Avaliação de topologia da rede neural e parâmetros do aprendizado de máquina por reforço para agentes jogadores de videogames

Mateus de Freitas Rosa

UFU

Universidade Federal de Uberlândia
Faculdade de Computação
Bacharelado em Sistemas de Informação

Monte Carmelo - MG
2022

Mateus de Freitas Rosa

Avaliação de topologia da rede neural e parâmetros do aprendizado de máquina por reforço para agentes jogadores de videogames

Trabalho de Conclusão de Curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, Minas Gerais, como requisito exigido parcial à obtenção do grau de Bacharel em Sistemas de Informação.

Área de concentração: Sistemas de Informação

Orientador: Professor Dr. Leandro Nogueira Couto
Coorientador: Professor Dr. Murillo Guimarães Carneiro

Monte Carmelo - MG
2022

Este trabalho é dedicado a minha família, em especial meus pais por nunca medirem esforços para me incentivar, compreender, auxiliar, etc. Graças a eles sou quem sou. Ao meu irmão que sempre esteve ao meu lado para me apoiar, ajudar e ensinar. Ao meu orientador que dispôs até de finais de semana para me ajudar, ensinar e apoiar.

Agradecimentos

Agradeço aos meus pais por construírem quem sou hoje, e por estarem sempre ao meu lado independente das circunstancias.

Agradeço ao meu irmão Luis, que nunca mediu esforços para me ajudar em minhas dificuldades. Sempre esteve junto de mim para apoiar, ajudar e ensinar.

Agradeço minha família, em especial meus pais e irmãos que sempre me incentivaram, compreenderam e auxiliaram no meu percurso.

Agradeço ao meu orientador Leandro Couto por aceitar conduzir o meu trabalho de pesquisa. Dispor até de finais de semana para me auxiliar em minhas dificuldades. Me apresentar e ensinar à área de inteligência artificial a qual me identifico hoje.

Agradeço aos professores, pela correção e ensinamentos que me permitiram apresentar um melhor desempenho de formação profissional.

Agradeço aos meus colegas e amigos, que juntos compartilhamos momentos de aprendizado, amizades e companheirismo.

Agradeço a todos que direta ou indiretamente participaram do desenvolvimento deste trabalho.

Agradeço ao Criador por me permitir essa experiência incrível da vida.

"Acredito em Deus, todos os outros devem apresentar dados."
(Edwin R. Fisher e W. Edwards Deming)

Resumo

O aprendizado de máquina por reforço é uma área importante no ramo de inteligência artificial devido a sua versatilidade e adaptabilidade em cenários complexos, sem necessitar de um supervisor especialista. O uso de imagens como sinais de entrada aproxima o agente inteligente da forma em que o ser humano toma decisões, utilizando-se da riqueza de informações da imagem.

Em aprendizado por reforço com redes neurais, o desempenho depende muito da topologia da rede neural e de diversos parâmetros do aprendizado. Nesse sentido, este trabalho visa desenvolver um agente inteligente para jogos de Atari 2600 e também propõe alterações de topologia e parâmetros do aprendizado em relação a métodos da literatura.

Os resultados experimentais obtidos apresentam sucesso na evolução do agente na maximização das recompensas, mostrando que as topologias e parâmetros propostos são viáveis.

Palavras-chave: Aprendizado de máquina por reforço, Agente jogador de videogame, Inteligência Artificial, Redes Neurais Convolucionais, Visão computacional.

Abstract

Reinforcement machine learning has been an important area in the field of artificial intelligence due to its versatility and adaptability in complex settings, without the need of a specialist supervisor. The use of images as input signals makes the intelligent agent similar to how a human being makes decisions, using the image's rich information.

In reinforcement learning with neural networks, performance is highly dependant on the topology of the neural network and several lerarning parameters. Therefore, this aims to develop an intelligent agent for playingt Atari 2600 videogames, while also proposing changes to the topology and reinforcement learning parameters comapred to literature methods.

Experimental results were successful in the evolution of the agent in maximizing rewards, which shows the proposed topology and parameters are viable.

Lista de ilustrações

Lista de tabelas

Sumário

Capítulo 1

Introdução

O aprendizado de máquina por reforço tem sido uma área de pesquisa importante no ramo de Inteligência Artificial em razão de sua versatilidade e de sua adaptabilidade para a exploração de ambientes de decisão em problemas, e pelo alto potencial para superar o desempenho humano, sem a necessidade de um supervisor especialista. Videogames fornecem ótimos ambientes para aplicação de aprendizado de máquina por reforço, pois são compostos por desafios complexos e por situações comparáveis a realidade.

A aplicabilidade do aprendizado por reforço no mundo real é vasta, estendendo-se desde sistemas de otimização de tarefas empresariais até agentes de inteligência artificial jogadores de videogames. O uso do aprendizado por reforço no ambiente de jogos eletrônicos é interessante pois fornece ambientes e situações comparáveis às do mundo real, sendo ótimos simuladores para o desenvolvimento de agentes inteligentes a partir do aprendizado por reforço sem auxílio de um especialista, podendo assim superar habilidades de um ser humano. Além disso, um agente que explora o ambiente e adquire experiências ao longo do jogo, sem limitações externas de um especialista, tem a poderosa capacidade de encontrar novas formas únicas de atingir um determinado objetivo que potencialmente não tenham sido previstas por agentes humanos (DUTTA, 2018).

No ambiente de jogos de tabuleiro, por exemplo, em específico no jogo Go, a inteligência artificial AlphaGo (SILVER et al., 2016) desenvolvida pela equipe DeepMind da empresa Google, venceu o melhor jogador de Go do mundo, utilizando-se do aprendizado por reforço para otimizar suas habilidades a partir do aprendizado supervisionado. Isso é significativo porque Go é considerado por muitos o jogo de tabuleiro mais complexo já criado, com muito mais movimentos possíveis por jogada do que o xadrez, e "mais configurações possíveis de tabuleiro do que o número de átomos do universo"[1]. No entanto, mais tarde, em outra pesquisa desenvolvida pela mesma empresa, foi apresentada a inteligência artificial AlphaGo Zero (SILVER et al., 2017), que conseguiu superar sua antecessora utilizando abordagens mais puras do aprendizado por reforço, treinando-a do zero. Além de jogos de tabuleiro, o aprendizado de máquina por reforço também obteve

[1] https://www.scientificamerican.com/article/how-the-computer-beat-the-go-master/

bons resultados em jogos de videogame, o que é apresentado no trabalho de Mnih (MNIH et al., 2015), onde foram utilizados 49 jogos do videogame Atari 2600 para o treinamento de agentes jogadores.

A utilização de informações visuais aplicada a agente jogadores de videogame tem recebido destaque na literatura. Nesse sentido a utilização de imagens ricas em informações possibilita o agente a emular habilidades humanas (o aprendizado por reforço assemelha-se à forma pela qual um ser humano toma decisões em um jogo a partir da interpretação de uma serie de imagens). Conforme a capacidade de memória e o poder computacional aumentam, torna-se mais acessível o uso dessas informações e dessa tecnologia. Recentes publicações apresentam o desempenho de agentes inteligentes utilizando informações visuais junto a redes neurais convolucionais (JUSTESEN et al., 2019; MNIH et al., 2015; KEMPKA et al., 2016; LAMPLE; CHAPLOT, 2017).

1.1 Motivação

Agentes inteligentes capazes de realizar tarefas desafiadoras e/ou inéditas apenas com a própria experiência, sem auxílio de um especialista humano, evidenciam uma importante área da inteligência artificial. O aprendizado de máquina por reforço também é versátil, capaz de se adaptar a mudanças na função de recompensa, semelhante ao modo de aprendizado humano, reagindo dinamicamente a novas situações. Videogames fornecem ambientes de complexidade e de situações semelhantes as encontradas no mundo real, de forma que o estudo de inteligências artificiais para videogames contribui para a resolução de diversos problemas da área de aprendizado de máquina ainda abertos na literatura, como exploração, planejamento e aprendizado de representação e de modelo (MACHADO et al., 2018). Jogos eletrônicos, em específico de Atari 2600, apresentam ambientes de recompensas esparsas, não-óbvias, e amplo espaço de decisão. A utilização de informações visuais como entrada da rede também explora a visão computacional, uma área de alto interesse de pesquisa devido à riqueza de informações das imagens, e à sua complexidade, e também aproxima ainda mais o aprendizado do agente ao processo de aprendizado humano, o que parece combinar aprendizado por reforço com sistemas sensoriais de processamento, sendo a visão um dos sentidos mais importantes (MNIH et al., 2015).

1.2 Problema

A topologia perfeita de uma rede neural é um tema complexo e obscuro, uma vez que a própria rede neural é considerada como uma caixa preta. Não é uma tarefa fácil entender o que ocorre entre os nerônios e os pesos. Dessa forma o trabalho propõe encontrar uma topologia e parâmetros que possam ser aplicados, sem modificações, ao aprendizado de

mais de um jogo, utilizando as mesmas configurações. Recompensas esparsas também são um problema na área de aprendizado por reforço, pois os sinais de recompensas precisam ser transmitidos e relacionados com as ações que os produziram, um problema comum nos videogames que os tornam um problema desafiador.

1.3 Objetivos

O objetivo geral deste trabalho é o desenvolvimento do algoritmo de um agente inteligente capaz de jogar jogos eletrônicos, através da aplicação de aprendizado de máquina por reforço profundo, utilizando informação visual e redes neurais convolucionais, e também utilizando hardware e tempo de processamento limitados. Com políticas bem estabelecidas de recompensas, espera-se que o agente seja capaz de aprender e de jogar jogos baseando-se exclusivamente na informação visual dos estados do jogo. Também é esperado que o algoritmo utilizados nesta pesquisa seja capaz de se adaptar a mais de um jogo, desde que estes jogos não sejam muito diferentes em questão de ambiente, de objetivo e de políticas de recompensas, utilizando-se dos mesmos parâmetros e da mesma arquitetura de rede neural convolucional. O agente será treinado e avaliado no jogo 2D Pong através da ferramenta e ambiente de desenvolvimento de aprendizado de máquina por reforço Gym OpenAI (BROCKMAN et al., 2016), que inclui um framework para treinamento e teste de agentes de jogos do videogame Atari 2600. Por fim, como métrica de resultados e avaliações, serão realizadas comparações de topologia da rede neural e da influência de parâmetros considerados na pesquisa.

1.4 Objetivos específicos

1. Desenvolver o pré-processamento de imagens, com o objetivo de diminuir o poder computacional necessário para processá-las.

2. Implementar todo o código de um agente inteligente que utiliza do aprendizado de máquina por reforço e que esse agente seja capaz de jogar mais de um jogo com os mesmos parâmetros e a mesma estrutura de rede neural convolucional.

