UNIVERSIDADE DO ESTADO DA BAHIA
DEPARTAMENTO DE CIÊNCIAS EXATAS E DA TERRA
CURSO DE GRADUAÇÃO EM SISTEMAS DE INFORMAÇÃO

RAFAEL DA COSTA FONSÊCA

POLÍTICAS DE DECISÃO DE COMPORTAMENTOS NUM SISTEMA MULTIAGENTES UTILIZANDO APRENDIZAGEM POR REFORÇO

SALVADOR
2023

RAFAEL DA COSTA FONSÊCA

POLÍTICAS DE DECISÃO DE COMPORTAMENTOS NUM SISTEMA MULTIAGENTES UTILIZANDO APRENDIZAGEM POR REFORÇO

Monografia apresentada ao curso de Sistemas de Informação do Departamento de Ciências Exatas e da Terra da Universidade do Estado da Bahia - UNEB, como requisito parcial à obtenção do grau de bacharel em Sistemas de Informação. Área de Concentração: Ciência da Computação, Inteligência Artificial e Sistemas Multiagentes.

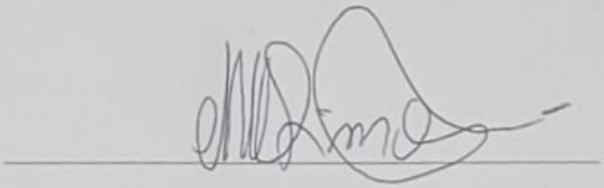
Orientador: Marco Antônio Costa Simões

SALVADOR

2023

Termo de Anuência do Orientador

Declaro para os devidos fins que li e revisei este trabalho e atesto sua qualidade como resultado final desta monografia. Confirmo que o referencial teórico apresentado é completo e suficiente para fundamentar os objetivos propostos e que a metodologia científica utilizada e os resultados finais são consistentes e com qualidade suficiente para submissão à banca examinadora final do Trabalho de Conclusão de Curso II do curso de Bacharelado em Sistemas de Informação.



Marco Antônio Costa Simões

RAFAEL DA COSTA FONSÊCA

POLÍTICAS DE DECISÃO DE COMPORTAMENTOS NUM SISTEMA MULTIAGENTES UTILIZANDO APRENDIZAGEM POR REFORÇO

Monografia apresentada ao curso de Sistemas de Informação do Departamento de Ciências Exatas e da Terra da Universidade do Estado da Bahia - UNEB, como requisito parcial à obtenção do grau de bacharel em Sistemas de Informação. Área de Concentração: Ciência da Computação, Inteligência Artificial e Sistemas Multiagentes.

Aprovada em:

BANCA EXAMINADORA

Marco Antônio Costa Simões (Orientador)
Universidade do Estado da Bahia – UNEB

Adailton de Jesus Cerqueira Junior
Universidade Salvador (UNIFACS)

Diego Gervasio Frias Suarez
Universidade do Estado da Bahia - UNEB

Dedico este trabalho aos meus amados pais, Anderson Trindade e Bárbara Adjane, minha esposa Gabrielle Almeida e minha irmã Bianca Costa. Seu amor incondicional, apoio constante e encorajamento são a força que impulsiona minha jornada. Sou eternamente grato por tê-los ao meu lado.

AGRADECIMENTOS

Gostaria de expressar meus sinceros agradecimentos a todos que contribuíram para a realização desta monografia. Sem o apoio e a colaboração de cada um de vocês, não seria possível alcançar este resultado. Primeiramente, gostaria de agradecer ao meu orientador Marco Simões, pela orientação, expertise e paciência ao longo deste processo. Agradeço imensamente ao Professor Diego Frias por suas valiosas contribuições essenciais para o desenvolvimento deste projeto. Sua orientação, insights e dedicação foram fundamentais em cada etapa, tornando este trabalho possível. Sem o seu apoio incansável, este projeto seria inviável. Agradeço sinceramente pela expertise, paciência e comprometimento que foram cruciais para o sucesso desta empreitada acadêmica. Agradeço também à minha família e a minha esposa pelo amor, incentivo e todo o suporte provido durante esta jornada. Aos membros do corpo docente que, nestes anos de formação e desenvolvimento profissional tiveram papel fundamental no meu crescimento acadêmico e pessoal. Aos meus amigos e companheiros de sala, agradeço pelo encorajamento e compreensão ao longo desta jornada acadêmica, dos quais a presença e incentivo foram fundamentais para minha motivação e bem-estar emocional. Expresso meu agradecimento a Gabriel Mascarenhas e Filipe Silva, companheiros do ACSO, pela colaboração inestimável. Sua contribuição foi de extrema importância para o êxito deste projeto.

Por fim, agradeço a todos que direta ou indiretamente me apoiaram ao longo deste trabalho. Mais uma vez, obrigado a cada um de vocês.

"Sempre parece impossível até que seja feito."

(Nelson Mandela)

RESUMO

Esta monografia aborda o desafio da tomada de decisão em sistemas multiagentes, utilizando o aprendizado por reforço como uma solução para este desafio. O problema investigado nesta monografia é o desenvolvimento de políticas de decisão que permitam a coordenação e colaboração entre agentes, de forma a melhorar o desempenho coletivo em cenários complexos, como o futebol de robôs simulados. Este problema vem da necessidade de encontrar estratégias e técnicas que permitam aos agentes aprenderem comportamentos adequados, considerando a dinamicidade da interação com outros agentes e com o ambiente. Esta monografia descreve a modelagem e implementação do ambiente de aprendizagem, treinamento dos agentes utilizando técnicas de aprendizado por reforço e a avaliação dos resultados obtidos. Essa investigação tem relevância para a aplicação em problemas reais de coordenação e decisão em sistemas multiagentes, onde a tomada de decisão eficiente pode ser crucial para alcançar resultados ótimos e melhorar a cooperação entre os agentes. Ao treinar os agentes de um time de futebol para selecionar os comportamentos mais eficientes com o uso de algoritmos como DQN e DDPG, observaram-se melhorias notáveis no desempenho defensivo da equipe, com uma redução significativa de 10 vezes nas perdas durante as partidas.

Palavras-chave: Aprendizado por reforço; Sistemas multiagentes; Políticas de decisão; Futebol de robôs simulados.

ABSTRACT

This thesis addresses the challenge of decision-making in multi-agent systems, using reinforcement learning as a solution to this challenge. The problem investigated in this thesis is the development of decision policies that enable coordination and collaboration among agents, in order to improve collective performance in complex scenarios, such as simulated robot soccer. This problem arises from the need to find strategies and techniques that allow agents to learn appropriate behaviors, considering the dynamic interaction with other agents and the environment. This thesis describes the modeling and implementation of the learning environment, agent training using reinforcement learning techniques, and the evaluation of the results obtained. This research is relevant for application in real-world coordination and decision problems in multi-agent systems, where efficient decision-making can be crucial to achieve optimal results and enhance cooperation among agents. By training the agents of a soccer team to select more efficient behaviors using algorithms like DQN and DDPG, notable improvements were observed in the defensive performance of the team, resulting in a significant reduction of 10 times in losses during matches.

Keywords: Reinforcement Learning; Multi-agent Systems; Decision Policies; Robot Soccer Simulation.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AM	Aprendizagem de máquina
APR	Aprendizagem por Reforço
DDPG	Deterministic Policy Gradient
DQN	Deep Q-Network
IA	Inteligência Artificial
MADDPG	Multi-Agent Deep Deterministic Policy Gradient
MDP	Processo de Decisão de Markov
MPE	Multi-agent Particle Envs
PPO	Proximal Policy Optimization
RCSSServer	RoboCup Soccer Simulator Server
SMA	Sistemas Multiagentes
TRPO	Trust Region Policy Optimization

SUMÁRIO

1 INTRODUÇÃO

A área de pesquisa abordada nesta monografia trata do uso da Inteligência Artificial (IA)(1), mais especificamente no uso de Aprendizagem por Reforço (APR) em Sistemas Multiagentes (SMA)(2). Esta é uma subárea da IA que lida com a interação e colaboração de múltiplos agentes que devem agir de forma autônoma em um ambiente compartilhado, onde esses agentes devem aprendem a tomar decisões para maximizar as recompensas, visando alcançar o objetivo do grupo.

O conceito de IA pode ser abordado de diferentes formas, passando por pontos como o processo de tomada de decisões e a capacidade de aprendizado, o desempenho melhorado de tarefas hoje realizadas por humanos e a racionalidade(1). De forma resumida, pode-se considerar IA como um campo da ciência da computação que dedica esforços à criação de sistemas que permitem aos computadores a realização de atividades que necessitem da inteligência humana.

Os SMA existem em diversos domínios, como por exemplo a robótica(3), jogos(4), sistemas de transporte(5)(carros autônomos e drones, por exemplo), economia(6) e gerenciamento de recursos(7). Este processo de aprendizado entre agentes autônomos, para a colaboração e coordenação de ações de forma adaptativa é uma área de pesquisa em crescimento(4)(5)(6)(8)(9)(10), devido ao seu potencial para resolver problemas complexos e reais que envolvem múltiplos agentes.

No contexto do aprendizado por reforço, os agentes devem interagir com um ambiente dinâmico, em constante mudança de estado dos objetos e também dos outros agentes ao redor, recebendo recompensas ou penalidades com base em suas ações. O objetivo da aprendizagem por reforço é o mapeamento de situações(estado atual do ambiente) para ações, fazendo com que o agente saiba como agir diante da configuração ou estado do mundo em que se encontra. Deste modo, estes devem ser capazes de realizar ações que maximizem suas recompensas acumuladas ao longo do tempo por meio da tentativa e erro(11).

As pesquisas realizadas para a realização deste trabalho giram em torno principalmente em métodos de treinamento, variando, por exemplo, em métodos centralizados(7), parcialmente descentralizados(5), e métodos descentralizados(12). Estes métodos exploram a

centralização ou descentralização das decisões entre os agentes, de modo que um agente pode ser responsável pelas decisões no método centralizado(7)(13)(14) , ou cada agente toma decisões individuais no método descentralizado(15)(12)(16) . Outra abordagem é a definição prévia de posicionamentos/comportamentos estratégicos para os agentes, e o modelo atua na escolha destes posicionamentos(14).

Outro ponto crucial abordado nas pesquisas refere-se aos testes das redes neurais, mais especificamente na avaliação da robustez em ambientes críticos(4). Foram conduzidos testes abrangentes para verificar a capacidade das Redes Neurais Profundas em lidar com uma variedade de condições e contextos, assegurando que esses modelos possam ser aplicados com eficácia em diferentes cenários do mundo real. A ênfase na validação abrange desde a precisão na execução de tarefas específicas até a capacidade de aprender e se adaptar diante de mudanças nas condições do ambiente, garantindo a confiabilidade e eficácia dessas redes em aplicações práticas.

No entanto, algumas lacunas foram identificadas, como por exemplo a falta de abordagens eficientes para a coordenação dos agentes em ambientes dinâmicos(7)(5). Estes artigos apresentam a dificuldade de alcançar um bom desempenho em ambientes multiagentes em relação aos treinamentos com agentes únicos. Isso ocorre devido a alguns fatores, como por exemplo o aumento da quantidade de variáveis a serem avaliadas, do tempo de treinamento e da imprevisibilidade do ambiente. A não consideração da coordenação entre os agentes pode gerar outras consequências, como por exemplo quando existem "agentes preguiçosos"(8). Este problema acontece quando os agentes tendem a minimizar esforços ou a computação desnecessária. Apesar da adaptação rápida, estes agentes tendem a possuir dificuldades na generalização, que é um ponto extremamente relevante para o aprendizado, principalmente com a dinamicidade dos ambientes. Outro fator crucial que afeta a performance dos agentes é a existência de objetivos conflitantes entre agentes, podendo desta forma atrapalhar no objetivo comum. Além disso existem também limitações com relação à escalabilidade em ambientes com maior quantidade de agentes(14). Estas limitações são causadas pelos fatores mencionados anteriormente e também pelo problema da observação parcial do ambiente, de modo que os agentes nem sempre possuem completa percepção do estado atual.

Diante dos pontos abordados acima relacionados à dificuldade existente de abordagens eficientes para a coordenação de múltiplos agentes em ambientes dinâmicos(7)(5)(8),

decidiu-se abordar como problema central o processo de tomada de decisão em sistemas multiagentes, com foco na escolha de comportamentos coletivos.

A fim de explorar o processo de tomada de decisão coletiva em sistemas multiagentes, foi escolhido o futebol de robôs simulados como cenário para a aplicação prática. Nesse contexto, equipes compostas por 11 agentes autônomos em cada time, interagem em um ambiente altamente dinâmico e competitivo. Deste modo, é possível observar o processo de coordenação de ações, a tomada de decisões conjuntas e a capacidade de adaptar os comportamentos de forma eficiente para alcançar um objetivo comum.

O futebol de robôs simulados apresenta desafios significativos para a tomada de decisões coletivas, como a coordenação de movimentos e a colaboração em estratégias de defesa e ataque. Além disso, existe também a necessidade de balancear as ações individuais e coletivas, considerando tanto as habilidades e estado dos jogadores individuais quanto os objetivos da equipe como um todo.

Para a execução deste trabalho foi escolhido time BahiaRT para a interação com o ambiente de futebol de robôs. O BahiaRT é um dos times participantes da RoboCup, uma iniciativa científica internacional dedicada a promover o avanço na área de robôs inteligentes no contexto do futebol de robôs.

Diante do que foi exposto, surgem as seguintes perguntas norteadoras:

- Quais são as aplicações práticas do Aprendizado por Reforço (APR) em sistemas multiagentes (SMA) no contexto do futebol de robôs, e qual é a relevância dessas aplicações?
- Quais são os principais desafios enfrentados na coordenação de múltiplos agentes em ambientes dinâmicos de futebol de robôs e como esses desafios impactam o desempenho das equipes?
- Como os agentes autônomos em equipes de futebol de robôs simulados interagem em ambientes altamente dinâmicos e competitivos, e quais são os principais fatores que influenciam seu comportamento?
- Em que medida as abordagens de APR podem ser escaladas para equipes maiores de futebol de robôs, e quais são os desafios específicos associados a essa escalabilidade?
- Quais vantagens são observadas ao integrar o aprendizado por reforço para aprimorar o desempenho das equipes de futebol de robôs?

