# A Deep Learning Approach to Web Bot Detection Using Mouse Behavioral Biometrics

Ang Wei, Yuxuan Zhao, and Zhongmin Cai(✉)

MOE KLINNS Lab, Xi'an Jiaotong University, Xi'an 710049, China
zmcai@sei.xjtu.edu.cn

**Abstract.** Web bots are automated scripts that perform online tasks like human. Abuse of bot technology poses various threats to the security of websites. Recently, mouse dynamics has been applied to bot detection by analyzing whether recorded mouse operations are consistent with human operational patterns. In this paper, we introduce a deep neural network approach to bot detection. We propose a new representation method for mouse movement data, which converts every mouse movement into an image containing its spatial and kinematic information. This representation method makes it possible to utilize CNN models to automate feature learning from mouse movement data. Experimental results demonstrate that our method is able to detect 96.2% of bots with statistical attack ability while traditional detection methods using hand-crafted features or RNN can only detect less than 30% of them.

**Keywords:** Bot detection · Mouse movement · CNN

## 1 Introduction

Web bots are automated scripts that run tasks without human intervention on the Internet. Malicious bots pose a serious threat to Internet users. They influence the results of online polls, spread spams, and commit various types of online crimes and frauds. According to Bot Traffic Report 2016 [1], malicious bots comprise 28.9% of online traffic and have caused considerable loss for the users and owners of websites.

Recently, mouse dynamics, a behavioral biometric, has been investigated for bot detection [2,3]. The basic idea is to analyze whether the mouse operation data is consistent with human operational patterns. Although web bots can generate mouse events, it's difficult for bots to perform mouse operations in a human manner. This makes it possible to differentiate bots from human users according to mouse dynamics.

Previous works relied on hand-crafted features and feed the features into a shallow machine learning model. In this paper, we introduce deep learning method to detect web bots using mouse dynamics. To do this, We tackle the


Z. Sun et al. (Eds.): CCBR 2019, LNCS 11818, pp. 388–395, 2019.
https://doi.org/10.1007/978-3-030-31456-9_43

difficulty of converting mouse movement sequences into suitable inputs for deep learning models by proposing a new representation method of mouse movements data. We transform every mouse movement sequence into an image containing its spatial and kinematic information. The representation makes it possible to apply CNN models to web bot detection using mouse dynamics. We build an experimental website and collect mouse movement data from both human subjects and web bots. The experimental results demonstrate that our method outperforms both traditional classifiers using hand-crafted features and RNN based approach in the detection of web bots with statistical attack [4,5] ability. The major contributions of this paper are listed as follows:

(1) We propose to employ deep neural networks to solve the problem of web bot detection using mouse dynamics.
(2) We introduce a new representation method for mouse movement data. It makes it possible to automate feature learning from mouse movement data using deep learning models.
(3) We establish a dataset of web accessing behaviors for both human and web bots by collecting their mouse operational data from an experimental website.
(4) Experimental results demonstrate that our method achieves an accuracy of 96.2% in detecting web bots with statistical attack ability while the accuracies of conventional methods are less than 30%.

## 2 Related Work

Mouse dynamics is a useful biometric for authentication. In recent years, it has also been applied in web bot detection. Gianvecchio et al. [6] proposed a non-interactive approach based on human observational proofs (HOPs) for continuous game bot detection. They monitored mouse and keystroke actions when users were playing an online game. Seven action metric features were used to train a cascade-correlation neural network for distinguishing bots from humans. In [3], the author used mouse dynamics to strengthen image CAPTCHA based bot detection. He extracted 20 features to create feature vectors for each session's data. In [2], the author employed mouse and keystroke dynamics to distinguish between human and bot in blog service Scenario. Eight features were defined to characterize mouse and keystroke actions. A classifier based on C4.5 algorithm was trained using the features of raw data collected from blog site.

Previous works made important progresses in behavioral based web bot detection. However, the widely-used features are usually coarse-grained features such as statistics of mouse movement sequences (e.g., mean and variance of velocity, acceleration, step size, angle). There is no feature proposed to accurately characterize the shape information of mouse trajectory. As a consequence, when facing more advanced human mimicry web bots with statistical attack abilities [7], existing detectors have a large chance to be deceived.

Motivated by the above analysis, in this work, we introduce a deep learning approach to web bot detection and propose an image representation of mouse

movement data for deep CNN models. A deep learning approach can address the discussed issues and automate the complex feature engineering process.

