Web Bot Detection Using Mouse Movement

Santiago Escuder Folch
I2CAT
Barcelona
santiago.escuder@i2cat.net

Albert Calvo Ibáñez
I2CAT
Barcelona
albert.calvo@i2cat.net

Nil Ortiz Rabella
I2CAT
Barcelona
nil.ortiz@i2cat.net

Josep Escrig Escrig
I2CAT
Barcelona
josep.escrig@i2cat.net

***Abstract*—Non-Legitime traffic in terms of automated internet bot traffic is a long-standing problem causing a huge economic impact and lack of trust in companies and administrations worldwide. For years, Artificial Intelligence and especially Machine Learning have been key players fighting and helping the stakeholder to analyze and detect fraud instances automatically. However, it does not exist a reliable ground truth public dataset to evaluate and compare the proposed methodologies in the literature. In this ongoing study, a public dataset consisting of human and bad-bot web mouse movements extracted from real bot engines is being developed. When finished, it will be uploaded publicly. In addition, this dataset is evaluated using two Machine Learning models. The first is gradient boosting algorithm based on tree classifiers and the second a Recurrent Neural Network. Both models obtain excellent preliminary results.**

***Index Terms*—Bot detection, Machine Learning, Mouse movement, Fraudulent traffic.**

Tipo de contribución: Investigación en desarrollo

I. Introduction

Research in Artificial Intelligence (AI) applied in online web fraud detection is a hot topic of research. Fraud detection and anomaly detection in the cyberdigital world is one of the principal source of economic loses. This loses are quantified in $5.8 billion only in the US targeting a wide umbrella of victims: private companies, public organizations and even critical systems such as hospitals or utilities companies [1]. In this paper, it is reviewed the specific case of web fraud target to identify fraudulent traffic in web sessions. The correct identification is crucial in the marketing domain allowing to quantify the real traffic of a web page. Through this paper it is research how AI could be applied in the domain allowing to detect fraudulent traffic in an efficient fashion. Specifically, in this paper it is used Supervised Classification models to classify legitimate and fraudulent traffic instances. Different kind of features are used to detect web fraud , from connection to behavioural features. In our case, we focus on the problem of fraudulent web traffic created by bad bots that simulate human behaviour to access web pages resources. The behavioural features that are used in this paper is mouse movement. In order to develop Machine Learning models, suitable datasets are needed. The availability of datasets is one of the challenges faced in this project. Most of the web mouse movement datasets are private. Furthermore, these datasets use synthetic data for fraudulent mouse movement and ad-hoc made applications such as login pages to collect legitimate mouse movement. This is due to the cost of extracting real data and the difficulty of data labeling. Another challenge faced is the volumetry of the data. Mouse movement data is obtained in huge amounts. However, an extensive pre-processing needs to be carried for this data to be useful.

The most important contribution of this study will be the creation of a public web mouse movement dataset that contains both fraudulent and legitimate samples. This dataset will have great significance because, as it has been stated previously, most of the datasets are not public. The key aspect of this project's dataset is the legitimate samples are real people browsing freely the internet. This movements are extracted from Bogazici mouse dynamics created by AA Kiliç et. al [2].

The remainder of this paper is structured as follows. *Section 2* includes a literature review, including other state of the art fraud detectors. *Section 3* includes the methodology followed to create the datasets and an explanation of the models used . *Section 4* describes the results of the project including an analysis of the dataset and the results of the preliminary trained models. The paper ends in *Section 5*, which are our main conclusion and discussing the future work.

II. State of the art

In this section it can be found an introduction to the concept of fraudulent traffic and legitimate traffic in *Legitimate and Fraudulent traffic*. A comprehensive review of the different Machine learning techniques found on the literature used to classify legitimate and fraudulent traffic in *Bot detection state of the art*.Finally *HTTPS based features* and *bimoetric features* are explained

A. Legitimate and Fraudulent traffic

Bots can be defined as software created by humans that are used in the Internet and automatically executes tasks (C. Harringer2018) [3]. There is a wide range of tasks such as respond messages, retrieve information, execute commands and use social network [3]. Harringer distinguishes 13 different types of bot, being social bots and chat bots them most common ones. The first bot technology developed was in 1966 a project called ELIZA that was the precursor of the chatbot (A. Godulla et al., 2021)[4].

