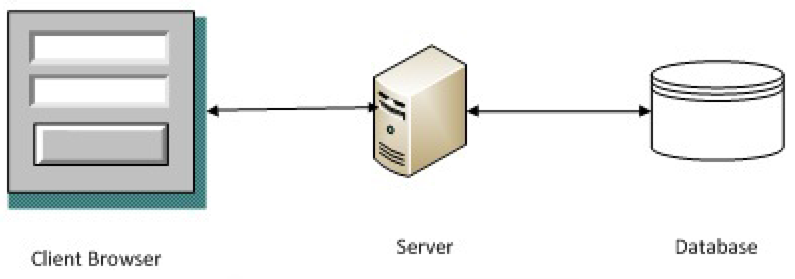
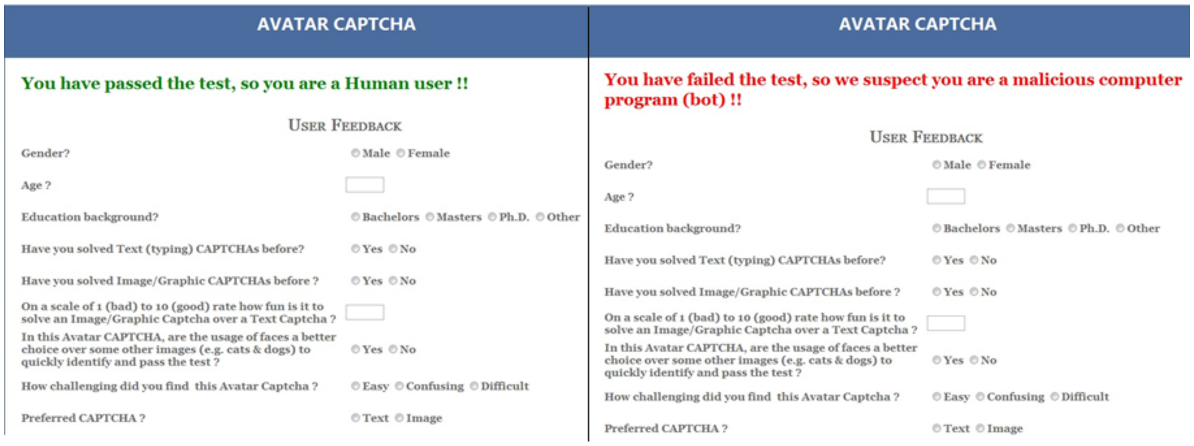


Figure 21: Client-Server architecture.

On hitting Submit, a feedback form is presented to the users to record their experiences and observations in solving this CAPTCHA. A messages is displayed above the form indicating the outcome of the challenge i.e. passed or failed as seen in Figure 22. These user responses help us obtain qualitative and quantitative information to improve the security and usability features of the CAPTCHA. The outcomes of the CAPTCHA test as well as the user responses are stored in two tables named *Test_Results* and *Survey_Feedback* within the database.



AVATAR CAPTCHA

You have passed the test, so you are a Human user !!

USER FEEDBACK

Gender?	○ Male ○ Female
Age ?	
Education background?	○ Bachelors ○ Masters ○ Ph.D. ○ Other
Have you solved Text (typing) CAPTCHAs before?	○ Yes ○ No
Have you solved Image/Graphic CAPTCHAs before ?	○ Yes ○ No
On a scale of 1 (bad) to 10 (good) rate how fun is it to solve an Image/Graphic Captcha over a Text Captcha ?	
In this Avatar CAPTCHA, are the usage of faces a better choice over some other images (e.g. cats & dogs) to quickly identify and pass the test ?	○ Yes ○ No
How challenging did you find this Avatar Captcha ?	○ Easy ○ Confusing ○ Difficult
Preferred CAPTCHA ?	○ Text ○ Image

AVATAR CAPTCHA

You have failed the test, so we suspect you are a malicious computer program (bot) !!

USER FEEDBACK

Gender?	○ Male ○ Female
Age ?	
Education background?	○ Bachelors ○ Masters ○ Ph.D. ○ Other
Have you solved Text (typing) CAPTCHAs before?	○ Yes ○ No
Have you solved Image/Graphic CAPTCHAs before ?	○ Yes ○ No
On a scale of 1 (bad) to 10 (good) rate how fun is it to solve an Image/Graphic Captcha over a Text Captcha ?	
In this Avatar CAPTCHA, are the usage of faces a better choice over some other images (e.g. cats & dogs) to quickly identify and pass the test ?	○ Yes ○ No
How challenging did you find this Avatar Captcha ?	○ Easy ○ Confusing ○ Difficult
Preferred CAPTCHA ?	○ Text ○ Image

Figure 22: Feedback form: On solving/failing the CAPTCHA.

The feedback survey form gets the following responses from the users:

Gender, age, education background, their experiences in solving text and image CAPTCHAs before, rating the fun factor in solving this Avatar CAPTCHA, justifying the choice of faces here, how challenging is it, their preferences in solving text or image CAPTCHAs, usage of this CAPTCHA on their websites, rate it and finally their comments or feedback on it. Once the user hits the Submit key the outcomes are stored in the Test_Results table. We capture the users IP address, success or failure outcome, number of avatars not selected, number of humans selected and time taken to give the test. Now if the users fill out the feedback form they are stored in the Survey_Feedback table.

3.1.1 Datasets

Facial images with upright frontal poses, complex backgrounds and varying illuminations were chosen and converted to grayscale images. This prevents the usage of color-based image recognition algorithms to detect unusually bright and uncommonly colored avatar faces, image backgrounds, etc. and consequently breach the system. In our proof of concept we used the following datasets:

- **Humans:** The Nottingham scans dataset [110] is used that contains a grayscale, facial images of 50 males and 50 females. They are primarily frontal and some profile views with differences in lighting and expression variation. Their resolutions vary from 358 x 463 to 468 x 536. For efficiency, thumbnail-sized images with resolutions of 100 x 123 are used. Figure 23 shows a sample of it.

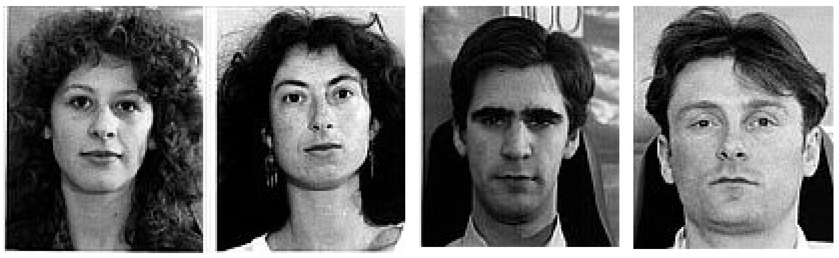


Figure 23: Sample of human faces from Nottingham scans dataset [110].

- **Avatars:** 100 samples of grayscale, frontal face avatar images [111] from the popular online virtual world Entropia Universe [11] are used. Each has a resolution of 407 x 549 pixels. Thumbnail-sized images with a resolution of 100 x 135 are used here for efficiency. Figure 24 shows a sample of it.

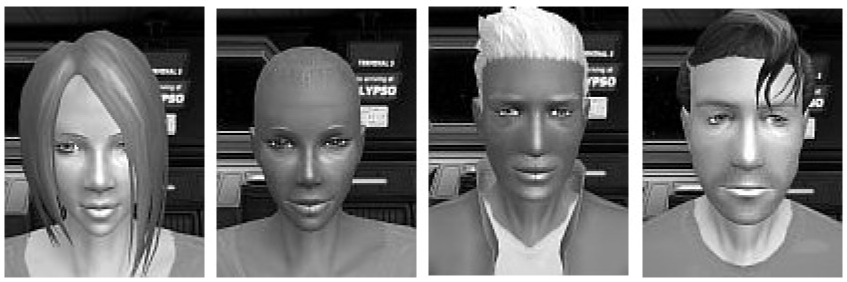


Figure 24: Sample of avatar faces from the Entropia Universe dataset [111].

3.1.2 Security

If a bot tries to break the CAPTCHA using a brute force approach, it will have a 50% (1/2) success in guessing the right image. So guessing 5 or 6 avatar images of 12 images will yield a success probability of $((1/2)^5 + (1/2)^6) / 2 = 0.0234375$, which is low. Users are unable to access the image databases. However, through a challenge presented at the ICMLA 2012 conference [112] to classify human and avatar facial images, it was learned that this problem was solved successfully [113, 114, 115, 116]. We aim to generate human and avatar datasets dynamically, obtained in real-time, from popular online websites such as Flickr and ActiveWorlds. These datasets will help us combat manual brute force attacks on the database by raising the computational costs and time.

3.1.3 Results

The results evaluated are records from 121 user test evaluations stored in the *Test_Results* table and 50 user feedback responses stored in the *Survey_Feedback* table within the database. Table 1 depicts an overview of their data.

Table 1: Overview of the CAPTCHA outcome data.

Outcome			**Avg. Submit Time (secs)**	**Avg. Success Time (secs)**
Success	**Failure**			
	Avatars missed (Avg)	***Humans checked (Avg)***		
101/111 = 90.99%	14	5	2689/121 = 22.2231	2410/101 = 23.8614

We observe that 91% of the users solved the CAPTCHA. Amongst the failures, on an average, one avatar missed out on being selected and one human face was accidentally selected. The submit time is the time when the user hits the Submit button to validate the test. The success time is the time reported when the CAPTCHA is solved. Average submit and success times of 22 and 24 seconds were reported respectively. Results from the *Survey_Feedback* table are split into two tables. Table 2 presents an overview of user demographics, text and image CAPTCHAs in general. Table 3 presents an overview of the Avatar CAPTCHA ratings by the users.

Table 2: **Part 1 of the user feedback data.**

Gender		**Age**		**Text CAPTCHA knowledge**	
Male	***Female***	***Min***	***Max***	***Yes***	***No***
30/50 = 60 %	20/50 = 40 %	18	61	42/50 = 84 %	8/50 = 16 %
	Education			**Image CAPTCHA knowledge**	
Bachelor's	***Master's***	***Ph.D.***		***Yes***	***No***
14/50 = 28 %	16/50 = 32 %	20/50 = 40 %		22/50 = 44 %	28/50 = 56 %

From Table 2 we observe that 56% of the users had never seen or solved an image CAPTCHA before. This signifies the need for this approach to be presented to the users.

Table 3: **Part 2 of the user feedback data.**

Usage of faces		**Preferred CAPTCHA**		**Website usage**		
Helping	***Unhelping***	***Image***	***Text***	***Yes***	***No***	
44/50 = 88 %	6/50 = 12 %	47/50 = 94 %	3/50 = 6 %	45/50 = 90 %	5/50 = 10 %	
		Rating			**Solvability**	
Excellent	***Good***	***Average***	***Poor***	***Easy***	***Confusing***	***Hard***
26/50 = 52 %	19/50 = 38 %	3/50 = 6 %	2/50 = 4 %	46/50 = 92 %	3/50 = 6 %	1/50 = 2%

Specific user comments are summarized and stated below:

- "Image CAPTCHAs are very easy"
- "A smaller, configurable interface would be nice as this takes a large amount of screen space"
- "I would prefer colored images over black and white (grayscale)"
- "Not everyone understands what an avatar is"
- "It is like a virtual keyboard where one uses a mouse"
- "Easy to solve"

The users also rated the "fun factor" of solving an image/graphic CAPTCHA over the traditional text CAPTCHAs on a scale of 1 (bad) to 10 (best). Figure 25 shows the outcomes in the labeled histogram.

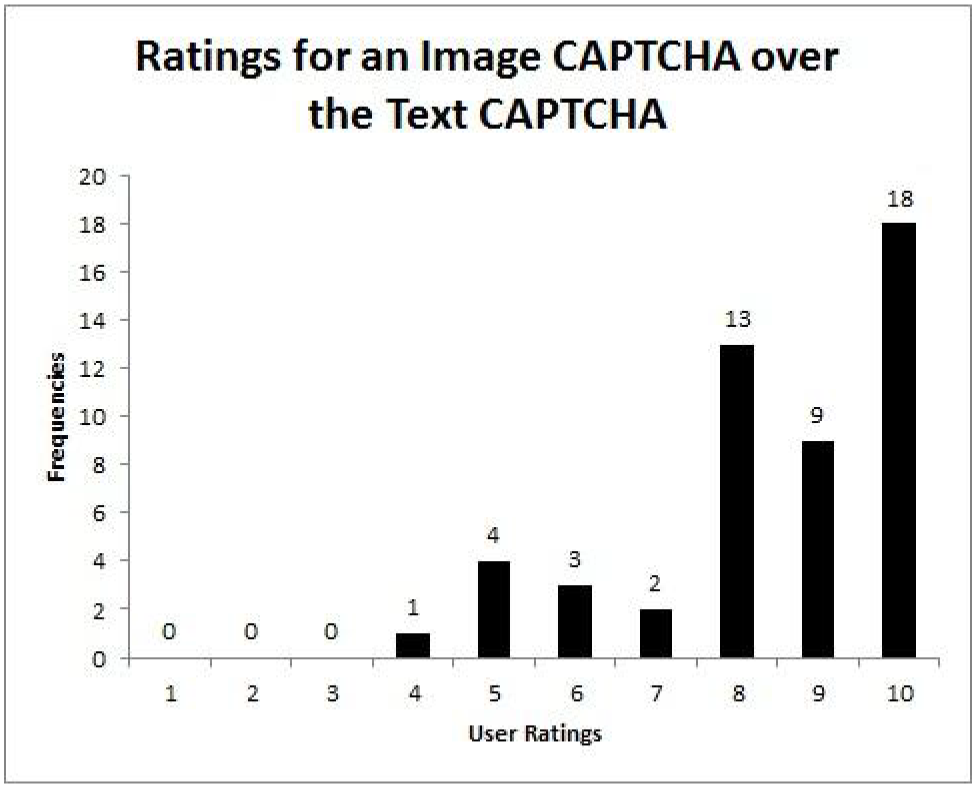


Figure 25: Histogram for user ratings of an image CAPTCHA over the text CAPTCHA with labeled frequencies.

3.1.4 Conclusions

Avatar CAPTCHA is a novel approach based on human computation relying on identification of avatar faces. Of 121 user tests recorded, 91% of the users solved the CAPTCHA. The average success time was 24 seconds. Of the 50 user feedback responses recorded 56% of the users had absolutely no knowledge about solving image CAPTCHAs, 94% of the users preferred image CAPTCHAs over text CAPTCHAs. 88% of the users stated that facial-image CAPTCHAs are easier to solve. 92% of the users rated it as easily solvable and 90% of the users had positive ratings for it. 90% of the users voted to use this on their websites. These statistics show it to be a convenient tool to filter out unauthorized access by bots. Designing CAPTCHAs indeed proves to be a challenge in building foolproof systems. A good approach is to make it fun and convenient for users to solve them.

3.2 Security

The security of the Avatar CAPTCHA prototype was tested by applying Local Directional Patterns (LDP) and two new face classification techniques to classify human faces from avatar faces and obtain baseline results. These two techniques were executed on a Gateway desktop computer with an Intel core i7 processor with a clock frequency of 3.4 GHz, 10 GB DDR3 memory and 2 TB hard drive.

1. Uniform Local Directional Pattern (ULDP), utilizes the uniform patterns from Local Directional Pattern (LDP).
2. Wavelet Uniform Local Directional Pattern (WULDP), applies the ULDP technique on the wavelet transform of the facial image.

Let us examine each of them in detail.

3.2.1 Applying Local Directional Patterns (LDP) for Human Avatar face classification

We implement the concept of LDP described in [15]. The Chi-square distance is used for classification. However, the concept of rotational invariance is absent. The implementation is performed using MATLAB on two datasets of human and avatar faces. Set I comprises of thumbnail-sized grayscale images presented as a challenge at International Conference on Machine Learning and Applications 2012 (ICMLA 2012) [112]. Set II comprises of thumbnail-sized grayscale images from the Avatar CAPTCHA prototype [59]. Kirsch masks, as described in [15], in all the eight directions are defined, are shown in Figure 6. For each image pixel these masks help evaluate the edge response value for each of its corresponding neighboring pixels. Table 4 lists the variables with their corresponding initial values.

Table 4: LDP parameters and their corresponding initial values.

Parameters	Neighbors (n)	Radius (r)	Window size (pixels)	Threshold (k)	Region size (pixels)	Bins	Human/Avatar Images
Initalized values	8	1	3x3	3	25x25	${}^{n}C_{k}={}^{8}C_{3}$ = 56 bins	52/48 (Set I) & 90/90 (Set II)

Here, we consider 8 neighbors for a window size of 3 x 3 pixels. A threshold value of k=3 is chosen to set the 3 most prominent edge response values to 1. The LDP coded image is divided into regions comprising of 25 x 25 pixels. If the size of a region ends up with less than this then it is selected as it is by default. The number of bins here is 56 and the human/avatar images is the ratio of the number of human and avatar images in the training dataset of Set I and II respectively. This value helps to evaluate the accuracy of the classifier by acting as a threshold. Now for 8 neighbors we end up with 0 to 2^8 -1 = 255 values of which only 55 values meet the threshold criteria of 3 bits set to 1. The rest of them are mapped to a single value.

This mapping helps in the binning process to generate the histogram. Thus, we end up with a histogram with 56 bins and consequently a feature descriptor of size 56. The datasets from Set I and II are divided into training and testing sets. The LDP coded image is evaluated for each of the training images from both the sets. This image is subdivided into regions depending on the region size. A histogram descriptor is extracted from each of these regions with 56 bins. These individual regional histogram descriptors are concatenated together to form the global image descriptor for each of the training images. A similar approach is followed for the testing images. The distance of the image descriptor of a testing image is compared with each of the training images using Chi-square distance classifier. These distances are sorted in the ascending order based on the training image index. The index of the image with the shortest distance to the testing image is noted. If this index value falls below the Human/Avatar threshold (HAT) value then it is classified as an avatar or else it is classified as a human. This accuracy, evaluated as a percentage, with the elapsed time in seconds is shown in the output.

3.2.1.1 Datasets

The following two datasets were used:

- **Set I:** This dataset comprises of thumbnail-sized, grayscale images from the ICMLA challenge [112] dataset. It is divided into 100 distinct training images (48 avatar faces and 52 human faces) and 30 distinct testing images (15 avatar and 15 human faces placed alternately). These images are preprocessed using histogram equalization and cropping to select the facial region only. Each image has a dimension of 50 x 75 pixels. A sample set is shown in Figure 26.

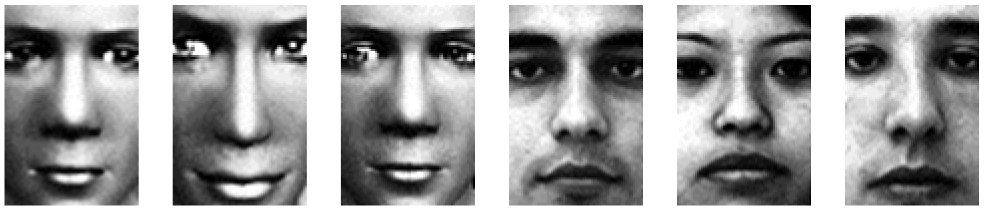


Figure 26: Sample images of human and avatar faces (Set I)[112].

- **Set II:** This dataset comprises of thumbnail-sized, grayscale images selected for the Avatar CAPTCHA prototype. It is divided into 180 distinct training images (90 avatar faces and 90 human faces) and 20 distinct testing images (10 avatar and 10 human faces placed alternately). There is no preprocessing on these images. Each image consists of the face, part of the body and has a dimension of 100 x 120 pixels. A sample set is shown in Figure 27.

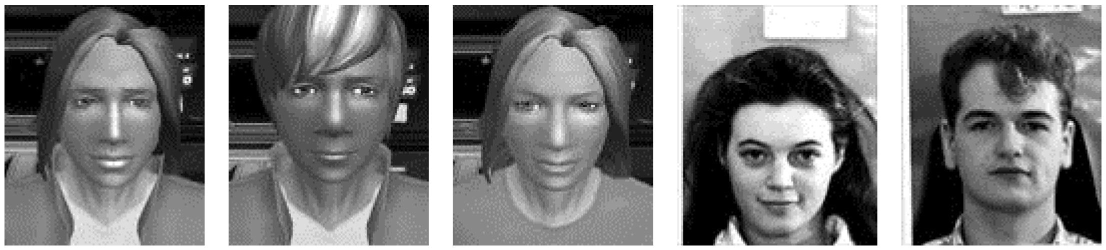


Figure 27: Sample images of human and avatar faces (Set II) [59].

3.2.1.2 Results

The results obtained on evaluating the LDP operator on Set I and II as seen in Table 5.

Table 5: Results of applying the LDP operator on Set I and Set II.

Dataset	Training	Testing	Dimension	Accuracy (%)
Set I	100	30	50 x 75	28/30 = 93%
Set II	180	20	100 x 120	20/20 = 100%

For Set I, we observe that 28 of 30 images are classified accurately yielding an accuracy rate of 93%. These images are cropped to select only the facial region and preprocessed using histogram equalization. This helps make it a challenging dataset and avoid template matching techniques to easily distinguish between the two classes of images. For Set II, we observe that 20 of 20 images are classified accurately yielding an accuracy rate of 100%. There are a few reasons for this. First, these images are not preprocessed i.e. no histogram equalization or cropping is performed. Second, the texture variance between human and avatar images are quite evident to the naked eye. Finally, the images being non-cropped contain the face as well as some part of the body, background, etc. that reveal some extra information. More details in an image helps make it an easier task to classify them.

