Proceedings of the 2nd International Conference on Inventive Communication and Computational Technologies (ICICCT 2018)
IEEE Xplore Compliant - Part Number: CFP18BAC-ART; ISBN:978-1-5386-1974-2

# *Driver Profiling Using Realistic Racing Games*

Manas Kale, M.V.Bedekar
Department of Computer Engineering,
MAEERs Maharashtra Institute of Technology
Kothrud, Pune, Maharashtra, INDIA
manaskale@hotmail.com, mangesh.bedekar@mitpune.edu.in

***Abstract*—All humans operating vehicles with several inputs(steering wheel, pedals, gears etc.) do so in a unique way. We attempt to identify drivers based on this characteristic driving profile. Users were made to play a realistic 3D driving simulation, StuntRally. Their keypress events were logged. These logs were used to train and test different classifiers such as Support Vector Machine(SVM), K Nearest Neighbour(KNN) and Naive Bayes(NB). The SVM performed best with an average testing accuracy of 80%.**

***Keywords*—Profiling, gaming, bio-metrics, keystroke identification, machine learning.***

## I. Introduction

User profiles are the virtual representation of each user and they include a variety of user information such as personal, interest and preference data[1][2]. These are essential to identifying users based on how they operate machines. Machine learning algorithms are essential in making computers learn these representations of users[3]. The content of a user profile varies from one application domain to another[4].In the case of a driver using a car, we can create a profile based on how he uses the various inputs to the car. The same concept can also be extended to users playing a car racing game. Users behavior while playing the game is monitored and analyzed to find patterns. In most cases, repetitive patterns existed - many actions were performed in the same order at different points in time. This observed behavior is different for different users. This user behavior patterns can thus be used to build a user profile.

This user behavior thus observed can be used to fine tune the game, to detect gamer churn prediction, identifying undesirable behavior in single player, multi-player or massively-multi player games[6].We can also identify and co-relate the users behavior in the virtual world to the users behavior in the real world[7].

## II. Related work

Can Et al.[5] show that it is possible to identify and authenticate users based on typing patterns using keystroke dynamics. As mentioned earlier, if the user is tasked with a skill-based activity like driving a car, there will be more identifiable patterns in keypress data. Dangra Et al.[8] were able to show that identifying drivers based on how they play a driving game is possible. Extending on their work, we used a 3D realistic driving game to train and test classifiers to identify drivers and used the observed data to comment on their driving behaviour in the real world.

## III. Implementation Details

### A. Selecting Game

We selected StuntRally[11] as the game to use because:

- The game is open source. It allows us to customize source code to suit our needs.
- It is a 3D game and simulates real physics such as tire friction, car braking, suspension etc.
- Provides a simple interface for the users to drive the car.
- It has multiplayer modes which allow us to collect data from several users at once.

Fig. 1. StuntRally.

### B. Log file format

The source code of the game was edited to create log files every time a user played a match. Specifically, the input event handling code was edited to write input events to a file. 6 keys were recorded:

- **UP** - Used to accelerate the car.
- **DOWN** - Used to apply normal brakes.
- **LEFT** - Used to turn left.
- **RIGHT** - Used to turn right.
- **SPACE** - Used to apply handbrake.
- **CTRL** - Used to provide boost in acceleration.

The format of the log file was as follows:

**DATE:**
**CAR NAME:**

**KEYNAME, KEYSTATE, TIMESTAMP**
where,

- **KEYNAME** : Name of key.
- **KEYSTATE** : State of key (pressed/released).
- **TIMESTAMP** : Timestamp when event occured in format HH:MM:SS.SSS.