Nowadays, bots are very present in the Internet taking up to 42.3% of web traffic [5]. Web Traffic types can be classified as:

Fraudulent traffic, also known as bad bot. These bad bots have 21 different uses that are compiled in The Open Web Application Security Project (OWASP) [6]. Some of the most commons threats are Data scraping, denial of service and fraud. In 2022 Imperva Bad Bot Report [5] it is reported that in 2021 27.7% of all the traffic is considered fraudulent traffic. Compared with the same study made in 2014, bad bots have increased by 5% [7]. Bad bots are also classified using their complexity.

- **Simple**: These bots use automated scripts to connect to sites without trying to disguise as a browser.
- **Moderate**: These bots can simulate being a browser when connecting to the site and have the ability to execute JavaScript.
- **Advanced**: These bots copy human behaviour to elude the security. Some human behavioural characteristics are mouse movement, mouse clicks and keybord use. In addition, they use more sophisticated ways to connect to the site such as malware installed in real browsers.
- **Evasive**: These bots are a mix of moderate and advanced bots. Their main characteristics to not be detected by security is change their IP addresses and user agents. In addition, they use what is known as ”low and slow” tactics where they keep a low profile to not be detected by the security. They tend to do a small amount but effective requests in order to not stand out among legitimate connections.

Legitimate traffic. This traffic is made up of human traffic and good bots. Good bots, also known as crawlers, are beneficial for the user and companies. Every day new web pages are created and companies such as Google, Facebook, Bing etc. need to keep track of these newly added web pages. To do so, crawlers index the different web pages or, in other words, crawlers create a map of the internet [8]. This way, the information can be retrieved efficiently and the web pages appear on the search engines. Crawlers when entering a web page collect all the links and recursively continues as it is explained in Chatterjee et al., 2017 [9]. The Imperva bad Bot report attribute to good bots in 2021 the 14.6% of the total web traffic.

B. Bot detection state of the art

In this section it is explained the types of procedures used to classify legitimate traffic from fraudulent traffic found in the literature and the two key aspects considered when reviewing the literature which are **transparency** and **the nature of the data**. Doran D et al,. 2016 summed up the different types of bot detectors in the literature [10]:

- **Syntactical log analysis** It consists on text processing and analyzing server logs that are created by the web traffic. From these logs different parameters such IP addresses or user-agents can be extracted. These two features can be sought in fraudulent IP and user-agent lists. The downside of this technique is that it only detects basic bots. However, the data can be easily extracted.
- **Traffic pattern analysis**. Statistically compare the characteristics and the behaviour of fraudulent traffic in a session against human traffic. One possible comparison is how bots and humans move through the links of a web page. For example, some bots use breath-first or depth-first algorithms when moving through the web page. The downside of this technique is that bots are constantly changing making the patterns obsolete with the need of fast updates.
- **Turing test systems**. A test is added to the web page and the user needs to take it. If the user passes the test it is considered human, if not it is considered bot. The downside of this technique is the cost of implementation and the deterioration the user experience. These tests are thoroughly explained later on this section.
- **Analytical learning techniques.** This technique uses the characteristics of the sessions as features to trains Machine Learning models, being each session a sample. One of the drawbacks is the cost of creating a training dataset. Behaviour data is defined as the interaction of the users with a web page that is being used. Two different types of behavioural features have been found when using analytical learning techinques.
    1) HTTPs and connection based features.
    2) Biometric features.

  This project is focused on creating a public dataset based on mouse movement, which is biometric behavioural feaure. Afterwards Machine Learning models will be trained using this dataset to classify between legitimate and fraudulent traffic.

C. HTTPS based features

HTTP based traffic classification uses HTTP data to classify the traffic. This kind of data is acquired when the user connects to the platform. These data is obtained in a transparent way for the user. It is a key aspect as the user does not have to interact with the page.

