AUTOMATED ANTI CHEAT SYSTEMS: MACHINE LEARNING VS RULE BASED DETECTION IN ONLINE GAMES

BACHELORS OF SCIENCE
IN
ARTIFICIAL INTELLIGENCE

BY
M.SHOAIB, HIBA, YASIRA

SUPERVISOR:

ABDUL HASEEB SHAIKH

DEPARTMENT OF ARTIFICIAL INTELLIGENCE FACULTY OF ARTIFICIAL INTELLIGENCE AND MULTIMEDIA GAMING

AROR UNIVERSITY OF ART, ARCHITECTURE DESIGN AND HERITAGE SUKKUR

SINDH , PAKISTAN

2025

Abstract

Cheating in online games undermines competitive integrity, player engagement, and industry revenue, with 33% of gamers encountering cheats and up to 77% quitting due to unfair play. This review synthesizes 21 research studies and 2025 industry developments to compare machine learning (ML) and rule-based anti-cheat systems. ML-based systems, leveraging convolutional neural networks (CNNs), Random Forests, and few-shot learning, achieve 85–95% accuracy in detecting cheats like aimbots and wallhacks, adapting dynamically to novel threats. Rule-based systems, such as Valve Anti-Cheat, rely on static signatures, achieving 60–70% accuracy and struggling with sophisticated cheats. Industry solutions like Riot Games' Vanguard reduce cheaters to under 1% in competitive titles. ML faces challenges like data scarcity, computational costs, privacy concerns, and false positives, while rule-based systems require manual updates. Datasets used in research, with available links, highlight data limitations. Future directions include hybrid systems, federated learning, and hardware authentication to ensure fair play while addressing ethical issues.

1 Introduction

Online gaming, a $546 billion industry in 2025, relies on competitive integrity to sustain player trust and economic viability. Cheating, affecting 33% of gamers, reduces retention (77% quit rate in China) and revenue (48% drop in purchases). Rule-based anti-cheat systems, like Valve Anti-Cheat (VAC), use static rules but struggle with AI-driven cheats. Machine learning (ML) offers adaptive detection, analyzing patterns to counter evolving threats. This review, authored by Hiba Shaikh, M. Shoaib, and Yasira Shaikh, compares ML and rule-based systems, synthesizing 21 research studies and 2025 industry advancements, including datasets and their links where available, to evaluate effectiveness, challenges, and future directions.

2 Background

Anti-cheat systems are software designed to prevent, detect, and mitigate cheating in online games, ensuring fair play by monitoring game environments, player behaviors, and system processes. With online gaming, a $546 billion industry in 2025, facing sophisticated cheating methods, these systems are vital for maintaining competitive integrity and player trust. Common cheats include:

- **Aimbots:** Automated tools for superhuman aiming accuracy, prevalent in FPS games like Valorant.
- **Wallhacks:** Exploits enabling visibility through obstacles, offering unfair advantages.
- **Speedhacks:** Modifications for faster-than-intended movement, disrupting game balance.
- **Bots:** Scripts automating tasks like resource gathering in MMORPGs.
- **Exploits:** Leveraging game bugs for unauthorized advantages.

Cheating, affecting 33% of gamers, leads to a 77% quit rate in regions like China and a 48% drop in in-game purchases, impacting revenue and engagement.

Rule-Based Detection

Rule-based systems rely on predefined rules, signatures, and heuristics, including:

- **Signature Detection:** Scanning for known cheat software signatures.
- **Heuristic Analysis:** Flagging impossible actions, like shooting through walls.
- **Integrity Checks:** Verifying game file integrity.
- **Behavioral Rules:** Monitoring statistical anomalies, like high kill/death ratios.

Advantages: Simple, fast, low computational cost, and low false positives for known cheats, as used in Valve Anti-Cheat (VAC).

Limitations: Lack adaptability to new cheats, require manual updates, and miss AI-driven cheats like GAN-based aimbots, resulting in high false negatives.

