UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO DE CIÊNCIAS, TECNOLOGIAS E SAÚDE DO CAMPUS ARARANGUÁ
CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

Raul Brum Weschenfelder

Implementação de um Agente Inteligente para o Controle de um Veículo Autônomo com Decisões Guiadas por Princípios Éticos

Araranguá
2024

Raul Brum Weschenfelder

Implementação de um Agente Inteligente para o Controle de um Veículo Autônomo com Decisões Guiadas por Princípios Éticos

Trabalho de Conclusão de Curso do Curso de Graduação em Engenharia de Computação submetido ao Centro de Ciências, Tecnologias e Saúde do Campus Araranguá da Universidade Federal de Santa Catarina para a obtenção do título de Bacharel em Engenharia de Computação.
Orientador: Prof. Anderson Luiz Fernandes Perez, Dr.

Araranguá
2024

Ficha catalográfica gerada por meio de sistema automatizado gerenciado pela BU/UFSC.
Dados inseridos pelo próprio autor.

Weschenfelder, Raul Brum
Implementação de um Agente Inteligente para o Controle de um Veículo Autônomo com Decisões Guiadas por Princípios Éticos / Raul Brum Weschenfelder ; orientador, Anderson Luiz Fernandes Perez, 2024.
25 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Campus Araranguá, Graduação em Engenharia de Computação, Araranguá, 2024.

Inclui referências.

1. Engenharia de Computação. 2. aprendizado por reforço. 3. inteligência artificial. 4. aprendizado profundo. 5. ética. I. Perez, Anderson Luiz Fernandes. II. Universidade Federal de Santa Catarina. Graduação em Engenharia de Computação. III. Título.

Raul Brum Weschenfelder

Implementação de um Agente Inteligente para o Controle de um Veículo Autônomo com Decisões Guiadas por Princípios Éticos

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do Título de Bacharel em Engenharia de Computação e aprovado em sua forma final pelo Curso de Graduação em Engenharia de Computação.

Araranguá, 26 de Junho de 2024.

Documento assinado digitalmente
Jim Lau
Data: 02/07/2024 13:54:05-0300
CPF: ***.464.702-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Jim Lau, Dr.
Coordenador do Curso

Banca Examinadora:

Documento assinado digitalmente
Anderson Luiz Fernandes Perez
Data: 02/07/2024 13:01:21-0300
CPF: ***.803.459-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Anderson Luiz Fernandes Perez, Dr.
Orientador

Documento assinado digitalmente
Alison Roberto Panisson
Data: 02/07/2024 12:55:19-0300
CPF: ***.887.420-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Alison Roberto Panisson, Dr.
Avaliador
UFSC

Documento assinado digitalmente
Cristian Cechinel
Data: 02/07/2024 14:26:57-0300
CPF: ***.628.449-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Cristian Cechinel, Dr.
Avaliador
UFSC

Implementação de um Agente Inteligente para o Controle de um Veículo Autônomo com Decisões Guiadas por Princípios Éticos

Raul Brum Weschenfelder *

2024, JUNHO

Resumo

Com o avanço da tecnologia, sistemas inteligentes com algum nível de autonomia são cada vez mais comuns no cotidiano. As decisões tomadas por tais sistemas são voltadas a otimizar seus resultados nas tarefas onde são aplicados, muitas vezes sem considerar valores humanos como ética. O conceito de aplicar ética na inteligência artificial, afim de regular o seu comportamento, apresenta diversas dificuldades em suas diferentes formas de implementação. A natureza dinâmica e complexa da ética faz com que a derivação de regras, ou a modelagem de um sistema ético seja uma tarefa difícil de ser executada pelo desenvolvedor. Neste trabalho, é proposta uma abordagem que utiliza aprendizado por reforço e aprendizado profundo para treinar agentes que consideram ética na sua tomada de decisão. A abordagem proposta implementa aprendizado profundo por reforço utilizando o algoritmo *Deep Q-Learnig* para treinar um agente com o objetivo de navegar um ambiente com dilemas éticos referentes as *Três Leis da Robótica*. Os dados de treinamento do agente guiado por ética são comparados a os de um agente comum afim de avaliar os impactos da introdução de ética no aprendizado do agente.

Palavras-chaves: aprendizado por reforço, inteligência artificial, ética.

*raul.weschenfelder@gmail.com

Implementation of an Intelligent Agent for Autonomous Vehicle Control with Ethically Guided Decision-Making

Raul Brum Weschenfelder *

2024, JUNE

Abstract

With technological advancements, intelligent systems with some level of autonomy are becoming increasingly common in everyday life. The decisions made by such systems aim to optimize their performance in their respective tasks, often without considering human values like ethics. The concept of applying ethics to artificial intelligence, in order to regulate its behavior, presents various challenges in its different forms of implementation. The dynamic and complex nature of ethics makes deriving rules or modeling an ethical system a difficult task for developers. This present work proposes an approach that uses reinforcement learning and deep learning to train agents that incorporate ethical considerations in their decision-making process. The proposed approach implements deep reinforcement learning using the *Deep Q-Learning* algorithm to train an agent with the goal of navigating an environment with ethical dilemmas related to the *Three Laws of Robotics*. The training data of the ethically guided agent is compared to that of a standard agent to evaluate the impacts of introducing ethics into the agent's learning process.

Key-words: reinforcement learning, artificial intelligence, deep learning, ethics.

*raul.weschenfelder@gmail.com

1 Introdução

A presença e a interação com sistemas que utilizam inteligência artificial (IA) têm se tornado cada vez mais comum, refletindo a crescente integração dessas tecnologias em diversas áreas, como transporte, medicina e forças armadas. Com a evolução da tecnologia, é natural que esses sistemas, dependendo de seu propósito, adquiram níveis variados de autonomia. No entanto, a autonomia dos sistemas inteligentes, quando não regida por condições legais, ou pelas regras do contexto onde estão inseridas, é orientada pela maximização de resultados em prol de seus objetivos, muitas vezes sem considerar valores humanos como moralidade ou ética.

A introdução de princípios éticos que regulam o comportamento de agentes autônomos não é um conceito novo. Em 1942, Isaac Asimov, em suas obras de ficção científica, apresentou as Três Leis da Robótica (ASIMOV, 1942), um conjunto de regras destinadas a limitar as ações das máquinas, impedindo-as de tomar decisões moralmente inaceitáveis para alcançar seus objetivos.

Com a crescente popularidade dos carros autônomos, a necessidade de incorporar ética na tomada de decisão de sistemas de IA se tornou ainda mais popular. Embora os sistemas de direção autônoma possam ser extremamente eficientes e, em alguns casos, mais seguros que motoristas humanos, eles ainda carecem do senso ético humano, como reduzir a velocidade para evitar molhar pedestres ao passar por uma poça de água ou desviar o veículo para evitar atropelamentos, mesmo que isso possa causar danos ao veículo e a seus passageiros.

A ausência de uma exigência legal para a incorporação de ética no desenvolvimento de sistemas inteligentes faz com que a implementação de restrições éticas dependa da iniciativa dos desenvolvedores. A tarefa de implementar ética em IA é repleta de obstáculos, começando pela dificuldade de encontrar uma forma de traduzir conceitos éticos para uma máquina, apesar de ser possível abordar ética de um ponto de vista lógico, o que facilita a tradução de ética para máquinas, problemas do mundo real não são simples de se descrever apenas verificando a veracidade de uma sentença. Outra preocupação durante o desenvolvimento é que os agentes devem ser capazes de lidarem com possíveis ambiguidades e conflitos entre as regras éticas.