Also, delimeter lines are also present which record the ending of a lap.

```
1  Log generated on :Wed Jan 17 13:00:38 2018
2
3  Car name is : ES
4
5  UP,PRESSED,13:01:14.213
6  UP,RELEASED,13:01:14.237
7  UP,PRESSED,13:01:14.284
8  UP,RELEASED,13:01:14.597
9  UP,PRESSED,13:01:14.623
10 RIGHT,PRESSED,13:01:20.041
11 RIGHT,RELEASED,13:01:20.257
```

Fig. 2. Example of log file.

```
190 UP,RELEASED,13:02:41.742
191 UP,PRESSED,13:02:42.388
192 UP,RELEASED,13:02:42.471
193 UP,PRESSED,13:02:42.674
194 lap 1,13:02:42.950,------------------------------------
195 UP,RELEASED,13:02:44.780
196 UP,PRESSED,13:02:44.950
197 LEFT,PRESSED,13:02:45.474
198 LEFT,RELEASED,13:02:47.245
199 RIGHT,PRESSED,13:02:47.646
```

Fig. 3. Example of lap ending.

### C. Data collection

A total of 8 subjects were made to play the game. Each played the same map with the same car. Each subject was required to play a minimum of 5 laps to eliminate outliers. This lap data was then split into two sets- one to train and other to test the classifier. Thus, we gathered approximately 2,500 events from each player.
Henceforth, a 'log file' will mean a single playing session of 5 or more laps.
K is the vector of all keys recorded.

$$K = [K_1, K_2, K_3, K_4, K_5, K_6]$$

where,
$K_1$ : UP
$K_2$: LEFT
$K_3$ : RIGHT
$K_4$: DOWN
$K_5$: SPACE
$K_6$: CTRL

$F_t$ is the vector of frequency of each keypress until time *t*.

$$F_t = [F_1, F_2, F_3, F_4, F_5, F_6]_t$$

where,
*t* : Elapsed time for lap in seconds.
$F_1$ : Frequency of $K_1$ keypress until *t* seconds.
$F_2$: Frequency of $K_2$ keypress until *t* seconds.
$F_3$ : Frequency of $K_3$ keypress until *t* seconds.
$F_4$: Frequency of $K_4$ keypress until *t* seconds.
$F_5$: Frequency of $K_5$ keypress until *t* seconds.
$F_6$: Frequency of $K_6$ keypress until *t* seconds.

$D_t$ is the vector of time for which each key was depressed in seconds until time *t* (key downtime).

$$D_t = [D_1, D_2, D_3, D_4, D_5, D_6]_t$$

where,
*t* : Elapsed time for lap in seconds.
$D_1$ : Time $K_1$ was pressed for until *t* seconds.
$D_2$: Time $K_2$ was pressed for until *t* seconds.
$D_3$ : Time $K_3$ was pressed for until *t* seconds.
$D_4$: Time $K_4$ was pressed for until *t* seconds.
$D_5$: Time $K_5$ was pressed for until *t* seconds.
$D_6$: Time $K_6$ was pressed for until *t* seconds.

For each user, one lap can be represented by the concatenation of *t* with the two vectors $F_t$ and $D_t$:

$$U = t||F_t||D_t$$

where *t* varies form 0 to lap end. **See graphs at the end for visual representation of *U***. We hypothesized that this 7-dimensional vector would contain sufficient unique features to identify users.

In some cases, users were not able to keep their vehicles on track and lost control. The resulting keypresses to get back on track introduced outliers which cannot be used to represent users. To remove these, per-lap data (set U as defined earlier) was averaged. This removed most, but not all outliers.
A 70% training and 30% testing data split was used. In other words, 70% of the laps played were averaged and used as training and the remaining 30% was used for testing.

### D. Classifiers Used

Python's scikit-learn[12]library was used for these machine learning algorithms.

*1) Support Vector Machine(SVM):* A SVM classifier represents labeled data in hyperdimensional space. SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces. SVM's are effective in high dimensional spaces and in cases where number of dimensions is greater than the number of samples. The classifier uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.

A one-against-one approach [9] was used for multi-class classification.

*2) K Nearest Neighbour(KNN):* Neighbors-based classification is a type of instance-based learning or non-generalizing learning: it does not attempt to construct a general internal model, but simply stores instances of the training data. Classification is computed from a simple majority vote of the nearest neighbors of each point: a query point is assigned the data class which has the most representatives within the nearest neighbors of the point[10]. After hyper-parameter tuning, K = 5 was found to give relatively the least error.

*3) Naive Bayes(NB):* Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes theorem with the naive assumption of independence between every pair of features.