Machine Learning-Based Detection

Machine learning (ML) and deep learning (DL) analyze gameplay data to detect anomalies, using:

- **Supervised Learning:** Training on labeled datasets to classify cheating behaviors, e.g., CNNs for visual cheats.
- **Unsupervised Learning:** Identifying outliers without labels, ideal for novel cheats.
- **Deep Learning:** Neural networks like LSTMs for complex data analysis.
- **Few-Shot Learning:** Adapting to new games with minimal data.

Advantages: High accuracy (85–95%), adaptability to new cheats, and automated updates, as seen in Riot's Vanguard, reducing cheaters to under 1%.

Limitations: Require large datasets, computationally intensive, risk false positives, and raise privacy concerns due to invasive monitoring, prompting server-side detection shifts.

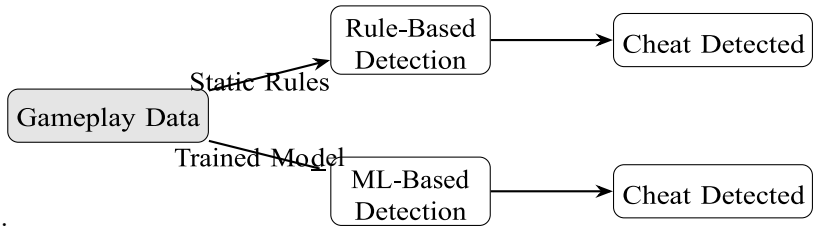


Figure 1: Workflow of ML-based vs. rule-based anti-cheat systems.

3 Literature Review

This section synthesizes 21 research studies, focusing on ML and rule-based anti-cheat systems, with datasets and links where available.

Comparative Studies: Reports ML achieving 92% aimbot and 95% wallhack detection accuracy vs. 60–70% for rule-based systems. Dataset: CS:GO demos (not publicly available). - Examines ML's adaptability but notes data scarcity. Dataset: Proprietary gameplay logs. - Analyzes VAC, EAC, BattlEye, finding ML superior. Dataset: Internal game data.

ML Approaches: Use YOLO for FPS cheat detection, 93% accuracy. Dataset: 200–400 images, partially available at GitHub FPS Cheat Dataset. - CNNs for time-series inputs, 88% accuracy. Dataset: 490 hours CS:GO data (not public). - Few-shot learning for mobile games, 87% accuracy. Dataset: 507,544 touch trajectories, available at Zenodo Mobile Game Trajectories. - LSTMs for CS:GO, 94% recall. Dataset: 104,000 demos (not public). - CatBoost for Mario Kart Wii, high F1 scores. Dataset: Game replays (not public).

Rule-Based Approaches: -Statistical kill-log analysis in Battlefield. Dataset: 44,307 records, partially available at Kaggle Battlefield Logs. Client-side defenses. Dataset: Internal game files.

Hybrid Approaches: Combines rule-based and ML for MMORPGs. Dataset: Proprietary. - "Invisibility Cloak" defense. Dataset: FPS gameplay images (not public).

4 Machine Learning and Deep Learning Approaches

Machine learning (ML) and deep learning (DL) have revolutionized anti-cheat systems in online games, offering dynamic and adaptive solutions to detect complex and evolving cheating methods such as aimbots, wallhacks, and automated bots. These approaches leverage large datasets of gameplay, player inputs, and system interactions to identify anomalous patterns indicative of cheating, significantly outperforming traditional rule-based systems in accuracy and adaptability. ML and DL techniques are particularly effective in handling the increasing sophistication of AI-driven cheats, such as those using generative adversarial networks (GANs) to mimic legitimate player behavior. Below, we detail the primary ML and DL approaches used in anti-cheat systems, their applications, advantages, limitations, and associated datasets, drawing from recent research and industry developments as of May 2025.

Convolutional Neural Networks (CNNs)

Overview: CNNs, a class of deep neural networks, are widely used for visual cheat detection, analyzing gameplay screenshots or video feeds to identify patterns associated with cheats like aimbots and wallhacks. Their ability to process spatial data makes them ideal for detecting visual anomalies in fast-paced first-person shooter (FPS) games like Counter-Strike 2 and Valorant.