O objetivo geral deste trabalho é desenvolver estratégias de aprendizado por reforço e coordenação de sistemas multiagentes para melhorar o desempenho de equipes de futebol de robôs em ambientes dinâmicos e competitivos. A intenção é aperfeiçoar a habilidade desses agentes na seleção de comportamentos por meio do emprego da aprendizagem por reforço. Essa abordagem possibilita a consideração da complexidade e da variabilidade do ambiente no qual os agentes operam. A adoção desse enfoque experimental acrescenta uma dimensão prática ao objetivo geral, viabilizando uma abordagem mais concreta e aplicada no desenvolvimento das estratégias propostas.

Para alcançar este objetivo geral, são estabelecidos os seguintes objetivos específicos:

- Realizar uma revisão sistemática da literatura relacionada ao uso de aprendizado por reforço e sistemas multiagentes em futebol de robôs, identificando desafios e abordagens existentes.
- Propor estratégias de APR e coordenação de agentes que levem em consideração os desafios específicos do ambiente de futebol de robôs.
- Implementar e testar as estratégias propostas em cenários de simulação de futebol de robôs, avaliando seu desempenho em termos de coordenação de equipes e tomada de decisões.
- Validar quantitativamente e qualitativamente as estratégias propostas, comparando seu desempenho com métodos convencionais.

Resultados Esperados:

- Modelagem de estratégias de aprendizado por reforço e coordenação de sistemas multiagentes especificamente adaptadas para o contexto do futebol de robôs.
- Implementação de sistemas de controle baseados nas estratégias propostas em cenários simulados de futebol de robôs.
- Demonstração da eficácia das estratégias propostas, incluindo melhorias na coordenação de equipes de robôs e na tomada de decisões em ambientes dinâmicos.

Essas perguntas norteadoras, objetivo geral, objetivos específicos e resultados esperados fornecem uma estrutura para a pesquisa relacionada a futebol de robôs, aprendizado por reforço e sistemas multiagentes.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, são estabelecidas as bases teóricas que sustentam a abordagem deste estudo. O objetivo é proporcionar uma compreensão sólida dos principais conceitos que subjazem ao desenvolvimento e à aplicação de soluções em SMA, com foco na tomada de decisões, utilizando o aprendizado por reforço como abordagem central.

A IA é um campo fundamental no qual se baseia esta pesquisa. A IA abrange a criação de sistemas capazes de realizar tarefas que normalmente requerem inteligência humana, como aprendizado, raciocínio e tomada de decisões. Nesse contexto, a Aprendizagem de máquina (AM) desempenha um papel crucial, uma vez que permite que os agentes aprendam com dados e melhorem seu desempenho ao longo do tempo, sem uma programação explícita(1).

Os SMA(2) constituem o foco central desta investigação. SMA envolvem a interação de múltiplos agentes em um ambiente comum, onde cada agente busca atingir seus próprios objetivos. A coordenação eficaz e a tomada de decisão colaborativa entre esses agentes são desafios complexos e críticos, especialmente em cenários dinâmicos(14) e competitivos(4).

A aprendizagem por reforço (APR)(11) é outro ponto fundamental para este projeto. No contexto da APR, agentes interagem com um ambiente, buscando aprimorar seu desempenho através da maximização de recompensas associadas a ações específicas. Este paradigma destaca-se pela capacidade dos agentes aprenderem comportamentos eficazes mediante a experiência. A APR desempenha um papel crucial ao permitir a adaptação e tomada de decisões eficientes por parte dos agentes, contribuindo para a melhoria contínua de estratégias e comportamentos(14).

Ao compreender essas teorias e abordagens, a intenção é capacitar os leitores a familiarizarem-se com os conceitos fundamentais e os alicerces teóricos que serão essenciais para contextualizar os capítulos subsequentes desta monografia. Além disso, reconhece-se que essa base teórica servirá como um guia fundamental à medida que se exploram soluções para a tomada de decisão em sistemas multiagentes, com ênfase no aprimoramento da cooperação e do desempenho coletivo em situações desafiadoras.

2.1 INTELIGÊNCIA ARTIFICIAL

A definição clássica de IA, conforme proposta por Alan Turing, é ancorada no conceito da "Máquina de Turing Universal"(17). Em sua obra seminal "Computing Machinery and Intelligence", Turing apresentou a ideia de uma máquina capaz de realizar qualquer tarefa computacional que pudesse ser descrita por meio de um algoritmo(17). Ele estabeleceu o famoso "Teste de Turing"como um critério para determinar se uma máquina exibe inteligência comparável à humana. Segundo o teste, se um observador humano não conseguir distinguir, com base apenas em interações, se as respostas vêm de uma máquina ou de um humano, então a máquina pode ser considerada inteligente.

Essa definição incorpora o conceito de capacidade computacional universal da Máquina de Turing, sugerindo que a inteligência pode ser replicada por meio de processos algorítmicos. A abordagem de Turing contribuiu significativamente para moldar a compreensão inicial da IA e continua a ser uma referência central no campo. No entanto, é importante notar que, ao longo do tempo, outras definições e perspectivas surgiram, enriquecendo o entendimento da IA e incluindo elementos como aprendizado de máquina, redes neurais e abordagens mais contemporâneas.

Dentro desse contexto, diversos pesquisadores e estudiosos têm fornecido definições para a IA(1), abordando diferentes perspectivas sobre seu significado e alcance. Nesta seção, apresentaremos algumas dessas definições que ajudam a compreender melhor o campo da IA.

1. Pensando como um humano: Uma das definições enfatiza a capacidade das máquinas de pensar e ter mentes, de forma análoga à inteligência humana. Deste modo, a IA pode ser descrita como um esforço para criar máquinas que possam pensar e ter uma mente completa(18). Essa definição destaca o aspecto da simulação da mente humana por meio de sistemas computacionais.
2. Pensando racionalmente: Outra definição destaca o aspecto do pensamento racional e das atividades associadas ao pensamento humano. A IA é descrita como a automatização de atividades que normalmente são associadas ao pensamento humano, como a tomada de decisões, a resolução de problemas e o aprendizado. Essa definição enfatiza a capacidade dos sistemas de realizar tarefas intelectuais de maneira racional(19).
3. Agindo como seres humanos: Uma abordagem mais orientada para o comportamento

humano define a IA como a arte de criar máquinas capazes de executar funções que exigem inteligência quando realizadas por pessoas(20). Essa definição enfoca a capacidade dos sistemas de imitar ou reproduzir o comportamento humano de maneira inteligente.

4. Agindo racionalmente: Uma definição mais geral e abrangente afirma que a IA está relacionada ao desempenho inteligente de artefatos(21). Isso implica que a IA busca criar sistemas que possam agir de maneira racional, fazendo a coisa certa com base em seu conhecimento e objetivos.

Essas diferentes definições demonstram as perspectivas sobre a IA e as diferentes abordagens adotadas para compreender e desenvolver sistemas inteligentes. Por meio destas, é possível ter uma noção a respeito da influência que estas abordagens têm no processo de pesquisa e desenvolvimento de projetos na área como um todo. É importante destacar também que o foco em uma ou mais abordagens destas depende do objetivo da aplicação que será desenvolvida. Neste projeto específico, a abordagem mais apropriada é quarta (Agindo Racionalmente), uma vez que o agente toma decisões com base em experiências anteriores, visando otimizar a recompensa e alcançar seus objetivos. Essa perspectiva destaca a importância da tomada de decisões informadas e orientadas para o sucesso, considerando a aprendizagem com base em resultados passados como um elemento essencial para a eficácia das ações do agente.

2.2 APRENDIZADO DE MÁQUINA

A AM(22) é uma área da IA que visa o desenvolvimento de técnicas computacionais voltadas para a aquisição de conhecimento pelos sistemas. Trata-se da construção de programas de computador que, por meio da solução bem-sucedida de problemas anteriores, tomam decisões embasadas em experiências acumuladas. Esses sistemas não apenas aprendem com seus êxitos(e falhas) passados, mas também adaptam seu comportamento com o tempo, refinando continuamente suas capacidades. Essa capacidade de automação do aprendizado é essencial para lidar com dados complexos e cenários dinâmicos, permitindo que os sistemas evoluam e melhorem seu desempenho ao longo do tempo(23).

Existem diferentes abordagens e paradigmas dentro da aprendizagem de máquina. Uma delas é o aprendizado supervisionado, no qual os algoritmos recebem um conjunto de exemplos de treinamento que contêm tanto as entradas como os rótulos dos resultados esperados. Com base nesses exemplos, o sistema aprende a mapear as entradas para as classes corretas e,

posteriormente, é capaz de classificar novos exemplos não rotulados(23).

Outro paradigma é o aprendizado não supervisionado, no qual os algoritmos analisam os exemplos de treinamento sem rótulos de classe e procuram identificar padrões, estruturas ou agrupamentos presentes nos dados. Essa abordagem é útil quando não se tem conhecimento prévio sobre as classes ou quando se deseja explorar a estrutura dos dados de forma mais geral(23).

Além disso, a aprendizagem de máquina também pode envolver técnicas de aprendizado por reforço, onde o sistema aprende a tomar ações em um ambiente para maximizar uma recompensa. No âmbito das técnicas de aprendizado simbólico, os algoritmos utilizam representações simbólicas e manipulação de símbolos para expressar conhecimento e tomar decisões(11). Por outro lado, as técnicas de aprendizado não simbólico se fundamentam em representações numéricas e processamento de dados de alta dimensionalidade, sem depender explicitamente de símbolos ou interpretações simbólicas(11).

Os sistemas de aprendizagem de máquina podem ser aplicados em uma variedade de áreas, como reconhecimento de padrões, processamento de linguagem natural, visão computacional, medicina, finanças, entre outros. O objetivo é utilizar essas técnicas para melhorar a capacidade de tomar decisões, prever resultados ou automatizar tarefas de forma mais eficiente do que os métodos tradicionais.

2.2.1 Aprendizado por reforço

A aprendizagem por reforço é um processo no qual um agente aprende a tomar decisões e mapear situações para ações de forma a maximizar um sinal numérico de recompensa(11). Nesse tipo de aprendizagem, o agente não recebe instruções explícitas sobre quais ações tomar, mas precisa descobrir por si mesmo quais ações proporcionam a maior recompensa através de tentativa e erro. Em casos mais desafiadores e interessantes, as ações podem afetar não apenas a recompensa imediata, mas também as próximas situações e, consequentemente, todas as recompensas subsequentes, que chamamos de recompensa atrasada. Essas duas características - busca por tentativa e erro e recompensa atrasada - são os aspectos mais importantes e distintivos da aprendizagem por reforço.

Diferentemente de outros tipos de aprendizagem, a aprendizagem por reforço se

concentra na interação do agente com o ambiente e na busca por maximizar a recompensa ao longo do tempo. O agente aprende a tomar ações que são recompensadas positivamente e a evitar ações que são recompensadas negativamente. Ao explorar o ambiente e experimentar diferentes ações, o agente acumula conhecimento sobre as consequências de suas ações em relação às recompensas recebidas. Esse processo de aprendizagem contínua permite ao agente ajustar seu comportamento e desenvolver uma política de ação que otimiza a obtenção de recompensas ao longo do tempo.

Enquanto a aprendizagem por reforço busca maximizar uma recompensa, a aprendizagem supervisionada e não supervisionada têm objetivos diferentes. A aprendizagem por reforço é especialmente adequada para situações em que exemplos de comportamento desejado são difíceis ou impossíveis de rotular e quando o agente precisa aprender a partir de sua própria experiência interativa com o ambiente.

2.2.2 Processo de Decisão de Markov

O Processo de Decisão de Markov (MDP) é um modelo matemático utilizado em teoria de controle e aprendizado de máquina para representar situações em que um agente toma decisões sequenciais em um ambiente incerto. Ele é denominado de Markov devido à propriedade de Markov, que significa que o estado futuro do sistema depende apenas do estado presente e não da sequência completa de eventos anteriores(24).

No OpenAI Gym(25), o MDP é uma abstração fundamental para modelar ambientes em que agentes tomam decisões sequenciais em estados sucessivos. O Gym fornece uma interface consistente para definir ambientes MDP, que foi utilizada como base para o BahiaRT Gym(26). O MDP é definido por cinco componentes:

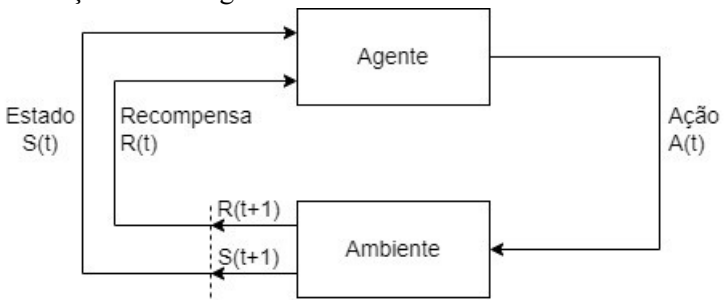
- **Conjunto de Estados (S):** Representa todos os estados possíveis do sistema, ou seja, as situações em que o agente pode se encontrar.
- **Conjunto de Ações (A):** Descreve as ações disponíveis para o agente realizar no ambiente.
- **Função de Transição de Estados (P):** Determina a probabilidade de transição de um estado para outro, dado uma ação. Essa função reflete a dinâmica do ambiente.
- **Função de Recompensa (R):** Atribui uma recompensa numérica a cada par estado-ação. O objetivo do agente é maximizar a recompensa cumulativa ao longo do tempo.
- **Fator de Desconto (γ):** Representa a preferência do agente por recompensas imediatas em

comparação com recompensas futuras. Valores próximos de 1 indicam mais consideração para recompensas futuras.

Este processo é representado de forma cíclica na Figura 1, delineando um ciclo contínuo de interação. Neste ciclo, o agente, guiado por estratégias ajustadas com base no feedback, busca otimizar suas ações para alcançar metas específicas no ambiente dinâmico.

O processo de decisão de Markov é resolvido através da formulação de uma política, que é uma estratégia que especifica qual ação o agente deve tomar em cada estado para otimizar sua recompensa ao longo do tempo(11). Diversos algoritmos, como a iteração de valor e métodos baseados em políticas, são empregados para encontrar a política ótima. Dentro desse contexto, é essencial compreender o ciclo típico de aprendizado por reforço, onde o agente interage continuamente com o ambiente. Inicialmente, o agente observa o estado atual do ambiente e, com base nessa observação, toma uma decisão sobre qual ação executar. Essa ação é então aplicada ao ambiente, resultando em uma mudança de estado e na recepção de uma recompensa que reflete o desempenho do agente. A recompensa, fundamental para o aprendizado, fornece um sinal indicativo de quão boa foi a ação tomada em relação aos objetivos do agente. O agente utiliza essas observações e recompensas para ajustar sua estratégia ao longo do tempo, refinindo sua política de ações para alcançar um desempenho mais otimizado. Esse ciclo de observação, ação, recompensa e ajuste de estratégia é repetido iterativamente durante o processo de treinamento, permitindo que o agente aprimore suas habilidades e tome decisões mais informadas em diferentes estados do ambiente.

