Revista Brasileira de Computação Aplicada, Julho, 2022

DOI: 10.5335/rbca.v14i2.12500
Vol. 14, Nº 2, pp. 26–34
Homepage: seer.upf.br/index.php/rbca/index

ISSN 2176-6649

ARTIGO ORIGINAL

Aprendizado por Reforço e Jogos: uma proposta focada na análise de algoritmos

Reinforcement learning and computer games: an approach focused on algorithms analysis

Diego B. da Costa [1], Giancarlo Lucca [2], *Diana F. Adamatti [1]

[1]Centro de Ciências Computacionais - Universidade Federal do Rio Grande (C3/FURG), [2]Programa de Pós-Graduação em Modelagem Computacional - Universidade Federal do Rio Grande (PPGMC/FURG)

{dbdcosta96, giancarlo.lucca88, *dianaada}@gmail.com

Recebido: 19/04/2021. **Revisado:** 05/04/2022. **Aceito:** 10/07/2022.

Resumo

O mercado de jogos movimenta bilhões de dólares por ano e está crescendo exponencialmente. O aprendizado por reforço é uma técnica de tentativa e erro que está diretamente relacionada a esse mercado. Assim, o estudo dessas técnicas em jogos populares torna-se relevante, como o estudo de caso deste projeto, o jogo *Pac-man*. Este trabalho tem como objetivo utilizar métricas para validar os resultados obtidos na simulação de algoritmos de aprendizado por reforço e sua validação baseada em algumas métricas, como recompensadas ganhas pelo agente, a exploração do ambiente, sua completude e o tempo de cada simulação. Vários testes foram realizados com cada algoritmo testado e os resultados demonstram que para ambientes com comportamentos com imprevisibilidade, o aprendizado por reforço tende a demorar muito a convergir.

Palavras-Chave: Aprendizado de Máquina; Aprendizado por Reforço; Jogos Computacionais.

Resumo

The gaming market moves billions of dollars a year and is growing exponentially. Reinforcement learning is a trial and error technique that is directly correlated with this market. Thus, the study of these techniques in popular games becomes relevant, as the case study of this project, the *Pac-man*. This work aims to use metrics to validate the results obtained from the simulation of reinforcement learning algorithms. The rewards earned by the agent, the exploration, its completeness and the time for each simulation will be validated. Several tests were performed with each algorithm and the results show that for environments with unpredictable behaviors, reinforcement learning tends to take a long time to converge.

Keywords: Computer Games; Machine Learning; Reinforcement Learning.

1 Introdução

Pode se dizer que a natureza de um jogo é muito mais antiga que a própria cultura (Huizinga, 1971). Se considerado em definições menos rigorosas, os animais brincavam antes mesmo que o homem iniciasse sua vida lúdica. Sendo assim, ainda segundo Huizinga (1971), o jogo é mais do que simplesmente um evento fisiológico ou reflexo psicológico, ultrapassando os limites de atividades puramente biológicas ou físicas. O jogo pode ser considerado uma função

significante, ou seja, que encerra um determinado sentido. Existe um fator implícito nos jogos, que transcendem as necessidades imediatas da vida e que conferem sentido à ação naquele contexto. Dessa forma, o autor assume que, o fato de ter uma função significante, implica na presença de um elemento não material na própria essência.

Já Crawford (2011) evidencia os quatro elementos fundamentais de todos os jogos:

- **Representação**: um jogo deve ser um sistema fechado e formado que possa representar um subconjunto da realidade. Deve ser um sistema, auto-suficiente, onde as partes interagem entre si, com regras representando um subconjunto da realidade;
- **Interação**: é responsável por tornar um jogo interativo, diferente de, por exemplo, de histórias, que possuem um desfecho fixo. Por meio da interação, é possível alterar a suposta representação da realidade. Ainda, por meio desse conceito, injeta-se um elemento social ou interpessoal no evento;
- **Conflito**: um elemento natural presente na grande parte dos jogos. O jogador busca ativamente atingir objetivos e sobrepujar por meio do percalço;
- **Segurança**: devido a presença de conflito, que visa criar uma situação de perigo, vê-se necessário estabelecer um mecanismo, por meio dos jogos, em que o jogador se submeta a essas experiências psicológicas de conflito ou perigo, sem ocasionar em danos físicos, permitindo assim a desassociar as consequências das ações, a segurança.

Para Juul (2011), existem seis principais requisitos que um jogo deve satisfazer para ser classificado dessa forma:

i. Ser um sistema formal baseado em regras bem definidas;
ii. A possibilidade de resultados quantificáveis e variados;
iii. Para cada possível resultado, seja possível associar valores diferentes;
iv. Prover a possibilidade de jogadores gastarem esforços para tentar influenciar no resultado;
v. Cria um elo motivacional entre os resultados e o jogador;
vi. Ter a possibilidade de que as consequências das suas atividades possam ser opcionais e negociáveis.

Os requisitos abordados por Juul (2011) e os elementos definidos por Crawford (2011) apresentam uma visão distinta, porém de certa forma, complementar. Em paralelo com a criação de jogos modernos, surgiu a Inteligência Artificial (AI - *Artificial Intelligence*) (Russell and Norvig, 2013). Em meados dos anos 50, o principal questionamento levantado por essa nova área era: "será que computadores podem pensar?". Russel and Norvig (1995) fornecem uma definição mais formal para o conceito, onde a inteligência artificial tem o foco de compreender entidades inteligentes, simulando o comportamento humano.

