UTMIST

AI^2

REINFORCEMENT LEARNING

TOURNAMENT

TECHNICAL GUIDE

Feb 9th-13th

Table of Contents

Introduction to Enviornment

Tournament Environment

What is an environment?
An environment is the world in which an AI agent learns. It defines the rules, interactions, and consequences of actions.

What is the enviornment used in this tournament?
The AI^2 environment is based on the 1-v-1 platform fighting game **Brawlhalla**. The objective of the game is to knockout their opponents.

Game Format

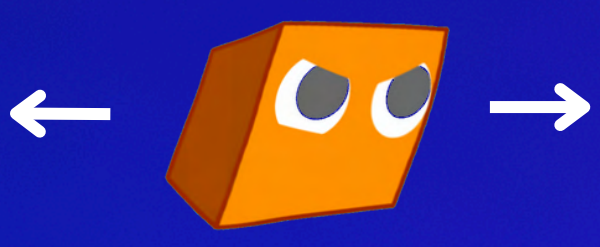
1. Each player starts with a set number of **stocks/lives**
2. A player loses a stock when they are knocked into a **knockout zone**
3. The match ends when a player **runs out of stocks** or **time runs out**
4. If time runs out, the winner is determined by: a) The player with the most stocks remaining. b) If stocks are tied, the player who took the least damage on their last stock wins

Knockout Zone at the border of the map

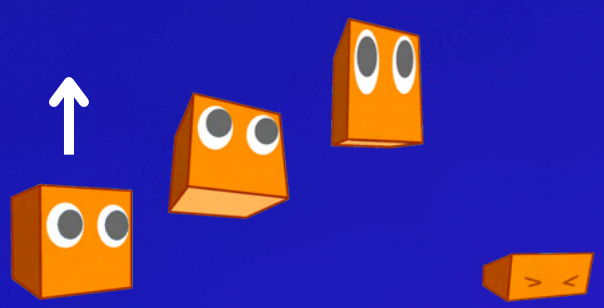
Introduction to Enviornment

Basic Mechanics

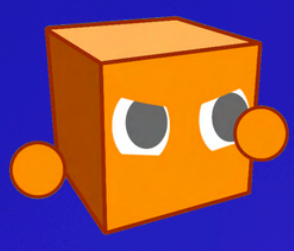
Walk/run



Jump



Dodge

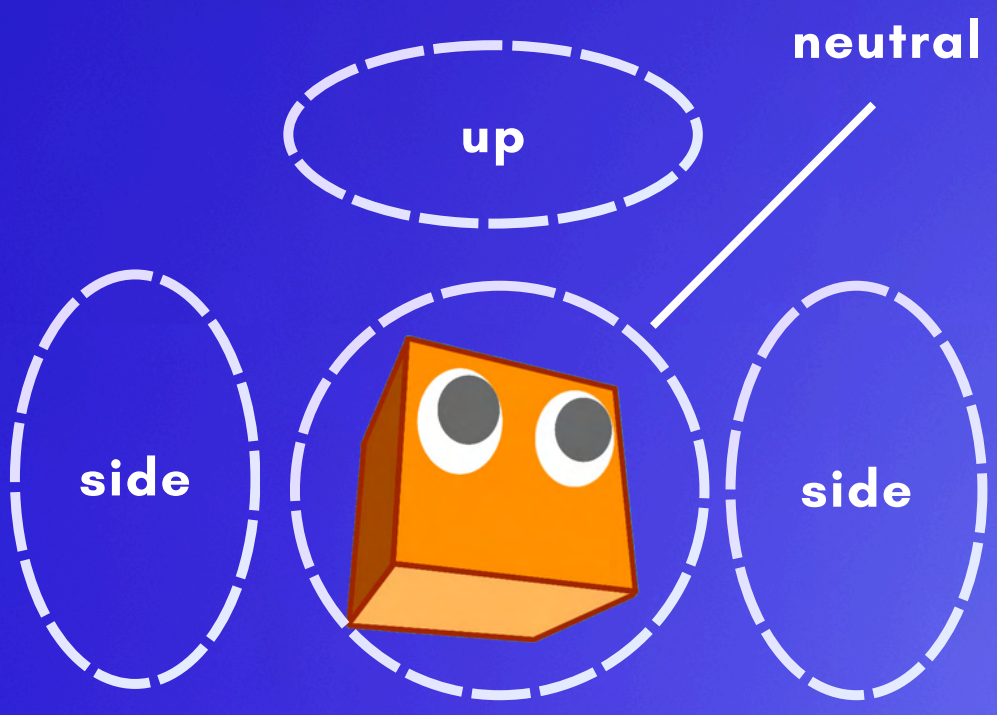


Attack

All attacks have 3 parameters:

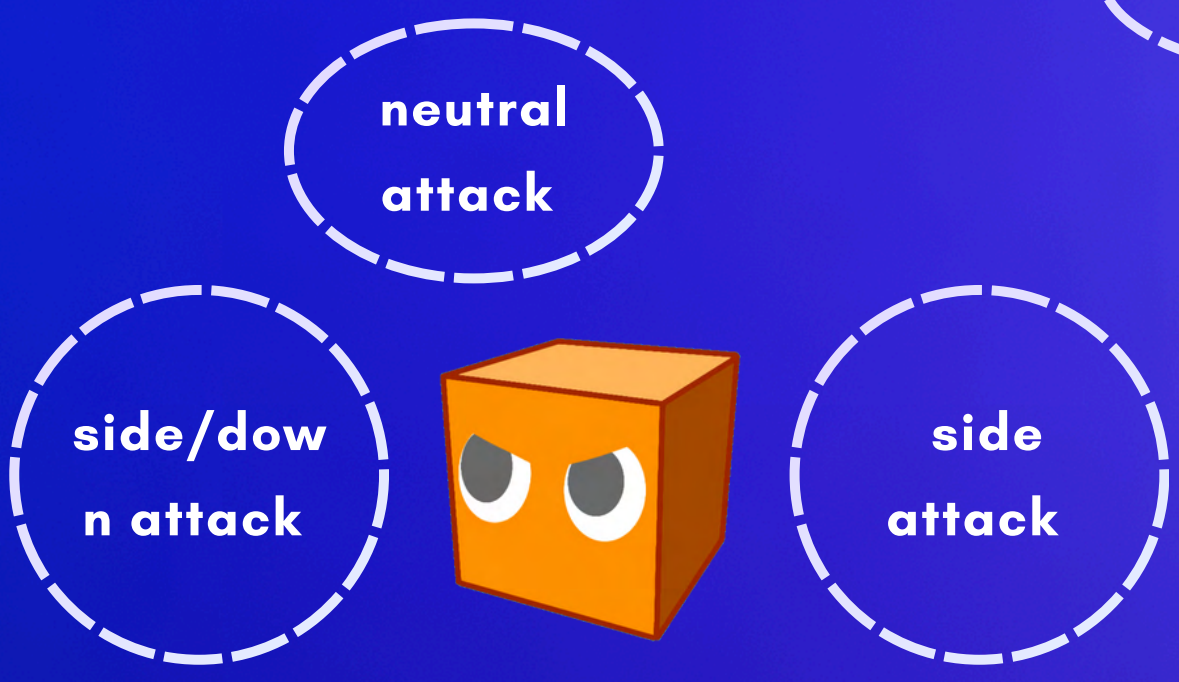
- grounded or aerial
- heavy or light
- direction

Aeriel Attacks



down

Grounded Attacks



Basics of Reward Functions

Q: What is a Reward Function?

A reward function takes in information from:

a. the environment
b. agent's own actions/states, and
c. opponent's actions/states

to provide a numerical reward signal. This reward helps the agent **evaluate the effectiveness of its actions** in achieving a goal.

Q: How does Reward Function design influence Behaviour?

Reward functions shape behavior by defining what is considered **"successful/unsuccessful"** to the agent. Agent's are trained to maximize cumulative reward, so it **learns behaviours based on what actions/events are rewarded in the reward funciton.** You can modify what behaviours you would like your agent to do by including adding different reward functions to your training.

EXAMPLE: BASIC OFFENCE REWARD

Rewards oponent taking damage, encouraging dealing damage to the opponent

$$Reward_t^{offsense} = SumDamageTaken_{t+1}^{opp} - SumDamageTaken_t^{opp}$$

EXAMPLE: BASIC DEFENCE REWARD

Negatively rewards taking damage, enocuraging not getting hit by opponent

$$Reward_t^{defence} = SumDamageTaken_t^{self} - SumDamageTaken_{t+1}^{self}$$

Existential State/Env Reward

Q: What are State/Enviornment Rewards?