3.2.1.3 Conclusion

The accuracy estimates achieved are on a small dataset. They can vary while dealing with real-time images. These results give us an insight on the possible modifications that can be applied in terms of preprocessing or altering the appearance of the real-time CAPTCHA images to make it robust and thereby improve its security. If higher accuracy rates are achieved, we have solved a difficult AI problem and consequently the CAPTCHA images must be processed, avoiding over distortion beyond visual identification. If lower accuracy rates are achieved, we must improve the LDP descriptor to subsequently improve the classification.

3.2.2 Applying Uniform Local Directional Patterns (ULDP) and Wavelet Uniform Local Directional Patterns (WULDP) for Human Avatar face classification

3.2.2.1 Datasets

The datasets comprised of upright frontal faces with plain/non-plain backgrounds and varying illuminations. All images were 400 x 400 pixels in dimension.
The datasets used were:

1. **Humans**

   - Set C - Caltech: Images from the California Institute of Technology [117] with non-plain backgrounds and varying illuminations.
   - Set F - FERET: Images from the FERET [118] dataset with plain backgrounds with varying illuminations.

Figure 28 shows sample images from the C and F datasets respectively.

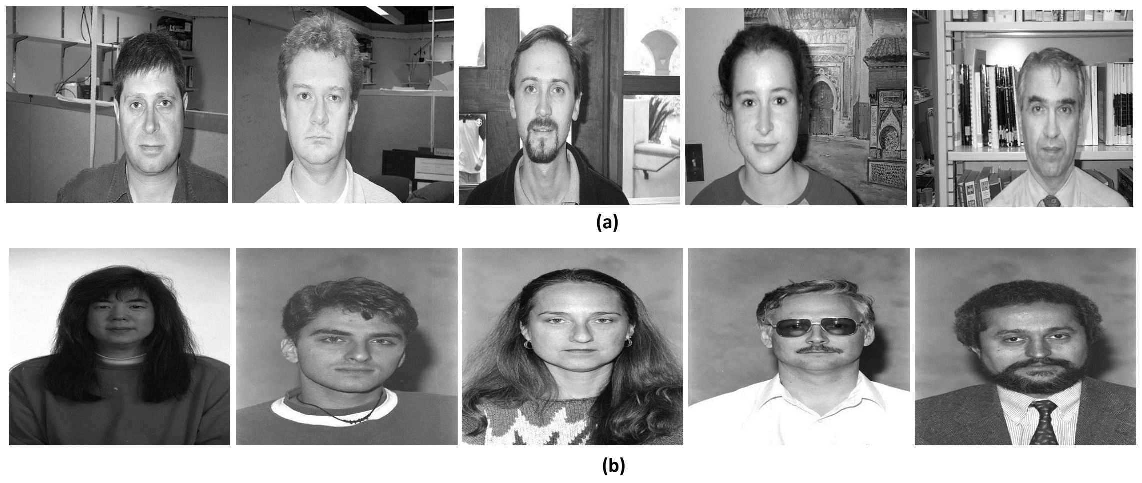


Figure 28: Sample human facial images from (a) Caltech (b) FERET datasets.

2. **Avatars**

- Set E - Entropia Universe: Images obtained from a scripting technique designed to automatically collect avatar faces [111] with non-plain backgrounds.
- Set SL - Second Life: Images obtained from the same scripting technique as that of Entropia [111] with non-plain backgrounds and varying illuminations.
- Set EV - Evolver: Images from an automated bot, used to collect avatar images [119] with plain background and varying illuminations.

Figure 29 shows sample images from the E, SL and EV datasets respectively. Six human-avatar dataset combinations are used altogether: CE, CSL, CEV, FE, FSL, FEV. Each combination has a total of 300 images (150 human and 150 avatar images).

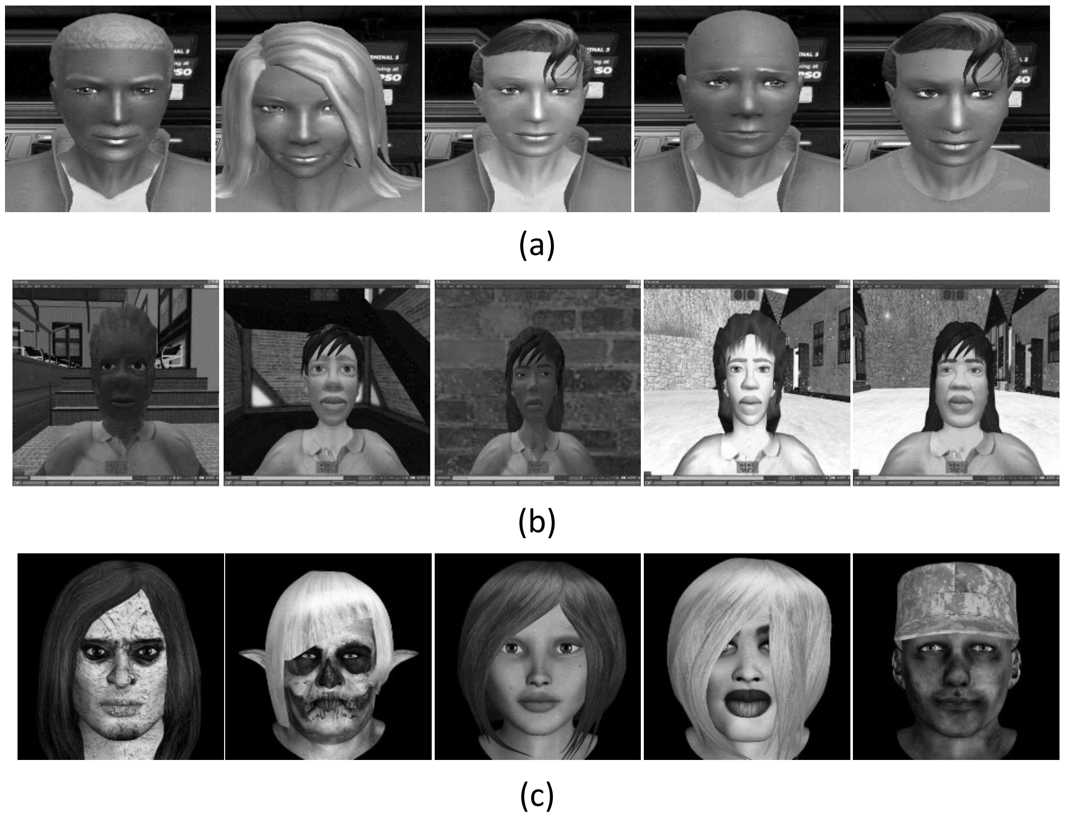


Figure 29: Sample avatar facial images from (a) Entropia Universe (b) Second Life (c) Evolver datasets.

3.2.2.2 Applying ULDP over an image

We considered the 8-neighbors for each pixel in the image. We obtained an 8-bit binary pattern i.e. values that range from 0-255. Of these, 56 values were LDP patterns with k=3. Of these 56 values, we obtained 8 uniform (8-bit binary patterns with no more than 2 bit transitions (0-1 or 1-0)) LDP values. These ULDP values obtained were 7, 14, 28, 56, 112, 131, 193 and 224. They were used to create 8-bin histograms, reducing the feature vector dimension from 56 (for LDP) to 8 (for ULDP). First, we applied Gaussian noise with its default parameters (zero mean and unit variance) on all the images. This was done to test the robustness of the algorithm in presence of noise. Next, we subdivided each 400 x 400 image into regions of size 80 x 80. Thus, we ended up with 25 regions per image. Next, we applied a 3 x 3 window with radius=1, neighbors=8 and threshold(k)=3 to each region to obtain the ULDP coded image using a mapping table. This table mapped each ULDP value to a different bin and the remaining values to one single bin. Thus, we ended up with a 7+1=8 bin local histogram for each region and a 25x8 bin histograms for the entire image. These histograms are concatenated to form a 1 x 200 bin global histogram which is the global descriptor for each image. Figure 30 (a) shows the ULDP coded image generation process and Figure 31 (a) shows the global descriptor generation process for an image.

3.2.2.3 Wavelet Uniform Local Directional Patterns (WULDP)

Here we applied WULDP to classify the images from each dataset as a human or avatar. First, we applied the Gaussian noise with its default parameters (zero mean and unit variance) on all the images. Next, we performed first-level decomposition on the input noisy image through 2D discrete wavelet transform using Daubechies wavelet filter db2 [16] to obtain the approximation image. This approximation image had a resolution half of that of the original image i.e. 200 x 200. This sped up its processing time. We subdivided each image into regions of size 40 x 40 thus, ending up with 25 regions per image. Finally, we applied the ULDP technique on this approximation image to obtain the WULDP coded image. Figure 30 (b) shows the WULDP coded image generation process and Figure 31 (b) shows the global descriptor generation process for an image.

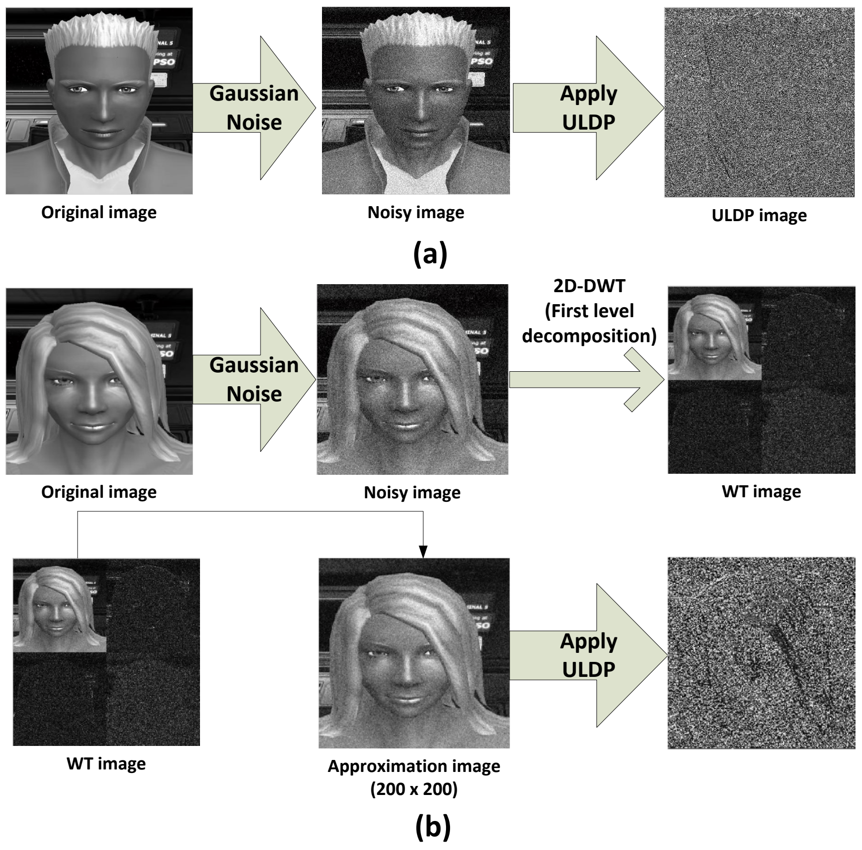


Figure 30: Coded image generation process (a) ULDP (b) WULDP.

For both the techniques, a 10-fold cross validation was performed over each dataset. The training set comprised of 270 random images whereas the test set comprised of the remaining 30 images from the set. The Chi Square distance was used to classify the images yielding accuracies for each dataset. Training times, test times as well as accuracies for each fold were recorded. We report the average training times, test times as well as the overall accuracy for each dataset.

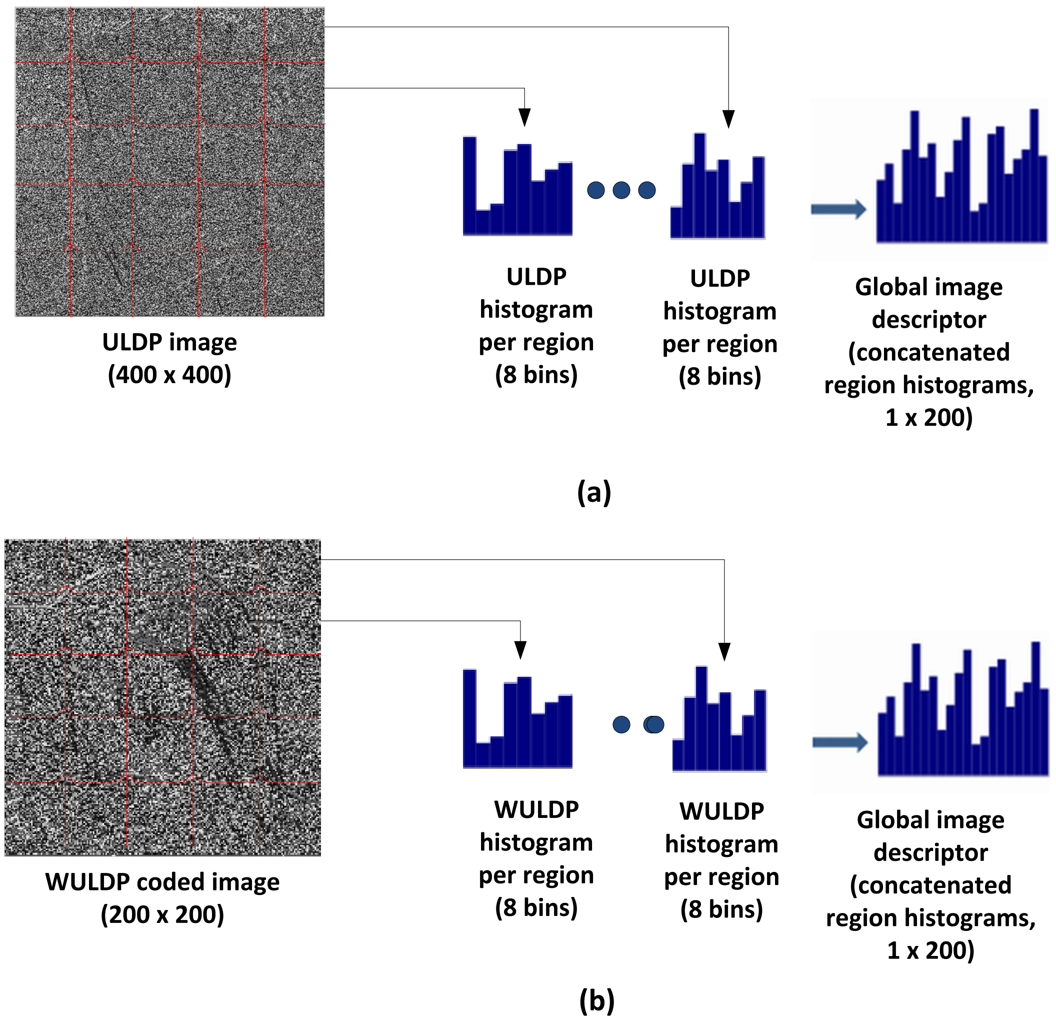

ULDP image
(400 x 400)
ULDP histogram per region (8 bins)
ULDP histogram per region (8 bins)
Global image descriptor (concatenated region histograms, 1 x 200)
(a)
WULDP coded image
(200 x 200)
WULDP histogram per region (8 bins)
WULDP histogram per region (8 bins)
Global image descriptor (concatenated region histograms, 1 x 200)
(b)

Figure 31: Global image descriptor generation (a) From ULDP coded image (b) From WULDP coded image.

3.2.2.4 Results

Results from ULDP and WULDP for each dataset are presented in Table 6 and Table 7 respectively with the average values over 10 folds of cross-validation.

Table 6: **Results over 10 folds of cross-validation for each dataset for ULDP. Image = 400 x 400 pixels, Window size = 3 x 3, Radius=1, Neighbors = 8, Region size = 80 x 80.**

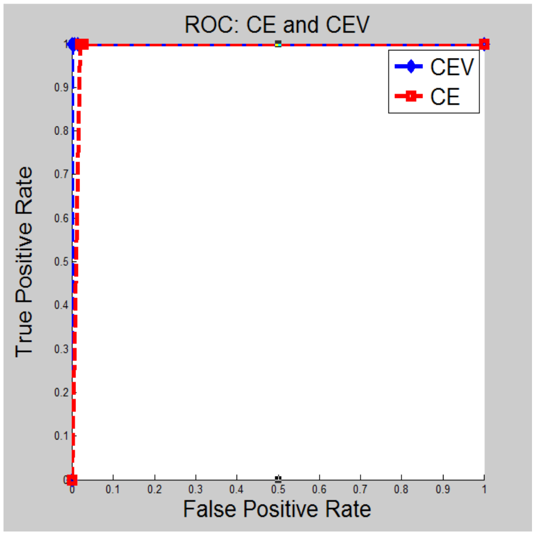
Datasets	**Avg. Training time (secs)**	**Avg. Test time (secs)**	**Overall Accuracy (%)**
CE	45.97	45.87	99
CSL	49.07	48.49	95
CEV	47.46	47.03	100
FE	47.21	47.24	100
FSL	48.21	48.28	97.33
FEV	51.91	48.66	100
Average	48.30	47.59	98.55

From Table 6 we observe that for ULDP, best accuracies were achieved for the CEV, FE and FEV datasets. The EV and F datasets had plain backgrounds which provided distinct patterns for classification. However, when evaluated against each other the remarkable results demonstrate the power of the ULDP descriptor. Executing this technique on the entire image yielded higher average training and testing times as that for WULDP.

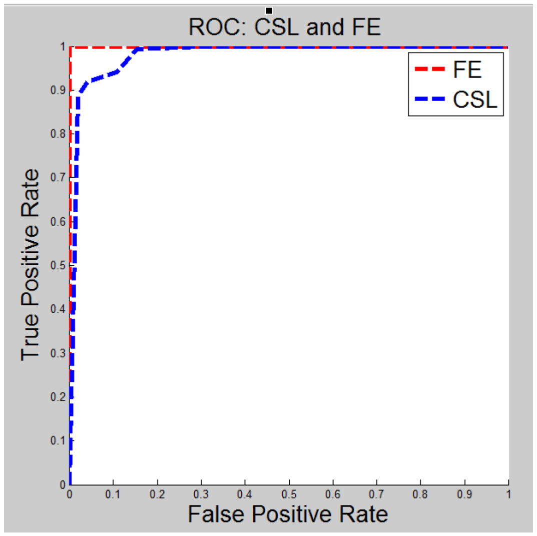
Table 7: **Results over 10 folds of cross-validation for each dataset for WULDP. Image = 200 x 200 pixels, Window size = 3 x 3, Radius=1, Neighbors = 8, Region size = 80 x 80.**

Datasets	**Avg. Training time (secs)**	**Avg. Test time (secs)**	**Overall Accuracy (%)**
CE	11.18	11.16	82.67
CSL	11.07	11.08	87.33
CEV	11.66	11.57	94.67
FE	11.37	11.46	87.33
FSL	11.66	11.55	96.33
FEV	12.15	12.17	89
Average	11.51	11.49	89.55

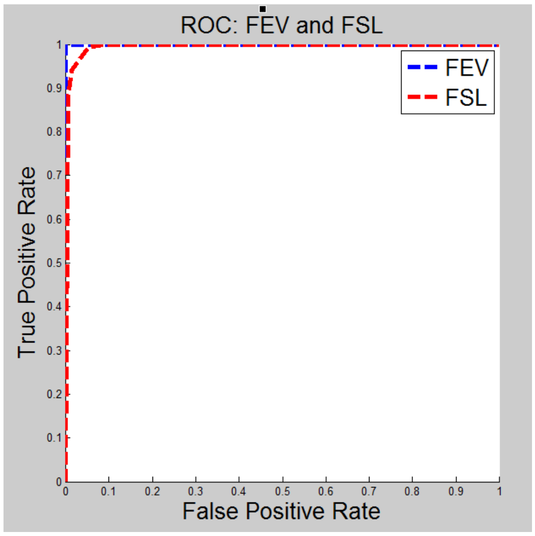
From Table 7 we observed that for WULDP, good accuracies were achieved for the FSL and CEV datasets. Since this was executed on the approximate image it yields lower average training and testing times. Figure 32 shows the ROC curves for both the techniques. The ULDP curves are segregated into three parts for clarity. Overall, we observe that ULDP descriptors are better than WULDP in classifying human and avatar facial images.



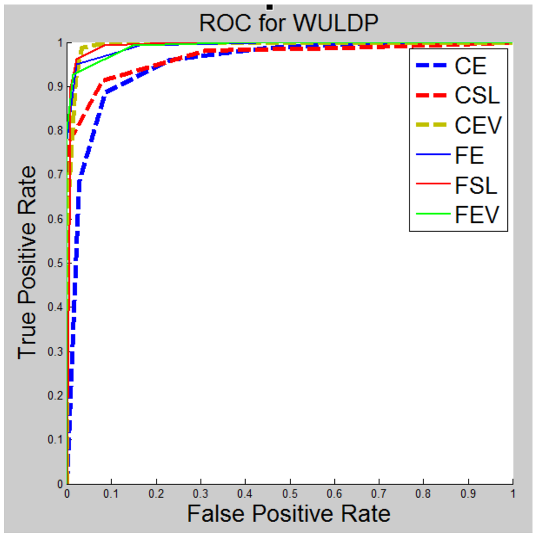
(a)



(b)



(c)



(d)

Figure 32: ROC curves for ULDP (a) CE and CEV (b) CSL and FE (c) FEV and FSL (d) WULDP.