A *Game AI* (termo utilizado para distinguir a inteligência artificial utilizada em jogos e no meio acadêmico) surgiu somente no ano de 1974, com os jogos *Pursuit* (Pursuit, 2021) e *Qwak* (Qwark, 2021). A principal distinção entre os dois termos (*Game AI* e *AI*) diz respeito a sua finalidade: enquanto o segundo visa imitar ou simular ações complexas, o primeiro tem o intuito de transformar o ambiente em desafiador e divertido para o usuário (Schwab, 2004).

Aprendizado por reforço (ou *Reinforcement Learning* - RL) pode ser definido como o problema onde um agente inteligente deve aprender um comportamento através da abordagem de tentativa e erro em um ambiente dinâmico[1] (Kaelbling et al., 1996, p.237-p.238). Existem duas principais estratégias para resolver esses problemas: a primeira abordagem diz respeito a algoritmos genéticos (visa resolver problemas de otimização a partir da busca por um melhor caminho), ou com o uso de técnicas estatísticas e de programação dinâmica (Sutton and Barto, 2018).

Os temas propostos nesta pesquisa são pesquisados já a alguns anos. O trabalho de Mnih et al. (2013) é um dos trabalhos mais marcantes quando tratamos de aprendizado por reforço. Nele, os autores utilizam uma rede neural para aprender uma política de controle a partir de uma entrada de vídeo em ambientes complexos de reforço. Para isso, foi utilizado a implementação presente em Bellemare et al. (2012). Foram definidas formas de comparação entre os resultados: recompensa e valor das funções de reforço. Ainda, é possível observar figuras que tentam demonstrar a estabilidade dos resultados obtidos. Vale ressaltar que no trabalho proposto por Mnih et al. (2013) não são exploradas as causas dos resultados, ou tão pouco valores referentes a desvio padrão, valor central ou mínimo obtido a partir da simulação. Não existe uma forma para validar a exploração do cenário por parte do agente, como será apresentado neste trabalho.

Seguindo o mesmo modelo do anterior, o trabalho de van Hasselt et al. (2015) propõe uma abordagem capaz de resolver os problemas encontrados no algoritmo que utiliza redes neurais e Q-Learning. Conforme os autores, o *Double Q-Learning* (DQN) superestima os valores da equação, devido ao operador que sempre resulta no valor máximo, ocasionando assim em computações tendenciosas, ou seja, valores superiores ou inferiores aos reais obtidos pelo agente. Para solucionar o problema, é proposto um algoritmo que utiliza dois pesos diferentes para as redes, visando reduzir essa inconsistência.

Semelhante aos anteriores, a contribuição dada por Dabney et al. (2017), ocorre de forma similar aos antecedentes. O artigo visa reutilizar a implementação criada por Mnih et al. (2013) e adaptá-la para a regressão quantílica. Inicialmente, são conceitos os principais elementos utilizados na aprendizagem por reforço, seguido por um conceito de aprendizado distribuído.

Especificamente para o estudo de caso proposto, o jogo Pacman, alguns trabalhos realizaram análise de algoritmos de aprendizado de máquina, como Tziortziotis et al. (2014) e Bom et al. (2013).

Trabalhos mais atuais na área de jogos com aprendizado por reforço, como Brown et al. (2020), o qual apresenta um algoritmo que aplica o aprendizado por reforço para convergência em jogos de soma zero, buscando o equilíbrio de Nash. Os resultados do trabalho mostram que o algoritmo proposto tem valores similares ao AlphaZero, e tem melhor desempenho para ambientes onde a informação é

[1] Um ambiente que se altera de forma contínua.

imperfeita, ou seja, não completa.

Os trabalhos referenciados neste artigo seguem uma abordagem semelhante: demonstrar os algoritmos elaborados, junto a fundamentação teórica necessária para a compreensão do mesmo. Porém, as métricas para a validação são escassas e a apuração dessas mesmas é praticamente inexistente em alguns casos. Assim, este trabalho apresenta, na seção de metodologia, as métricas utilizadas especificamente para o estudo de caso em estudo.

O objetivo principal deste estudo é demonstrar os elementos mais relevantes presentes na aprendizagem por meio do reforço e seus algoritmos. É importante ressalta que além de testar e analisar alguns algoritmos aplicado a um estudo de caso em jogo computacional a partir de algumas métricas definidas.

O texto está organizado da seguinte maneira. Na Seção 2 os principais conceitos relacionados com o trabalho são apresentados. A Seção 3 consiste na metodologia empregada no estudo. Após, os resultados obtidos são apresentados na Seção 4. Por fim, as conclusões do estudo são feitas na Seção 5.

2 Conceitos preliminares

Nesta seção são apresentados os conceitos preliminares relacionados com o restante do texto. Inicialmente, na subseção 2.1 os conceitos de aprendizagem por reforço são apresentados. Após, na subseção 2.1, os algoritmos que são utilizados nesta pesquisa para a análise são descritos.

2.1 Aprendizado por reforço

Para Russell and Norvig (2013), a tarefa de aprendizado por reforço consiste em usar essas recompensas observadas para aprender uma política ótima (entende-se por política, uma solução para especificar ações para o agente em qualquer estado existente) para o ambiente. Os autores consideram ainda que essa abordagem por reforço é o único caminho possível em domínios complexos, para treinar um programa com desempenho em alto nível. É difícil para um humano fornecer avaliações precisas e suficientes para treinar uma função de avaliação diretamente a partir de exemplos. O intuito, com o reforço, é informar para o agente se ele finalizou ou não a etapa, a fim de que esse aprenda uma função de avaliação que forneça estimativas razoáveis a respeito de como vencer a partir de qualquer posição dada.