Existential rewards are based off of the agent's state in the environment. Relevant reward functions in this category uses information such as the agent's position, health, or other persistent game states to determine the total reward.

It is worth noting, that these are all independent of actions. Through training the agents implicitly learn what actions to complete to result in getting these rewards.

When designing your own existential reward functions think about what enviornment and agent states are desirable and undesirable.

EXAMPLE: DANGER ZONE

Negatively rewards the agent for every time step it is in the "danger zone (DZ)" (i.e., far away from platform), encouraging the agent to stay on/near the platform and not straying near the edge of the map.

$$Reward_t^{DZ} = \begin{cases} -1, & \text{if the agent's position is in the Danger Zone (DZ)} \\ 0, & \text{otherwise} \end{cases}$$

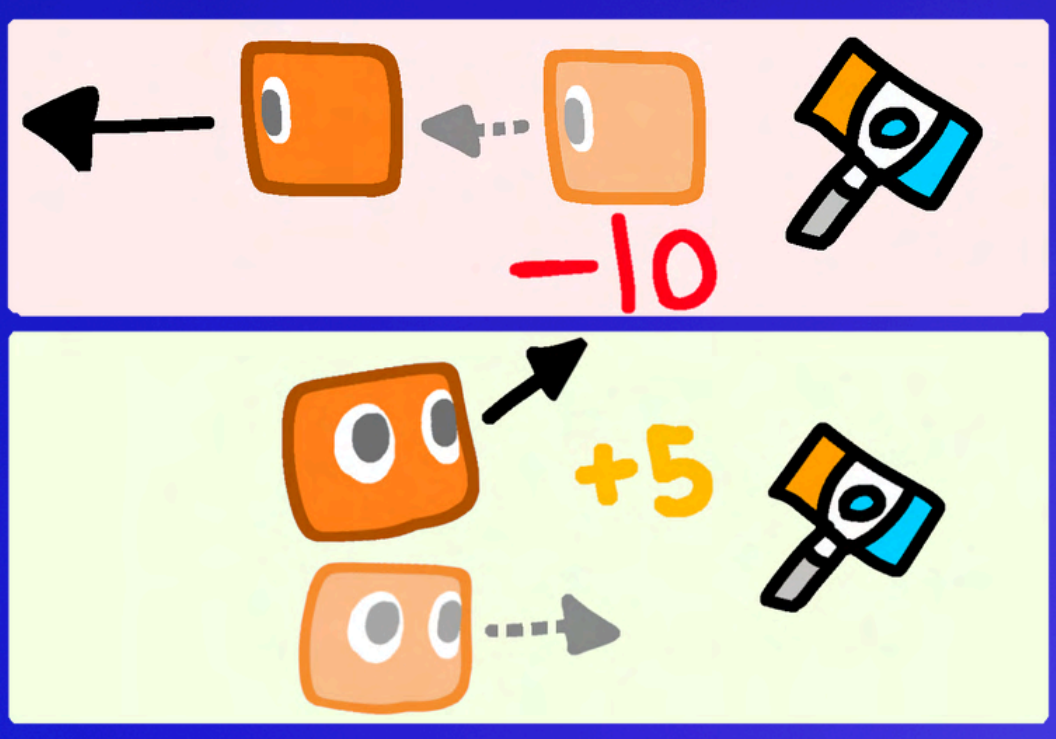
$$\text{DZ} = \{(x, y) \mid x \notin [x^{\text{safe}}_{\text{min}}, x^{\text{safe}}_{\text{max}}] \text{ or } y \notin [y^{\text{safe}}_{\text{min}}, y^{\text{safe}}_{\text{max}}]\}$$

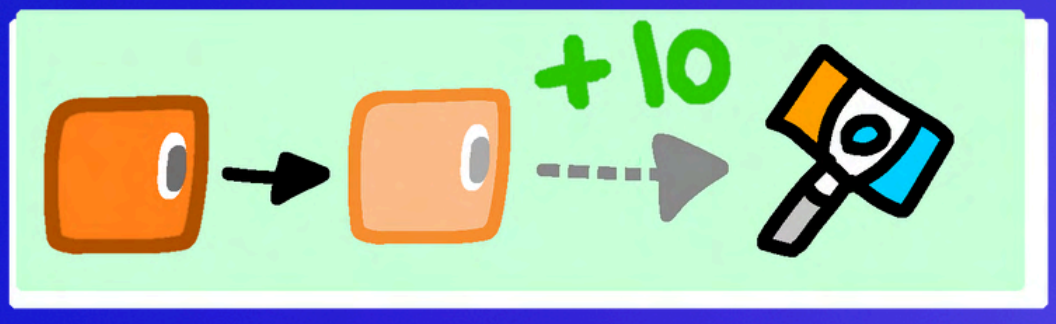
Modulo Existential Reward

Q: What are “Modulo Existential” Rewards?