In order to exemplify HTTP features, previous definitions needs to be done. First HTTP (Hypertex Transfer Protocol) is a protocol used in the transmission of documents. An HTTP request is an action made by the client to a host in a server to access resources of, for example, a web page. Finally, as it is defined in Suchacka et al.,2021 [11] a session is a sequence of HTTP requests with the same IP address and same user-agent. The time span between two requests should be less than 30 minutes.

Some examples of HTTP features used in the literature are: Total HTTP requests done in a session, total amount of HTTP GET/POST/HEAD requests and session time amongst others in Iliou et al.,2019 [12].

Amongst the literature two types o machine learning have been found. Supervised and unsupervised learning methods have been reviewed.

1) **Supervised Learning**: In Doran et al., 2016 [10] data is extracted in sessions. For each request several variables are extracted. In order to classify the sessions a DTMC (Discrete Time Markov Chain) is used. DTMC are formed by several states, in this case the states correspond to the different requests made. They sustain that bots and humans do not follow the same request pattern.
   In Suchacka et al., 2021 [11] the data is also gathered using the concept of sessions. In addition, new variables extracted from HTTP requests are extracted. However, sessions are classified using MLP (MultiLayer Perceptron). The features extracted from each request are used as input of the MLP and in order to exploit the relation between the different requests in a session a probabilistic sequential analysis is used.
   Both aforementioned articles are ”on the fly”, which means that they can be used in real time and there is no need to process the whole session in order to classify it.

In both articles the data is gathered in an e-commerce web, thus making the data real. In order to label the data external information is used.

2) **Unsupervised Learning**: Labeling the data is expensive and time consuming. There are approaches such as Rovetta et al,. 2020 [13] that uses unsupervised learning. This way labeling the data is avoided making the process less expensive and more efficient. Similar results are obtained as supervised models are but it cannot be used ”on the fly”. the data is also collected from a real e-commerce web.

D. *Biometric based features*

As it is defined in V.Matyas et al,. 2003 ”Biometrics are automated methods of authentication based on measurable human physiological or behavioral characteristics” [14]. Physiological biometrics are measures extracted directly from the human body. Some examples are fingerprint scan, iris scan and facial scan. On the other hand, behavioural biometrics are measurements based on human actions such as gaig, voice and mouse dynamics.

The use of biometrics is widely spread for person identification and person authentication. For example in border controls or even to unlock your phone. Mouse dynamics can also be used as intruder detector, comparing the mouse features of the owner of the computer with the person using it. In this project mouse dynamics are used to detect weather the user in a web page is a bot or not. Below, the articles reviewed for this projects are explained.

In Ang Wei et al,. 2019 a database is created ad-hoc for the project. It consists in a login landing page that records the mouse movements. This login page has three different fields to fill. Human samples are collected by volunteers that log in in this web page. Bot data is collected using 4 types of bots that also log in in this page. These types of bots follow 4 different types of movements: linear, curve, polyine and semi-straigh line. The mouse features are collected in sequences that are made of consecutive mouse events. These sequences have the following form: $[(x_1, y_1, t_1), ..., (x_n, y_n, t_n)]$, being x and y the pixel coordinate and t the timestamps of the mouse event. The data is further processed creating an image of the mouse sequence. Finally a CNN based model is used to classify the sequences [15]. In H Niu et al,. 2021 the same database is used. However, instead of creating an image out of the movement, the velocity is calculated creating a sequence such as $[(dx_1, dy_1, dx_1/dt, dy_1/dt), ..., (dx_n, dy_n, dx_n/dt_n, dy_n/dt_n)]$. These sequences are then passed through LSTM based models [16]. Out of both articles reviewed, the later obtains better results and is more efficient as the data processing has less cost. Finally, A.Acien et al,. 2021 [17] created a new database based on Shen et al., 2014 [18]. The legitimate samples consists on volunteers repeating a image CAPTCHA type test. When doing this test the mouse movement is captured. The fraudulent samples are created as in Ang Wei and also using GANs (Generative Adversial Netwroks) to create synthetic samples.