Além das dificuldades técnicas, a implementação de ética em sistemas inteligentes enfrenta obstáculos significativos na aceitação pelo público e na competitividade no mercado. A percepção pública da ética dos sistemas de IA pode variar amplamente, e um sistema que seja considerado eticamente aceitável em uma cultura pode não ser visto da mesma forma em outra. Isso devido ao fato de que ética é um conceito dinâmico, influenciado por fatores culturais, sociais e econômicos (MAXMEN, 2018), o que complica a criação de normas universais aplicáveis a sistemas autônomos.

A introdução da ética na tomada de decisão de sistemas inteligentes é uma área que demanda atenção e colaboração entre desenvolvedores, órgãos regulamentadores e a sociedade em geral. O objetivo é garantir o desenvolvimento de sistemas autônomos inteligentes que não apenas executem suas tarefas de maneira eficiente, mas também respeitem os valores e princípios éticos que são fundamentais para a convivência humana.

Este trabalho visa contribuir com os esforços de desenvolvedores no objetivo de introduzir ética na IA, isso, através da analise de uma proposta de abordagem para o desenvolvimento de um sistema autônomo inteligente guiado por princípios éticos.

1.1 Objetivos do Trabalho

Esta Seção apresenta o objetivo geral e os objetivos específicos deste trabalho de conclusão de curso.

1.1.1 Objetivo Geral

Este trabalho visa apresentar uma proposta de abordagem para inserção de ética na tomada de decisão de agentes autônomos utilizando a combinação de aprendizado por reforço e aprendizado profundo.

1.1.2 Objetivos Específicos

Para a realização do objetivo geral deste trabalho de conclusão de curso, faz-se necessário o cumprimento dos seguintes objetivos específicos:

1. Estudar as técnicas de aprendizado por reforço e aprendizado por reforço profundo;
2. Estabelecer princípios éticos para serem implementados como restrições na tomada de decisão por parte do agente inteligente;
3. Implementar o algoritmo proposto com as restrições éticas definidas;
4. Avaliar os resultados obtidos durante o treinamento do agente com restrições éticas;

1.2 Metodologia

Para a realização deste trabalho buscou-se na literatura da área trabalhos que implementem princípios éticos em agentes autônomos inteligentes. A partir deste estudo foi proposto o desenvolvimento de um agente de aprendizado por reforço treinado usando o algoritmo de aprendizado profundo *Deep Q-Learning* (DQL). Para a implementação do agente foi desenvolvido a representação de um ambiente de aprendizado por reforço integrado com o simulador de robótica *CoppeliaSim* que atua em conjunto com a implementação em *Python* do algoritmo de treinamento.

1.3 Organização do Trabalho

Este Trabalho de Conclusão de Curso, além desta introdução, está organizado da seguinte forma: na Seção 2 são apresentados os trabalhos correlatos encontrados na literatura; na Seção 3 são discutidos os fundamentos por traz da técnica e algoritmo utilizados neste trabalho; na Seção 4 é descrito o desenvolvimento da implementação proposta; na Seção 5 são apresentados os resultados de treinamento do agente desenvolvido e é feita uma analise sobre eles; por fim, na Seção 6 são apresentadas as conclusões derivadas da analise feita na seção anterior e são propostas alterações na abordagem implementada para o desenvolvimento de trabalhos futuros.

2 Estado da Arte

A implementação de ética em sistemas inteligentes pode ser abordada de várias maneiras. Esta seção apresenta trabalhos que exploram diferentes implementações e discute seus resultados, vantagens e desvantagens. Os métodos discutidos serão divididos em dois grupos: baseados em regras e baseados em aprendizado de máquina.

2.1 Implementações Baseadas em Regras

Os métodos baseados em regras consistem em inserir um conjunto de princípios éticos preestabelecidos na tomada de decisão. O agente inteligente não pode tomar decisões que entrem em conflito com essas normas éticas. Assim, os princípios éticos funcionam como regras que ditam o comportamento do agente. Essas regras atuam como uma forma de validação, determinando se a ação decidida pelo sistema pode ou não ser executada. Dentre os métodos baseados em regras serão discutidos o Governador Baseado em Regras, Satisfação de Restrições e Verificação Formal.

2.1.1 Método de Governador Baseado em Regras

A implementação desse método utiliza lógica formal, pois sistemas computacionais não conseguem interpretar, de forma clara e sem ambiguidades, sentenças escritas em linguagem natural (BRINGSJORD; ARKOUDAS; BELLO, 2006). A abordagem de introduzir normas éticas em sistemas inteligentes com lógica formal permite traduzir conceitos e princípios humanos para uma forma simbólica, simplificando o aspecto ético e removendo possíveis ambiguidades.

Embora a abordagem baseada em lógica formal facilite a compreensão de ética para as máquinas, ela apresenta dificuldades na implementação. O conjunto de normas éticas deve ser cuidadosamente escolhido pelo desenvolvedor, pois um conjunto de normas muito complexo torna-se difícil de ser representado em lógica formal, prejudicando a tomada de decisão. A exclusão de sentenças éticas que não podem ser traduzidas para lógica formal, ou que gerem contradições e ambiguidades, pode fazer com que o agente se torne incapaz de atuar de maneira autônoma em determinados dilemas éticos dentro do seu contexto de aplicação.

2.1.2 Método de Satisfação de Restrições

As técnicas de satisfação de restrições tentam verificar se as ações propostas por um sistema inteligente satisfazem um conjunto de restrições. As restrições são referentes aos estados do ambiente com o qual o sistema interage. Ao contrário das regras vistas na Seção 2.1.1, que especificam quais ações podem ou não serem executadas, as restrições especificam o que não pode acontecer no ambiente.

A vantagem de utilizar restrições é que existe uma certeza de que um estado que quebre um principio ético não será atingido, desde que a sua determinada restrição seja satisfeita. Esse método é utilizado em tarefas onde o custo de quebrar um principio ético é muito alto, o que leva a utilização dessa implementação em sistemas capazes de exercerem força letal (ARKIN, 2008). O uso de técnicas de satisfação de restrições também traz a vantagem de explicar quais restrições impediram o sistema de executar uma ação. Entretanto, dependendo do contexto da aplicação do sistema, derivar restrições para determinados dilemas éticos pode ser uma tarefa difícil, fazendo com que a restrição, caso mal formulada, seja impossível de ser validada, resultando em imprecisões na tomada de decisão.

2.1.3 Método de Verificação Formal

A verificação formal se baseia em utilizar técnicas de modelagem para descrever o comportamento do sistema inteligente, de forma que possa ser feita uma validação matemática se suas ações quebram algum princípio ético. Uma das técnicas utilizadas em

verificação formal é o *Model Checking*, que tem o objetivo de verificar se um sistema atende um conjunto de especificações. Nessa técnica, é criado um modelo matemático do sistema inteligente e do ambiente, assim é possível verificar, em qualquer ponto da interação entre o sistema e o ambiente, se a ação executada leva a um estado que quebra alguma das especificações éticas do ambiente (DENNIS et al., 2016).

Ter um modelo matemático que fornece o melhor curso de ação traz a vantagem de poder planejar as ações do agente para diferentes cenários, independente de decisões atômicas do agente, fornecendo ao sistema a possibilidade de seguir o curso de ação que leva ao estado onde nenhuma, ou o mínimo possível, de princípios éticos são violados. Porém, em ambientes reais, onde existem diversas variáveis que fogem do controle do desenvolvedor, a tarefa de modelar o ambiente se torna complexa, ou até impossível, fazendo com que o sistema não consiga fornecer uma ação que faça o modelo convergir para um estado desejado.

2.2 Implementações Baseadas em Aprendizado de Máquina

Diferente dos métodos baseados em regras vistos na Seção 2.1, onde é necessário que o desenvolvedor do sistema construa uma representação lógica ou matemática dos princípios éticos que governam o ambiente, na abordagem de aprendizado de máquina, o próprio agente constrói uma representação para as regras éticas presentes no ambiente. Para isso, é preciso quantificar o impacto das ações do agente em relações ao princípios éticos, o que na maioria dos casos, é uma tarefa mais simples do que modelar os conceitos éticos assim como são postulados. Para a implementação desse método, são utilizados algoritmos de aprendizado por reforço.