Figura 1 – A interação entre o agente e o ambiente em um Processo de Decisão de Markov



Fonte: O próprio autor

2.2.3 Algoritmos de aprendizagem por reforço

Os algoritmos de aprendizagem por reforço desempenham um papel fundamental na capacidade de um agente de aprender ações e políticas que maximizam a recompensa em um

ambiente. A seguir, serão explicados alguns dos algoritmos relevantes neste domínio.

2.2.3.1 Q-Learning

Q-Learning: O Q-Learning é um algoritmo fundamental no contexto de aprendizado por reforço, oferecendo uma abordagem robusta para que agentes aprendam comportamentos ótimos em ambientes Markovianos controlados(11). O algoritmo opera através da iteração e atualização de uma tabela de valores de ação, denominada Q-table, com base nas recompensas obtidas do ambiente.

Em sua forma mais básica, conhecida como Q-learning de um passo, o algoritmo atualiza os valores de Q de acordo com a equação:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t) \right]$$

Nesta equação, $Q(St, At)$ representa o valor de ação atual para o par estado-ação, α é a taxa de aprendizado, $Rt + 1$ é a recompensa obtida na transição atual, γ é o fator de desconto, e $\max_a Q(St + 1, a)$ denota o máximo valor de ação esperado para o próximo estado.

A aplicação do Q-Learning é vasta, sendo utilizado em diversos domínios, incluindo jogos, robótica e sistemas de controle. A capacidade do algoritmo de aprender ações ótimas mediante interações com o ambiente o torna uma escolha valiosa para situações em que é necessário um agente autônomo aprimorar seu desempenho ao longo do tempo.

Ao longo do treinamento, o agente realiza escolhas de ação com base nos valores correntes de Q, recebe recompensas e ajusta seus valores de Q para refletir as experiências acumuladas(11). Essa abordagem iterativa, guiada pela maximização de recompensas futuras, permite que o agente aprenda estratégias eficientes para otimizar seu comportamento no ambiente Markoviano em que está inserido.

2.2.3.2 DQN

O Deep Q-Network (DQN) é uma evolução do algoritmo Q-learning, combinando técnicas de aprendizado profundo com reforço para abordar desafios em ambientes complexos. O DQN foi proposto por Mnih et al.(27), construindo sobre os princípios do Q-learning para

aprimorar a eficácia do aprendizado em ambientes mais desafiadores.

Assim como o Q-learning, o DQN tem como objetivo treinar um agente para tomar decisões sequenciais, maximizando a soma esperada de recompensas ao longo do tempo. No entanto, ao contrário do Q-learning tradicional, o DQN utiliza redes neurais profundas para aproximar a função de valor de ação Q, proporcionando maior expressividade e capacidade de generalização.

A função Q no DQN é definida da mesma forma que no Q-learning(l 1), representando a soma das recompensas descontadas ao longo do tempo. No entanto, em vez de armazenar uma tabela de valores Q para cada par estado-ação, o DQN utiliza uma rede neural profunda para aprender uma representação contínua e generalizável da função Q. Isso é particularmente útil em ambientes de alta dimensionalidade(27).

A arquitetura do DQN inclui várias camadas que, para um estado s específico, gera um vetor de valores de ação $Q(s, ;\theta)$, onde θ denota os parâmetros da rede. Em um espaço de estado de n dimensões e um espaço de ação contendo m ações, a rede neural funciona como uma transformação de R^n para R^mMnih et al.(27).

Dois componentes fundamentais do algoritmo incluem a implementação de uma rede alvo (*target network*) e a utilização da técnica de experiência de replay, conforme proposto por Mnih et al.(27).

A *target network*, reconhecida pelos parâmetros θ^-, compartilha semelhanças com a rede online, que representa a instância principal da rede neural no contexto do DQN. A distinção fundamental reside no fato de que os parâmetros da *target network* são replicados a cada τ passos da rede online, ou seja, $\theta_t^- = \theta_t$, permanecendo constantes nos demais momentos. Essa abordagem visa introduzir estabilidade ao processo de treinamento, reduzindo flutuações na função de valor de ação durante o aprendizado. A rede online, por sua vez, desempenha o papel principal na tomada de decisões do agente e na atualização dos parâmetros ao longo do treinamento. A formulação do alvo utilizada pelo DQN é determinada pela expressão:

$$Y_{\mathrm{DQN}_t} \equiv R_{t+1} + \gamma \max_a Q(S_{t+1}, a; \theta_t^-)$$

Nesta equação:

- Y_{DQN_t} é o alvo utilizado para treinamento no passo de tempo t.
- R_{t+1} é a recompensa obtida no próximo passo de tempo.
- γ é o fator de desconto.
- $\max_a Q(S_{t+1}, a; \theta_t^-)$ representa o valor máximo estimado para o próximo estado S_{t+1} pela rede alvo.

O replay buffer é outra das técnicas utilizadas no DQN para otimizar o aprendizado(27). O replay buffer armazena uma coleção de experiências passadas do agente, incluindo informações sobre estados, ações, recompensas e estados subsequentes. Durante o treinamento, em vez de usar apenas as experiências mais recentes, o agente amostra aleatoriamente mini-lotes do replay buffer. Isso quebra a correlação temporal entre as experiências consecutivas e promove a estabilidade no aprendizado, pois o modelo é treinado em uma mistura diversificada de situações passadas. Essa abordagem ajuda a melhorar a eficiência e a robustez do processo de aprendizado por reforço profundo.

Para enfrentar o desafio da exploração em ambientes complexos de alta dimensionalidade, o DQN implementa estratégias como o epsilon-greedy, que equilibra a exploração e a exploração do espaço de ações(27). Essa abordagem epsilon-greedy significa que, durante a seleção de ações, o agente escolhe a ação com o maior valor Q na maioria das vezes, mas ocasionalmente opta por uma ação aleatória com uma probabilidade epsilon. Isso permite explorar novas possibilidades no ambiente e evitar ficar preso em políticas subótimas.

2.2.3.3 DDPG

O DDPG(28) é um algoritmo de aprendizado por reforço projetado para enfrentar os desafios inerentes à aplicação direta de métodos baseados em Q-learning em espaços de ação contínuos. A limitação surge da necessidade de otimizar a política gananciosa(greedy policy) em cada passo de tempo, uma tarefa computacionalmente custosa em espaços de ação contínuos, especialmente quando se utiliza aproximadores de função grandes e não restritos e espaços de ação não triviais.

Para superar essas limitações, o DDPG adota uma abordagem ator-crítico baseada no algoritmo DPG (Deterministic Policy Gradient) proposto por Silver et al.(29). No contexto do DDPG, o algoritmo mantém uma função ator parametrizada, denotada por $\propto(s|\theta_\propto)$, que especifica a política atual, deterministicamente mapeando estados para ações específicas. Paralelamente, o

crítico, representado por $Q(s, a)$, é aprendido utilizando a equação de Bellman(19), semelhante ao Q-learning.

A atualização do ator ocorre aplicando a regra da cadeia ao retorno esperado a partir da distribuição inicial J em relação aos parâmetros do ator. Esta atualização é expressa como:

$$\nabla_{\theta_\propto} J \approx \mathsf{E}_{s_t \sim \rho_\beta} \quad \nabla_{\theta_\propto} Q(s, a|\theta_Q)_{\, s=s_t, a=\propto(s_t|\theta_\propto)}$$

Essa expressão é equivalente a uma estimativa da expectativa da derivada do Q em relação às ações, multiplicada pela derivada das ações em relação aos parâmetros do ator. Isso reflete a busca pela direção na qual a política deve ser ajustada para maximizar o retorno esperado.

O DDPG foi desenvolvido para enfrentar desafios em problemas de controle contínuo, nos quais as ações do agente são selecionadas a partir de espaços contínuos. Apresentado por Lillicrap et al.(28), o DDPG combina elementos de métodos determinísticos de política e gradiente de políticas, demonstrando eficácia em ambientes complexos.

O funcionamento do DDPG é baseado em duas redes principais: a rede ator (política) e a rede crítica (valor). A rede ator mapeia diretamente os estados para ações determinísticas no espaço contínuo, enquanto a rede crítica avalia a qualidade dessas ações, fornecendo feedback para orientar o aprendizado. O algoritmo utiliza um esquema de atualização baseado em gradientes de políticas determinísticas, permitindo uma abordagem eficaz em espaços de ação contínuos.

Uma característica distintiva do DDPG é a utilização de uma política determinística. Isso implica que, dada uma observação específica, a ação é escolhida de maneira determinística pela rede ator. Além disso, o algoritmo emprega uma técnica conhecida como experiência de reprodução (replay buffer), que armazena e amostra aleatoriamente experiências passadas durante o treinamento, contribuindo para a estabilidade do aprendizado.

O DDPG oferece vantagens notáveis, sendo particularmente adequado para espaços de ação contínuos, com aplicações bem-sucedidas em problemas de controle. A estabilidade no treinamento é promovida pela política determinística e pela experiência de reprodução. No entanto, algumas desvantagens são observadas, como a sensibilidade aos hiperparâmetros,

exigindo ajustes cuidadosos, e uma possível limitação na ênfase à exploração em ambientes desconhecidos.

Em termos matemáticos, o treinamento do DDPG envolve a minimização de uma função de perda que combina elementos da política e da função de valor Q(28).

2.2.3.4 Exploração e Exploração Baseada em Incerteza

A exploração desempenha um papel crítico na aprendizagem por reforço, pois o agente precisa equilibrar a escolha de ações conhecidas por serem eficazes com a exploração de ações desconhecidas que podem levar a recompensas ainda maiores. A seguir, serão abordadas algumas das estratégias de exploração, como a incerteza é usada para otimizá-las e outros pontos relevantes sobre o tema:

- Exploração vs. explotação: Na aprendizagem por reforço, o agente enfrenta o dilema entre a exploração, que envolve a escolha de ações desconhecidas para descobrir recompensas potenciais, e a explotação, que envolve a escolha de ações conhecidas por serem eficazes para maximizar as recompensas imediatas(11). O equilíbrio adequado entre essas duas abordagens é crucial para o sucesso do agente.
- Estratégias de Exploração Clássicas: Estratégias de exploração clássicas incluem o método epsilon-greedy, no qual o agente escolhe a ação com a maior recompensa estimada com probabilidade $1 - \varepsilon$ e uma ação aleatória com probabilidade ε(27). Essa abordagem é eficaz, mas pode ser simplista em ambientes complexos.
- Upper Confidence Bound (UCB): A estratégia UCB é uma abordagem avançada que leva em consideração a incerteza. Ela atribui maior prioridade às ações que têm o potencial de proporcionar recompensas substancialmente melhores com base em uma medida de incerteza, muitas vezes usando um intervalo de confiança superior(11). Isso permite que o agente explore ações que têm a maior incerteza em relação às recompensas.
- Exploração Baseada em Modelos: Em cenários onde o agente tem um modelo do ambiente, a exploração baseada em modelos pode ser usada para simular o comportamento do agente em estados desconhecidos(30). Isso permite que o agente avalie o resultado de diferentes ações antes de realizá-las, o que pode ser uma estratégia eficaz de exploração.
- Exploração Estocástica: Em ambientes com recompensas estocásticas, a exploração pode ser desafiadora. Estratégias como a exploração baseada em incerteza podem ajudar a

determinar quais ações têm maior probabilidade de levar a recompensas consistentemente maiores, mesmo em ambientes incertos(31).

- A Exploração em Aprendizado por Reforço Hierárquico: Em aprendizado por reforço hierárquico, a exploração pode ocorrer em diferentes níveis de abstração(32). O agente deve explorar tanto ações de alto nível, que correspondem a tarefas mais amplas, quanto ações de baixo nível, que correspondem a ações detalhadas dentro de uma tarefa.
- A Importância da Exploração Adequada: Uma exploração adequada é crucial para garantir que o agente não fique preso em um ótimo local subótimo ou não encontre ações que levem a recompensas significativas(11). A seleção de estratégias de exploração apropriadas é uma área ativa de pesquisa em aprendizado por reforço.
- Exploração em Ambientes de Aprendizado por Reforço em Tempo Real: Em ambientes em tempo real, a exploração pode ser ainda mais desafiadora, pois o agente deve tomar decisões rápidas(11). Estratégias eficazes de exploração são fundamentais para o desempenho do agente em tempo real.

A exploração eficaz é um dos desafios mais importantes na aprendizagem por reforço(11). Estratégias que equilibram a busca por recompensas imediatas com a exploração de ações desconhecidas são essenciais para que os agentes aprendam eficazmente em ambientes complexos e incertos. A pesquisa em exploração continua a evoluir, visando melhorar o desempenho dos agentes em uma variedade de cenários de aprendizado por reforço.

2.2.3.5 Problemas de Aprendizado por Reforço Contínuo e Parcialmente Observável

- Desafios em Ambientes Contínuos: Em muitos cenários do mundo real, as ações e estados não são discretos, mas sim contínuos(28). Isso significa que o agente deve escolher ações em um espaço contínuo, o que pode ser um desafio em comparação com problemas discretos. Estratégias avançadas, como o uso de algoritmos de otimização, são necessárias para lidar com a complexidade de ambientes contínuos.
- Processos de Decisão Parcialmente Observáveis (POMDPs): Nem sempre o agente tem acesso completo às informações sobre o ambiente(11). Em ambientes parcialmente observáveis, o agente lida com informações incompletas e incertezas sobre o estado atual. Resolver POMDPs é computacionalmente desafiador e requer técnicas avançadas, como filtragem de Kalman, redes neurais recorrentes e métodos de planejamento parcialmente

observáveis.

