Cheat Prediction and Contrastive Analysis in Games Based on Machine Learning Methods

Xingjie Lin
School of Computer and Software, Dalian Neusoft University of Information, Dalian,116023, China

Keywords: Machine Learning, Game Cheating Detection, Decision Tree, Logistic Regression, Random Forest

Abstract: The research topic of this study is cheat prediction in games and its importance lies in maintaining a fair and enjoyable gaming environment. The main idea of this article is exploring machine learning models of game cheating detection. Specifically, the focus of this study is on the criteria and search strategies for analyzing the four models. Machine learning has significant advantages in generalization of data fitting. The implicit relationships between data are further inferred and represented. The comparative analysis of those models revealed that Logistic Regression model outperformed others with 90% accuracy. The results show that Logistic Regression has competitive performance due to its unique features. This study provides valuable insights for game developers and researchers to develop effective cheat detection systems. The future prospects and directions for cheat detection research are also discussed. Overall, this research contributes to the advancement of cheat detection in games, which can promote fairness and trust in the gaming community.

1 INTRODUCTION

In recent years, due to the rapid developing of machine learning, significant progress has been made in research in the area of artificial intelligence. Machine learning enables computers to study and create predictions and decisions lines from different data. This technology is essential for analyzing large datasets and identifying patterns, making it increasingly important for solving complex problems in domains such as healthcare, finance, and transportation. Many researchers have explored different machine learning algorithms, they use different models to solve various problems in fields (Pinto et al. 2021).

The target of the study is analyzing the performances of different machine learning algorithms in monitoring game cheating. Relevant features will be extracted from game data to identify patterns indicative of cheating behavior. Various machine learning models will be experimented with to build predictive models that can accurately classify cheating instances. The performance of different models will be compared and analyzed to determine their strengths, weaknesses, and suitability for real-world game cheating detection scenarios (Davar et al. 2022). Game cheat detection is an important issue that can significantly impact the fairness and integrity of the gaming experience. This study explores the implications and practical applications of using machine learning to enhance the overall gaming experience for players. This study introduces four machine learning models (Zeng 1996, Zhang et al. 2021, Wang et al. 2016). Anomaly detection and pattern recognition are also discussed as useful techniques for detecting cheating behavior. Implementing cheat detection through neural networks is also explored in this study, covering topics such as data collection and preprocessing, construction of neural network models, feature engineering, sample labeling, model training and optimization, as well as model evaluation and deployment (Zhang 2016, Wang 2014). Additionally, the issue of imbalance in game cheat detection and artificial intelligence is addressed, discussing various methods such as imbalanced data collection, sample resampling, class weight adjustment, anomaly detection techniques, and ensemble learning (Lu 2007, Fang et al. 2011, Che & Geng 2013).

The following is the layout of this article in part 2, a comprehensive analysis and discussion were conducted on the different roles and ways in which

Lin, X.
Cheat Prediction and Contrastive Analysis in Games Based on Machine Learning Methods.
DOI: 10.5220/0012821100004547

In *Proceedings of the 1st International Conference on Data Science and Engineering (ICDSE 2024)*, pages 464-469
ISBN: 978-989-758-690-3

four different models, play in cheating detection in games. The strengths and limitations of each model are examined, highlighting their effectiveness in identifying cheating behaviors. In part 3, the results of the analysis are presented and discussed. The findings explain the capabilities of different models and their respective contributions to cheat detection. Comparisons are made between the models, examining their performance, accuracy, and efficiency in detecting cheating behaviors. Part 4 summarizes the conclusions and future prospects of this study. The importance of implementing robust cheat detection systems in maintaining fairness and integrity within the gaming community is reiterated. Furthermore, future prospects and directions for research in cheat detection are outlined, emphasizing the need for continuous updates and improvements to combat evolving cheating tactics.