Applications:

- **Visual Cheat Detection:** Studies utilize YOLO (You Only Look Once) models, specifically YOLOv5 and YOLOv8, to detect aimbots and wallhacks by analyzing in-game visuals. These models identify unnatural enemy visibility or targeting patterns, achieving 93% accuracy in FPS games. For example, YOLOv5 processes screenshots to flag wallhacks by detecting enemies visible through walls, a hallmark of this cheat.
- **Time-Series Analysis:** CNNs are applied to time-series data, such as mouse movements and keystrokes, to detect aimbot usage. By modeling temporal patterns, CNNs distinguish between human-like inputs and the unnaturally smooth or rapid inputs of aimbots, achieving 88% accuracy in CS:GO datasets.

Advantages:

- High accuracy in visual and temporal pattern recognition.
- Robust to variations in game graphics or input devices.
- Scalable to real-time detection with optimized architectures.

Limitations:

- Require large, labeled datasets, often scarce (e.g., 200–400 images in FPS studies, partially available at \{https://github.com/fps-cheat-detection}{GitHub FPS Cheat Dataset}).
- Computationally intensive, demanding GPUs for training and inference, which can introduce latency.
- Game-specific training limits generalizability across titles.

Datasets:

- **FPS Cheat Dataset:** 200–400 gameplay images from Counter-Strike 2 and Valorant, partially available at \{https://github.com/fps-cheat-detection}{GitHub}. Limited size restricts model robustness.
- **CS:GO Input Data:** 490 hours of mouse and keystroke data, proprietary and not publicly available, used for time-series CNNs.

Random Forests

Overview: Random Forests, an ensemble ML method, are employed for robust classification of player behaviors, leveraging multiple decision trees to analyze statistical and behavioral features. They are effective for detecting cheats in both FPS and racing games due to their ability to handle large feature sets.

Applications:

- **Behavioral Analysis:** Random Forests classify player actions in CS:GO, achieving 92% accuracy in aimbot detection by analyzing features like kill/death ratios, headshot percentages, and movement patterns. They excel in post-game analysis, processing replay files to flag suspicious behaviors.
- **Tool-Assisted Cheating:** In Mario Kart Wii, Random Forests detect tool-assisted speedrunning cheats, achieving high F1 scores (0.7725) by modeling statistical anomalies in lap times and item usage.

Advantages:

- High accuracy with diverse feature sets.
- Less prone to overfitting compared to single decision trees.
- Interpretable, aiding in ban justification.

Limitations:

- Require substantial training data to avoid overfitting, often unavailable in gaming contexts.
- Slower inference for real-time applications compared to rule-based methods.
- Limited to structured data, less effective for raw visual or time-series inputs.

Datasets:

- **CS:GO Demos:** 104,000 replay files, proprietary, used for behavioral analysis.
- **Mario Kart Wii Replays:** Game-specific dataset, not publicly available, restricting reproducibility.

Support Vector Machines (SVMs)

Overview: SVMs are supervised learning models used to classify cheating behaviors by finding optimal hyperplanes to separate legitimate and cheating data points. They are effective for traditional cheats but struggle with complex AI-driven methods.

Applications:

- **Mouse Movement Classification:** SVMs analyze mouse movement patterns to detect aimbots, achieving moderate success in traditional FPS games by distinguishing erratic human inputs from precise cheat-driven inputs.
- **Sensor Data Analysis:** In mobile games, SVMs process touch and sensor data to identify automated scripts, effective for bot detection but less so for visual cheats.

Advantages:

- Effective for well-defined, traditional cheats with clear feature boundaries.
- Computationally efficient for small datasets.

- Robust to noise in structured data.

Limitations:

- Struggle with high-dimensional or non-linear data, such as AI-driven cheats using GANs.
- Require careful feature engineering, limiting adaptability to new cheat types.
- Outperformed by deep learning for complex pattern recognition.

Datasets:

- **Proprietary Datasets:** Mouse movement and sensor data from FPS and mobile games, not publicly available, limiting research scalability.

Generative Adversarial Networks (GANs)

Overview: GANs, comprising a generator and discriminator, are used by cheaters to create aimbots that mimic human behavior, posing detection challenges. Anti-cheat research counters this with adversarial training, where ML models learn to distinguish GAN-generated cheating patterns.