Amongst the reviewed mouse movement articles all the data is collected in a non-transparent way. Users have to solve image CAPTCHAs [17] or login landing [15][16] pages making the data collection non-transparent

To conclude, after reviewing the literature, HTTP features seem to be more convenient when detecting bots as they are always transparent. In addition, real fraudulent data can be extracted as it can be labeled by using IP and user agents blacklists, whereas fraudulent mouse movement data in the literature is created by using bots in ad-hoc landing pages.

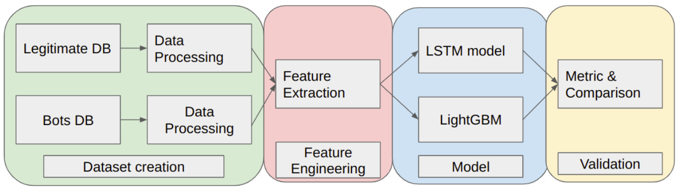

Figure 1. Methodology framework.

III. Mehtodology

In this section is it explained the methodology used throughout the project. In Fig.1 it can be found the diagram of the complete methodology framework. The main parts are: *Dataset Creation*, *Features Engineering*, *Models* and *Validation*.

A. *Dataset creation*

One of the key aspect of this project is to create a free mouse movement public dataset that contains both human and bad-bot samples. Reviewed datasets are either private or the mouse movement is not free. This project’s dataset is created in two steps:

- **Human mouse movement**: This part of the dataset was created by using Bogazici mouse dynamics created by AA Kiliç et. al[2]. It is a public dataset that contains real free human mouse movement.
- **Bad-bot mouse movement**: This part of the dataset was created by using bots. This bots were adapted from two different GitHub repositories: bezmouse by vicentbavitz [19] and self-driving-desktop by hofstadter-io [20]

1) Human Dataset creation: Bogazici mouse dynamics dataset contains free mouse movements of 43 users recorded during 10 days. This database was meant to be used to train models for intrusion detection. It labels each sample depending on what the user is doing. For example, if the user is coding in an IDE it will be labeled as *Development*; if the user is browsing the internet it is labeled as *Browsing*. In addition, this database also contains the use of the muse buttons and their actions. This database has been chosen because it has been developed in a laboratory environment, created with real people moving the mouse freely.

In this project only the *Browsing* category is considered as this project is focused in web usage of the mouse. *Browsing* samples take up to 51% of the whole database. In addition, only the mouse movement data is considered whilst the buttons data is not. One inconvenient found in this database was that the size of the monitors and the amount of monitors used by the users is not indicated. The size of monitors from users with a single monitor can be discovered by looking a the maximum and minimum coordinate values. However, if more than one monitor was used negative values appeared and the sizes could not found. To solve this problem only data coming from one monitor and a size of $1920x1080$ is used.

In the public database, each sample is taken each $1ms$. It is a very high sampling rate that leads to a very high amount of data. In order to reduce the amount of data a resampling is made each $100ms$. Finally, the article by Ang Wei [15] the length of each sequences is set to 60 samples.

Another problem faced is that human samples tend to be empty, this means that most sequences did not have any movement. To solve this problem first we have to define the position differentials $d_x[n]$ and $d_y[n]$ *Ec.(1)*. Only sequences with an AoM(Anount of Movement) *Ec.(2)* higher than 50% are used.

$$\begin{aligned} d_x[n] &= x[n] - x[n-1] \\ d_y[n] &= y[n] - y[n-1] \end{aligned} \tag{1}$$

$$\%samples \begin{cases} d_x[n] \neq 0 \\ d_y[n] \neq 0 \end{cases} \quad \forall 0 < n < N \tag{2}$$

2) Bad-bot dataset creation: The fraudulent part of the dataset is created adapting github bot repositories. The repositories used are

- Self-driving-desktop by hofstadter-io [20]. This repository follows text input orders to create mouse movement. It was adapted so it returns sequences of samples of the desired length. This repository can create sequences that follow different types of movements such as linear and quadratic movments.
- Bezmouse by vicentbavitz [19]. This repository was barely adapted as it already returns a sequence of samples following Beziere curves. Beziere curves are parametric curves that given a set of discrete control points it creates a continuous curve.

