UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
INSTITUTO DE COMPUTAÇÃO
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

MATHEUS CUNHA SIMÕES
DANIEL JIMENEZ SEPULVEDA

UM ESTUDO DO APRENDIZADO POR REFORÇO COM ENFOQUE NO ENSINO DE XADREZ PARA INICIANTES

RIO DE JANEIRO
2021

MATHEUS CUNHA SIMÕES
DANIEL JIMENEZ SEPULVEDA

UM ESTUDO DO APRENDIZADO POR REFORÇO COM ENFOQUE NO ENSINO DE XADREZ PARA INICIANTES

Trabalho de conclusão de curso de graduação apresentado ao Instituto de Computação da Universidade Federal do Rio de Janeiro como parte dos requisitos para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. João Carlos Pereira da Silva

RIO DE JANEIRO
2021

Simões, Matheus Cunha
S357e Um estudo do aprendizado por esforço com enfoque no ensino de xadrez para iniciantes / Matheus Cunha Simões, Daniel Jimenez Sepulveda. – Rio de Janeiro, 2021.

55 f.

Orientador: João Carlos Pereira da Silva.

Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) - Universidade Federal do Rio de Janeiro, Instituto de Computação, Bacharel em Ciência da Computação, 2021.

1. Aprendizado por reforço. 2. Aprendizado de máquina. 3. Jogos. 4. Xadrez. I. Sepulveda, Daniel Jimenez. II. Silva, João Carlos Pereira da (Orient.). III. Universidade Federal do Rio de Janeiro, Instituto de Computação. IV. Título.

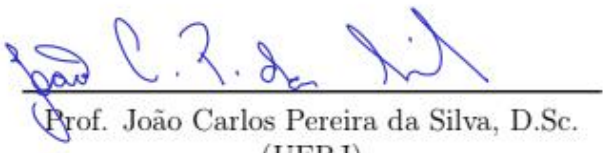
MATHEUS CUNHA SIMÕES
DANIEL JIMENEZ SEPULVEDA

UM ESTUDO DO APRENDIZADO POR REFORÇO COM ENFOQUE NO ENSINO DE XADREZ PARA INICIANTES

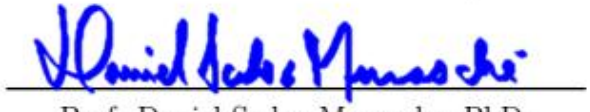
Trabalho de conclusão de curso de graduação apresentado ao Instituto de Computação da Universidade Federal do Rio de Janeiro como parte dos requisitos para obtenção do grau de Bacharel em Ciência da Computação.

Aprovado em 22 de novembro de 2021

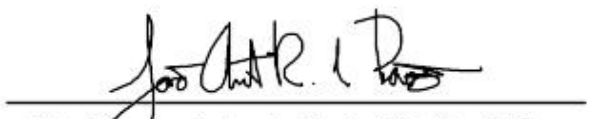
BANCA EXAMINADORA:



Prof. João Carlos Pereira da Silva, D.Sc.
(UFRJ)



Prof. Daniel Sadoc Menasche, PhD.
(UMASS)



Prof. João Antonio Recio Paixão, D.Sc.
(PUC-RJ)

AGRADECIMENTOS

Agradecemos ao nosso orientador João Carlos Pereira da Silva por aceitar conduzir este trabalho de pesquisa, pelo incentivo, paciência e atenção conosco.

Aos nosso professores do curso de Ciência da Computação da Universidade Federal do Rio Janeiro pela excelência da qualidade técnica.

A nossa família que sempre esteve ao nosso lado nos apoiando ao longo de toda essa trajetória. Sendo um agradecimento especial a Luís Fernando Jímenez, que contribuiu para o desenvolvimento desta pesquisa, com seu conhecimento técnico de xadrez e suas respectivas turmas infantis pelos testes produzidos.

Por fim, a todos que participaram, direta ou indiretamente do processo de construção deste trabalho, enriquecendo nosso processo de aprendizado.

RESUMO

O aprendizado por reforço permite que seja realizado o treinamento de agentes autônomos que determinam quais são as melhores ações dado o ambiente em que estão inseridos. A utilização dos agentes autônomos em jogos está cada vez mais comum e incentiva a busca pelo entendimento das decisões tomadas pelo software e a descoberta de novas técnicas de treinamento. Este trabalho tem como objetivo aplicar o aprendizado por reforço a partir de configurações específicas do xadrez, avaliar a performance dos agentes obtidos dado as particularidades de cada configuração e disponibilizá-los para partidas contra pessoas com diferentes níveis de conhecimento do jogo. Nos experimentos, os agentes exploram o espaço de estados do jogo e recebem recompensas por movimentos que resultem em vitória de modo que sejam identificadas as jogadas boas. São aplicadas três configurações de dificuldade variada e que são utilizadas em aulas de xadrez para demonstrar conceitos específicos a jogadores iniciantes. Além disso, para melhor performance e avaliação, utilizamos técnicas e ferramentas para representação do tabuleiro e visualização das jogadas computacionalmente. Por fim, são apresentados os resultados dos experimentos, as limitações e desafios encontrados no treinamento e possibilidades de trabalhos futuros para obter melhor desempenho.

Palavras-chave: aprendizado por reforço; aprendizado de máquina; jogos; xadrez.

ABSTRACT

Reinforcement learning allows for the training of autonomous agents who determine what are the best actions given the environment in which they are inserted. The use of autonomous agents in games is increasingly common and encourages the search for understanding the decisions made by the software and the discovery of new training techniques. This work aims to apply reinforcement learning from specific chess configurations, evaluate the performance of the agents obtained given the particularities of each configuration and make them available for matches against people with different levels of knowledge of the game. In experiments, agents explore the game state space and receive rewards for moves that result in victory so that good moves are identified. Three different difficulty settings are applied and are used in chess lessons to demonstrate specific concepts to beginning players. In addition, for better performance and evaluation, we use techniques and tools for representation of the board and visualization of moves computationally. Finally, the results of the experiments, the limitations and challenges found in training and possibilities for future work to obtain better performance are presented.

Keywords: learning by reinforcement; machine learning; games; chess.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Vemos cada vez mais a importância e o uso de inteligência artificial aplicado a diferentes cenários, desde carros autônomos a sistemas de recomendação em serviços de streaming. Exitem diversos casos em que o aprendizado de máquina é utilizado em jogos como o famoso Deep Blue (CAMPBELL; JR; HSU, 2002). O jogador criado pela IBM foi responsável por vencer do campeão mundial de xadrez Garry Kasparov em 1997 através do uso do algoritmo Minimax executando em paralelo em supercomputadores (HSU; CAMPBELL; JR, 1995).

Os métodos associados a aprendizado de máquina são, normalmente, divididos entre aprendizado supervisionado, aprendizado não supervisionado e aprendizado por reforço. No aprendizado supervisionado queremos determinar uma função que, dado parâmetros de entrada, retorne o valor esperado a partir de exemplos de pares entrada-saída. O objetivo no aprendizado não supervisionado é realizar a identificação de padrões nos dados onde não há rótulos para cada entrada.