3. Determinar os efeitos de alteração de parâmetros no desempenho do agente.

4. Avaliar o desempenho do agente treinado, avaliando o quão factível é o aprendizado de máquina para solução do problema com recursos de hardware e tempo de treinamento limitados.

1.5 Organização da Monografia

Os capítulos seguintes estão organizados da seguinte forma: o capítulo 2 traz a fundamentação teórica e uma explicação das técnicas e métodos de aprendizado de máquina da literatura que serão usados neste trabalho; o capítulo 3 descreve a metodologia usada no desenvolvimento da proposta deste trabalho, destacando os parâmetros propostos para serem modificados nos experimentos; o capítulo 4 apresenta os experimentos realizados, os resultados obtidos, e uma discussão deles; no capítulo 5 são apresentadas as conclusões do trabalho, bem como propostas de trabalhos futuros para continuidade da pesquisa.

Capítulo 2

Fundamentação Teórica

2.1 Aprendizado de máquina por reforço

O aprendizado de máquina por reforço difere dos aprendizados supervisionado e não supervisionado por se basear em otimização de recompensas e não em tarefas, classificação ou agrupamento de classes. Uma das maiores diferenças entre o aprendizado por reforço e o aprendizado supervisionado é que no aprendizado por reforço a exploração do espaço de decisão é mais visível (KAELBLING; LITTMAN; MOORE, 1996). Além disso o aprendizado por reforço é uma técnica de aprendizagem para qualquer tipo de situações sequenciais, como um jogo que depende de uma sequência de ações para atingir um determinado objetivo (DUTTA, 2018). Conforme afirma Mnih (MNIH et al., 2015, p. 1, tradução nossa), "A teoria da aprendizagem por reforço fornece uma explicação normativa profundamente enraizada em perspectivas psicológicas e neurocientíficas sobre o comportamento animal, de como os agentes podem otimizar seu controle de um ambiente".

No aprendizado por reforço o agente deve ser capaz de desenvolver um comportamento inteligente a partir de iterações, de tentativas e erros em um ambiente sequencial e dinâmico (KAELBLING; LITTMAN; MOORE, 1996). O objetivo do agente é encontrar um conjunto de ações que maximize o ganho futuro na função de recompensa (MNIH et al., 2015). Cada ação aplicada a um estado do ambiente produz uma recompensa e um próximo estado (SUTTON; BARTO, 2018). Essas recompensas consistem em sinais positivos, negativos ou neutros, os quais representam incentivos obtidos pelo agente após a execução de uma determinada ação no ambiente. Conforme exposto na literatura (Machado et al.; 2018), é necessário o incentivo à exploração do ambiente a fim de evitar máximos locais (Taïga et al.; 2019).

2.2 Agentes Jogadores de Videogames

A aplicação de agentes inteligentes jogadores não é nova, como é apresentado no trabalho seminal de Arthur Samuel com seu agente inteligente jogador de damas que obteve bons resultados contra jogadores experientes (SAMUEL, 1959). Tempos depois, a partir avanços em pesquisas e em poder computacional, foi apresentada a inteligencia artificial Alpha Go (SILVER et al., 2016), desenvolvida pelo grupo DeepMind da Google. A AlphaGo foi treinada com o aprendizado de máquina supervisionado utilizando históricos de movimentos de peças do jogo Go realizados por especialistas, além da adoção do aprendizado de máquina por reforço para otimizar as jogadas do agente. Isso possibilitou que a AlphaGo fosse capaz de vencer o campeão mundial de Go. No entanto, logo no ano seguinte ela foi superada por sua sucessora AlphaGo Zero (SILVER et al., 2017). A AlphaGo Zero foi treinada utilizando apenas aprendizado de máquina por reforço, permitindo que o agente iniciasse o jogo sem nenhum conhecimento e que adquirisse ao longo do treinamento.

Jogos de videogames tentam simular situações de complexidades próximas às do mundo real, com ambientes complexos, reações físicas, objetivo de jogos, recompensas esparsas, grande espaço de decisão, etc, os quais exigem uma certa habilidade do jogador. Dessa forma, jogos de videogame são ótimos simuladores de ambientes para a aplicação de agentes inteligentes, como é apresentado no trabalho de Mnih (MNIH et al., 2015).

2.2.1 Agentes baseados em aprendizado de máquina supervisionado

O aprendizado supervisionado depende de um especialista para gerar exemplos de treinamento a um modelo. Esses exemplos contêm estados e objetivos; ou seja, dado um estado atual, qual seria a ação do especialista de acordo com seus exemplos. No treinamento, o modelo é requisitado a tomar uma decisão de acordo com um estado conhecido pelos exemplos do especialista. Após, é calculado o erro, comparando a resposta do agente com a resposta conhecida e é aplicado a correção dos pesos da rede neural de acordo com erro. Esse processo é realizado até que a rede consiga se adaptar aos exemplos fornecidos pelo especialista.

A aplicação do aprendizado supervisionado depende de exemplos fornecidos geralmente por um ser humano; ou seja, são gravadas ações, estados do jogo e outras informações adicionais enquanto o jogador humano joga (SILVER et al., 2016). Devido a essa necessidade, o processo para o treinamento do agente pode demorar, a não ser que o ser humano seja substituído por outra máquina capaz de agilizar o processo. Porém, para isso o problema já teria que ter sido resolvido por outra máquina. Além disso, um agente treinado apenas com o aprendizado supervisionado é limitado ao conhecimento do especialista, sendo incapaz de encontrar novas jogadas campeãs possíveis. Por esse motivo, frequentemente

agentes baseados em aprendizado supervisionado são combinados com aprendizado por reforço para romper a limitação do conhecimento do especialista. (SILVER et al., 2017; SHELHAMER et al., 2016)

2.2.2 Agentes baseados em aprendizado de máquina por reforço

O aprendizado por reforço não depende de exemplos de um especialista. Todo conhecimento do agente é obtido a partir de sua própria experiência jogando o jogo. O agente é treinado de acordo com os sinais de recompensas fornecidas pelo ambiente. Se uma ação é interessante para alçar o objetivo, o agente recebe um sinal de recompensa positivo; do contrário, recebe um sinal negativo. Caso não faça diferença, recebe um sinal nulo. Ademais o aprendizado por reforço também necessita de métodos de exploração do ambiente para evitar máximos locais.

A aplicação do aprendizado por reforço não depende de exemplos de um especialista. Desta forma não é necessária a etapa de coleta de exemplos de jogadas prévias já que o agente aprende enquanto joga. No entanto, o agente necessita de um sinal de recompensa vindo do ambiente Esse sinal, pode ser produzido com muita ou pouca frequência. Jogos são ótimos ambientes para a aplicação de agentes baseados no aprendizado por reforço, já que recompensam positivamente ações que se aproximam do objetivo e recompensam negativamente ações que se distanciam do objetivo. Jogos com recompensas esparsas, ou seja, obtidas com pouca frequência, geralmente quando o agente vence ou perde o jogo, são um desafio na área de pesquisa, pois se torna necessária a propagação do sinal de recompensa para as ações que levaram o agente até o estado que o recompensou (KAELBLING; LITTMAN; MOORE, 1996). Além disso, com iniciativas de incentivos a exploração, o agente é capaz de evitar máximos locais e de encontrar conjuntos de ações inéditas, que não seriam possível com um aprendizado que necessita de exemplos de um especialista.

2.3 Redes Neurais Artificiais (NN)

Rede Neural Artificial é um algoritmo de aprendizado de máquina que, em teoria, tenta simular o sistema neural biológico. A primeira Rede Neural nasceu com os trabalhos de Warren McCulloch e Walter Pitts em 1943, no artigo seminal (MCCULLOCH; PITTS, 1943). No artigo, os autores descrevem como uma rede neural biológica deve funcionar e, a partir destas observações, modelaram a primeira rede neural artificial registrada com o uso de circuitos elétricos. A pesquisa foi importante tanto para o estudo de processos neurais biológicos quanto para o estudo sobre inteligências artificiais. (HAYKIN; NETWORK, 2004).

Uma Rede Neural Artificial de forma resumida, é uma representação matemáticas de neurônios biológicos e de suas interligações. Ela é organizada por camadas, sendo

entrada, saída e oculta (essa última consiste em toda e qualquer camada entre a entrada e a saída). Essas camadas interligadas formam um fluxo de entrada de informações e de saída de respostas, que, entre estas, podem ou não existir camadas ocultas. A Figura 1 ilustra uma Rede Neural com camadas de entrada, oculta e saída.

- ❏ A camada de entrada não é composta por neurônios artificiais, mas por entrada de dados.
- ❏ A camada oculta é composta por conjuntos de neurônios que são interligados à camada de entrada e também à próxima camada oculta, se houver; se não houver, são interligados diretamente à camada de saída. A camada oculta é responsável por permitir que a rede neural consiga encontrar padrões em conjuntos de dados não lineares; ou seja, uma rede neural artificial composta apenas por camada de entrada e saída é capaz de modelar apenas problemas lineares.
- ❏ A camada de saída é composta por um conjunto de neurônios que satisfaça a necessidade e a modelagem do problema. A camada de saída é responsável por entregar uma resposta para a qual foi modelada.

(KIRK, 2017)

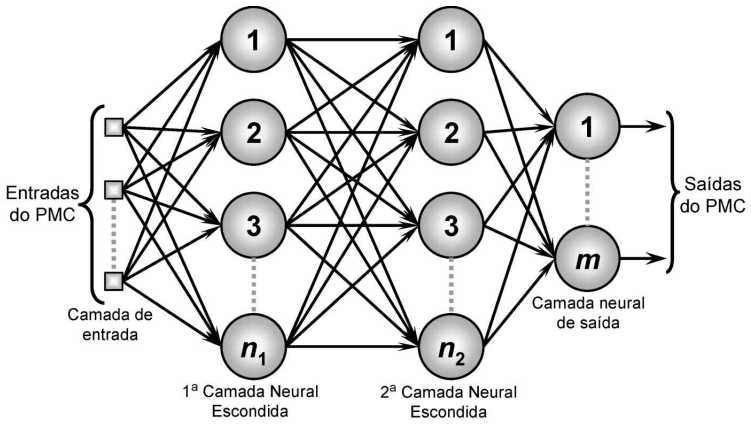


Figura 1 – Estrutura de uma Rede Neural Artificial

Fonte: https://miro.medium.com/max/1400/1*piYTTh83qsQJVUMOZKmN5w.png

2.4 Neurônio Artificial

Neurônio Artificial é uma representação matemática do neurônio biológico. O neurônio artificial é composto por uma função de ativação que caso satisfeita, dispara seu sinal para os próximos neurônios da próxima camada; caso não satisfeita, seu sinal é anulado. Esse processo é uma representação matemática da sinapse de uma estrutura neural biológica. O parâmetro de entrada para a função de ativação de um neurônio artificial é definido

como o somatório do produto do sinal do neurônio (ou entrada) da camada anterior com o respectivo peso da ligação. A Figura 2 é uma representação visual do processo sináptico de um neurônio artificial (KIRK, 2017).