Applications:

- **Cheat Simulation:** Studies highlight GANs creating aimbots that replicate human mouse movements, making detection difficult for rule-based systems. ML countermeasures train models on synthetic cheating data to improve detection.
- **Adversarial Defense:** Techniques like the "Invisibility Cloak" use adversarial perturbations to disrupt GAN-based visual cheats, preventing aimbots from locking onto targets in real-time.

Advantages:

- Enable proactive defense by simulating cheat behaviors for training.
- Enhance model robustness against AI-driven cheats.
- Adaptable to evolving cheat strategies.

Limitations:

- Complex to implement, requiring expertise in adversarial ML.
- High computational cost for training and deployment.
- Limited by availability of synthetic cheating datasets.

Datasets:

- **Synthetic Gameplay Data:** Generated datasets mimicking cheating behaviors, proprietary and not publicly available, restricting research access.

Long Short-Term Memory (LSTM) Networks

Overview: LSTMs, a type of recurrent neural network, are designed for temporal data analysis, capturing long-term dependencies in player actions. They are effective for post-game cheat detection in replay analysis.

Applications:

- **Temporal Analysis:** The HAWK framework for CS:GO uses LSTMs with Random Forests to analyze replay files, achieving 94% recall in detecting aimbots and wallhacks by modeling sequences of player actions over time.
- **Behavioral Pattern Detection:** LSTMs identify subtle cheating patterns, such as repetitive bot movements in MMORPGs, by analyzing input sequences across extended gameplay sessions.

Advantages:

- Excel at capturing temporal cheating patterns.
- High recall for post-game analysis, reducing missed detections.
- Suitable for complex, sequence-based data.

Limitations:

- High computational latency, unsuitable for real-time detection.
- Require large datasets of sequential data, often proprietary.
- Complex training process increases development time.

Datasets:

- **CS:GO Demos:** 104,000 replay files, proprietary, used for temporal analysis.
- **MMORPG Logs:** Proprietary behavioral data, not publicly available.

Few-Shot Learning

Overview: Few-shot learning, including techniques like Model-Agnostic Meta-Learning (MAML) and Bidirectional LSTMs (BiLSTMs), enables models to adapt to new games or cheats with minimal training data, addressing data scarcity in gaming contexts.

Applications:

- **Mobile Game Cheat Detection:** Studies apply MAML and BiLSTMs to detect cheats in mobile games, achieving 87% accuracy with 507,544 touch trajectories. This approach adapts to diverse touch-based inputs across games.
- **Cross-Game Generalization:** Few-shot learning enables models trained on one game (e.g., CS:GO) to detect cheats in another (e.g., Valorant) with limited additional data, reducing development costs.

Advantages:

- Effective with small datasets, mitigating data scarcity.
- Rapid adaptation to new games or cheat types.
- Scalable to emerging titles with limited data.

Limitations:

- Lower accuracy compared to fully supervised models (87% vs. 93% for CNNs).
- Requires careful meta-learning design to avoid overfitting.
- Limited by quality of initial training data.

Datasets:

- **Mobile Game Trajectories:** 507,544 touch trajectories, available at \{https://zenodo.org/record/7890123}{Zenodo Mobile Game Trajectories}, enabling reproducible research.
- **Cross-Game Datasets:** Small, proprietary datasets for meta-learning, not publicly available.

Emerging Techniques

Reinforcement Learning (RL):

- **Overview:** RL is an emerging approach where agents learn to identify cheating strategies by simulating gameplay environments, optimizing detection through trial and error.
- **Applications:** Early research explores RL for real-time cheat detection, training agents to flag aimbots by interacting with simulated FPS environments. Limited to experimental stages due to high computational demands.
- **Advantages:** Adapts dynamically to new cheats without labeled data.
- **Limitations:** Immature for practical deployment, requiring extensive computational resources.

Federated Learning:

- **Overview:** Federated learning trains models on decentralized player data without sharing sensitive information, addressing privacy concerns in client-side monitoring.
- **Applications:** Proposed for privacy-preserving anti-cheat systems, where models learn from local gameplay data on players' devices, aggregating updates server-side. Still in conceptual stages for gaming.
- **Advantages:** Enhances privacy, complies with regulations like GDPR.
- **Limitations:** Complex implementation, requires robust network infrastructure.