Diferente dos demais, no aprendizado por reforço temos agentes que realizam ações inteligentes a partir de prévias interações com o ambiente e estados que representam a situação a cada momento. Cada ação tomada por um agente pode alterar o estado corrente. Essa modelagem é comumente utilizada em jogos pois como em exemplos simples de jogos de tabuleiro, temos que as configurações do tabuleiro representam os estados e cada movimento realizado pelos jogadores podem ser substituídos por ações tomadas por um agente. Se o movimento realizado leva o jogo para uma configuração mais próxima da vitória, o agente recebe uma recompensa para reforçar a decisão tomada.

Em 2015 o software AlphaGo (SILVER et al., 2016) venceu sua primeira partida contra um jogador profissional de Go por 5-0 e posteriormente venceu Lee Sedol, vencedor de 18 títulos mundiais. O jogo Go apresenta um nível de complexidade muito maior ao encontrado no xadrez e os desenvolvedores do AlphaGo utilizaram aprendizado por reforço a partir da combinação de buscas avançadas em árvores com redes neurais profundas para obter resultados tão significativos.

Há diversos desafios na obtenção de bons resultados a partir do aprendizado por reforço. Jogos como o Go possuem uma quantidade muito grande de configurações possíveis. Consequentemente, o espaço de estados que deve ser explorado pelo agente pode ser muito grande e é necessário muito tempo de treinamento em máquinas que atendam os requisitos de memória e processamento. Além disso, temos que a escalabilidade do treinamento pode impactar o resultado do conhecimento já adquirido. Como a única forma de comunicação com o agente é a partir do feedback dado a suas ações, a interação com novos ambientes e novas escolhas pode gerar um esquecimento das ações já aprendidas. Por fim, pela busca de maximizar a recompensa, o agente pode encontrar um ótimo local e não solucionar o

problema proposto.

Motivados pelo uso do aprendizado por reforço em jogos de xadrez para auxiliar no ensino de jogadores iniciantes, vamos explorar configurações não tradicionais do jogo utilizadas em aulas pelo professor Luís Jímenez. As configurações foram criadas por Luís a partir de sua experiência prática no ensino do jogo para mostrar algum movimento ou estratégia especifica ao aluno. O objetivo das configurações não é mostrar como jogar em um arranjo preciso das peças dentro de uma partida tradicional completa, mas ensinar como as peças funcionam separadamente e enfatizar a sinergia entre elas para obter sequências de bons movimentos.

Durante o processo de aprendizado do jogo de xadrez para crianças, é importante manter o aluno motivado e ativo. Porém, quando não está em aula síncrona, é necessário um outro jogador para jogar com o aluno e auxiliá-lo com estratégias. Com agentes inteligentes, suprimos a necessidade de outro jogador e ainda é possível utilizá-los para aprender e analisar movimentos. Como os agentes são "justos"do ponto de vista que ganham quando possuem uma vantagem e buscam as melhores jogadas quando estão em desvantagem, podemos observar seus movimentos para verificar quais são as estratégias interessantes.

Nosso objetivo neste trabalho é aplicar o aprendizado por reforço a partir de configurações do jogo de xadrez, avaliar a performance dos agentes obtidos dado as características de cada configuração e disponibilizá-los para partidas contra pessoas com diferentes níveis de conhecimento do jogo para obter conclusões sobre aprendizado. O jogo de xadrez completo gera uma quantidade muito grande de possibilidades fazendo necessário a utilização de técnicas avançadas e computadores robustos como na implementação do Deep Blue. Portanto, vamos explorar o aprendizado por reforço iniciando as partidas nas configurações criadas pelo professor Luís. Os experimentos vão utilizar 3 configurações com características diferentes e níveis de dificuldade fácil, médio e difícil respectivamente. Além disso, para obter melhor desempenho e facilitar a avaliação, utilizamos técnicas e ferramentas para representar o tabuleiro com o espaço de estados a ser explorado pelo agente e para auxiliar na visualização das jogadas feitas durante as partidas no treinamento.

No capítulo 2 apresentamos a teoria do xadrez necessária para o entendimento dos experimentos. No capítulo 3, apresentamos as técnicas auxiliares utilizadas na implementação para realizar a representação do tabuleiro e a visualização das jogadas das partidas. No capítulo 4, descrevemos o método de aprendizado por reforço utilizado em nossos experimentos. No capítulo 5, descrevemos as configurações e suas características, as tentativas de treinamento em cada uma e os resultados obtidos. Além disso, disponibilizamos os agentes treinados nas duas primeiras configurações no endereço https://ajedrez.aplicando.com.co para realizar jogos contra alunos iniciantes e jogadores experientes e obter dados sobre o comportamento do agente como um auxiliar no ensino do xadrez. Por fim, no capítulo 6, concluímos o trabalho.

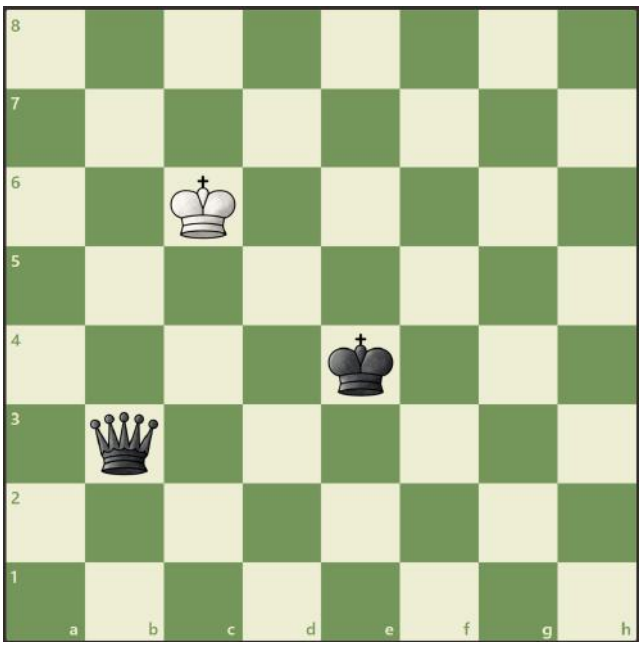
2 TEORIA DO XADREZ

Neste capítulo vamos explicar um pouco dos conceitos do xadrez usados nos diferentes experimentos, para poder entender se o agente aprendeu as estratégias certas ou não, assim como o valor das peças no jogo e a leitura do tabuleiro para acompanhar as análises.

2.1 COORDENADAS NO TABULEIRO DE XADREZ

O tabuleiro de xadrez é um tabuleiro de 64 casas, composto por 8 filas e 8 colunas. Cada fila vai estar associada a um número (1,2,3,4,5,6,7,8) e cada coluna com uma letra (a,b,c,d,e,f,g,h), com isso podemos nos referir a uma casa no tabuleiro como uma coordenada composta pela letra da coluna e o número da fila (Figura 1).

Figura 1 – Tabuleiro de xadrez



Tabuleiro com rei preto na casa e4, rei branco na casa c6 e rainha preta na casa b3.