Já para Sutton and Barto (2018), o aprendizado por reforço pode ser definido como o que fazer para maximizar uma recompensa numérica de sinal. Não é explícito para o aprendiz qual ações tomar, mas sim descobrir quais ações levam a maior recompensa. Ainda, levando em consideração o ponto de vista dos autores, a busca através da tentativa e erro, aliado a recompensa atrasada, são as duas principais características que definem a abordagem.

Algoritmos de aprendizado por reforço visam, principalmente, obter uma política ótima a partir de um modelo de transição de estados (ou seja, a probabilidade de alternar de um estado x para um estado y). O agente necessita interagir com o ambiente, diretamente, para obter informações, que por meio de um algoritmo adequado pode produzir uma política Kaelbling et al. (1996).

Existem duas principais categorias quando tratamos de aprendizado por reforço:

- **Livre de modelo**: não apresenta um modelo de transição de estado;
- **Baseado em modelo**: a partir do modelo de transição de estados, o algoritmo aprende a controlar o ambiente.

Já Russell and Norvig (2013) segmentam o aprendizado por reforço de acordo com a abordagem de aprendizado utilizado:

- um agente baseado na utilidade: visa aprender uma função de utilidade sobre os estados, utilizando-a dessa forma para selecionar as ações que maximizem essa utilidade;
- um agente baseado em Q (Q-learning): aprende por meio de uma função **ação-valor**, denominada de Q, que visa fornecer a utilidade adequada para cada ação e estado;
- um agente reativo: mapeia diretamente estados para ações.

2.2 Algoritmos Analisados

Existem diferentes algoritmos que podem ser utilizados para tratar o problema abordado neste estudo. Caspi et al. (2017) propõem uma classificação para algoritmos de aprendizado por reforço. A seguir, na Figura 1, apresentamos a categorização destes algoritmos.

No que se segue, apresentamos os algoritmos que são **livres de modelo**, ou seja, não possuem um modelo de transição definido e que estão baseados na utilização de agentes **Q**, capazes de aprender uma função que mapeia os estados e ações.

2.2.1 *Deep Q Network (DQN)*

O algoritmo é uma variação do Q-Learning (Watkins and Dayan, 1992, Watkins, 1989), adaptada por Mnih et al. (2013), cujo o intuito é utilizar uma rede neural convolucional para aprender políticas a partir de imagens em ambientes complexos que utilizam a abordagem de aprendizado por reforço. Dessa forma, utiliza-se gradiente estocástico descendente para atualizar os pesos da rede e um mecanismo de experiência de repetição. A experiência de repetição, estabelecida em Lin (1992), visa reusar as experiências adquiridas pelo agente, relembrando ações passadas e apresentando essas para o algoritmo de aprendizado.

Para isso, considera-se um ambiente ε, uma sequência de ações, observações e recompensas. A cada passo de tempo, o agente deve selecionar uma ação a_t do conjunto de ações válidas presentes no ambiente e essa ação passa pelo ambiente, modifica o estado interno e retorna a recompensa.

2.2.2 *Double Deep Q Network (DDQN)*

Assim como a abordagem anterior, DQN, o *Double Deep Q-Networks* (DDQN) também é uma implementação que utiliza redes neurais. O principal problema em utilizar funções de valor que utilizam o valor máximo esperado

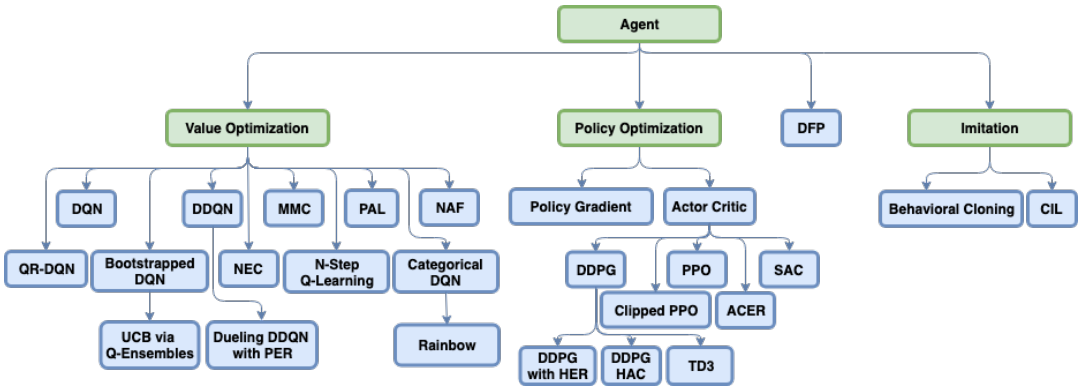

Figura 1: Relação dos algoritmos de aprendizado por reforço proposto por Caspi (Caspi et al., 2017).

é que podem acabar resultando em um viés positivo ou negativo, ou seja, os valores obtidos são tendenciosos e não representam de forma fidedigna o valor real.