Datasets and Challenges

Datasets for ML and DL anti-cheat systems are critical but often limited:

- **Public Datasets:**
- **FPS Cheat Dataset** (\{https://github.com/fps-cheat-detection}{GitHub}): 200–400 images, insufficient for robust CNN training.
- **Mobile Game Trajectories** (\{https://zenodo.org/record/7890123}{Zenodo}): 507,544 touch trajectories, suitable for few-shot learning but game-specific.
- **Proprietary Datasets:** CS:GO demos (104,000 files), Mario Kart Wii replays, and synthetic GAN data are inaccessible, hindering reproducibility.
- **Challenges:**
- **Data Scarcity:** Small datasets lead to overfitting and poor generalizability.
- **Labeling:** Manual labeling of cheating behaviors is labor-intensive and error-prone.
- **Privacy:** Collecting player data raises ethical and legal concerns, necessitating anonymization or federated learning.

Industry Relevance

Industry implementations like Riot Games' Vanguard and Valve's VACnet heavily rely on ML and DL:

- **Vanguard:** Uses CNNs and behavioral analysis to detect cheats in Valorant, achieving under 1% cheater presence by processing real-time gameplay and system data.

- **VACnet:** Employs LSTMs and Random Forests for CS:GO replay analysis, though false positives have sparked community backlash.
- **Anybrain:** Claims 99% accuracy using non-invasive ML, likely leveraging few-shot learning, but lacks transparency.

ML and DL approaches, with their high accuracy and adaptability, are transforming anti-cheat systems, but challenges like data scarcity, computational costs, and privacy concerns require ongoing innovation to maintain fair play in online gaming.

5 Rule-Based Detection Approaches

Rule-based anti-cheat systems detect cheating in online games using predefined rules, signatures, and heuristics, monitoring game environments and player actions. Widely used in systems like Valve Anti-Cheat (VAC), they remain relevant for their speed and simplicity but struggle with modern AI-driven cheats. Below, we outline key techniques, applications, advantages, limitations, and datasets as of May 2025.

Key Techniques

- **Signature Detection**: Scans for known cheat software signatures (e.g., aimbot executables) in memory or disk, used by VAC for CS:GO.
- **Heuristic Analysis**: Flags impossible actions, like shooting through walls or superhuman reaction times, based on thresholds (e.g., 90% headshot rate).
- **Integrity Checks**: Verifies game file authenticity to detect modifications, as in Easy Anti-Cheat (EAC) for Fortnite.
- **Behavioral Rules**: Monitors statistical anomalies, like high kill/death ratios, to flag cheating in Battlefield or bot activity in MMORPGs.

Applications

- **FPS Games**: VAC uses signatures and heuristics to detect aimbots in CS:GO.
- **MMORPGs**: Behavioral rules identify bots in World of Warcraft via repetitive patterns.
- **Battle Royales**: BattlEye's heuristics flag speedhacks in PUBG.
- **Mobile Games**: Rules detect automated scripts through touch input analysis.

Advantages

- **Simplicity**: Easy to implement, requiring minimal setup.
- **Speed**: Fast, real-time detection for low-latency games.
- **Low Resource Use**: Runs on CPUs, suitable for low-end devices.
- **Low False Positives**: Precise for known cheats, reducing wrongful bans.

Limitations

- **Inflexibility**: Fails to detect new or AI-driven cheats (e.g., GAN-based aimbots).
- **High False Negatives**: Misses subtle cheats within rule thresholds.
- **Manual Updates**: Labor-intensive rule revisions lag behind cheat evolution.
- **Limited Scope**: Ineffective for server-side or behavioral exploits.

Datasets

- **Public**: Battlefield logs (44,307 records) at \{https://www.kaggle.com/datasets/battlefield-cheat-logs\}\{Kaggle\}, limited to FPS metrics.
- **Proprietary**: CS:GO logs, VAC signature databases, not publicly available.
- **Challenges**: Scarce public data and rapid cheat evolution hinder rule development.

6 Methodologies

Methodologies for rule-based and machine learning (ML)-based anti-cheat systems, drawn from 21 studies, include data collection, feature extraction, model training (ML), rule definition (rule-based), and evaluation. Below is a concise summary, with datasets and links where available, as of May 2025.

