# Decoding VR Motion Sickness With The VR.net Dataset

Vinayak Gajjewar*
Swathi Bhat*
vinayakgajjewar@ucsb.edu
swathi_bhat@ucsb.edu
UC Santa Barbara
Santa Barbara, California, USA

## ABSTRACT

The VR.net dataset [19] represents a pioneering effort to standardize datasets for studying motion sickness in virtual reality (VR) environments. This novel dataset is intended to support the development of machine learning-based prediction models for motion sickness triggers and facilitate releated user studies. Given its recent introduction, the dataset's characteristics and potential research applications remain largely unexplored. Our research aims to conduct a comprehensive investigation of the VR.net dataset, focusing on identifying patterns, analyzing data, and uncovering any inherent issues. We evaluate the dataset's strengths and weaknesses, determining what aspects are effective and where it may fall short. Additionally, we juxtapose our findings with existing user studies to validate our results, highlight discrepancies, and uncover potential issues within the dataset or the studies themselves. This comparative analysis aims to enhance the reliability and applicability of the VR.net dataset in VR motion sickness research.

Up to 40% of virtual reality users experience VR motion sickness, with symptoms including fatigue, disorientation, and nausea. We apply statistical analyses and machine learning to in-game camera view and rotation matrices to find factors correlated with VR motion sickness. In this work, we present a comprehensive list of issues and shortcomings with the VR.net dataset. We also present an explanation of how to pre-process game data with an eye towards machine learning applications. We also present a statistical analysis of VR.net and a prototype machine learning model to predict the onset of motion sickness in VR users.

## CCS CONCEPTS

• **Human-centered computing** → *HCI design and evaluation methods*;

## KEYWORDS

Virtual Reality, Motion Sickness, Dataset Analysis, Extended Reality, Machine Learning

*Both authors contributed equally to this research.


*Conference acronym 'XX, June 03–05, 2024, Woodstock, NY*

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

**ACM Reference Format:**
Vinayak Gajjewar and Swathi Bhat. 2024. Decoding VR Motion Sickness With The VR.net Dataset. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX)*. ACM, New York, NY, USA, 12 pages. https://doi.org/XXXXXXX.XXXXXXX



## 1 INTRODUCTION

Despite extensive research on VR motion sickness, including studies on inducing factors and potential remedies, the absence of a standardized dataset has been a significant gap. Numerous studies applyin machine learning techniques for motion sickness prediction have emerged, yet they lack a common dataset. The recent introduction of the VR.net dataset [19] addresses this gap by providing consistency, enhancing generalizability, and enabling researchers to benchmark solutions aimed at reducing motion sickness.

Given the novelty of this dataset, existing research using it is limited. One study [10] primarily focuses on behavioral biometrics using kinetic signatures in VR, while another [15] compared eight XR motion datasets, evaluating various specifications such as coordinate systems and file formats. The primary focus of VR motion sickness rovolves around machine learning-based prediction models for potential motion sickness triggers. However, these is a notable lack of thorough exploratory analysis of the dataset to uncover underlying patterns and trends. Before applying machine learning models, it is essential to comprehensively understand the dataset's contents. Additionally, previous implementations using the VR.net dataset have mostly utilized images from gameplay sessions for prediction, leaving the CSV data largely unexplored.

## 2 MOTIVATION AND PROBLEM STATEMENT

### 2.1 Motivation

Almost 40% of VR users experience motion sickness during interaction with VR games/scenarios. These users experience a range of symptoms including fatigue, disorientation, and nausea. According, much research has been performed to find the causes and predict the onset of VR motion sickness. However, these studies cite a lack of a training datasets as a major hindrance to this goal. We aim to evaluate the merits of the VR.net dataset and its ability to train an ML solution to predict the onset of VR motion sickness.

Conducting a thorough data analysis before applying machine learning models is essential for several reasons.

(1) Data analysis helps uncover hidden patterns, relationships, and characteristics crucial for informed decision-making.
(2) Data analysis aids in detecting outliers, missing values, and other potential issues within the dataset that could affect the performance and accuracy of ML models.

(3) A detailed analysis provides insights that assist in choosing the appropriate ML model and fine-tuning its parameters for optimal performance.
(4) Early detection of biases in the dataset can help mitigate them, ensuring that the ML models built on this data are fair and unbiased.

A thorough analysis could identify discrepancies with prior studies and uncover issues within the dataset or the studies themselves.

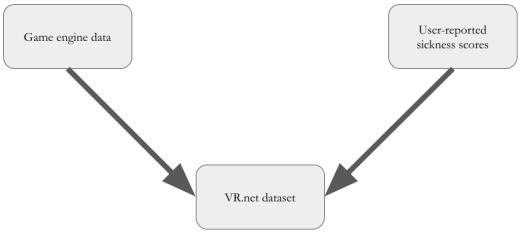


**Figure 1: The VR.net dataset combines game engine data with user-reported motion sickness scores.**

## 2.2 Problem statement

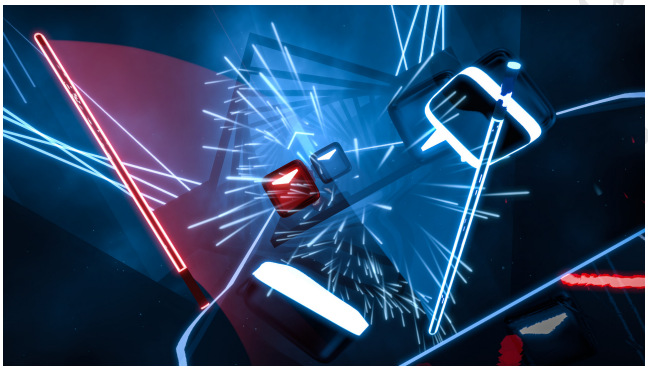

**Figure 2: Screenshot of the VR game Beat Saber, in which gamers slice flying blocks in time to music.**

We have identified several key areas for analysis within the VR motion sickness dataset. For each area, we reference existing research studies that have conducted user studies in similar domains. Our objective is to analyze and visualize this data, compare it with the referenced studies, and identify any discrepancies. Table 1 outlines the areas we analyzed along with the corresponding research studies in similar domains.

# 3 RELATED WORK

Our work focuses on VR.net [19], a dataset designed to aid research on VR motion sickness. The lack of standardized datasets in this field makes it challenging to compare results across various studies, as each relies on its own methods and user studies for data collection. Since this dataset is new, no current studies use it to predict VR motion sickness. [15] compared various XR-related datasets, focusing on parameters such as coordinate systems, rotation representations, file formats, and time formats, but it was not specifically related to VR motion sickness.

In addition to analyzing and identifying gaps in the VR.net dataset, our work aims to validate previous research on VR sickness, referencing results from multiple papers. Studies such as [12, 13, 17, 18] found that increased camera movements can make users more susceptible to VR sickness compared to a still camera. VR sickness can also result from excessive camera accelerations or decelerations, as shown in [8, 9]. Additionally, users may experience more discomfort with rotational camera movements compared to translational movements [3]. Several studies [6, 11] have linked VR sickness to overly wide FOV settings, noting that restricting the FOV can effectively alleviate both subjective and objective symptoms of VR sickness. Inappropriate DoF (Depth of Field) settings can also cause visual discomfort [4].

With numerous studies involving user research and others using ML-based approaches to predict motion sickness, our work aims to analyze this novel VR motion sickness dataset and identify scenarios where ML-based approaches can be most effectively applied.