2 METHODOLOGY

The general process for game cheat detection involves several steps, as shown in the Figure 1. First, relevant player behavior data and features are collected, such as game time, score, and action frequency. The next step is to model the probability of a player cheating based on these features. Logistic regression calculates the likelihood of cheating, while decision trees partition the feature space to classify players as cheaters or non-cheaters. To improve accuracy and robustness, random forests can be employed by combining predictions from multiple decision trees. Another approach is to utilize Support Vector Machines (SVMs), which consider player behavior features like game command sequences and action frequencies to classify players into cheat or non-cheat categories. Lastly, performance analysis is conducted to evaluate the effectiveness of these methods in detecting and classifying cheating behaviors in games.

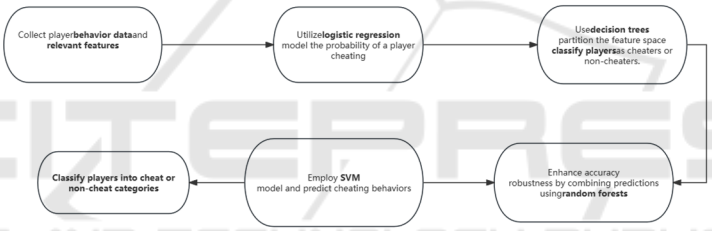

Figure 1: The process of this study (Photo/Picture credit: Original).

2.1 Logistic Regression for Game Cheat Detection

Logistic regression is a widely used model that can be used to detect cheating behavior in games. In this model, the probability of a player cheating is estimated based on their behavior and relevant features. This is a classification algorithm that detects whether players are cheating (Che & Geng 2013, Ciocca et al. 2015).

In the context of game cheat detection, logistic regression analyzes various behavioral aspects of players, such as game time, score, and action frequency. These features are used to build a mathematical model that calculates the likelihood of a player engaging in cheating activities. The model assigns a probability value between 0 and 1, where a value close to 1 indicates a higher probability of user cheating.

To apply logistic regression in game cheat detection, a training dataset is collected, consisting of labeled instances where cheating behaviors are known. The model learns from this dataset to identify and judge the relationship between player's own behavior and cheating behavior. As long as it is trained, the model can be used to predict the cheating probability of new instances (Murata et al. 2016).

When a player engages in the game, their behavior is observed and fed into the logistic regression model. Based on the calculated probability, the model can classify the player as either a potential cheater or a non-cheater. This information can then be used by game developers or administrators to take appropriate actions, such as implementing additional monitoring or enforcing penalties.

Overall, logistic regression plays a vital role in game cheat detection by using player behavior and relevant features to estimate the probability of cheating. Its ability to classify players based on cheating likelihood assists in maintaining fair

gameplay and preserving the integrity of the gaming environment.

2.2 Decision Tree for Game Cheat Detection

Decision trees are widely used in cheat detection for games. Decision tree is an algorithm that divides the feature space based on the player's behavioral characteristics, such as game duration, score, and operation frequency. It creates a hierarchical structure of decisions and outcomes, which enables the classification of players as cheaters or non-cheaters (Shenoy 1998).

In the application of game cheating detection, decision trees analyze player behavior by evaluating different characteristic behaviors. This algorithm starts from the root node of the entire dataset and recursively segments the data based on the selected features. Each segmentation node is selected by finding features and thresholds that better classify instances into categories of cheaters and non-cheaters.

As the decision tree grows, it creates branches and leaves that represent different decisions and outcomes. For instance, if the game time exceeds a certain threshold and the score is unusually high, the decision tree may classify the player as a potential cheater. On the other hand, if the frequency of actions is within expected limits, the player may be classified as a non-cheater (Hailemariam et al. 2011).

To apply decision trees in cheat detection, a training dataset is required, consisting of labeled instances where cheating behaviors are known. The decision tree algorithm learns from this dataset to construct an optimal tree structure that maximizes the accuracy of classification. Once trained, the decision tree can classify new instances and use it to determine whether players may cheat.

The path from the root to the leaf in the decision tree shows all if else conditions that lead to classification results, providing interpretable results. This allows game developers or administrators to gain insights into the specific behaviors that indicate cheating and take appropriate actions accordingly.