Unlike standard existential rewards, which directly depend on an agent’s state, modulo existential rewards involve **additional computations**—such as vector operations, transformations, or nonlinear functions—to evaluate the agent’s position or other state variables.

If you want to learn more about these types of rewards, they are typically called **geometric** or **vector-based** rewards. Some examples of these type of rewards could be realted to edge-guarding, stage-control, and weapon-camping.





EXAMPLE: TARGET POSITION MOVEMENT

Rewards the agent for moving toward a target position by computing the alignment between its movement direction and the optimal path. Encourages efficient navigation by reinforcing movement in the correct direction rather than just reaching the target.

$$\text{Reward}_t^{\text{movement}} = \left(\frac{\mathbf{p}_{t+1} - \mathbf{p}_t}{\|\mathbf{p}_{t+1} - \mathbf{p}_t\|} \right) \cdot \left(\frac{\mathbf{p}_{\text{target}} - \mathbf{p}_t}{\|\mathbf{p}_{\text{target}} - \mathbf{p}_t\|} \right)$$

Single Event / Sparse Reward

Q: What are Single Event/Sparse Rewards?

Single Event or Sparse rewards are given only when specific conditions or **milestones are met**, rather than continuously throughout the game. This may include knocking out the opponent, landing a combo, or winning a match.

Q: How can agents get Event/Sparse Rewards?

The sequence of actions required to earn these rewards is not always obvious, so additional design considerations may be needed to help the agent learn and get these rewards effectively. This can be achieved through:

Shaping Rewards: Providing smaller, incremental rewards for progress toward the goal (e.g., rewarding damage dealt before a knockout).

Exploration Strategies: Encouraging diverse action selection to help discover effective strategies.

EXAMPLE: KNOCKOUT REWARD

Provides a large reward when opponent is knocked out, encouraging agent to get the opponent knocked out

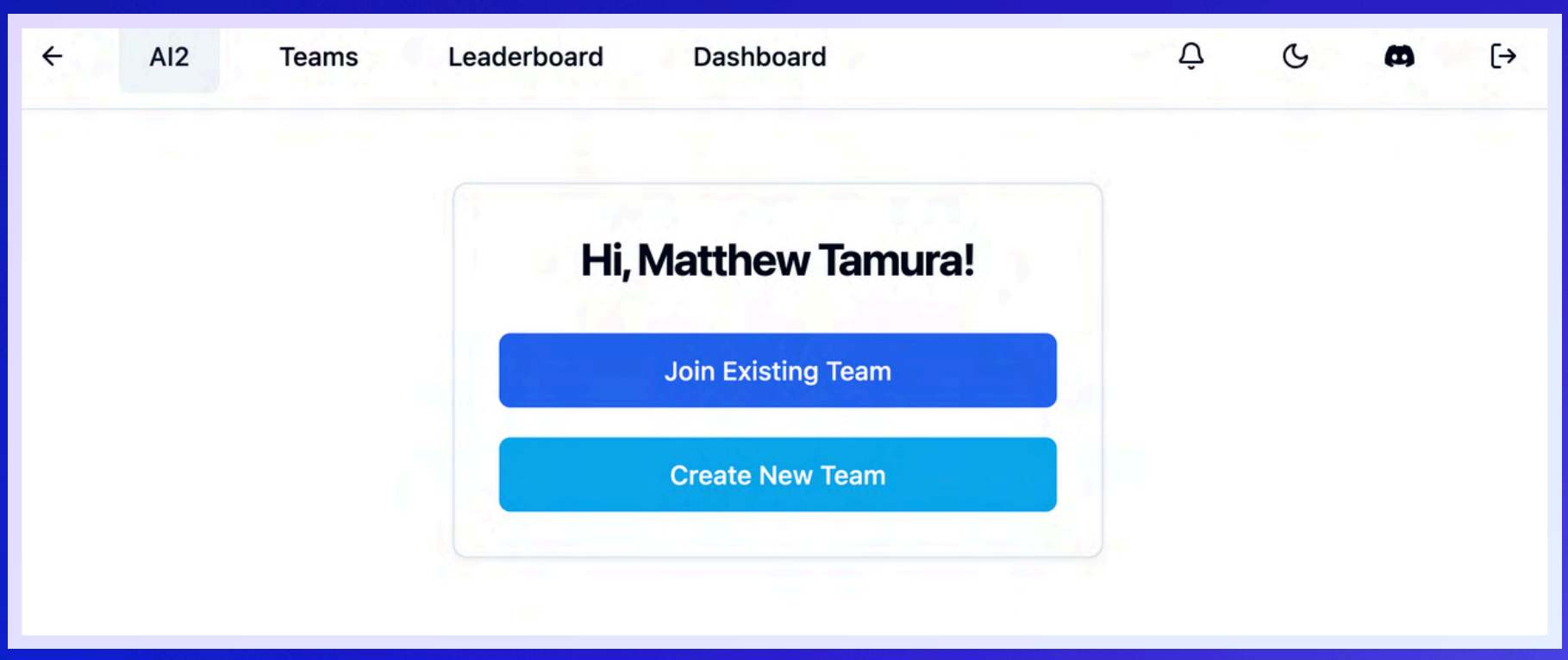
$$Reward_t^{KO} = \begin{cases} +50, & \text{if opponent is knocked out at time } t \\ 0, & \text{otherwise} \end{cases}$$