**Figure 3: Screenshot of the VR game Epic Roller Coasters, which simulates a variety of rollercoasters.**

**Table 1: Mean sickness score across games**

| Game | Mean sickness score |
|---|---|
| Beat Saber | 1.79 |
| Epic Roller Coasters | 2.75 |
| Voxel Shot VR | 1.29 |
| Cartoon Network Journeys VR | 1.57 |
| VR Monster Awakens | 2.21 |

# 4 METHODS

## 4.1 Dataset details and evaluation

Currently, version 1 of the VR.net dataset is publicly available, and Table 2 provides a summary of its contents. Our analyses utilize

**Table 2: VR.net dataset summary**

| Parameter | Details |
|---|---|
| Number of participants | 36 |
| Number of games | 10 |
| Number of playing sessions | 62 |
| Data provided per session | 5 CSV files |
| Motion sickness data | 62 FMS ratings |

the motion sickness data and additional data from each playing session.

(1) Camera.csv contains the view and projection matrices of the in-game camera. This data can be used to analyze how increased camera movements, acceleration/deceleration, and rotational movements affect motion sickness.
(2) Pose.csv records the locations of the VR headset and controllers over time. This data helps determine if excessive headset movements increase the likelihood of motion sickness.
(3) Control.csv provides information on the location of the headset and controllers, as well as the actions performed with them. This data can help analyze if restricting controller movement induces motion sickness.
(4) Light.csv contains data on lighting conditions during gameplay. This data can be studied to determine the impact of light intensity and color on motion sickness.
(5) Object.csv includes the transformation matrix and bounding boxes for in-game objects during each session. This data can be used to examine whether higher object density and vection contribute to VR motion sickness.

We focused on the following areas in our work and validated our results against existing research:

(1) Overall trend of motion sickness scores over time
(2) Effect of light intensity on motion sickness
(3) Impact of increasing object density on motion sickness
(4) Influence of headset rotational movements on motion sickness scores
(5) Effect of camera acceleration/deceleration on motion sickness

Additionally, we conducted an in-depth analysis of specific games that appeared particularly promising for studying these phenomena.

## 4.2 Tools and technologies

We used Python via Google Colab and the Pandas, numpy, and Scikit-Learn libraries to perform our analyses. We used the Plotly [1] library to create our figures, and AutoGluon [2] for machine learning.

## 4.3 Data transformation

Before performing statistical analyses or applying machine learning models, we had to transform the raw data given through the VR.net dataset into tangible numerical features that capture relevant patterns in the data. The file camera.csv gives us the projection matrix and view matrix for the in-game camera. The pose.csv file gives us velocity and angular velocity of the headset. The below paragraphs detail how we decompose each of these matrices into numerical features.

We also decompose the camera projection matrix into four quantities: near clipping plane, far clipping plane, FOV in the x-direction, FOV in the y-direction, and aspect ratio.

```
near = projmat[2][3] / (projmat[2][2] - 1.0)
far = projmat[2][3] / (projmat[2][2] + 1.0)
yfov = 2.0 * np.arctan(1.0 / projmat[1][1])
aspectr = projmat[1][1] / projmat[0][0]
xfov = 2.0 * np.arctan(np.tan(yfov / 2.0) * aspectr)
xfovdeg = np.degrees(xfov)
yfovdeg = np.degrees(yfov)
```

Finally, we decompose the camera view matrix into position, rotation, forward, and up vectors.

```
position = viewmat[:3, 3]
rotation = viewmat[:3, :3]
forward = -viewmat[:3, 2]
up = viewmat[:3, 2]
```

We convert the camera rotation matrix into its quaternion form using the below equations. [16] Quaternions are superior to matrices for representing 3D rotations because of gimbal lock or whatever. A quaternion is an alternative way to specify a rotation based on an axis (represented by a 3D vector) and a rotation relative to that axis. A quaternion is represented by 4 elements q= (q1, q2, q3, q4) where q2 q3 and q4 represent that 3d axis and q1 is the rotation about that axis. The below equations show how we convert from a rotation matrix to a quaternion.

```
def rot2quat(rot):
  w = np.sqrt(1 + rot[0, 0] + rot[1, 1] + rot[2, 2]) / 2
  x = (rot[2, 1] - rot[1, 2]) / (4 * w)
  y = (rot[0, 2] - rot[2, 0]) / (4 * w)
  z = (rot[1, 0] - rot[0, 1]) / (4 * w)
  return np.array([w, x, y, z])
```

# 5 MACHINE LEARNING

## 5.1 Problem specification

We model VR sickness prediction as a multi-class classification problem, where each sickness score from 1 - 5 is a unique category. Given camera and headset information from a particular timestamp, we aim to build a machine learning solution to predict the sickness score. We used the popular machine learning library AutoGluon [2] to train, validate, and test our model. We used an 80/20 split to separate our training and test data. Table 3 presents each feature our model uses to learn this problem.

**Table 3: ML model features**

| Feature name | Description |
|---|---|
| rotquat[4] | Quaternion representation of in-game camera rotation matrix |
| pos[3] | View matrix position vector |
| forward[3] | View matrix forward vector |
| up[3] | View matrix up vector |
| rot[9] | Rotation matrix of in-game camera |

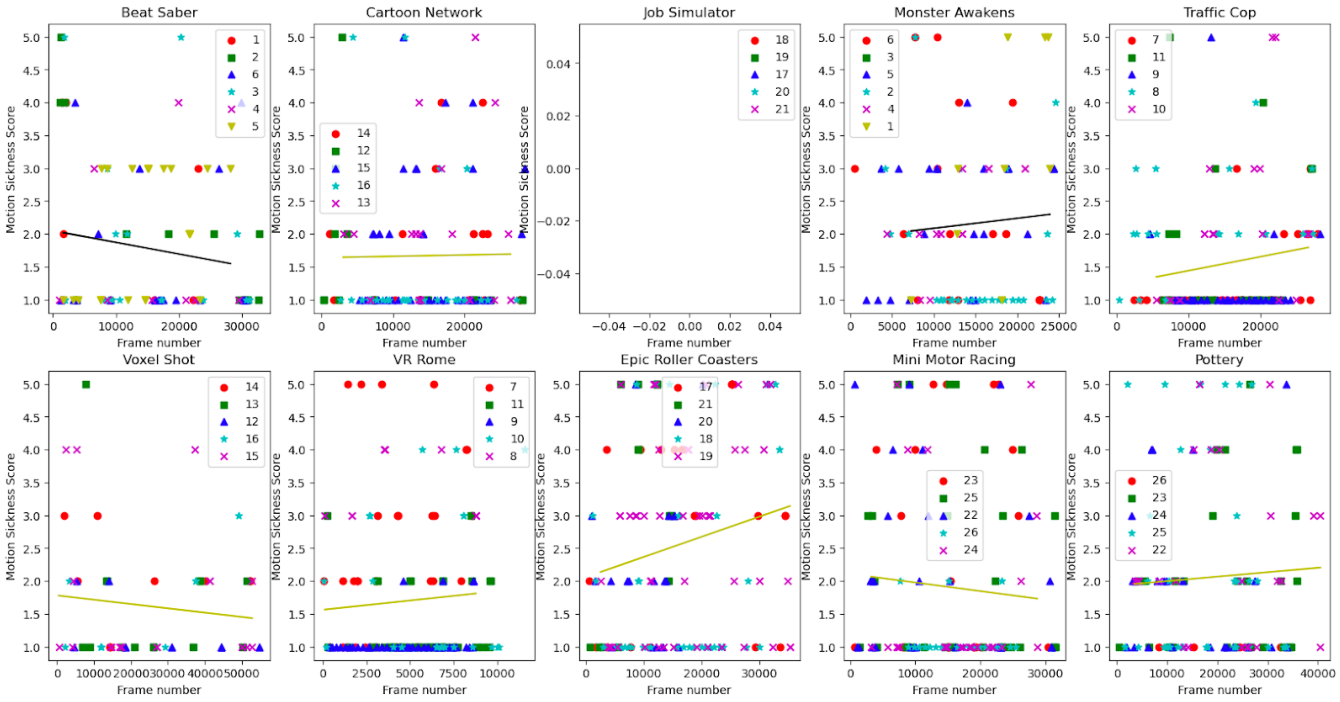


**Figure 4: Motion sickness over time for all 10 games in VR.net dataset**

**Table 4: Variables with |correlation| > 0.5 with sickness score for Beat Saber.**

| Feature name | Correlation with sickness score |
|---|---|
| pos1 | -0.52 |
| rot0 | -0.58 |