Assim, os autores, van Hasselt et al. (2015), sugerem a parametrização da função de valor $Q(s,a;\theta_t)$, para facilitar o aprendizado dos valores das ações. A função clássica, após tomar uma ação A_t em um estado S_t e observar uma recompensa R_{t+1}, resultando em um estado R_{t+1} pode ser reescrita da seguinte forma (Eq. (1)):

$$\theta_{t+1} = \theta_t + \alpha(Y_t^Q - Q(S_t, A_t; \theta_t))\nabla_{Q_t} Q(S_t, A_t; \theta_t) \qquad (1)$$

Onde:

- α é um escalar que representa o tamanho do passo;
- e $Y_t^Q \equiv R_{t+1} + \gamma \max_a Q(S_{t+1}, a; \theta_t)$.

2.2.3 *Quantile Regression Deep Q Network (QRDQN)*

Semelhante ao DQN, esse algoritmo visa utilizar redes neurais para aprender uma política ótima. Segundo Dabney et al. (2017), as mudanças foram mínimas se comparadas ao DQN. Dentre essas modificações, destacam-se as seguintes:

- a arquitetura da rede neural é idêntica, apenas alterando a camada de saída, para o tamanho $|A|xN$, onde N é o hiper-parâmetro dado pela regressão quantil;
- a função de perda foi alterada.

A partir dessas alterações, obteve-se um algoritmo semelhante ao modificado, porém com a utilização da regressão quantil (Koenker and Hallock, 2001) e a alteração da função de perda para a rede neural.

2.2.4 *Categorical Deep Q Network (Categorical DQN)*

Este algoritmo mais complexo matematicamente que os anteriores, porém que segue a linha de todos presentes nesse trabalho e foi desenvolvido por Bellemare et al. (2017). Seguindo os moldes dos antecessores, a principal alteração ocorre na forma como é computado o resultado obtido a partir da equação de reforço.

A função de valor utilizada por esse algoritmo é composta por diversas outras. A equação de Bellman definida anteriormente é redefinida, se utilizarmos essa como um vetor em $\mathbb{R}^{XxA}$ e as recompensas obtidas também como vetor, podemos reescrever a equação da seguinte maneira (Eq. (2)):

$$TQ(x,a) := \mathbb{E}R(x,a) + \gamma \mathbb{E}_p max_{a' \epsilon A} Q(x', a') \qquad (2)$$

Onde:

- x representa o estado;
- a a ação;
- $R(x,a)$, a recompensa esperada dado o estado x e a;
- γ o fator de desconto.

A partir dessa equação, pode-se projetar em seu suporte, diminuindo a complexidade.

2.2.5 *N-Step Q Learning (NStep DQN)*

Outra variação do algoritmo DQN, o n-step Q-learning, elaborado por Mnih et al. (2016), onde a função Q(s,a) é atualizada por meio de n passos definida como $r_t + \gamma_{t+1} + ... + \gamma^{n-1} r_{t+n-1} + \max_a \gamma^n Q(s_{t+n}, a)$, ou seja, o resultado de uma recompensa afeta diretamente os n valores precedentes de pares de ações e estados. Dessa forma, a propagação dessas recompensas é mais relevante e mais eficiente para a abordagem de reforço.

Ainda é destacado uma abordagem assíncrona, que define e atribui nome ao método. É possível determinar n, que serão executados em paralelo, resultando em um valor para a função. Inicialmente, o algoritmo seleciona uma ação de acordo com uma política de exploração até um número máximo de passos pré-definido ou até um estado terminal ser alcançado (entende-se por terminal, situações que finalizam o jogo, como o personagem perder todas as vidas ou até mesmo finalizar o ambiente). Após a realização da ação, o agente recebe esse total de recompensas do ambiente e computa um gradiente para cada par de estado e ação desde a última atualização.

3 Metodologia

Esta seção descreve a metodologia empregada no trabalho. Primeiramente, o jogo utilizado é apresentado, mostrando as principais características. Por fim, as métricas que são utilizada para analisar os algoritmos são descritas.

3.1 Estudo de caso

O jogo escolhido como estudo de caso foi o Pac-man (Pittman, 2015). Esse jogo foi licenciado pela *Midway* na América[2].

Segundo Pittman (2015), o objetivo do personagem é simples: é possível se movimentar nas quatro direções com o intuito de guiar o *Pac-man* através do labirinto preenchido por pontos para serem engolidos. Existem quatro monstros fantasmas que perseguem o personagem principal para captura-lo. O intuito deve ser devorar todos os pontos tentando evitar os inimigos mortais.

Cada etapa consiste em um ciclo onde o protagonista possui três vidas - categorizando um episódio - e sempre que é capturado, o número é reduzido em um. Vale ressaltar, que neste estudo consideramos um total de **dez mil episódios**. Caso o número seja reduzido a zero e ele é pego novamente, o jogo acaba.

Ao coletar todos os pontos presentes no ambiente, todos os elementos são postos em sua posição inicial e o jogo recomeça. A novo turno após a coleta, os inimigos ficam cada vez mais mortais e mais rápidos, alterando assim, além de sua velocidade, o tempo em que eles podem ser derrotados pelo protagonista.