In Wei et al[15] bot mouse movements can be of 4 different types: that are Straight lines, semi-straight lines, regular curves and irregular curves. In our case, to mimic we use 4 different type of movements:

- **Linear**. It creates an straight line between two points. The bot moves with constant velocity.
- **Quadratic**. It follows a quadratic curve between two points. Two types are used: Quadratic with higher acceleration at the beginning and quadratic with higher acceleration at the end.
- **Exponential**. It follows an exponential curve between two points. Two types are used: Exponential with higher acceleration at the begining and Exponential with higher acceleration at the end.
- **Bezier**. It creates a Beziere curve between two points given a set of control points. The bot moves with constant velocity.

In order to create this sequences the initial and the final coordinate are created randomly following an uniform distribution. Where x coordinate can have values $0 < x < 1919$ and the y coordinate can have values between $0 < y < 1019$. The length of the sequences is the same as the legitimate ones, which is 60. The AoM of the fraudulent sequences is always 100%, this means that there is always movement.

B. Feature Engineering

In this section the different features used are explained. Two different sets of features are created. Sequence features are used to train Deep Learning models, specifically a LSTM (Long Short-Term memory). Scalar features are meant to train Decission Tree based Mahine Learning models. Decission Tree features are scalar value whilst Deep learning features are vectors of fixed length.

1) Sequence features: Sequence features are extracted for each sequences and they are similar to [15]. A sequence consists on 60 pair of coordinates such as $[(x_1, y_1), ..., (x_{60}, y_{60})$. In order to train the LSTM we calculate the differential of movement for each sample creating $[(0,0), (dx_2, dy_2), ..., (dx_{60}, dy_{60}]$. As it can be observed the first differential is (0,0) because it cannot be calculated as there is no previous sample. Thus making the first differential not usable. This creates a sequence such as $[(dx_2, dy_2,), ..., (dx_{60}, dy_{60})]$ with length 59. In Wei et al [15], this features are a bit different $[(dx_1, dy_1, dx_1/dt, dy_1/dt), ..., (dx_n, dy_n, dx_n/dt_n, dy_n/dt_n)]$. As it can be observed each sample contains both differentials and the velocities. In our case calculating the features d_x/d_t does not make sense as our time differential dt is constant (100ms). This would only add a constant multiplicative factor to the differentials.

2) Scalar features: In this case 12 different features are calculated per each sequence of longitud N.

- $V_x mean$ and $V_y mean$. They are defined as the mean of the differentials in each direction. *Ec.(3)*

$$V_j mean = \frac{\sum_{i=1}^{N}(d_j[i])}{N} \quad j = [x, y] \tag{3}$$

- $Vmean$ It is defined as the mean of module of d_x and d_y *Ec.(4)*.

$$\begin{aligned} Vmod[i] &= \sqrt{d_x[i]^2 + d_y[i]^2} \\ Vmean &= \frac{\sum_{i=1}^{N} Vmod[i]}{N} \end{aligned} \tag{4}$$

- $Vmax$, $V_x max$ and $V_y max$ They are defined as the maximum value of each velocity.
- $A_x mean$ and $A_y mean$. They are defined as the mean differentials of each velocity *Ec.(5)*.

$$A_j mean = \frac{\sum_{i=1}^{N}(d_j[i] - d_j[i-1])}{N} \quad j = [x, y] \tag{5}$$

- $Amean$ It is defined as the mean of module of the differentials of d_x and d_y *Ec.(6)*.

$$\begin{aligned} Amod[i] &= \sqrt{(d_x[i] - d_x[i-1])^2 + (d_y[i] - d_y[i-1])^2} \\ Amean &= \frac{\sum_{i=1}^{N} Amod[i]}{N} \end{aligned} \tag{6}$$

- $Amax$, $A_x max$ and $A_y max$ They are defined as the maximum value of each acceleration.