In summary, decision trees are a valuable tool for cheat detection in games. By partitioning the feature space based on player behavior features, decision trees can classify players as cheaters or non-cheaters, providing actionable information for maintaining fair gameplay and ensuring the integrity of the gaming environment.

2.3 Random Forest for Game Cheat Detection

Random forests are a powerful technique used in cheat detection for games. Random forest integrates multiple decision trees to ensure the accuracy of its data and robustness of cheating detection models. By aggregating predictions from individual trees, better performance can be achieved in identifying cheating behavior compared to decision trees (Zhao et al. 2018).

In the application of game cheating detection, random forest consists of serious of decision trees, everyone of that is trained on different subsets of relevant data. The trees in the random forest are constructed by randomly selecting features and data samples. This randomness helps to decorrelate the trees and reduce overfitting.

When a new instance needs to be classified, each tree in the random forest independently predicts whether the player is a cheater or a non-cheater. The final classification is determined by synthesizing predictions from various trees. The majority of voting items in the decision tree determine the final classification of the user.

Random forests offer several advantages in cheat detection. Firstly, they improve the accuracy of predictions by reducing the variance associated with individual decision trees. The ensemble of trees helps to capture a more comprehensive range of cheating behaviors and generalizes well to unseen instances. Secondly, random forests are robust to irrelevant features such as noise. They can process a large number of features without overfitting, making them more suitable for complex cheating detection scenarios (Liu et al. 2012).

In practical applications, game developers or administrators can utilize random forests to detect cheating behaviors in real-time. By continuously monitoring player behavior and feeding it into the random forest model, cheating instances can be identified promptly. Appropriate actions, such as issuing warnings or applying penalties, can then be taken to maintain fair gameplay and protect the gaming environment.

In conclusion, random forests are a valuable tool in cheat detection for games. By integrating predictions from multiple decision trees, Random Forest enhances the accuracy of cheating detection, effectively identifies cheating behavior, and ensures a gaming experience for players.

2.4 SVM for Game Cheat Detection

Support vector machines can be used to model and detect cheating behavior in games. SVMs are one of the supervised learning methods used in classification, regression, or other tasks. In the context of cheat detection, SVMs help classify players into cheat or non-cheat categories by considering player behavior features such as game command sequences and action frequencies (Ferretti 2008).

The basic idea behind SVMs is to find the optimal decision boundary which distinguishes the two sets of data with the maximum margin. The training of SVM maps input data points to a high-dimensional space, where hyperplanes can be constructed to separate these two classes. A hyperplane is a decision boundary that maximize margin. The point closest to the hyperplane is the support vector, commonly used to determine the position of the hyperplane. The goal of SVM is to minimize classification errors while maximizing margin.

The SVM model learns from this dataset to construct an optimal decision boundary that classifies new instances into cheat or non-cheat categories based on their behavior features. For example, if a player's game command sequence is significantly different from that of other players, the SVM model may classify the player as a potential cheater.

SVMs have several advantages in cheat detection. Firstly, they can handle high-dimensional data and work well with small datasets. Secondly, they are effective in identifying complex cheating behaviors that involve multiple variables. SVMs are also robust to noisy or irrelevant features, making them suitable for real-world applications (Ferretti 2009).

In practical applications, game developers or administrators can use SVMs to monitor player behavior in real-time and identify potential cheaters. By continuously feeding player behavior features into the SVM model, cheating instances can be detected promptly, and appropriate actions can be taken to maintain fair gameplay and protect the gaming environment.

In conclusion, SVMs are a valuable tool in cheat detection for games. By considering player behavior features and constructing an optimal decision boundary, SVMs can classify players into cheat or non-cheat categories, enabling effective identification of cheating behaviors and maintaining the integrity of the gaming experience.

3 RESULT AND DISCUSSION

This chapter provides machine learning techniques for deep analysis of cheating detection in games. It aims to equip game developers with practical insights and guidelines to improve the integrity and fairness of their games while enhancing the gaming experience for players.

