Waldo Hub White Paper

1st Edition
By the Waldo Intelligence Team
June 10, 2024

1. Overview

Abstract:
Waldo Hub is a new platform that leverages state-of-the-art deep learning to robustly detect cheaters in video games. By applying modern deep learning models to game data, Waldo can characterize player behavior with minimal supervision, ensuring a fair and enjoyable gaming experience for all participants.

Problem Statement:
The prevalent issue in online gaming is cheating, which deteriorates user experience and threatens the integrity of competitive play. Not only has the complexity of modern games increased, but the emergence of AI-driven, highly sophisticated cheating methods has also necessitated a novel approach: combating machine learning cheats with machine learning detection.

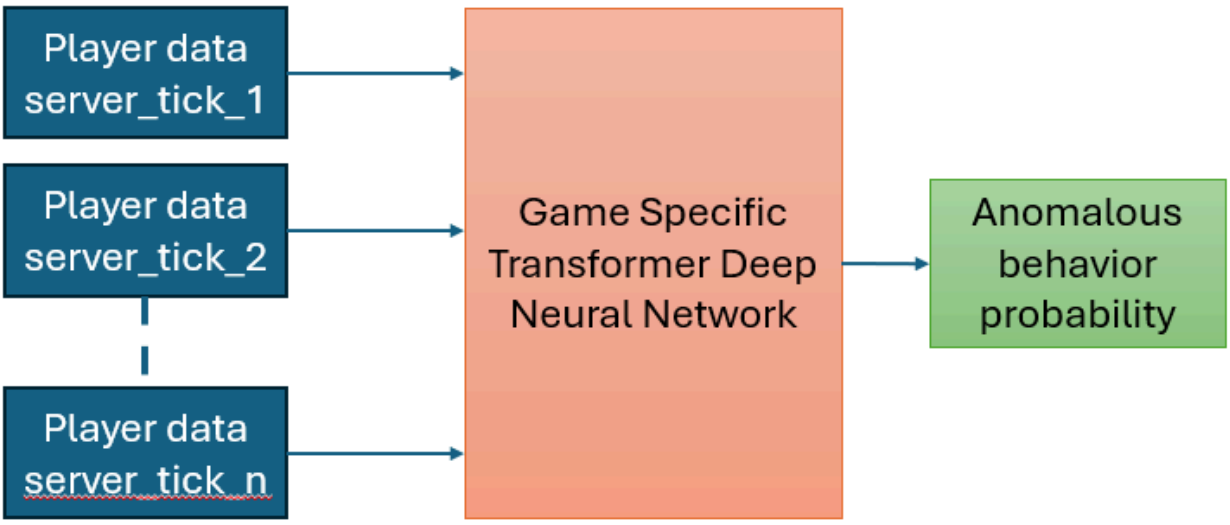
Value Proposition:
Our solution employs advanced transformer models that analyze sequences of gameplay data across various types. This approach allows Waldo to adapt to any game environment, predicting anomalous behaviors indicative of cheating with high accuracy. Our approach requires minimal supervision, which significantly lowers the barrier to entry for game developers aiming to ensure fair play.

2. Model Architecture

Overview of Model Category:
Transformers have revolutionized the field of machine learning due to their effectiveness in handling sequence data. Their ability to model complex dependencies makes them particularly suited for interpreting the nuanced behaviors of players in video games, which are inherently sequential and rich in contextual information.

High-Level Architecture and Flow:
The core of our architecture is a transformer-based model that processes sequences of gameplay actions. The model is designed to extract temporal features from game data, which are crucial for identifying patterns associated with cheating.

Architecture Diagram:



Proof of Concept and Similar Models:
Machine Learning, increasingly utilized for anti-cheat mechanisms in games like Rainbow 6 Siege, forms the core of our approach. As members of our team have previously demonstrated the effectiveness of transformers in non-video game domains, we developed a model tailored for cheat detection that analyzes pixel-based data from video streams with high success. Leveraging this expertise, we are now applying transformers to interpret direct game data. This shift to focusing on raw gameplay actions promises to significantly enhance our model's capability to accurately identify cheating behavior, moving beyond simple anomaly detection to robust, context-aware cheat recognition.