## IV. Results

*Graphs*

*Refer last page of document for graphs. In the given graphs, only 5 user's data has been plotted to reduce clutter and increase visibility.*
*Each line represents either testing or training dataset.*

Fig.4 shows the frequency of UP and LEFT keys pressed by 5 users for their training and testing data. We observe that

*different user's plots are relatively separated*. The classifiers learns this separation when being trained. We can also see that some user's driving characteristics are drastically different from others. Users 1 and 3 pressed the UP key far less frequently than others. These users pressed the key and rarely let go of it- even when taking sharp turns or maneuvering difficult terrain. This is characteristic of aggressive driving. Also, all user's training and testing datasets are relatively similar except User 1. User 1's initial sessions involved a crashing into the terrain- this also explains User 1 taking a longer time to complete the session.

In Fig.5 we see again that User 1 and 3 pressed the SPACE key more frequently than the average. This is in accordance with the earlier observation- these users never let go of the accelerator. In order to slow their speed, they had to use breaking more than the others. The other users simply let go of the UP key when they encountered turns in order to reduce their speed. This correlation between two features belies the assumption of NB classifier(that features are not related to each other), and thus led us to believe that the NB classifier will not perform as good as the other two.

Such unique characteristics help identify unique users. Some users have similar styles of driving, but with enough data with low variance, we can train a classifier to identify these characteristics. We have only seen two combinations out of the several possible with the feature vector $U$.

However, *all* feature combinations cannot be used. For example, in Fig.6 we see all users pressed the LEFT and RIGHT keys with relatively the same frequency, even for User 1 and 3. All the plots are close to each other and it is difficult to classify a single user using only this feature combination. For each LEFT key press, the user has to press and equal amount of RIGHT key in order to keep the vehicle centered on the road.

Fig. 7 shows the variance between training and testing datasets for each user. In accordance with the previously mentioned figures, User 1 shows a moderate amount of variance between their training and testing data. User 8(not shown in previous graphs) shows the most variance between training and testing sessions. This user is the most difficult to classify with such variance.

Fig.8 shows the accuracy of different classifiers used for each user. SVM was found to be the most robust with an average accuracy of 80%, followed by KNN with 62% and NB with 53%. In accordance with the variance plot, we see that the classifiers had the most difficulty with User 8 - the user with the most variance. All 3 classifiers performed best with the user with the least variance (User 5).

## V. Conclusion and Future Scope

The results show that a simple set of 6 keys is enough to identify users based only on their keystrokes. In addition to user identification, we can also comment on how users will behave in the real world - some users drive more aggressively than others. To further improve the accuracy, we can use a Neural Network for better classification. A larger dataset may also be generated through multiplayer matches. Also, if used in an actual vehicle with sensors we will have much more analog data to work with.

This can find various other applications such as vehicle security, authorization and identification systems, personalized gaming services and many more.

## References

[1] G. Araniti, P. D. Meo, A. Iera and D. Ursino (2003), "Adaptive controlling the QoS of multimedia wireless applicationsthrough user profiling techniques", IEEE Journal on selected ar-eas in communication, 21(10), pp. 1546-1556.

[2] M. J. Martin-Bautista, D. H. Kraft, M. A. Vila, J. Chen andJ. Cruz (2002), "User profiles and fuzzy logic for web retrieval issues", Soft Computing (Focus), 15(3-4), pp. 365-372.

[3] Ayse Cufoglu, "User Profiling - A Short Review", International Journal of Computer Applications (0975 8887), Volume 108 - No. 3, December 2014.

[4] Schiaffino S., Amandi A. (2009), "Intelligent User Profiling". In: Bramer M. (eds) Artificial Intelligence An International Perspective. Lecture Notes in Computer Science, vol 5640. Springer, Berlin, Heidelberg, Pp. 193-216.

[5] Y. S. Can and F. Alagz, "User identification using Keystroke Dynamics," 2014 22nd Signal Processing and Communications Applications Conference (SIU), Trabzon, 2014, pp. 1083-1085 DOI: 10.1109/SIU.2014.6830421.

[6] Kyong Jin Shim et. al., "Analyzing Human Behavior from Multiplayer Online Game Logs - A Knowledge Discovery Approach," IEEE Intelligent Systems (February 2011 Issue).