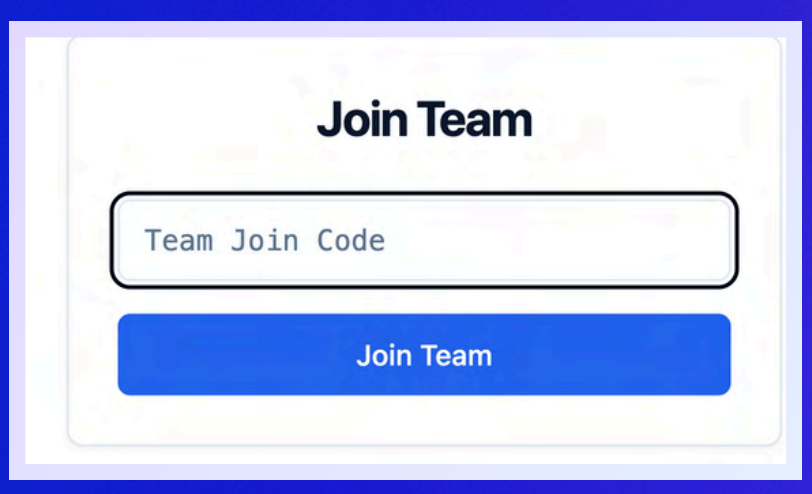
Registering your team

Join our Website

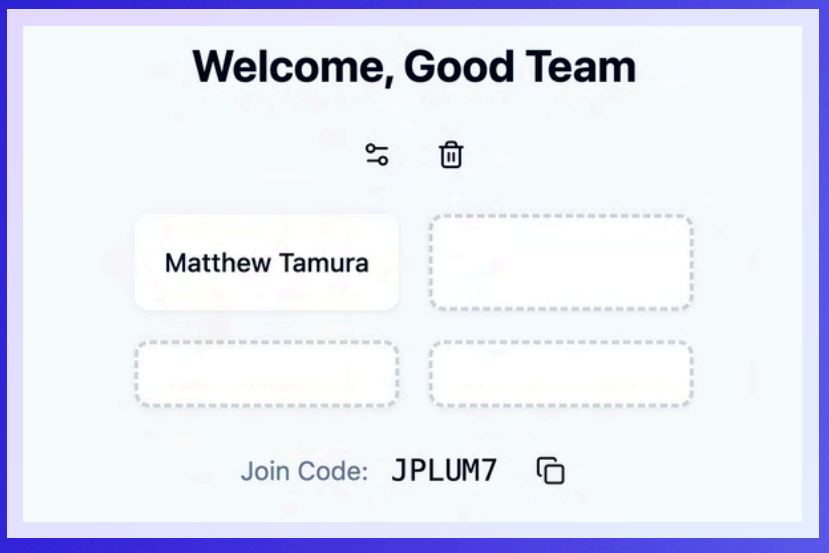
Head to our website: https://www.utmist.ca/ai2