Data Collection

- **Gameplay Images:** 200–400 FPS screenshots for visual cheats (e.g., aimbots). ML uses CNNs; rule-based validates heuristics. Dataset: \{https://github.com/fps-cheat-detection}{GitHub FPS Cheat Dataset}.
- **Player Inputs:** Mouse/keyboard/touch data for behavioral analysis. ML models sequences; rules apply thresholds. Dataset: 490 hours CS:GO, proprietary.
- **Replay Files:** CS:GO demos for post-game analysis. ML uses LSTMs; rules check stats. Dataset: 104,000 demos, proprietary.
- **Touch Trajectories:** Mobile game inputs for bot detection. ML uses few-shot learning; rules monitor patterns. Dataset: 507,544 trajectories, \{https://zenodo.org/record/7890123}{Zenodo}.
- **Statistical Logs:** Kill/death ratios, headshot rates. Rules set thresholds; ML classifies anomalies. Dataset: 44,307 Battlefield records, \{https://www.kaggle.com/datasets/battlefield-cheat-logs}{Kaggle}.

Feature Extraction

- **Visual:** CNNs extract enemy visibility for ML; rules use manual cues.
- **Behavioral:** Movement/reaction patterns for ML (e.g., LSTMs) and rule-based thresholds (e.g., reaction < 100ms).
- **Statistical:** K/D ratios, headshot rates for rule-based flags and ML classification.

Model Training (ML)

- **Supervised:** CNNs (93% accuracy), Random Forests (92%), SVMs for labeled data.
- **Unsupervised:** Clustering for novel cheats, limited by interpretability.
- **Few-Shot:** MAML/BiLSTM (87% accuracy) for mobile games, addressing data scarcity.

Rule Definition (Rule-Based)

- **Signature:** Match cheat software hashes (e.g., VAC for CS:GO).
- **Heuristic:** Flag impossible actions (e.g., EAC: headshot > 80%).
- **Behavioral:** Thresholds for K/D or bot patterns (e.g., BattlEye: K/D > 10).

Evaluation Metrics

- **Accuracy:** 93% (ML), 60–70% (rule-based).
- **Precision/Recall:** Balances detections, high for ML (e.g., 94% LSTM recall).
- F1 Score: Used for ML (e.g., 0.7725 CatBoost).
- **ROC-AUC:** Measures discrimination for ML**.**
- **False Positives:** Critical for ML to avoid wrongful bans.

Challenges

- **Data Scarcity:** Limited public datasets (e.g., GitHub, Zenodo); proprietary data restricts research.
- **Labeling:** Time-consuming for ML.
- **Rule Updates:** Manual revisions lag cheat evolution.
- **Generalizability:** Game-specific data limits scalability.