[7] Ahmad, M.A., Shen, C., Srivastava J.,"Predicting Real World Behaviors from Virtual World Data", N. (Eds.) , 2014, Springer.

[8] Bharat S. Dangra, Digvijaysingh Rajput, M. V. Bedekar, Suja S. Panicker, "Profiling of Automobile Drivers Using Car Games", International Conference on Pervasive Computing, Sinhgad College of Engineering, Pune, 810 January, 2015. DOI: 10.1109/PERVASIVE.2015.7087173

[9] S. Knerr,L.Personnaz,G.Dreyfus, "Single-layer learning revisited : a stepwise procedure for building and training a neural network",in:F.F.Souli, J.Hrault (Eds.), Neurocomputing:Algorithms,Architectures and Applications,NATOASI Series, vol.68,Springer-Verlag,Berlin Heidelberg,1990,pp.4150.

[10] Altman, N. S. (1992). "An introduction to kernel and nearest-neighbor nonparametric regression". The American Statistician. 46 (3): 175185. DOI:10.1080/00031305.1992.10475879.

[11] StuntRally: https://stuntrally.tuxfamily.org/ .

[12] Scikit-learn: http://scikit-learn.org/ .

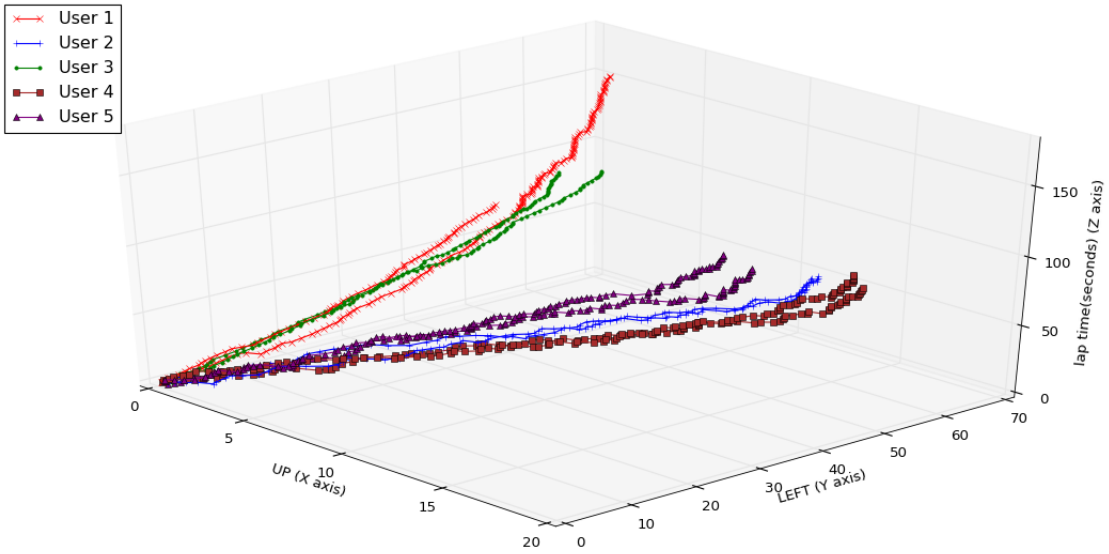


Fig. 4. Frequency of UP vs LEFT keys.