2.2 PEÇAS NO TABULEIRO DE XADREZ

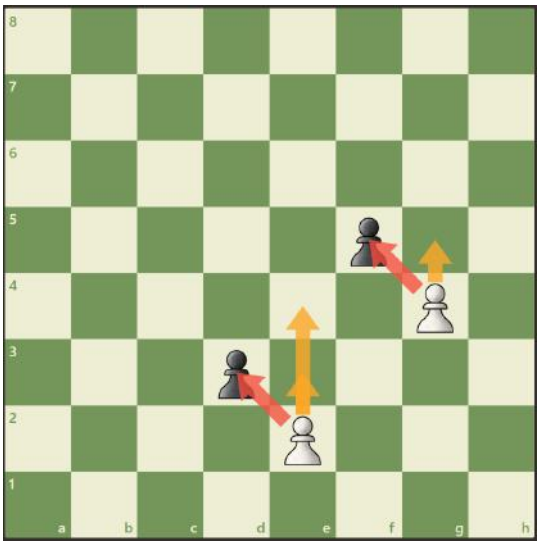
No xadrez existem 6 tipos de peças, cada uma delas com valores e formas de se mover diferentes no tabuleiro, fazendo com que cada uma seja mais útil que outras em diferentes posições do tabuleiro.

2.2.1 Peão

No início de uma partida, cada jogador tem oito peões que são dispostos nas fileiras 2 para as brancas e 7 para as pretas. O peão move-se verticalmente na coluna que encontra-se, sendo incapaz de recuar (Figura 2). No primeiro movimento de cada peão, a partir do ponto de partida, é possível avançar duas casas e, a partir daí, uma. Seu valor é de 1 e é a referência para os valores das outras peças.

Um peão pode capturar uma peça que esteja na diagonal adjacente a sua posição. Quando o peão está na quinta fileira temos um caso especial onde ele pode capturar en pessant, ou seja, o peão adversário na coluna adjacente que avançar duas casas em seu primeiro movimento. Ao atingir a oitava linha, um peão é substituído por qualquer outra peça, exceto o rei. Este movimento é chamado de coroação ou promoção. Quando há a promoção, a peça não pode ser removida na mesma jogada.

Figura 2 – Movimentação do peão

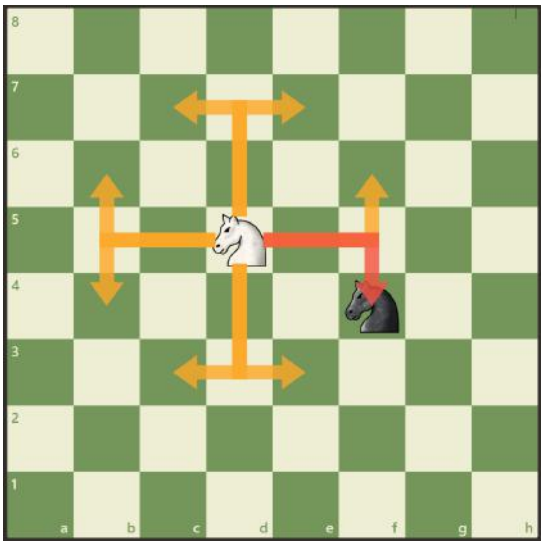


Na figura temos um exemplo de movimentos dos peões, o peão de e2 pode capturar o de d3, ou se movimentar para e3 ou e4. O peão de g4 pode se movimentar para g5 ou capturar o peão de f5.

2.2.2 Cavalo

O cavalo é uma peça menor de um valor aproximado de três peões ou pontos. Tem um movimento assemelhado a um "L"(Figura 3) e, diferente das outras peças, pode pular as peças intervenientes. Captura tomando a casa ocupada pela peça adversária, sendo sempre no final do L.

Figura 3 – Movimentação do cavalo

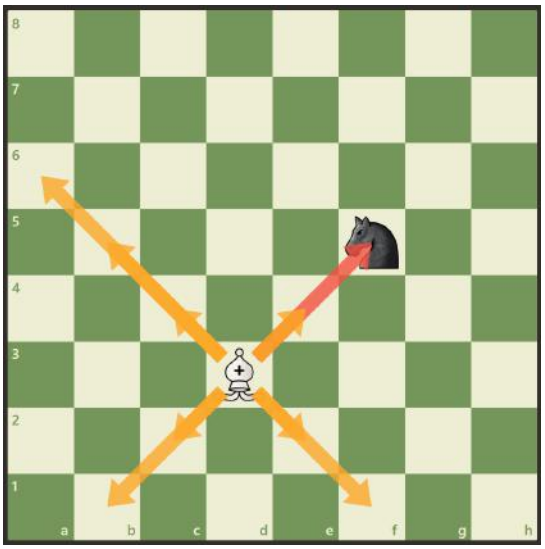


Na figura temos um cavalo na casa d5, podendo se mover para f6, e7, c7, b6, b4, c3, e3 e podendo capturar o cavalo de f4.

2.2.3 Bispo

O Bispo é uma peça menor de valor aproximado de três peões ou pontos. Movimenta-se em diagonal (Figura 4), não podendo pular peças intervenientes, e captura tomando o lugar ocupado pela peça adversária. Devido às características de seu movimento tem a deficiência da fraqueza da cor, ou seja, seu movimento fica limitado à cor da casa de onde inicia a partida.

Figura 4 – Movimentação do bispo



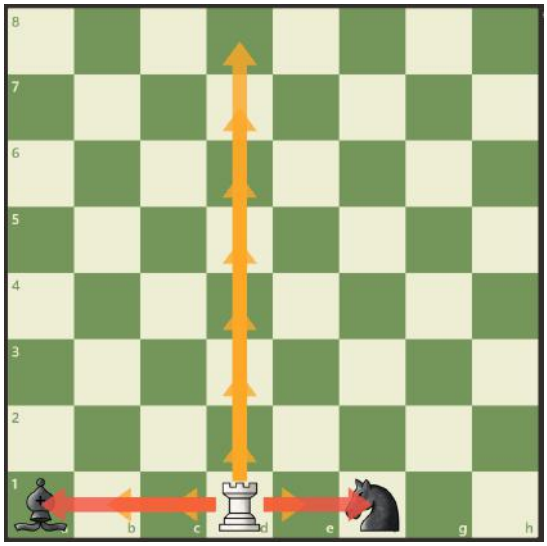
Na figura o bispo está na casa d3, e pode se mover para as casas e4, e2, f1, c4, b5, a6, c2, b1, e pode capturar o cavalo de f5.

2.2.4 Torre

A Torre é uma peça maior do xadrez com um valor relativo de aproximadamente cinco peões ou pontos, podendo variar em função de seu posicionamento em colunas ou fileiras abertas, ou formações estratégicas.

Movimenta-se em linhas retas nas colunas e fileiras do tabuleiro (Figura 5), não podendo, entretanto, pular peças adversárias ou aliadas e captura ao ocupar a casa deixada pelo adversário.

Figura 5 – Movimentação da torre



Na figura a torre pode se movimentar para e1, c1, b1, d2, d3, d4, d5, d6, d7, d8 e capturar as peças pretas em f1 e a1.