This chapter first investigates models that is well used for game cheating detection. Those models were discussed in detail. Each technique is explained in terms of its strengths, weaknesses, and suitability for cheat detection in games.

In addition to machine learning techniques, the chapter explores various types of variables that can be considered in cheat detection. These variables encompass both discrete and continuous aspects related to game actions, data, events, player input behaviors, and network data. Understanding these variables provides valuable insights into player behavior patterns, enabling the identification of anomalies indicative of cheating.

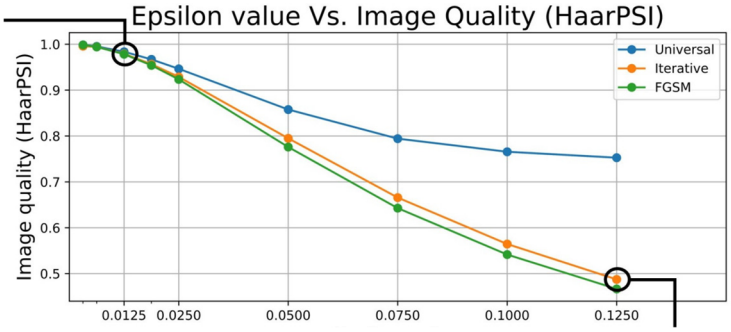

Figure 2: Using HaarPSI measurement ϵ the impact on perceived image quality (Aditya et al. 2013)

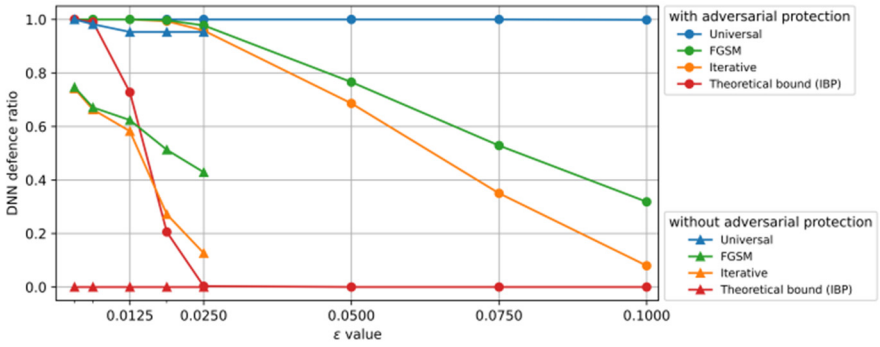

Figure 3: Using different cost functions to train the numerical values of the same DNN architecture (Aditya et al. 2013)

The above two figures represent a series of research and data analysis on the topic of "Machine Learning-Based Game Cheating Analysis" published by Aditya Jonnalagadda et al. on March 18, 2021 (Aditya et al. 2013).

The researchers discuss the measurement of true positives in detecting cheating frames and how it is affected by adversarial attacks. It compares the performance of a deep neural network (DNN) detector trained with different loss functions: one without any protection against attacks and another with protection using uncertainty loss and interval bound propagation (IBP). The ratio of true positives preserved under attack is used as a representative of the network's defense ability (compare with Figure 2).

Researchers investigated the effects of loss function and IBP on the performance of DNN under adversarial attacks through experiments. This experiment was conducted in ε Under consistent conditions, aiming to balance attack intensity and frame quality. The results indicate that without IBP, the entire dataset may be vulnerable to attacks (compare with Figure 3).

Furthermore, the chapter offers a comprehensive discussion on the implementation of cheat detection using neural networks. It covers crucial steps such as data collection and preprocessing, building the neural network model, feature engineering, labeling samples, model training and optimization, and model evaluation and deployment. This detailed walkthrough ensures game developers have a clear understanding of the entire process.

Addressing the issue of class imbalance in cheat detection is another key focus of the chapter. The imbalance between cheating and non-cheating instances can pose challenges in accurately detecting cheating behaviors. To overcome this, the chapter presents various solutions, including imbalanced data collection, sample resampling, class weight adjustment, anomaly detection approaches, and ensemble learning. These strategies assist in improving the overall performance of cheat detection models.