- Aprendizado de Políticas em Ambientes Contínuos e Parcialmente Observáveis: Aprender políticas em ambientes contínuos e parcialmente observáveis é uma tarefa difícil(33). Abordagens avançadas, como Aprendizado Profundo por Reforço (DRL) e redes neurais recorrentes, são usadas para representar políticas e funções de valor em espaços contínuos e ambientes parcialmente observáveis.
- Exploração Eficiente: Em ambientes contínuos e parcialmente observáveis, a exploração eficiente é um desafio significativo. Os agentes precisam explorar ações em espaços contínuos enquanto lidam com informações incompletas sobre o ambiente(11). Estratégias como a exploração baseada em incerteza são cruciais para encontrar políticas eficazes em tais cenários.
- Transferência de Conhecimento: Transferir conhecimento de um ambiente de treinamento para um ambiente de teste, muitas vezes com características diferentes, é uma questão crítica(34). A pesquisa se concentra em métodos avançados de transferência de conhecimento para permitir que os agentes aprendam mais eficientemente em ambientes contínuos e parcialmente observáveis.
- Aplicações em Robótica e Automação: Problemas de aprendizado por reforço contínuo e parcialmente observável são frequentes em robótica e automação, onde os agentes precisam lidar com estados e ações contínuos e incertezas sensoriais. Essas aplicações têm um impacto significativo na pesquisa em aprendizado por reforço.
- Algoritmos Avançados e Abordagens Híbridas: Algoritmos avançados, como o A3C (Actor-Critic com Múltiplos Agentes) e o TRPO (Otimização de Política com Restrição de Trpo), são usados para abordar problemas contínuos e parcialmente observáveis(35)(36). Além disso, abordagens híbridas que combinam aprendizado por reforço com técnicas de aprendizado supervisionado ou aprendizado por imitação são exploradas para resolver desafios complexos(37).

Problemas de aprendizado por reforço em ambientes contínuos e parcialmente observáveis são de grande importância, especialmente em cenários do mundo real, onde a simplicidade dos problemas discretos raramente se aplica. Abordar esses desafios requer uma combinação de técnicas avançadas e criatividade na pesquisa em aprendizado por reforço.

2.2.3.6 Aplicações Avançadas de Aprendizado por Reforço

A aprendizagem por reforço tem aplicações amplas e impactantes em domínios como robótica autônoma, jogos, finanças e medicina, onde abordagens avançadas têm sido aplicadas para resolver problemas complexos.

- Robótica Autônoma: O aprendizado por reforço desempenha um papel crucial na robótica autônoma, onde os agentes, como robôs, aprendem a realizar tarefas complexas, como navegação, manipulação de objetos e interação com o ambiente(38). Estratégias avançadas, como aprendizado por reforço hierárquico, permitem que os robôs lidem com tarefas multiestágio e ambientes não estruturados.
- Jogos: Os jogos são um campo de teste popular para algoritmos de aprendizado por reforço, incluindo o uso de Aprendizado Profundo por Reforço (DRL). Os agentes aprendem a jogar jogos complexos, como xadrez, Go e jogos de arcade, diretamente dos dados brutos do jogo(27). O DRL demonstrou superar jogadores humanos em vários jogos.
- Medicina: É usado para otimizar tratamentos médicos, como administração de medicamentos, terapia de reabilitação e tratamento personalizado(39). Os agentes de aprendizado por reforço podem adaptar os tratamentos com base nas respostas do paciente e nas mudanças nas condições de saúde.
- Sistemas de Recomendação: Em sistemas de recomendação, os agentes de aprendizado por reforço são usados para personalizar sugestões para os usuários, como recomendações de produtos, conteúdo online e anúncios(40). Esses agentes aprendem com interações passadas e o comportamento do usuário.
- Sistemas de Controle Industrial: No campo do controle industrial, o aprendizado por reforço é aplicado para otimizar processos de fabricação, controle de drones industriais e manutenção preditiva(41). Os agentes aprendem a realizar tarefas complexas em ambientes industriais.
- Aprendizado por Reforço Multi-agente: Em cenários onde vários agentes interagem, como jogos competitivos ou colaborativos, o aprendizado por reforço multi-agente é uma área avançada(5). Os agentes devem aprender a colaborar ou competir com outros agentes, criando estratégias complexas.
- Aprendizado por Reforço em Ambientes Complexos: Em ambientes complexos, como ambientes urbanos para veículos autônomos, ambientes subaquáticos ou espaciais, o

aprendizado por reforço é aplicado para permitir que agentes tomem decisões em cenários desafiadores e dinâmicos(14).

As aplicações avançadas de aprendizado por reforço têm um impacto significativo em uma ampla gama de domínios, desde robótica até medicina e finanças. O campo continua a evoluir à medida que novas abordagens são desenvolvidas para resolver problemas complexos e melhorar a tomada de decisões em cenários do mundo real.

2.2.3.7 Desafios Abertos e Tendências Futuras

- Exploração Eficiente em Ambientes Complexos: Um desafio contínuo é desenvolver estratégias de exploração mais eficientes, especialmente em ambientes complexos, onde o agente deve lidar com um grande número de estados e ações(11).
- Generalização de Aprendizado por Reforço: Tornar os agentes de aprendizado por reforço capazes de generalizar o conhecimento adquirido em um ambiente para tarefas semelhantes em diferentes ambientes é um desafio crucial(14). Isso envolve a criação de políticas que sejam mais transferíveis e menos dependentes de domínio.
- Aprendizado por Reforço Multi-agente: À medida que cenários com múltiplos agentes se tornam mais comuns, lidar com a complexidade das interações entre agentes é um desafio em crescimento(42). A pesquisa futura se concentrará em políticas multi-agente mais sofisticadas e em garantir resultados justos e equilibrados em ambientes competitivos.
- Aprendizado por Reforço com Dados Limitados: Em muitas situações do mundo real, coletar dados de treinamento pode ser caro ou perigoso(14). Portanto, métodos que permitem que os agentes aprendam eficazmente com dados limitados serão uma tendência importante.
- Aprendizado por Reforço com Incerteza e Variabilidade: Muitos ambientes do mundo real são intrinsecamente incertos e variáveis(43). Desenvolver técnicas avançadas para que os agentes aprendam a lidar com essa incerteza será uma área importante de pesquisa.

2.2.3.8 OpenAI Gym

O OpenAI Gym(25) é uma ferramenta para pesquisa em aprendizagem por reforço. Ele oferece uma coleção em crescimento de problemas de referência que expõem uma interface comum e um site onde as pessoas podem compartilhar seus resultados e comparar o desempenho

dos algoritmos. Este artigo discute os componentes do OpenAI Gym e as decisões de design que foram tomadas para o software.

A APR é um ramo da aprendizagem de máquina que se preocupa em tomar sequências de decisões. A RL possui uma rica teoria matemática e encontrou uma variedade de aplicações práticas. Avanços recentes que combinam aprendizagem profunda com aprendizagem por reforço levaram a uma grande empolgação na área, pois ficou evidente que algoritmos gerais, como gradientes de política e Q-learning, podem alcançar bom desempenho em problemas difíceis, sem a necessidade de engenharia específica do problema.

Para avançar no campo da aprendizagem por reforço, a comunidade de pesquisa precisa de bons benchmarks para comparar algoritmos. Vários benchmarks foram lançados, como o Arcade Learning Environment (ALE), que expõe uma coleção de jogos Atari 2600 como problemas de aprendizagem por reforço, e recentemente o benchmark RLLab para controle contínuo. O OpenAI Gym tem como objetivo combinar os melhores elementos dessas coleções anteriores em um pacote de software que seja conveniente e acessível. Ele inclui uma diversidade de tarefas (chamadas de ambientes) com uma interface comum, e essa coleção crescerá com o tempo. Os ambientes são versionados de forma a garantir que os resultados sejam significativos e reproduzíveis à medida que o software é atualizado.

O OpenAI Gym se concentra no cenário episódico da aprendizagem por reforço, onde a experiência do agente é dividida em uma série de episódios. Em cada episódio, o estado inicial do agente é amostrado aleatoriamente a partir de uma distribuição, e a interação ocorre até que o ambiente atinja um estado terminal. O objetivo na aprendizagem por reforço episódico é maximizar a expectativa de recompensa total por episódio e alcançar um alto nível de desempenho em poucos episódios.

O OpenAI Gym foi projetado com base na experiência dos autores no desenvolvimento e comparação de algoritmos de aprendizagem por reforço, e na experiência com coleções de benchmarks anteriores. Algumas decisões de design foram tomadas para maximizar a conveniência dos usuários e permitir diferentes estilos de interface de agente. O foco é na complexidade de amostragem, não apenas no desempenho final, e os ambientes são estritamente versionados para garantir a comparabilidade dos resultados. O OpenAI Gym também incentiva a revisão por pares e a compartilhar de código e ideias entre os usuários.

2.2.3.9 Stable Baselines

O Stable Baselines(44) é uma biblioteca de código aberto projetada para facilitar o desenvolvimento e a experimentação de algoritmos de aprendizado por reforço. Desenvolvida pela equipe do OpenAI(25), essa biblioteca oferece uma coleção de algoritmos de aprendizado por reforço robustos e de fácil utilização, tornando-a uma escolha popular entre pesquisadores e desenvolvedores que buscam abordar problemas de controle e tomada de decisão em ambientes complexos.

Uma das características notáveis do Stable Baselines é a sua simplicidade de uso. Ele oferece uma interface simples e intuitiva que permite aos desenvolvedores implementar algoritmos de aprendizado por reforço em seus projetos com facilidade. Isso significa que os usuários podem se concentrar mais na definição do problema e menos na complexidade de implementação dos algoritmos de aprendizado por reforço.

Além disso, o Stable Baselines inclui uma variedade de algoritmos populares, como Proximal Policy Optimization (PPO), Trust Region Policy Optimization (TRPO), DDPG e muito mais. Isso oferece aos desenvolvedores uma ampla gama de opções para escolher o algoritmo que melhor se adapta às necessidades do seu projeto. Além disso, a biblioteca oferece suporte para ambientes contínuos e discretos, o que a torna flexível o suficiente para lidar com uma variedade de cenários de aprendizado por reforço.

O Stable Baselines também se destaca na escalabilidade, permitindo treinar modelos de aprendizado por reforço em paralelo, aproveitando o poder computacional de GPUs e CPUs. Isso é essencial quando se lida com tarefas complexas que exigem um grande número de interações para atingir o desempenho desejado.

Em resumo, o Stable Baselines se estabeleceu como uma ferramenta poderosa e confiável para projetos de aprendizado por reforço. Sua simplicidade, variedade de algoritmos e escalabilidade tornam-no uma escolha atraente para aqueles que buscam implementar soluções de IA eficazes em uma ampla gama de aplicações, desde controle de robôs até otimização de políticas de tomada de decisão.

2.2.3.10 BahiaRT Gym

O BahiaRT Gym(26) é um ambiente personalizado para o uso do ambiente de ferramentas do OpenAI Gym, projetado para atender às necessidades específicas do ambiente de futebol de robôs simulados com o uso de APR.

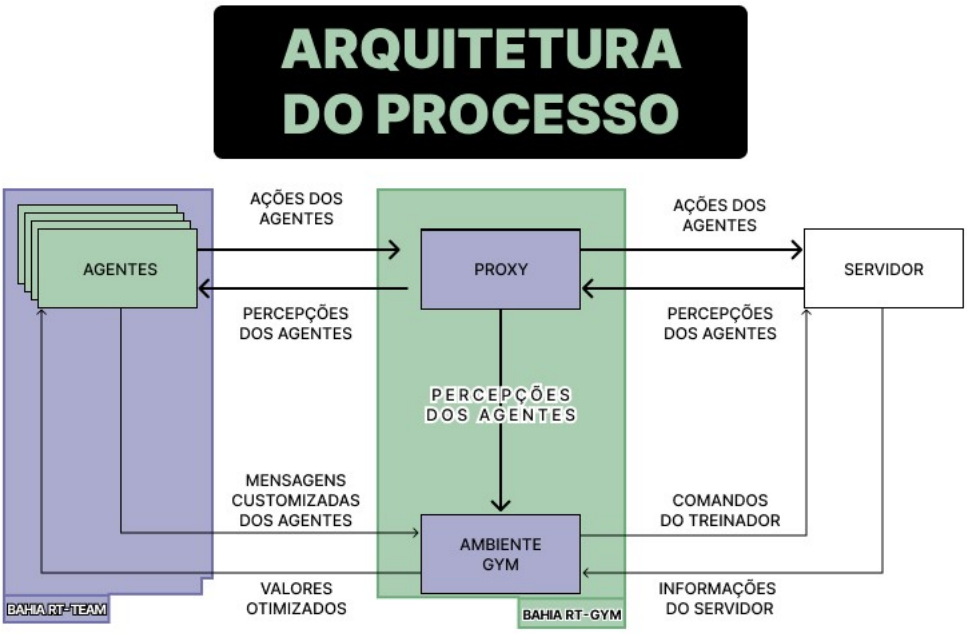
Uma das principais vantagens do BahiaRT Gym é a arquitetura utilizada como base para a sua construção, conforme ilustrado na Figura 2. A arquitetura do BahiaRT Gym pode ser explicada através de 3 componentes:

1. **Time:** Contém o código fonte dos agentes. Cada agente age independentemente e possui comunicação única com o servidor, passando pelo proxy do BahiaRT Gym. Os agentes também possuem comunicação direta com o proxy(sem passar pelo servidor), implementada via socket, para o envio e recebimento de mensagens específicas sobre o treinamento, como qual ação executar ou qualquer outro tipo de mensagem dos agentes.
2. **BahiaRT Gym:** Contém o proxy e o BahiaRT Gym propriamente dito. O proxy, que se baseia no Magma Proxy(45), tem como objetivo coletar dados que trafegam entre os agentes o servidor e garantir a independência de problemas/delays de rede. Deste modo, o proxy do BahiaRT Gym é um substituto do Magma Proxy, porém com melhorias voltadas para a coleta de dados da simulação, como a posição da bola, agentes, tempo de jogo etc. Além disso, o proxy também é utilizado para realizar a comunicação direta com o time através de sockets, permitindo o envio e recebimento de mensagens específicas relacionadas ao treinamento. O outro componente é o BahiaRT Gym, que recebe informações do proxy. No BahiaRT Gym, reside a modelagem completa do ambiente, abrangendo o espaço de observação, as ações disponíveis, o modelo a ser treinado e outros elementos essenciais. Aqui é onde os algoritmos, parâmetros e demais configurações são incorporados. O BahiaRT Gym desempenha um papel vital ao empregar essas informações no treinamento dos agentes, ajustando estratégias com base no feedback acumulado ao longo do tempo. Este é o componente central da arquitetura, visto que é o responsável pelo treinamento do modelo.
3. **Servidor:** O servidor é o componente que gerencia a simulação, incluindo aspectos físicos, controle dos agentes, entre outras funcionalidades. É responsável por manter o ambiente de simulação, garantindo a integridade das interações entre os agentes e as condições do jogo.