2.2.5 Rainha

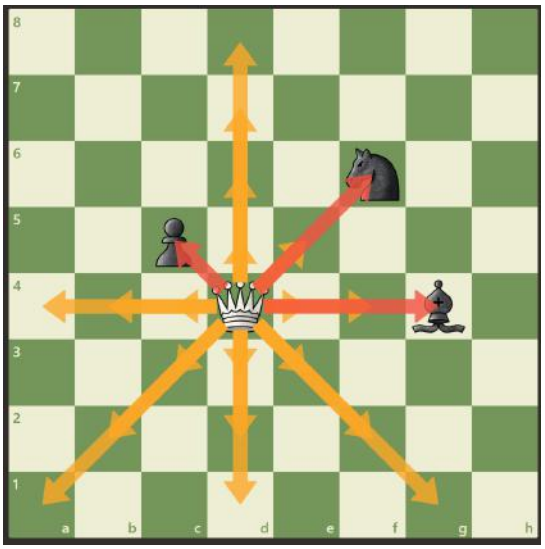
A Dama ou Rainha é uma peça maior do jogo de xadrez, é a peça de maior valor relativo do jogo, usualmente valorada entre nove e dez pontos. Ela se movimenta em linhas retas pelas fileiras (Figura 6), colunas e diagonais no tabuleiro. Não pode pular suas próprias peças ou as adversárias e captura tomando a casa ocupada pela adversária.

2.2.6 Rei

O Rei não pode ser trocado durante uma partida, ele é considerado uma peça de valor inestimável. Durante uma partida, o Rei não pode permanecer sob ameaça das peças adversárias em nenhum instante, devendo ser colocado em segurança imediatamente no movimento seguinte, caso seja atacado.

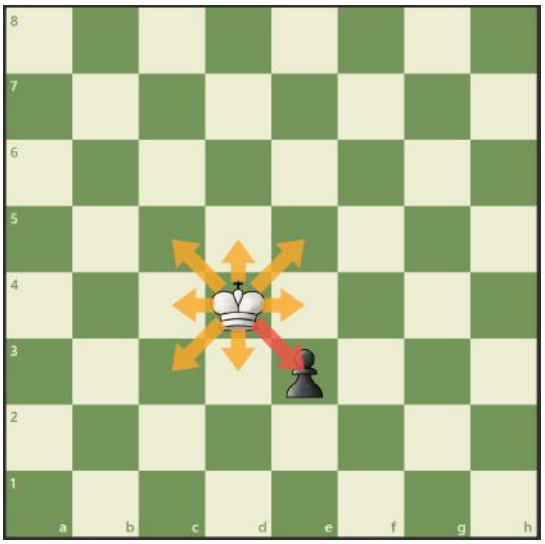
A sua movimentação consiste no deslocamento de uma casa na direção horizontal, vertical ou diagonal (Figura 7), desde que ela não esteja sob ataque adversário.

Figura 6 – Movimentação da rainha



Na figura a rainha pode se movimentar para d5, d6, d7, d8, e5, e4, f4, e3, f2, g1, d3, d2, d1, c3, b2, a1, c4, b4, a4 e pode capturar as peças pretas em f6, g4 e c5.

Figura 7 – Movimentação do rei



Na figura o rei pode se movimentar para d5, c5, c4, c3, d3, e4 e e5, além disso pode capturar o peão inimigo em e3.

2.3 DESCREVENDO OS MOVIMENTOS

Com os movimentos das peças claros e a forma de descrever casas do tabuleiro como coordenadas, podemos agora escrever a sequência de jogadas de um jogo numa determinada posição. Para isso vamos seguir as seguintes regras:

- Cada tipo de peça (que não seja um peão) é identificada por uma letra maiúscula, K para o rei, Q para dama, R para a torre, B para o bispo, e N para o cavalo.

- Os peões não são indicados por uma letra, mas pela ausência dessa letra.
- Cada jogada é indicada pela letra da peça, mais a coordenada da casa de destino. Por exemplo Be5 (bispo se move para e5), Nf3 (cavalo se move para f3), c5 (peão se move para c5 – sem inicial no caso de jogadas com peão).
- Quando uma peça faz uma captura, um x é colocado entre a inicial e a casa de destino. Por exemplo, Bxe5 (bispo captura a peça em e5). Quando um peão faz uma captura, a coluna da qual o peão partiu é usada no lugar da inicial da peça. Por exemplo, exd5 (peão na coluna e captura a peça em d5).

2.4 DESCREVENDO UMA PARTIDA PELAS JOGADAS

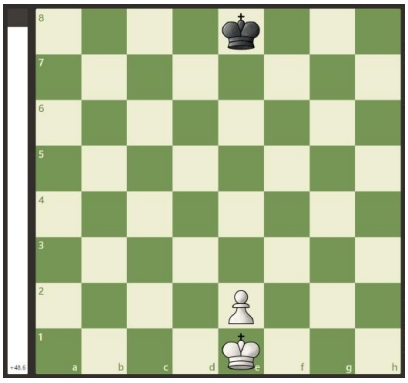
Dado que temos uma forma simplificada de descrever os movimentos, se temos conhecimento da configuração inicial do tabuleiro, podemos descrever a partida inteira apenas com as jogadas que foram realizadas. A vantagem dessa notação é sua simplicidade e sua utilização em softwares para exportação e importação, como veremos na seção 3.2.1.

Como sabemos onde as peças começam na partida, cada rodada é descrita com o movimento de uma peça branca e em seguida o movimento de uma peça preta. Com isso, cada jogador possui um movimento por rodada e o fim do jogo é indicado por 1-0 caso o branco ganhe, 1/2-1/2 caso empate e 0-1 caso o preto ganhe.

Por exemplo, dado a configuração da Figura 8, podemos ter o seguinte jogo:

1. Kd2 Ke7 2. Kd3 Kf7 3. e4 Ke6 4. Kd4 Kd6 5. e5 Ke6 6. Ke4 Ke7 7. Kd5 Kd7 8. e6 Ke8 9. Kd6 Kd8 10. Ke5 Ke8 11. Kd5 Ke7 12. Ke5 Ke8 13. Kd6 Kd8 14. e7 Ke8 15. Ke6 1/2-1/2

Figura 8 – Exemplo de configuração inicial



Além disso, um caso especial dentro dessa representação é quando um peão atravessa o tabuleiro e é promovido. Nesse caso, temos um movimento como e8=Q, onde um peão branco chega na linha 8 e é promovido para uma rainha (Queen).

3 REPRESENTAÇÃO E VISUALIZAÇÃO DO TABULEIRO NO CÓDIGO

O primeiro passo para poder trabalhar em algoritmos com jogos de tabuleiro é encontrar uma forma de representar o tabuleiro computacionalmente de modo que podemos facilmente realizar operações como validação das jogadas e alterar a configuração das peças após o movimento de um dos jogadores. A forma de representação é muito importante pois ela pode adicionar limitações ou aumentar a demanda computacional antes mesmo de adicionarmos algoritmos de aprendizado ao jogo.

Para facilitar análises qualitativas de jogadas individuais podemos utilizar métodos e ferramentas para visualização do tabuleiro e as jogadas realizadas durante uma partida. Nesse caso, queremos passar a representação do tabuleiro para uma forma mais intuitiva para as pessoas. Neste capítulo vamos introduzir como foi feita a representação e visualização do tabuleiro que auxiliou tanto na implementação do algoritmo de aprendizado quanto na análise das partidas resultantes.