CHAPTER IV
IMPROVING THE CAPTCHA DESIGN AND SECURITY

In this section, we describe our approach towards improving the CAPTCHA, hardening its security by applying Mouse Dynamics as a behavioral biometric and evaluating it in real-time to segregate human users from bots.

4.1 Approach

To begin with, the Avatar CAPTCHA prototype served as the starting point. First, the checkboxes are eliminated and the images are made click-friendly for users to select/deselect them. This helps in obtaining valuable mouse click data that helps in applying Mouse dynamics. Next, human and avatar face datasets are obtained from Flickr [120] using its API [121]. Flickr [120] is an image hosting web service with an evolving database of images. Millions of new images are being added everyday. It is an extremely challenging task to brute force its database. Public-accessible images are accessed and are solely used for research purposes by complying with Flickr APIs Terms of Use [122]. A pool comprising of different human and avatar groups from Flickr [120] is created to demonstrate the CAPTCHA as a proof of concept . Misclassified images are kept track of. Two approaches are followed in designing this CAPTCHA based on the way the datasets are obtained :

4.1.1 Real-time

The following steps briefly describe the pseudocode involved in obtaining real-time images from Flickr for the CAPTCHA :

1. *GroupPicker*: Select the two image groups randomly for human and avatars on Flickr [120] from the pool. Obtain 100 random images per group based on their dates taken (day, month and year).

2. *AvatarImageSelector*: This represents the main class of images to be chosen to solve the CAPTCHA. Pick out either 5 or 6 random images (of 100) from the avatar group. This

will be the avatar set.

3. *HumanImageSelector*: Pick out the remaining images randomly (of 100) from the human group . This will be the human set.

4. Combine the two sets from steps (1) and (2) together to form the *Dictionary*.

5. *MissingImageChecker*: Checks for missing images in the *Dictionary*. If there are missing images, loop back to get a different image from the appropriate class and update the dictionary.

6. Repeat steps (1) to (5) for a new challenge.

Figure 33 outlines the sequence of the components mentioned in the above steps. It is an iterative process as it loops back to pick different groups for each challenge.

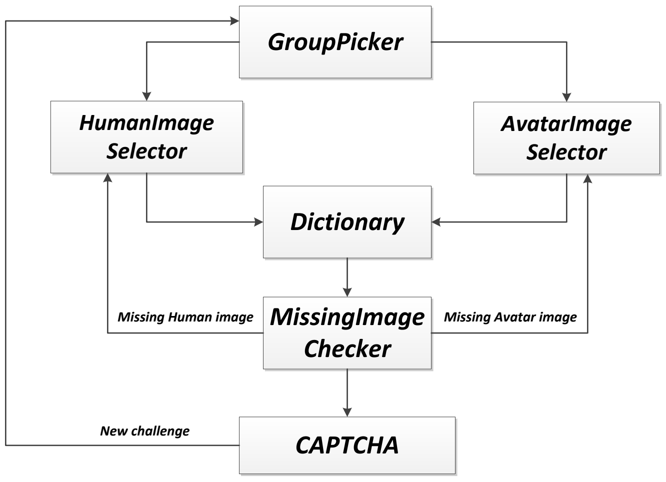


Figure 33: Flowchart for the real-time approach.

This approach works well and misclassified images are stored in separate folders. However, it is slow as it takes approximately 17 seconds to load each challenge. This is because it goes to Flickr to get the images for the *Dictionary* before checking for missing images amongst them. On learning this, a secondary approach was implemented using a database which will store the images from Flickr on a regular basis and serve as an intermediary data source for the CAPTCHA challenges.

4.1.2 Maintaining a database

The following steps briefly describe the pseudocode involved in creating a database for our CAPTCHA:

1. *GroupPicker*: Select the two image groups randomly for human and avatars on Flickr [120] from a pool of several different human-avatar groups. Obtain 100 random images per group based on their dates taken (day, month and year).

2. *AvatarImageSelector*: This represents the main class of images to be chosen to solve the CAPTCHA. Pick out 7 random images (of 100) from the avatar group. This will be the avatar set.

3. *HumanImageSelector*: Pick out the 7 random images (of 100) from the human group. This will be the human set.

4. Combine the two sets from steps (1) and (2) together to form the *Dictionary*.

5. *MissingImageChecker*: Checks for missing images in the *Dictionary*. If there are missing images, loop back to get a different image from the appropriate class and update the dictionary.

6. *CollisionChecker*: Checks for non-repetitive images within each challenge by computing hashes using Secure Hash Algorithm 256 (SHA256) and subsequently comparing them.

7. *ImageDatabase*: Save the image URLs , their class information and their misclassification count to the database.

8. Repeat steps (1) to (7) to populate the database.

Figure 34 outlines the sequence of the components mentioned in the above steps. It is an iterative process as it loops back to populate the database which is subsequently accessed to load each challenge.

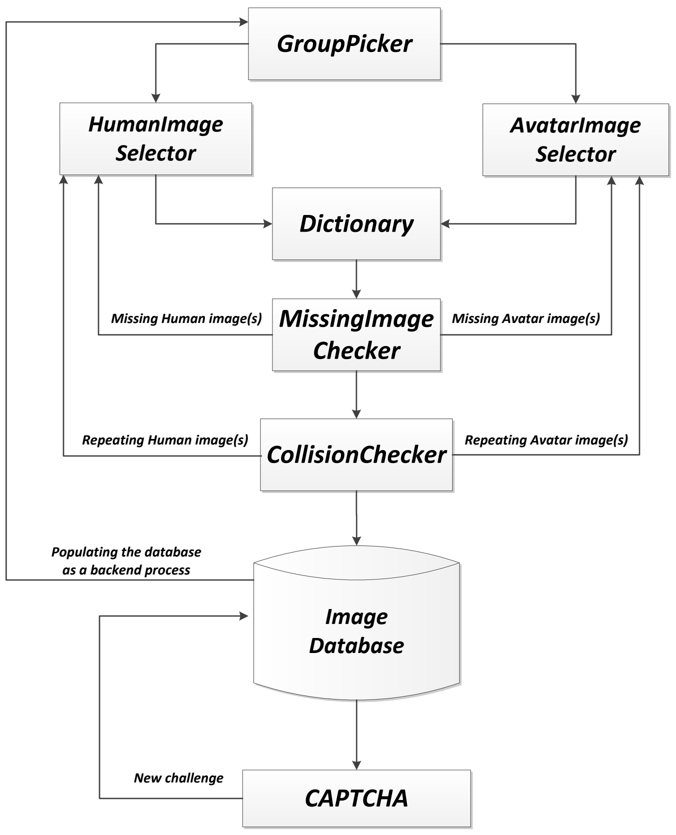


Figure 34: Flowchart for the database approach.

4.1.3 CAPTCHA Description

To solve the CAPTCHA the user has to identify and select all the avatar (artificial) images by clicking on them. Each challenge presents 12 image tiles of both human and avatar images. As the mouse cursor hovers over an image tile, a preview of that corresponding image is displayed at the top right corner of the page to help identify/recognize the image better. Figure 35 shows a snapshot of this CAPTCHA.

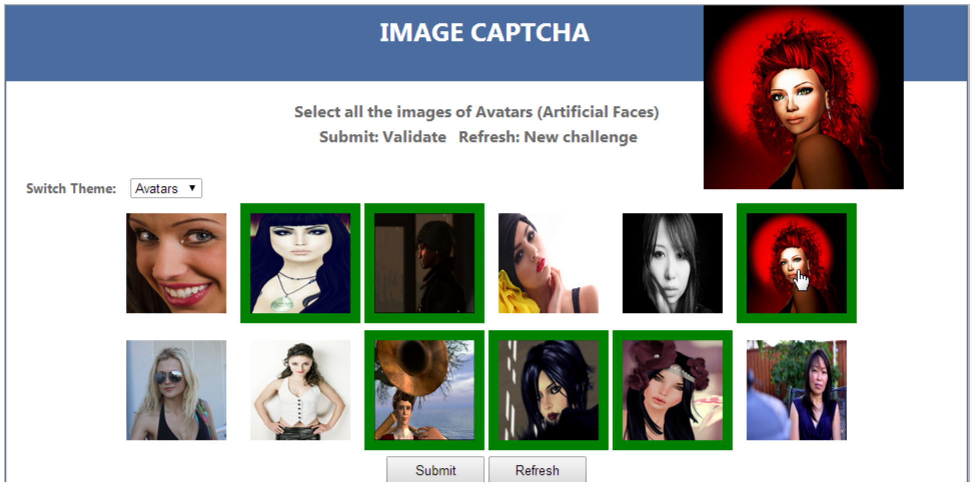


IMAGE CAPTCHA
Select all the images of Avatars (Artificial Faces)
Submit: Validate   Refresh: New challenge
Switch Theme:
Avatars
Submit
Refresh

Figure 35: Snapshot of the Avatar CAPTCHA.

The Submit button validates the challenge leading to a feedback page and the Refresh button presents a new challenge with different sets of human and avatar images. The user feedback responses are analyzed in the Results section.

4.1.4 Theme variations

We have also used two other themes besides human and avatar images. This is primarily done to move out of the two-class (human-avatar) classification/recognition problem to a broader one vs n-class problem to help diversify the datasets and make it a more challenging problem for a bot to break. The themes used are *Zoo* and *Objects*. The former theme consists of images of mammals, birds, reptiles and insects commonly grouped together under the term 'animals'. The latter theme consists of images of aeroplanes, helicopters, books, sodas and candies. Figure 36 shows snapshots of both these themes.

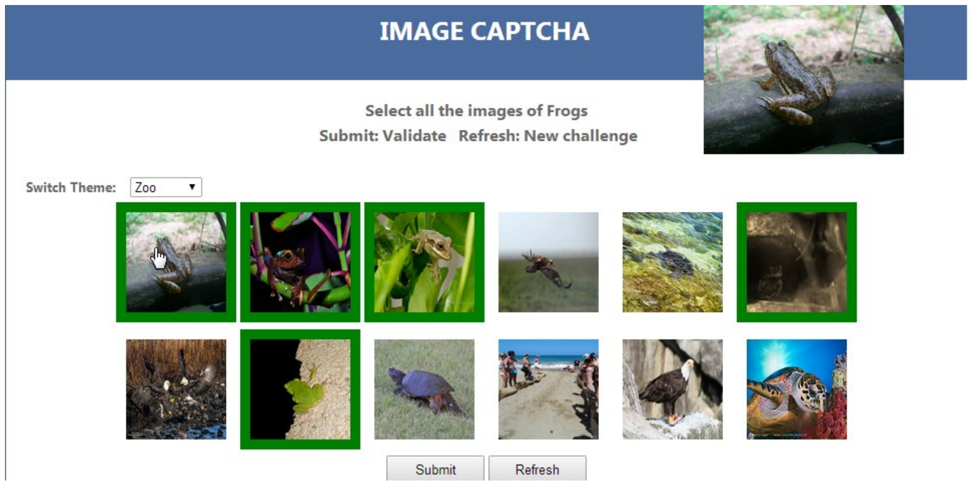


IMAGE CAPTCHA
Select all the images of Frogs
Submit: Validate Refresh: New challenge
Switch Theme: Zoo
Submit
Refresh

(a)

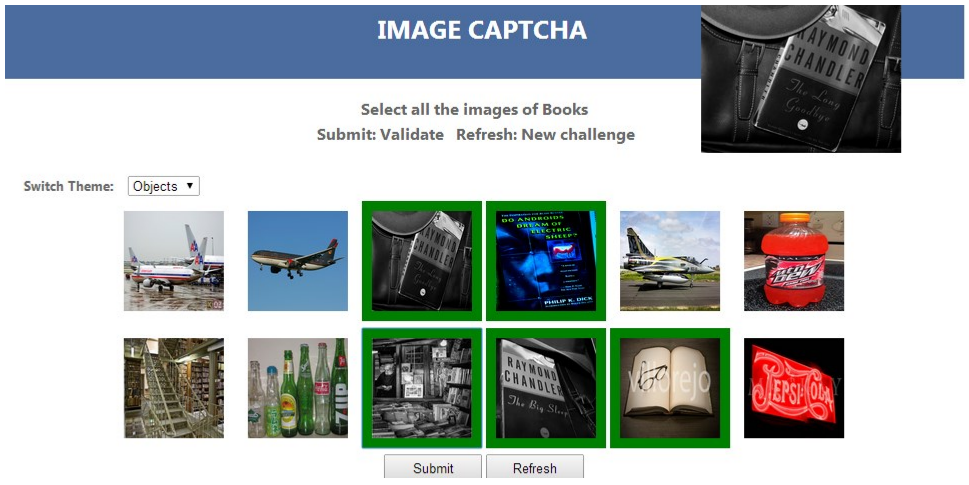


IMAGE CAPTCHA
Select all the images of Books
Submit: Validate Refresh: New challenge
Switch Theme: Objects
Submit
Refresh

(b)

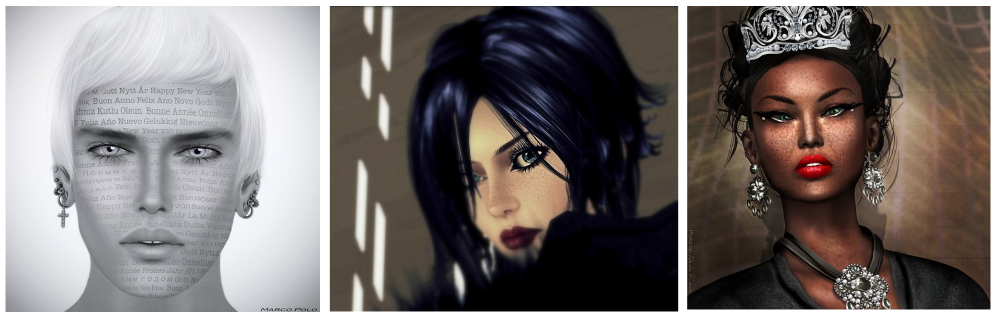
Figure 36: Snapshots of (a) Zoo CAPTCHA (b) Object CAPTCHA.

4.1.5 Datasets

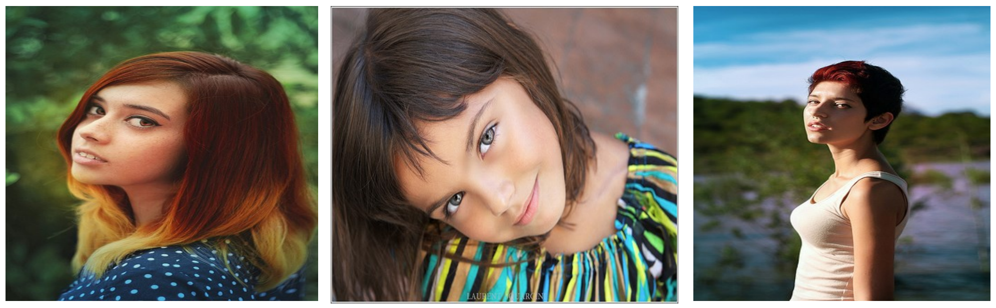
The datasets comprised of different images based on the themes. Table 8 lists the different groups of images used. Color images were used with resolutions varying between 300 x 500 to 500 x 300. Figure 37 shows a few samples of the human and avatar images.

Table 8: **Different image groups for the CAPTCHAs.**

Theme	Group
Avatar	Humans, Avatars
Zoo	Ants, Butterflies, Alligators, Dolphins, Elephants, Frogs, Giraffes, Horses, Iguanas, Jaguars, Kangaroos, Lizards, Gorillas, Bears, Owls, Parrots, Snakes, Rhinos, Scorpions, Tigers, Wolves, Eagles, Spiders, Turtles
Object	Aeroplanes, Helicopters, Books, Sodas, Candies



(a)



(b)

Figure 37: **Sample images of (a) Avatar and (b) Human.**

4.1.6 User evaluation

The two possible outcomes from this are Passed (Human) or Failed (potentially a bot). Now the failure is mainly due to either missing out on selecting the avatar image(s) or selecting the human image(s). The count of these values (misclassification frequency) are tracked by saving it in the database and the respective misclassified images are stored in the system. This helps us possibly eliminate such images in our future implementation. The outcomes and feedback are stored in two tables: *User_Test_Results* and *User_Survey_Feedback* in the database. The responses obtained from the feedback form are correspondingly stored in the

User_Survey_Feedback table. On hitting Submit, the following information is stored for each challenge in the *User_Test_Results* table*:*

- A numerical id, associated with each challenge, auto incrementing each time a challenge is submitted.
- The user computer's IP address.
- Success / Failure outcome (1 / 0).
- Number of images missed .
- Number of images selected incorrectly.
- Session time (time taken to submit the challenge).

Our CAPTCHA is designed using ASP.NET with Microsoft SQL Server as the database. Here, we followed the database approach to load images for each challenge. We recorded human user data following two approaches: Recruited subjects and Online users. For the former, we recruited 16 subjects to our lab to evaluate our CAPTCHA and get some feedback on it. Each user session was maximum 30 minutes in length. The first 9 subjects were asked to solve 72 consecutive challenges from the Zoo CAPTCHA and the remaining 7 subjects were asked to solve 24 consecutive challenges from the Avatar, Zoo and Object CAPTCHAs respectively. At the end of each session users provided their feedback on our CAPTCHA. Thus, we altogether received 888 attempts and 17 feedback responses. The images used were 100 x 100 pixel thumbnail images. For the latter, we posted the Zoo CAPTCHA online (http://darryl.cecsresearch.org) and emailed users requesting them to solve and provide their feedback on it. Altogether, we received 98 attempts and 49 feedback responses. A common feedback from the recruited subjects was that the images used were of low resolution (100 x 100 pixels) rendering them difficult for identification. Thus, we improved the image resolutions to approximately 320 x 240 pixels and used them for our online CAPTCHA dataset [123].

4.1.7 Results

Our results are reported based on an image dataset of different animal species with a maximum of 72 CAPTCHA challenges. The results are organized into three parts. In Part 1, we report results for the first 9 recruited subjects who attempted 72 consecutive Zoo CAPTCHA challenges. In Part 2, we report results for the remaining 7 recruited subjects who attempted 24 consecutive challenges from the Avatar, and Zoo CAPTCHAs respectively. In Part 3, we report the results for users who attempted to solve the Zoo CAPTCHA online. M1: Missed 1 correct image, S1: Selected 1 incorrect image, M1-S1: Missed 1 correct image and Selected 1 incorrect image.

Table 9. presents the results for Part 1. Here we observe that, subjecting users to attempt 72 consecutive challenges yields a best accuracy of 76.39%.The failure to solve the CAPTCHA is mostly due to users missing out on selecting one image. Perhaps, the intensity of focus the task demanded and thumbnail resolution (100 x 100 pixels) of the images used led to this outcome [123].

Table 9: Overview of Part 1 Zoo CAPTCHA data (First 9 recruited subjects solving 72 consecutive challenges).

User	**Success (%)**	**Failure**			**Avg. Submit time (secs)**	**Avg. Success time (secs)**
		M1	***S1***	***M1-S1***		
1	30.56	37	1	1	21.97	20.23
2	47.22	25	4	1	14.85	13.15
3	66.67	16	1	0	24.53	23.75
4	70.83	4	8	1	13.83	13.94
5	58.33	22	2	1	28.90	28.90
6	44.44	29	1	4	18.97	18.25
7	31.94	36	2	2	22.81	22.52
8	76.39	8	4	2	14.67	13.8
9	69.44	12	3	3	17.5	17.5
Average	39.66	24.14	2.71	1.42	19.78	19.12

Table 10 and Table 11 present the results for Part 2. Here we observe that, subjecting users to fewer consecutive challenges yields a best accuracy of 87.50%. Failure due to users missing out on selecting one image is relatively low as compared to Part 1 results. Failure to solve the CAPTCHA is mostly due to users missing out on selecting one image. Accidentally selecting an image outside the requested class is almost negligible.

Table 10: **Overview of Part 2 data (7 recruited subjects solving 24 consecutive challenges from Zoo CAPTCHA).**

User	**Success (%)**	**Failure**			**Avg. Submit time (secs)**	**Avg. Success time (secs)**
		M1	***S1***	***M1-S1***		
10	75	3	2	1	16.29	15
11	83.33	3	1	0	20.88	21.10
12	62.50	5	1	1	22.04	20.87
13	87.50	1	2	0	15.42	14.05
14	70.83	2	1	1	18.42	17.41
15	41.67	8	1	0	13.71	13.30
16	62.50	7	1	0	29.33	28.47
Average	69.05	4.14	1.29	0.43	19.44	18.60

Table 11: **Overview of Part 2 data (7 recruited subjects solving 24 consecutive challenges from Avatar CAPTCHA).**

User	**Success (%)**	**Failure**			**Avg. Submit time (secs)**	**Avg. Success time (secs)**
		M1	***S1***	***M1-S1***		
10	78.26	2	2	0	14.69	13
11	64	2	8	1	20.04	19.68
12	75	0	5	0	22.41	22.16
13	20.83	7	7	2	12.20	9.4
14	87.5	2	1	0	20.12	20.38
15	66.66	4	6	2	16.33	15.81
16	58.33	5	4	0	27.20	24.92
Average	64.36	3.14	4.71	0.71	18.99	17.90

Table 12 presents the feedback results of 16 recruited users. A brief description of each alphabet-symbolized category is given below:

A: Minimum/Maximum age of users

B: Experience in solving Text CAPTCHAs?