## 3 Overview of Our Approach

The framework of our approach is shown in Fig. 1, which depicts the training as well as the online testing process of web bot detection. In online testing, the decision that whether a user is a bot or human depends on classification results of most mouse movement sequences extracted from his operation data. If a majority of mouse movement sequences are classified as bot-generated, the user will be classified as a bot. Thus in the rest of paper, we will focus on the analysis of individual mouse movement sequence and discuss the situation of single mouse movement sequence based classification.

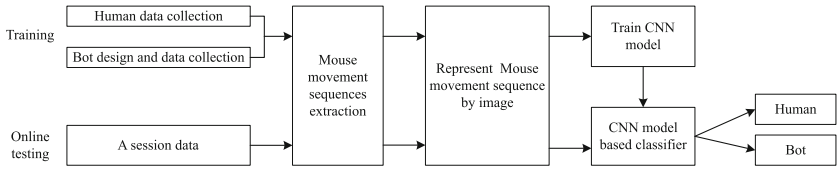



**Fig. 1.** Framework of our approach

To avoid confusion in the subsequent discussion, we at first introduce some notations that will be used throughout the paper.

1. *A mouse movement sequence* is the records of a series of consecutive mouse move events. It has the form $\{(x_1, y_1, t_1), \ldots, (x_i, y_i, t_i), \ldots, (x_n, y_n, t_n)\}$, where $(x_i, y_i, t_i)$ denotes the coordinates and timestamp of a mouse move event.
2. *A Mouse trajectory* is a time-space curve corresponding to a mouse movement sequence.
3. *Step size* denotes the distance between two consecutive points.
4. *Event interval* denotes the time difference between two consecutive points.

### 3.1 Representation of Mouse Movements

The first step in processing mouse behavioral data is to separate the continuous sequence of mouse events into sequence segments corresponding to individual mouse movements. This is performed based on the observation that the event intervals between consecutive points in a mouse movement do not exceed 100 ms. In our study, we discard mouse movements having less than 15 events, which contain not enough information for bot detection. Then, we propose a two-step method to transform a mouse movement sequence into an image that can be an input for a CNN model.

**Map Spatial Information into Image.** One step is to map the spatial information such as shape and distances between consecutive events into an image. Firstly, we determine the size of the input image. We denote the max_range of a mouse movement as the maximal difference between x coordinates or y coordinates of two arbitrary points on its trajectory. Figure 2(a) shows most max_ranges of human mouse movements are smaller than 600 pixels. So we set the size of image to be $600 * 600$. Secondly, we translate mouse movement sequence to the center of image. Thirdly, we zoom in or out mouse movement sequences according to the ratio of its max range and the image size (600 pixels). This transforms all different mouse movement sequences into similar sizes. At last, we represent every point in mouse movement sequence by a dot in the image according to its x and y coordinates. In order not to make an input image too sparse, the radius of dots in an image is set to 4 pixels. Although there are translation and zooming operation in the conversion process, we don't change the overall shape of mouse movement sequence.

**Map Kinematic Information into Image.** The distinguishing characteristics of a mouse movement for web bot detection also contain kinematic information. This information is reflected in its event interval series and step size series, which are denoted as $(\Delta t_1, \ldots, \Delta t_{n-1})$ and $(\Delta d_1, \ldots, \Delta d_{n-1})$ respectively. We employ the color information, BGR (Blue, Green and Red), of a dot for a mouse trajectory point in the input image to embed the kinematic information. We use $\Delta d_i$ and $\Delta t_i$ to set the G and R color values of the corresponding dot. The mapping function from step size to G color value is denoted as g(x) and the mapping function from event interval to R color value is denoted as f(x). g(x) and f(x) map $\Delta d_i$ and $\Delta t_i$ into an integer ranging from 0 to 255. Figure 2(b) is the converted input image of a human mouse movement sequence.

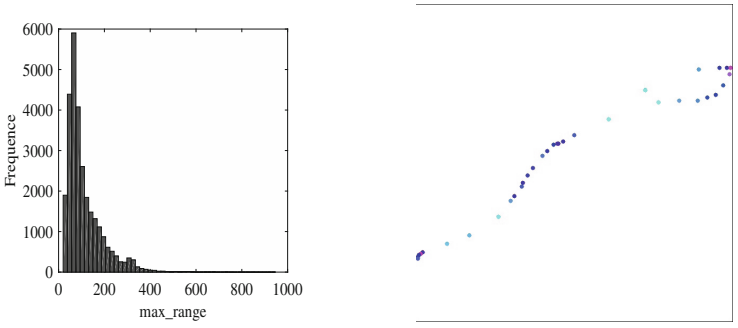



**Fig. 2.** (a) Max_range distribution of human mouse movement sequences. (b) A conversion results of human mouse movement sequences

### 3.2 Deep Classification Model

We choose ResNet [8], a classical CNN model, as our bot detection classifier. Specificlly, we use the ResNet50 architecture from PyTorch. Data augmentation technique is employed by randomly rotating the image by 0, 90, 180, 270°, which can increase the variety of training samples and prevent overfitting. Besides, we load a pre-trained model from ImageNet to initialize the weights instead of training our model from random initializations.