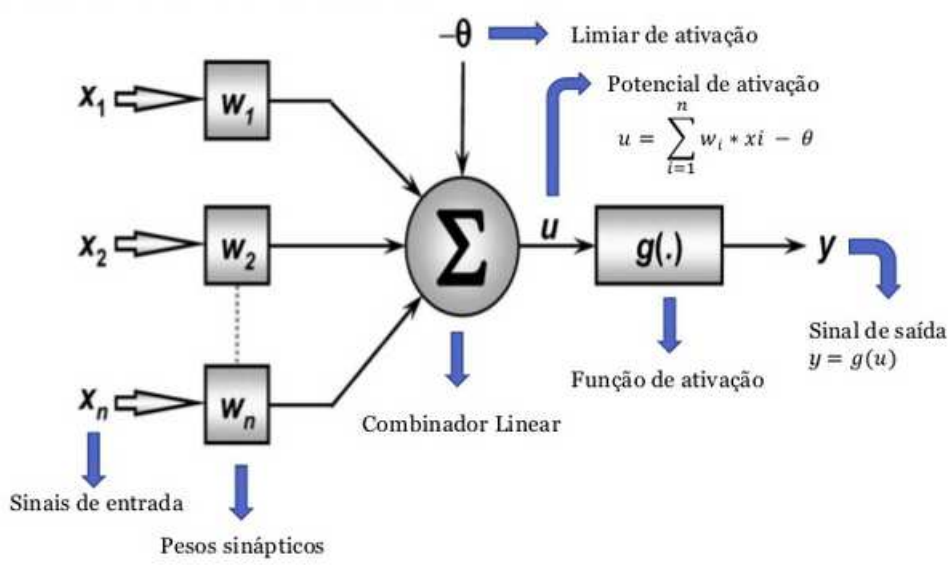


Figura 2 – Neurônio Artificial

Fonte: https://medium.com/@avinicius.adorno/redes-neurais-artificiais-418a34ea1a39

2.5 Redes Neurais Convolucionais Artificiais (CNN)

As Redes Neurais Convolucionais foram um desenvolvimento revolucionário no uso de Redes Neurais, com diversas aplicações, especialmente na área de imagens e visão computacional (LI et al., 2021). Em resumo, uma CNN tem como foco extrair traços significativos de uma imagem. Imagens são dados não estruturados que podem ou não conter informações relevantes para o processo de decisão e aprendizado de uma rede neural. Importante destacar que, em alguns casos, esses dados são muito extensos dependendo da dimensão da imagem. Por esse motivo é importante algum algoritmo que consiga separar as informações relevantes das não relevantes antes de iniciar o processo neural. Dessa forma, a rede convolucional é responsável por extrair as principais informações da imagem e, consequentemente, diminuir a quantidade de pixels que serão enviados à rede neural. Este processo reduz a possibilidade de surgimento de eventuais erros na rede neural por conta da simplificação dos dados de entrada, assim como diminui a quantidade de ligações de neurônios, colaborando também com o desempenho do processo neural.

A figura 3 apresenta a comparação de uma imagem antes e depois do processo convolucional. Após esse processo, a imagem é reduzida, mantendo apenas informações que foram julgadas relevantes no processo. Nesse processo a rede convolucional aplica filtros

na imagem para a extração de características; isso produz varias pequenas imagens no fim do processo, sendo o formato largura x altura x quantidade de filtros finais

Figura 3 – Antes e depois da convolução de uma imagem. Fonte: Autoral

((a)) Imagem (80x80).

((b)) Convolução da imagem (6x6x64)

2.6 Estrutura do Agente

O agente inteligente baseado em aprendizado por reforço tem como principais responsabilidades:

- ❏ Exploração
- ❏ Maximização dos sinais de recompensa.
- ❏ Memorização.

2.6.1 Exploração

Para que um jogador atinja uma boa pontuação ou até mesmo vitória em um jogo, ele deve construir uma combinação de ações dado um estado que ao longo do tempo culminará em uma maximização do somatório de recompensas. No entanto, há jogos que são compostos por diversas possíveis combinações de ações dado um estado, onde cada combinação resulta ao agente uma determinada pontuação final. Isso significa que, dependendo do jogo, existirá diversas combinações que levará o agente a um máximo local forçando-o a manter a mesma estratégia sempre, já que esta será a melhor já conhecida por ele. Nesse caso a exploração do ambiente força que o agente escolha ações involuntárias, o que resultará no conhecimento de novos estados e novas possibilidades de estratégias, o que permitirá que o agente escape de máximos locais e procure máximos globais.

Neste trabalho adicionamos uma taxa de exploração do ambiente. Esta taxa decaí de 100% a 7% de acordo com as mudanças de estado do ambiente.

2.6.2 Maximização dos sinais de recompensa

No aprendizado por reforço existe apenas um agente, o qual deve interagir com o ambiente onde foi inserido e conseguir aprender de acordo com as politicas estabelecidas no ambiente. Essas politicas consistem em sinais de recompensas fornecidos pelo ambiente dado uma ação do agente em um determinado estado do ambiente. Dessa forma, o fluxo de um agente baseado em aplicações puras do aprendizado por reforço consiste em executar ações A_t no ambiente em um estado S_t e receber como resposta do próprio ambiente um próximo estado S_{t+1} e uma recompensa R_t pela ação executada. A Figura (4) apresenta este fluxo visualmente.

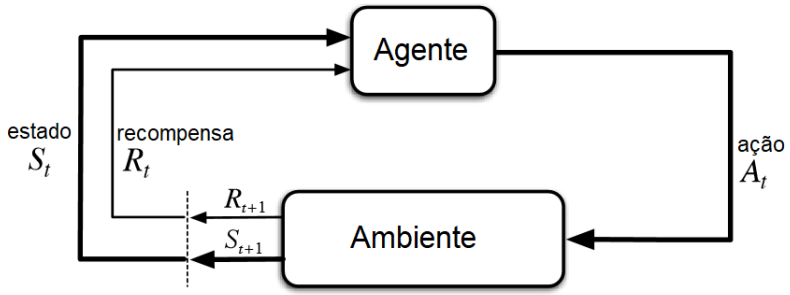


Figura 4 – Estrutura do aprendizado por reforço

Fonte:

A chave para a aplicação do aprendizado por reforço é a propagação das recompensas para estados anteriores. Cenários em que recompensas são obtidas com pouca frequência (ambientes de recompensas esparsas) necessitam de abordagens que consigam propagar as poucas recompensas para os estados anteriores. Diante de algumas abordagens sugeridas pela literatura, neste trabalho utilizamos o algoritmo Q-Learning apresentado na equação (1).

$$Q_{new}(s_t, a_t) = R_t + \gamma max(Q(s_{t+1}, a_t)) - Q(s_t, a_t) \tag{1}$$

O algoritmo Q-Learning é uma implementação poderosa da técnica de aprendizado por reforço, e também uma das mais frequentemente usadas (JANG et al., 2019; LEVINE et al., 2020). O Q-Learning tem como objetivo maximizar os sinais de recompensas futuras que serão obtidas pelo agente. Dessa forma este algoritmo é capaz de encontrar ações que, dado o estado atual, levará o agente a maximizar os sinais de recompensas obtidos.

A melhor ação para um determinado estado S_t do ambiente, independente dos demais, nem sempre é a melhor para o resultado final de um ambiente. Visto isto, o aprendizado

por reforço necessita de um algoritmo otimizador de ações, de forma que as escolhas também considerem os acontecimentos futuros. Isso significa que o otimizador de recompensas deve encontrar combinações de ações que permitam maximizar o somatório final de recompensas.

A equação (1) pontua um valor Q para cada S_t, A_t. Esse valor $Q(S_t, A_t)$ cria uma tabela de pontuações estado S_t, ação A_t, onde é definido as melhores ações dado um determinado estado. Neste trabalho por utilizarmos uma Rede Neural, o algoritmo Q-Learning utiliza essa tabela de estado, ação para treinar o modelo Neural.

2.6.3 Memorização

Assim como o processo de aprendizado animal, o agente artificial inteligente também aprende de acordo com as experiencias do passado que viveu. O processo de maximização dos sinais de recompensas depende da memorização de estados, recompensas e ações do passado, pois estes serão utilizados no aprendizado do agente. Dessa forma, o aprendizado por reforço necessita da memorização de um certo espaço temporal, para que o agente aprenda a maximizar recompensas de acordo com as próprias experiências.

Neste trabalho o agente é capaz de memorizar o estado atual S_t, estado futuro S_{t+1}, recompensa $R_t(S_t, A_t)$ e ação A_t. Com essas variáveis temporais o agente já é capaz de aprender a jogar os jogos de Atari. No capitulo de resultados será apresentado a influência da capacidade de memória do agente.

Capítulo 3

Desenvolvimento

Este trabalho propõe o desenvolvimento de um agente inteligente capaz superar desafios propostos por jogos de Atari, utilizando abordagens puras do aprendizado por reforço e como fonte de informações, o uso de imagens e sinais de recompensas, obtidos no decorrer do jogo. Dito isto apresentaremos a aplicação pura do aprendizado por reforço, não incrementando melhorias ou podas na teoria.

Este trabalho tem como base o artigo "Human-level control through deep reinforcement learning"(MNIH et al., 2015). Nele utilizaremos parâmetros já encontrados pelo autor para a realização dos experimentos e comparações.

3.1 Estrutura da Rede Neural Convolucional (CNN)

A Rede Neural Convolucional é o cérebro do agente, essa será estimulado pelo ambiente junto com o algorítimo otimizador no processo de aprendizagem. O algoritmo Q-Learning, otimizador utilizado na pesquisa, treina uma rede neural, onde fornece o valor $Q(S_t, A_t)$ como resposta para a entrada da rede neural convolucional.

Uma Rede Neural Convolucional é dividida em duas etapas que formam um fluxo de entrada e saída de dados: Rede Convolucional e Rede Neural. A figura 5 ilustra o processo de uma rede neural convolucional, desde a entrada e processamento da imagem na rede convolucional, até a tomada de decisão como saída da rede neural.