**Table 5: Variables with |correlation| > 0.5 with sickness score for Epic Roller Coasters.**

| Feature name | Correlation with sickness score |
|---|---|
| pos1 | -0.69 |
| pos2 | 0.65 |
| pos3 | -0.64 |
| forward2 | -0.64 |
| up2 | 0.64 |
| rot1 | 0.59 |
| rot5 | 0.64 |
| rot6 | -0.69 |

## 6 RESULTS

### 6.1 Dataset issues

As part of our analysis, we identified several issues or issues that could hamper the usage of the dataset when using in ML-related applications.

(1) All the data for the 'Job Simulator' game was missing. Ideally, there should have been a folder for each playing session containing the 5 csv files for that session. While the motion sickness scores for 5 participants playing this game was included, the rest of the data was missing.

(2) Light related data for all participants playing 'Voxel Shot VR' was missing. As a result, none of the analysis related to light-related data could be performed for this game.

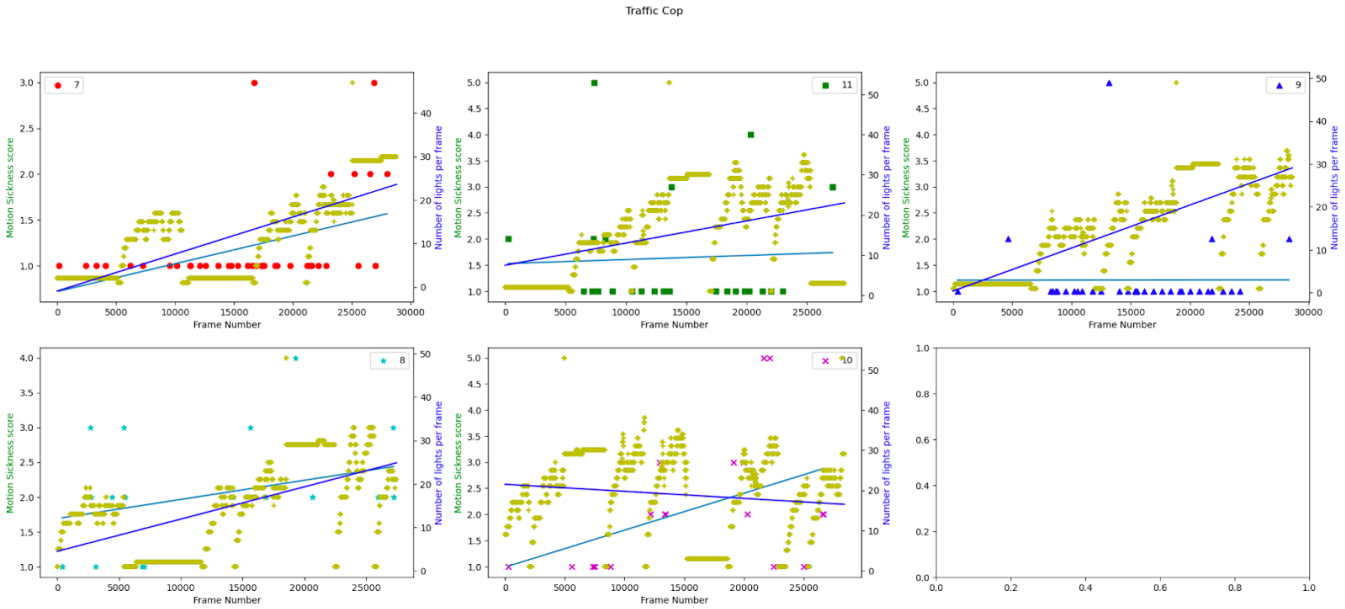


**Figure 5: Effect of light density on motion sickness for Traffic Cop game**

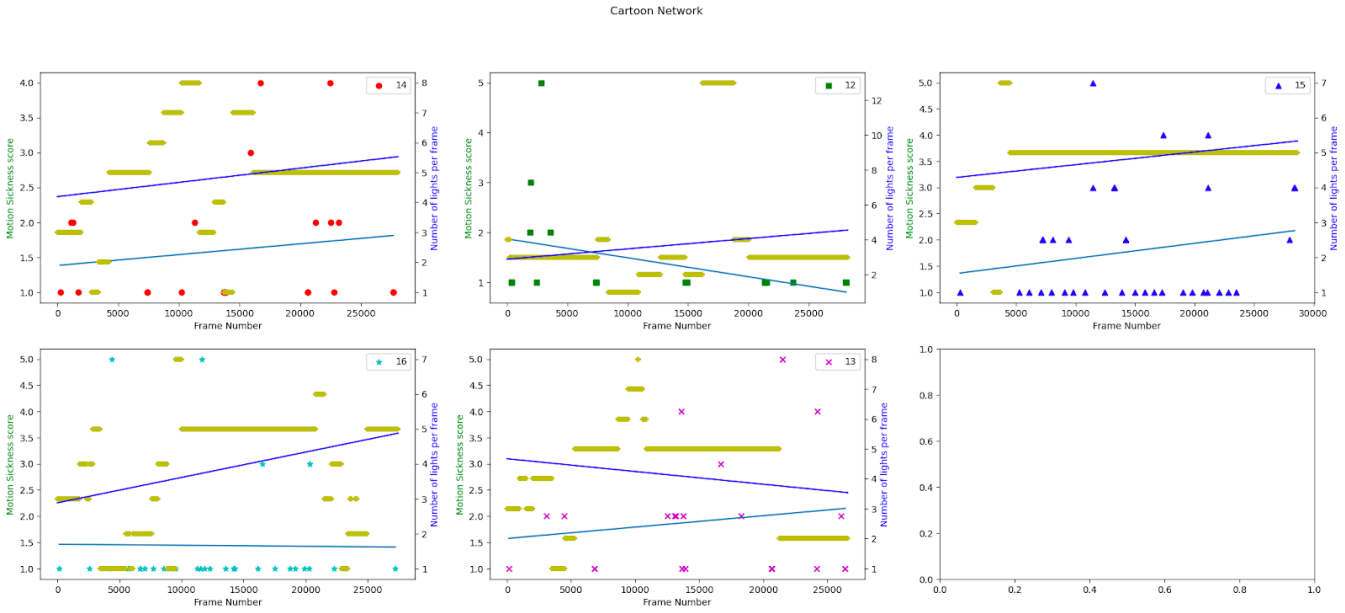


**Figure 6: Effect of light density on motion sickness for Cartoon Network game**

(3) Object.csv for the 'Monster Awakens' game had commas within the 'name' column value. This would be problematic during parsing the data if using Python libraries like Pandas. A probable fix would be to modify the column values from "CFXM4_Space Thruster C (Blue-Purple, CFX Blend)" to "CFXM4_Space Thruster C (Blue-Purple CFX Blend)" , "CFX4 Smoke Trail A (Curved, Black)_F4D(Clone)" to "CFX4 Smoke Trail A (Curved Black)_F4D(Clone)"