2.2.1 Método de Aprendizado por Reforço

No método de aprendizado por reforço, o sistema aprende a ajustar o seu comportamento através de recompensas e punições por suas ações no ambiente. Isso permite o sistema a aprender do zero o seu objetivo e os princípios éticos dentro do ambiente (ABEL; MACGLASHAN; LITTMAN, 2016). Esse comportamento é alcançado através dos algoritmos de aprendizado por reforço, onde o objetivo é maximizar a recompensa do sistema no ambiente, sendo atribuídas recompensas positivas para quando o sistema toma uma decisão que não quebre nenhum princípio ético no contexto do ambiente.

Implementações que utilizam aprendizado por reforço trazem a vantagem de serem capazes de representarem ambientes complexos e dinâmicos, principalmente em métodos que utilizam redes neurais. A capacidade de representar ambientes dinâmicos aumenta o número de problemas onde é possível introduzir ética e também permite representar conceitos éticos mais complexos.

O sistema derivar a sua própria representação dos conceitos éticos pode ser citado como outra vantagem, pois isenta a necessidade do desenvolvedor do sistema de construir as representações das normas éticas, removendo um ponto de erro humano e inserção de bias. Entretanto, é comum que, em aprendizado por reforço, o sistema encontre uma solução que não é ótima para o problema, isso pode resultar em comportamentos onde o sistema toma decisões que não respeitem os princípios éticos do ambiente afim de alcançar os seus objetivos.

3 Fundamentação Teórica

Esta seção apresenta os conceitos fundamentais para o entendimento do algoritmo utilizado na implementação. Primeiro, aborda-se o aprendizado de máquina, que utiliza algoritmos para identificar padrões em dados. Em seguida, discute-se o aprendizado por reforço, onde agentes aprendem através da interação com o ambiente, recebendo recompensas ou penalidades, e o aprendizado profundo, que usa redes neurais profundas para identificar padrões complexos em dados. Por fim, é abordado o conceito de aprendizado por reforço profundo, que combina aprendizado por reforço com redes neurais profundas para resolver problemas em ambientes com maior dinamicidade.

3.1 Aprendizado de Máquina

Aprendizado de máquina, é a área de inteligência artificial focada no desenvolvimento de sistemas capazes de aprender, a partir de dados, a executarem tarefas sem que sejam explicitamente programados para isso (ZHOU, 2021). De maneira genérica, aprendizado de máquina se resume em utilizar algoritmos capazes de aprenderem padrões em dados referentes a um problema, de modo que o sistema seja capaz de realizar predições e tomar decisões de maneira autônoma.

Para a implementação de aprendizado de máquina em um problema é preciso, primeiramente, coletar e processar dados que tragam informações relevantes sobre a tarefa a ser executada. Em seguida, é determinada uma abordagem de aprendizado adequada para a tarefa. Dentro de aprendizado de máquina existem diversas abordagens, cada uma com características que favorecem o aprendizado para uma categoria de tarefas. Por fim, é escolhido um algoritmo de aprendizado, que assim como a abordagem, também é decidido de acordo com a tarefa a ser executada (MITCHELL, 1997).

O treinamento do sistema, ocorre em iterações, onde o algoritmo de treinamento é exposto ao conjunto de dados de treinamento, ao qual é aplicado o algoritmo de aprendizado. Em seguida, o sistema é testado utilizando o conjunto de dados de teste, onde é avaliada sua performance na tarefa a qual foi aplicado. O treinamento é repetido até que o sistema alcance uma performance satisfatória. A performance está ligada capacidade do sistema de generalizar os conhecimentos adquiridos do conjunto de dados de treinamento e aplicá-los em dados novos.

Dentro do domínio de aprendizado de máquina existem diferentes maneiras de abordar cada problema, cada abordagem descreve uma técnica de treinamento e como modelar os dados para tal técnica. Dentre as principais abordagens estão o aprendizado supervisionado, aprendizado não supervisionado e aprendizado por reforço.

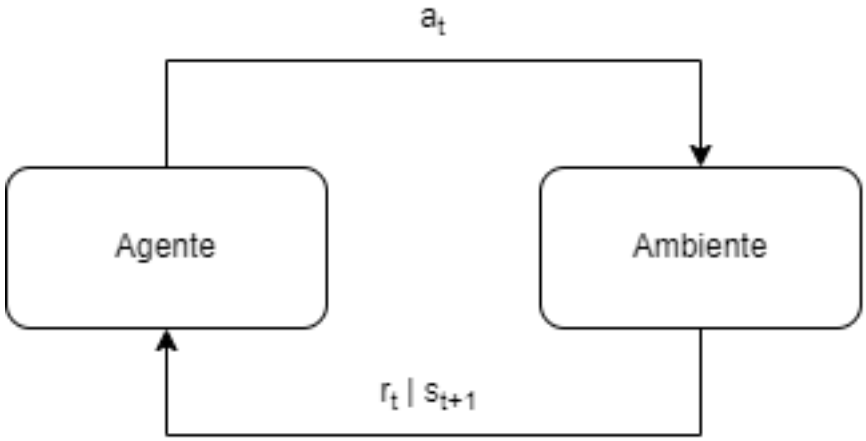
3.2 Aprendizado por Reforço

Dentre as diferentes abordagens de aprendizado de máquina o aprendizado por reforço é a técnica voltada para o desenvolvimento de sistemas, comumente chamados de agentes, capazes de tomarem decisões sequenciais com o objetivo de executar uma, ou um conjunto de tarefas. Diferente das demais abordagens, no aprendizado por reforço o agente aprende através de um processo de tentativa e erro, onde o agente interage diretamente com o ambiente no qual irá atuar, e molda o seu comportamento de acordo com recompensas, ou punições, fornecidas pelo ambiente.

No contexto de aprendizado por reforço, o ambiente representa um sistema externo

com o qual o agente interage em intervalos de tempo t discretos ($t = 0, 1, 2, ...$). A cada intervalo o agente executa uma ação a_t existente no espaço de ações A, baseada nas informações do estado s_t existente no espaço de estados S do ambiente, também chamado de espaço de observações, e recebe uma recompensa r_t, calculada através de uma função de recompensa, e informações sobre o estado subsequente s_{t+1} (SUTTON; BARTO, 2018), assim como exemplificado na Figura 4. O objetivo do agente é aprender um mapeamento entre os estados e as ações, conhecido dentro de aprendizado por reforço como politica, que maximize a recompensa cumulativa do ambiente.

Figura 1 – Interação entre agente e ambiente no aprendizado por reforço.



Fonte: Elaborado pelo autor (2024).

Durante o treinamento, inicialmente, o agente toma decisões de maneira aleatória, como uma forma de explorar o ambiente, afim de adquirir as primeiras informações de como suas ações influenciam o ambiente e a recompensa. Após um número arbitrário de iterações, o agente começa a utilizar o seu conhecimento adquirido do ambiente para tomar as decisões. Esse conceito de explorar o ambiente e em seguida utilizar seu conhecimento é chamado de *exploration vs. exploitation*, e é um dos conceitos fundamentais do aprendizado por reforço. O equilíbrio entre explorar e utilizar o conhecimento adquirido possibilita a descoberta de novas políticas para maximizar a recompensa cumulativa.

Outro conceito fundamental dentro de aprendizado por reforço é a função valor. A função valor é responsável por estimar a recompensa cumulativa do ambiente para cada par de estado e ação (s_t, r_t), de modo a guiar o agente a escolher ações que levam a estados com a maior recompensa possível, assim, recursivamente, maximizando a recompensa.