3. Data Collection Process

Types of Data Needed:
To effectively train our models, we will collect large datasets of game engine and server data from various video games. This data will primarily consist of in-game actions and state information recorded at each tick, providing a rich, sequential dataset for analysis.

Pre-Training Approach:
Initially, the model will undergo pre-training using an autoregressive technique similar to those used in training large language models. Since our pre-training does not require labeled data, this phase substantially reduces the need for human annotation, accelerating the understanding of general gameplay patterns and behaviors.

Requirement for Labeled Data:
For the fine-tuning phase, where the specific task is to identify cheating behavior, a smaller set of labeled data will be necessary. This data will be used to teach the model what player actions are linked with cheating, enhancing its ability to distinguish between fair play and malicious activities.

Data Format:
Our system is designed to handle arbitrary game data in any common structured form (csv, json, yaml, etc.). Initially, data ingestion will be facilitated through a RESTful API, allowing for the seamless upload of large volumes of game data.

Volume of Data:
A substantial volume of data, ranging from hundreds of thousands to a few millions of samples, will be necessary to ensure the robustness of the model. More data generally leads to better model performance and generalization.

Data Privacy and Protection:
Protecting customer data is a paramount concern. All data will be handled with strict adherence to privacy laws and regulations. Techniques such as data anonymization and encryption will be employed to secure data both in transit and at rest.

4. Model Training Process

Training Techniques:
The training will be conducted using modern, state-of-the-art machine learning frameworks like PyTorch. For larger models, techniques such as data parallelism and gradient checkpointing may be employed to optimize training efficiency and resource usage.

Hardware for Training:
All model training will be hosted on our dedicated hardware. This centralized approach ensures that training environments are optimized and controlled, leading to more consistent model performance. Though, we are exploring options to allow local training for developers who already have local compute.

Metrics for Success:
Success will be quantitatively measured by the model's accuracy and precision in

detecting cheaters using held-out test data provided by customers. This data will never be used in training, ensuring that our evaluations are unbiased and reflective of real-world performance.

Data Collection of Metrics:
Key performance metrics will include accuracy, precision, recall, and the F1 score. Additional metrics may include computational efficiency during inference and the adaptability of the model across different games.

5. Product / Deliverable

End Product Description:
The primary deliverable to our customers is a comprehensive API that facilitates easy uploading of game data, training of detection models, and running inference to identify potential cheating behavior. This API allows game developers to integrate cheating detection seamlessly into their game environments without the need for extensive machine learning expertise.

Intellectual Property Ownership:
We retain all rights to the intellectual property of the software, including the source code of the trained models. This ownership ensures ongoing development and improvement of the detection capabilities while allowing us to offer the service under a controlled, secure environment. Any data uploaded by our customers remains their intellectual property. Customers can request data deletion or withdraw from our service at any time.

Pricing Model:
Our pricing strategy is transparent and straightforward. We charge for the computational resources, networking bandwidth, and storage used during training and inference– with a modest markup to cover development and operational expenses. This model ensures that customers only pay for what they use, making it cost-effective and scalable.

Model Delivery and Integration:
Customers will have the flexibility to integrate the cheating detection service directly via API. In the future, we plan on working individually with customers to enable deploying the trained model to run inference locally. This dual approach will allow customers to choose the best integration method that suits their security and performance requirements.

6. API Integrations / Waldo Hub Website

Waldo Hub Website:
The Waldo Hub website serves as the central platform for project setup, management, and monitoring. It provides an intuitive interface where customers can manage their accounts, configure model parameters, upload data, and view analytics on model performance. This website is essential for ensuring that customers can easily navigate and utilize our services efficiently.

Waldo REST API:
The Waldo REST API is a critical component of our service, designed to facilitate the seamless integration of our cheating detection capabilities into customer software. It supports a range of functions from data upload to model training and inference, ensuring that developers can integrate these capabilities directly into their game environments.

Language SDKs and Game Engine Plugins:
We plan to support a variety of language SDKs and plugins for popular game engines, such as Unreal Engine and Unity. This support will make it even easier for game developers to integrate Waldo's cheating detection into their games, regardless of the development environment they are accustomed to using.

Feature Development Status:
Both the Waldo Hub and the REST API are currently works in progress, with ongoing development to enhance functionality and user experience. We are committed to regular updates and improvements based on customer feedback and technological advancements.