3.1 REPRESENTAÇÃO DO TABULEIRO

Uma das principais maneiras de representar o tabuleiro é com uma hash, onde cada configuração do tabuleiro possui um valor associado. Neste trabalho utilizaremos o método de Zobrist Hashing, introduzido em (ZOBRIST, 1970).

O objetivo do Zobrist Hashing é transformar uma configuração arbitrária do tabuleiro a partir de números aleatórios, facilitando a representação e operações no código. Além disso, esse hash possui bom desempenho para o cálculo de novas posições, pois se baseia apenas no uso do XOR em sua construção.

Cada peça recebe um valor aleatório para cada casa do tabuleiro e as configurações do tabuleiro são formadas realizando o XOR de cada peça. Com isso, para números de 32 bits podemos ter até 65 mil configurações sem colisão e para número de 64 bits podemos ter até 4 bilhões de configurações.

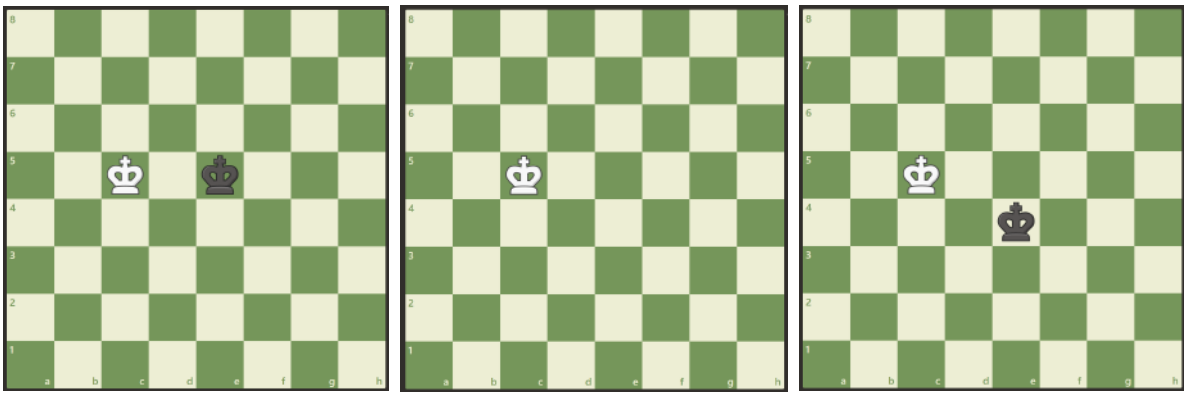
No começo do nosso trabalho gerávamos sempre números aleatórios para compor as posições do tabuleiro. Depois percebemos que era mais fácil ter uma base fixa de números aleatórios para que cada novo teste possa ser comparado com os outros testes anteriores. Assim temos 64 valores aleatórios para cada peça do jogo, um valor para cada posição da peça no tabuleiro. Para isso criamos um código auxiliar que associa cada peça em uma posição a um valor aleatório e armazenamos em um arquivo tendo certeza que os números associados a todas as posições nunca vão colidir. Logo, quando vamos executar um jogo, inicialmente carregamos os valores aleatórios do arquivo que já tinham sido pré-calculados para facilitar a comparação e representação dos resultados posteriormente.

3.1.1 Algoritmo para gerar novas configurações

Utilizando a Zobrist Hashing, não é necessário calcular o hash do zero para gerar novas posições a partir de uma já calculada. No jogo de xadrez precisamos realizar apenas duas operações XOR para obter a nova hash a partir do movimento de uma peça.

No movimento de uma peça realizamos o XOR com o número associado à peça na posição inicial para removê-lo do tabuleiro e então outro XOR com o número associado à peça na posição final (Figura 9).

Figura 9 – Gerando uma nova configuração



(a) Configuração atual. (b) Remoção da peça movida. (c) Nova configuração.

Como exemplo, podemos supor que na Figura 9a o número aleatório para o rei branco em c5 é 87 e que para o rei preto em e5 é 45. Logo, a hash para a configuração da Figura 9a será bin(87) XOR bin(45) = bin(122), onde bin(i) corresponde a representação binária do inteiro i. Para realizar o movimento do rei preto para uma nova casa, primeiro removemos a peça do tabuleiro realizando novamente o XOR com o valor associado a peça na posição atual (bin(122) XOR bin(45) = bin(87)), depois outro XOR com o valor associado a nova posição da peça (supondo que o rei preto na nova posição, e4, seja 55, teríamos bin(87) XOR bin(55) = bin(96)) e conseguimos a nova hash correspondente a nova configuração do tabuleiro (Figura 9c).

No caso em que o movimento leva a uma posição onde já tem uma peça do outro jogador, devemos removê-la do tabuleiro também e seria necessário realizar mais uma operação XOR. Porém, conseguimos facilmente perceber a simplicidade para o cálculo de novas configurações a partir de uma já calculada e sua velocidade de ordem 1, pois são necessárias apenas algumas operações básicas para chegar ao resultado, sem a necessidade de calcular o hash do zero. Além disso, o uso da hash baseada na operação XOR fornece uma facilidade para comparar as configurações dentro de uma partida, auxiliando na identificação de um empate causado pela repetição da mesma configuração 3 vezes.

3.2 VISUALIZAÇÃO DAS JOGADAS

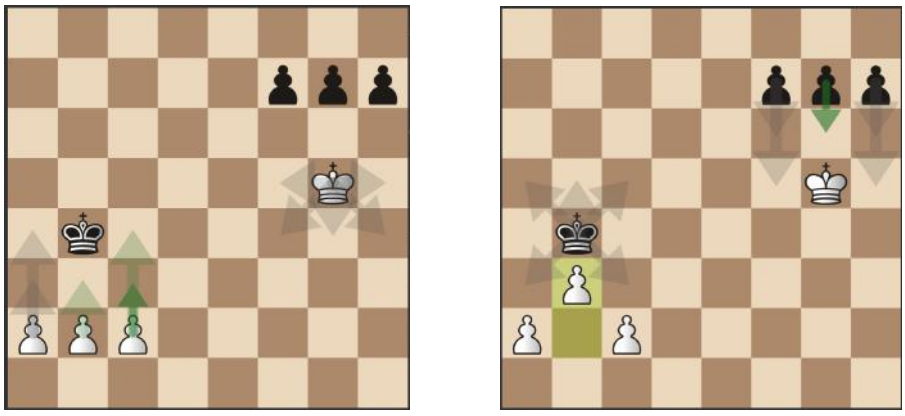
Na internet existem diversas ferramentas e bibliotecas com a lógica e as regras básicas do xadrez em diferentes linguagens de programação. Porém como em nossos experimentos, utilizamos configurações de tabuleiros que podem não seguir as regras convencionais, criamos nossa própria versão e validamos com duas ferramentas da internet bem famosas, o Chess.com e o Opening tree.

3.2.1 Opening tree[1]

O Opening tree é uma ferramenta na internet de código aberto usada para ter uma visão consolidada de todos os jogos feitos e poder ver um histórico das jogadas que foram feitas em diferentes posições. Para uma pessoa que esta aprendendo a jogar xadrez serve para ver as respostas de diferentes adversários em varias posições. No nosso trabalho, ela serve para duas coisas importantes: a primeira é para validar que as jogadas feitas no treinamento por reforço são jogadas válidas que seguem as regras do xadrez, e a segunda é para poder ver todas as tentativas que o modelo teve durante o treinamento e a competição.