O objetivo na construção do BahiaRT Gym foi permitir uma compatibilidade com qualquer linguagem de programação, permitindo que equipes que trabalhem com o futebol de robôs simulados utilizem e contribuam para o desenvolvimento do ambiente sem grandes dificuldades. Isso garante que o BahiaRT Gym seja útil para a comunidade de pesquisa em geral, promovendo a colaboração e o avanço no campo da aprendizagem por reforço.

Além disso, o BahiaRT Gym possui uma arquitetura de comunicação robusta que permite a interação eficiente entre o ambiente, o servidor e os agentes de treinamento, conforme explicado anteriormente. Essa arquitetura de comunicação é essencial para o treinamento e avaliação dos algoritmos de aprendizagem por reforço, proporcionando uma interação contínua entre os diferentes componentes do sistema, visto que as observações provenientes do servidor são necessárias como input para o modelo, e os resultados do modelo precisam ser enviados para os agentes como ações a serem realizadas, como chutar ou andar.

Figura 2 – Arquitetura do BahiaRT Gym



Fonte: O próprio autor

Em resumo, o BahiaRT Gym é um ambiente baseado no OpenAI Gym, desenvolvido de modo personalizado com o objetivo de facilitar a pesquisa e o desenvolvimento em aprendizagem por reforço no ambiente de futebol de robôs.

2.3 SISTEMAS MULTIAGENTES

Os SMA são sistemas compostos por múltiplos agentes que interagem entre si para alcançar objetivos individuais e coletivos. Uma das principais razões para utilizar SMA no projeto de um sistema é quando o domínio em questão exige essa abordagem. Por exemplo, em situações onde diferentes pessoas ou organizações possuem objetivos e informações proprietárias, muitas vezes conflitantes, um sistema multiagente se faz necessário para lidar com as interações entre elas. Cada organização deseja representar seus próprios interesses e não está disposta a ceder o controle a um único sistema que as englobe todas. Portanto, é preciso que cada uma tenha seu próprio sistema, com agentes que reflitam suas capacidades e prioridades, e que esses agentes sejam combinados em um sistema multiagente por meio das técnicas apropriadas(46).

Um exemplo ilustrativo apresentado por Stone e Veloso(46) é o agendamento hospitalar, que envolve interesses diversos de diferentes profissionais. Nesta situação, enfermeiros desejam minimizar o tempo de internação dos pacientes, enquanto operadores de raios-X buscam maximizar a utilização de seus equipamentos. Diante da diferença de cronogramas e objetivos, é necessário representar os dois profissionais como agentes separados dentro de um sistema multiagente, de modo a garantir que os interesses sejam considerados. Este é um exemplo real que trata de objetivos que podem ser conflitantes diante do cenário de pacientes no hospital, quais os problemas relacionados, entre outros.

Além disso, mesmo em domínios que poderiam usar sistemas não distribuídos, existem motivos para adotar SMA. A divisão de tarefas entre múltiplos agentes pode acelerar a operação do sistema, permitindo a computação paralela. Domínios que podem ser divididos em componentes independentes se beneficiam da abordagem multiagente. Além disso, o paralelismo de SMA ajuda a lidar com restrições impostas por requisitos de raciocínio em tempo limitado.

A robustez é outra vantagem dos sistemas multiagentes. Ao compartilhar o controle e as responsabilidades entre diferentes agentes, o sistema se torna capaz de tolerar falhas individuais. Domínios que exigem resiliência se beneficiam dessa característica, pois, em caso de falha de um ou mais agentes, todo o sistema não é afetado. Embora um sistema multiagente não precise necessariamente ser implementado em múltiplos processadores, para obter total robustez contra falhas, os agentes devem estar distribuídos em várias máquinas.

A escalabilidade é outra das vantagens dos sistemas multiagentes. Como esses

sistemas são modularizados por natureza, é mais fácil adicionar novos agentes a um sistema multiagente do que adicionar novas capacidades a um sistema monolítico. Isso torna os SMA mais adequados para sistemas cujas capacidades e parâmetros podem precisar ser alterados ao longo do tempo ou variar entre os agentes.

Do ponto de vista de desenvolvimento, a modularidade dos sistemas multiagentes facilita alguns pontos da programação,como a ientificação de sub tarefas e atribuição de controlesdessas tarefas a diferentes agentes. Portanto, quando a escolha é entre usar um sistema multiagente ou um sistema de agente único, os SMA costumam ser a opção mais simples(16). No entanto, é importante mencionar que existem domínios que são naturalmente abordados de uma perspectiva onisciente, onde uma visão global é fornecida, ou com controle centralizado, onde ações paralelas não são possíveis e não há incerteza nas ações. Nesses casos, sistemas de agente único devem ser utilizados.

Em resumo, os sistemas multiagentes oferecem várias vantagens, como a capacidade de lidar com domínios complexos e com múltiplas entidades com objetivos e informações conflitantes, a possibilidade de computação paralela e escalabilidade, a robustez contra falhas individuais, a modularidade e simplicidade de programação, e o potencial de explorar e compreender a inteligência por meio da interação. Cada domínio e contexto específico deve ser avaliado para determinar se um sistema multiagente é a abordagem mais apropriada.

3 METODOLOGIA

A metodologia de desenvolvimento adotada neste trabalho se baseia em princípios ágeis e na combinação das metodologias Scrum e Kanban para o gerenciamento eficiente do projeto de pesquisa. O Bitbucket será a plataforma escolhida para o versionamento do código-fonte, garantindo um controle efetivo das alterações realizadas.

O projeto começa com a definição clara dos objetivos de pesquisa e do escopo do trabalho. O Scrum é então implementado para gerenciar as atividades operacionais da pesquisa. Isso envolve a organização do trabalho em sprints de curta duração, tipicamente de uma semana, para manter o foco nas metas estabelecidas.

Dentro do Scrum, um fluxo de trabalho é estabelecido com etapas como "A fazer, Em andamento,"e "Concluído."Papéis na equipe são designados, e reuniões semanais são realizadas para planejar, revisar e controlar as atividades. A metodologia Kanban é adaptada para complementar o Scrum e oferecer um controle mais eficiente das tarefas.

Ferramentas online, como o Trello ou similar, são usadas para criar um quadro Kanban virtual, onde todas as tarefas e etapas necessárias no desenvolvimento da pesquisa são organizadas e categorizadas. Além disso, o Bitbucket é a escolha para o versionamento do código-fonte do projeto. Esta plataforma utiliza as funcionalidades do Git para manter um registro detalhado do progresso, permitindo colaboração e controle de alterações no código.

A colaboração é incentivada entre os membros da equipe, permitindo que eles contribuam e registrem alterações no código e na documentação do projeto. O Bitbucket também mantém um histórico detalhado das modificações realizadas ao longo do tempo. Esta metodologia, baseada em princípios ágeis, oferece uma estrutura eficiente para o desenvolvimento e acompanhamento do projeto de pesquisa. A combinação de Scrum e Kanban com o Bitbucket como plataforma de versionamento de código é essencial para manter o controle, garantir colaboração eficiente e documentação detalhada, contribuindo para o sucesso da pesquisa.

3.1 FERRAMENTAL TÉCNICO

O ferramental técnico utilizado para a utilização do time de futebol de robôs envolve uma série de escolhas que foram justificadas de acordo com as necessidades e objetivos do projeto. A seguir, estão descritos os principais componentes com as respectivas justificativas:

1. Linguagem C++: A escolha da linguagem C++ é a utilizada como base para o time de futebol de robôs BahiaRT foi feita devido à sua eficiência e desempenho. O C++ é uma linguagem compilada que permite um controle de baixo nível sobre o hardware, sendo um ponto fundamental para a execução rápida e eficiente de algoritmos em tempo real.
2. RoboViz: O RoboViz é um visualizador gráfico usado para exibir a simulação do jogo de futebol de robôs em tempo real. Ele permite a visualização dos robôs, do campo e das interações entre os agentes, proporcionando uma maneira intuitiva de acompanhar o desempenho do time durante a simulação. Esta é uma das formas qualitativas de validar manualmente o comportamento do time.
3. Scripts em Bash: Os scripts em Bash são utilizados para automatizar tarefas e facilitar a execução do time de futebol de robôs. Eles permitem a configuração e execução do ambiente de simulação de forma simplificada, como por exemplo a execução dos times, reiniciar o ambiente nos casos de fim de uma partida, entre outros.
4. RoboCup Soccer Simulator Server (RCSSServer): O RCSSServer é o simulador do ambiente de futebol de robôs. Ele permite a criação de partidas, controle dos robôs e a comunicação entre o time e o ambiente. A escolha do RCSSServer foi feita por ser o simulador padrão da Robocup(47).
5. BahiaRT Gym: O BahiaRT Gym é um ambiente personalizado desenvolvido para a integração do time de futebol de robôs com o framework de aprendizagem por reforço OpenAI Gym. O BahiaRT Gym foi escolhido por fornecer ferramentas e funcionalidades específicas para o treinamento de agentes usando algoritmos de aprendizagem por reforço, e também pela flexibilidade, adaptabilidade e compatibilidade com diferentes linguagens de programação, permitindo uma integração eficiente e personalizada do time com o ambiente de aprendizagem.
6. RTX 3060: A GPU RTX 3060 foi empregada para conduzir o treinamento dos agentes do time de futebol de robôs. A escolha desta GPU é fundamentada na sua capacidade de processamento robusta e alto desempenho, atendendo às demandas de treinamento

de modelos de inteligência artificial. Destaca-se que essa escolha não apenas acelera significativamente o processo de treinamento, mas também otimiza a execução do ambiente de simulação (RCSSServer).

Essas escolhas foram feitas com base nas características, requisitos e compatibilidade com os demais componentes do sistema, visando proporcionar um ambiente de desenvolvimento e treinamento eficaz para o time de futebol de robôs, além de garantir a interoperabilidade com outros sistemas e a otimização de desempenho necessário para o treinamento de agentes de forma eficiente.

3.2 PASSOS PARA A EXECUÇÃO DO PROJETO

Depois desta breve apresentação do aparato técnico utilizado para o desenvolvimento da solução, abaixo estão listados os passos em detalhes para a execução prática do projeto.

3.2.1 Modelagem do ambiente de aprendizagem

O objetivo da modelagem é definir os elementos essenciais do ambiente, como as funções de recompensa, o campo de observação dos agentes, as ações disponíveis e as condições de início e término de cada episódio de treinamento.

Para definir as funções de recompensa, foi necessário analisar o objetivo do projeto e identificar os comportamentos desejados dos agentes. Por exemplo, os agentes podem receber uma recompensa positiva quando marcam um gol ou realizam uma jogada cooperativa eficiente, e uma recompensa negativa quando cometem faltas ou tomam decisões inadequadas. Essas funções de recompensa são projetadas de forma a incentivar os agentes a adotar comportamentos desejáveis e melhorar seu desempenho coletivo.

O campo de observação dos agentes é uma representação das informações disponíveis para cada agente em um determinado momento. Pode incluir a posição dos jogadores, a posição da bola, a direção do movimento, entre outros. A definição do campo de observação depende da natureza do problema e das informações relevantes para que os agentes tomem decisões adequadas.

As ações disponíveis para os agentes são as escolhas que eles podem fazer em cada momento, como se mover para uma determinada posição, chutar a bola, marcar um jogador

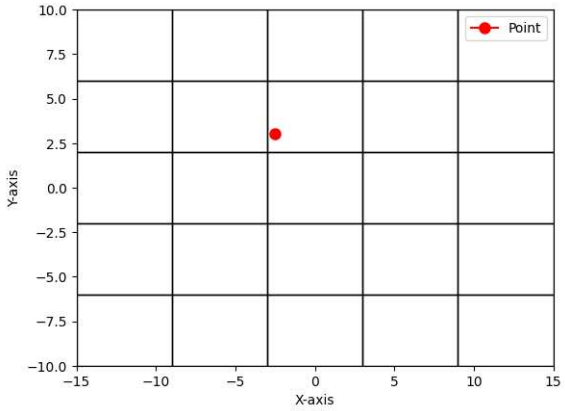
adversário, entre outros. Essas ações são definidas de acordo com as capacidades dos agentes e as regras do jogo.

As condições de início e término de cada episódio de treinamento são definidas para delimitar o tempo e as situações em que o treinamento ocorre. Por exemplo, um episódio pode começar quando a bola está em jogo e terminar quando ocorre um gol, uma falta grave ou quando o tempo estipulado para o episódio se esgota.

Ao modelar o ambiente de aprendizagem, foi importante considerar as características específicas do futebol de robôs 3D e adaptar o ambiente de acordo com as necessidades e objetivos do projeto. A modelagem é um processo iterativo, no qual são feitas análises, ajustes e refinamentos para criar um ambiente que proporcione desafios adequados e possibilite o treinamento eficiente dos agentes.

- Critérios de recompensa: Estabeleceu-se que serão atribuídas pontuações de +10 quando a equipe marcar gols e -10 quando sofrer gols. Quando o agente estiver com a posse de bola (ou seja, quando a bola estiver dentro de um raio de 2m de um dos agentes da equipe), ele receberá +5. Por outro lado, na ausência de posse de bola(bola sem nenhum agente no raio de 2m), a pontuação será de -5. Em situações de conflito pela posse de bola entre os dois times(agentes de times opostos no raio de 2m da bola), a pontuação será -3. Adicionalmente, uma recompensa foi atribuída para refletir a posição em x da bola, variando de -15 a +15, representando de uma ponta do campo à outra. Nesse contexto, quando a bola está no lado defensivo, a recompensa é negativa, e à medida que avança para o campo do oponente, a recompensa aumenta. Essa estratégia de recompensas vinculada à posição da bola oferece aos agentes uma orientação clara sobre o desempenho desejado, incentivando comportamentos defensivos eficazes em seu próprio território e táticas ofensivas ao se aproximarem do gol adversário.
- Espaço de observação: Para o espaço de observação, as posições de todos os aliados e dos oponentes foram definidas a partir de uma matriz, identificando em qual espaço da matriz o agente ou a bola se encontram, conforme exemplo na Figura 3, onde o ponto vermelho é a posição de um dos agentes dentro de um dos espaços da matriz. O eixo x abrange a distância de um gol ao outro (30 unidades no total), enquanto o eixo y representa a extensão entre as laterais do campo (10 unidades no total).
- Espaço de ação: As ações de alto nível, nomeadamente passiveMarker, offensiveMarker e

Figura 3 – Exemplo do espaço de observação



Fonte: O próprio autor

carryBall, foram concebidas para definir comportamentos específicos a serem adotados pelos agentes da equipe. Cada ação representa uma estratégia distinta no contexto do jogo. Aqui está uma breve descrição de cada uma delas:

– passiveMarker (Marcador Passivo):

Esta ação leva o agente a adotar uma postura defensiva ou de marcação mais passiva. O agente pode estar focado em monitorar a movimentação dos jogadores adversários, bloquear potenciais linhas de passe ou manter uma posição estratégica para auxiliar na defesa da equipe.