Por mais que o jogo seja baseado em histórias infantis e tenha uma temática leve, o comportamento dos inimigos podem vir a ser complexos. Cada fantasma possui um comportamento único e três modos operantes:

- **perseguição**: nesse modo, cada um possui um comportamento distinto para perseguir e capturar *Pac-man* pelo labirinto.
- **dispersão**: os fantasmas desistem da perseguição por um breve período de tempo e acabando se posicionando nos cantos do labirinto. Após um determinado tempo, eles voltam para a etapa anterior.
- **assustado**: o último modo ocorre apenas quando o protagonista come um energizador, deixando os inimigos assustados. Nos níveis iniciais, as cores dos inimigos mudam, indicando que estão vulneráveis.

Ainda, segundo Pittman (2015), Toru Iwatani tinha o costume de fazer um jogo de palavra com as temáticas presentes em suas obras. O fantasma vermelho, conhecido como **Blinky**, era representado na sua versão original pela palavra *oikake*, o que significa atropelar ou perseguir. Dentre os quatro, esse é o mais agressivo e persegue o agente diretamente pelo labirinto. No seu modo de perseguição, o destino é sempre o nosso herói. Além disso, a velocidade de *Blinky* varia conforme os pontos são consumidos pelo cenário.

Já o fantasma rosa, denominado de *Pinky*, é caracterizado por tentar emboscar o personagem principal. No Japão, é definido como um personagem *machibuse*, ou seja, um personagem que visa realizar uma emboscada. No seu modo de perseguição, **Pinky** não visa atacar diretamente o jogador. A sua movimentação é sempre destinada alguns passos a frente e nunca diretamente ao personagem.

O fantasma azul claro é o mais perigoso dentre todos. Nomeado de **Inky**, na versão americana, é o mais maleável entre seus companheiros. Definido em sua versão original como *kimagure*, ou excêntrico, esse possui o comportamento mais mutável. Dessa forma, ele pode mudar de comportamento para qualquer um dos fantasmas de forma imprevisível. No seu modo de perseguição, o destino é definido de forma complexa: para descobrir onde *Inky* se moverá, será necessário a posição atual do *Pac-man* e orientação (para onde está se movendo) e a posição final de seu colega *Blinky*, para assim intermediar dois passos para frente do protagonista.

O último fantasma, **Clyde**, é caracterizado na sua origem como *otoboke*, ou fingindo ignorância. A velocidade se equipara com a de seus outros companheiros, mas o comportamento é totalmente distinto. Em vez de perseguir diretamente, ele visa fazer sua patrulha nos cantos do labirinto. O seu modo de perseguição é baseado na distância (Euclidiana) atual até o *Pac-man*, que caso seja superior a um número delimitado de blocos, o personagem seguirá na sua patrulha.

O ambiente utilizado foi disponibilizado por meio de uma *toolkit* gratuita, que possui a implementação de diversos jogos para fins de inteligência artificial, conforme Brockman et al. (2016). A representação segue fielmente as informações expostas anteriormente, como a movimentação dos inimigos do personagem principal.

3.2 Métricas

Para avaliar os resultados obtidos a partir da realização dos testes, foram utilizadas as seguintes métricas de avaliação:

- Completude do cenário pelo o agente;
- Recompensa obtida;
- Total de passos realizados para a finalização da etapa;
- Tempo de execução de cada simulação (total de episódios).

Entende-se como relevante a análise de completude ou não de um cenário, para validar se o agente realmente conseguiu aprender de forma satisfatória. Porém, essa métrica pode não se mostrar muito eficiente para jogos que não possuem um final pré-definido, ou seja, não existe nenhuma sinalização clara para o agente a respeito da conclusão.

Além disso, como os algoritmos para aprendizado por reforço são baseados na avaliação de uma recompensa para determinadas ações, é imprescindível a avaliação das recompensas obtidas. Tal como, o desvio padrão referente a essa variável, a média, o valor mínimo e o valor máximo.

Com o intuito de ter uma avaliação mais precisa do desempenho, é necessário ainda analisar o total de passos realizados até a finalização da etapa e o tempo total de execução da simulação. O mecanismo referente ao total de passos foi proposto com o intuito de descobrir quantas ações são necessárias para que cada algoritmo complete o cenário, ou no caso de ambientes que não possuem um final bem definido, obter a maior recompensa.

A métrica referente ao tempo de execução segue o mesmo intuito, avaliar o tempo necessário (leia-se o total de episódios) necessários para a finalização de um cenário. O total de episódios pode variar de acordo com a técnica usada para a aprendizagem, então é necessário observar