3.1.1 Rede Convolucional

Os parâmetros da Rede Convolucional utilizados na pesquisa são os mesmos utilizados pelo artigo de Mnih (MNIH et al., 2015). A Rede é composta por 3 camadas com as seguintes configurações apresentadas na Figura 6

O encolhimento da imagem na saída do processo da Rede Convolucional, apresentado na figura 5, acontece devido ao parâmetro "strides", ele provoca saltos de pixels no momento das multiplicações da matriz de convolução com a matriz imagem, ignorando

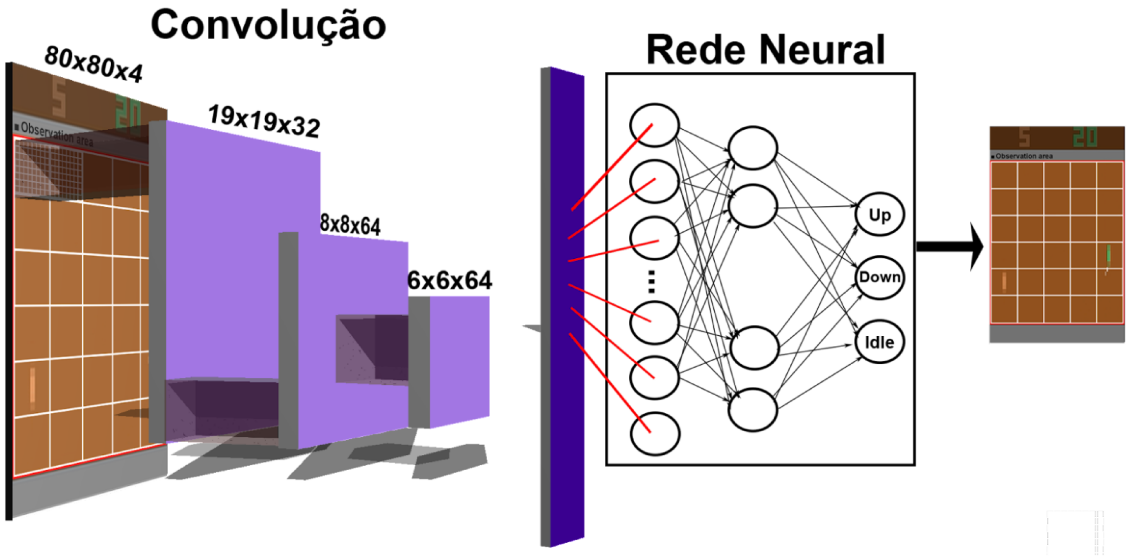


Figura 5 – Ilustração Rede Neural Convolucional

Filter	Kernel	Strides
32	(8,8)	4
64	(4,4)	2
64	(4,4)	1

Figura 6 – Parâmetros utilizados na Rede Convolucional

alguns pixels e causando uma redução na imagem resultante. No capítulo de resultados será analisada a influência das dimensões da imagem de entrada na rede convolucional no processo de aprendizado do agente.

O parâmetro "filtros"é referente à quantidade de matrizes de convolução que serão aplicadas na imagem. Esse parâmetro aplicado a uma imagem 2D, define o número de planos da terceira dimensão.

O parâmetro "Kernel Size"refere ao formato da matriz de convolução.

3.1.2 Redes Neurais

Em resumo uma Rede Neural tem como foco identificar e apreender padrões a partir de determinados dados preexistentes. Na estrutura definida neste trabalho, esses dados preexistentes referem-se aos resultados (saídas) da Rede Convolucional, os quais alimentarão (entradas) a Rede Neural. Anteriormente à entrada desses dados na Rede Neural ocorre o processo de planificação (flatten) desses dados, por meio do qual se reduz a tridi-

mensionalidade (matriz) deles para uma única dimensão (vetor). Esses dados, reduzidos a uma única dimensão, alimentarão a rede neural, a qual buscará identificar e apreender os padrões existentes neles. Esse processo possibilitará a Rede Neural responder com ações a um dado estado do ambiente com a finalidade de maximizar os sinais de recompensa.

O treinamento da Rede Neural Convolucional por ser executado a cada $mode(quadros, K) = 0$, altera os pesos com muita frequência causando instabilidade no processo de predição. Nesse cenário este trabalho, baseado no artigo de Mnih (MNIH et al., 2015), dispõe de duas redes neurais com a mesma arquitetura, porém pesos diferentes, com a finalidade de suavização no processo de aprendizado. A primeira rede neural é a responsável por escolher (prever) as ações dado os estados, já a segunda é utilizada no processo de treinamento. A primeira rede é atualizada com os pesos da segunda a cada N quadros do jogo.

Os parâmetros de topologia utilizados na Rede Neural aplicados neste trabalho foram os adotados pelo artigo de Mnih (MNIH et al., 2015) e também uma proposta com mais camadas, seguindo a tendência da literatura de usar mais camadas para redes neurais, especialmente no caso de CNN (MNIH et al., 2016; SIDDIQUE; SAKIB; SIDDIQUE, 2019; CHOLDUN et al., 2019). Esses parâmetros podem ser agrupados da seguinte forma:

- Mnih:
  - Camada de entrada: neurônios: Definido pelo tamanho da imagem resultante da Rede Convolucional (Largura * Altura * Quantidade de Filtros).
  - 1º Camada oculta: neurônios: 512, ativação: relu.
  - Camada de saída: neurônios: (Ações disponíveis pelo ambiente).
- Topologia proposta:
  - Camada de entrada: neurônios: Definido pelo tamanho da imagem resultante da Rede Convolucional (Largura * Altura * Quantidade de Filtros).
  - 1º Camada oculta: neurônios: 512, ativação: relu.
  - 2º Camada oculta: neurônios: 256, ativação: relu.
  - 3º Camada oculta: neurônios: 64, ativação: relu.
  - Camada de saída: neurônios: (Ações disponíveis pelo ambiente).

3.2 Parâmetros do agente utilizados nos experimentos

A Tabela 1 exibe os parâmetros utilizados pelo Experimento Mnih, os utilizados no artigo de Mnih (MNIH et al., 2015) e o Experimento Próprio que foi proposto nesta

Tabela 1 – Parâmetros de pré-processamentos e do agente utilizados nos experimentos. Valores " = " da coluna "*ExperimentoPróprio*" indicam parâmetros iguais aos do Mnih.

Parâmetro	Experimento Mnih	Experimento Próprio
batch-size	32	=
freq-update-nn	10000	=
frames-skip	4	1
cut-top	34	=
cut-bottom	16	=
cut-left	7	=
cut-right	7	=
memory-size	200000	=
min-learning-rate	0.0001	=
max-learning-rate	0.0001	=
epochs-interval-lr	1	=
gamma	0.99	=
exploration-rate	1	=
exploration-min	0.07	1
exploration-map	70000: (1, 0.6), 100000: (0.6, 0.07)	=
k-frames	4	=
initial-start-size	5000	=
input-shape	80x80x4	=

pesquisa, em conjunto com a topologia própria proposta apresentada na seção anterior. Em vista disso esta pesquisa propõe avaliação da influência desses novos parâmetros juntamente com a nova topologia. Assim, no capítulo de Resultados será apresentada e analisada a influência de alguns parâmetros considerados importantes.

3.2.1 Dicionário de parâmetros

Esta sub-seções descreve os parâmetros citados na Tabela 1.

- batch-size: Número de amostras que serão propagadas pela rede por episódio.
- freq-update-nn: A cada freq-update-nn de frames os pesos da primeira rede neural serão atualizados com os pesos da segunda rede neural.
- frames-skip: A cada frames-skip de frames será chamada a função de treino do modelo neural convolucional.
- cut-top: Números de tiras de pixels que serão recortadas no topo da imagem.
- cut-bottom: Números de tiras de pixels que serão recortadas na parte de baixo da imagem.
- cut-left: Números de tiras de pixels que serão recortadas no lado esquerdo da imagem.

- cut-right: Números de tiras de pixels que serão recortadas no lado direito da imagem.
- memory-size: Tamanho da memória do agente.
- min-learning-rate: Taxa de aprendizado mínimo do agente.
- max-learning-rate: Taxa de aprendizado máximo do agente.
- epochs-interval-lr: Número de épocas que a taxa de aprendizagem percorrerá o intervalo min-learning-rate e max-learning-rate.
- gamma: Variável constante.
- exploration-rate: Taxa de exploração inicial. Valores entre (1 e 0)
- exploration-min: Taxa de exploração final. Quando o parâmetro exploration-rate é menor que exploration-min, a taxa de exploração para de decair.
- exploration-map: Mapa chave/valor de decaimento do exploration-rate. Nesse mapa de chave/valor a Chave representa o limite em que a regra do Valor será exercida. Por exemplo, considere o exploration-map igual a 70000: (1, 0.6), 100000: (0.6, 0.07). Dos frames 0 a 70000, a taxa de exploração decai linearmente de 1 a 0.6. Dos frames 700001 a 100000 ela decai de 0.6 a 0.07. Seguindo essa linha, o valor de desconto do exploration-rate será descrito pela equação

$$discountExploration = (max(Valor) - min(Valor))/(Chave/kFrames) \quad (2)$$

- k-frames: Número de imagens agrupadas que serão dispostas para a Rede Neural Convolucional.
- initial-start-size: Número de frames iniciais que o processo de treino não será chamado. Inicia a memória.
- input-shape: Formato da imagem que será enviada a Rede Neural Convolucional.

3.3 Informações de entrada da Rede Neural Convolucional

Nesta pesquisa será utilizado imagens como dado de entrada para a rede neural convolucional. Essas imagens antes de alimentarem o processo neural são pré-processadas. O fluxo de pré-processamento realizado nos experimentos desta pesquisa é:

- 1º: Corte das partes irrelevantes da imagem, de forma a manter apenas a parte central da imagem.

- ❏ 2º: O redimensionamento da imagem.
- ❏ 3º: Remover as dimensões RGB, mantendo apenas a escala cinza da imagem.
- ❏ 4º: Divisão do valor dos pixels por 255, para manter os valores entre 0 e 1.

3.4 Dimensão tempo adicionada no processo neural

Os jogos de Atari que foram escolhidos nesta pesquisa são jogos que dependem da movimentação de objetos, logo o agente deve ser capaz entender esse movimento. Considerando uma única imagem como informação de entrada para o processo neural convolucional, é impossível que o agente compreenda a direção e velocidade do movimento de qualquer objeto. Isso acontece porque uma única imagem do estado, apresentará todos os objetos parados no tempo, sem nenhuma indicação de movimento e velocidade dos mesmos. Nesse cenário, o artigo (MNIH et al., 2013) traz a solução que será utilizada nesta pesquisa. A solução consiste em empilhar uma sequência k de estados dado um tempo t, ou seja, a imagem que alimentará a rede neural convolucional será composta por uma sequência k de estados anteriores ao tempo t do estado da imagem.