– offensiveMarker (Marcador Ofensivo):

Ao escolher a ação offensiveMarker, o agente é direcionado a adotar uma postura mais ativa na marcação. Isso pode incluir pressionar os jogadores adversários mais próximos à bola, buscar interceptar passes e perturbar as jogadas ofensivas dos oponentes.

– carryBall (Transportar a Bola):

Essa ação indica que o agente deve se envolver ativamente no transporte da bola. De modo prático, o agente inicia uma jogada de ataque carregando a bola em direção ao gol adversário. Isso implica uma participação ativa no jogo ofensivo da equipe.

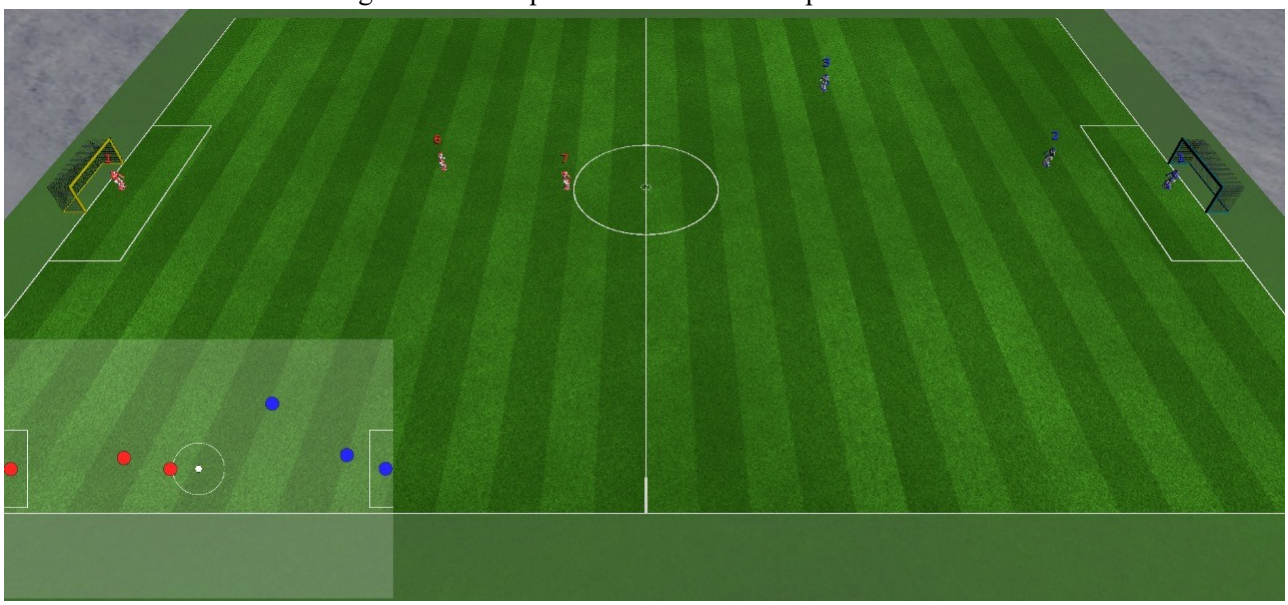
Ao integrar essas ações de alto nível, a equipe pode coordenar estratégias complexas, adaptando-se dinamicamente às mudanças nas condições de jogo. Essas ações proporcionam uma abordagem modular para o comportamento dos agentes, permitindo uma flexibilidade significativa na tomada de decisões durante a simulação do jogo.

- Condições de início e témino: No início da simulação, os agentes são estrategicamente posicionados conforme a configuração padrão de jogo. A conclusão da simulação pode

ocorrer de duas maneiras: após um período máximo de 3 minutos ou instantaneamente quando uma equipe marca um gol ou sofre um gol. Essas condições de início e término estabelecem os parâmetros temporais e os eventos cruciais que moldam o curso da simulação, proporcionando um ambiente controlado para avaliação e análise do comportamento dos agentes no contexto do jogo.

A Figura 4 mostra um exemplo de estado inicial da partida. Para o treinamento, decidiu-se de utilizar um conjunto restrito de apenas 6 agentes, começando com 3 de cada lado(1 goleiro e outros 2 jogadores), em vez de utilizar todos os jogadores, formando um time completo. Vale ressaltar que, durante o treinamento, foram utilizados tanto o código do Magma Offenburg(48) quanto o do BahiaRT, embora apenas os jogadores do BahiaRT estivessem sendo treinados.

Figura 4 – Exemplo de estado inicial da partida



Fonte: O próprio autor

Essa decisão foi fundamentada em considerações estratégicas. Primeiramente, essa abordagem visou controlar a complexidade do ambiente durante as fases iniciais do treinamento, proporcionando uma compreensão mais clara e controlada das interações entre os poucos agentes. Além disso, a limitação no número de agentes contribuiu para uma eficiência computacional, reduzindo os requisitos computacionais iniciais e permitindo uma experimentação mais rápida. A escolha também facilitou a depuração, uma vez que é mais fácil rastrear e corrigir problemas quando o número de agentes é menor. Por fim, a transição gradual de um ambiente menos complexo para um mais desafiador foi considerada, evitando possíveis problemas de convergência

que podem surgir quando muitos agentes são introduzidos simultaneamente. Essa estratégia foi essencial para estabelecer uma base sólida para o treinamento, permitindo uma progressão gradual em direção a cenários mais complexos à medida que o desempenho dos agentes se aprimora.

Outro ponto crucial no processo de treinamento é que apenas um agente da equipe BahiaRT foi submetido ao treinamento. Essa escolha foi influenciada por considerações técnicas associadas à arquitetura do BahiaRT Gym. A configuração atual da ferramenta implica que o proxy envie mensagens simultaneamente a todos os agentes, o que resulta na distribuição idêntica da mesma mensagem para cada um deles. Como resultado, a condução do treinamento para mais de um agente simultaneamente se tornou impraticável. Para treinar vários agentes, seria necessário criar um Multi-agent Particle Envs (MPE), que é uma das ramificações do OpenAI Gym voltada para sistemas multiagentes(42)(49). Somente com modificações que permitissem o controle de informações enviadas e recebidas por cada agente de maneira independente seria viável implementar um MPE.

Uma compreensão mais aprofundada dessa limitação é dada na Seção 5, onde são explorados detalhadamente os desafios enfrentados, além da apresentação de uma proposta construtiva para contribuir com o avanço e aprimoramento contínuo do BahiaRT Gym. Essa análise e proposta visam oferecer insights valiosos para superar as atuais limitações, aprimorando a versatilidade e eficácia da ferramenta em futuras iterações de treinamento.

3.2.2 Implementação do ambiente de aprendizagem

O ambiente de aprendizagem foi implementado, sendo integrado ao time de futebol de robôs BahiaRT. Foram efetuadas modificações no código do BahiaRT para coletar e enviar dados para o ambiente de aprendizagem, além de receber as ações dos agentes treinados. Nessa etapa, um ambiente de treinamento foi criado com base no BahiaRT Gym, seguindo a modelagem do ambiente. A escolha do BahiaRT Gym foi motivada pela sua especificidade para a área de futebol de robôs simulados, permitindo aproveitar recursos e funcionalidades específicas do futebol de robôs 3D, como o campo de observação dos agentes. Vale destacar que o BahiaRT Gym utiliza o OpenAI Gym como base, uma ferramenta amplamente adotada pela comunidade em pesquisas de APR. Além disso, sua integração direta com as bibliotecas e funcionalidades fornecidas pelo OpenAI-Gym facilita a modelagem e implementação do ambiente de aprendizagem.

3.2.3 Treinamento dos agentes

O treinamento dos agentes foi conduzido por meio de algoritmos de aprendizado por reforço, com destaque para o uso dos algoritmos Deep Q-Network (DQN) e Deep Deterministic Policy Gradient (DDPG) neste projeto. No cenário do ambiente de aprendizagem desenvolvido, os agentes foram treinados em episódios simulados de jogo. Em cada episódio, os agentes interagiram com o ambiente, observaram o estado do jogo, tomaram ações com base em suas políticas de tomada de decisão e receberam recompensas conforme o desempenho de suas ações.

O treinamento dos agentes seguiu um processo iterativo, onde realizaram várias rodadas de interação com o ambiente, ajustando suas políticas e aprendendo com os resultados obtidos. Essas iterações foram repetidas até que os agentes alcançassem um desempenho satisfatório de acordo com os critérios predefinidos.

Para o treinamento, foi empregada uma GPU GeForce RTX 3060, aproveitando sua capacidade de processamento paralelo e desempenho otimizado para acelerar as operações intensivas de cálculos envolvidas nos algoritmos de aprendizado por reforço. O uso desta GPU proporcionou uma execução mais rápida e eficaz das iterações de interação dos agentes com o ambiente. Essa escolha estratégica de hardware contribuiu para a otimização do processo de treinamento, resultando em tempos de convergência mais rápidos e, consequentemente, facilitando o desenvolvimento e aprimoramento dos modelos de aprendizado dos agentes.

Neste capítulo, foram abordados aspectos cruciais do projeto, incluindo a modelagem detalhada do ambiente, implementação das estratégias propostas e considerações gerais para o treinamento dos agentes. Além disso, discutimos o ferramental técnico essencial para a execução bem-sucedida do projeto, delineando passos práticos que facilitarão sua replicação e compreensão. No entanto, a completa avaliação e análise dos resultados obtidos serão minuciosamente exploradas no capítulo 4. Ao adentrar nos experimentos e suas implicações, espera-se uma compreensão mais profunda do impacto e eficácia das abordagens adotadas.

4 RESULTADOS

4.1 IMPLEMENTAÇÃO

A implementação do modelo no ambiente do BahiaRT Gym foi concluída com sucesso, proporcionando uma integração eficaz com o time de futebol de robôs BahiaRT. Essa integração permitiu a interação direta dos agentes com o ambiente simulado, criando assim um espaço propício para o refinamento das estratégias de ação dos agentes, bem como o uso dos algoritmos do Stable Baselines.

Também foi implementada a comunicação entre o proxy (26) e o time BahiaRT, que foi estabelecida por meio de sockets, proporcionando uma troca eficiente de informações entre o ambiente de simulação e o modelo. Essa integração via socket desempenha um papel crucial ao facilitar a transmissão de dados, garantindo uma comunicação eficaz durante as interações entre os agentes e o ambiente de aprendizado. O proxy, implementado tanto no time quanto no BahiaRT Gym, atua como um intermediário confiável, permitindo a sincronização e a transferência de informações essenciais para o treinamento e avaliação dos agentes no contexto do futebol de robôs simulado. Essa abordagem reforça a coesão entre o ambiente de simulação e o modelo de aprendizado, proporcionando uma base sólida para o desenvolvimento e aprimoramento das estratégias dos agentes. As ações foram mapeadas no código em C++ do time BahiaRT, garantindo uma sincronia entre o modelo de aprendizado e a execução das ações no ambiente de simulação, contribuindo para o alinhamento das decisões do modelo com as ações executadas pelos agentes no código principal do time BahiaRT.

4.2 TREINAMENTO

O treinamento dos modelos de aprendizado, empregando os algoritmos Deep Deterministic Policy Gradient (DDPG) e Deep Q-Network (DQN). A escolha destes é respaldada pela capacidade desses algoritmos em representar políticas complexas e não lineares(27)(5). No contexto de sistemas multiagentes, onde as estratégias dos agentes podem ser altamente sofisticadas e interdependentes, a flexibilidade na representação das políticas é uma característica essencial. O DQN destaca-se na aprendizagem de funções de valor Q em ambientes com espaços de ações

discretas, sendo uma escolha sólida quando as ações dos agentes podem ser discretizadas(27).

Por outro lado, o DDPG é particularmente adequado para ambientes com espaços de ações contínuas, oferecendo a capacidade de aprender políticas determinísticas (5). Em situações em que os agentes precisam tomar decisões em espaços contínuos de ações, como controlar a velocidade ou direção de movimento de um robô no contexto do futebol de robôs, o DDPG torna-se uma escolha relevante e eficaz. Nesse contexto, é importante destacar que foi necessária a discretização da variável de ação antes do envio para o time BahiaRT. Essa discretização se fez necessária pelo fato de o espaço de ação ser discreto, conforme explicado na modelagem do ambiente de aprendizagem.

No entanto, dado que o ambiente do futebol de robôs muitas vezes impõe ações contínuas, a decisão de criar um ambiente de treinamento inicial utilizando o DDPG é estratégica. A intenção é preparar uma base sólida para a implementação futura do Multi-Agent Deep Deterministic Policy Gradient (MADDPG)(42)(49), uma vez que este é uma extensão do DDPG projetada para sistemas multiagentes. O MADDPG é especialmente eficaz quando há interações complexas entre os agentes, promovendo a cooperação e a aprendizagem de políticas que levam em consideração as ações dos outros agentes no ambiente.

Assim, a escolha de implementar o ambiente de treinamento inicial com DDPG é estratégica. Isso visa criar uma estrutura que pode ser evoluída para o MADDPG quando o desafio de comunicação entre o proxy e os agentes for resolvido. Essa abordagem permite uma transição suave para um modelo mais avançado, mantendo a flexibilidade necessária para lidar com os desafios específicos apresentados pelo ambiente complexo de sistemas multiagentes no futebol de robôs.

A escolha entre DQN e DDPG também é influenciada pela disponibilidade de conhecimento prévio e experiências bem-sucedidas desses algoritmos em problemas semelhantes(5)(27).

Para o processo de treinamento, a GPU GeForce RTX 3060 foi empregada. Os modelos finais foram treinados ao longo de um intervalo de aproximadamente 22 a 26 horas(cada), com 7 a 10 mil episódios para cada algoritmo, desconsiderando a fase de testes. Esta fase de testes ocorreu antes do treinamento dos modelos finais. Nela foram feitos ajustes na modelagem da função de recompensa e do espaço de observação, visando aprimorar o desempenho do modelo e reduzir o tempo de treinamento.