C: Experience in solving Image CAPTCHAs?

D: Animal images helpful in solving this CAPTCHA?

E: Preference for image CAPTCHAs over text CAPTCHAs

F: Use this CAPTCHA on websites?

Table 12: **User feedback data from the 16 recruited subjects.**

A	**B**	**C**	**D**	**E**	**F**
22/52	13/16 = 81.25%	7/16 = 43.75%	11/16 = 68.75%	14/16 = 87.50%	14/16 = 87.50%
	Solvability			**Rating**	
Easy	***Confusing***	***Difficult***	***Excellent***	***Good***	***Average***
7/16 = 43.75%	7/16 = 43.75%	2/16 = 12.5%	3/16 = 18.75%	10/16 = 62.50%	3/16 = 18.75%

From Table 12 we observe that 7 of 16 users found it confusing to solve perhaps due to the low resolution of images which made it somewhat difficult to identify the images and 13 of 16 users rated this CAPTCHA positively. Similarly, Table 13 presents the results for Part 3 and Table 14 presents the feedback results from 56 online users respectively.

M-All: Missed all the correct images. S-All: Selected all the incorrect images. Rest: All intermediate cases

Table 13: **Overview of Part 3 data (Online users solving the Zoo CAPTCHA challenges).**

Challenges submitted	**Success**	**Failure**		
		M1	***S1***	***M1-S1***
228	95/228 = 41.66%	29/133 = 21.80%	8/133 = 6.01%	7/133 = 5.26%
Avg. Submit time (secs)	**Avg. Success time (secs)**	***M-All***	***S-All***	***Rest***
6213/228 = 27.25	2518/95 = 26.51	24/133 = 18.04%	1/133 = 0.75%	64/133 = 48.12%

Table 14: **User feedback from online users.**

A	**B**	**C**	**D**	**E**	**F**
19/64	44/56 = 78.57%	21/56 = 37.50%	45/56 = 80.35%	47/56 = 83.92%	45/56 = 80.35%
	Solvability			**Rating**	
Easy	***Confusing***	***Difficult***	***Excellent***	***Good***	***Average***
39/56 = 69.64%	16/56 = 28.57%	1/56 = 1.78%	27/56 = 48.21%	20/56 = 35.71%	9/56 = 16.07%

The most common feedback comment received from the recruited subjects was *"The images have a low resolution rendering them difficult to recognize, thus confusing."*. On improving the image resolutions for the online users, we received some valuable feedback. Few are stated below:

"Good one. Looks good !!"

"I think images are better than words which are sometimes not legible"

"Innovative idea. Looks good and works well."

"Way better than Text CAPTCHAs !! "

"Some images were difficult to recognize when the target animal is very small in the image."

"Use frequently noticed images."

From the above comments we note that some images confused the users in recognizing them, especially those with smaller images. Some images were incorrectly selected or not selected at all. These images are termed as *Misclassified* and were tracked in our database. A few sample misclassified images from this group are shown in Figure 38.

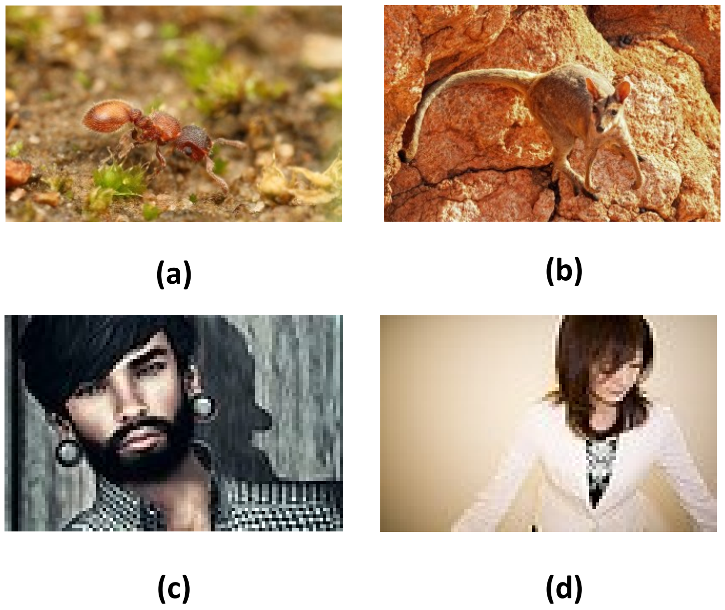


Figure 38: (a) Ant, mistaken for a Spider (b) Kangaroo not selected (camouflage) (c) Male avatar, mistaken for an human (d) Female human, mistaken for an avatar.

4.2 Behavioral Biometrics: Mouse Dynamics

In this section, we examine the security of the CAPTCHA using Mouse Dynamics i.e. monitoring the mouse usage activities while the CAPTCHA is being solved and subsequently using this feature to segregate human users from bots. Mouse usage data from human users and a bot, simulated by an adversary as an attempt to break the CAPTCHA, are obtained.

4.2.1 Acquiring mouse usage data from human users

4.2.1.1 System Design

For our study, we developed our CAPTCHA as a Windows application to be displayed on a web browser. It is setup on a Dell Inspiron 1440 laptop with a Pentium Dual-Core 2.2 GHz processor, 4 GB RAM and 64-bit Windows 7 operating system. It is equipped with a 14" Monitor (1366 x 768 resolution) and a Microsoft Wireless Notebook Presenter 8000 mouse. All system parameters related to the mouse such as speed and sensitivity configurations were fixed and unaltered during the course of the study. The CAPTCHA was written as a Windows application in C# and presented users with challenges to be solved. During data collection the CAPTCHA was presented on the laptop monitor and the mouse data was recorded as each user attempted to solve it.

The following mouse usage activities by users were non-obtrusively recorded and constitute our raw mouse data:

- On screen cursor position coordinates (*x and y*).
- Mouse movement.
- Mouse Click.
- Mouse Release.
- Timestamps (in ms)

The Javascript Date function was used to timestamp the mouse operations and the Windows-event clock resolution of 15.625 milliseconds, corresponding to 64 updates per second was used to perform the conversion to milliseconds [124]. During data collection each subject was invited to solve the CAPTCHA challenges on the same laptop free independent of other subjects and the data was collected one by one on the same platform. Figure 39 shows an overview of our approach which depicts the Training as well as the real-time classification process of the incoming mouse data samples.

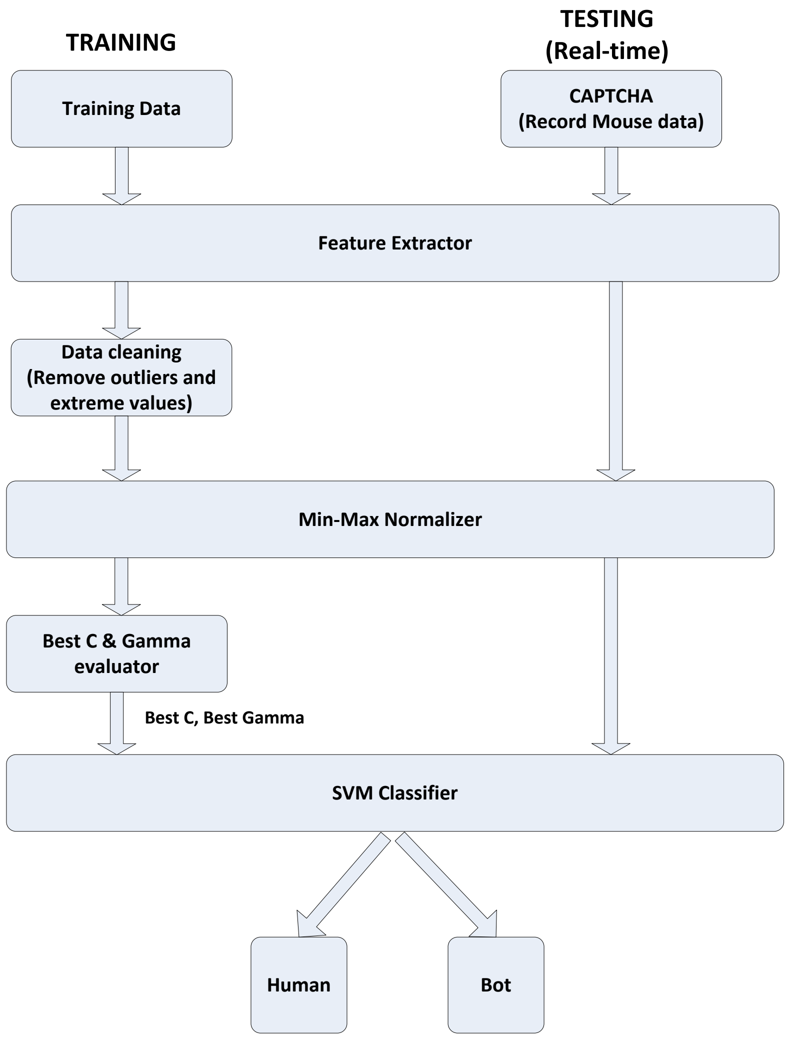


Figure 39: **Overview of our MD approach.**

4.2.1.2 Subjects and Data collection

We recruited 16 subjects, 13 male and 3 females, from within our lab, the department and some from the university. Each user session constituted of solving consecutive CAPTCHA challenges. Each user had one session. The first 9 users solved 72 challenges from the Zoo CAPTCHA. The next 7 users solved 24 challenges each from the Zoo, Avatar and Object CAPTCHAs respectively. The data collection process took 15 days. As each subject solved each challenge their mouse usage activity data were non-obtrusively recorded in the background using JavaScript and stored in separate text files. The final dataset comprised of 1150 samples. Figure 40 shows a visualization sample of the recorded mouse data for a human user for one session.

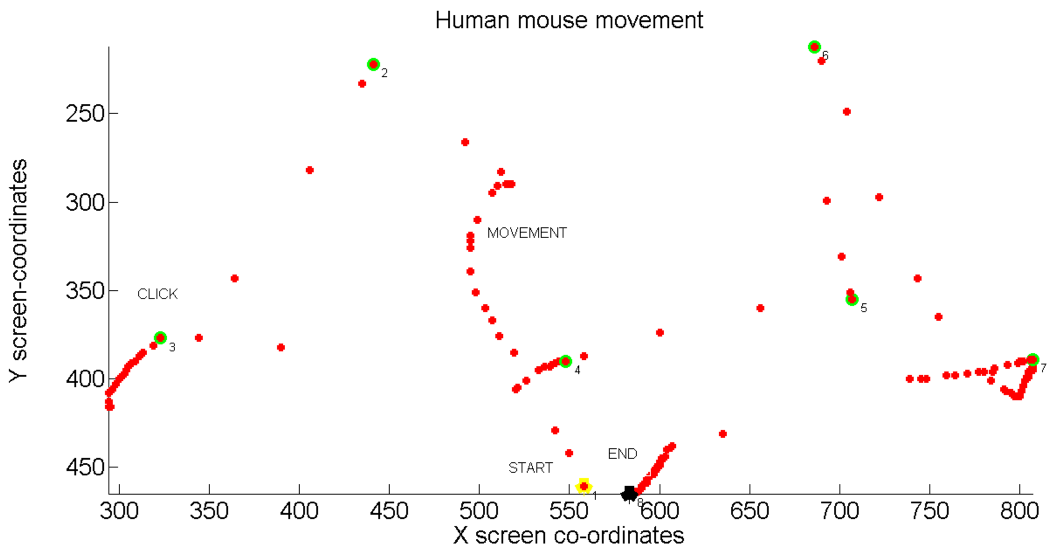


Figure 40: Visualization sample of recorded mouse data for a human user.

4.2.2 Acquiring mouse usage data from the bot

Someone else besides me served as an adversary to design a bot with an attempt to penetrate the CAPTCHA. He had little to no knowledge in the domain of image processing coming into the project. Knowledge was briefly gained in curriculum where he learned about it and the different methods associated with it, by working on this project and modifying the algorithm bit by bit to until it became workable. In bot writing however, he had a bit of experience when he used to write mouse-bots to move the mouse pointers to random locations on the screen but nothing in the way of mimicking human-like mouse movements. In Computer Science, his knowledge is as much as his curriculum and based on his research experience working on this project. With absolutely no access to the source code except for the CAPTCHA web page HTML interface and theoretical knowledge about how it worked, he began the process of solving the CAPTCHA by breaking the task into smaller, manageable chunks. He pulled the images from the web page and downloaded them to the system to use for processing. Either five or six images of a particular group were to be selected to solve it. A script was written to save all images and a reverse image search was performed using Google's Image Search [125] to pull "tags" from the images. Following this, several attempts were made to execute it over the CAPTCHA. However, the results were not promising. Reliable image tags could not be pulled of other search engines as well such as Flickr [120] and TinEye [126] since they yielded inconclusive results.

On completing the rendition of the script a new challenge faced by the adversary was to fool the mouse dynamics aspect of the system in addition to solving the image recognition problem. To mimic the mouse movements, the adversary first observed his own movements when using the computer. He learned that his mouse cursor movements resembled a parabolic path and the mouse shakes a bit when the user first begins to use it. Mimicking these aspects in a script was the next significant challenge in the development process. To transform this process into code a few calculations were performed with the parabolic movements of the mouse cursor simulated using the equation for a parabola and have the bot's motion follow this pattern. To simplify the process, only four possible parabolas that could be formed were used: up-left, up-right, down-left and down-right.

The next design section involved mimicking the shake of the mouse as performed by a human using it. A routine was created for this purpose that is randomly called throughout the iteration of the script that modifies the mouse cursor position by a few pixels in random directions. On random occasions, the script overshoots its destination which is compensated by sliding the mouse cursor back. Image recognition is by no means a trivial task. To evaluate how a computer could determine the similarity between the images, they were re-sized to 300 x 300 pixels, divided into 10 x 10 pixel chunks and stored in individual arrays. Next, each chunk is compared to the corresponding chunk in the other image using pixel by pixel comparison. The chunk similarity threshold is set to 35% to determine that two images are similar and they are subsequently added to a list of similar images along with their similarity percentages. Thus for each challenge, every image is compared to every other image and five or six images of the twelve presented with higher similarities are chosen by the bot script. This script is fully functional for selecting the number of images that is currently required by the CAPTCHA to solve its challenges. On implementing the basic image recognition algorithm, random alike images from the internet were compared to validate the algorithm. Next, all these features were integrated into one finished bot script. On completing the integration process, some final testing was performed during which it was realized that the script must scrape the images from the CAPTCHA web page itself, save, compare and subsequently simulate the mouse movement to select them.

Figure 41 shows a visualization sample of the recorded mouse data for a bot for one session.

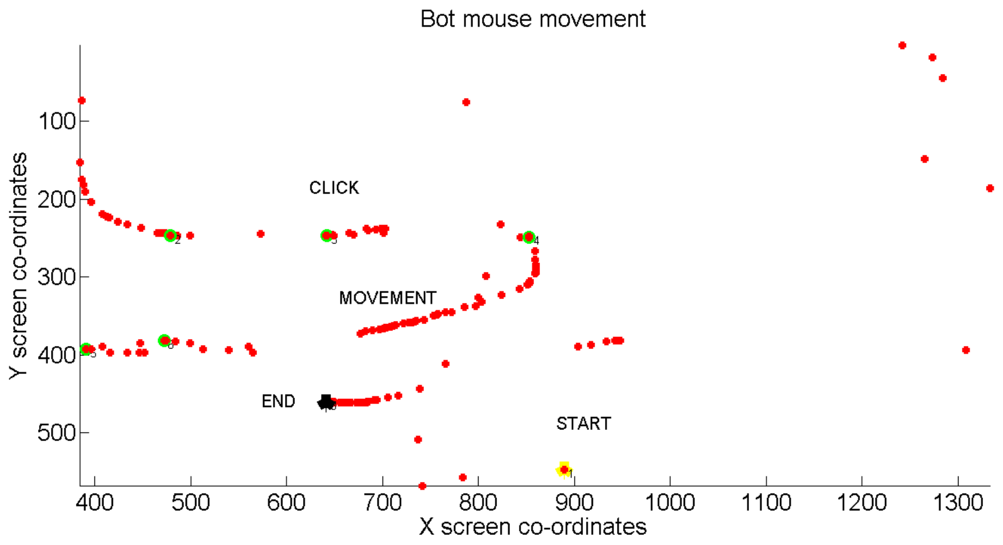


Figure 41: Visualization sample of recorded mouse data for a bot.

4.2.2.1 Gather bot training data samples (B1)

The goal here is to acquire mouse usage data samples for bot attempts to solve the Avatar and Zoo CAPTCHA challenges respectively. We also evaluate the bot's accuracy of solving the challenges using image recognition. These samples, both correct and incorrect, together constitute the bot profile which we combine with the human user profile to help build our training dataset. A MATLAB code is run to evaluate the best cost and best gamma values that help achieve the best classification accuracy for SVM. A range of cost and gamma values are evaluated for this purpose. The value of cost is varied from -15 to 5 in steps of 1 and the value of gamma is varied from -15 to 5 in steps of 1. WEKA [127] is used to perform a 10-fold cross validation utilizing the best cost and best gamma values on the accurate human-bot samples to yield accuracies. This help us train and build the SVM classifier model to classify CAPTCHA attempts as either that by a human or a bot.

4.2.2.2 Bot performance on individual animal image groups (B2)

The goal here is to evaluate the bot's image recognition performance on individual animal image groups from the Zoo CAPTCHA mentioned in Table 8. This helps us determine the animal image groups that are easy/difficult for the bot to solve.

4.2.2.3 Mouse dynamics classifier performance against the bot (B3)

The goal here is to evaluate the performance of the SVM classifier in real-time to distinguish between human users and bots based on their respective mouse usage activities as they attempt to solve challenges from the Avatar and Zoo CAPTCHA respectively.

4.2.3 Feature extraction

We extract the following 20 features based on this data collected to create feature vectors and build Mouse Dynamic Signature (MDS) profiles for both, human users and bots for each session:

- Average Speed (AS): The average of all the speed values between consecutive data points. The speed between the two consecutive data points is evaluated as the ratio of Euclidean distances between their cursor position coordinates over the time difference between them. Equation 1 shows how it is evaluated.

$$AS = \frac{\sum_{i=1}^{n} \frac{EuclideanDistance(x_i, y_1, x_{i+1}, y_{i+1})}{(t_{i+1} - t_i)}}{n} \tag{1}$$

  where, *n: total number of raw data points recorded per session and*

$$EuclideanDistance(x_1, y_1, x_2, y_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{2}$$

- Average Acceleration (AA): The average of all the acceleration values between consecutive data points. The acceleration between two consecutive data points is evaluated as the ratio of the speed (*s*) between their speed over the time difference between them. Equation 3 shows how it is evaluated.

$$AA = \frac{\sum_{i=1}^{n} \frac{EuclideanDistance(x_i, y_1, x_{i+1}, y_{i+1})}{(t_{i+1} - t_i)^2}}{n} \tag{3}$$

  where, *n: total number of raw data points recorded per session.*

- Average Angle of Curvature (AAOC): The average of all the angles of curvature values between consecutive data points. The angle of curvature, measured between three consecutive data points, is evaluated as the angle formed between the two line segments joining them. It is summarized in equations 4 - 8. below:

$$AAOC = \frac{\sum_{i=1}^{n} AOC_i(p_i, p_{i+1}, p_{i+2})}{n} \tag{4}$$

$$a = EuclideanDistance(p_{ix}, p_{iy}, p_{(i+1)x}, p_{(i+1)y}) \tag{5}$$

$$b = EuclideanDistance(p_{(i+1)x}, p_{(i+1)y}, p_{(i+2)x}, p_{(i+2)y}) \tag{6}$$

$$c = EuclideanDistance(p_{(i+2)x}, p_{(i+2)y}, p_{ix}, p_{iy}) \tag{7}$$

$$AOC_i = Cos^{-1}((a^2 + b^2 - c^2)/(2ab)) \tag{8}$$

where, *p_1,p_2,p_3,.......,p_n are consecutive raw data points, n: total number of raw data points recorded per session.*

- Average Click time (AC): The average of all the times measured between a mouse click and a mouse release. The click time is measured between data point pairings of a mouse click and its next immediate mouse release. Equation 9 shows how it is evaluated.

$$AC = \frac{\sum_{i=1}^{n}(t_r - t_c)}{n} \tag{9}$$

where, *t_c: timestamp for a mouse click, t_r: timestamp for the mouse release following t_c, n: total number of raw data points recorded per session.*

- Average Silence time (ASIL): The average of all the times measured during inactivity of the mouse. The silence time is measured for those successive data points where the *x* and *y* cursor coordinates are equal. This signifies no mouse movement. Equation 10 shows how it is evaluated.

$$ASIL = \frac{\sum_{i=1}^{n}(t_2 - t_1)}{n} \tag{10}$$

where, *t_1: timestamp for cursor position (x_1, y_1) , t_2: timestamp for cursor position (x_2, $y_{,2}$) provided x_1=x_2 and y_1=y_2 and $t_1 \neq t_2$ and n: total number of raw data points recorded per session.*

- Average Movement time (AM): The average of all the times measured when the mouse is moving i.e. not silent. The movement time is measured for those data points where the *x* and *y* cursor coordinates are not equal. This signifies mouse movement and its evaluation is shown in Equation 11.

$$AM = \frac{\sum_{i=1}^{n}(t_2 - t_1)}{n} \tag{11}$$

where. *t_1: timestamp for cursor position (x_1, y_1) and t_2: timestamp for cursor position (x_2, y_2) provided $x_1 \neq x_2$ and $y_1 \neq y_2$ and $t_1 \neq t_2$ and n: total number of raw data points recorded per session.*

- Average speeds in eight directions (S_1 - S_8): For mouse movements, there is a difference between the x and y co-ordinates of two consecutive mouse data samples. For example, if (x_1, y_1)and (x_2, y_2) are two such points then the angle θ (converted to degrees from radians) made by the line joining these two points with the Y-axis is given as shown in Equation 12.

$$\theta = \arctan(d_x, d_y)\left(\frac{180}{\pi}\right) \tag{12}$$

where, d_x = Absolute difference between x_1 and x_2; d_y = Absolute difference between y_1 and y_2,

We subdivide each quadrant into two equal sectors of 45 degrees each. Thus, we end up with eight sectors altogether based on increments of 45 degree angles with the Y-axis. The average speeds (S_1 - S_8) are evaluated accordingly depending on which quadrant sector the line lies based on θ using the general formula for speed as shown in Equation 13.

$$Speed(S_i) = \frac{EuclideanDistance(x_1, y_1, x_2, y_2)}{(t_2 - t_1)} \tag{13}$$

where, *t_1: Time stamp for the first point, t_2: Time stamp for the second point and $t_1 \neq t_2$ and i: Sector number from 1-8 .*

Figure 42 illustrates a visual representation of this.