[2] https://www.arcade-museum.com/

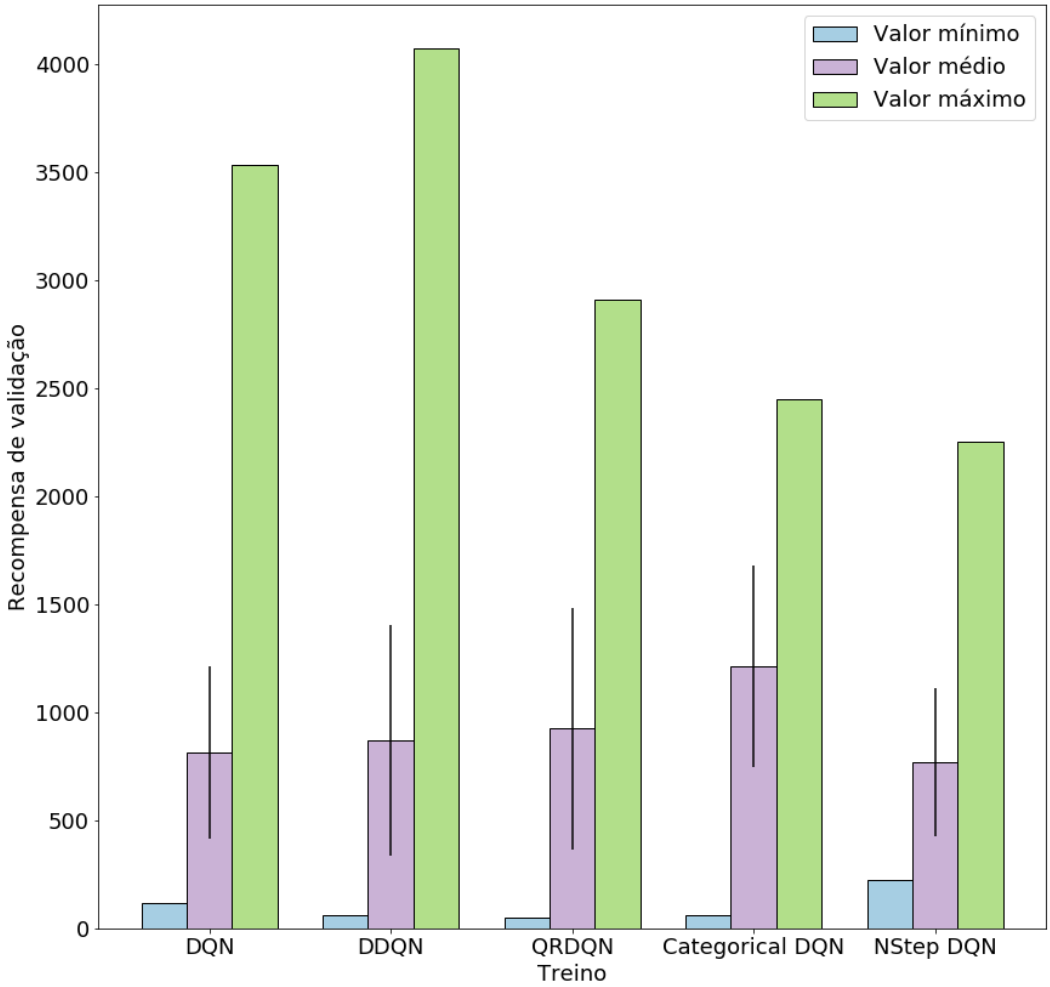

Figura 2: Comparação entre os algoritmos testados ao observar a variável recompensa de validação

para cada método utilizado o total de iterações realizadas para a finalização do cenário ou obtenção de uma recompensa expressiva.

4 Resultados obtidos

Como mencionado anteriormente, os resultados são considerados dez mil episódios, para realizar cada a simulação de algoritmo, tornando assim, a métrica definida anteriormente (tempo de execução para simulações), um valor estabelecido.

A Fig. 2 visa comparar o resultado obtido entre todos os algoritmos simulados para a variável recompensa. No eixo das abscissas estão as abordagens presentes na comparação, sendo elas: DQN, DDQN, QRDQN, Categorical DQN e N-Step DQN. No eixo das ordenadas está a variável de estudo para o gráfico. Para fator de comparação, utilizou-se os valores da última fase do treino dividido em etapas para o algoritmo DQN. Foram analisados resultados como: valor máximo, valor mínimo e valor médio (e seu desvio padrão).

A Tabela 1 demonstra os resultados obtidos. As linhas representam os algoritmos e seus resultados, enquanto as colunas, as variáveis analisadas. O valor mínimo foi similar em diversos casos, como: DDQN, QRDQN e Categorical DQN. A abordagem de múltiplos agente, foi 54% superior ao resultado mais próximo. Já ao analisar os valores médios, todos se mantiveram próximos, com exceção dos resultados obtidos a partir do Categorical DQN, que foi cerca de 31% maior que o valor mais próximo (QRDQN). Já para o valor máximo, destacou-se a abordagem DDQN, possuindo um valor 14% superior ao seu antecessor.

Utilizou-se da mesma abordagem comparativa para o tamanho do episódio. A Fig. 3 compara todos os resultados obtidos para a variável citada, onde no eixo X estão dispostos os algoritmos comparados, enquanto no eixo Y está o objetivo de estudo em questão.

Todos os resultados para essa variável se mostraram próximos, sendo difícil de realizar uma distinção, como pode-se verificar na Tabela 2. Tendo em vista que todos os algoritmos são abordagens que não utilizam da otimização de política, o resultado foi próximo ao esperado. Destaca-se ainda, o que havia sido citado para a recompensa: o desvio padrão permaneceu alto.

Os valores centrais foram estritamente próximos para

Tabela 1: Valores mínimo, médio, máximo e desvio padrão para os algoritmos testados ao observar a variável recompensa de validação

	Valor mínimo	Valor médio	Valor máximo	Desvio padrão
DQN	120	813	3530	398
DDQN	60	869	4070	532
QRDQN	52	924	2908	559
Categorical DQN	60	1213	2448	468
N-Step DQN	222	769	2250	344