Na figura 10a vemos as jogadas que o jogador com as peças brancas examinou durante o treinamento (setas verde claro) e qual foi o movimento que ele aprendeu (seta verde escura) como sendo a melhor jogada (b3). Da mesma forma, o jogador com as peças pretas aprendeu a jogar em g6 ao final do treinamento. Com isso podemos visualizar os jogos jogados pelo modelo e entender qualitativamente como foi o aprendizado.

Figura 10 – Gerando uma nova configuração



(a) Primeira Jogada. (b) Segunda Jogada.

[1] https://www.openingtree.com/

3.2.2 Chess.com[2]

O chess.com é um site famoso para jogar xadrez online, onde podemos jogar contra outros jogadores, fazer exercícios, jogar torneios e treinar jogando contra um computador. O site é usado em nossos experimentos para avaliar as configurações do tabuleiro que utilizamos e determinar se há vantagem para algum dos jogadores. Além disso, podemos testar os modelos gerados para disputar jogos contra os programas do site e avaliar o desempenho.

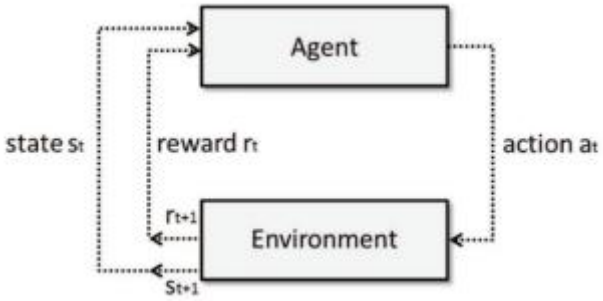
[2] https://www.chess.com/

4 APRENDIZADO POR REFORÇO

O aprendizado por reforço é uma técnica utilizada para treinar modelos permitindo que um agente realize ações e interaja com o ambiente, a fim de maximizar suas recompensas. O agente deve descobrir quais ações resultam em boas recompensas explorando suas possibilidades. Em muitos casos temos que ações podem não parecer interessantes a curto prazo, mas possuem grande impacto no resultado final. As principais características do método de aprendizado por reforço são a busca por tentativa e erro e recompensa tardia.

Na Figura 11 temos um diagrama simples de como é feito o aprendizado. O agente realiza um ação interagindo com o ambiente, resultando em um novo estado para o sistema. O resultado das ações geram recompensas ao agente para guiar seu aprendizado de modo a maximizá-las.

Figura 11 – Diagrama do processo de aprendizado por reforço



Fonte: (MORALES; ZARAGOZA, 2012)

Além disso, há variações na forma de determinar as ações do agente. Podemos utilizar uma estratégia de permitir o agente explorar os estados sem nenhum conhecimento prévio, de modo que, apenas a partir das recompensas, é determinado a qualidade das ações e dos estados. Nesse caso, é necessário uma maior quantidade de interações para descobrir todos os estados possíveis e convergir os valores associados a qualidade de cada um. Por outro lado, podemos utilizar funções de qualidade auxiliares para indicar quais seriam os estados mais interessantes para o agente. As funções auxiliares determinam um valor para cada estado baseado nas características que ele possui. Porém, para construir a função, precisamos codificar formas de avaliação do ambiente e há a possibilidade de enviesar as ações ou evitar que seja encontrada uma solução.

No caso do jogo de xadrez, podemos avaliar uma configuração do tabuleiro a partir das peças que estão presentes e seus respectivos posicionamentos. Contudo, a dificuldade de levantar e implementar todas as regras e estratégias necessárias para esse método geram uma margem de erro muito grande. Sendo assim, mesmo com a necessidade de explorar

estados "ruins"e mais iterações para convergência dos valores associados a cada estado, optamos por não utilizar métodos de avaliação dos estados que não se baseassem no uso das recompensas pelas ações do agente.

Existem diversos problemas com relação ao aprendizado por reforço em situação do mundo real. Em muitos casos, o espaço de estados possíveis é muito grande para ser explorado por completo ou podemos ter muitos agentes interagindo ao mesmo tempo em um determinado ambiente.

O aprendizado por reforço possui diversas aplicação práticas como no controle de semáforos de transito (AREL et al., 2010), robótica (KOBER; BAGNELL; PETERS, 2013), sistemas de recomendação (ZHENG et al., 2018) e vários jogos como o famoso caso do AlphaGo (SILVER et al., 2016). Nesse trabalho vamos mostrar como podemos utilizar o aprendizado por reforço em configurações não tradicionais do jogo de xadrez, sua aplicação como auxiliar para o ensino de jogadores iniciantes e quais são suas limitações quando aumentamos o espaço de estados.

4.1 ESTRATÉGIA PARA O APRENDIZADO

Para a nossa estratégia de aprendizado por reforço, nos baseamos em um algoritmo básico apresentado em (SUTTON; BARTO, 2018). No exemplo do livro, foi apresentado um aprendizado para o jogo tic-tac-toe (jogo da velha) com o uso da fórmula para determinar a qualidade dos estados:

$$V_{k+1}(S_t) \leftarrow V_k(S_t) + \alpha[V_k(S_{t+1}) - V_k(S_t)] \tag{4.1}$$

Na equação 4.1 temos que V_k é a função que retorna a estimativa no instante k para a vitória a partir do estado atual S_t, próximo estado S_{t+1} e uma constante positiva $\alpha > 0$ (chamada *step-size parameter*). Essa equação pode ser reescrita da forma mais intuitiva como:

$$NewEstimate \leftarrow OldEstimate + StepSize[Target - OldEstimate] \tag{4.2}$$

No caso do jogo de xadrez, cada estado corresponde a uma configuração do tabuleiro e a transição de estados é feita pela movimentação de uma peça. A estimativa de vitória de um estado S corresponde ao valor de qualidade de cada estado. Inicialmente, todas as configurações não terminais recebem o valor de 0,5, as configurações terminais que indicam a vitória recebem 1,0 e derrota 0,0, ou seja:

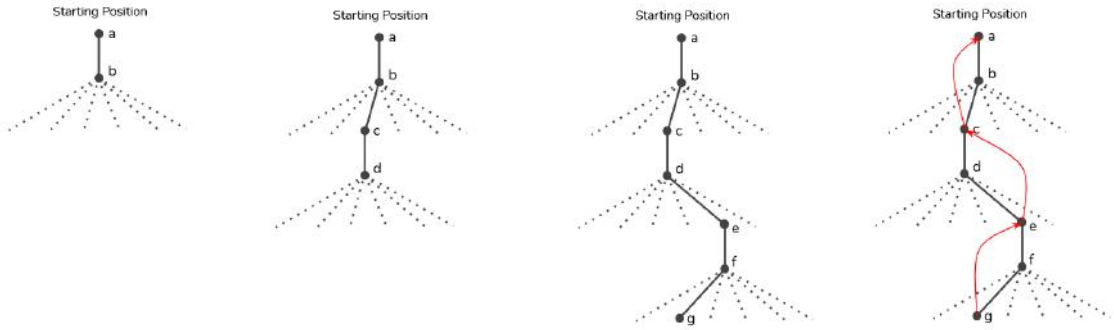
$$V(S) = \begin{cases} 0{,}5 \text{ se } S_t \text{ é não terminal} \\ 1{,}0 \text{ se } S_t \text{ é um estado de vitória} \\ 0{,}0 \text{ se } S_t \text{ é um estado de derrota} \end{cases}$$

Com isso, pela fórmula, temos que como α é positivo, no caso em que o jogo chega em uma vitória todos os estados percorridos recebem um incremento no valor, indicando que esses estados possuem maior chance de levar a uma vitória. No caso em que o jogo chega em uma derrota, todos os estados percorridos recebem um decremento, indicando que possuem menor chance de levar a uma vitória. Após realizar várias partidas atualizando os valor para os estados, temos que as jogadas boas que levaram a mais vitória possuem maior qualidade do que as jogadas que levaram a mais derrotas.