## 6.2 Analysis results

As Figure 4 shows, we observed a general trend of increasing motion sickness over time for all game play sessions. While this by itself does not yield any conclusive patterns, it did show that motion sickness increased over time for majority of the participants. Each of the graphs in the figure corresponds to a game in the dataset and each marker corresponds to

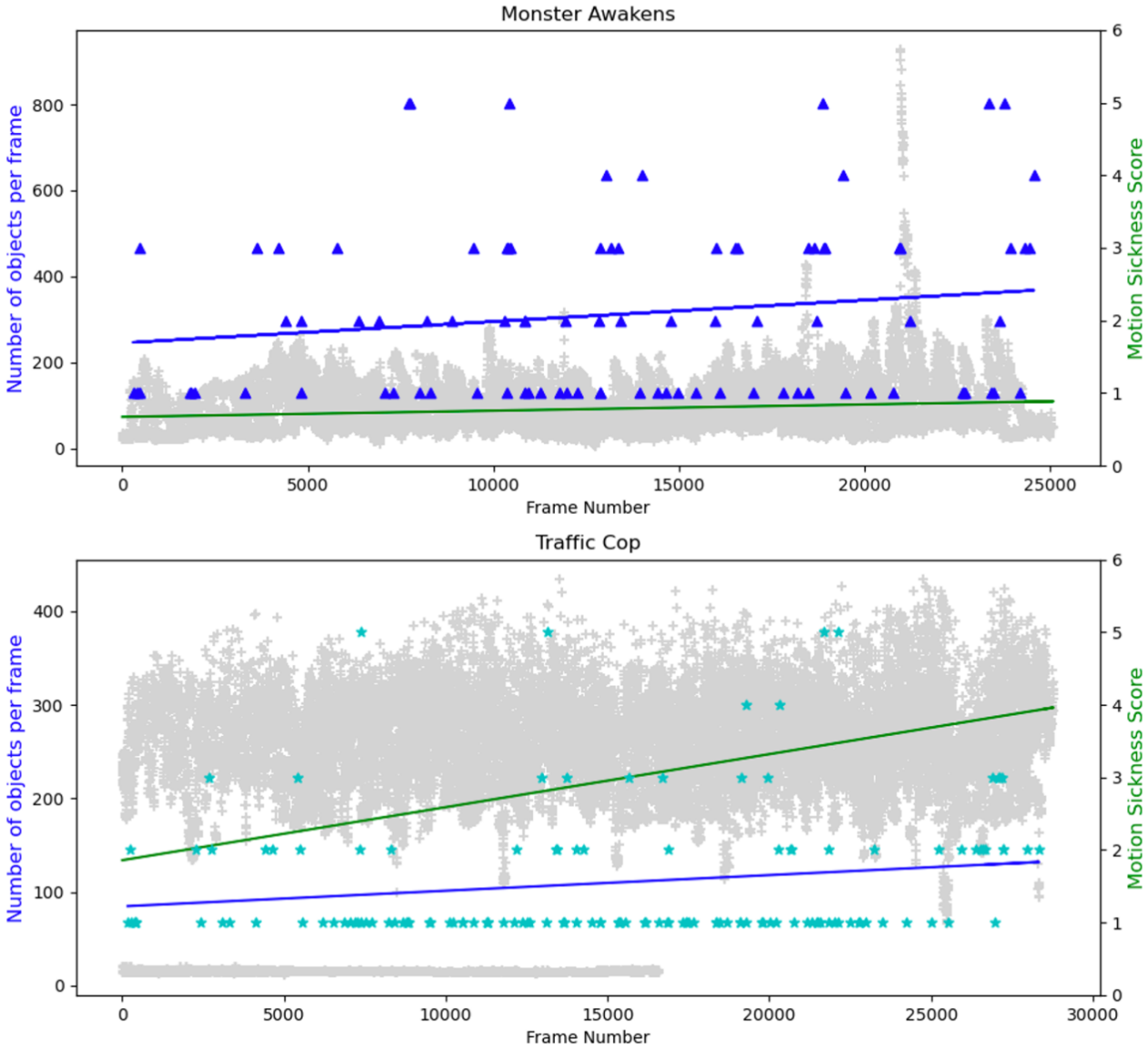


**Figure 7: Object density and motion sickness scores for Monster Awakens and Traffic Cop games.**

Figure 5, Figure 6 show the effect of increasing light density on motion sickness. The results are only included for two games (Cartoon Network and Traffic Cop) since these games were the only ones with changing light density. The rest had constant light density throughout the game session. For these two games, an increasing light density led to an increase in motion sickness scores. We feel this makes sense since an increasing density of light can produce more visual stimuli, making it more difficult for the brain to interpret the surroundings and contributing to an increase in motion sickness symptoms. The trend lines in the figure correspond to motion sickness scores and light density (green and blue respectively) with each graph corresponding to one game and each of the markers denoting participants playing the same game.

Analyzing object density with motion sickness however did not yield anything significant as shown in Figure 11 and Figure 12. This is mostly due to extremely high object density and very few motion sickness scores available to perform meaningful time series analysis on them.

Traffic Cop VR is one of the ten games in version 1 of VR.net that yielded positive outcomes in our analysis. In this VR game, players assume the role of a traffic cop at an intersection, directing cars to stop or proceed while evading various objects hurled at them. The gameplay involves significant turning and rotational head movements. Our previous analysis indicated that Traffic Cop's motion sickness scores increased over time, correlating with higher object density (as more objects must be dodged and more cars