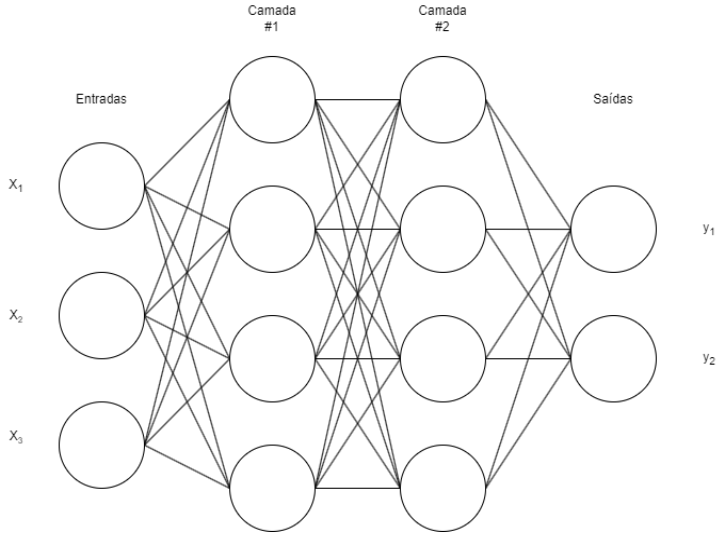
3.3 Aprendizado Profundo

O aprendizado profundo é uma das áreas em aprendizado de máquina onde os modelos utilizados são conhecidos como redes neurais artificiais (RNA). As RNA possuem uma grande capacidade de generalização, conseguindo extrair conceitos abstratos para formar uma representação hierárquica de dados, fazendo com que sejam amplamente utilizadas em aplicações que apresentam elevado grau de complexidade (GOODFELLOW; BENGIO; COURVILLE, 2016).

As RNA's são compostas por camadas de nodos, chamados de neurônios artificiais. As camadas da RNA são conectadas umas as outras através de seus nodos, onde cada

conexão possui um peso w_n atribuído a ela.

Figura 2 – Estrutura genérica de uma RNA profunda.



Fonte: Elaborado pelo autor (2024).

A Figura 2 ilustra um RNA profunda genérica, composta por duas camadas ocultas mais as camadas de entrada e saída. Neste exemplo, as camadas são ditas densamente conectadas, pois existem conexões entre todos os nodos das camadas adjacentes.

Cada neurônio artificial recebe como entrada o sinal gerado pelos nerônios da camada anterior, executa uma soma ponderada pelos pesos w_n de cada conexão e então aplica uma função de ativação no resultado, assim gerando sua saída. A função de ativação manipula o valor calculado no nodo de forma a indicar para os neurônios adjacente se o resultado é relevante para a tomada de decisão. A repetição desse processo ao longo de todas camadas da RNA resulta em um mapeamento dos dados da camada de entrada para a camada de saída.

Durante o treinamento da RNA os pesos de suas conexões são ajustados através de algoritmos que implementam retro-propagação do gradiente de erro (MITCHELL, 1997). O erro em uma RNA é dado como a diferença entre o resultado da camada de saída e o resultado esperado. O erro, junto da acurácia, são as principais métricas de performance no aprendizado profundo. O erro se refere a diferença entre o resultado da camada de saída da RNA e valor esperado, enquanto a acurácia se refere a porcentagem de vezes em que a rede neural inferiu o resultado corretamente.

Redes neurais são estruturas complexas que possuem diversos parâmetros em seu treinamento que influenciam na sua performance, como o número de camadas e a quantidade de neurônios em cada camada. De maneira geral, a escolha dos parâmetros é dada de acordo com o tipo de tarefa para qual a RNA é aplicada, a variação e ajustes de parâmetros deve ser feita de acordo com a avaliação das métricas de performance.

3.4 Aprendizado por Reforço Profundo

O aprendizado profundo por reforço combina os conceitos e métodos vistos nas Seções 3.2 e 3.3 para criar agentes capazes de atuarem em ambientes de maior complexidade. Nesta abordagem as redes neurais são utilizadas para aproximar a função valor e mapear a

politica que resulta na maior recompensa cumulativa do ambiente. A utilização de uma RNA no aprendizado por reforço permite representar estados e ações de maneira mais detalhada dentro de um ambiente, porém o treinamento da rede neural agrega grande custo computacional e de tempo no treinamento do agente.

A interação entre o agente e o ambiente ocorre da mesma maneira como descrita na Seção 3.2, onde a cada iteração com o ambiente os pesos da rede neural são ajustados. O objetivo do treinamento é fazer com que o resultado da rede neural aproxime a função valor, responsável por relacionar os estados e ações as recompensas. As entradas da rede neural são as observações do agente, ou os estados do ambiente, e as saídas são os valores quantificando o grau de contribuição de cada ação para o objetivo de completar a tarefa.

Uma das principais implementações de aprendizado por reforço profundo é o algoritmo *Deep Q-Learning* (DQL) (MNIH et al., 2013). O DQL utiliza redes neurais profundas para aproximar a Função valor-Q, nome dado a função valor utilizada no algoritmo de aprendizado por reforço *Q-Learning* (WATKINS; DAYAN, 1992).

$$Q(s_t, a_t) = r_t + \gamma[\max(Q(s_{t+1}, a_{t+1}))] \tag{1}$$

A Equação [1], derivada da *equação de Bellman*, é a função valor utilizada no algoritmo DQL. Nela $Q(s_t, a_t)$ representa o valor da ação a_t no estado s_t. A recompensa r_t é somada ao valor máximo esperado de $Q(s_{t+1}, a_{t+1})$, que representa o valor da ação a_{t+1} no estado s_{t+1}. A soma da recompensa somada com o valor máximo esperado do estado subsequente é ponderada pelo fator de desconto γ, que indica o quanto recompensas imediatas são valorizadas em relação a recompensa cumulativa.

Apesar de trazer as vantagens citadas anteriormente em relação a os agentes comuns de aprendizado por reforço, a implementação de redes neurais faz com que o processo de treinamento seja lento, exigindo, em alguns casos, milhões de iterações para o agente encontrar uma politica. Métodos de aceleração por *hardware* são utilizados para acelerar o treinamento, realizando os cálculos do treinamento da RNA em uma placa de processamento gráfico, por exemplo, porém isso não garante que a política encontrada pelo agente seja ótima para o ambiente.

4 Desenvolvimento

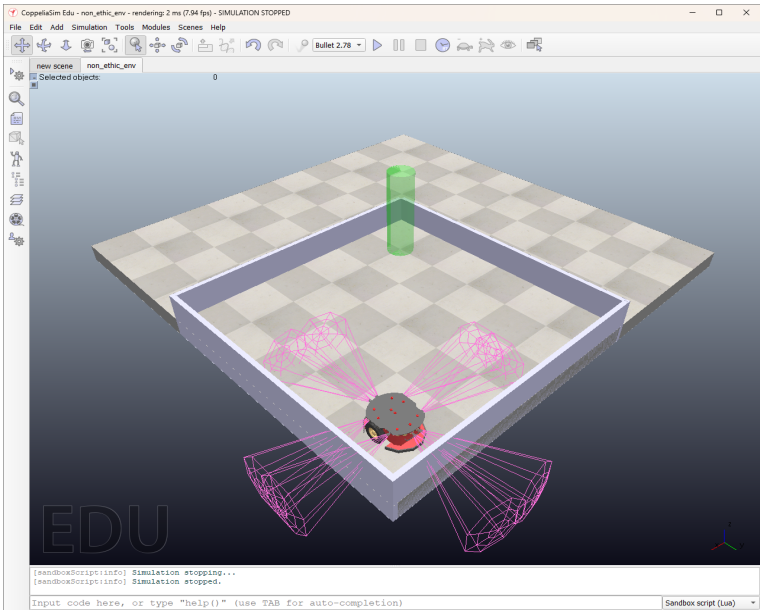
Esta seção descreve a implementação de um agente inteligente em um simulador de robótica, onde o agente representa um robô móvel que tem como objetivo navegar no ambiente a partir de um ponto de origem até um ponto de destino evitando colisões. Em seguida, é descrita a implementação do algoritmo de aprendizado, e suas variações em relação ao algoritmo tradicional. Por fim, são apresentadas as alterações necessárias para a introdução de ética no ambiente, e para que o agente considere ética na sua tomada de decisão