In conclusion, this chapter provides a comprehensive overview of cheat detection in games, offering practical insights into machine learning techniques and data analysis methods. By implementing these guidelines, game developers can effectively identify and deter cheating behaviors, ensuring fairness and integrity within their games. However, it is important to acknowledge that cheat detection is an ongoing battle, as cheaters continually develop new tactics. Therefore, continuous research, monitoring, and updates to cheat detection algorithms are crucial to stay ahead of emerging cheating techniques.

By leveraging the knowledge presented in this chapter, game developers can establish robust cheat detection systems that promote a more enjoyable and equitable gaming environment for all players. Through constant vigilance and adaptation, developers can maintain the integrity of their games, fostering a positive gaming community.

4 CONCLUSION

In summary, the use of machine learning technology for cheating detection in game management is crucial to ensuring fairness and honesty in the game. These four models are frequently used approaches that can effectively identify cheating behaviors. By analyzing various relevant variables in the game, developers can obtain valuable information on the player's behavior patterns and operating habits, thereby accurately detecting cheating situations.

To implement cheat detection using neural networks, developers must follow several crucial

steps, including data collection and preprocessing, model building, feature engineering, sample labeling, model training and optimization, and evaluation and deployment. Addressing the issue of class imbalance is also important, as it poses challenges in accurately detecting cheating behaviors. Some solutions such as imbalanced data collection, sample resampling, class weight adjustment, anomaly detection methods, etc. can also help improve the overall performance of game cheating detection models. However, it is important to acknowledge that cheat detection is a continuous battle, as cheaters continually develop new tactics. Therefore, continuous research, monitoring, and updates to cheat detection algorithms are crucial to stay ahead of emerging cheating techniques. In conclusion, game developers can leverage the insights presented in this study to enhance the gaming experience for all players. By implementing effective cheat detection systems, developers can maintain the fairness and integrity of their games, fostering a positive gaming community. The next step for developers is to continue refining and improving cheat detection algorithms to stay ahead of evolving cheating tactics and ensure a fair gaming environment for all players.

REFERENCES

J. P. Pinto, A. Pimenta, P. Novais, Machine Learning (2021) pp. 3037–3057.

G. Davar, S. Ashkan, T. Hadis, S. M. Ali, Concurrency and computation: practice and experience (2022) pp. e6533.1-e6533.17

P. Zeng, Zeitschrift fur Angewandte Mathematik und Mechanik (1996) pp. 565-566.

C. Zhang, Y. Guo, M. Li, Computer Engineering and Applications (2021) pp. 57-69.

L. Wang, M. Han, X. J. Li, et. al, Computer Engineering and Applications (2021) pp. 42-52.

S. Zhang, Journal of Jiangsu Polytechnic University (2016) pp. 14-17.

X. Wang, Science and Technology Research (2014) pp. 575-575.

H. Lu, Journal of Suzhou University, (2007) p. 5.

K. Fang, J. Wu, J. Zhu, et al, Statistics and Information Forum (2011) p. 7.

L. Che, L. Geng, University of Electronic Science and Technology of China (2013).

G. Ciocca, C. Cusano, R. Schettini, Multimed Tools Appl (2015) pp. 3013–3034.

A. Murata, Y. Ohta, M. Moriwaka, Advances in Intelligent Systems and Computing (2016).

P. P. Shenoy, Theory and Decision (1998) pp. 149–171.

E. Hailemariam, R. Goldstein, R. Attar, A. Khan, Spring Simulation Multiconference (2011).

Z. Zhao, K. G. Mehrotra, C. K. Mohan, Recent Trends and Future Technology in Applied Intelligence (2018).

Y. Liu, Y. Wang, J. Zhang, Information Computing and Applications (2012).

S. Ferretti, Multimed Tools Appl (2008) pp. 339–363.

S. Ferretti, Handbook of Multimedia for Digital Entertainment and Arts (2009).

J. Aditya， F. Iuri, S. Seth et. al, arXiv:2103 (2013).