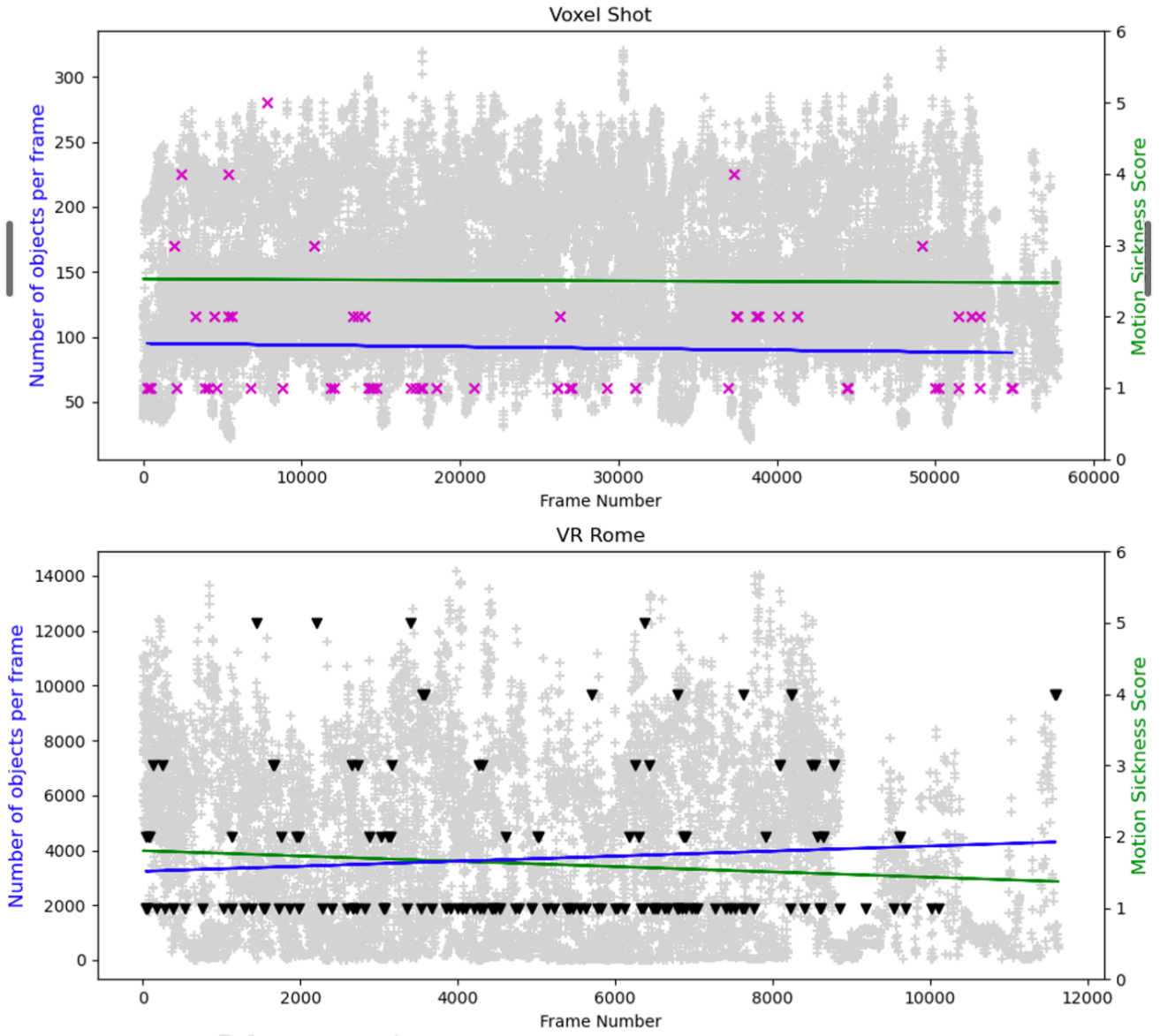


**Figure 8: Object density and motion sickness scores for Voxel Shot and VR Rome games.**

managed) and increased light intensity. Due to the nature of the game, we believe that variations in angular velocity, acceleration, and jerk contribute to the rise in motion sickness. Figures 6, 7, and 8 illustrate how changes in angular velocity, angular acceleration, and angular jerk affect motion sickness over time.

A comparison of the motion sickness encountered by the identical group of participants during VR Rome and Traffic Cop gameplay is illustrated in Figure 9.

## 6.3 Correlations for Beat Saber and Epic Roller Coasters

As shown in Table 4 and Figure 13, only two features had a statistically significant correlation with sickness score: `pos1` and `rot0`. They both display a strong negative correlation with sickness score. As shown in Table 5 and Figure 14, eight different features display a statistically-significant correlation with sickness score.

## 6.4 Per-game trends vs. generalizing across games

In this section, we will present some of our observations from working with the VR.net motion sickness dataset.

An unfortunate reality of this iteration of the VR.net dataset is that most of the games chosen do not cause enough motion sickness for us to find what factors cause it. Therefore, we had to prune our search space, and for our correlation analysis, we chose to focus on only two games, Beat Saber and Epic Roller Coasters. These games