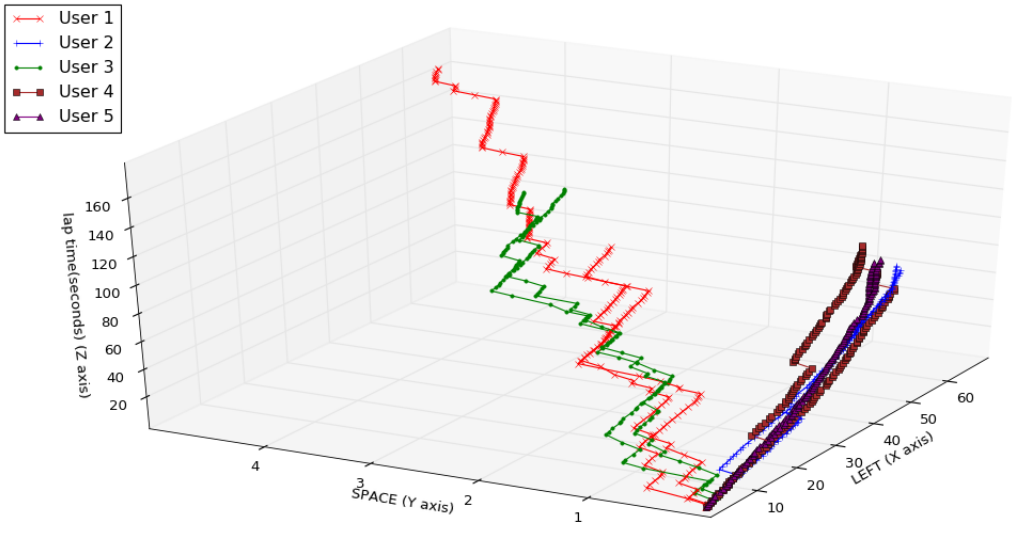


Fig. 5. Frequency of LEFT vs SPACE keys.

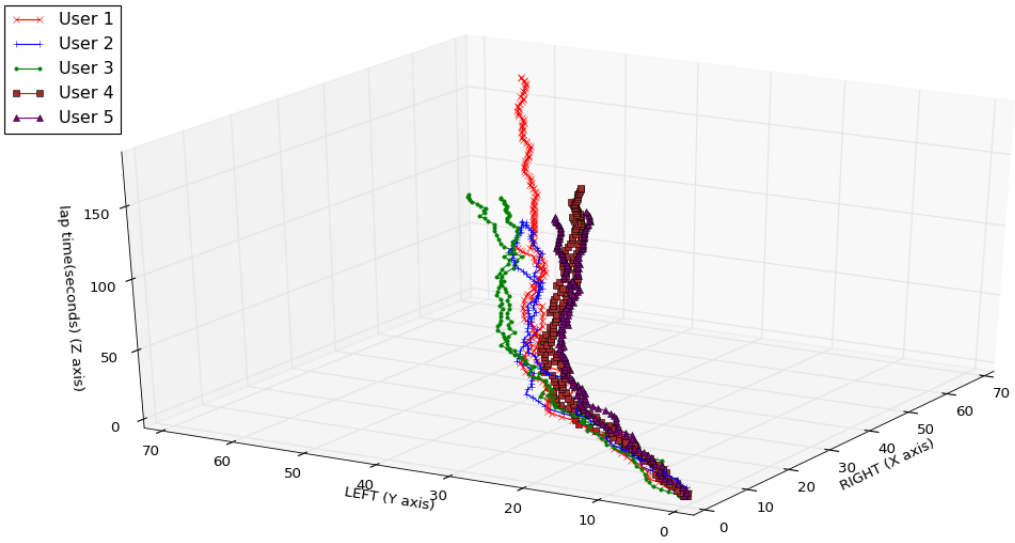


Fig. 6. Frequency of LEFT vs RIGHT keys.