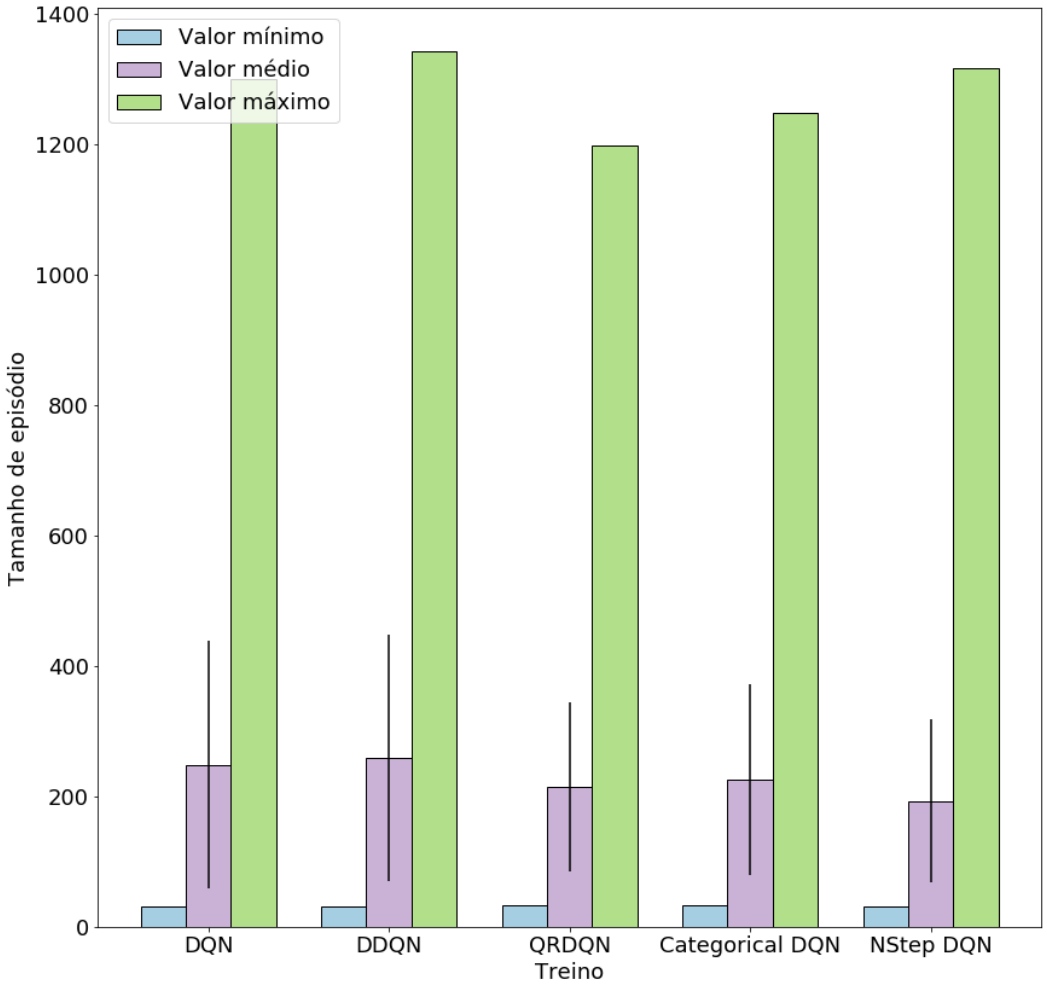

Figura 3: Comparação entre os algoritmos testados ao observar a variável tamanho de episódio

Tabela 2: Valores mínimo, médio, máximo e desvio padrão para os algoritmos testados ao observar a variável tamanho de episódio

	Valor mínimo	Valor médio	Valor máximo	Desvio padrão
DQN	32	249	1301	190
DDQN	32	259	1343	189
QRDQN	33	215	1199	130
Categorical DQN	33	227	1248	146
N-step DQN	32	193	1317	125

todos os casos. O algoritmo que explorou o cenário de forma superior, em seu melhor caso, foi o DDQN, seguido pelo N-Step. Os valores mínimos foram próximos para todos os casos. Vale ressaltar que nenhuma das abordagens foi capaz de concluir o cenário, ou seja, passar para a próxima fase do ambiente.

5 Conclusões

O aprendizado por reforço é uma abordagem que utiliza da tentativa e erro para obter um melhor desempenho em ambientes complexos. Esses métodos de reforço estão diretamente correlacionados com os jogos, tendo em vista que são aplicados em diversos jogos.

Neste trabalho foram utilizadas métricas para a validação dos resultados obtidos a partir da simulação. Entre as métricas analisadas estão: as recompensas obtidas pelo agente, a exploração do mapa, a completude do cenário e o tempo de simulação. Para a recompensa e o tamanho do episódio, foram comparados valores referentes a valores mínimo, médio, máximo e desvio padrão. Já para o número de episódios para finalização, foi definido o número máximo a ser testado, dez mil episódios. Para o cado da finalização propriamente dita, esta não ocorreu, nenhum dos agentes conseguiu concluir o ambiente com nenhum dos algoritmos testados.

Cabe ressaltar ainda que neste artigo foram utilizados 5 diferentes algoritmos de aprendizado por reforço que não utilizam otimização de reforço. Desta forma, o presente trabalho pode servir de base para estudos de algoritmos de RL além de possibilitar a comparação com diferentes abordagens (e utilizando outras métricas de avaliação): DDPG, PPO, TD3, CIL e outras.

Para a recompensa e o tamanho do episódio, o desvio padrão foi elevado devido a aleatoriedade do ambiente. O comportamento dos algoritmos, nas etapas iniciais da simulação se mostrava similar a uma função linear e, de acordo com o decorrer dos testes, esse comportamento parecia se tornar uma distribuição não-linear. Ao avaliar os valores máximos obtidos pelas recompensas, os resultados foram promissores.