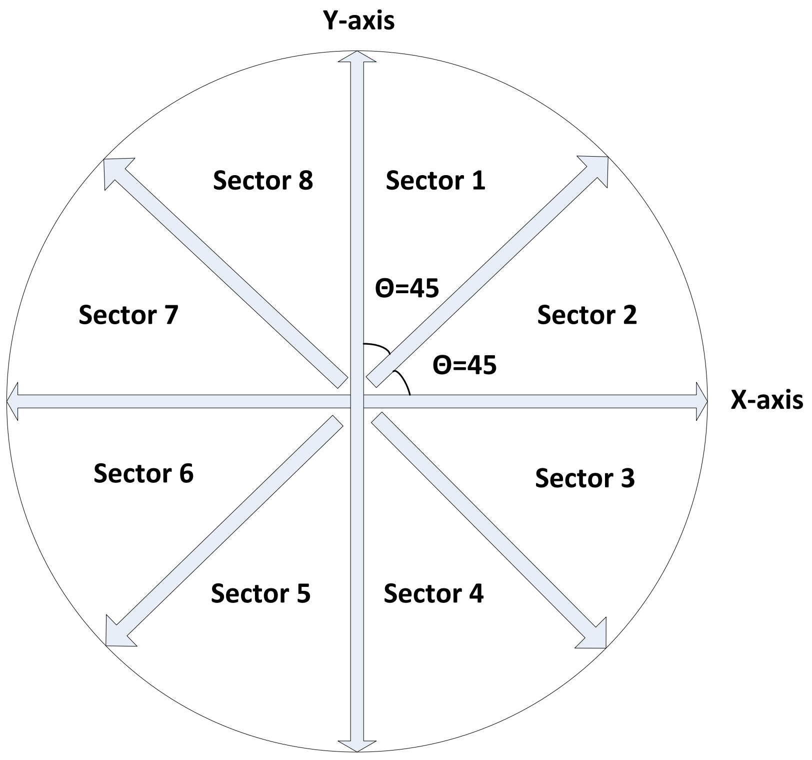


Figure 42: The eight sectors for the eight directional speed feature evaluation.

- Speed Standard Deviation (SSD): The standard deviation evaluated over the speed values for each data point based on the AS obtained.

- Acceleration Standard Deviation (ASD): The standard deviation evaluated over the acceleration values for each data point based on the AA obtained.

- Angle of Curvature Standard Deviation (AOCSD): The standard deviation evaluated over the angle of curvature values for the data points based on the AAOC obtained.

- Click Standard Deviation (CSD): The standard deviation evaluated over the click time values for the data points based on the AC obtained.

- Silence Standard Deviation (SiSD): The standard deviation evaluated over the silence time values for the data points based on the ASILobtained.

- Movement Standard Deviation (MSD): The standard deviation evaluated over the movement time values for the data points based on the AM obtained.

The general formula for the evaluation of standard deviation (SD) for each of these features is given in Equation 14.

$$SD = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^2} \tag{14}$$

where, x_i*: Value of each data point,* $\overline{x}$*: Mean value of the feature and n: total number of raw data points recorded per session.*

We use the Support Vector Machine (SVM) classifier here. It is a well known supervised learning model which can be used for classification and regression analysis. Here the training vectors are mapped to a higher dimensional space and SVM finds a linear separating hyperplane with the maximal margin in this higher dimensional space. C is the penalty parameter for the error term. Different kernel functions are used to achieve the classification. Four of the basic kernels used to devise the kernel functions are: linear, polynomial, radial basis function (RBF) and sigmoid. For more details on SVM please refer [22]. In our work, we have used the LibSVM software distribution [128] for the SVM classifier using the RBF kernel as a similarity measure function.

In the next section, we present the results for B1, B2 and B3

4.2.4 Results

4.2.4.1 B1

The bot samples were executed in batches of 100, 200, 300, 400 and 500 over the Avatar (AC) and Zoo CAPTCHA (ZC) respectively and the accuracies of the bot solving them were noted along with the session times for both the CAPTCHA themes. Table 15 shows the results.

Table 15: Results for B1.

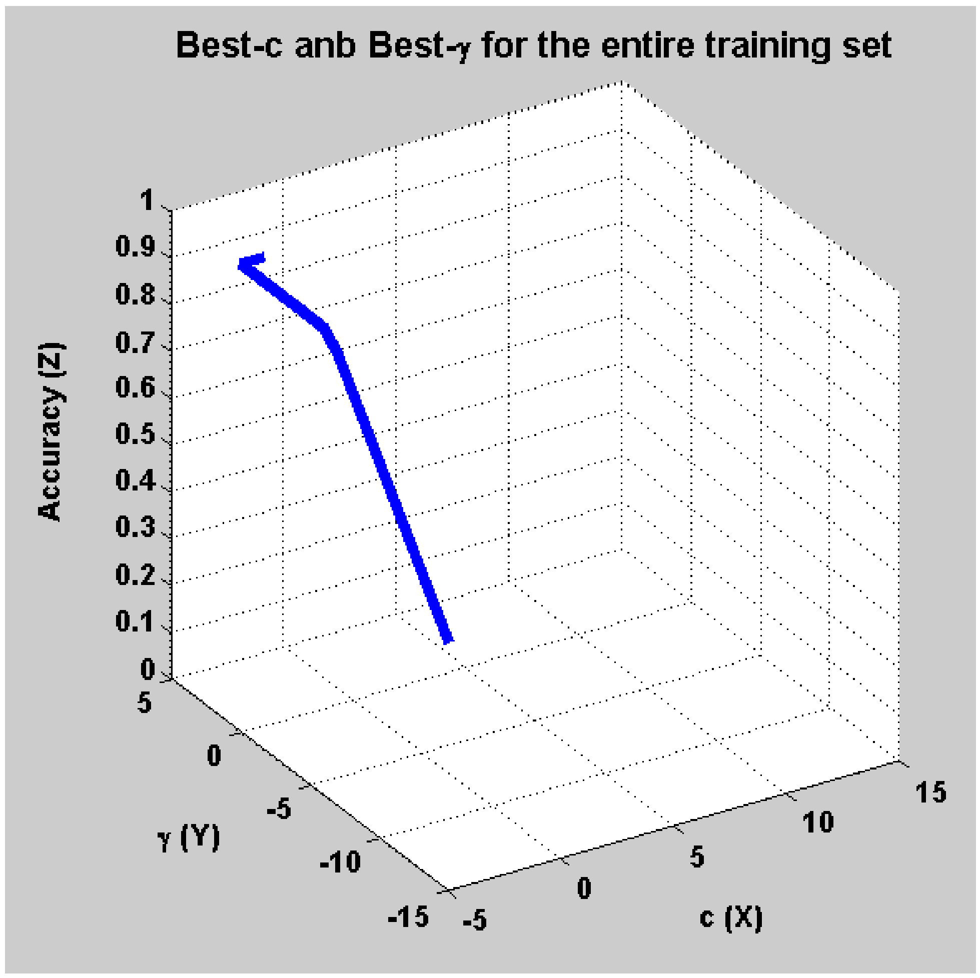
Training samples	AC	ZC	Accuracy on AC (%)	Accuracy on ZC (%)	Avg. Session Time on AC (secs)	Avg. Session Time on ZC (secs)
100	50	50	1/50 = 2%	4/50 = 8%	966/50 = 19.32	1088/50 = 21.66
200	100	100	3/100 = 3%	13/100 = 13%	1977/100 = 19.77	2046/100 = 20.46
300	150	150	3/150 = 2%	9/150 = 6%	3509/145 = 24.20	3333/150 = 22.22
400	200	200	0/200 = 0%	22/200 = 11%	3639/200 = 18.19	3716/200 = 18.58
500	250	250	4/250 = 1.6%	22/250 = 8.8%	4582/250 = 18.32	4544/250 = 18.17
Sum = 1500	750	750				

We combined equal number of human and bot samples to form the final training set comprising of altogether 2300 samples i.e. 1150 samples from each class. WEKA [127] was used to analyze it. Cleaning was performed by deleting the outliers and extreme data values using Interquartile Range and RemoveWithValues techniques respectively. This resulted in a pruned training set with 1751 instances altogether. Next, the instances were normalized using Min-Max Normalization which constitutes the normalized training set. Next, this normalized training set was analyzed using MATLAB to evaluate the best cost (c) and best gamma (γ). A grid search was performed by varying the values of c and γ. c was varied between the values of -5 and 15 with increments of 1 and γ was varied between the values of -15 and 5 with increments of 1. For each combination of c and γ a 10-fold cross validation was performed on the training set to evaluate the best-c and best-γ for the best cross validation (cv) accuracy achieved. The best-c and best-γ achieved for a best-cv of 100% were 0.0625 and 1 respectively.

Figure 43 shows a sample plot depicting the grid search process in obtaining the best-c and best-γ for the best-cv achieved over the training set.

4.2.4.2 B2

Table 16 shows the results of the bot solving CAPTCHA challenges comprising of different groups of animals in which few images are repeated between different challenges. The table enlists the animal group name, number of challenges attempted by the bot altogether and the number of challenges solved by it.



Start: X=-5, Y=-15, Z=0.5316 End: X=-4, Y=0, Z=1

Figure 43: **Plot for the best-c and best-γ obtained on the normalized training set.**

Table 16: **Outcome of the bot solving CAPTCHA challenges of different animal groups.**

Animal Group	Challenges attempted	Challenges solved	Animal Group	Challenges attempted	Challenges solved
Ants	74	6	Jaguars	73	2
Alligators	76	10	Kangaroos	75	3
Bears	76	9	Lizards	74	1
Butterflies	76	6	Owls	75	6
Dolphins	73	4	Parrots	69	0
Eagles	75	0	Rhinos	61	0
Elephants	76	6	Scorpions	73	2
Frogs	75	1	Snakes	77	9
Giraffes	72	4	Spiders	77	6
Gorillas	73	6	Tigers	75	3
Horses	75	5	Turtles	72	4
Iguanas	77	9	Wolves	75	6

Figure 44. shows few sample images of that were identified and unidentified by the bot.

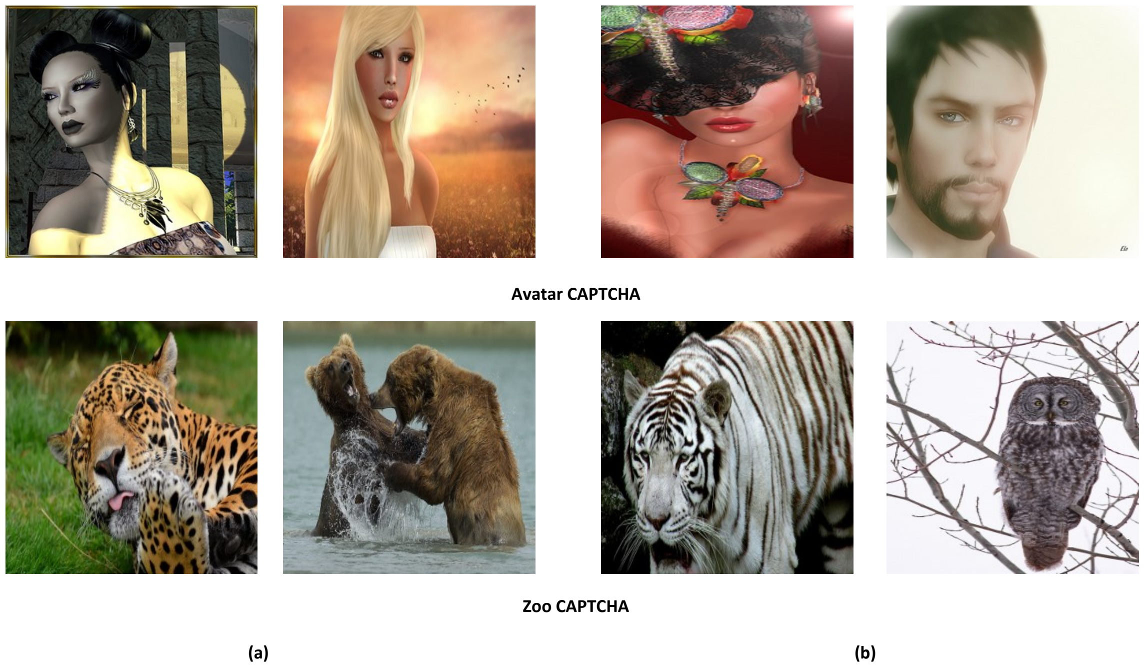


Figure 44: Sample images from both CAPTCHAs (a) Identified (b) Unidentified by the bot.

4.2.4.3 B3

Here, we validate our SVM classifier against real-time human and bot samples to estimate its accuracy.

First, we visualize the raw mouse training data with different set of features using WEKA [127]. The plots are shown in the Figure 45. From this we observe that there is a clear distinctive pattern in the features of human users and the bot. Moreover, this being a two-class (Human vs Bot) classification problem and on observing the plots, we chose SVM as our classifier with a linear kernel to distinguish between human users and the bot based on their mouse usage patterns.

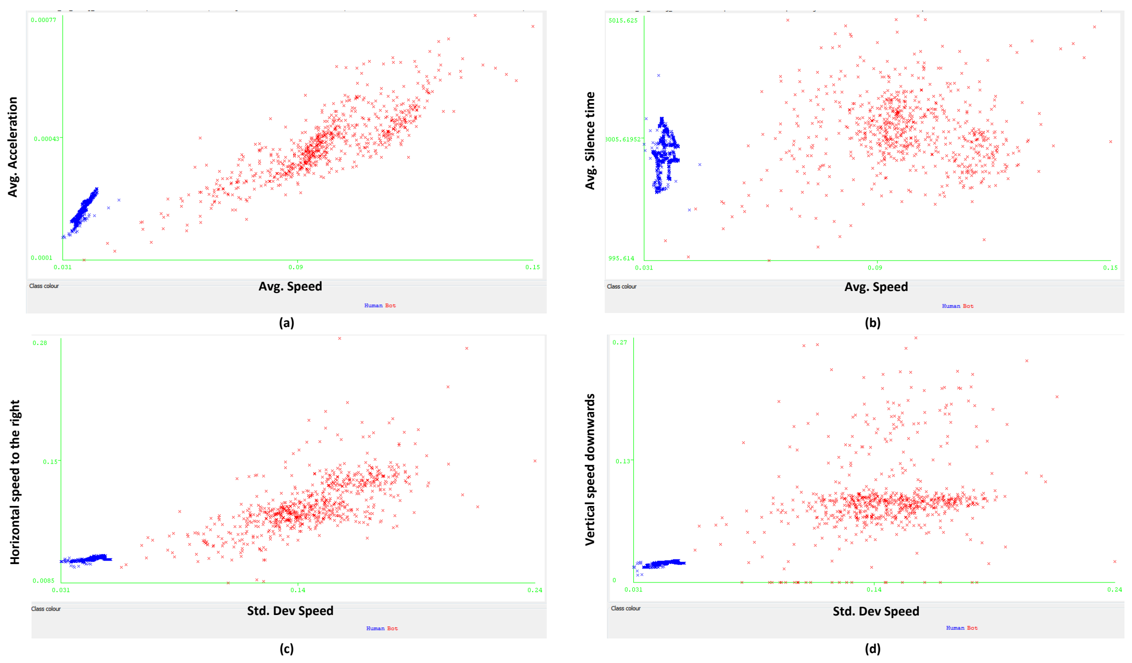
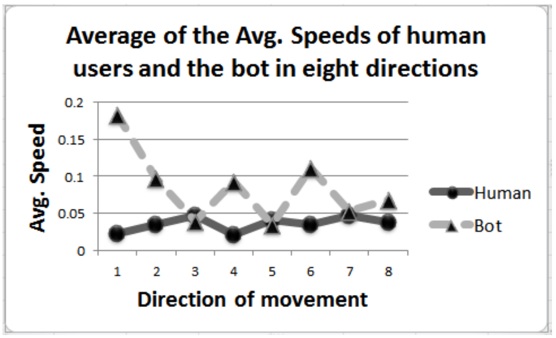


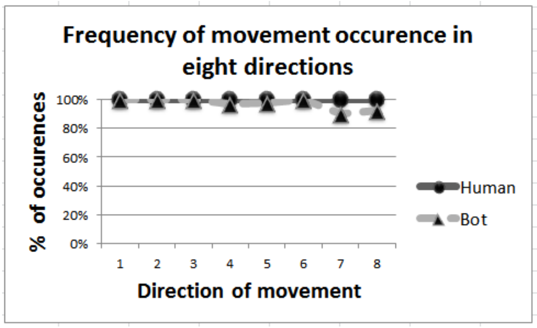
Figure 45: Plots of different set of features (a) Avg. Speed vs Avg. Acceleration (b) Avg. Speed vs Avg. Silence time

(c) Std.Dev Speed vs Horizontal speed (to the right) (d) Std.Dev Speed vs Vertical speed (down).

We also examine the average speeds in eight directions between human users and the bot in terms of the average of each average speed and their frequency of mouse movement occurrences in each of the eight directions. Figure 46 shows the histograms of both. Here we observe distinctive patterns in the average of the average speeds between the human user and the bot. The human user mouse movement had a 100% occurrence rate in each of the eight directions whereas the bot did not move in directions 1, 4, 5, 7 and 8 respectively on certain instances.



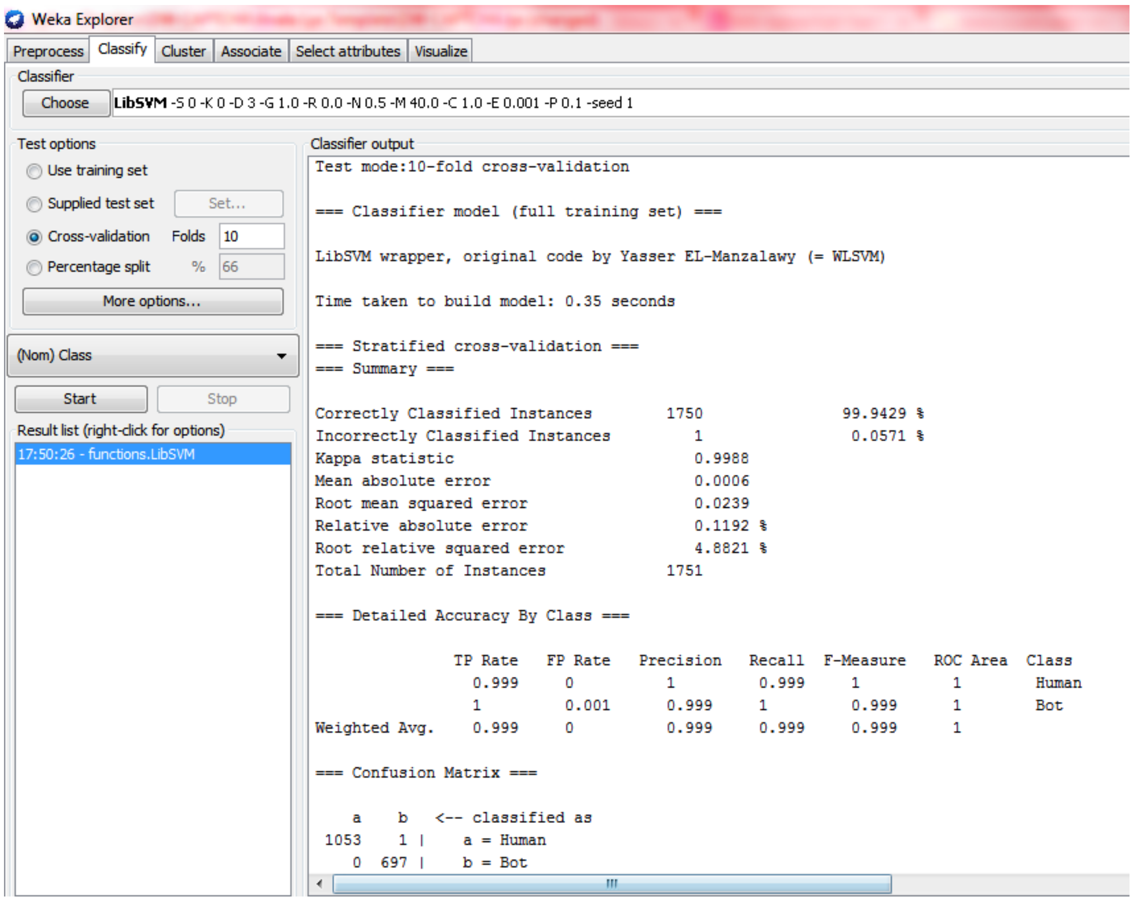
(a)



(b)

Figure 46: (a) Average of the average speeds (b) Frequency of mouse movement occurrences in each of the eight directions for human users and the bot.