O uso da Mlp Policy(44) para modelar as estratégias dos agentes durante o treinamento dos modelos DDPG e DQN é fundamentado em diversas considerações. Primeiramente, redes neurais perceptron multicamada (MLP) são conhecidas por sua habilidade em aprender representações complexas e não lineares dos dados(50). Em ambientes dinâmicos, como o futebol de robôs simulado, onde as estratégias dos agentes podem ser altamente sofisticadas, essa capacidade de representação é crucial para capturar a complexidade das interações e adaptações necessárias.

Além disso, o futebol de robôs caracteriza-se por um ambiente em constante mudança, onde múltiplos agentes interagem simultaneamente. A MLP Policy, por sua capacidade de adaptação, permite que os agentes ajustem suas estratégias em resposta às variações dinâmicas do ambiente, proporcionando uma resposta eficaz a cenários complexos e dinâmicos.

A escolha da MLP Policy é reforçada pelo seu potencial para aprender hierarquias de representações. Esse aspecto é particularmente valioso quando as estratégias dos agentes envolvem múltiplos níveis de abstração. A capacidade da MLP Policy em aprender representações hierárquicas contribui para a modelagem eficaz de estratégias complexas em ambientes multiagentes.

Além disso, ao seguir configurações convencionais recomendadas para algoritmos de aprendizado por reforço, a escolha da MLP Policy está alinhada com práticas estabelecidas na comunidade de aprendizado de máquina, proporcionando uma abordagem padronizada e comparável. Utilizar a mesma política para ambos os modelos DDPG e DQN facilita a avaliação direta do desempenho, isolando os efeitos do algoritmo em si nas comparações.

Portanto, a justificativa para empregar a MLP Policy reside na sua versatilidade, capacidade de adaptação e habilidade em representar estratégias sofisticadas, fatores essenciais para enfrentar os desafios inerentes ao ambiente complexo do futebol de robôs simulado(50).

Para a avaliação dos modelos, foram utilizadas métricas como taxa de vitórias, recompensa acumulada, número de gols marcados e sofridos. Os resultados apresentados na seção a seguir refletem o desempenho dos melhores modelos treinados, selecionados com base na maximização da recompensa acumulada. Essa abordagem assegura que as métricas refletem o rendimento ótimo alcançado pelos agentes em relação à função de recompensa e às ações tomadas durante o treinamento.

4.3 ANÁLISE QUANTITATIVA

Para avaliar o desempenho dos modelos treinados, foram conduzidos testes consistentes com 21 partidas de 5 minutos cada. Estes testes foram realizados em confronto direto contra o time Magma Offenburg(48), um oponente reconhecido pelo seu desempenho competitivo, estando entre os primeiros em diversos anos da RoboCup(47). Importante ressaltar que todos os testes foram conduzidos mantendo as condições do cenário de treinamento 3x3, consistente com o ambiente utilizado durante o processo de treinamento dos agentes. Este cenário específico foi escolhido para garantir uma transição fluida entre o ambiente de treinamento e o ambiente de teste, promovendo a generalização eficaz das estratégias aprendidas pelos modelos. Para a avaliação dos resultados, as métricas mencionadas anteriormente foram empregadas. A coleta dessas métricas foi realizada tanto utilizando o código original do BahiaRT quanto com o agente treinado.

Algoritmo	Vitórias	Derrotas	Empates	Recompensa Acumulada	Gols BahiaRT	Gols Oponente
Código base	5	10	6	-57	9	18
DDPG	3	3	15	171	8	7
DQN	6	1	14	586	9	2

Tabela 1 – Resultados dos Algoritmos

Com base nos dados coletados na Tabela 1, observou-se uma melhoria significativa no comportamento defensivo da equipe ao analisar a quantidade de empates e gols sofridos em comparação com o código base. A introdução de recompensas negativas à medida que o oponente avança no campo permitiu que os agentes utilizassem eficientemente as marcações para impedir o avanço do time adversário. No entanto, não foi identificado um impacto significativo na quantidade de gols marcados pelo time BahiaRT, e nem na quantidade de vitórias. Essa lacuna pode ser atribuída à falta de comportamentos ofensivos e chutes entre as ações disponíveis para os agentes. Uma possível explicação para os resultados observados é uma tendência a uma maior posse de bola, uma vez que a função de recompensa atribui retornos positivos ao elevar o valor de x (avanço da bola). Assim, com os valores mais elevados de recompensa obtidos através do treinamento com DDPG e DQN, é possível que a posse de bola do BahiaRT tenha experimentado um aumento significativo. Essa tendência pode ser atribuída à estratégia dos agentes em evitar o avanço e chutes por parte do oponente, ações que, de acordo com a modelagem da função de recompensa, poderiam reduzir significativamente os retornos positivos acumulados. Embora

não disponhamos de dados concretos e métricas que comprovem inequivocamente um aumento na posse de bola por parte do time, é possível sugerir que a melhoria nas recompensas obtidas pode estar correlacionada a uma potencial maior posse de bola. Essa hipótese sugere uma possível relação entre o desempenho do time e a posse de bola, embora não seja uma afirmação definitiva de causa e efeito. Esta análise ressalta a importância crucial da função de recompensa na formação dos comportamentos dos agentes, indicando a possibilidade de que a maximização da posse de bola tenha sido uma estratégia potencialmente eficaz para otimizar as recompensas durante os treinamentos, embora não possamos afirmar de maneira conclusiva devido à falta de dados específicos e métricas. Em suma, a análise dos resultados destaca o impacto positivo nas estratégias defensivas, enquanto aponta para a necessidade de considerar e aprimorar as táticas ofensivas na próxima fase de desenvolvimento.

5 CONCLUSÃO E TRABALHOS FUTUROS

Com base nas investigações realizadas ao longo deste trabalho, é possível delinear conclusões significativas sobre o impacto e a eficácia do Aprendizado por Reforço APR em sistemas multiagentes SMA no contexto do futebol de robôs simulados. As perguntas norteadoras estabelecidas proporcionaram uma análise abrangente das aplicações práticas, desafios enfrentados, interações entre agentes autônomos, escalabilidade de abordagens de APR e vantagens observadas na integração do aprendizado por reforço para aprimorar o desempenho das equipes de futebol de robôs.

Os resultados obtidos evidenciam uma melhora significativa no desempenho defensivo da equipe de futebol de robôs, na posse de bola e no avanço no campo do oponente. Essas melhorias são refletidas através das recompensas obtidas pelos agentes, indicando uma adaptação bem-sucedida das estratégias aprendidas durante o treinamento.

O projeto, centrado no uso dos algoritmos DDPG e DQN para aprendizagem por reforço em sistemas multiagentes no contexto do futebol de robôs simulados, demonstrou ser uma abordagem eficaz. A escolha desses algoritmos, orientada pelos espaços de observação e ação no ambiente, revelou-se adequada para lidar com as complexidades do cenário dinâmico do futebol de robôs.

No entanto, é crucial reconhecer tanto os benefícios quanto as limitações associadas ao uso de DDPG e DQN. O DDPG, por sua natureza determinística e sua capacidade de lidar com espaços de ação contínuos, oferece vantagens notáveis em ambientes complexos como o futebol de robôs. Sua estabilidade e eficácia na convergência para políticas ótimas são pontos positivos a serem considerados. Por outro lado, sua sensibilidade a hiperparâmetros e a necessidade de cuidadosa sintonia podem representar desafios práticos.

Já o DQN, ao ser aplicável a espaços de ação discretos, apresenta uma solução sólida para certos contextos, garantindo uma exploração mais estável e eficiente. Contudo, a sua complexidade aumenta significativamente em ambientes com espaços de ação contínuos, o que pode limitar sua aplicabilidade em cenários mais desafiadores.

Dessa forma, a escolha entre DDPG e DQN deve levar em consideração as características específicas do ambiente e as demandas do problema em questão, ponderando os benefícios e as limitações inerentes a cada algoritmo.

Além disso, é fundamental observar que tanto o DDPG quanto o DQN possuem uma limitação intrínseca, uma vez que operam com conhecimento limitado durante o treinamento. Ambos os algoritmos têm acesso apenas ao estado e à recompensa do agente treinado, o que pode restringir sua capacidade de modelar efetivamente interações complexas e dependências entre os diferentes agentes presentes no ambiente.

Essa limitação torna-se evidente quando comparada a abordagens mais avançadas, como o MADDPG. O MADDPG, ao contrário do DDPG e do DQN, permite que cada agente tenha conhecimento não apenas de seu próprio estado e recompensa, mas também da observação e da ação dos outros agentes no ambiente(42). Essa troca de informações entre os agentes proporciona uma representação mais completa e contextualizada do ambiente, facilitando a modelagem de estratégias cooperativas e competitivas.

Portanto, ao considerar a escolha entre DDPG, DQN e outras abordagens, é crucial ponderar não apenas as características do ambiente, mas também o nível de informação compartilhado entre os agentes. Enquanto o DDPG e o DQN oferecem soluções sólidas, o MADDPG destaca-se pela sua capacidade de capturar interações mais sofisticadas em sistemas multiagentes, possibilitando uma compreensão mais holística do ambiente de treinamento.

Além disso, a análise das interações entre agentes em ambientes dinâmicos e competitivos forneceu insights valiosos sobre os fatores que influenciam o comportamento dos agentes autônomos. Um exemplo concreto desses insights revelou que, na prática, os agentes utilizaram o comportamento carryBall para aumentar as recompensas. Por outro lado, quando a bola estava sob posse do time adversário ou de outro aliado, observou-se uma tendência consistente de os agentes executarem comportamentos defensivos de forma proativa, visando prevenir a tomada da bola pelo outro time. Essas estratégias específicas destacam a capacidade dos agentes em adaptar suas ações com base nas condições do jogo, evidenciando a eficácia da aprendizagem por reforço na geração de comportamentos inteligentes e estratégicos. As melhorias observadas no desempenho defensivo, na posse de bola e no avanço no campo indicam que as estratégias aprendidas são aplicáveis de forma efetiva durante as partidas.

Ao abordar a escalabilidade das abordagens de APR para equipes maiores de futebol de robôs, identificou-se desafios específicos associados a essa ampliação. Um exemplo prático disso seria possivelmente a necessidade de aumentar a quantidade de segmentos do campo para aumentar a precisão da identificação das posições dos agentes, conforme a Figura 3. Esse aumento na granularidade do campo é crucial para lidar com a complexidade adicional introduzida por uma equipe maior, proporcionando uma representação mais precisa do ambiente de jogo.

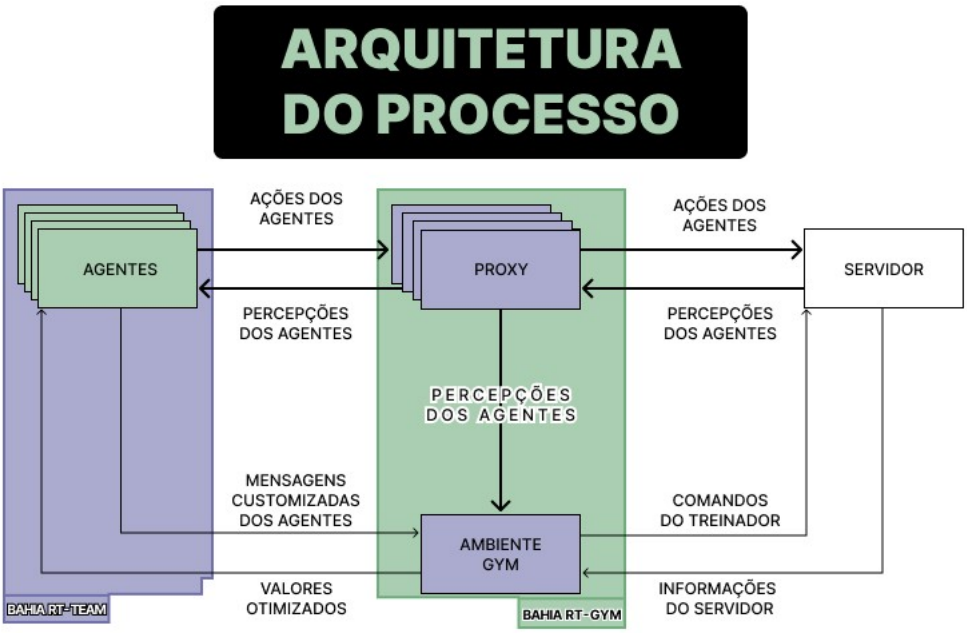
Além disso, o espaço de observação seria significativamente expandido, já que teria que comportar todos os agentes aliados e oponentes presentes no campo. Essa expansão no espaço de observação implica em um aumento substancial na dimensionalidade dos dados, tornando necessário um tempo de treinamento mais prolongado para os modelos absorverem as complexas interações em jogo. Essa demanda por maior capacidade computacional pode resultar na necessidade de hardware mais robusto, destacando a importância de recursos computacionais ao escalar as abordagens de APR para equipes de maior porte no contexto do futebol de robôs. Esses desafios identificados sublinham a necessidade de uma abordagem cuidadosa e adaptativa ao lidar com a escalabilidade de sistemas multiagentes em ambientes mais complexos. No entanto, os resultados positivos obtidos até o momento sugerem que, com considerações cuidadosas e ajustes, essas abordagens podem ser estendidas para equipes de maior porte.

Em suma, este trabalho oferece uma contribuição substancial para a compreensão e aplicação da APR em sistemas multiagentes no contexto do futebol de robôs. As melhorias observadas no desempenho das equipes indicam não apenas a viabilidade, mas também o potencial impacto positivo dessas abordagens no avanço da pesquisa e desenvolvimento nesse campo.