## 4 Mouse Movement Data Collection

### 4.1 Data Collection Scenario

We use user registration process as our web bot detection scenario and build a registration website to collect mouse movement data. The website is a Java web application that consists of a front-end webpage and a back-end service program. There are three input fields, including username, password and password confirmation, and a submission button, on the front-end webpage. An event logger implemented by JavaScript is embedded in the front-end webpage, which silently collects a user's mouse operations when she accesses the website.

### 4.2 Human and Web Data Collection

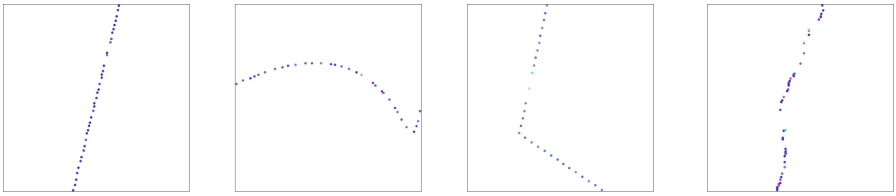
There are 6 rounds of data collection during 2 months. Every subject is asked to finish the registration task ten times in each round. All subjects are told to operate mouse in a natural way according to their habits. Totally, 50 subjects participated in our data collection and we obtained over 3000 sessions of human data.

We use 4 kinds of bot scripts to generate the experimental data from web bots. As shown in Fig. 3, the scripts will generate 4 kinds of mouse trajectories including: straight-line, curve, polyline and semi-straight line, respectively. For semi-straight line bot, we add random disturbance to the coordinates of points on a straight-line mouse trajectory. Trajectories of curve bot are 2 and 3 order Bezier curves. To generate diversified bot activities, we employ three ways to generate step size series and event interval series for bot mouse movement sequence: (1) constant values, (2) uniformly distributed values and (3) values with a Gaussian distribution. Parameter settings are configured according to the norms of human activities. In this way, we generate 10000 mouse movement sequences with 2000 for straight-line bot, 3000 for curve bot, 3000 for semi-straight line bot and 2000 for polyline bot respectively.

### 4.3 Statistical Attack Dataset

The basic idea of statistical attack is to estimate the probability density functions of features from a group of people and then use the most probable feature values to generate the forgery [9]. According to the method shown in [4,5,9], we

generate bot samples with statistical attack ability. Kernel density estimation is used to calculate the probability density estimate of step size series and event interval series of human mouse movement sequences respectively. In the process of synthesizing bot mouse movement sequences, the step size series and event interval series are generated according to the corresponding probability density estimate. This can guarantee that the step size series and event interval series of bots have similar distribution of human's. In this way, we generate 4000 mouse movement sequences for statistical attack dataset, with 1000 for each kind of bot.



(a) straight-line (b) curve (c) polyline (d) semi-straight line

**Fig. 3.** Examples of 4 kinds of bot mouse trajectories

## 5 Experiment

### 5.1 Baselines for Comparison

The first baseline is the Gradient Boosting Decision Tree (GBDT) classifier using nine hand-crafted features. The nine features are defined in related work [2] and [3]. These features are movement efficiency, the average and the standard deviation of speed, acceleration, step size, and movement angle. Movement angle means supplementary angle of the angle that between three consecutive points and movement efficiency is displacement over distance.

The second baseline is an RNN model based method. Mouse movement sequence element for one time step is 3-dimensional vector $(x, y, t)$. We convert element for one time step to $(\frac{dx}{dt}, \frac{dy}{dt})$ according to [10] and pad the sequence to length 60 as the input for RNN model.

### 5.2 Dataset Preparation

We extract over 10000 mouse movement sequences from human mouse movement data. The number of mouse movement sequences of bots is 10000. They are randomly partitioned into 5 complementary subsets. In every run of experiment, we use four of the five subsets for training. The validation set consists of half of the remaining subset and the testing set consists of the other half. We also use the statistical attack dataset discribed in Sect. 4.3 for further test. To remove the interference of the authentication effect, mouse movements from the same human subject are either used for training or testing, but not for both.