Conclui-se, por meio desse trabalho, que as abordagens de aprendizado por reforço não se mostraram eficientes para cenários que envolvem comportamentos com imprevisibilidade por parte do ambiente, tendendo a demorar muito a sua convergência, caso ocorra. Cada método se destacou de forma distinta para os valores analisados. Dessa forma, o objetivo principal do trabalho foi parcialmente alcançado, o qual era determinar uma abordagem que melhor se adequasse ao problema.

Este trabalho pode ser utilizado como base para novas analises, porém, é possível utilizar uma gama maior de algoritmos e diferentes abordagens, como proposto por Caspi et al. (2017) e apresentado na Fig. 1.

Como trabalhos futuros vislumbra-se utilizar abordagens orientadas a política (DOPG, PPG, etc., conforme Fig. 1), para o mesmo ambiente ou até mesmo ambientes distintos, comparado aos resultados obtidos com este trabalho.anteriormente.

Referências

Bellemare, M. G., Dabney, W. and Munos, R. (2017). A distributional perspective on reinforcement learning, *CoRR* **abs/1707.06887**. http://arxiv.org/abs/1707.06887.

Bellemare, M. G., Naddaf, Y., Veness, J. and Bowling, M. (2012). The arcade learning environment: An evaluation platform for general agents, *CoRR* **abs/1207.4708**. http://arxiv.org/abs/1207.4708.

Bom, L., Henken, R. and Wiering, M. (2013). Reinforcement learning to train ms. pac-man using higher-order action-relative inputs, *2013 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pp. 156–163. https://doi.org/10.1109/ADPRL.2013.6615002.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J. and Zaremba, W. (2016). OpenAI Gym. https://doi.org/10.48550/arXiv.1606.01540.

Brown, N., Bakhtin, A., Lerer, A. and Gong, Q. (2020). Combining deep reinforcement learning and search for imperfect-information games, *CoRR* **abs/2007.13544**. https://doi.org/10.48550/arXiv.2007.13544.

Caspi, I., Leibovich, G., Novik, G. and Endrawis, S. (2017). Reinforcement learning coach. https://doi.org/10.5281/zenodo.1134899.

Crawford, C. (2011). The Art of Computer Game Design (eletronic version).

Dabney, W., Rowland, M., Bellemare, M. G. and Munos, R. (2017). Distributional reinforcement learning with quantile regression, *CoRR* **abs/1710.10044**. http://arxiv.org/abs/1710.10044.

Huizinga, J. (1971). *Homo ludens: o jogo como elemento da cultura*, Vol. 4, Editora da Universidade de S. Paulo, Editora Perspectiva.

Juul, J. (2011). *Half-real: Video games between real rules and fictional worlds*, MIT press.

Kaelbling, L. P., Littman, M. L. and Moore, A. W. (1996). Reinforcement Learning: A Survey, *Journal of Artificial Intelligence Research* 4 pp. 708–713. https://doi.org/10.1613/jair.301.

Koenker, R. and Hallock, K. F. (2001). Quantile regression, *Journal of economic perspectives* **15**(4): 143–156. https://doi.org/10.1257/jep.15.4.143.

Lin, L.-J. (1992). Reinforcement learning for robots using neural networks, Carnegie Mellon University, USA. UMI Order No. GAX93-22750.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T. P., Harley, T., Silver, D. and Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning, *CoRR* **abs/1602.01783**. https://doi.org/10.48550/arXiv.1602.01783.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D. and Riedmiller, M. A. (2013). Playing atari with deep reinforcement learning, *CoRR* **abs/1312.5602**. https://doi.org/10.48550/arXiv.1312.5602.

Pittman, J. (2015). The pac-man dossier. Disponível em: https://pacman.holenet.info/.

Pursuit (2021). Manual do jogo. Disponível em https://archive.org/details/ArcadeGameManualPursuit/page/n23.

Qwark (2021). Manual do jogo. Disponível em https://archive.org/stream/ArcadeGameManualQwak/qwak#mode/2up.

Russel, S. and Norvig, P. (1995). *Artificial Intelligence: A Modern Approach*, Alan Apt, New Jersey.

Russell, S. and Norvig, P. (2013). *Inteligência Artificial*, 3º edn, Elsevier Editora Ltda, Rio de Janeiro.

Schwab, B. (2004). *AI Game Engine Programming*, 1 edn, Charles River Media, Hingham.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcmenet Learning An Introduction*, 2º edn, The MIT Press, Cambridge, MA.

Tziortziotis, N., Tziortziotis, K. and Blekas, K. (2014). Play ms. pac-man using an advanced reinforcement learning agent, *in* A. Likas, K. Blekas and D. Kalles (eds), *Artificial Intelligence: Methods and Applications*, Springer International Publishing, Cham, pp. 71–83. https://doi.org/10.1007/978-3-319-07064-3_6.

van Hasselt, H., Guez, A. and Silver, D. (2015). Deep reinforcement learning with double q-learning, *CoRR* **abs/1509.06461**. https://doi.org/10.48550/arXiv.1509.06461.

Watkins, C. J. C. H. (1989). Learning from delayed rewards.

Watkins, C. J. C. H. and Dayan, P. (1992). Techincal Note: Q-Learning, *Machine Learning* **8**(3-4): 279–292. https://doi.org/10.1023/A:1022676722315.