Por exemplo, definido k igual a 4 e largura e altura igual a 80, a imagem enviada para o processo neural convolucional terá o formato de (80x80x4), sendo 4 imagens sequênciais no tempo empilhadas $(S_{t-3}, S_{t-2}, S_{t-1}, S_t)$.

Capítulo 4

Experimentos e Análise dos Resultados

A Análise dos resultados obtidos através dos experimentos no jogo Pong de Atari será apresentada neste Capítulo. Também será avaliado, de maneira simplificada, o desempenho do agente no jogo Boxing e Breakout como experimentos bônus. O objetivo dos experimentos bônus é apenas avaliar a capacidade do mesmo algoritmo e modelo em se adaptar em jogos com ambientes e espaço de decisão diferentes. Portanto, a análise de resultado para estes dois jogos focará apenas nesse aspecto, enquanto a análise de resultados para o jogo Pong (jogo foco da pesquisa), analisará a influência de parâmetros e topologia da Rede Neural Convolucional. Esses experimentos foram realizados com o framework *gym* da OpenIA no ambiente Atari-2600, em específico com os jogos Pong, Boxing e Breakout. A Figura 7 mostra capturas de tela de cada um dos jogos citados.

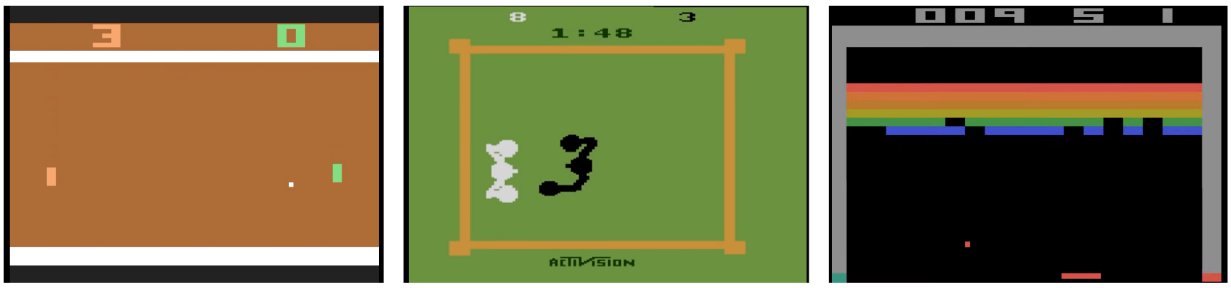


Figura 7 – Jogos de Atari 2600 utilizados nos experimentos. Da esquerda para a direita, os jogos são Pong, Boxing e Breakout.

Os experimentos além de apresentarem comparações entre os modelos (base) propostos por Mnih (Experimento Mnih) e o proposto nesta pesquisa (Experimento Próprio), citados na Tabela 1, também exibem a influencia de determinados parâmetros em relação a cada modelo base, referenciados na Tabela 1.

4.1 Jogo Pong

O Jogo Pong consiste em duas hastes que têm como objetivo impedir que a bola de passe do limite do jogador. Quando um dos dois jogadores não consegue acertar a bola e ela está na direção do jogador, o oponente deste jogador recebe 1 ponto. Uma partida de Pong termina quando um dos jogadores atinge o limite de 20 pontos. A regra estabelecida para contabilizar a recompensa final do agente é a pontuação final do agente menos a pontuação final do oponente. Essa recompensa final pode variar de -20 a 20 pontos.

4.1.1 Definições de avaliação

Nas sub-seções abaixo serão analisados os resultados obtidos nos experimentos no jogo Pong. Estes experimentos terão como base os modelos de parâmetros definidos na Tabela 1, por meio dos quais se avaliará o aprendizado do agente a partir de um aspecto geral e a partir de variações nos parâmetros destacados na Tabela 1.

Para a melhor compreensão dos gráficos e das informações apresentadas nas sub-seções abaixo, cada experimento será definido com o seguinte nome e definição:

Todos os experimentos são baseados nos Experimento Mnih e Experimento Próprio citados na 1

- Mnih-control-(80x80): *Experimento Mnih*.
- Mnih-memory25k-(80x80): Experimento baseado no *Experimento Mnih*, com alteração no parâmetros:
  - memory-size: 25000.
- own-control-(80x80): *Experimento Próprio*.
- own2-variation-lr-(80x80): Experimento baseado no *Experimento Próprio*, com variação nos parâmetros:
  - min-learning-rate: 0.00005
  - epochs-interval-lr: 50
- own3-reduced-learning-(80x80): Experimento baseado no *Experimento Próprio*, com variação nos parâmetros:
  - min-learning-rate: 0.00008
  - max-learning-rate: 0.00008
- own4-(84x84): Experimento baseado no *Experimento Próprio*, com variação nos parâmetros:

  - input-shape: 84x84x4

- own5-memory25k-(84x84): Experimento baseado no *Experimento Próprio*, com variação nos parâmetros:
  - input-shape: 84x84x4
  - memory-size : 25000

4.1.2 Tempo de processamento por experimento

Devido a demora na realização de experimentos e prazo da pesquisa, foi necessário a utilização de duas máquinas distintas para o processamento dos experimentos citados no item anterior. Essas máquinas foram:

- UFU: Máquina com 16GB de RAM e um processador de 12 núcleos.
- AWS: Máquina com 32GB de RAM e um processador de 8 núcleos.

Tabela 2 – Tempo de processamento dos experimentos

Experimento	Tempo de processamento (horas)	Máquina
Mnih-control-(80x80)	10.36	UFU
Mnih-memory25k-(80x80)	7.66	UFU
own-control-(80x80)	9.08	AWS
own2-variation-lr-(80x80)	9.3	AWS
own3-reduced-learning-(80x80	8.74	AWS
own-4(84x84)	10.99	AWS
own5-memory25k-(84x84)	8.54	UFU

4.1.3 Análise de aprendizagem do agente

A Figura 8 apresenta todos os experimentos posicionados em uma mesma escala do eixo X e Y. O eixo X representa a quantidade de treinos e o eixo Y representa a média das 50 recompensas mais recentes. Esse gráfico exibe a efetividade de cada modelo no jogo Pong por quantidade de treinos. Cada experimento apresentado na Figura 8 explora um aspecto diferente dos parâmetros ou da topologia da Rede Neural. Esses aspectos serão explorados nas sub-seções seguintes.

4.1.4 Análise de tempo de processamento dos modelos

Na Figura 9(a) o experimento own-4(84x84) obteve o melhor desempenho ao logo dos treinos. No entanto na Figura 9(b), analisando a relação entre o tempo em segundos e a média de recompensas, constata-se que o experimento own2-variation-lr-(80x80)

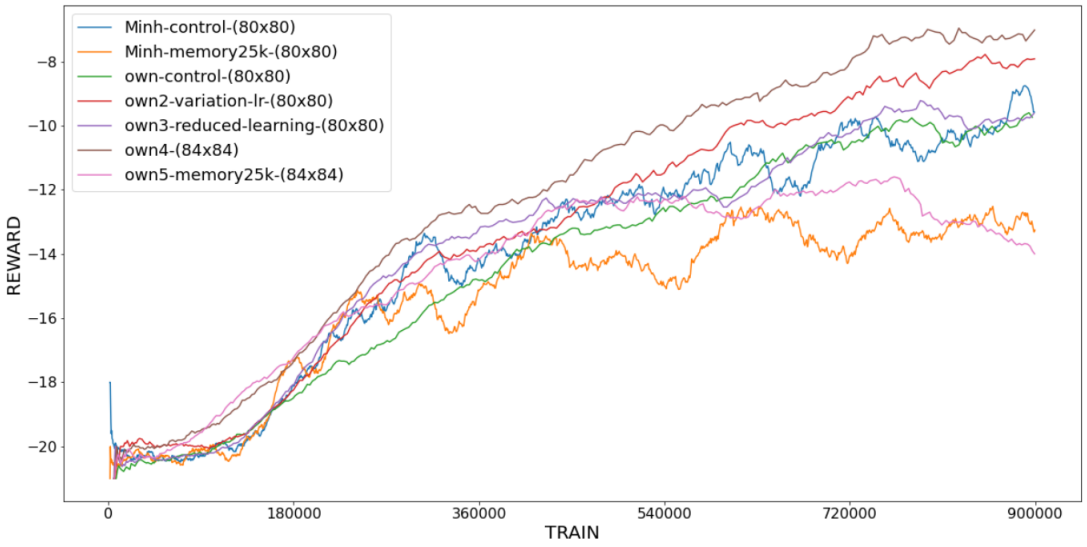
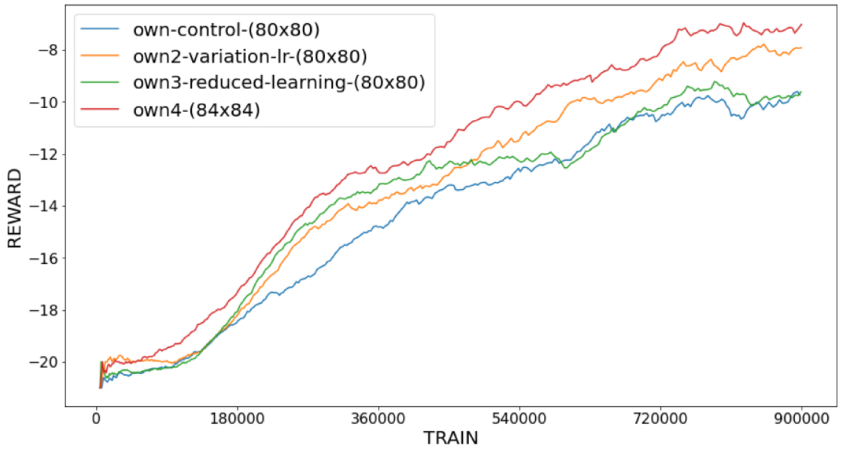
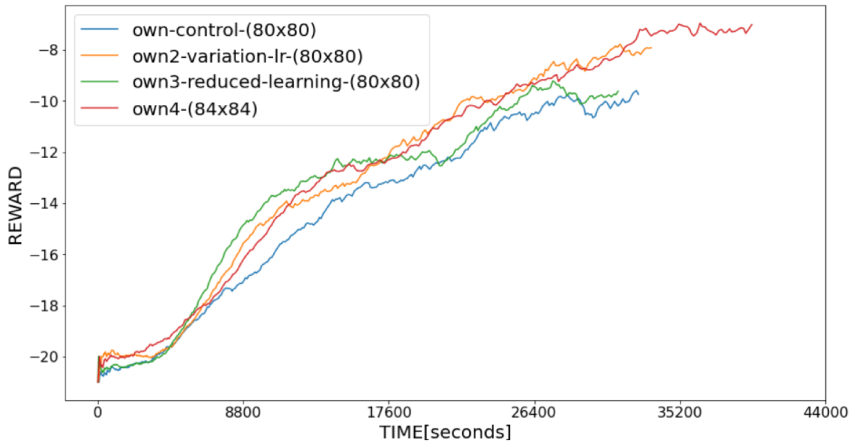


Figura 8 – Evolução no aprendizado em relação à quantidade de treinos em todos os experimentos no jogo Pong



((a)) Relação de Treinos por Média de Recompensas



((b)) Relação de Tempo (segundos) por Média de Recompensas.