Ao adentrar a prática com o BahiaRT Gym, foi identificada uma potencial área de aprimoramento em sua arquitetura, sendo este trabalho uma contribuição significativa para a comunidade, buscando endereçar e otimizar aspectos identificados durante a aplicação prática do framework. Atualmente, o BahiaRT Gym estabelece uma conexão única entre o servidor, os agentes e o ambiente por meio de um único proxy. Contudo, a estrutura de comunicação entre os agentes e o ambiente revela limitações, onde as mensagens recebidas por um agente são compartilhadas com todos os demais, resultando em uma falta de eficiência de desempenho e independência na interação com o proxy. Neste contexto, uma sugestão de aprimoramento

na arquitetura seria implementar comunicação independente entre agentes e proxy, para que não aconteça o compartilhamento de mensagens entre os ambientes de treinamento dos agentes, conforme apresentado na figura 5. Essa modificação permitiria o controle individualizado de cada proxy, promovendo uma comunicação mais eficaz e independente entre os agentes e o ambiente, contribuindo significativamente para aprimorar a eficiência e flexibilidade do sistema como um todo. Este ajuste na arquitetura proporcionaria uma abordagem mais robusta e adequada às exigências específicas do treinamento de agentes em ambientes complexos de aprendizado por reforço.

Figura 5 – Proposta de arquitetura para o BahiaRT Gym



Fonte: O próprio autor

Essa limitação impacta negativamente a aplicação do algoritmo MADDPG(42)(49), que representa o estado da arte em sistemas multiagentes. Neste algoritmo, o modelo precisa receber como entrada os estados de observação, ações e recompensas de todos os agentes durante o treinamento. A situação atual inviabiliza o treinamento e o processo de receber e enviar ações, exigindo uma implementação mais robusta na comunicação. É crucial estabelecer um sistema no qual cada agente receba apenas sua mensagem correspondente, permitindo assim que cada agente execute suas ações de maneira independente.

Outro ponto crucial a ser considerado é a velocidade de simulação. Ao operar o time sem o proxy, nota-se que a velocidade de simulação atinge, no mínimo,500% (5 vezes o tempo

real), considerando a presença dos 6 agentes em campo. Por outro lado, ao incorporar o proxy, a eficiência cai para 150%, aproximadamente equivalente ao tempo real (100%). Essa discrepância na velocidade de simulação destaca a influência significativa do proxy no desempenho do sistema.

A solução para esse desafio se depara com o dilema de remover o proxy, e implementar o BahiaRT Gym diretamente no código do time. No entanto, essa abordagem comprometeria a modularização, que é essencial para permitir que outros times da comunidade aproveitem a ferramenta. Considerando esse impasse, uma proposta viável para otimizar o treinamento seria a paralelização, possibilitando a execução simultânea de múltiplos servidores e instâncias do time. Essa abordagem apresenta-se como um ponto estratégico para aprimorar significativamente o BahiaRT Gym, proporcionando ganhos substanciais de eficiência no treinamento.

Em última análise, este trabalho se destaca pelo pioneirismo na área, ao optar por utilizar a aprendizagem por reforço para o processo decisório(alto nível) em detrimento de abordagens tradicionais, como o aprendizado de caminhada e chutes(baixo nível). Essa escolha inovadora representa uma contribuição única para o campo, explorando o potencial da aprendizagem por reforço na melhoria do desempenho dos jogadores de futebol de robôs na escolha de comportamentos. Vale ressaltar que, até o momento desta pesquisa, eram escassos os trabalhos dedicados à aplicação de políticas de decisão baseadas em aprendizagem por reforço no contexto do futebol de robôs. Portanto, este estudo não apenas destaca-se pelo pioneirismo na adoção dessa abordagem, mas também preenche uma lacuna significativa na literatura, apresentando novas perspectivas e impulsionando o avanço do conhecimento em sistemas multiagentes aplicados ao cenário desafiador do futebol de robôs.

Portanto, ao evidenciar melhorias práticas e identificar desafios específicos, este trabalho oferece insights valiosos que podem orientar futuras pesquisas e desenvolvimentos nesta área em constante evolução. A adaptação contínua e o refinamento das abordagens de APR, impulsionados pelos resultados e desafios observados, são cruciais para a realização do potencial pleno dessas técnicas inovadoras no contexto do futebol de robôs simulados.

REFERÊNCIAS

1 RUSSELL, S. J. **Artificial intelligence a modern approach**. [S.l.]: Pearson Education, Inc., 2010.

2 HOEK, W. Van der; WOOLDRIDGE, M. Multi-agent systems. **Foundations of Artificial Intelligence**, Elsevier, v. 3, p. 887–928, 2008.

3 DUDEK, G.; JENKIN, M. R.; MILIOS, E.; WILKES, D. A taxonomy for multi-agent robotics. **Autonomous Robots**, Springer, v. 3, p. 375–397, 1996.

4 GUO, J.; CHEN, Y.; HAO, Y.; YIN, Z.; YU, Y.; LI, S. Towards comprehensive testing on the robustness of cooperative multi-agent reinforcement learning. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2022. p. 115–122.

5 YANG, R.; LI, S.; JIN, B. A new approach to training multiple cooperative agents for autonomous driving. In: IEEE. **2022 International Joint Conference on Neural Networks (IJCNN)**. [S.l.], 2022. p. 1–8.

6 JING, Y.; ZHIQIANG, D.; JIEMAI, G.; SIYUAN, C.; BING, Z.; YAJIE, W. Coordinated control strategy of electricity-heat-gas integrated energy system considering renewable energy uncertainty and multi-agent mixed game. **Frontiers in Energy Research**, Frontiers Media SA, v. 10, p. 943213, 2022.

7 CHIDUME, C. S.; NNAMANI, C. O. Intelligent user-collaborative edge device apc-based mec 5g iot for computational offloading and resource allocation. **Journal of Parallel and Distributed Computing**, Elsevier, v. 169, p. 286–300, 2022.

8 BAI, Y.; GONG, C.; ZHANG, B.; FAN, G.; HOU, X.; LU, Y. Cooperative multi-agent reinforcement learning with hypergraph convolution. In: IEEE. **2022 International Joint Conference on Neural Networks (IJCNN)**. [S.l.], 2022. p. 1–8.

9 KAR, S.; MOURA, J. M.; POOR, H. V. Distributed reinforcement learning in multi-agent networks. In: IEEE. **2013 5th IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)**. [S.l.], 2013. p. 296–299.

10 SALAMEH, H. B.; ALHAFNAWI, M.; MASADEH, A.; JARARWEH, Y. Federated reinforcement learning approach for detecting uncertain deceptive target using autonomous dual uav system. **Information Processing & Management**, Elsevier, v. 60, n. 2, p. 103149, 2023.

11 SUTTON, R. S.; BARTO, A. G. **Reinforcement learning: An introduction**. [S.l.]: MIT press, 2018.

12 XUAN, P.; LESSER, V. Multi-agent policies: from centralized ones to decentralized ones. In: **Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 3**. [S.l.: s.n.], 2002. p. 1098–1105.

13 VAMVOUDAKIS, K. G.; KOKOLAKIS, N.-M. T. et al. Synchronous reinforcement learning-based control for cognitive autonomy. **Foundations and Trends® in Systems and Control**, Now Publishers, Inc., v. 8, n. 1–2, p. 1–175, 2020.

14 XU, J.; HUANG, F.; WU, D.; CUI, Y.; YAN, Z.; ZHANG, K. Deep reinforcement learning based multi-auvs cooperative decision-making for attack–defense confrontation missions. **Ocean Engineering**, Elsevier, v. 239, p. 109794, 2021.

15 YANG, Y.-C. E.; CAI, X.; STIPANOVIĆ, D. M. A decentralized optimization algorithm for multiagent system–based watershed management. **Water resources research**, Wiley Online Library, v. 45, n. 8, 2009.

16 ZHANG, K.; YANG, Z.; BAŞAR, T. Decentralized multi-agent reinforcement learning with networked agents: Recent advances. **Frontiers of Information Technology & Electronic Engineering**, Springer, v. 22, n. 6, p. 802–814, 2021.

17 TURING, A. M. **Computing machinery and intelligence**. [S.l.]: Springer, 2009.

18 HAUGELAND, J. **Artificial intelligence: The very idea**. [S.l.]: MIT press, 1989.

19 BELLMAN, R. E. Artificial intelligence: can computers think? **(No Title)**, 1978.

20 KURZWEIL, R.; RICHTER, R.; KURZWEIL, R.; SCHNEIDER, M. L. **The age of intelligent machines**. [S.l.]: MIT press Cambridge, 1990. v. 580.

21 NILSSON, N. J. **Artificial intelligence: a new synthesis**. [S.l.]: Morgan Kaufmann, 1998.

22 ZHOU, Z.-H. **Machine learning**. [S.l.]: Springer Nature, 2021.

23 MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre aprendizado de máquina. **Sistemas inteligentes-Fundamentos e aplicações**, v. 1, n. 1, p. 32, 2003.

24 PUTERMAN, M. L. Markov decision processes. **Handbooks in operations research and management science**, Elsevier, v. 2, p. 331–434, 1990.

25 BROCKMAN, G.; CHEUNG, V.; PETTERSSON, L.; SCHNEIDER, J.; SCHULMAN, J.; TANG, J.; ZAREMBA, W. Openai gym. **arXiv preprint arXiv:1606.01540**, 2016.

26 SIMÕES, M. A.; MASCARENHAS, G.; FONSECA, R.; SANTOS, V. M. dos; MASCARENHAS, F.; NOGUEIRA, T. BahiaRT Setplays Collecting Toolkit and BahiaRT Gym. **Software Impacts**, v. 14, p. 100401, nov. 2022. ISSN 26659638. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S2665963822000938>.

27 MNIH, V.; KAVUKCUOGLU, K.; SILVER, D.; GRAVES, A.; ANTONOGLOU, I.; WIERSTRA, D.; RIEDMILLER, M. Playing atari with deep reinforcement learning. **arXiv preprint arXiv:1312.5602**, 2013.

28 LILLICRAP, T. P.; HUNT, J. J.; PRITZEL, A.; HEESS, N.; EREZ, T.; TASSA, Y.; SILVER, D.; WIERSTRA, D. Continuous control with deep reinforcement learning. **arXiv preprint arXiv:1509.02971**, 2015.

29 SILVER, D.; LEVER, G.; HEESS, N.; DEGRIS, T.; WIERSTRA, D.; RIEDMILLER, M. Deterministic policy gradient algorithms. In: PMLR. **International conference on machine learning**. [S.l.], 2014. p. 387–395.

30 WILLIAMS, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. **Machine learning**, Springer, v. 8, p. 229–256, 1992.

31 SZEPESVÁRI, C. **Algorithms for reinforcement learning**. [S.l.]: Springer Nature, 2022.

32 SUTTON, R. S.; PRECUP, D.; SINGH, S. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. **Artificial intelligence**, Elsevier, v. 112, n. 1-2, p. 181–211, 1999.

33 SCHULMAN, J.; WOLSKI, F.; DHARIWAL, P.; RADFORD, A.; KLIMOV, O. Proximal policy optimization algorithms. **arXiv preprint arXiv:1707.06347**, 2017.

34 AL-SHEDIVAT,M.; BANSAL, T.; BURDA, Y.; SUTSKEVER, I.; MORDATCH, I.; ABBEEL, P. Continuous adaptation via meta-learning in nonstationary and competitive environments. **arXiv preprint arXiv:1710.03641**, 2017.

35 MNIH, V.; BADIA, A. P.; MIRZA, M.; GRAVES, A.; LILLICRAP, T.; HARLEY, T.; SILVER, D.; KAVUKCUOGLU, K. Asynchronous methods for deep reinforcement learning. In: PMLR. **International conference on machine learning**. [S.l.], 2016. p. 1928–1937.

36 SCHULMAN, J.; LEVINE, S.; ABBEEL, P.; JORDAN, M.; MORITZ, P. Trust region policy optimization. In: PMLR. **International conference on machine learning**. [S.l.], 2015. p. 1889–1897.

37 SILVER, D.; HUANG, A.; MADDISON, C. J.; GUEZ, A.; SIFRE, L.; DRIESSCHE, G. V. D.; SCHRITTWIESER, J.; ANTONOGLOU, I.; PANNEERSHELVAM, V.; LANCTOT, M. et al. Mastering the game of go with deep neural networks and tree search. **nature**, Nature Publishing Group, v. 529, n. 7587, p. 484–489, 2016.

38 DIANKOV, R.; KUFFNER, J. Openrave: A planning architecture for autonomous robotics. **Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-08-34**, v. 79, 2008.

39 CORONATO, A.; NAEEM, M.; PIETRO, G. D.; PARAGLIOLA, G. Reinforcement learning for intelligent healthcare applications: A survey. **Artificial Intelligence in Medicine**, Elsevier, v. 109, p. 101964, 2020.

40 AFSAR, M. M.; CRUMP, T.; FAR, B. Reinforcement learning based recommender systems: A survey. **ACM Computing Surveys**, ACM New York, NY, v. 55, n. 7, p. 1–38, 2022.

41 NIAN, R.; LIU, J.; HUANG, B. A review on reinforcement learning: Introduction and applications in industrial process control. **Computers & Chemical Engineering**, Elsevier, v. 139, p. 106886, 2020.

42 LOWE, R.; WU, Y. I.; TAMAR, A.; HARB, J.; ABBEEL, O. P.; MORDATCH, I. Multi-agent actor-critic for mixed cooperative-competitive environments. **Advances in neural information processing systems**, v. 30, 2017.

43 RUSSELL, S. Learning agents for uncertain environments. In: **Proceedings of the eleventh annual conference on Computational learning theory**. [S.l.: s.n.], 1998. p. 101–103.

44 RAFFIN, A.; HILL, A.; GLEAVE, A.; KANERVISTO, A.; ERNESTUS, M.; DORMANN, N. Stable-baselines3: Reliable reinforcement learning implementations. **The Journal of Machine Learning Research**, JMLRORG, v. 22, n. 1, p. 12348–12355, 2021.

45 MAGMAOFFENBURG. **Magma Proxy**. 2023. Disponível em: <https://github.com/magmaOffenburg/magmaProxy>.

46 STONE, P.; VELOSO, M. Multiagent systems: A survey from a machine learning perspective. **Autonomous Robots**, Springer, v. 8, p. 345–383, 2000.

47 ROBOCUP. **Robot Wolrd Cup**. 2017. Disponível em: <http://www.robocup.org/>.

48 MAGMAOFFENBURG. **Magma Release**. 2022. Retrieved June 29, 2023, from https://github.com/magmaOffenburg/magmaRelease.

49 MORDATCH, I.; ABBEEL, P. Emergence of grounded compositional language in multi-agent populations. **arXiv preprint arXiv:1703.04908**, 2017.

50 DESAI, M.; SHAH, M. An anatomization on breast cancer detection and diagnosis employing multi-layer perceptron neural network (mlp) and convolutional neural network (cnn). **Clinical eHealth**, Elsevier, v. 4, p. 1–11, 2021.