4.1 Implementação do Agente e do Ambiente

A implementação foi feita utilizando o simulador de robótica *CoppeliaSim* (ROHMER; SINGH; FREESE, 2013) em conjunto com a linguagem de programação *Python*. O ambiente é composto por um cenário onde um robô móvel deve navegar de seu ponto de origem até um ponto de destino evitando colisões. Neste cenário, o agente é um robô *Pioneer P3-DX* equipado com dois atuadores para locomoção e dezesseis sensores de

proximidade distribuídos ao redor de seu chassi. A interação com o ambiente é realizada através da biblioteca *ZMQ Remote API* (COPPELIAROBOTICS, 2020).

Figura 3 – Simulação do agente e do ambiente dentro do simulador *CoppeliaSim*.



Fonte: Elaborado pelo autor (2024).

A representação do ambiente foi construída seguindo os padrões estabelecidos pela implementação da biblioteca *Gymnasium* (TOWERS et al., 2023). A biblioteca *Gymnasium* é uma biblioteca de código aberto que fornece uma interface de programação padronizada para o teste e treinamento de agentes de aprendizado por reforço. As principais funcionalidades da biblioteca são: *reset*, responsável por voltar o ambiente a seu estado inicial a cada ciclo de treinamento, e *step*, responsável por executar a ação do agente no ambiente e transicionar entre os estados.

Os estados do ambiente, que são observados pelo agente, incluem as posições x e y do robô, sua velocidade, a posição do objetivo, a distância até o objetivo e as leituras dos sensores de proximidade. Essas observações são fundamentais para que o agente possa entender o ambiente em que está operando e tomar decisões adequadas.

Tabela 1 – Espaço de observações S.

Propriedade do ambiente	**Valores**
Posição x do agente	$[0.0, 2.0]$
Posição y do agente	$[0.0, 2.0]$
Velocidade do agente	$[-3.0, 4.0]$
Posição x do objetivo	$[0.0, 2.0]$
Posição y do objetivo	$[0.0, 2.0]$
Distância euclidiana até o objetivo	$[0.1, 2.82]$
Medições dos sensores de proximidade	$[0.0, 1.0]$

Fonte: Elaborado pelo autor (2024).

A Tabela 1 representa o espaço de observações S e detalha as propriedades que o agente considera ao tomar decisões. Cada propriedade é listada junto com o intervalo

de valores que pode assumir. O número de propriedades define a dimensão da camada de entrada da RNA utilizada no treinamento do agente.

O processo de implementação começa com a definição do ambiente de simulação no *CoppeliaSim*. O ambiente é configurado com um cenário cercado por paredes e pontos de origem e destino pré-definidos. A simulação é dividida em 100 intervalos que totalizam 30s de simulação. A cada intervalo, o agente toma uma ação baseada nas observações do estado atual do ambiente. As ações consistem em controlar os motores para que o robô se movimente em direção ao objetivo, ao mesmo tempo em que evita colisões com as paredes.

Tabela 2 – Espaço de ações A.

Ação	**Valores**	**Valor discreto**
Direita	$(3.0, -3.0)$	1
Esquerda	$(-3.0, 3.0)$	2
Frente	$(4.0, 4.0)$	3

Fonte: Elaborado pelo autor (2024).

A Tabela 2 representa o espaço de ações A e seus valores, cada um correspondendo a um par específico de velocidades para os atuadores do robô *(atuador esquerdo, atuador direito)*. A quantidade de ações influencia diretamente na dimensão da camada de saída da RNA utilizada no treinamento do agente. Cada neurônio na camada de saída corresponde a uma das possíveis ações, permitindo que a RNA determine a melhor ação a ser executada com base nas observações do estado atual do ambiente. As ações são discretas, assim, por exemplo, caso o segundo neurônio da camada de saída apresente o maior valor, a segunda ação será executa, sendo essa mover o agente para a esquerda.

A biblioteca *ZMQ Remote API* (COPPELIAROBOTICS, 2020) é utilizada para a comunicação entre o código *Python*, responsável pelo treinamento, e o simulador. A biblioteca permite a manipulação dos objetos e entidades presentes na simulação, leitura dos sensores e execução das ações do robô. O ciclo de aprendizado por reforço envolve a inicialização do ambiente, a execução das ações pelo agente, a atualização das observações e a execução do treinamento do agente.

O fragmento de código apresentado na Figura 4 é parte da implementação da interação entre o agente e o ambiente durante o treinamento. Este código exemplifica o ciclo típico de treinamento em um ambiente de aprendizado por reforço, onde o agente interage com o ambiente, coleta experiência, e realiza o treinamento da rede neural.

Tabela 3 – Função de recompensa.

Conceito avaliado	**Recompensa**
Colisão	-1
Objetivo	$+1$
Tempo	-1

Fonte: Elaborado pelo autor (2024).

A Tabela 3 apresenta as recompensas r_t e punições atribuídas as ações do agente. Ao atingir a região objetivo o agente é recompensado, caso o agente colida com alguma das paredes que limitam o cenário, recebe uma punição. O conceito de *Tempo* representa ações que não levam o agente ao objetivo e não resultam em colisões, sua recompensa é negativa para encorajar o agente a resolver a tarefa no menor número de interações possível, buscando maneiras de aprimorar a politica encontrada para o ambiente. Ao

Figura 4 – Fragmento de código que implementa a interação entre o agente e o ambiente.

```
for ep in range(1, episodes):
    done = False
    truncated = False
    state, _ = env.reset(randomize=True)

    ...

    while not (done or truncated):
        action = agent.choose_action(state)
        new_state, reward, done, truncated, _ = env.step(action)

        ...

        agent.memory.store_transition(state, action, reward, new_state,
            done)
        training_metrics = agent.train()

        ...

        agent.update_target_model(episode=ep)

        state = new_state

    ...

env.close()
```

Fonte: Elaborado pelo autor (2024).

longo de diversas iterações, as recompensas vão moldar o comportamento do agente. É importante ressaltar que a função de recompensa não fornece instruções de como resolver a tarefa, mas sim atribui valores numéricos as consequências das ações do agente.

4.2 Implementação do Algoritmo de Aprendizado

O algoritmo de aprendizado utilizado na implementação é o *Deep Q-Learning*, visto na Seção 3.4. A rede neural utilizada no treinamento foi construída utilizando as bibliotecas de desenvolvimento de algoritmos de aprendizado de máquina *TensorFlow* (ABADI et al., 2015) e *Keras* (CHOLLET et al., 2015), que combinadas, proporcionam o desenvolvimento rápido de redes neurais confiáveis.

A rede neural implementada é composta pela camada de entrada, com dimensão igual ao espaço de observações S, duas camadas ocultas densamente conectadas, cada uma com 512 neurônios e função de ativação *ReLU*. Finalmente, a camada de saída é densamente conectada e sua dimensão é determinada pelo espaço de ações A. A camada de saída não possui uma função de ativação, o que é uma prática comum para RNA utilizadas em aprendizado por reforço, pois as saídas correspondem diretamente às ações possíveis que o agente pode tomar no ambiente.