Figura 9 – Comparação dos experimentos que foram realizados na máquina AWS, que estão evidenciados na Tabela 2.

tem um desempenho muito próximo ao experimento own-4(84x84). Nesse sentido e considerando a Tabela 2, o experimento own2-variation-lr-(80x80) consumiu 1,69 horas a menos que o experimento own-4(84x84). Dessa forma o experimento own4-(84x84) teve o melhor desempenho na relação treinos e maximização de recompensas, porém o experimento own4-(84x84) necessita de maior poder computacional do que o experimento own2-variation-lr-(80x80). Nesse sentido o experimento own2-variation-lr(80x80) tem um bom resultado se comparado ao own4-(84x84), que na Figura 9(a) é o melhor, e consome menos tempo de processamento.

4.1.5 Análise de parâmetros

Nesta sub-seção serão avaliadas as influências de parâmetros no jogo Pong, considerando como modelos base aqueles propostos na Tabela 1, *Experimento Mnih* e *Experimento Próprio*.

4.1.5.1 Avaliação de influência do tamanho da memória no modelo Experimento Mnih

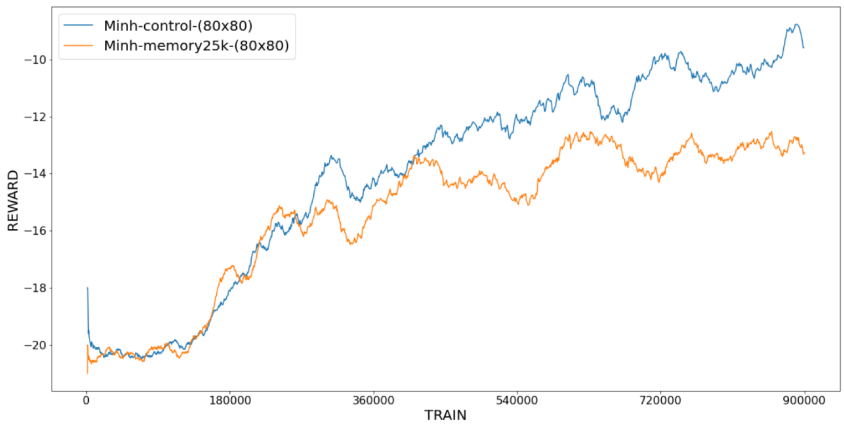
O parâmetro memory-size informa o tamanho da memória do agente, ou melhor, a quantidade de eventos históricos que ele terá armazenado na memória. Esses eventos adicionados na memória são utilizados nos treinos. Nesse sentido, quanto mais memória, mais estados distintos e até raros existirão na memória.

Na Figura 10(a) o experimento Mnih-control-(80x80) se sobressai, em relação ao aprendizado ao longo dos treinos, quando comparado ao experimento Mnih-memory25k-(80x80) em relação ao aprendizado no longo dos treinos. Na Figura 10(b), o experimento Mnih-control-(80x80) se sobressai, evidenciando uma tendência maior de evolução da maximização das recompensas, quando comparado ao experimento Mnih-memory25k-(80x80).

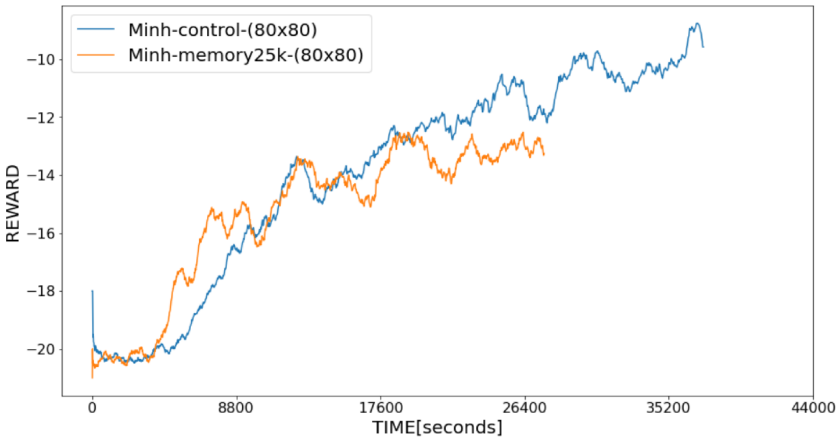
4.1.5.2 Avaliação de influência do formato da imagem que alimenta a Rede Neural Convolucional.

A Figura 11 compara o formato da imagem que alimenta o processo da Rede Neural. O experimento own-control-(80x80) preprocessa a imagem (através de corte e escala, como citado no capítulo anterior) para o formato (80x80) (largura, altura) de pixels, já o experimento own4-(84x84) preprocessa a imagem para o formato (84x84) de pixels. Essa diferença de 4 pixels de largura e altura resulta em uma saída de (7x7x64) da rede convolucional. Enquanto uma imagem (80x80) resulta em uma saída de (6x6x64) da rede convolucional, o aumento desses 4 pixels de altura e largura apresentam melhoras no aprendizado do agente. Os dados apresentados na Figura 11, abrem a hipótese de que uma matriz resultado de uma rede convolucional ligeiramente maior, consegue expressar mais detalhes de uma imagem. Em contrapartida esse ligeiro aumento gera mais

custo computacional para o processamento da Rede Neural. Ao realizar o processo de planificação de uma matriz (6x6x64), obtem-se 2.304 neurônios de entrada, já realizando esse mesmo processo em uma matriz de (7x7x64), obtem-se 3.136 neurônios de entrada. Esse aumento de neurônios de entrada gera um custo alto de processamento para a Rede Neural. Nesse caso, os dois resultados tem suas melhores peculiaridades, um atinge melhores resultados em menos treinos, porém exige um custo computacional, enquanto o outro tem um desempenho um pouco menor, mas em compensação exige menos custo computacional.

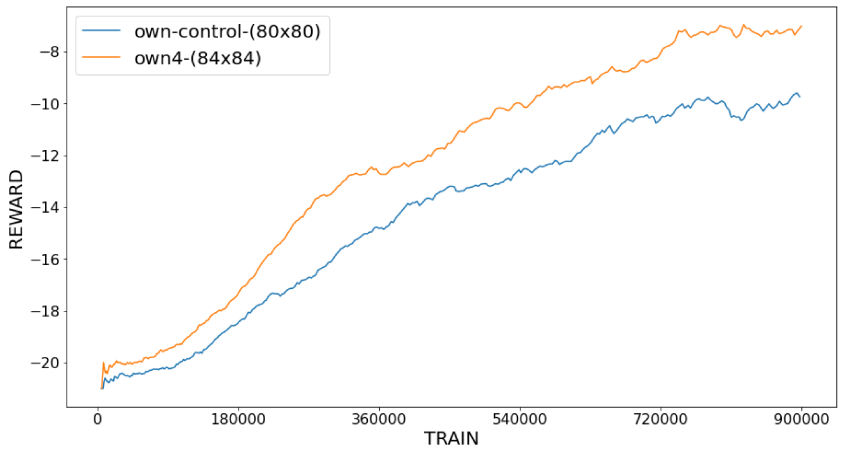


((a)) Relacionando Treinos com Recompensas Média.

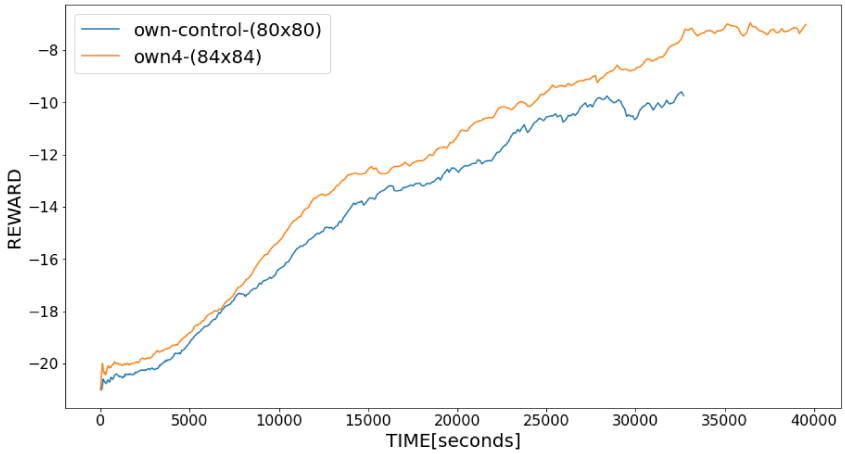


((b)) Relacionando Tempo (segundos) com Recompensas Média.

Figura 10 – Comparação dos experimentos Mnih-control-(80x80) e Mnih-memory-25k-(80x80) relacionando Treinos e Tempo com média de recompensas. O experimento Mnih-memory-25k-(80x80) é baseado no experimento Mnih-control-(80x80) citado na Tabela 1, com a única diferença no parâmetro memory-size para 25000



((a)) Relacionando Treinos com Recompensas Média.



((b)) Relacionando Tempo (segundos) com Recompensas Média.