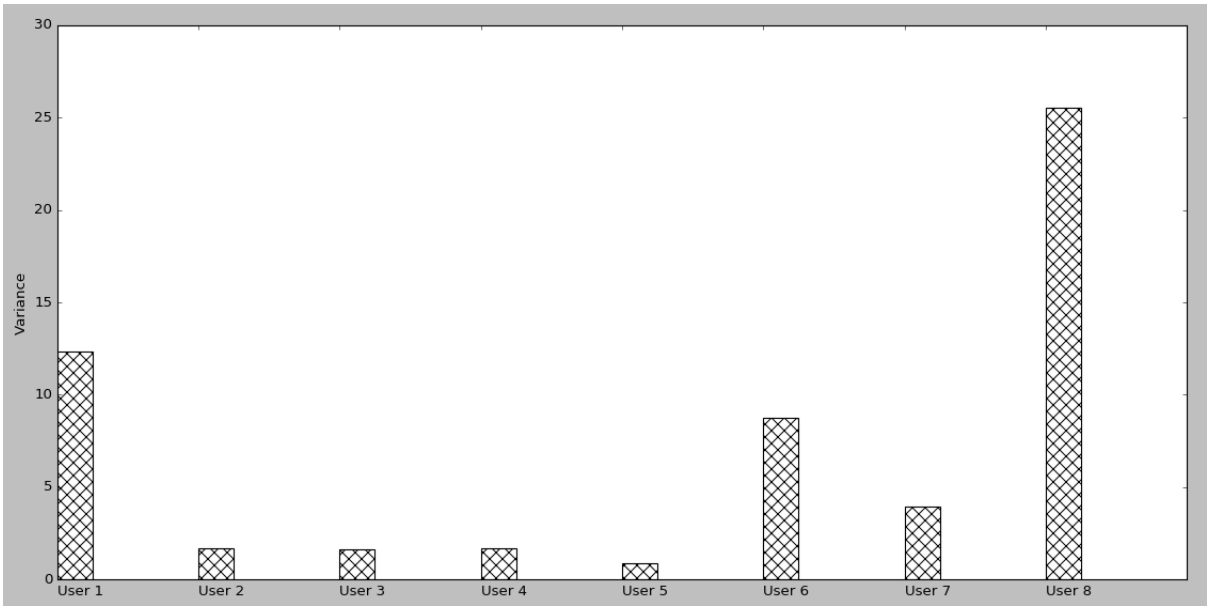


Fig. 7. Variance between training and testing data.

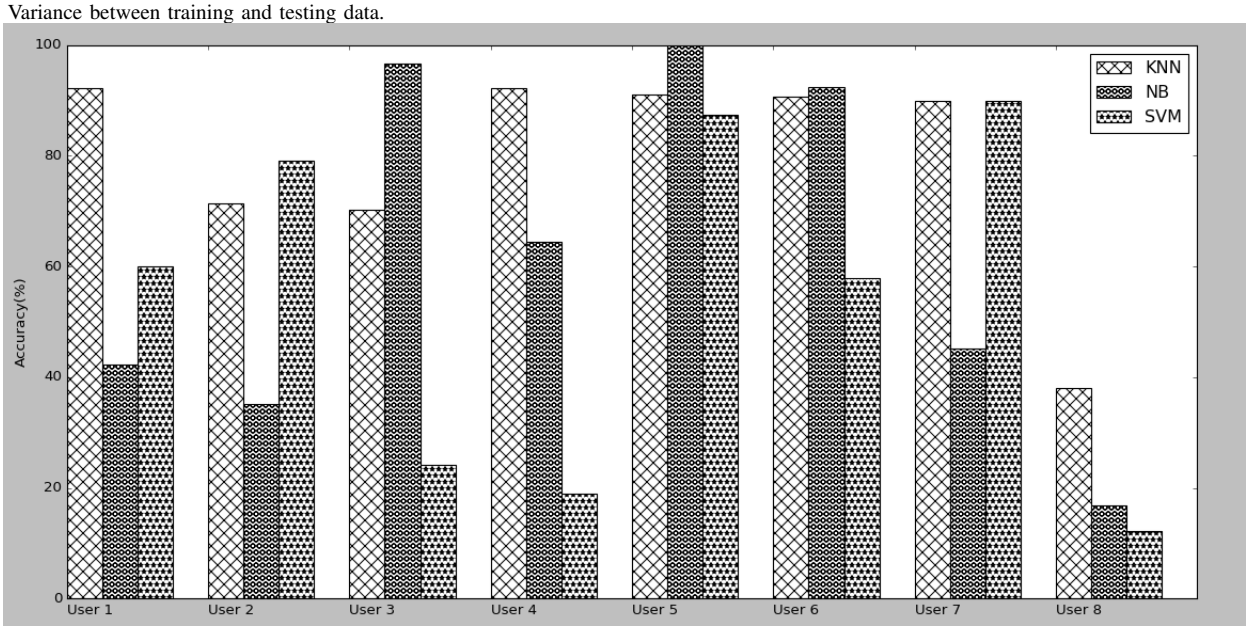


Fig. 8. Classification results.