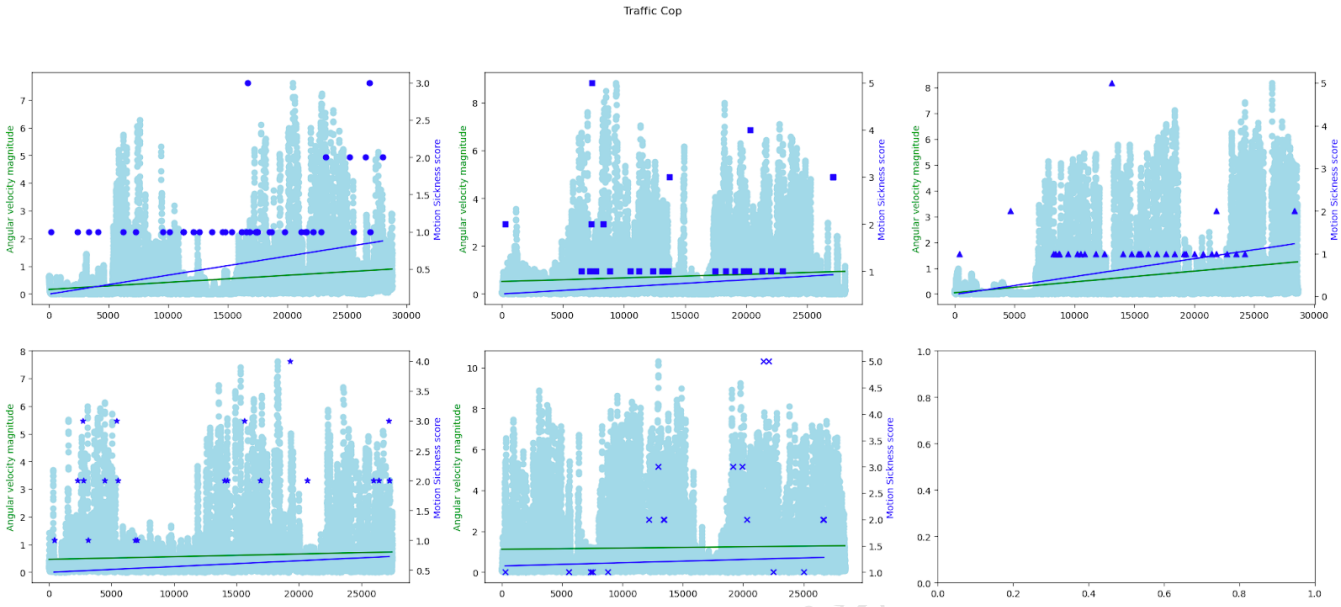


**Figure 9: Angular velocity and motion sickness scores for Traffic Cop game.**

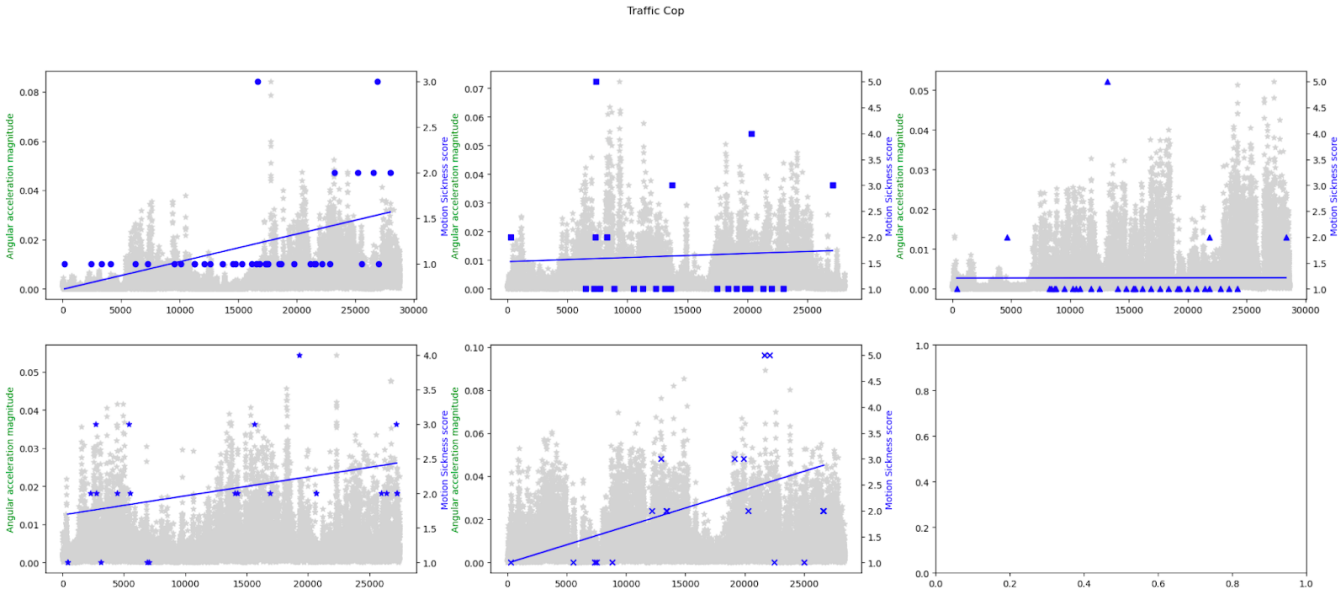


**Figure 10: Angular acceleration and motion sickness scores for Traffic Cop game**

had a mean sickness score of 1.79 and 2.75, respectively. The other games in this dataset either had a mean sickness score that was too low or did not display any clear correlations. When discovering what factors correlate with sickness scores, we found that they vary significantly from game to game. Only a little information carries over from game to game regarding what causes simulator sickness. This fact leads us to believe that it is best to look at trends from an individual game to determine what causes motion sickness in that game.

However, most games in this dataset need more data points for rigorous analysis and understanding of which factors correlate with motion sickness for that particular game. Therefore, we also decided to experiment with combining data from all the games and training a machine-learning model on the resulting dataset, as discussed in a previous section. This strategy worked well, as our resulting model has a test accuracy of 92% across all games considered. This result demonstrates that training a machine-learning model with

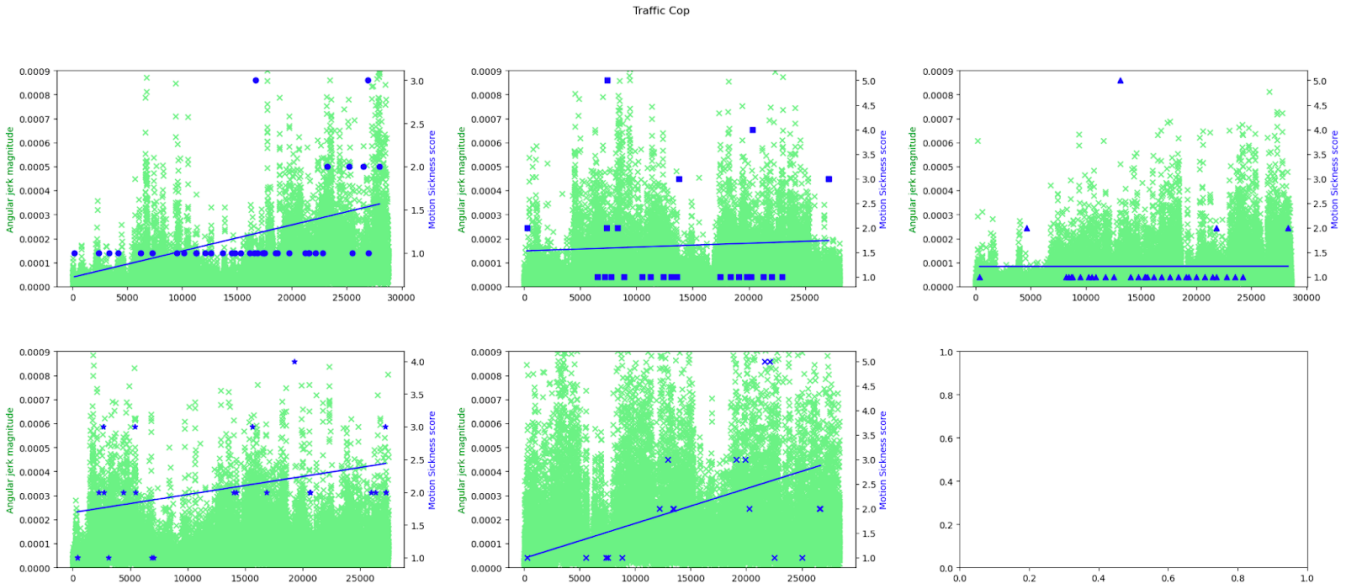


**Figure 11: Angular jerk and motion sickness scores for Traffic Cop game**

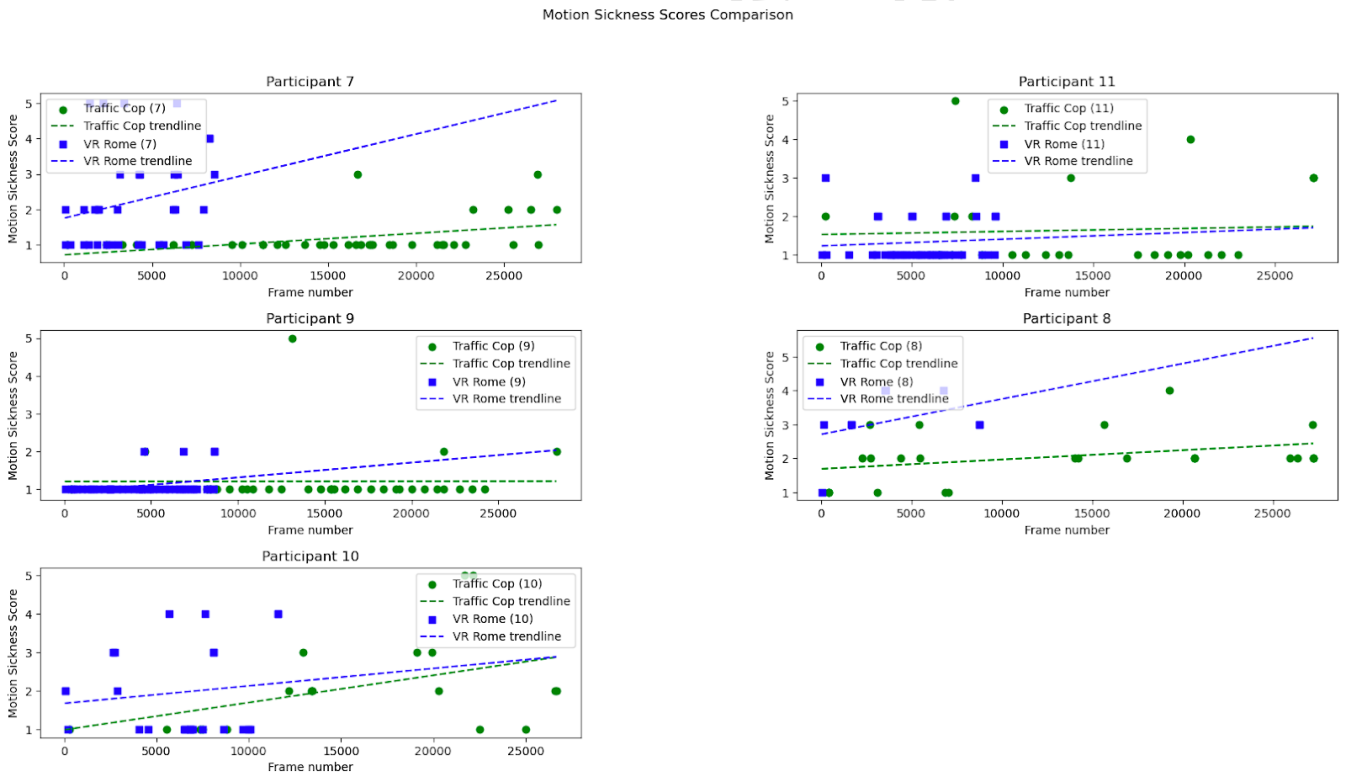


**Figure 12: Motion sickness scores for the same set of participants for Traffic Cop vs VR Rome**

high performance across VR scenarios and applications is possible and practical.