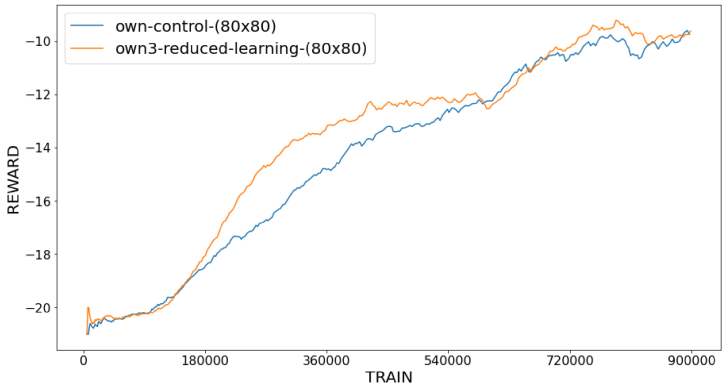
Figura 11 – Comparação dos experimentos own-control-(80x80) e own4-control-(84x84) relacionando Treinos e Tempo com média de recompensas. O experimento own4-control-(84x84) é baseado no experimento own-control-(80x80) citado na Tabela 1, com a única diferença no parâmetro input-shape para (84x84)

4.1.5.3 Avaliação de influência da taxa de aprendizado

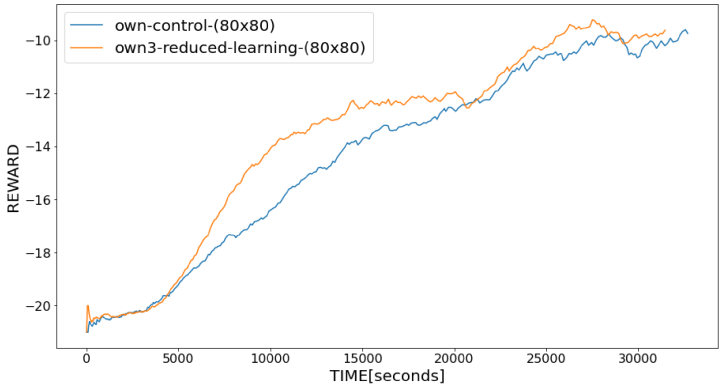
Os experimentos da Figura 12 não apresentam diferenças significativas nos resultados. A alteração na taxa de aprendizado alterou pouco no desempenho do agente.

4.1.5.4 Avaliação de influência da taxa de aprendizado variando

Na Figura 13 o experimento own2-variation-lr-(80x80) apresenta uma tendência de crescimento maior do que a apresentada pelo own-control-(80x80). Nesse sentido, a aplicação de um decaimento na taxa de exploração ao longo dos episódios aparenta colaborar no aprendizado do agente. Essa técnica de diminuir a taxa de exploração ao longo dos episódios, força que a rede neural chegue próximo a uma convergência inicialmente mais rápido e de acordo com o passar dos episódios diminuindo a taxa de aprendizado a rede neural vai convergindo os pesos com mais sutileza.

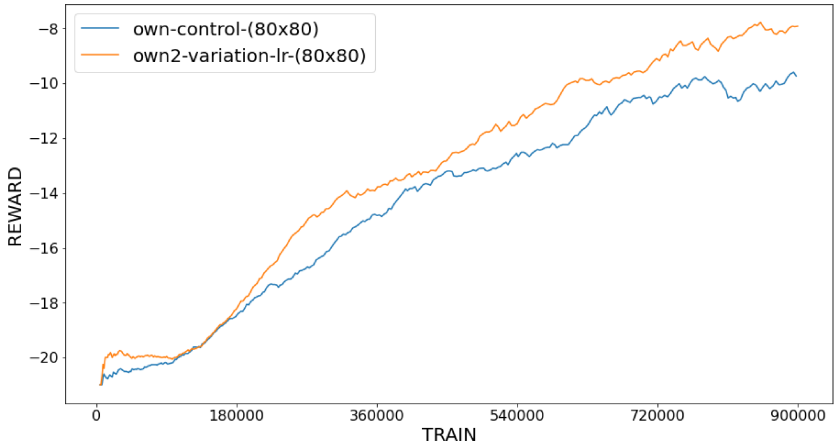


((a)) Relacionando Treinos com Recompensas Média.

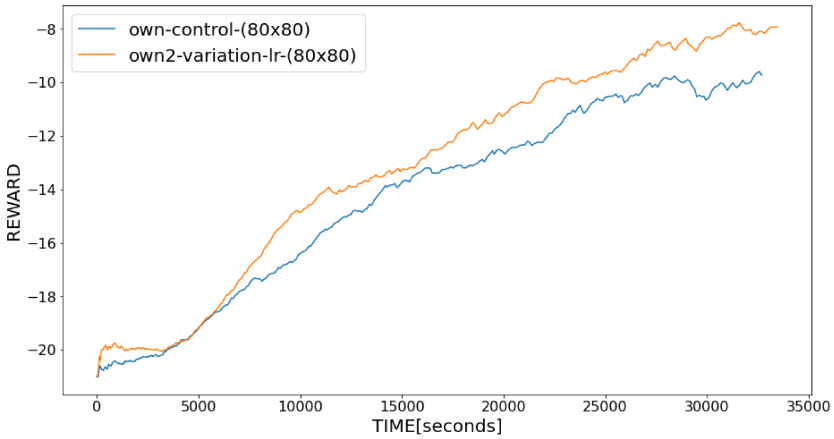


((b)) Relacionando Tempo (segundos) com Recompensas Média.

Figura 12 – Comparação dos experimentos own-control-(80x80) e own3-reduced-learning-(80x80) relacionando Treinos e Tempo com média de recompensas. O experimento own3-reduced-learning-(80x80) é baseado no experimento own-control-(80x80) citado na Tabela 1, com a diferença nos parâmetros max-learning-rate e min-learning-rate, ambos para 0.00008



((a)) Relacionando Treinos com Recompensas Média.



((b)) Relacionando Tempo (segundos) com Recompensas Média.

Figura 13 – Comparação dos experimentos own-control-(80x80) e own2-variation-lr-(80x80) relacionando Treinos e Tempo com média de recompensas. O experimento own2-variation-lr-(80x80)-(80x80) é baseado no experimento own-control-(80x80) citado na Tabela 1, com a diferença nos parâmetros max-learning-rate para 0.00005 e epochs-interval-lr para 50

4.2 Jogo Boxing

O jogo Boxing consiste em dois bonecos cada um compostos por um circulo central, que representa a cabeça do boneco e dois braços um a direita e a esquerda desse circulo central. Os braços do boneco são ativados individualmente. Quando o jogador aciona a ação de golpear com o braço esquerdo, o braço esquerdo se estica na horizontal, a assim também para o contrário. Quando um dos jogadores acerta um golpe no circulo central (cabeça) de seu oponente, esse jogador recebe uma pontuação. O jogo finaliza quando um dos jogadores atingem 100 pontos ou quando se passa 2 minutos. A pontuação final de um jogador é a diferença entre a pontuação desse jogador com a de seu oponente. Essa recompensa final pode variar de -100 a 100 pontos.

Nesta seção de análise dos experimentos realizados no jogo Boxing, foi realizado apenas a comparação das topologias de rede neural convolucional.

4.2.0.1 Análise de aprendizagem do agente

Este experimento terá como base os modelos de parâmetros definidos na Tabela 1 e possibilitarão a avaliação do aprendizado do agente no geral.

Para o melhor entendimento do gráfico e informações apresentados a seguir, cada experimento será definido com o seguinte nome e definição:

Os experimentos são baseados nos Experimento Mnih e Experimento Próprio citados na Tabela

De acordo com a Figura 14 o desempenho do experimento Minih-control-(80x80) é superior ao experimento Own-control-(80x80).

1.

- ❏ Mnih-control-(80x80): *Experimento Mnih.*
- ❏ Own-control-(80x80): *Experimento Próprio.*

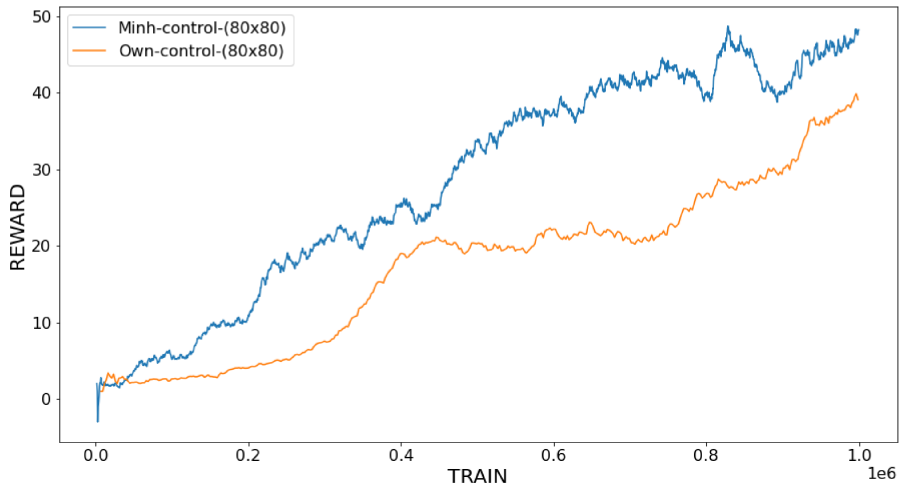


Figura 14 – Comparação dos experimentos Mnih-control-(80x80) e Own-control-(80x80), citados na Tabela 1, relacionando Treinos e Tempo com média de recompensas.

4.3 Jogo Breakout

O jogo Breakout consiste em uma haste que fica movimentando horizontalmente na parte inferior do cenário e um dos objetivo é não deixar a bola do jogo ultrapassar o limite da haste. O jogador também tem o objetivo de rebater essa bola para a parte superior do cenário. Essa parte superior do cenário é composto por blocos quebráveis. A cada

vez que o agente rebate a bola para essa parte superior e quebra algum bloco, ele recebe pontuação positiva.

4.3.1 Análise de aprendizagem do agente

Nas sub-seções abaixo serão analisados os resultados obtidos nos experimentos do jogo Breakout. Estes experimentos terão como base os modelos de parâmetros definidos na Tabela 1 e possibilitarão a avaliação do aprendizado do agente no geral.

Para o melhor entendimento dos gráficos e informações apresentados nas sub-seções abaixo, cada experimento será definido com o seguinte nome e definição:

Todos os experimentos são baseados nos Experimento Mnih e Experimento Próprio citados na 1

- Own-control-(80x80): Experimento controle.