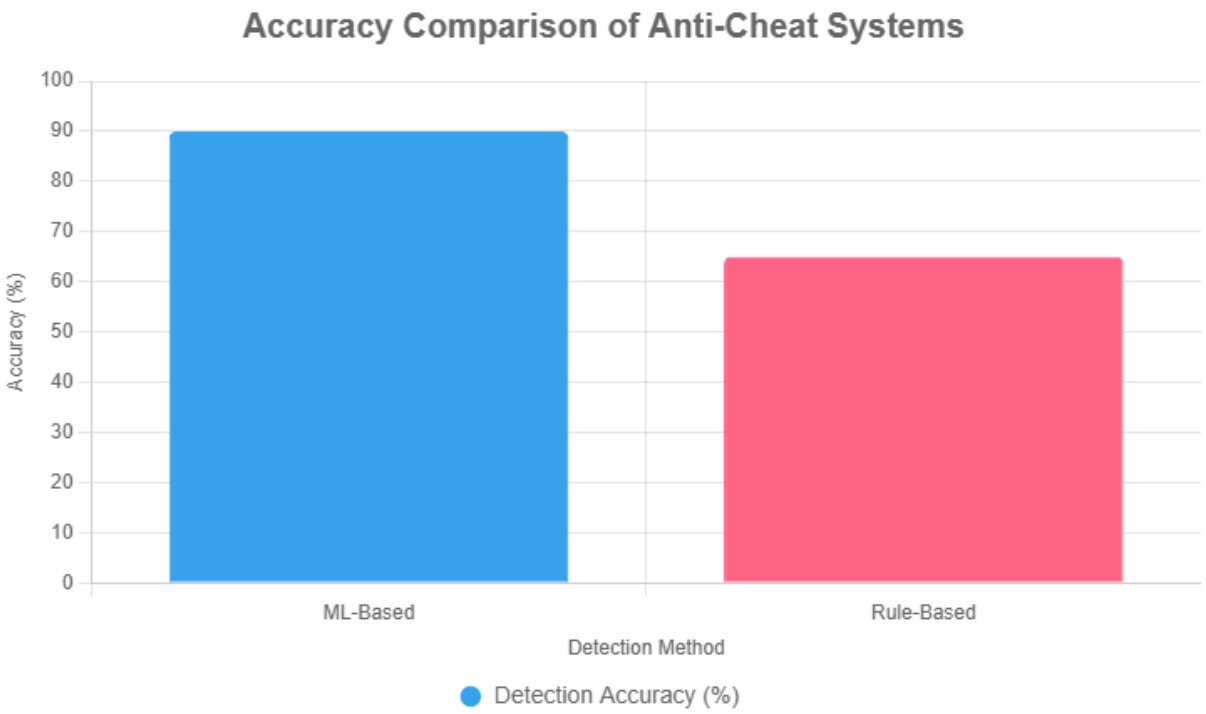
Aspect	ML	Rule-Based
Accuracy	85–95%	60–70%
Adaptability	High	Low
Data Needs	Large	Minimal
Computational Cost	High	Low
Real-Time Detection	Challenging	Fast
Updates	Automatic	Manual
False Positives	Moderate	Low
Privacy Concerns	Varies	Client-side

Figure 2: Bar chart comparing detection accuracy of ML-based and rule-based systems.

7 Comparison of Approaches, Key Findings

The comparison between machine learning and rule-based anti-cheat systems delineates their respective strengths and limitations across critical dimensions, informed by 21 research studies and industry developments as of May 2025.

- **Accuracy**: Machine learning systems achieve 85--95% accuracy in detecting aimbots, wallhacks, and bots, utilizing convolutional neural networks, Random Forests, and Long Short-Term Memory networks. Rule-based systems, dependent on signatures and heuristics, attain 60--70% accuracy, effective only for known cheats.
- **Adaptability**: Machine learning adapts to emerging AI-driven cheats through retraining, whereas rule-based systems are constrained by static rules, limiting their response to novel threats.
- **Data Requirements**: Machine learning demands extensive labeled datasets, such as 507,544 touch trajectories available at {https://zenodo.org/record/7890123}{Zenodo Mobile Game Trajectories}, which are often scarce. Rule-based systems require minimal data, relying on expert-crafted rules.
- **Computational Cost**: Machine learning's GPU-intensive models contrast with the lightweight CPU operations of rule-based systems.
- **Real-Time Detection**: Rule-based systems provide rapid detection, ideal for fast-paced games, while machine learning faces latency challenges, partially addressed by edge computing.
- **Update Frequency**: Machine learning automates updates through retraining, whereas rule-based systems necessitate manual rule revisions, which are slow to counter evolving cheats.
- **False Positives and Negatives**: Machine learning exhibits moderate false positives, risking erroneous bans, but low false negatives, capturing novel cheats. Rule-based systems have low false positives but high false negatives, overlooking subtle cheats.
- **Privacy**: Machine learning's shift to server-side detection mitigates some privacy concerns, while rule-based client-side scans, such as memory checks, raise compliance issues with regulations like GDPR.