Next, we analyze the normalized training set using WEKA by applying the SVM classifier using the linear kernel, the default parameters and 10-fold cross validation. Figure 47 shows the output of the classifier on the training set with the Confusion Matrix.



LibSVM -S 0 -K 0 -D 3 -G 1.0 -R 0.0 -N 0.5 -M 40.0 -C 1.0 -E 0.001 -P 0.1 -seed 1

Cross-validation Folds 10

```
Test mode:10-fold cross-validation

=== Classifier model (full training set) ===

LibSVM wrapper, original code by Yasser EL-Manzalawy (= WLSVM)

Time taken to build model: 0.35 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        1750               99.9429 %
Incorrectly Classified Instances         1                0.0571 %
Kappa statistic                          0.9988
Mean absolute error                      0.0006
Root mean squared error                  0.0239
Relative absolute error                  0.1192 %
Root relative squared error              4.8821 %
Total Number of Instances             1751

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.999     0          1         0.999     1          1        Human
                 1         0.001      0.999     1         0.999      1        Bot
Weighted Avg.    0.999     0          0.999     0.999     0.999      1

=== Confusion Matrix ===

    a    b   <-- classified as
 1053    1 |    a = Human
    0  697 |    b = Bot
```

Figure 47: WEKA output of applying the SVM classifier with linear kernel on the training set.

We observe that the Linear SVM classifier yields an accuracy of 99.94% and from the confusion matrix observe that it misclassified only one human as a bot. Next, we validate this classifier with real-time human and bot mouse data samples as they attempt to solve the CAPTCHA. For our implementation we used the open source SVM package LIBSVM 4.0 [128]. LIBSVM is an integrated tool for support vector classification. This dual-themed CAPTCHA was hosted online at
http://www.darryl.cecsresearch.org/avatarzoocaptcha/ with human users and the bot solving the challenges. We used the C# library for LIBSVM with the Linear kernel. Humans are labeled 1 and bots 0 for the SVM process. Figure 48 shows a screenshot of the CAPTCHA which displays the outcome of the previous CAPTCHA challenge as well as that of the mouse classifier.

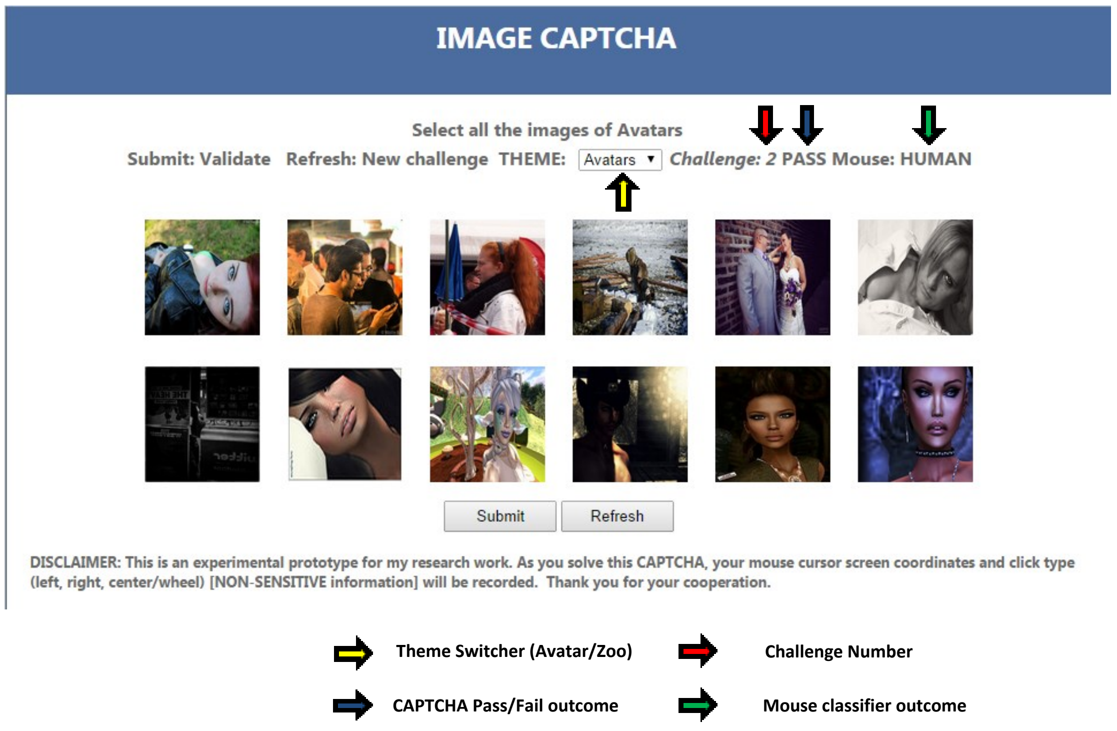


IMAGE CAPTCHA

Select all the images of Avatars

Submit: Validate Refresh: New challenge THEME: Avatars *Challenge: 2* PASS Mouse: HUMAN

DISCLAIMER: This is an experimental prototype for my research work. As you solve this CAPTCHA, your mouse cursor screen coordinates and click type (left, right, center/wheel) [NON-SENSITIVE information] will be recorded. Thank you for your cooperation.

Figure 48: Screenshot of the CAPTCHA depicting the outcomes of its validation and the mouse classifier.

The classifier validated their mouse usage activities as human or bot together with the CAPTCHA validation. First, 5 human users were made to solve 20 CAPTCHA challenges with the first 10 Avatar and the last 10 Zoo thus resulting in 100 human samples. Next, 7 human users were made to solve 40 CAPTCHA challenges with the first 20 Avatar and the last 20 Zoo thus, yielding 280 more human samples. Altogether, 100 + 280 = 380 human samples were acquired. Finally, the bot was executed on the CAPTCHA solving 380 challenges, first 190 Avatar and the last 190 Zoo yielding 380 bot samples.

Table 17: **Outcome of the Linear SVM classifier on real-time human and bot samples.**

Class	**Total validation samples**	**Accurate detection**	**Percentage**
Human	380	325	= 325/380 = 85.52%
Bot	380	380	= 380/380 = 100%

The results of the classifier are shown in Table 17. Here, we observe that the classifier performs exceptionally well in detecting the bot whereas it has an accuracy of close to 86% in detecting humans attempting to solve the CAPTCHA.

CHAPTER V
CONCLUSION AND FUTURE WORK

5.1 Conclusion

In today's online world, automated computer programs known as "bots" are posing security threats by affecting computer systems and causing attacks such as Denial of Service (DoS). These bots execute malicious scripts and predefined functions on an affected system. They allow hackers/attackers (bot creators) access to multiple computers at one time to create "botnets" to spread viruses, steal sensitive information such as passwords, bank account details, credit card numbers, rig online polls, sign up for free email accounts, send out spam emails as well as commit online crime. A CAPTCHA (Completely Automated Public Turing Tests to tell Computers and Humans Apart) is a program that generates and grades tests that most humans can pass but computers cannot. It is used as a tool to distinguish human users from bots. Text-based CAPTCHAs are the most popular however, they have limitations such as vulnerability to OCR attacks, overly distorted characters, English-language dependence and limited character-set size. One potential solution to overcome this is to use images and design Image CAPTCHAs.

Our work here involves designing one such novel image CAPTCHA to identify between natural (human) and artificial (avatar) faces. We test the security of our CAPTCHA by classifying the human and avatar faces using Uniform Local Directional Patterns (ULDP) and Wavelet Uniform Local Directional Patterns (WULDP) respectively. We also evaluate the security of our CAPTCHA by hosting a challenge at the ICMLA 2012 conference. The goal of this challenge was to determine how good are computer algorithms to classify human and avatar faces. On learning the potential pitfalls from this we improve and strengthen the CAPTCHA by acquiring images from Yahoo's Flickr based on random upload dates with various theme based challenges besides human-avatar (Avatar) such as animal (Zoo) and item (Object) images. Adding multiple themes thus presents a challenging computer vision problem for bots to solve and attack the CAPTCHA.

The security of the CAPTCHA is hardened by incorporating Behavioral biometrics in the form of Mouse Dynamics with it. Here it is tested by segregating human users from bots based on their mouse usage activities such as movement, click,etc. as they attempt to solve the CAPTCHA. Mouse Dynamics extracts features such as movement speed, click time, angle of curvature, etc. to build Mouse Dynamic Signature (MDS) profiles for human users and bots. Real-time evaluation of this security aspect is performed by building an SVM classifier using these MDS's to segregate human users from bots.

5.2 Future Work

There are possible extensions to our work based on extending the CAPTCHA design, improving the security to strengthen it and applying different behavioral biometric techniques with exploration of subtle features to help strengthen the human-bot classifier.

The CAPTCHA design can be improved in a few ways. Several different themes and Flickr groups can be explored and used in the CAPTCHA thus adding variety as well as strengthening it by posing an even greater challenge to image recognition bot algorithms. The number of images displayed can also be altered to present users an optimal set of images to identify and solve the CAPTCHA. The overall user interaction experience can be made much more fun and interesting for e.g. by designing a game from this CAPTCHA. Accessibility issues can be addressed as well. Besides Flickr, different image sources with available APIs can also be incorporated to provide a larger pool of images for the CAPTCHA challenges. Noise and distortion techniques can be examined to thwart image recognition algorithms. Mouse Dynamics can also be used as an efficient technique towards user authentication and intrusion detection i.e. different users can be profiled based on their mouse usage activities and can help segregate one user from another. Besides Mouse Dynamics, the user interaction with the Graphical User Interface (GUI) can also be examined and used as a behavioral biometric tool on the CAPTCHA. Several additional features can be explored to help strengthen the human-bot SVM classifier. Moreover, several different classifiers such as Decision trees, Naive Bayes, K-Nearest Neighbor etc. can also be used to perform a comparative analysis.

REFERENCES

[1] Marian Merritt. Bots and botnets: A growing threat. `http://us.norton.com/botnet/promo`.

[2] A. M. Turing. Computing Machinery and Intelligence. *Mind*, 59(236):433–460, 1950.

[3] L. Von Ahn, M. Blum, N. Hopper, and J. Langford. CAPTCHA: Using Hard AI Problems for Security. In *Proc. International Conference on the Theory and Applications of Cryptographic Techniques*, volume 2656 of *Lecture Notes in Computer Science*, pages 294–311, Warsaw, Poland, 2003. Eurocrypt.

[4] L. Von Ahn, M. Blum, and J. Langford. The CAPTCHA project. `http://www.captcha.net`.

[5] P. Y. Simard, D. Steinkraus, and J. C. Platt. Best practices for Convolutional Neural Networks applied to Visual Document Analysis. In *Proc. Seventh International Conference on Document Analysis and Recognition*, pages 958–963, Edinburgh, Scotland, 2003.

[6] D. Misra and K. Gaj. Face Recognition CAPTCHAs. In *Proc. International Conference on Internet and Web Applications and Services (AICT-ICIW '06)*, pages 122–122, Guadeloupe, French Caribbean, 2006.

[7] Free image-based CAPTCHA: VidoopCAPTCHA. `http://www.webresourcesdepot.com/free-image-based-captcha-vidoopcaptcha/`.

[8] Miriam Dyck, Maren Winbeck, Susanne Leiberg, Yuhan Chen, Rurben C. Gur, and Klaus Mathiak. Recognition profile of emotions in natural and virtual faces. *PLoS ONE*, 3(11):e3628, 2008.

[9] Shari Trewin, Mark Laff, Vicki Hanson, and Anna Cavender. Exploring visual and motor accessibility in navigating a virtual world. *ACM Transactions on Accessible Computing (TACCESS)*, 2(2):1–35, 2009.

[10] Second Life. http://www.secondlife.com.

[11] Entropia Universe. http://www.entropiauniverse.com.

[12] Sims Online. http://www.thesims.com.

[13] Kelley school of business, its executive education program to unveil virtual campus in second life. http://newsinfo.iu.edu/news/page/normal/8773.html, 2008.

[14] Marion Boberg, Petri Piippo, and Elina Ollila. Designing Avatars. In *Proc. 3rd International conference on Digital Interactive Media in Entertainment and Arts*, pages 232–239, Athens, Greece, 2008. ACM.

[15] Taskeed Jabid, Md. Hasanul Kabir, and Oksam Chae. Local Directional Pattern (LDP) - A robust image descriptor for Object Recognition. In *Proc. 7th IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 482–487, Boston, Massachussets, USA, 2010.

[16] Christophe Garcia, Giorgos Zikos, and Giorgos Tziritas. *A Wavelet-based Framework for Face Recognition*, pages 84–92. 5th European Conference on Computer Vision, 1998.

[17] Abdallah A. Mohamed, Darryl D'Souza, Naouel Baili, and Roman V. Yampolskiy. Avatar face recognition using wavelet transform and hierarchical multi-scale LBP. In *Proc. 10th International Conference on Machine Learning and Applications*, volume 1, pages 194–199, Honululu, HI, 2011.

[18] Nan Zheng, Aaron Paloski, and Haining Wang. An efficient user verification system via mouse movements. In *Proc. 18th ACM conference on Computer and communications security*, pages 139–150, Chicago, Illinois, USA, 2011.

[19] Roman V. Yampolskiy and Venu Govindaraju. Behavioural biometrics: A survey and classification. *International Journal of Biometrics*, 1(1):81–113, 2008.

[20] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[21] Support vector machine. http://en.wikipedia.org/wiki/Support_vector_machine.

[22] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. A practical guide to support vector classification.

[23] V Ragavi and G Geetha. CAPTCHA celebrating its quattuordecennial-a complete reference. *International Journal of Computer Science Issues (IJCSI)*, 8(6), 2011.

[24] Moni Naor. Verification of a human in the loop or identification via the turing test. *Unpublished draft from http://www. wisdom. weizmann. ac. il/ naor/PAPERS/human abs. html*, 1996.

[25] L. Von Ahn, M. Blum, N. Hopper, and J. Langford. CAPTCHA: Using hard ai problems for security. In Eli Biham, editor, *Proc. International Conference on the Theory and Applications of Cryptographic Techniques*, volume 2656 of *Lecture Notes in Computer Science*, pages 294–311, Warsaw, Poland, 2003. Eurocrypt.

[26] Henry S Baird. *Complex image recognition and web security*, pages 287–298. Springer, 2006.

[27] Henry S Baird and Kris Popat. *Human interactive proofs and document image analysis*, pages 507–518. Springer, 2002.

[28] H.S. Baird, A.L. Coates, and R.J. Fateman. Pessimalprint: A reverse turing test. *International Journal on Document Analysis and Recognition*, 5(2):158–163, 2003.

[29] M. Chew and H.S. Baird. Baffletext: A human interactive proof. *Society of Photo-Optical Instrumentation Engineers (SPIE)*, 2003.

[30] H.S. Baird, M.A. Moll, and S.Y. Wang. Scattertype: A legible but hard-to-segment CAPTCHA. In *Proc. Eighth International Conference on Document Analysis and Recognition, (ICDAR).*, pages 935–939, Seoul, South Korea, 2005. IEEE.

[31] Jon Bentley and Colin Mallows. CAPTCHA challenge strings: Problems and improvements. In *Document Recognition and Retrieval XIII*, pages 60670H–60670H–7, San Jose, California, USA, 2006. SPIE.

[32] Sui-Yu Wang and Henry S Baird. CAPTCHA challenge tradeoffs: Familiarity of strings versus degradation of images. In *18th International Conference on Pattern Recognition, ICPR.*, volume 3, pages 164–167, Hong Kong, China, 2006. IEEE.

[33] Rony Ferzli, Rida Bazzi, and Lina J Karam. A CAPTCHA based on the human visual systems masking characteristics. In *International Conference on Multimedia and Expo*, pages 517–520, Toronto, Canada, 2006. IEEE.

[34] K. Chellapilla, K. Larson, P. Simard, and M. Czerwinski. Designing human friendly human interaction proofs (HIPs). In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 711–720, Portland, Oregon, USA, 2005. ACM.

[35] Kumar Chellapilla, Kevin Larson, Patrice Y Simard, and Mary Czerwinski. *Building segmentation based human-friendly human interaction proofs (HIPs)*, pages 1–26. Springer, 2005.

[36] Bilal Khan, Khaled S Alghathbar, Muhammad Khurram Khan, Abdullah M AlKelabi, and Abdulaziz AlAjaji. *Using Arabic CAPTCHA for Cyber Security*, pages 8–17. Springer, 2010.

[37] Sushma Yalamanchili and M Kameswara Rao. A framework for Devanagari script-based CAPTCHA. *arXiv preprint arXiv:1109.0132*, 2011.

[38] Dong Chen. Research of the chinese CAPTCHA system based on AJAX. *WSEAS Trans. Cir. and Sys.*, 8(1):53–62, 2009.

[39] M Hassan Shirali-Shahreza and Mohammad Shirali-Shahreza. Multilingual CAPTCHA. In *International Conference on Computational Cybernetics, ICCC.*, pages 135–139, Gammarth, Tunisia, 2007. IEEE.

[40] Pawel Lupkowski and Mariusz Urbanski. Semcaptcha: The user-friendly alternative for OCR-based CAPTCHA systems. *Speech and Language Technology*, 11:278–289, 2008.

[41] C. R. Macias and E. Izquierdo. Visual word-based CAPTCHA using 3d characters. In *3rd International Conference on Crime Detection and Prevention (ICDP)*, pages 1–5, London, United Kingdom, 2009.

[42] Greg Mori and Jitendra Malik. Recognizing objects in adversarial clutter: Breaking a visual CAPTCHA. In *Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages I–134–I–141 vol. 1, Madison, Wisconsin, USA, 2003. IEEE.

[43] Greg Mori, Serge Belongie, and Jitendra Malik. Efficient shape matching using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(11):1832–1837, 2005.

[44] Gabriel Moy, Nathan Jones, Curt Harkless, and Randall Potter. Distortion estimation techniques in solving visual CAPTCHAs. In *Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages II–23–II–28 Vol. 2, Washington DC, USA, 2004. IEEE.

[45] Arasanathan Thayananthan, Bjoern Stenger, Torr Philip HS, and Roberto Cipolla. Shape context and chamfer matching in cluttered scenes. In *Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages I–127–I–133 vol. 1, Madison, Wisconsin, USA, 2003. IEEE.

[46] Miroslav Ponec. Visual reverse turing tests: A false sense of security. In *Information Assurance Workshop*, pages 305–311, West Point, NY, USA, 2006. IEEE.

[47] Jeff Yan and Ahmad Salah El Ahmad. Breaking visual CAPTCHAs with naive pattern recognition algorithms. In *Twenty-Third Annual Computer Security Applications Conference (ACSAC).*, pages 279–291, Miami Beach, FL, USA, 2007. IEEE.

[48] PY Simard. Using machine learning to break visual human interaction proofs (HIPs. *Advances in neural information processing systems*, 17:265–272, 2005.

[49] J. Yan and A.S. El Ahmad. A low-cost attack on a Microsoft CAPTCHA. In *Proceedings of the 15th ACM conference on Computer and communications security*, pages 543–554, Alexandria, VA, USA, 2008. ACM.

[50] H.S. Baird and J.L. Bentley. Implicit CAPTCHAs. In *Document Recognition and Retrieval XII*, pages 191–196, San Jose, California, USA, 2005. SPIE.

[51] M.Chew and J.Tygar. Image recognition CAPTCHAs. In *Proc. Information Security Conference*, volume 3225, pages 268–279.

[52] L. von Ahn. Human Computation. In *Proc. 46th ACM/IEEE Design Automation Conference*, pages 418–419, San Francisco, California, USA, 2009.

[53] J. Elson, J. Douceur, J. Howell, and J. Saul. ASIRRA: A CAPTCHA that exploits interest-aligned manual image categorization. In *Proc. 14th ACM conference on Computer and Communications Security*, pages 366–374, Alexandria, Virginia, USA, 2007.

[54] R. Gossweiler, M. Kamvar, and S. Baluja. What's up CAPTCHA?: A CAPTCHA based on image orientation. In *Proceedings of the 18th international conference on World Wide Web*, pages 841–850, Madrid, Spain, 2005. ACM.

[55] Ritendra Datta, Jia Li, and James Z. Wang. IMAGINATION: a robust image-based CAPTCHA generation system. In *Proc. 13th annual ACM international conference on Multimedia*, pages 331–334, 1101218, 2005. ACM.

[56] Y. Rui and Z. Liu. ARTiFACIAL: Automated Reverse Turing test using FACIAL features. *Multimedia Systems*, 9(6):493–502, 2004.

[57] Gaurav Goswami, Richa Singh, Mayank Vatsa, Brian Powell, and Afzel Noore. Face recognition CAPTCHA. In *Fifth International Conference on Biometrics: Theory, Applications and Systems (BTAS)*, pages 412–417, Arlington, Virginia, USA, 2012. IEEE.