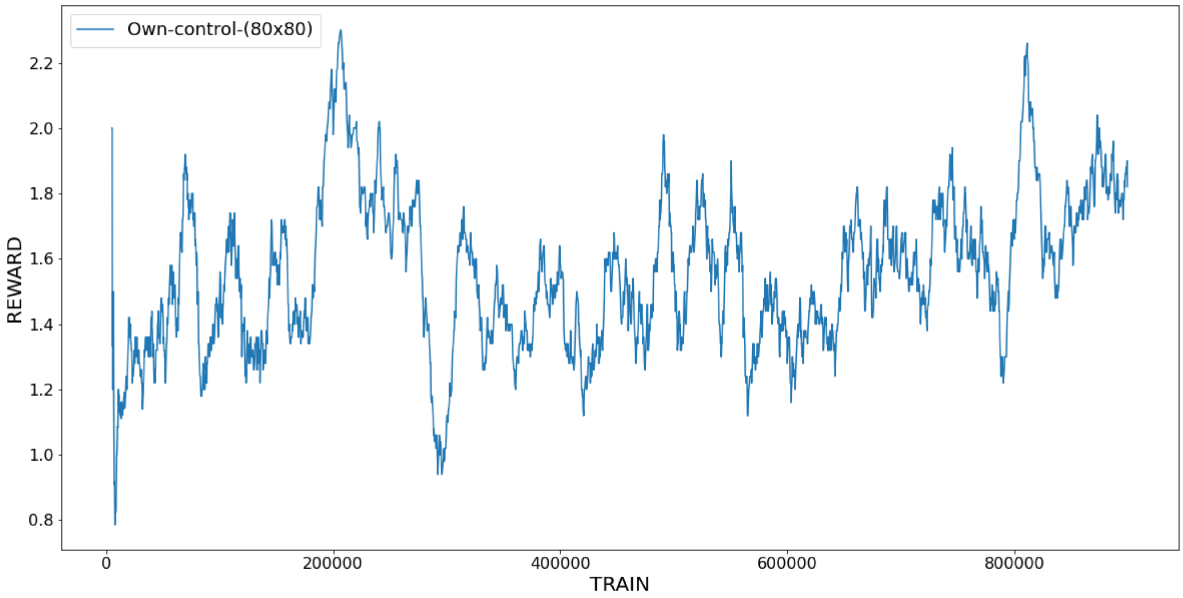


Figura 15 – Evolução do aprendizado no experimento Own-control-(80x80) relacionando Treinos com Média de Recompensas.

A Figura 15 apresenta a evolução do agente no experimento Own-control-(80x80) relacionando Treinos com Média de Recompensas. De acordo com os dados apresentados na Figura 15, os parâmetros e abordagem utilizados no experimento não obtiveram sucesso. A hipótese do não sucesso no experimento é o parâmetro memory-size. No experimento Own-control-(80x80) foi utilizado o tamanho de 200000 de memory-size, capacidade em armazenar os frames mais recentes, já no artigo de Mnih (MNIH et al., 2013) é citado o parâmetro memory-size como um milhão de capacidade em armazenar frames mais recentes. O fato de não realizarmos o experimento com o parâmetro memory-size de um

milhão foi a limitação computacional, para esse experimento seria necessário uma máquina com aproximadamente 110GB de memória RAM (com menos memória é possível realizar o projeto com a paginação dos dados, porém essa alternativa deixaria o processo super lento). A análise dessa proposta fica aberta para próximos trabalhos.

Capítulo 5

Conclusão

O desenvolvimento de um agente baseado no aprendizado por reforço com uma topologia de rede neural convolucional e parâmetros que o permita maximizar as recompensas obtidas em mais de um jogo de Atari, não é uma tarefa fácil, em razão da complexidade e obscuridade na definição de uma topologia de rede neural convolucional, e a combinação dessa mesma rede neural convolucional com os parâmetros escolhidos. Desta forma, o problema abordado foi relevante.

Este trabalho utilizou a informação visual (imagens) do jogo de Atari, logo o pré-processamento dessas imagens que alimentarão o processo neural convolucional é tão importante quanto o desenvolvimento da própria rede neural convolucional, e permite que a dimensão da entrada seja reduzida e os dados de entradas simplificados, o que é essencial para lidar com limitações de hardware e tempo de processamento.

As imagens resultantes do pré-processamento alimentaram a rede convolucional para o treinamento ou predição de novas ações. O processo de treino é realizado por algumas épocas até que o agente consiga explorar o ambiente e espaço de decisão do problema. Ao longo dos treinos o agente vai encontrando ações que o evoluem no processo de maximização das recompensas. Esse processo foi realizado para os jogos Pong, Boxing e Breakout e considerando as mesmas topologias rede neural convolucional e parâmetros combinados obtiveram resultados positivos do aprendizado do agente, especialmente no caso dos jogos Pong e Boxing (que são visualmente mais simples que o Breakout) é possível dizer que a topologia proposta obteve desempenho comparável com o da literatura.

Os resultados obtidos a partir de experimentos revelam que o agente foi capaz de evoluir no aprendizado de maximização de recompensas nos jogos Pong e Boxing. Esses resultados também esclareceram a influência de certos parâmetros e topologia de rede neural combinados no agente. Pode-se dizer que a redução do parâmetro de memória no jogo Pong não foi muito prejudicial ao desempenho (ao mesmo tempo que ofereceu ganhos altos em tempo de processamento), e a mudança da escala das imagens de 80x80 para 84x84 obteve bons resultados, apesar do tempo de processamento maior.

5.1 Trabalhos Futuros

Propõe-se a realização de mais experimentos que avaliem a influência de outros parâmetros do aprendizado por reforço, e avaliação de outras topologias de redes neurais convolucionais. Em particular, seria interessante rodar a análise de parâmetros realziada para o Pong nos jogos Boxing e Breakout. Outra possível continuação para este trabalho de pesquisa é a aplicação do aprendizado por forço em outros jogos de Atari ou em outras plataformas jogáveis.

O enfoque deste trabalho foi em recursos de hardware e tempo de processamento limitados, mas uma continuação importante deste trabalho é a realização de experimentos mais longos, para analisar a convergência dos valores das redes neurais em um prazo mais longo de treinamento. Especialmente cabe ressaltar os experimentos com memória menor, que apesar de uma melhora mais lenta no desempenho, foram significativamente mais rápidos que o caso de memória maior, podendo potencialmente ser rodados por mais iterações no mesmo espaço de tempo.

Uma das evoluções no aprendizado por reforço aplicado a visão computacional que visa reduzir a dimensão das entradas é o uso de detecção de objetos na imagem original, antes do processamento pela rede neural. Isso pode ser realizado com um algoritmo do tipo "blob finder", para extrair, por exemplo, as hastes e a bolinha do Pong. Um exemplo de aplicação disso no aprendizado para agentes jogadores de videogame pode ser encontrado em Kulkarni *et al.* (KULKARNI et al., 2019), e é algo que pode ser incorporado a pesquisas derivadas deste trabalho.

Referências

BROCKMAN, G. et al. Openai gym. **arXiv preprint arXiv:1606.01540**, 2016. Citado na página 14.

CHOLDUN, I. et al. Determining the number of hidden layers in neural network by using principal component analysis. In: SPRINGER. **Proceedings of SAI Intelligent Systems Conference**. [S.l.], 2019. p. 490–500. Citado na página 26.

DUTTA, S. **Reinforcement Learning with TensorFlow: A beginner's guide to designing self-learning systems with TensorFlow and OpenAI Gym**. [S.l.]: Packt Publishing Ltd, 2018. Citado 2 vezes nas páginas 12 e 16.

HAYKIN, S.; NETWORK, N. A comprehensive foundation. **Neural networks**, v. 2, n. 2004, p. 41, 2004. Citado na página 18.

JANG, B. et al. Q-learning algorithms: A comprehensive classification and applications. **IEEE access**, IEEE, v. 7, p. 133653–133667, 2019. Citado na página 22.

JUSTESEN, N. et al. Deep learning for video game playing. **IEEE Transactions on Games**, IEEE, v. 12, n. 1, p. 1–20, 2019. Citado na página 13.

KAELBLING, L. P.; LITTMAN, M. L.; MOORE, A. W. Reinforcement learning: A survey. **Journal of artificial intelligence research**, v. 4, p. 237–285, 1996. Citado 2 vezes nas páginas 16 e 18.

KEMPKA, M. et al. Vizdoom: A doom-based ai research platform for visual reinforcement learning. In: IEEE. **2016 IEEE Conference on Computational Intelligence and Games (CIG)**. [S.l.], 2016. p. 1–8. Citado na página 13.

KIRK, M. **Thoughtful machine learning with Python: A test-driven approach**. [S.l.]: "O'Reilly Media, Inc.", 2017. Citado 2 vezes nas páginas 19 e 20.

KULKARNI, T. D. et al. Unsupervised learning of object keypoints for perception and control. **Advances in neural information processing systems**, v. 32, 2019. Citado na página 43.

LAMPLE, G.; CHAPLOT, D. S. Playing fps games with deep reinforcement learning. In: **Thirty-First AAAI Conference on Artificial Intelligence**. [S.l.: s.n.], 2017. Citado na página 13.

LEVINE, S. et al. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. **arXiv preprint arXiv:2005.01643**, 2020. Citado na página 22.

LI, Z. et al. A survey of convolutional neural networks: analysis, applications, and prospects. **IEEE transactions on neural networks and learning systems**, IEEE, 2021. Citado na página 20.

MACHADO, M. C. et al. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. **Journal of Artificial Intelligence Research**, v. 61, p. 523–562, 2018. Citado na página 13.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. **The bulletin of mathematical biophysics**, Springer, v. 5, n. 4, p. 115–133, 1943. Citado na página 18.

MNIH, V. et al. Asynchronous methods for deep reinforcement learning. In: PMLR. **International conference on machine learning**. [S.l.], 2016. p. 1928–1937. Citado na página 26.

_____. Playing atari with deep reinforcement learning. **arXiv preprint arXiv:1312.5602**, 2013. Citado 2 vezes nas páginas 29 e 40.

_____. Human-level control through deep reinforcement learning. **nature**, Nature Publishing Group, v. 518, n. 7540, p. 529–533, 2015. Citado 5 vezes nas páginas 13, 16, 17, 24 e 26.

SAMUEL, A. L. Some studies in machine learning using the game of checkers. **IBM Journal of research and development**, IBM, v. 3, n. 3, p. 210–229, 1959. Citado na página 17.

SHELHAMER, E. et al. Loss is its own reward: Self-supervision for reinforcement learning. **arXiv preprint arXiv:1612.07307**, 2016. Citado na página 18.

SIDDIQUE, F.; SAKIB, S.; SIDDIQUE, M. A. B. Recognition of handwritten digit using convolutional neural network in python with tensorflow and comparison of performance for various hidden layers. In: IEEE. **2019 5th International Conference on Advances in Electrical Engineering (ICAEE)**. [S.l.], 2019. p. 541–546. Citado na página 26.

SILVER, D. et al. Mastering the game of go with deep neural networks and tree search. **nature**, Nature Publishing Group, v. 529, n. 7587, p. 484–489, 2016. Citado 2 vezes nas páginas 12 e 17.

_____. Mastering the game of go without human knowledge. **nature**, Nature Publishing Group, v. 550, n. 7676, p. 354–359, 2017. Citado 3 vezes nas páginas 12, 17 e 18.

SUTTON, R. S.; BARTO, A. G. **Reinforcement learning: An introduction**. [S.l.]: MIT press, 2018. Citado na página 16.