## 6.5 Machine learning model

As seen in Figure 15, a lot of the models we considered have a reasonably high accuracy. However, where they differ is in their inference

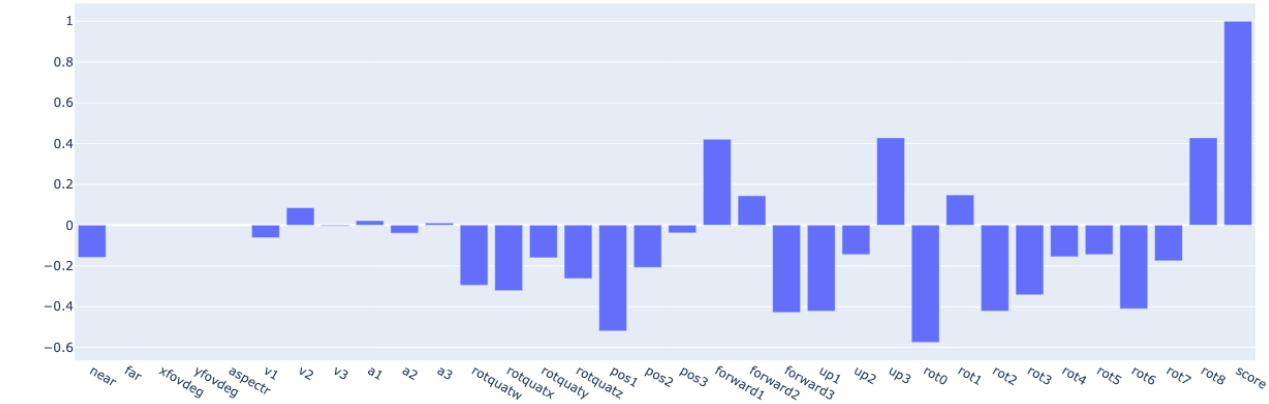


**Figure 13: Correlation for each variable with sickness score for Beat Saber.**

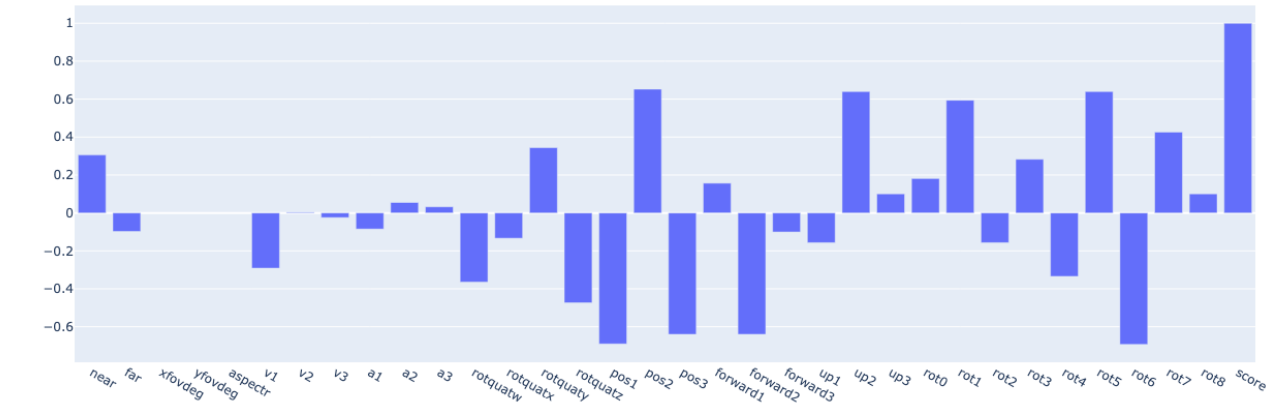


**Figure 14: Correlation for each variable with sickness score for Epic Roller Coasters.**

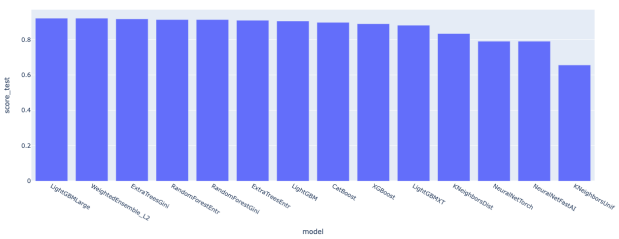


**Figure 15: Accuracy for each ML model.**

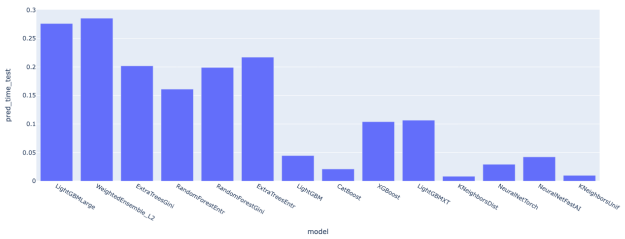


**Figure 16: Prediction time for each ML model.**

**Table 6: LightGBMLarge model hyperparameters**

| Hyperparameter name | Value |
|---|---|
| learning_rate | 0.03 |
| num_leaves | 128 |
| feature_fraction | 0.9 |
| min_data_in_leaf | 3 |

time. We report the inference time for each model in Figure 16, where we can see a lot of variation. The models with the highest inference time are `WeightedEnsemble_L2`, `LightGBMLarge`, and `ExtraTreesEntr`. The models with the lowest inference time are `KNeighborsDist` and `KNeighborsUnif`.

The overall best-performing model, `LightGBMLarge`, had the following hyperparameters as reported in Table 6. LightGBM [7] is a gradient boosting framework that uses tree-based learning algorithms to speed up learning time and reduce memory footprint.

## 6.6 Feature importance

We compute feature importance [14] for a particular feature by measuring how much our model's accuracy drops if we randomly shuffle the values for that feature across rows. If the accuracy drops significantly post-shuffle, that particular feature is essential for how the model makes predictions. If the accuracy does not drop, the feature does not help train our model to predict VR motion sickness.

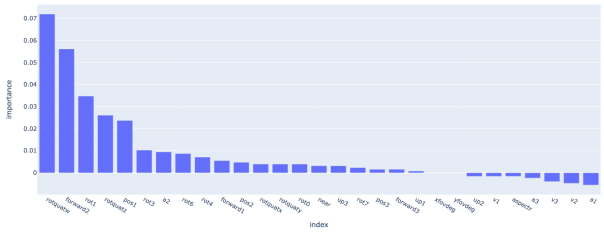


**Figure 17: Feature importance.**

As seen above, the most important features include camera rotation variables, which tells us that these are really important for our model to make good predictions. The least important features include FOV and aspect ratio information, represented by `xfovdeg`, `yfovdeg`, and `aspectr`. This could be because FOV stays mostly the same between games.