Diferente das implementações tradicionais de *Deep Q-Learning*, a implementação utilizada nesse trabalho apresenta duas modificações. A primeira modificação é o *Double DQN* (DDQN), que é uma extensão do *Deep Q-Learning* que utiliza duas redes neurais arquiteturalmente idênticas. No algoritmo *Double DQN*, uma RNA é usada para selecionar a melhor ação no próximo estado, enquanto outra rede é usada para avaliar o valor dessa

ação. Isso ajuda a reduzir o viés de superestimação (HASSELT; GUEZ; SILVER, 2015), problema comum que surge em implementações de *Deep Q-Learning*, resultando em um aprendizado mais estável e preciso.

A segunda modificação é a utilização de memória de repetição de experiência. Essa memória é uma estrutura que armazena estados de transição do ambiente, ou seja, a memória é composta por estados s_t, ações a_t, recompensas r_t e os estados subsequentes das ações s_{t+1}. Durante o treinamento, amostras aleatórias são retiradas dessa memória para treinar a rede neural. Essa abordagem traz vantagens importantes, como a diminuição da correlação entre os dados e uma maior distribuição de estados nos dados de treinamento (LIN, 1993).

O fragmento de código na Figura 5 implementa o treinamento do agente no algoritmo DDQL. Primeiramente, é retirado da memória de experiência um lote de amostras aleatórias das transições de estados do ambiente. Cada amostra é composta por um estado, uma ação, a recompensa por executar a ação, o estado resultante da ação e o sinal de terminação do ambiente $(s_t, a_t, r_t, s_{t+1}, done)$. Em seguida, é utilizada a RNA primaria para prever a recompensa cumulativa dos estados atuais $Q(s_t, a_t)$, e a RNA secundaria para prever a recompensa cumulativa dos estados resultantes $Q(s_{t+1}, a_{t+1})$. Por fim, é utilizada a função valor, apresentada na equação (1), para atualizar as recompensas cumulativas estimadas pela RNA primaria, e então a rede neural é treinada para reduzir a diferença entre os valores atualizados pela função valor e os valores estimados pela RNA secundaria.

Figura 5 – Fragmento de código que implementa o algoritmo DQL.

```
    def train(self):
        if len(self.memory) < self.batch_size:
            return

        batch = self.memory.sample(self.batch_size)
        states, actions, rewards, next_states, dones = zip(*batch)

        states = np.array(states)
        next_states = np.array(next_states)
        actions = np.array(actions)
        rewards = np.array(rewards)
        dones = np.array(dones)

        state_q = self.model.predict(states)
        new_state_q = self.target_model.model.predict(next_states)

        target_q = np.copy(state_q)

        batch_indexes = np.arange(self.batch_size, dtype=np.int32)

        target_q[batch_indexes, actions] = rewards + (self.gamma * np.max(
            new_state_q, axis=1) * dones)

        self.model.fit(states, target_q, epochs=1, verbose=0)
```

Fonte: Elaborado pelo autor (2024).

4.3 Introdução de Princípios Éticos no Agente

A introdução de princípios éticos na tomada de decisão do agente começa com a adaptação do ambiente. A tarefa se mantém a mesma descrita na Seção 4.1, assim como o

objetivo do agente, exceto que, com a inserção de princípios éticos, surgem dilemas éticos dentro do ambiente, e para acomodar tais dilemas, são adicionadas entidades no ambiente que representam conceitos éticos que o agente deve levar em consideração enquanto executa ações afim de completar seu objetivo.

O conjunto de princípios, ou regras, éticas utilizados na implementação proposta são as *Três Leis da Robótica*. Para representar as leis da robótica são criadas regiões no espaço que simbolizam os principais conceitos de cada uma das leis. As regiões representam os conceitos de evitar que mal ocorra a um ser humano (*avoid harm*), causar mal a um ser humano de maneira intencional (*inflict harm*) e causar mal a si próprio (*self harm*). Tais regiões, assim como a região objetivo do ambiente, compõem as observações do agente e passam a fazer parte do espaço de observações S, representado na Tabela 4. As *Três Leis da Robótica* são as seguintes:

1. Um robô não pode ferir um ser humano ou, por inação, permitir que um ser humano sofra algum mal.
2. Um robô deve obedecer às ordens que lhe sejam dadas por seres humanos, exceto nos casos em que entrem em conflito com a Primeira Lei.
3. Um robô deve proteger sua própria existência, desde que tal proteção não entre em conflito com a primeira ou segunda Lei.

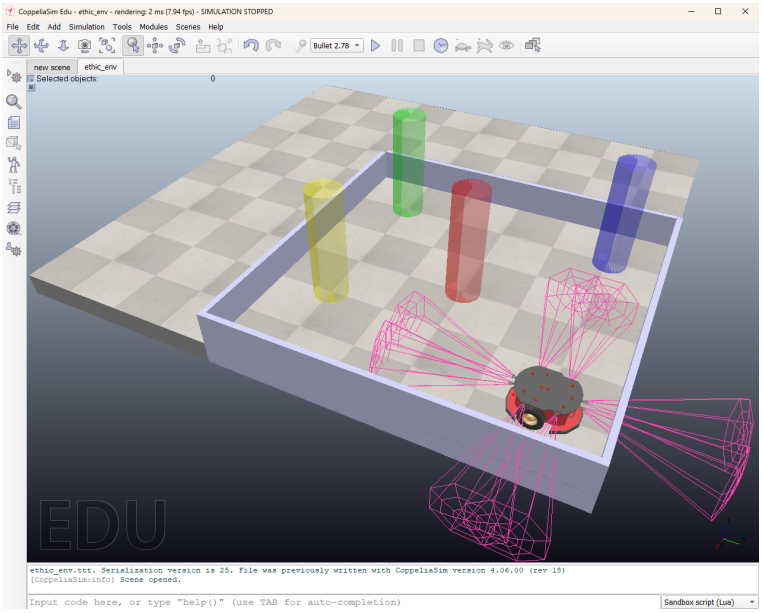
Tabela 4 – Espaço de observações considerando princípios éticos S'.

Propriedade do ambiente	**Valores**
Posição x do agente	[0.0, 2.0]
Posição y do agente	[0.0, 2.0]
Velocidade do agente	[-3.0, 4.0]
Posição x do objetivo	[0.0, 2.0]
Posição y do objetivo	[0.0, 2.0]
Distância euclidiana até o objetivo	[0.1, 2.82]
Posição x da região *avoid harm*	[0.0, 2.0]
Posição y da região *avoid harm*	[0.0, 2.0]
Distância euclidiana até a região *avoid harm*	[0.1, 2.82]
Posição x da região *inflict harm*	[0.0, 2.0]
Posição y da região *inflict harm*	[0.0, 2.0]
Distância euclidiana até a região *inflict harm*	[0.1, 2.82]
Posição x da região *self harm*	[0.0, 2.0]
Posição y da região *self harm*	[0.0, 2.0]
Distância euclidiana até a região *self harm*	[0.1, 2.82]
Medições dos sensores de proximidade	[0.0, 1.0]

Fonte: Elaborado pelo autor (2024).

A Figura 6 apresenta a simulação do agente e do ambiente dentro do simulador *CoppeliaSim* com a adição das regiões referentes a os princípios éticos. As regiões em amaralo, vermelho e azul representam, respectivamente, os coneitos de *avoid harm*, *inflict harm* e *self harm*. As novas regiões possuem as mesmas propriedades que a região objetivo, ou seja, o agente conhece suas posições e a distância até as mesmas, assim como representado na Tabela 4.

Figura 6 – Simulação do agente e do ambiente com as novas regiões representando os princípios éticos.



Fonte: Elaborado pelo autor (2024).

Do ponto de vista do agente, as novas regiões introduzidas não possuem significado até que sejam atribuídas recompensas as interações do agente com tais regiões. Para isso, surge a necessidade de modificar a função de recompensa do ambiente. Como a tarefa do agente se mantém a mesma, as alterações na função de recompensa se resumem em adições na Tabela 5.