[58] Gaurav Goswami, Brian M Powell, Mayank Vatsa, Richa Singh, and Afzel Noore. FaceD-CAPTCHA: Face detection based color image CAPTCHA. *Future Generation Computer Systems*, 31:59–68, 2014.

[59] D. D'Souza, P. C. Polina, and R. V. Yampolskiy. Avatar CAPTCHA: Telling computers and humans apart via face classification. In *Proc. IEEE International Conference on Electro/Information Technology (EIT)*, pages 1–6, Indianapolis, Indiana, USA, 2012.

[60] Jonghak Kim, Sangtae Kim, Joonhyuk Yang, Jung-hee Ryu, and KwangYun Wohn. Face-CAPTCHA: A CAPTCHA that identifies the gender of face images unrecognized by existing gender classifiers. *Multimedia Tools and Applications*, pages 1–23, 2013.

[61] Montree Imsamai and Suphakant Phimoltares. 3D CAPTCHA: A next generation of the CAPTCHA. In *International Conference on Information Science and Applications (ICISA)*, pages 1–8, Seoul, South Korea, 2010. IEEE.

[62] Jing-Song Cui, Jing-Ting Mei, Xia Wang, Da Zhang, and Wu-Zhou Zhang. A CAPTCHA implementation based on 3D animation. In *International Conference on Multimedia Information Networking and Security, (MINES).*, volume 2, pages 179–182, Hubei, China, 2009. IEEE.

[63] Ibrahim Furkan Ince, Yucel Batu Salman, Mustafa Eren Yildirim, and Tae-Cheon Yang. Execution time prediction for 3D interactive CAPTCHA by keystroke level model. In *Fourth International Conference on Computer Sciences and Convergence Information Technology (ICCIT).*, pages 1057–1061, Seoul, South Korea, 2009. IEEE.

[64] Mohammad Shirali-Shahreza and Sajad Shirali-Shahreza. Online collage CAPTCHA. In *Eighth International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, pages 58–58, Santorini, Greece, 2007. IEEE.

[65] TS Ravi Kiran and Y Rama Krishna. Combining CAPTCHA and graphical passwords for user authentication. 2(4):29–35, 2012.

[66] T. Yamamoto, T. Suzuki, and M. Nishigaki. A proposal of four-panel cartoon CAPTCHA: The concept. In *Proc. 13th International Conference on Network-Based Information Systems (NBiS)*, pages 575–578, Takayama, Japan, 2010. IEEE.

[67] H. Gao, D. Yao, H. Liu, X. Liu, and L. Wang. A novel image based CAPTCHA using jigsaw puzzle. In *Proc. IEEE 13th International Conference on Computational Science and Engineering (CSE)*, pages 351–356, Hong Kong, China, 2010. IEEE.

[68] A. Jain, A. Raj, and T. Pahwa. Sequenced picture CAPTCHA: Generation and its strength analysis. In *Proc. International Conference for Internet Technology and Secured Transactions (ICITST).*, pages 1–8, London, United Kingdom, 2009. IEEE.

[69] Mohammad Shirali-Shahreza. Highlighting CAPTCHA. In *Conference on Human System Interactions*, pages 247–250, Krakow, Poland, 2008. IEEE.

[70] M Hassan Shirali-Shahreza and Mohammad Shirali-Shahreza. An anti-sms-spam using CAPTCHA. In *ISECS International Colloquium on Computing, Communication, Control and Management (CCCM)*, volume 2, pages 318–321, Guangzhou, China, 2008. IEEE.

[71] Rosa Lin, Shih-Yu Huang, Graeme B Bell, and Yeuan-Kuen Lee. A new CAPTCHA interface design for mobile devices. In *Proceedings of the Twelfth Australasian User Interface Conference-Volume 117*, pages 3–8, Australia, 2011. Australian Computer Society, Inc.

[72] Donn Morrison, Stephane Marchand-Maillet, and Eric Bruno. Tagcaptcha: Annotating images with CAPTCHAs. In *Proceedings of the ACM SIGKDD Workshop on Human Computation*, pages 44–45, New York, USA, 2009. ACM.

[73] Bin B. Zhu, Jeff Yan, Qiujie Li, Chao Yang, Jia Liu, Ning Xu, Meng Yi, and Kaiwei Cai. Attacks and design of image recognition CAPTCHAs. In *Proc. 17th ACM conference on Computer and communications security*, pages 187–200, Chicago, Illinois, USA, 2010. ACM.

[74] David Lorenzi, Jaideep Vaidya, Emre Uzun, Shamik Sural, and Vijayalakshmi Atluri. *Attacking Image Based CAPTCHAs Using Image Recognition Techniques*, volume 7671 of *Lecture Notes in Computer Science*, chapter 23. Springer Berlin Heidelberg.

[75] Philippe Golle. Machine learning attacks against the Asirra CAPTCHA. In *Proc. 15th ACM conference on Computer and communications security*, pages 535–542, 1455838, 2008. ACM.

[76] Carlos Javier Hernandez-Castro, Arturo Ribagorda, and Yago Saez. Side-channel attack on the HumanAuth CAPTCHA. In *Proceedings of the International Conference on Security and Cryptography (SECRYPT)*, pages 1–7, Athens, Greece, 2010. IEEE.

[77] Md Zia Uddin, Jeong Tai Kim, and Tae-Seong Kim. Depth video-based gait recognition for smart home using local directional pattern features and hidden markov model. *Indoor and Built Environment*, 23(1):133–140, 2014.

[78] Ravinder Kumar, Pravin Chandra, and Madasu Hanmandlu. Local directional pattern (LDP) based fingerprint matching using SLFNN. In *Second International Conference on Image Information Processing (ICIIP)*, pages 493–498, Shimla, India, 2013. IEEE.

[79] Darryl D'Souza and Roman Yampolskiy. Natural vs artificial face classification using uniform local directional patterns and wavelet uniform local directional patterns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 27–33, Columbus, Ohio, USA, 2014.

[80] T. Jabid, M. H. Kabir, and O. Chae. Facial expression recognition using local directional pattern (LDP). In *Proc. 17th IEEE International Conference on Image Processing (ICIP)*, pages 1605–1608, Hong Kong, China, 2010.

[81] Fujin Zhong and Jiashu Zhang. Face recognition with enhanced local directional patterns. *Neurocomputing*, 119:375–384, 2013.

[82] T. Jabid, M. Hasanul Kabir, and Chae Oksam. Gender classification using local directional pattern (LDP). In *Proc. 20th International Conference on Pattern Recognition (ICPR)*, pages 2162–2165, Istanbul, Turkey, 2010.

[83] Ingrid Daubechies. The wavelet transform, time-frequency localization and signal analysis. *IEEE Transactions on Information Theory*, 36(5):961–1005, 1990.

[84] Chih-Chin Lai and Cheng-Chih Tsai. Digital image watermarking using discrete wavelet transform and singular value decomposition. *IEEE Transactions on Instrumentation and Measurement*, 59(11):3060–3063, 2010.

[85] AhcÃšne Bouzida, Omar Touhami, Rachid Ibtiouen, Adel Belouchrani, Maurice Fadel, and Abderrezak Rezzoug. Fault diagnosis in industrial induction machines through discrete wavelet transform. *IIEEE Transactions on Industrial Electronics*, 58(9):4385–4395, 2011.

[86] Hasan Demirel and Gholamreza Anbarjafari. Discrete wavelet transform-based satellite image resolution enhancement. *IEEE Transactions on Geoscience and Remote Sensing*, 49(6):1997–2004, 2011.

[87] Haifeng Hu. Variable lighting face recognition using discrete wavelet transform. *Pattern Recognition Letters*, 32(13):1526–1534, 2011.

[88] K Ramesha and KB Raja. *Face Recognition System Using Discrete Wavelet Transform and Fast PCA*, pages 13–18. Springer, 2011.

[89] Preeti Rai and Pritee Khanna. Gender classification using radon and wavelet transforms. In *International Conference on Industrial and Information Systems (ICIIS)*, pages 448–451, Mangalore, India, 2010. IEEE.

[90] Abdallah A Mohamed and Roman V Yampolskiy. Using discrete wavelet transform and eigenfaces for recognizing avatars faces. In *17th International Conference on Computer Games (CGAMES)*, pages 143–147, Louisville, Kentucky, USA, 2012. IEEE.

[91] Chao Shen, Zhongmin Cai, Xiaohong Guan, and Roy Maxion. Performance evaluation of anomaly-detection algorithms for mouse dynamics. *Computers and Security*, 45:156–171, 2014.

[92] AgusFanar Syukri, Eiji Okamoto, and Masahiro Mambo. *A user identification system using signature written with mouse*, volume 1438 of *Lecture Notes in Computer Science*, chapter 36, pages 403–414. Springer Berlin Heidelberg, 1998.

[93] Ross AJ Everitt and Peter W McOwan. Java-based internet biometric authentication system. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9):1166–1172, 2003.

[94] Shivani Hashia, Chris Pollett, and Mark Stamp. On using mouse movements as a biometric, May 9 - May 12 2005. CiteSeerX - Scientific Literature Digital Library and Search Engine [http://citeseerx.ist.psu.edu/oai2] (United States) ER.

[95] H Gamboa, ALN Fred, and AK Jain. Webbiometrics: user verification via web interaction. In *Biometrics Symposium*, pages 1–6, Baltimore, Maryland, USA, 2007. IEEE.

[96] Kenneth Revett, Hamid Jahankhani, SÃ©rgio Tenreiro MagalhÃ£es, and Henrique M. D. Santos. *A Survey of User Authentication Based on Mouse Dynamics Global E-*

Security, volume 12 of *Communications in Computer and Information Science*, pages 210–219. Springer Berlin Heidelberg, 2008.

[97] Patrick Bours and Christopher Johnsrud Fullu. A login system using mouse dynamics. In *Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)*, pages 1072–1077, Kyoto, Japan, 2009. IEEE.

[98] Y. Aksari and H. Artuner. Active authentication by mouse movements. In *24th International Symposium on Computer and Information Sciences (ISCIS)*, pages 571–574, Guzelyurt, Turkey, 2009.

[99] Hugo Gamboa and Ana Fred. A behavioral biometric system based on human-computer interaction. *SPIE-The International Society for Optical Engineering*, 5404:381–392, 2004.

[100] Maja Pusara and Carla E. Brodley. User re-authentication via mouse movements. In *Proc. ACM workshop on Visualization and data mining for computer security*, pages 1–8, 1029210, 2004. ACM.

[101] D. A. Schulz. Mouse curve biometrics. In *Biometrics Symposium: Special Session on Research at the Biometric Consortium Conference*, pages 1–6, Baltimore, Maryland, USA, 2006. IEEE.

[102] A. A. E. Ahmed and I. Traore. A new biometric technology based on mouse dynamics. *IEEE Transactions on Dependable and Secure Computing*, 4(3):165–179, 2007.

[103] Chao Shen, Zhongmin Cai, and Xiaohong Guan. Continuous authentication for mouse dynamics: A pattern-growth approach. In *42nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 1–12, Boston, Massachussetts, USA, 2012. IEEE.

[104] Soumik Mondal and Patrick Bours. Continuous authentication using mouse dynamics. In *International Conference of the Biometrics Special Interest Group (BIOSIG)*, pages 1–12, Darmstadt, Germany, 2013. IEEE.

[105] Roman V. Yampolskiy and Venu Govindaraju. Behavioral biometrics for verification and recognition of malicious software agents. In *Sensors, and Command, Control, Communications, and Intelligence (C3I) Technologies for Homeland Security and Homeland Defense VII*, volume 6943, Orlando, Florida, USA, 2008.

[106] Roman V. Yampolskiy and Venu Govindaraju. Use of behavioral biometrics in intrusion detection and online gaming. In *Biometric Technology for Human Identification III*, pages 62020U–62020U–10, Orlando, Florida, USA, 2006. SPIE.

[107] Steven Gianvecchio, Zhenyu Wu, Mengjun Xie, and Haining Wang. Battle of botcraft: fighting bots in online games with human observational proofs. In *Proceedings of the 16th ACM conference on Computer and communications security*, pages 256–268, Chicago, Illinois, USA, 2009. ACM.

[108] Zi Chu, Steven Gianvecchio, Aaron Koehl, Haining Wang, and Sushil Jajodia. Blog or block: Detecting blog bots through behavioral biometrics. *Computer Networks*, 57(3):634–646, 2013.

[109] Zi Chu, Steven Gianvecchio, Haining Wang, and Sushil Jajodia. Who is tweeting on twitter: human, bot, or cyborg? In *Proceedings of the 26th annual computer security applications conference*, pages 21–30, New Orleans, Louisiana, USA, 2014. ACM.

[110] Nottingham_scans. Mainly frontal views, some profile, some differences in lighting and expression variation.

[111] J. N. Oursler, M. Price, and R. V. Yampolskiy. Parameterized generation of Avatar face dataset. In *14th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games*, pages 17–22, Louisville, Kentucky, USA, 2009.

[112] Roman V. Yampolskiy and Darryl D'Souza. 11th International Conference on Machine Learning and Applications ICMLA 2012.

[113] Toshihiko Yamasaki and Tsuhan Chen. Face recognition challenge: Object recognition approaches for human/avatar classification. In *11th International Conference on Machine Learning and Applications (ICMLA)*, volume 2, pages 574–579, Boca Raton, Florida, USA, 2012. IEEE.

[114] Brian Cheung. Convolutional neural networks applied to human face classification. In *11th International Conference on Machine Learning and Applications (ICMLA)*, volume 2, pages 580–583, Boca Raton, Florida, USA, 2012. IEEE.

[115] Mohammed Korayem, Abdallah A Mohamed, David Crandall, and Roman V Yampolskiy. Learning visual features for the Avatar CAPTCHA recognition challenge. In *11th International Conference on Machine Learning and Applications (ICMLA)*, volume 2, pages 584–587, Boca Raton, Florida, USA, 2012. IEEE.

[116] Salem Alelyani and Huan Liu. Ensemble feature selection in face recognition: ICMLA 2012 challenge. In *11th International Conference on Machine Learning and Applications (ICMLA)*, volume 2, pages 588–591, Boca Raton, Florida, USA, 2012. IEEE.

[117] M. Weber. A Frontal face dataset collected by Markus Weber at the California Institute of Technology. `http://www.vision.caltech.edu/Image_Datasets/faces`.

[118] PJ Phillips. The Facial Recognition Technology (FERET) Database. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22, 2004.

[119] James Kim, Darryl D'Souza, and Roman V. Yampolskiy. Automated Collection of High Quality 3D Avatar images. In *Proc. 23rd Midwest Artificial Intelligence and Cognitive Science*, volume 841, page 78, Cincinnati, Ohio, USA, 2012.

[120] Flickr from Yahoo. `http://www.flickr.com/`.

[121] Yahoo. Flickr API. `http://www.flickr.com/services/api/`.

[122] Yahoo. Flickr APIs terms of use. `http://www.flickr.com/services/api/tos/`.

[123] Darryl D'Souza, Jacob Matchuny, and Roman Yampolskiy. Zoo CAPTCHA: Telling computers and humans apart via animal image classification. In *The Third ASE International Conference on Cyber Security*, Stanford, California, USA, 2014. Academy of Science and Engineering (ASE),© ASE 2014.

[124] Chao Shen, Zhongmin Cai, Xiaohong Guan, Youtian Du, and Roy A Maxion. User authentication through mouse dynamics. *Information Forensics and Security, IEEE Transactions on*, 8(1):16–30, 2013.

[125] Google images. `https://www.google.com/imghp`.

[126] Tineye reverse image search. `https://www.tineye.com`.

[127] Geoffrey Holmes, Andrew Donkin, and Ian H Witten. WEKA: A machine learning workbench. In *Proceedings of the Second Australian and New Zealand Conference on Intelligent Information Systems*, pages 357–361, Brisbane, Australia, 1994. IEEE.

[128] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

APPENDIX A

Here, I briefly describe the role of the functions used in this work.

void Page_Load (object, EventArgs):

- Sets a random number of images from the right class to be displayed.
- Reads the images from the database
- Shuffles the set of 12 images to be displayed in the random order
- Displays them on the web page
- Starts a stopwatch timer to begin the session recording time

void GetImagesfromDB (string[], int, int, int[], bool):

- Switch themes appropriately
- Pick a set of 12 images from a random location in the image database

string[] Shuffler (string[], int[], int):

- Shuffles the set of 12 images to be displayed in random order

void ValidateUser(int, int,string[]):

- Validates the user choices on hitting the Submit button
- Tracks and saves the misclassified images to the database

void Display(bool, string[], int, int, int[]):

- Displays the shuffled images on the web page

void SubmitKey_Click(object, EventArgs):

- Validate the challenge
- Stop the stopwatch timer to end the session recording time

- Record the raw mouse data into files
- Extract the features to form the feature vector
- Write the feature vector to file
- Normalize the feature vector using min-max normalization
- Write the normalized feature vector to file
- Apply the LibSVM classifier i.e. train and test

void Refresh(bool):

- Calls the Page_Load function to display
- Reset the selected/highlighted tiles from the previous challenge

void LibSVMTrain():

- Train the SVM classifier by reading the training file and applying the appropriate kernel

void LibSVMTest():

- Evaluate the incoming mouse data based on the trained/learned model

void ReadFromTextFile(string[], int, int, ref double[,]):

- Read the saved text files containing raw mouse data to form the raw training dataset. Each saved text file contains data per CAPTCHA challenge attempted

void SpeedAccelerationRecorder(mousedata, mousedata, List<double>[13],double):

- Evaluates the Euclidean distance between consecutive data points and the corresponding time difference based from the timestamps
- Evaluates speed, acceleration, silence time, movement time and the speed in all the eight directions

double StdDeviationCalculator(List<double>, double):

- Evaluates the Standard deviation for the recorded mouse data points

double EuclideanDistanceComputer(int, int, int, int):

- Evaluates the Euclidean distance between two recorded mouse data points

void AngleofCurvatureRecorder(mousedata, mousedata, mousedata, List<double>):

- Evaluates the Angle of Curvature between three consecutive recorded mouse data points

void NormalizeMinMaxData(double[], double[,]):

- Apply min-max normalization to the training dataset

void WriteRawDataToFile(bool, List<mousedata>, int, int):

- Write the raw mouse data to individual text files as users attempt to solve the CAPTCHA challenges

void WriteNormalizedDataToFile(bool, double[], string, int):

- Write the normalized data into two separate text files based on two different formats for processing. One for WEKA and the other for MATLAB

void WriteToFile(bool, double, string[], int):

- Write the feature vector data into two separate text files based on two different formats for processing. One for WEKA and the other for MATLAB

void Submit_Feedback_Click(object, EventArgs):

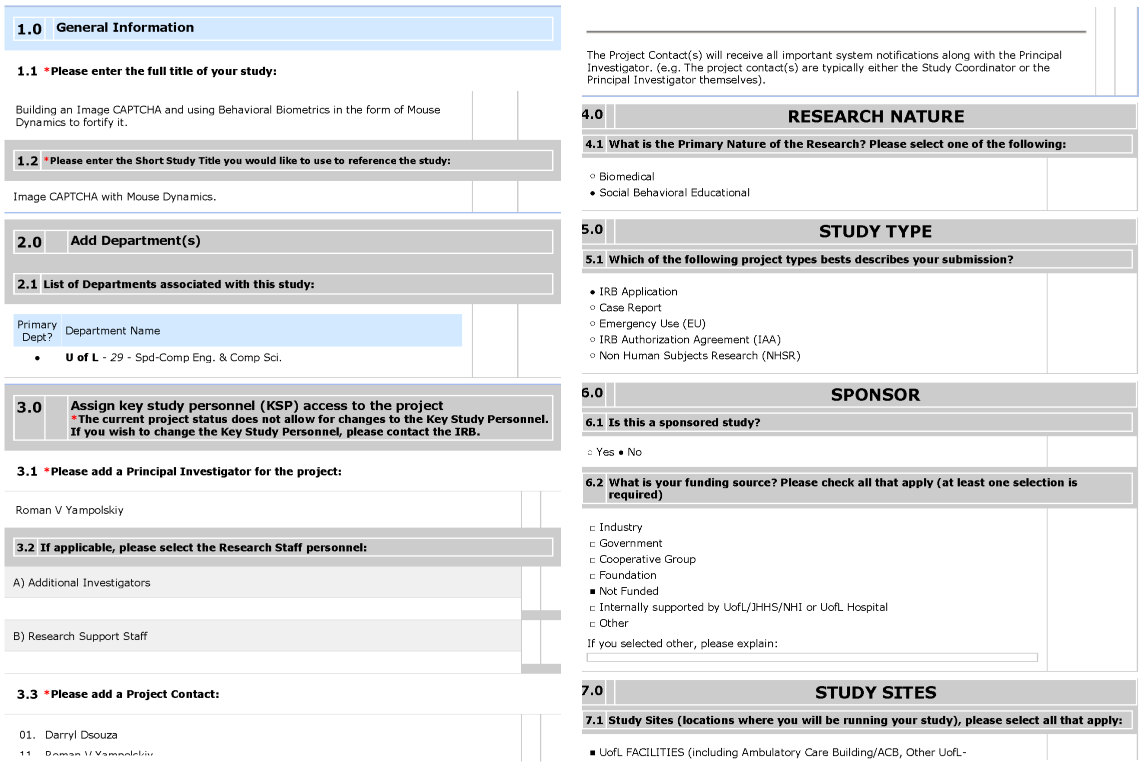
- Check if any fields on the Feedback page is empty or invalid
- If not, then save the entries to the database
- Redirect to the Thank you page

void BackToTest_Click(object, EventArgs):

- Redirect back to a new CAPTCHA challenge

APPENDIX B

Here are the application and approval forms from the Institutional Review Board (IRB) for recruiting human subjects to solve the CAPTCHA challenges



1.0 General Information

1.1 *Please enter the full title of your study:

Building an Image CAPTCHA and using Behavioral Biometrics in the form of Mouse Dynamics to fortify it.