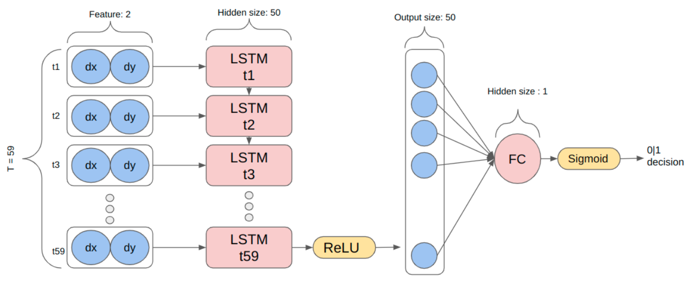

Figure 2. LSTM model.

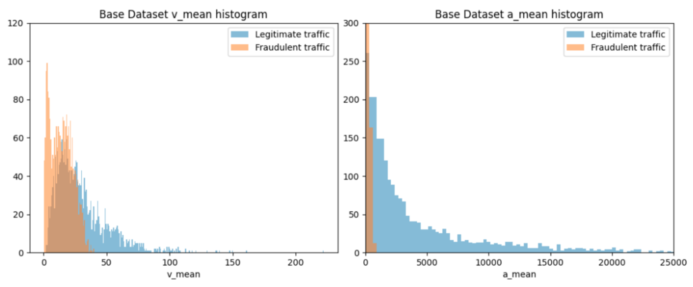

Figure 3. Velocity and acceleration distributions.

C. Models

1) LightGBM: GBDT (Gradient Boosting Decision Tree) is a machine learning algorithm with some successful implementations such as XGBOOST by T Chen et al [21]. GBDT are very efficient, accurate and with a high interpretability. However, GBDT algorithms tend to be inefficient when the feature dimension and the data size is high. LightGBM by Guolin Ke et al [22] optimizes the GBDT algorithm and solves the problems mentioned obtaining in most cases the same accuracy. In our case, there is not much difference between using XGBOOST and LightGBM as the feature dimension and dataset size are small.

In order to train the LightGBM model the dataset is divided in train and test dataset. The train dataset is the 75% of the dataset whilst the test dataset is the 15% remaining data. In addition the training is carried using a cross-validation technique called Stratified KFold . Stratified KFold divides the train dataset in Kfolds that preserves the percentage of samples for each class. In this case the number of folds K is 5. In order to search the best hyperparameters a Grid Search is used. Grid Search creates all the possible hyperparameters combinations given a set of hyperparameters.

2) LSTM: Long short-term memory (LSTM) is an artificial neural network with feedback connections. This feedback connections allows to exploit the time characteristics of the inputs [23]. Mouse movement is based on time series which makes it suitable to use LSTM in this project. Time based characteristics cannot be exploited using normal feed forward networks. In Fig. 2 it is shown the architecture of the model used. The main characteristics are:

- LSTM layer: N is the number of LSTM layers with hidden size 50 is the Number of the hidden size.
- Rectified Linear Unit (ReLu) activation Layer. It is a widely used activation layer in the training of Deep neual Netwroks.
- Forward fully connected Layer (FC): It has a hidden size of. It is used to reduce the dimesionality to 1 in order to take a decision.
- Sigmoid Activation Layer. This layer is used in used in binary classification.

D. Validation

The metric used in this project are TPR *Ec.(7)* (True positive Rate) and TNR *Ec.(8)* (True negative rate) and Accuracy *Ec.(9)*. Both models compute the same metrics and are used to compare the performance between models. In addition, TPR and TNR are also used in [15] and are also used to compare our project with theirs.