Tabela 5 – Função de recompensa considerando princípios éticos.

Conceito avaliado	**Recompensa**
Colisão	-1
Objetivo	$+1$
Avoid harm	$+1$
Inflict harm	-1
Self harm	-1
Tempo	-1

Fonte: Elaborado pelo autor (2024).

A inclusão de recompensas para os dilemas éticos introduzidos no ambiente é apresentada na Tabela 5. Para a região *avoid harm*, caso o agente atinja essa região, é atribuída uma recompensa positiva, simbolizando que o agente tomou uma ação que evitou uma violação da primeira lei. Em contrapartida, caso o agente complete o objetivo sem atingir esta região, a recompensa é negativa, indicando que o agente deixou mal ocorrer a um ser humano por inatividade, simbolizando uma violação da primeira lei. Ainda referente a primeira lei, é atribuída uma recompensa negativa ao agente caso o mesmo atinja a região *inflict harm*. Quanto a terceira lei, caso o agente atinja a região *self harm*, é atribuída uma recompensa negativa por violar a terceira lei. A segunda lei, apesar de não gerar uma recompensa de maneira direta, gera uma restrição no comportamento do agente. Caso o caminho do agente até o objetivo seja interceptado pela região que simboliza causar

mal intencional a um ser humano, o agente deve encontrar um caminho alternativo para completar seu objetivo.

5 Avaliação do Algoritmo Proposto

Para avaliar o desempenho do algoritmo DDQN no objetivo de introduzir ética em um agente autônomo foram desenvolvidos dois agentes. Ambos agentes foram treinados com o mesmo algoritmo, mesmos parâmetros e para a mesma tarefa, exceto que, o primeiro agente, não considera princípios éticos na tomada de decisão.

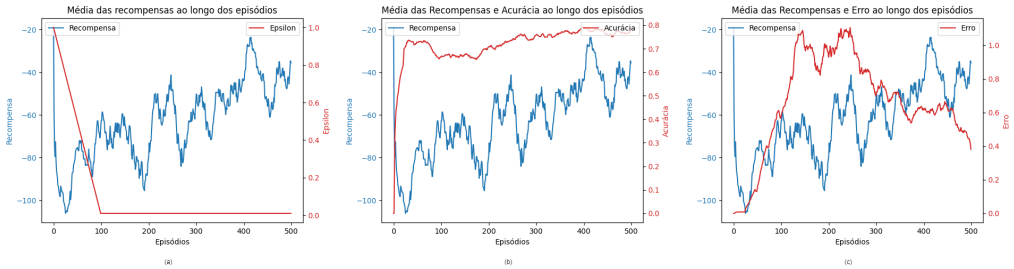
O ponto a ser avaliado é o impacto de introduzir ética na tomada de decisão do agente. Como visto na Seção 4.3, a introdução de ética resulta em alterações no espaço de observações S do agente, consequentemente na dimensão da camada de entrada da RNA, e na função de recompensa do ambiente.

O espaço S', apresentado na Tabela 4 possui maior dimensionalidade em relação S, apresentado na Tabela 1. A introdução de ética requer propriedades a mais dos que as presentes em S para descrever os estados do ambiente. De maneira geral, a introdução de propriedades no espaço de observações agrega complexidade no processo de treinamento do agente, fazendo com que sejam necessárias alterações nos parâmetros para comportar as novas observações.

Aumentar as dimensões da camada de entrada da RNA também traz preocupações para a performance do agente. Alterar a RNA resulta em uma alteração na arquitetura do modelo de aprendizado, que é o principal fator que provoca a necessidade de alterar os parâmetros de treinamento. Assim como descrito na Seção 3.3, os parâmetros influenciam diretamente na performance do agente e na qualidade da politica derivada.

O comportamento do agente é moldado pela função de recompensa, logo é esperado que alterações nas recompensas resultem em alterações no comportamento do agente. Apesar de que alterar o comportamento do agente seja o objetivo de introduzir ética na tomada de decisão, se as alterações não forem feitas de maneira adequada, o agente pode acabar tendo desvios não desejados em seu comportamento. A Tabela 3 representa um comportamento relativamente fácil de ser inferido pelo agende, pois existe uma única recompensa positiva, entretanto, na Tabela 5 existem duas recompensas positivas. A existência de mais de uma recompensa positiva pode resultar em comportamentos ambíguos no agente, onde a politica pode convergir para apenas um dos conceitos avaliados pela função de recompensa.

Figura 7 – Dados de treinamento do agente não ético.



Fonte: Elaborado pelo autor (2024).

Analisando a Figura [7], que apresenta a média das recompensas (a), da acurácia

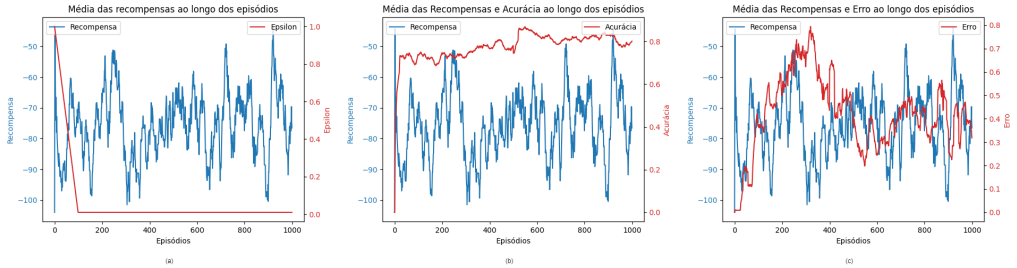
(b) e do erro (c), ao decorrer dos episódios de treinamento do agente não ético, é visto que a curva referente a média das recompensas (curva em azul), apresenta uma tendência de crescimento conforme se passam os episódios. Essa tendência de crescimento indica que o agente está aprendendo com a experiência de interagir com o ambiente.

A curva em vermelho na Figura (a) representa o valor de Épsilon, variável que controla o equilíbrio entre *exploration* e *exploitation*, citado na Seção 3.2. Conforme o valor Épsilon diminui, o agente reduz em ϵ porcento a quantidade de ações aleatórias e passa a inferir a melhor ação para o estado em que se encontra. Épsilon é mantido em 1% ao longo do treinamento para inserir pequenas aleatoriedades na tomada de decisão, fazendo com que o agente transacione para possíveis novos estados e descubra novas politicas.

Ainda na Figura [7], as Figuras (b) e (c), apresentam, respectivamente, a média da acurácia e do erro ao decorrer dos episódios de treinamento. A acurácia e o erro, métricas mencionadas na Seção 3.3, indicam o quão bem a RNA aprendeu a aproximar a função valor. Na Figura (b), vemos que a média da acurácia da RNA apresenta tendência de crescimento junto com a média das recompensas, porém, isso não garante que a politica derivada pelo agente é ótima para o problema, apenas indica que a aproximação da função valor está convergindo. O mesmo é valido para o erro, Figura (c), que apresenta tendência de atenuação, indicando que, em torno do episódio de número 500, na média dos 20% $(1 - Acurácia)$ de casos em que a RNA inferiu de maneira incorreta a função valor, a diferença do valor inferido para o valor calculado através da função valor, equação (1) foi de, na média, 40%.

Aplicando a mesma análise para a Figura 8, que apresenta os dados de treinamento do agente ético, na Figura (a), é visto que a curva da média das recompensas não apresenta tendência de crescimento ao decorrer dos episódios. A ausência de indícios de crescimento na médias das recompensas, junto com as variações abruptas na média dentro de um intervalo pequeno de episódios, indicam que o agente não foi capaz de inferir uma politica para maximizar a recompensa.

Figura 8 – Dados de treinamento do agente ético.



Fonte: Elaborado pelo autor (2024).