1.2 *Please enter the Short Study Title you would like to use to reference the study:

Image CAPTCHA with Mouse Dynamics.

2.0 Add Department(s)

2.1 List of Departments associated with this study:

Primary Dept?	Department Name
•	**U of L** - *29* - Spd-Comp Eng. & Comp Sci.

3.0 Assign key study personnel (KSP) access to the project
***The current project status does not allow for changes to the Key Study Personnel. If you wish to change the Key Study Personnel, please contact the IRB.**

3.1 *Please add a Principal Investigator for the project:

Roman V Yampolskiy

3.2 If applicable, please select the Research Staff personnel:

A) Additional Investigators

B) Research Support Staff

3.3 *Please add a Project Contact:

01. Darryl Dsouza
11. Roman V Yampolskiy

The Project Contact(s) will receive all important system notifications along with the Principal Investigator. (e.g. The project contact(s) are typically either the Study Coordinator or the Principal Investigator themselves).

4.0 RESEARCH NATURE

4.1 What is the Primary Nature of the Research? Please select one of the following:

- ○ Biomedical
- [✓ Social Behavioral Educational]

5.0 STUDY TYPE

5.1 Which of the following project types bests describes your submission?

- [✓ IRB Application]
- ○ Case Report
- ○ Emergency Use (EU)
- ○ IRB Authorization Agreement (IAA)
- ○ Non Human Subjects Research (NHSR)

6.0 SPONSOR

6.1 Is this a sponsored study?

○ Yes [✓ No]

6.2 What is your funding source? Please check all that apply (at least one selection is required)

- ☐ Industry
- ☐ Government
- ☐ Cooperative Group
- ☐ Foundation
- [✓ Not Funded]
- ☐ Internally supported by UofL/JHHS/NHI or UofL Hospital
- ☐ Other

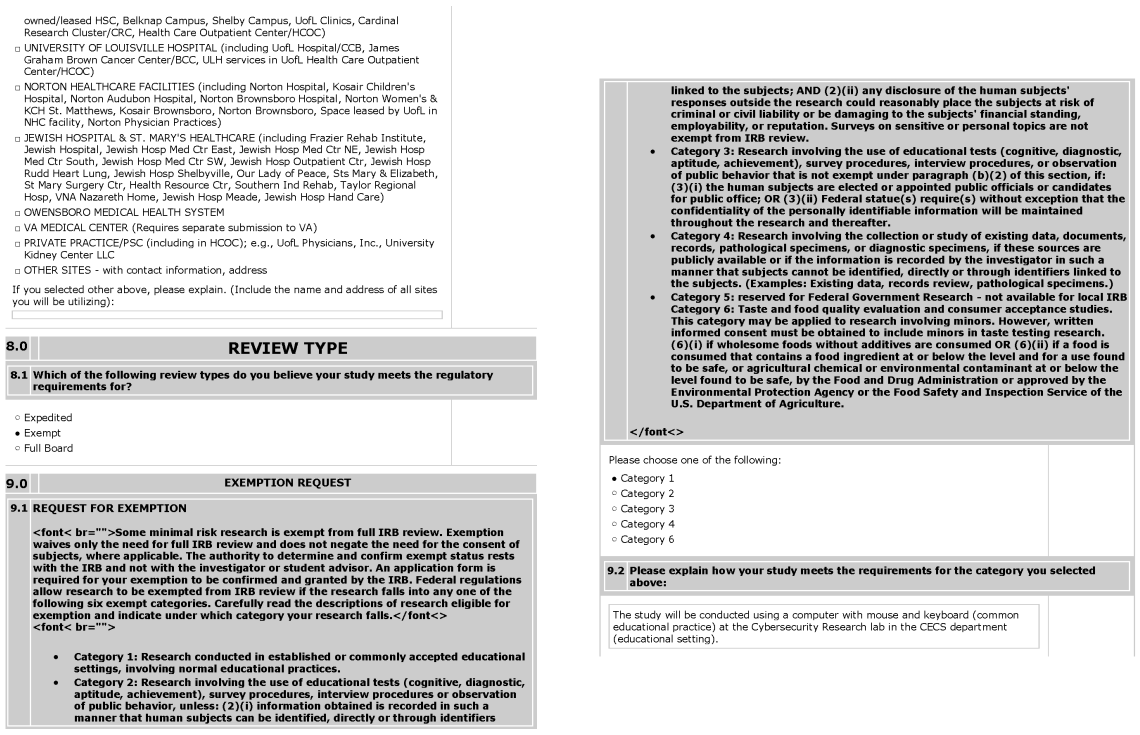
If you selected other, please explain:

7.0 STUDY SITES

7.1 Study Sites (locations where you will be running your study), please select all that apply:

- [✓ UofL FACILITIES (including Ambulatory Care Building/ACB, Other UofL-]

Figure 49: IRB Application form (Pages 1-2)



owned/leased HSC, Belknap Campus, Shelby Campus, UofL Clinics, Cardinal Research Cluster/CRC, Health Care Outpatient Center/HCOC)

- ☐ UNIVERSITY OF LOUISVILLE HOSPITAL (including UofL Hospital/CCB, James Graham Brown Cancer Center/BCC, ULH services in UofL Health Care Outpatient Center/HCOC)
- ☐ NORTON HEALTHCARE FACILITIES (including Norton Hospital, Kosair Children's Hospital, Norton Audubon Hospital, Norton Brownsboro Hospital, Norton Women's & KCH St. Matthews, Kosair Brownsboro, Norton Brownsboro, Space leased by UofL in NHC facility, Norton Physician Practices)
- ☐ JEWISH HOSPITAL & ST. MARY'S HEALTHCARE (including Frazier Rehab Institute, Jewish Hospital, Jewish Hosp Med Ctr East, Jewish Hosp Med Ctr NE, Jewish Hosp Med Ctr South, Jewish Hosp Med Ctr SW, Jewish Hosp Outpatient Ctr, Jewish Hosp Rudd Heart Lung, Jewish Hosp Shelbyville, Our Lady of Peace, Sts Mary & Elizabeth, St Mary Surgery Ctr, Health Resource Ctr, Southern Ind Rehab, Taylor Regional Hosp, VNA Nazareth Home, Jewish Hosp Meade, Jewish Hosp Hand Care)
- ☐ OWENSBORO MEDICAL HEALTH SYSTEM
- ☐ VA MEDICAL CENTER (Requires separate submission to VA)
- ☐ PRIVATE PRACTICE/PSC (including in HCOC); e.g., UofL Physicians, Inc., University Kidney Center LLC
- ☐ OTHER SITES - with contact information, address

If you selected other above, please explain. (Include the name and address of all sites you will be utilizing):

8.0 REVIEW TYPE

8.1 Which of the following review types do you believe your study meets the regulatory requirements for?

- ○ Expedited
- ● Exempt
- ○ Full Board

9.0 EXEMPTION REQUEST

9.1 REQUEST FOR EXEMPTION

<font< br="">Some minimal risk research is exempt from full IRB review. Exemption waives only the need for full IRB review and does not negate the need for the consent of subjects, where applicable. The authority to determine and confirm exempt status rests with the IRB and not with the investigator or student advisor. An application form is required for your exemption to be confirmed and granted by the IRB. Federal regulations allow research to be exempted from IRB review if the research falls into any one of the following six exempt categories. Carefully read the descriptions of research eligible for exemption and indicate under which category your research falls.</font<>
<font< br="">

- **Category 1: Research conducted in established or commonly accepted educational settings, involving normal educational practices.**
- **Category 2: Research involving the use of educational tests (cognitive, diagnostic, aptitude, achievement), survey procedures, interview procedures or observation of public behavior, unless: (2)(i) information obtained is recorded in such a manner that human subjects can be identified, directly or through identifiers linked to the subjects; AND (2)(ii) any disclosure of the human subjects' responses outside the research could reasonably place the subjects at risk of criminal or civil liability or be damaging to the subjects' financial standing, employability, or reputation. Surveys on sensitive or personal topics are not exempt from IRB review.**
- **Category 3: Research involving the use of educational tests (cognitive, diagnostic, aptitude, achievement), survey procedures, interview procedures, or observation of public behavior that is not exempt under paragraph (b)(2) of this section, if: (3)(i) the human subjects are elected or appointed public officials or candidates for public office; OR (3)(ii) Federal statue(s) require(s) without exception that the confidentiality of the personally identifiable information will be maintained throughout the research and thereafter.**
- **Category 4: Research involving the collection or study of existing data, documents, records, pathological specimens, or diagnostic specimens, if these sources are publicly available or if the information is recorded by the investigator in such a manner that subjects cannot be identified, directly or through identifiers linked to the subjects. (Examples: Existing data, records review, pathological specimens.)**
- **Category 5: reserved for Federal Government Research - not available for local IRB Category 6: Taste and food quality evaluation and consumer acceptance studies. This category may be applied to research involving minors. However, written informed consent must be obtained to include minors in taste testing research. (6)(i) if wholesome foods without additives are consumed OR (6)(ii) if a food is consumed that contains a food ingredient at or below the level and for a use found to be safe, or agricultural chemical or environmental contaminant at or below the level found to be safe, by the Food and Drug Administration or approved by the Environmental Protection Agency or the Food Safety and Inspection Service of the U.S. Department of Agriculture.**

</font<>

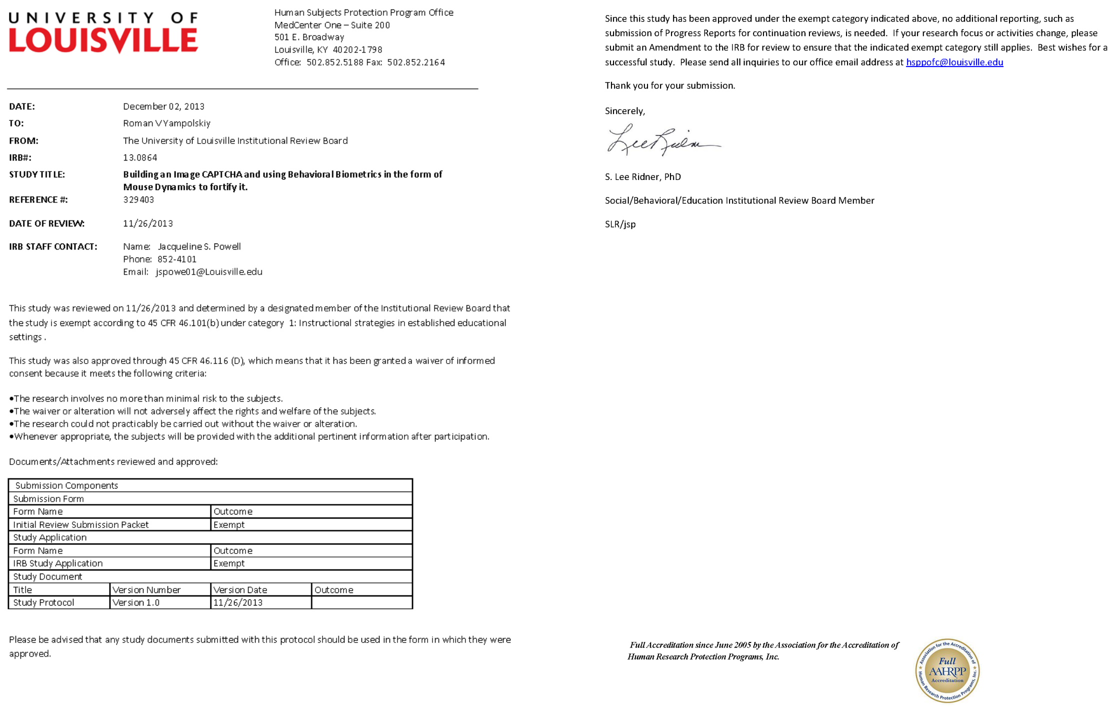
Please choose one of the following:

- ● Category 1
- ○ Category 2
- ○ Category 3
- ○ Category 4
- ○ Category 6

9.2 Please explain how your study meets the requirements for the category you selected above:

The study will be conducted using a computer with mouse and keyboard (common educational practice) at the Cybersecurity Research lab in the CECS department (educational setting).

Figure 50: IRB Application form (Pages 3-4)



UNIVERSITY OF LOUISVILLE

Human Subjects Protection Program Office
MedCenter One – Suite 200
501 E. Broadway
Louisville, KY 40202-1798
Office: 502.852.5188 Fax: 502.852.2164

DATE:	December 02, 2013
TO:	Roman V Yampolskiy
FROM:	The University of Louisville Institutional Review Board
IRB#:	13.0864
STUDY TITLE:	**Building an Image CAPTCHA and using Behavioral Biometrics in the form of Mouse Dynamics to fortify it.**
REFERENCE #:	329403
DATE OF REVIEW:	11/26/2013
IRB STAFF CONTACT:	Name: Jacqueline S. Powell Phone: 852-4101 Email: jspowe01@Louisville.edu

This study was reviewed on 11/26/2013 and determined by a designated member of the Institutional Review Board that the study is exempt according to 45 CFR 46.101(b) under category 1: Instructional strategies in established educational settings .

This study was also approved through 45 CFR 46.116 (D), which means that it has been granted a waiver of informed consent because it meets the following criteria:

- The research involves no more than minimal risk to the subjects.
- The waiver or alteration will not adversely affect the rights and welfare of the subjects.
- The research could not practicably be carried out without the waiver or alteration.
- Whenever appropriate, the subjects will be provided with the additional pertinent information after participation.

Documents/Attachments reviewed and approved:

Submission Components			
Submission Form			
Form Name		Outcome	
Initial Review Submission Packet		Exempt	
Study Application			
Form Name		Outcome	
IRB Study Application		Exempt	
Study Document			
Title	Version Number	Version Date	Outcome
Study Protocol	Version 1.0	11/26/2013	

Please be advised that any study documents submitted with this protocol should be used in the form in which they were approved.

Since this study has been approved under the exempt category indicated above, no additional reporting, such as submission of Progress Reports for continuation reviews, is needed. If your research focus or activities change, please submit an Amendment to the IRB for review to ensure that the indicated exempt category still applies. Best wishes for a successful study. Please send all inquiries to our office email address at hsppofc@louisville.edu

Thank you for your submission.

Sincerely,

[illegible]

S. Lee Ridner, PhD

Social/Behavioral/Education Institutional Review Board Member

SLR/jsp

Full Accreditation since June 2005 by the Association for the Accreditation of Human Research Protection Programs, Inc.

Figure 51: IRB Approval form (Pages 1-2)

CURRICULUM VITAE

DARRYL FELIX D'SOUZA

Cyber Security Laboratory, Department of Computer Engineering and Computer Science,

University of Louisville, Louisville, KY, 40292

Email: darryl_dsouza@hotmail.com

Bio: http://cecs.louisville.edu/security/DarrylBio.shtml

LinkedIn: http://linkedin.com/in/darryldsouza

Twitter: @imdarryldsouza

OBJECTIVE

Experienced in Behavioral Web Security by designing, developing and implementing solutions for intrusion detection by bots. Skilled in various software languages with hands-on experience in software development and programming. Ability to rapidly learn and utilize emerging technologies.

SKILLS

- **Programming**: C/C++, C#, Java, MATLAB
- **Scripting**: Python, PHP
- **Web Scripting**: HTML, JavaScript
- **Database**: MySQL, SQL Server
- **IDE**'s: Visual Studio, Eclipse

WORK EXPERIENCE

Ph.D. Candidate (August 2009 – October 2014)

Cyber Security Lab, Department of Computer Engineering and Computer Science, University of Louisville, KY.

- Detected avatar (artificial) faces for avatar recognition using Intel's OpenCV and Microsoft C#
- Recognized avatar faces using Daubechies Wavelets and
  Support Vector Machines (Team work)
- Built an image CAPTCHA prototype with human and avatar faces
  to tell humans from bots using C# and Microsoft SQL Server
- Scripted an automated process of collecting avatar faces from the web using Sikuli (Team work)
- Applied Local Directional Pattern (LDP) towards classifying human (natural) and avatar (artificial) faces

Web Developer (August 2007 – December 2008)

Department of Computer Engineering and Computer Science, University of Louisville, KY.

- Maintained and updated the web pages for the CECS Department
- Migrated the existing web pages to adapt to the
  Content Management System PLONE

XML Integrator (February 2007 – May 2007)

Zeus Learning, Mumbai, India

- Developed E-learning Microsoft Office software for
  Pearson Education
- Coded chapters of the software using XML and tested them

EDUCATION

Doctorate (Ph.D.) in Computer Science and Engineering
(August 2009 – October 2014)

Cyber Security Lab, JB Speed School of Engineering, University of Louisville, KY.

- Relevant Coursework: Biometrics, Computer Vision, Digital Image Processing, Artificial Intelligence, Web mining, Combinatorial Optimization, Simulations

- Research areas: Building image CAPTCHAs, human-artificial face classification, behavioral biometrics (mouse dynamics)
- Cumulative GPA: 3.66

Master of Science (M.S.) in Computer Science (August 2007 - May 2009)

JB Speed School of Engineering, University of Louisville, KY.

- Relevant Coursework: Data Mining, Computer Algorithms, Fundamental of Computer Networks, Introduction to Bioinformatics, Computational Cognitive Science, Databases, Cryptology
  and Computer Graphics
- Cumulative GPA: 3.83

Bachelor of Engineering (B.Eng.) in Computer Engineering (June 2002 - June 2006)

University of Mumbai, India

- Ranked in the top 10 students in a class of 65

ACADEMIC PROJECTS

Ph.D. (Doctoral)

- Building a face recognition system using MATLAB
- Image segmentation to identify and color regions in an image using MATLAB
- Performance comparison of Bubble Sort (on GPU) and Quick Sort (on CPU) using CUDA

Master of Science (M.S.)

- 3D Airplane game using Microsoft XNA Game Studio 3.0

Bachelors (B. Eng.)

- Xpressmail using PHP and MySQL
- Online E-greetings & Gift Store using HTML and Oracle

HONORS

- First place in Computer and Information Sciences at the 2013 Graduate Research Competition Awards, Kentucky Academy of Science Annual Meet 2013, Morehead State University, Morehead, KY

- Second place in Research Poster competition at the Graduate Research Symposium, University of Louisville, Louisville, KY, 2013
- First place in Computer and Information Sciences at the 2010 Graduate Research Competition Awards, Kentucky Academy of Science Annual Meet 2010, Western Kentucky University, Bowling Green, KY
- CECS Master of Science Award for attaining the highest cumulative scholastic standing in the Master of Science program at the JB Speed School of Engineering, University of Louisville, Louisville, KY, 2009

PRESENTATIONS AND TALKS

- “Natural vs Artificial Face Classification using Uniform Local Directional Patterns and Wavelet Uniform Local Directional Patterns”, Computer Vision and Pattern Recognition (CVPR), Columbus, OH, June 23, 2014
- “ Zoo CAPTCHA-Telling Computers and Humans Apart via Animal Image Classification”, Third ASE International Conference on Cyber Security, Stanford University, Stanford, CA, May 30, 2014
- “Image CAPTCHA with an educational purpose” to help kids identify images of animals, birds, reptiles and insects, Louisville Mini Maker Faire, Louisville, KY, September 28, 2013
- “Avatar CAPTCHA-Telling Computers and Humans Apart via Face Classification”, International Conference on Electro/Information Technology, Indianapolis, IN, May 8, 2012
- “Baseline Avatar Face Detection using an Extended Set of Haar-like Features”, The 23rd Midwest Artificial Intelligence and Cognitive Science Conference, University of Cincinnati, Cincinnati, OH, April 21, 2012

PUBLICATIONS

- “Natural vs Artificial Face Classification using Uniform Local Directional Patterns and Wavelet Uniform Local Directional Patterns”, Computer Vision and Pattern Recognition (CVPR), Columbus, OH,
June 23, 2014

- " Zoo CAPTCHA-Telling Computers and Humans Apart via Animal Image Classification", Third ASE International Conference on Cyber Security, Stanford University, Stanford, CA, May 30, 2014
- "Avatar CAPTCHA-Telling Computers and Humans Apart via Face Classification", International Conference on Electro/Information Technology, Indianapolis, IN, May 6-8, 2012
- "Baseline Avatar Face Detection using an Extended Set of Haar-like Features", The 23rd Midwest Artificial Intelligence and Cognitive Science Conference, University of Cincinnati, OH, April 21-22, 2012
- "Automated Collection of High Quality 3D Avatar Images", The 23rd Midwest Artificial Intelligence and Cognitive Science Conference, University of Cincinnati, OH, April 21-22, 2012

EXTRA-CURRICULAR ACTIVITIES

- Information Technology Coordinator (2010-2011): American International Relations Club (AIRC), University of Louisville, Louisville, KY
- Organizing Committee Officer (2010-2011) for "International Banquet", an annual cultural event organized at the University of Louisville and attended by more than 500 people every year
- Graduate Student Ambassador (2011-2012), School of Interdisciplinary and Graduate Studies, University of Louisville, Louisville, KY
- Member of the University of Louisville Racquetball club