Creating a Team: Dashboard





Join a team: Input team code

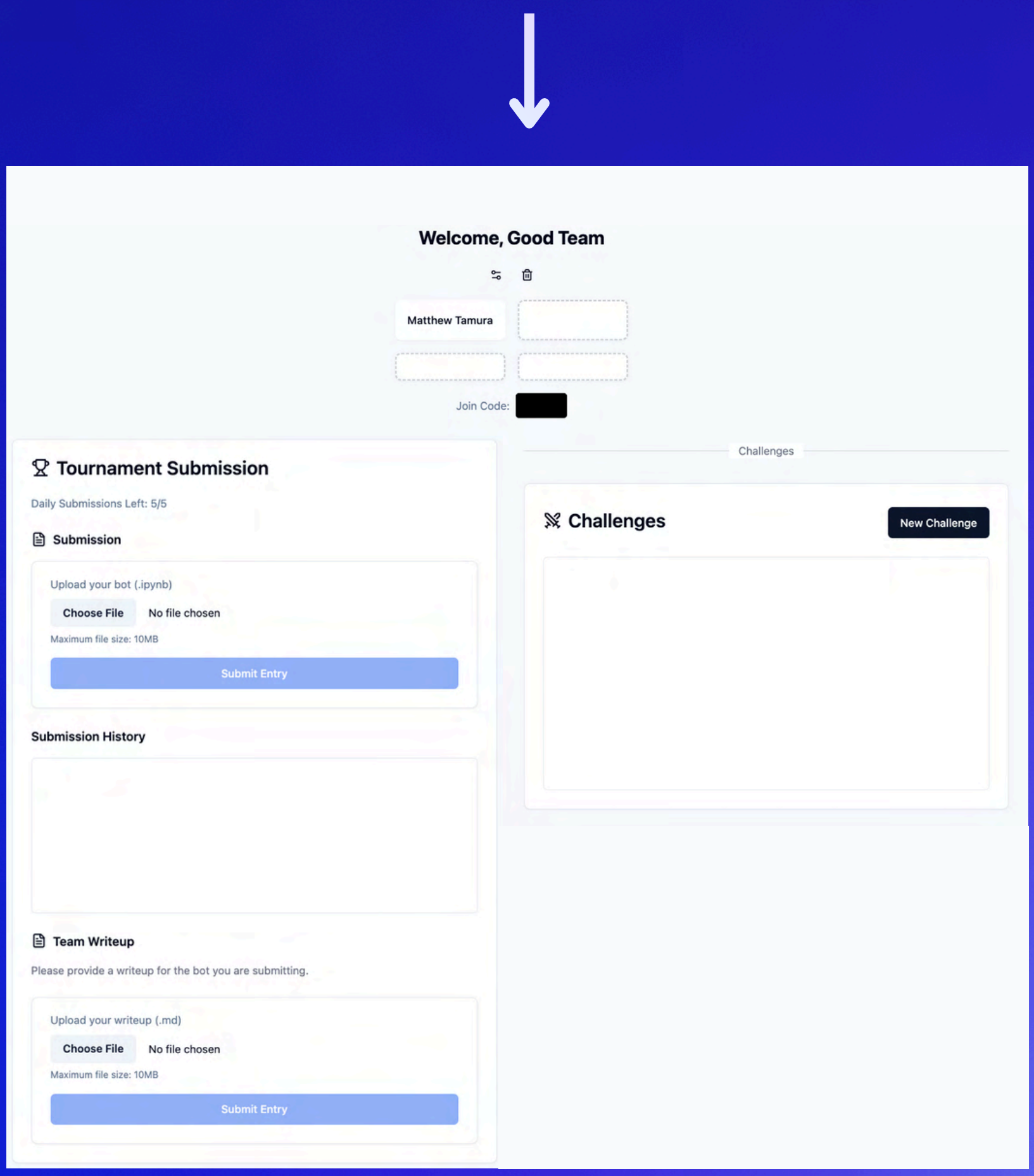


Create a team: and let others join via the team code

Submitting your agent

Submit .ipynb file

Click on black box to see **team join code**



Create a **writeup** detailing your solution

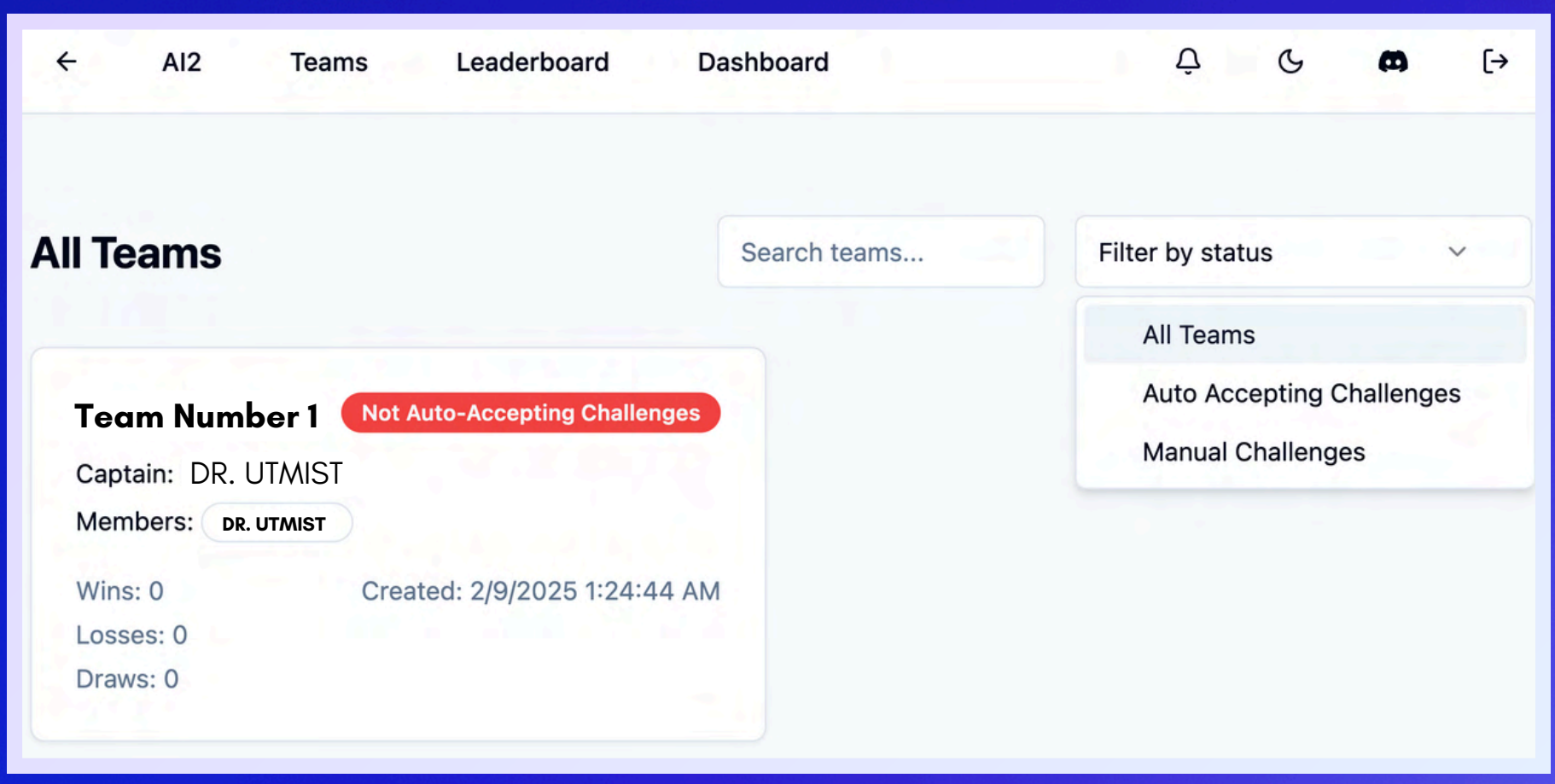
Challenge other teams (more details on next page)

Challenging others

Teams

The team view shows all the different teams in the tournament. Depending on a team's status, you can choose to **challenge a team** to a battle!

Challenges take the **latest agent submission** from the two teams and has them face off for one round against each other. Depending on the outcome of the battle, each team's ELO will be updated.



Leaderboard

The **ELO** from the challenges will be displayed on the leaderboard tab.