- **TPR** is defined as the proportion of correct predictions in predictions of positive class. In our case, positive class is fraudulent traffic *Ec.(7)*.

$$TPR = \frac{TP}{TP + FN} \tag{7}$$

- **TNR** is defined as the proportion of correct predictions in predictions of negative class. In our case Negative classes are legitimate traffic *Ec.(8)*.

$$TNR = \frac{TN}{TN + FP} \tag{8}$$

- **Accuracy** is defined as the proportion correct predictions and the total amount of predictions . In our case it is used to validate the LSTM model and compare the LightGBM and LSTM model *Ec.(9)*.

$$Accuracy = \frac{TP + TN}{TN + TP + FP + FN} \tag{9}$$

IV. Partial Results

In this section the results are explained. This section is divided in two parts.

- **Dataset analysis**: It is explained the characteristics of the dataset in Table I there is a summary of the characteristics of the dataset.
- **Model results and comparison**: It is explained and compared the results obtained with the LSTM and LightGBM models.

A. Dataset analysis

In total there are 2385 legitimate labeled sequences. The fraudulent sequences have been created using bots extracted from GitHub repositories. These bots follow 4 different types of movements which are: Linear, quadratic, exponential and bezier. For each type of movement it is created 397 samples which makes a total of 2382 fraudulent sequences. In total the dataset has 4767 sequences with an almost 50-50 balance between legitimate and fraudulent labels. As it can be observed in Table.I, the mean velocity is lower in the Fraudulent sequences. This is because the initial and end coordiantes of the movements are created randomly. This can lead to create short and slow mouse movements. The mean acceleration is way bigger in the Legitimate sequences than in the Fraudulent sequences. This is because Linear and Bezier movements do not have acceleration as they have constant velocity. We can further analyse the mean velocity and mean acceleration by looking at the histograms in Fig. **??**.

Table I
DATASET CHARACTERISTICS

Sequences	**# Sequences**	$\overline{AoM}$	$\overline{Vmean}$	$\overline{Amean}$	$\overline{Vmax}$	$\overline{Amax}$
Human	2385	66%	29.64	4813	232.7	67793
Bad-Bot	2382	100%	14.13	68.40	48.79	751.2
Dataset	4767					

B. Model results and comparison

In this paper two types of models have been used LSTM and LightGBM. In Table. II it can be found the results. The first column indicates the name of the model. The second column indicates the features used to train and test the model used in the training. When Features is Scalar velocities, only the features related with the velocity are used. When Features is Scalar All, both acceleration and velocity features are used. Finally, the third and fifth columns are the results

We start comparing LightGBM (1) and LightGBM (2). Both models obtain very high results on the Base test being LightGBM (2) the one with better results. This is because (1) only uses velocity related features that are not as discriminant as the acceleration. The LSTM (3) model obtains very similar results with LightGBM (2).

With the results obtained at this moment LSTM based models and LightGBM models obtain similar results. What it was expected is LSTM to have better results because the temporal relations can be exploited. However, the results are very high in both cases due to the simplicity of the fraudulent dataset.

Table II
RESULTS

Model	**Features**	**Base test**
(1) LightGBM	Scalar velocities	TPR: 0.980 TNR: 0.953 Acc: 0.966
(2) LightGBM	Scalar all	**TPR:0.997** **TNR: 1.000** **Acc: 0.998**
(3) LSTM	Sequences	**TPR: 1.000** **TNR: 0.981** **Acc: 0.990**

V. CONCLUSIONS AND FUTURE WORK

This project has achieved the creation of an open access free mouse movement dataset. The dataset can be used as a reference benchmark for novel legitime/non-legitime detection algorithms. However, this dataset can be easily improved by adding accelerations to the fraudulent data and controlling the AoM in each sequence. Other ways of creating fraudulent data can be used like Generative Adversary Networks (GANs). GANs objective is to model high-dimensional distribution of data. An application of GANs is data synthesis [24]. In our case, the synthesis of new mouse movements.

REFERENCES

[1] Unit21, “Fraud detection prevention for financial organizations: Ultimate guide,” https://www.unit21.ai/blog/fraud-detection-prevention-for-financial-organizations.