# 7 DISCUSSION

*Machine learning generalizability and interpretability.* We got a 92% test accuracy across all the games we tested, which proves that our model can generalize across VR scenarios to some extent, but more testing is needed to see how this model does to games that it hasn't seen before. One possibility is to hold out a game from its training dataset and evaluate its performance on that unseen game. In addition, we use a black-box ML model to make predictions, which delivers good accuracy but little to no explainability. Even the feature importance just tells us which features influence accuracy the most, not how the model uses those features. Future work could look at how to create a more inherently interpretable model.

*Time series analysis.* As mentioned above, participants report their sickness score only 5-10 times during a single game session. Time-series analysis on this dataset requires less sparseness in its per-session sickness data. A naïve solution is to combine data points from multiple sessions. However, time series analysis assumes that a data point recorded at a particular time is in some way semantically related to all the data points recorded before that time. This assumption does not hold if we combine data points from multiple independent gameplay sessions. An easy solution for this problem does not exist; therefore, future work could explore how to perform time-series analysis with sparse sickness data points.

*Scene object analysis.* Additionally, it could be interesting to integrate scene object data that the VR.net dataset supplies into our analysis. However, these files are enormous (on the order of gigabytes). Because we used Google Drive to host the dataset and Google Colab to perform the analysis, we could not use the object data for our correlation analysis or as inputs to our model. With more storage and computing resources, we can determine if this information helps find relevant co-variants with motion sickness.

*Bayesian networks.* Our work was limited to using machine learning for prediction, but it is well established in the literature [5] that for sparse data, traditional Bayesian methods work better than machine learning. It could be interesting to use Bayesian networks, with a node for each feature, to generate predictions for sickness score and compare its performance to ML methods.

# 8 CONCLUSION

We intend this paper to be a primer to the VR.net dataset to aid motion sickness research. In addition to performing exploratory data analysis and demonstrating how to perform feature engineering on this data, we identified several issues with this dataset. We contacted the authors with this information and our recommendations. We also evaluated several ML models' ability to predict the onset of VR motion sickness. We present a model with 92% accuracy and the ability to generalize well across various VR games and scenarios, proving simultaneously that such ML solutions are feasible and that there is much room for improvement in this space. We hope this work serves as a launchpad for future research to predict and mitigate end-user VR-related motion sickness.



# ACKNOWLEDGMENTS

Much thanks to Professor Höllerer for the support and encouragement and a great quarter of 291I.

# REFERENCES

[1] [n. d.]. *Plotly Open Source Graphing Library for Python.* https://github.com/plotly/plotly.py
[2] [n. d.]. *Tabular.* https://auto.gluon.ai/stable/tutorials/tabular/index.html
[3] Frederick Bonato, Andrea Bubka, and Stephen Palmisano. 2009. Combined pitch and roll and cybersickness in a virtual environment. *Aviation, space, and environmental medicine* 80, 11 (2009), 941–945.
[4] Kieran Carnegie and Taehyun Rhee. 2015. Reducing Visual Discomfort with HMDs Using Dynamic Depth of Field. *IEEE Computer Graphics and Applications* 35, 5 (2015), 34–41. https://doi.org/10.1109/MCG.2015.98
[5] Deepgram. [n. d.]. *Bayesian Machine Learning | Deepgram.* https://deepgram.com/ai-glossary/bayesian-machine-learning

[6] H.B.-L. Duh, J.W. Lin, R.V. Kenyon, D.E. Parker, and T.A. Furness. 2001. Effects of field of view on balance in an immersive environment. In *Proceedings IEEE Virtual Reality 2001*. 235–240. https://doi.org/10.1109/VR.2001.913791

[7] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems* 30 (2017).

[8] Behrang Keshavarz, Bernhard E Riecke, Lawrence J Hettinger, and Jennifer L Campos. 2015. Vection and visually induced motion sickness: how are they related? *Frontiers in psychology* 6 (2015), 129781.

[9] Jaekyung Kim, Woojae Kim, Sewoong Ahn, Jinwoo Kim, and Sanghoon Lee. 2018. Virtual Reality Sickness Predictor: Analysis of visual-vestibular conflict and VR contents. In *2018 Tenth International Conference on Quality of Multimedia Experience (QoMEX)*. 1–6. https://doi.org/10.1109/QoMEX.2018.8463413

[10] Jonathan Liebers, Patrick Laskowski, Florian Rademaker, Leon Sabel, Jordan Hoppen, Uwe Gruenefeld, and Stefan Schneegass. 2024. Kinetic Signatures: A Systematic Investigation of Movement-Based User Identification in Virtual Reality. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. 1–19.

[11] J.J.-W. Lin, H.B.L. Duh, D.E. Parker, H. Abi-Rached, and T.A. Furness. 2002. Effects of field of view on presence, enjoyment, memory, and simulator sickness in a virtual environment. In *Proceedings IEEE Virtual Reality 2002*. 164–171. https://doi.org/10.1109/VR.2002.996519

[12] Cheng-Li Liu and Shiaw-Tsyr Uang. 2012. A study of sickness induced within a 3D virtual store and combated with fuzzy control in the elderly. In *2012 9th International Conference on Fuzzy Systems and Knowledge Discovery*. 334–338. https://doi.org/10.1109/FSKD.2012.6234149

[13] Natalie McHugh, Sungchul Jung, Simon Hoermann, and Robert W. Lindeman. 2019. Investigating a Physical Dial as a Measurement Tool for Cybersickness in Virtual Reality. In *Proceedings of the 25th ACM Symposium on Virtual Reality Software and Technology* (Parramatta, NSW, Australia) *(VRST '19)*. Association for Computing Machinery, New York, NY, USA, Article 32, 5 pages. https://doi.org/10.1145/3359996.3364259

[14] Christoph Molnar. [n. d.]. *Permutation Feature Importance*. https://christophm.github.io/interpretable-ml-book/feature-importance.html

[15] Christian Rack, Vivek Nair, Lukas Schach, Felix Foschum, Marcel Roth, and Marc Erich Latoschik. 2024. Navigating the Kinematic Maze: Analyzing, Standardizing and Unifying XR Motion Datasets. In *2024 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. IEEE, 507–514.

[16] D. Rose. [n. d.]. *Rotation Quaternions, and How to Use Them*. https://danceswithcode.net/engineeringnotes/quaternions/quaternions.html

[17] Richard HY So, WT Lo, and Andy TK Ho. 2001. Effects of navigation speed on motion sickness caused by an immersive virtual environment. *Human factors* 43, 3 (2001), 452–461.

[18] Richard H. Y. So, Andy Ho, and W. T. Lo. 2001. A Metric to Quantify Virtual Scene Movement for the Study of Cybersickness: Definition, Implementation, and Verification. *Presence* 10, 2 (2001), 193–215. https://doi.org/10.1162/105474601750216803

[19] Elliott Wen, Chitralekha Gupta, Prasanth Sasikumar, Mark Billinghurst, James Wilmott, Emily Skow, Arindam Dey, and Suranga Nanayakkara. 2024. VR. net: A real-world dataset for virtual reality motion sickness research. *IEEE Transactions on Visualization and Computer Graphics* (2024).