Variações na média das recompensas, como as vistas nos dados de treinamento de ambos agentes, são comuns em algoritmos de aprendizado por reforço, isso devido a natureza da abordagem, que como visto na Seção 3.2 se assemelha a um processo de tentativa e erro. Entretanto, é desejado que a curva das recompensas seja mais suave, isso quando $\epsilon \to 0$, e atingindo seu valor máximo quando o número de episódios n é suficientemente grande. O agente ético foi treinado por mais iterações que a sua contraparte, passando por 1000 episódios de treinamento, ao longo dos quais não apresentou sinais de convergência para uma politica que resolva os dilemas éticos inseridos no ambiente.

Analisando a Figura 8 (b), a curva da média da acurácia, curva em vermelho, apresenta comportamento similar a da Figura 7 (b), o que é esperado dado que a acurácia é referente a capacidade do modelo de aproximar a função valor utilizada no DDQL, função que não sofreu alterações para inserção de ética na tomada de decisão. Porém, a curva referente ao erro, Figura 8 (c), apresenta grandes variações para intervalos curtos de episódios, e não demonstra uma tendência de atenuação, se mantendo em torno de 40%. Isso é um indicativo de que, quando a RNA infere um valor incorreto, o erro é grande o suficiente para fazer com que a aproximação da função valor não convirja para um valor ótimo para o ambiente.

Na média, o agente foi capaz de atingir recompensas cumulativas entre -80 e -60, abaixo da sua contraparte não ética, mas ainda acima de -100. Recompensas acima de -100, para o ambiente implementado, indicam que o agente foi no mínimo capaz de aprender a tarefa de navegar de seu ponto de origem até o ponto objetivo evitando colisões.

Observando o comportamento do agente ao longo de testes nota-se que o agente obtém recompensas próximas de -100 em momentos em que a politica derivada indicou que o melhor curso de ação era andar em círculos, evitando interações com as regiões que simbolizam os dilemas éticos. Tal comportamento, mesmo ferindo o princípio ético atribuído a região *avoid harm*, descrita na Seção 4.3, aponta que o agente conseguiu derivar uma politica que considera os princípios éticos na sua tomada de decisão.

6 Considerações Finais e Propostas para Trabalhos Futuros

A partir das analises feitas na Seção 4 sobre os dados de treinamento do agente ético implementado utilizando DDQL, conclui-se que a inclusão de princípios éticos na tomada de decisão não só afeta a construção do ambiente de aprendizado por reforço, mas também a estabilidade do treinamento e a qualidade da politica derivada para a tarefa proposta no ambiente.

Mesmo que os resultados não parecerem encorajar o uso de aprendizado por reforço profundo na implementação de ética na tomada de decisão, é necessário ressaltar que os resultados apresentados neste trabalho são referentes a uma de diferentes formas de implementar ética em um ambiente de aprendizado por reforço, além de que os parâmetros, tanto de treinamento quanto os da RNA, podem ser modificados para atingirem melhores resultados.

6.1 Trabalhos Futuros

A abordagem construída nesse trabalho apresenta diversos pontos onde pode ser alterada para o desenvolvimento de trabalhos no mesmo escopo. Dentro de aprendizado por reforço, é possível experimentar com diferentes funções de recompensa e técnicas de exploração do ambiente. No contexto de aprendizado profundo é possível utilizar diferentes arquiteturas de redes neurais para processar os dados do ambiente. Quanto ao algoritmo de treinamento DDQL, podem ser exploradas diferentes implementações do algoritmo bem como outros algoritmos de aprendizado por reforço profundo, como o *Proximal Policy Optimization* (SCHULMAN et al., 2017), que apresenta politicas mais robustas para ambientes complexos com um tempo de convergência menor em relação a outros algortimos de aprendizado por reforço.

Referências

ABADI, M. et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. Software available from tensorflow.org. Disponível em: <https://www.tensorflow.org/>. Citado na página [17].

ABEL, D.; MACGLASHAN, J.; LITTMAN, M. L. Reinforcement learning as a framework for ethical decision making. In: *AAAI Workshop: AI, Ethics, and Society*. [s.n.], 2016. Disponível em: <https://api.semanticscholar.org/CorpusID:14717578>. Citado na página [10].

ARKIN, R. C. Governing lethal behavior: embedding ethics in a hybrid deliberative/reactive robot architecture. In: *Proceedings of the 3rd ACM/IEEE International Conference on Human Robot Interaction*. New York, NY, USA: Association for Computing Machinery, 2008. (HRI '08), p. 121–128. ISBN 9781605580173. Disponível em: <https://doi.org/10.1145/1349822.1349839>. Citado na página [9].

ASIMOV, I. *Runaround*. [S.l.]: Street Smith, 1942. Citado na página [7].

BRINGSJORD, S.; ARKOUDAS, K.; BELLO, P. Toward a general logicist methodology for engineering ethically correct robots. *IEEE Intelligent Systems*, v. 21, p. 38–44, 2006. Disponível em: <https://api.semanticscholar.org/CorpusID:6757843>. Citado na página [9].

CHOLLET, F. et al. *Keras*. 2015. <https://keras.io>. Citado na página [17].

COPPELIAROBOTICS. *zmqRemoteApi*. [S.l.]: GitHub, 2020. <https://github.com/CoppeliaRobotics/zmqRemoteApi.git>. Citado (2) vezes nas páginas [15 e 16].

DENNIS, L. et al. Formal verification of ethical choices in autonomous systems. *Robotics and Autonomous Systems*, v. 77, p. 1–14, 2016. ISSN 0921-8890. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0921889015003000>. Citado na página [10].

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>. Citado na página [12].

HASSELT, H. van; GUEZ, A.; SILVER, D. Deep reinforcement learning with double q-learning. 2015. Cite arxiv:1509.06461Comment: AAAI 2016. Disponível em: <http://arxiv.org/abs/1509.06461>. Citado na página [18].

LIN, L. J. *Reinforcement Learning for Robots Using Neural Networks*. Tese (Doutorado) — Carnegie Mellon University, Pittsburgh, January 1993. Citado na página [18].

MAXMEN, A. Self-driving car dilemmas reveal that moral choices are not universal. *nature*, 2018. Citado na página [7].

MITCHELL, T. *Machine Learning*. McGraw-Hill, 1997. (McGraw-Hill International Editions). ISBN 9780071154673. Disponível em: <https://books.google.com.br/books?id=EoYBngEACAAJ>. Citado (2) vezes nas páginas [11 e 13].

MNIH, V. et al. Playing atari with deep reinforcement learning. 12 2013. Citado na página [14].

ROHMER, E.; SINGH, S. P. N.; FREESE, M. Coppeliasim (formerly v-rep): a versatile and scalable robot simulation framework. In: *Proc. of The International Conference on Intelligent Robots and Systems (IROS)*. [S.l.: s.n.], 2013. Citado na página [14].

SCHULMAN, J. et al. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. Citado na página [23].

SUTTON, R. S.; BARTO, A. G. *Reinforcement Learning: An Introduction*. Second. The MIT Press, 2018. Disponível em: <http://incompleteideas.net/book/the-book-2nd.html>. Citado na página [12].

TOWERS, M. et al. *Gymnasium*. Zenodo, 2023. Disponível em: <https://zenodo.org/record/8127025>. Citado na página [15].

WATKINS, C. J. C. H.; DAYAN, P. Q-learning. *Machine Learning*, v. 8, n. 3, p. 279–292, may 1992. ISSN 1573-0565. Disponível em: <https://doi.org/10.1007/BF00992698>. Citado na página [14].

ZHOU, Z. *Machine Learning*. Springer Singapore, Imprint: Springer, 2021. ISBN 9789811519680. Disponível em: <https://books.google.com.br/books?id=Rb17zwEACAAJ>. Citado na página [11].