[2] A. A. Kılıç, M. Yıldırım, and E. Anarım, “Bogazici mouse dynamics dataset,” *Data in Brief*, vol. 36, p. 107094, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352340921003784

[3] C. Harringer, “„good bot, bad bot“?: Zur problematik von bot-ontologien,” *Information - Wissenschaft Praxis*, vol. 69, no. 5-6, pp. 257–264, 2018. [Online]. Available: https://doi.org/10.1515/iwp-2018-0040

[4] A. Godulla, M. Bauer, J. Dietlmeier, A. Lück, M. Matzen, and F. Vaaßen, “Good bot vs. bad bot: Opportunities and consequences of using automated software in corporate communications,” 2021.

[5] Imperva, “2022 imperva bad bot report evasive bots drive online fraud,” https://www.imperva.com/resources/reports/2022-Imperva-Bad-Bot-Report.pdf.

[6] OWASP, “Owasp automated threats to web applications,” https://owasp.org/www-project-automated-threats-to-web-applications/.

[7] I. 2014, “2014 bot traffic report: Just the droids you were looking for,” https://www.imperva.com/blog/bot-traffic-report-2014.

[8] M. Abu Kausar, V. Dhaka, and S. Singh, “Web crawler: A review,” *International Journal of Computer Applications*, vol. 63, pp. 31–36, 02 2013.

[9] S. Chatterjee, A. Nath, and M. S. Student, “Auto-explore the web-web crawler article in international journal of innovative research in computer and communication engineering,” 2017. [Online]. Available: www.ijircce.com

[10] D. Doran and S. S. Gokhale, “An integrated method for real time and offline web robot detection,” *Expert systems*, vol. 33, no. 6, pp. 592–606, 2016.

[11] G. Suchacka, A. Cabri, S. Rovetta, and F. Masulli, “Efficient on-the-fly web bot detection,” *Knowledge-Based Systems*, vol. 223, 7 2021.

[12] C. Iliou, T. Kostoulas, T. Tsikrika, V. Katos, S. Vrochidis, and Y. Kompatsiaris, “Towards a framework for detecting advanced web bots,” in *Proceedings of the 14th International Conference on Availability, Reliability and Security*, 2019, pp. 1–10.

[13] S. Rovetta, G. Suchacka, and F. Masulli, “Bot recognition in a web store: An approach based on unsupervised learning,” *Journal of Network and Computer Applications*, vol. 157, 5 2020.

[14] V. Matyas and Z. Riha, “Toward reliable user authentication through biometrics,” *IEEE security privacy*, vol. 1, no. 3, pp. 45–49, 2003.

[15] A. Wei, Y. Zhao, and Z. Cai, “A deep learning approach to web bot detection using mouse behavioral biometrics,” in *Chinese Conference on Biometric Recognition*. Springer, 2019, pp. 388–395.

[16] H. Niu, J. Chen, Z. Zhang, and Z. Cai, “Mouse dynamics based bot detection using sequence learning,” in *Chinese Conference on Biometric Recognition*. Springer, 2021, pp. 49–56.

[17] A. Acien, A. Morales, J. Fierrez, and R. Vera-Rodriguez, “Becaptcha-mouse: Synthetic mouse trajectories and improved bot detection,” *Pattern Recognition*, vol. 127, p. 108643, 2022.

[18] C. Shen, Z. Cai, X. Guan, and R. Maxion, “Performance evaluation of anomaly-detection algorithms for mouse dynamics,” *Computers & security*, vol. 45, pp. 156–171, 2014.

[19] https://github.com/vincentbavitz/bezmouse.

[20] https://github.com/hofstadter-io/self-driving-desktop.

[21] T. Chen, T. He, M. Benesty, V. Khotilovich, Y. Tang, H. Cho, K. Chen *et al.*, “Xgboost: extreme gradient boosting,” *R package version 0.4-2*, vol. 1, no. 4, pp. 1–4, 2015.

[22] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, “Lightgbm: A highly efficient gradient boosting decision tree,” *Advances in neural information processing systems*, vol. 30, 2017.

[23] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[24] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. Bharath, “Generative adversarial networks: An overview,” *IEEE Signal Processing Magazine*, vol. 35, 10 2017.