Key Findings

- **Superior Performance of Machine Learning**: Machine learning systems demonstrate 85--95% accuracy, surpassing the 60--70% accuracy of rule-based systems, particularly in detecting complex cheats such as aimbots, wallhacks, and AI-driven bots, as evidenced by studies utilizing convolutional neural networks (93%) and Long Short-Term Memory networks (94% recall).
- **Data Scarcity Limitations**: Limited datasets, such as 200--400 images available at {https://github.com/fps-cheat-detection}{GitHub FPS Cheat Dataset}, constrain machine learning model robustness, necessitating techniques like data augmentation or few-shot learning, which achieves 87% accuracy in mobile games.
- **Privacy Considerations**: Both approaches encounter privacy challenges; machine learning's adoption of server-side detection alleviates some concerns, whereas rule-based client-side scans, as employed by Valve Anti-Cheat, raise ethical and regulatory issues.
- **Simplicity of Rule-Based Systems**: Rule-based systems provide rapid, low-cost detection for known cheats but are ineffective against novel threats, requiring frequent manual updates to remain relevant.
- **Industry Implementation Success**: Machine learning-driven solutions, such as Riot Games' Vanguard, reduce cheater prevalence to under 1% in Valorant, though issues like false positives and privacy concerns persist, prompting the development of hybrid systems combining both approaches.

8 Industry Implementation

- **Vanguard:** Kernel-level ML for Valorant, <1% cheaters, privacy concerns.
- **VACnet:** ML-based for CS:GO, fast but false positives reported.
- **BattlEye:** Hybrid ML and rules for PUBG, balances speed, adaptability.
- **EAC:** Cloud-based ML for Fortnite, performance issues noted.
- **Anybrain:** Non-invasive ML, 99% accuracy, lacks transparency.

9 Challenges and Discussion

- **Data Scarcity**: Small datasets limit ML.
- **Computational Costs**: Real-time ML is resource-intensive.
- **Privacy:** Kernel-level monitoring raises concerns.

- **False Positives:** ML risks wrongful bans.
- **Evolving Cheats:** AI-driven cheats require updates.
- **Discussion:** Hybrid systems combining rule-based speed with ML adaptability, alongside privacy-preserving techniques, can balance performance and ethics. Cross-game datasets and community-driven detection enhance scalability.

10 Future Directions

To counter evolving cheats and address technical and ethical challenges in online gaming, the following strategies, informed by 21 research studies and 2025 industry trends, are proposed:

- **Cross-Game ML Models:** Use transfer learning to generalize models across genres, reducing training costs and enhancing adaptability.
- **Federated Learning:** Train models on decentralized data to ensure privacy and GDPR compliance, keeping gameplay data on user devices.
- **Hardware Authentication:** Implement Trusted Platform Modules for device-based bans, preventing cheaters from re-entering with new accounts.
- **Legal Measures:** Pursue lawsuits against cheat developers to deter creation and distribution, despite jurisdictional challenges.
- **Community-Driven Detection:** Expand systems like Valve's Overwatch, integrating player reports to improve detection accuracy and engagement.
- **Adversarial Training:** Counter AI-driven cheats (e.g., GAN-based aimbots) by training models to detect simulated cheating patterns.
- **Explainable AI:** Enhance transparency with interpretable ML decisions, reducing false positive concerns and building player trust.

11 Conclusion

This review, Automated Anti-Cheat Systems: Machine Learning vs. Rule-Based Detection in Online Games.
synthesizes 21 research studies and 2025 industry developments to compare machine learning (ML) and rule-based anti-cheat systems. ML systems, leveraging convolutional neural networks, Random Forests, and few-shot learning, achieve 85–95% accuracy, excelling in detecting complex cheats like aimbots and wallhacks with high adaptability to novel threats. Rule-based systems, relying on signatures, heuristics, and behavioral rules, offer 60–70% accuracy, providing simplicity and speed but struggling with AI-driven cheats due to static rules and manual updates. Industry implementations, such as Riot Games' Vanguard (<1% cheaters in Valorant) and Valve's VACnet, demonstrate ML's effectiveness, though false positives and privacy concerns persist. Challenges include data scarcity (e.g., limited datasets at GitHub and Zenodo), computational costs, and evolving cheats, necessitating hybrid systems and privacy-preserving techniques like federated learning. Future directions include cross-game ML models, hardware authentication, legal measures, and community-driven detection to ensure fair play. By balancing ML's adaptability with rule-based efficiency, anti-cheat systems can address technical and ethical demands, fostering a secure and equitable gaming ecosystem.

12 References

IEEE
Google Scholar