Como o objetivo do agente é chegar em uma vitória, dado que ele está no estado S_t, ele deve escolher ir para o estado S_{t+1} com maior valor de qualidade. Assim, se ele perde, esses estados que eram considerado bons são atualizados com valores menores. Além disso, caso o maior valor esteja presente em dois ou mais estados adjacentes a S_t, escolhemos aleatoriamente que caminho seguir entre os melhores.

Para que a sequência de jogadas não se repita sempre e haja a oportunidade de encontrar outros movimentos que levam a vitória, temos um variável associada a probabilidade de exploração. Sendo assim, antes de cada movimento, podemos decidir seguir um caminho que não seja o atual melhor. Ao escolher uma jogada aleatoriamente, temos a chance de explorar novas situações. Além disso, para não prejudicar os estados antes do movimento exploratório, após o resultado da partida, atualizamos apenas os estados posteriores com o feedback. Tomamos essa ação pois podemos estar saindo de um estado ótimo para um estado ruim com a exploração e realizar a atualização da qualidade nesse caso pode resultar em diminuir o valor de um estado desejável.

Figura 12 – Sequência de movimentos no aprendizado

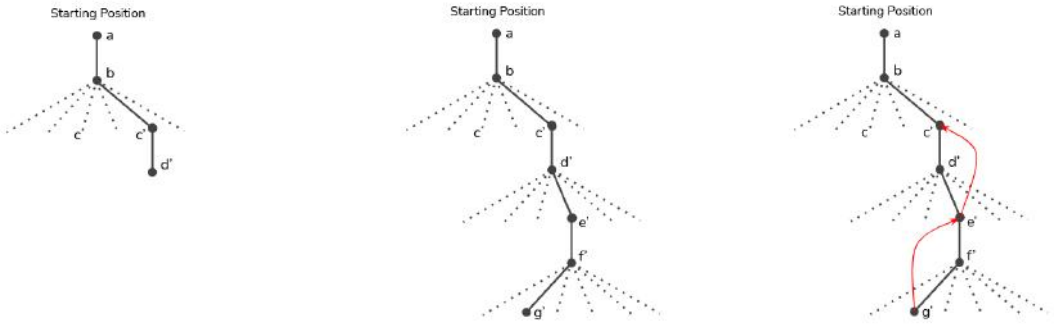


(a) Momento 1 (b) Momento 2 (c) Momento 3 (d) Recompensa

Na figura 12 temos um exemplo de execução do algoritmo. Seja essa a primeira partida do treinamento, temos que todos os estados não terminais possuem valor 0.5, a configuração de vitória com valor 1.0 e derrota com 0.0. Começamos no estado inicial a e digamos que o movimento do oponente que leva ao estado b. O agente determina os estados adjacentes a b. Em 12b, o agente decide ir para o estado c e o oponente leva ao estado d. Seguimos esses passos até um estado terminal g. Vamos supor que g corresponde

a vitória para o agente. Então, os estados de escolha do agente recebem uma atualização no valor pela fórmula 4.1. Se $\alpha = 0.1$, temos que os novos valores para os estados serão: $V(a) = 0.5005, V(c) = 0.505, V(e) = 0.55$.

Figura 13 – Sequência de movimentos no aprendizado - partida 2



(a) Momento 1 (b) Momento 2 (c) Recompensa

Digamos que na figura 13 temos a execução da segunda partida do treinamento de exemplo. Nesse caso, no primeiro momento (13a), como o estado c possui valor maior que os demais, o agente deveria seguir a este estado. Porém, a partir da taxa de exploração, o agente seguiu um novo caminho que levou a uma nova sequência de estados. Vamos supor que g' corresponde a derrota para o agente. Então, apenas os estados de escolha do agente após a exploração recebem uma atualização no valor. Logo, os novos valores para os estados serão: $V(c') = 0.495, V(e') = 0.45$.

Além disso, é interessante destacar que em uma mesma partida podemos realizar o treinamento de dois agentes simultaneamente. Como vimos no exemplo, o agente atualiza apenas os estados referentes aos seus movimentos e a partida é bem divida em turnos. Então, utilizando dois dicionários e realizando as atualizações dos valores dos estados independentemente por jogador, obtemos um agente treinado para cada lado do jogo (peças pretas e peças brancas no caso do xadrez).

5 EXPERIMENTOS

Nesse capítulo vamos ver diferentes configurações de peças no tabuleiro com dificuldades variadas que foram utilizadas como estado inicial em nossos experimentos e os resultados obtidos do treinamento de agentes para cada uma dessas configurações. Como cada configuração possui características especificas, além de realizar uma análise quantitativa baseada na quantidade de vitórias dos jogadores, vamos utilizar as ferramentas mencionadas na seção 3.2 para realizar uma avaliação qualitativa das jogadas realizadas pelo modelo gerado.

Em cada configuração vamos realizar um aprendizado por reforço usando a estratégia da seção 4.1 variando os seguintes parâmetros, que permitem alterar a busca e o resultado dos agentes:

- *ExplorationRate* (probabilidade de exploração): Para cada jogador, em cada jogada, permite explorar jogadas diferentes à melhor alternativa corrente, tentando achar possíveis jogadas alternativas que, à primeira vista, pareçam ruins mas que no longo prazo geram melhores resultados. Com uma alta taxa de exploração, o agente vai tentar a cada jogada experimentar algo novo, evitando um aprendizado concreto e jogando aleatoriamente. Com uma baixa taxa de exploração podemos estar perdendo jogadas excelentes que no começo pareciam ruins.

- *StepSize* (Constante positiva utilizada na fórmula de feedback): Taxa de aprendizado que retroalimenta o valor das jogadas feitas no jogo, para gerar na memória valores maiores ou menores que diferenciam as jogadas boas das ruins. Uma alta taxa de aprendizado significa que, se um jogo teve um resultado ruim, as chances de tentar todas as jogadas deste jogo vão ser baixas, e se o jogo teve um resultado bom, vou sempre jogar as mesmas jogadas. Para uma baixa taxa de aprendizado se aplicaria o contrário.

- *Número de jogos para treinamento*: Quantidade de jogos que os agentes vão ter para treinar e aprender as jogadas boas e ruins. Com um número grande de jogos, teremos mais jogadas exploradas e melhor conhecimento sobre quais jogadas são ruins ou boas.