### 5.3 Experimental Results

In the testing phase, we choose true positive rate (TPR) and true negative rate (TNR) as evaluate metrics, which are used in related bot detection work [2]. TPR is the ratio of the number of human samples which are correctly classified to the number of all the human samples. TNR is the ratio of the number of bot samples which are correctly classified to the number of all the bot samples.

**Comparison of Baselines Against Our Proposed Method.** This experiment concerns the comparison of two baselines against our proposed method in terms of performance on testing set and the statistical attack dataset. Table 1 shows our method and GDBT can achieve more than 99% on TPR and TNR. This indicates both approaches are effective when coping with common bot samples.

**Table 1.** Comparison on test set

| Model | Avg TPR | Avg TNR |
|---|---|---|
| GDBT | 0.9934 | 0.9925 |
| RNN | 0.934 | 0.974 |
| Our method | 0.9981 | 0.9984 |

**Table 2.** Comparison on statistical attack dataset

| | Avg TNR | | | | |
|---|---|---|---|---|---|
| | Straight-line | Curve | Polyline | Semi-straight line | Overall |
| GDBT | 0.186 | 0.503 | 0.210 | 0.063 | 0.240 |
| RNN | 0.205 | 0.427 | 0.368 | 0.172 | 0.293 |
| Our method | 0.936 | 0.999 | 0.989 | 0.924 | 0.962 |

Further more, we compare three methods on statistical attack dataset. Table 2 shows that our method can still detect 96.2% bot samples while two baselines only detect 24.0% and 29.3% bot samples respectively. As discussed in Sect. 4.3, for statistical attack dataset, some features of bot samples accord with distribution of human samples. Thus, methods using hand-craft features fail to detect these bot samples. In addition, it's hard for RNN model to learn the spatial information contained in mouse movement data, so the performance of RNN model decreases a lot. Our representation method directly visualize the difference of the distinguishing information containing in a mouse movement sequence. This makes it easier for a CNN model to learn the difference between human and bot samples, and thus has a better performance against statistical attack dataset.

## 6 Conclusion

In this paper, we propose a new method to represent mouse movement data by images and apply CNN model to web bot detection using mouse dynamics. This method makes it possible to automate feature learning from mouse movement data and improve the detection performance of more advanced bots. The experimental results demonstrate our method is effective in detecting bots with statistical attack ability.

**Acknowledgments.** This work is supported by NSFC (Grant No. 61772415).

## References

1. Zelfman, I.: Bot traffic report 2016. Imperva Incapsula Blog (2017)
2. Chu, Z., Gianvecchio, S., Koehl, A., Wang, H., Jajodia, S.: Blog or block: Detecting blog bots through behavioral biometrics. Comput. Netw. **57**(3), 634–646 (2013)
3. D'Souza, D.F.: Avatar captcha: telling computers and humans apart via face classification and mouse dynamics. Electronic theses and dissertations. paper 1715 (2014)
4. Serwadda, A., Phoha, V.V.: Examining a large keystroke biometrics dataset for statistical-attack openings. ACM Trans. Inf. Syst. Secur. (TISSEC) **16**(2), 8 (2013)
5. Stanciu, V.D., Spolaor, R., Conti, M., Giuffrida, C.: On the effectiveness of sensor-enhanced keystroke dynamics against statistical attacks. In: Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy, pp. 105–112. ACM (2016)
6. Gianvecchio, S., Wu, Z., Xie, M., Wang, H.: Battle of botcraft: Fighting bots in online games with human observational proofs. In: Proceedings of the 16th ACM Conference on Computer and Communications Security, pp. 256–268. ACM (2009)
7. Jin, J., Offutt, J., Zheng, N., Mao, F., Koehl, A., Wang, H.: Evasive bots masquerading as human beings on the web. In: 2013 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), pp. 1–12. IEEE (2013)
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
9. Song, Y., Cai, Z., Zhang, Z.L.: Multi-touch authentication using hand geometry and behavioral information. In: 2017 IEEE Symposium on Security and Privacy (SP), pp. 357–372. IEEE (2017)
10. Chong, P., Tan, Y.X.M., Guarnizo, J., Elovici, Y., Binder, A.: Mouse authentication without the temporal aspect–what does a 2d-cnn learn? In: 2018 IEEE Security and Privacy Workshops (SPW), pp. 15–21. IEEE (2018)