Em cada experimento definimos listas de possíveis valores para cada parâmetro do treinamento (ExplorationRate, StepSize, Games). Com isso, temos um agente para cada combinação de valores dos parâmetros.

5.1 AVALIAÇÃO

Após chegarmos ao final do treino, quando atingimos a quantidade de jogos passado como parâmetro de entrada, entramos em uma fase de avaliações para determinar se houve algum avanço no aprendizado do jogador. Para cada experimento, podemos utilizar três formas de avaliar os agentes e seu aprendizado. O objetivo dessa fase é identificar o melhor agente e analisar seu desempenho.

5.1.1 Teste

O treino de um agente é feito com ele jogando nas peças brancas contra ele mesmo jogando nas peças pretas. Assim, um agente aprende jogando contra si mesmo, explorando jogadas e atualizando o valor dos estados de acordo com o resultado final de cada partida.

A primeira avaliação que fazemos é jogar o agente contra ele mesmo alguma quantidade de vezes (normalmente utilizamos 1000 jogos). Nesse caso, como estamos apenas testando e não treinando, não utilizamos a fórmula de feedback 4.1 para atualizar os valor de cada estado depois da partida e não realizamos jogadas de exploração.

O objetivo dessa primeira avaliação é oferecer um resultado preliminar para análise do aprendizado. Como utilizamos apenas configurações inciais especificas do tabuleiro de xadrez, muitas delas podem oferecer uma vantagem para um dos lados. Sendo assim, podemos considerar a quantidade de vitórias de cada lado, branco e preto, para verificar se condiz com alguma vantagem oferecida pela própria configuração do tabuleiro.

5.1.2 Torneio

Depois que temos vários agentes que treinaram utilizando valores diferentes nos parâmetros de acordo com o experimento, um ponto interessante para olharmos é como identificar qual dos agentes é potencialmente o "melhor". Temos que em nossos experimentos, assim como no jogo completo de xadrez, cada partida resulta em vitória, derrota ou empate. Com isso, utilizamos uma estratégia que se baseia em fazer um torneio entre os agentes para obter um ranking.

Cada agente se resume em dois jogadores, um de peças pretas e outro de peças brancas, onde cada jogador é definido por um conjunto de valores associados a qualidade dos estados que utilizamos para determinar as jogadas que devem ser feitas. Sendo assim, podemos montar o torneio de duas formas diferentes:

- **Baseado em cores** - Cada jogador de peças brancas joga contra todos os jogadores de peças pretas. Nesse caso o foco está em cada jogador e temos como resultado 2 rankings, um para os jogadores de peças pretas e outro para os jogadores de peças brancas. Como o agente é dividido, pode ser mais fácil identificar os melhores jogador de cada cor, pois se temos um agente que joga bem com as peças pretas mas

se não joga bem com as peças brancas terá uma boa posição no ranking de jogadores de peças pretas. Além disso, fazemos no máximo 3 jogos em que o primeiro jogador a vencer 2 jogos ganha. Com isso, os rankings gerados terão as seguintes colunas:

  - *Explo*: Taxa de exploração utilizada no treino
  - *Step*: Constante utilizada na fórmula de feedback
  - *Jogos*: Quantidade de jogos no treino
  - *2-0*: Número de partidas vencida de 2 a 0
  - *2-1*: Número de partidas vencida de 2 a 1
  - *1-2*: Número de partidas perdidas de 1 a 2
  - *0-2*: Número de partidas perdidas de 0 a 2

- **Baseada em agentes** - Cada agente joga contra todos os outros agentes. Nesse só temos um ranking e são feitos 2 jogos entre cada agente, um jogo de cada lado, ou seja, se um agente jogou com as peças brancas no primeiro jogo, então ele joga com as peças pretas no seguinte. Com isso, os rankings gerados terão as seguintes colunas:

  - *Explo*: Taxa de exploração utilizada no treino
  - *Step*: Constante utilizada na fórmula de feedback
  - *Jogos*: Quantidade de jogos no treino
  - *2-0*: Número de partidas vencida de 2 a 0
  - *1-1*: Número de partidas empatadas em 1 a 1
  - *0-2*: Número de partidas perdidas de 0 a 2

5.1.3 Jogar contra pessoas

Uma forma de avaliar os agentes de forma mais qualitativa é realizando partidas contra pessoas, especialmente no caso em que a pessoa possui um conhecimento técnico sobre o jogo. Dessa forma, além de conseguir os resultados de quem está vencendo as partidas, temos uma visão melhor sobre as jogadas que são feitas no meio do jogo.

Ao jogar contra uma pessoa, cada jogada é levada em consideração. Com isso, é mais fácil identificar momentos em que o agente realiza um movimento ruim, que entregue uma vantagem mesmo que momentânea. Além disso, é possível controlar qual estratégia utilizar contra o agente e verificar qual é sua resposta.

As pessoas que ajudaram a testar os agentes durante todo o projeto estão divididas nas seguintes faixas etárias

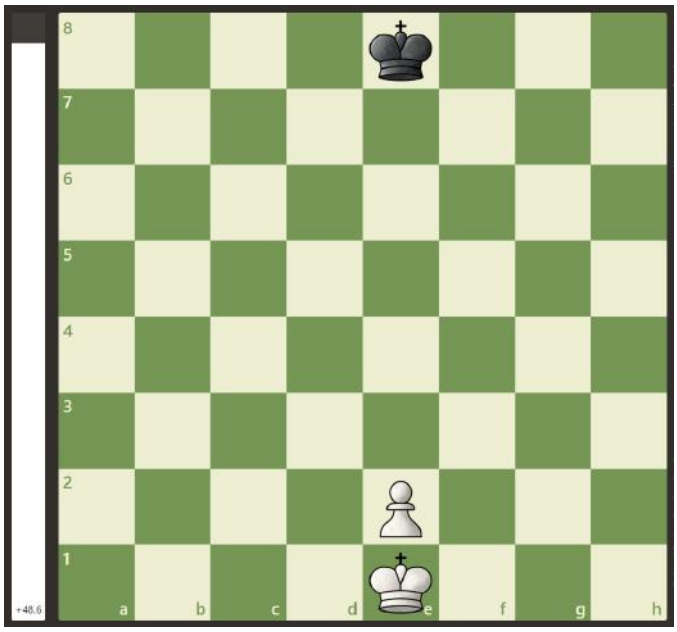
- 55% das pessoas tem entre 3 e 6 anos e são nossos jogadores iniciantes no xadrez

- 35% das pessoas tem entre 7 e 10 anos, nesses jogadores temos iniciantes e avançados.
- 10% das pessoas tem mais de 10 anos, que são nossos avaliadores com conhecimento avançado no jogo, os quais nos ajudaram com analises mais profundos sobre os agentes.

5.2 CONFIGURAÇÃO 1 - PRINCESA DE GALES

Em nossos primeiros experimentos, utilizamos a configuração do jogo chamada Princesa de Gales (Figura 14) como estado inicial.

Figura 14 – Configuração Princesa de Gales



Configuração do jogo Princesa de Gales com rei branco em e1, peão branco em e2 e rei preto em e8.

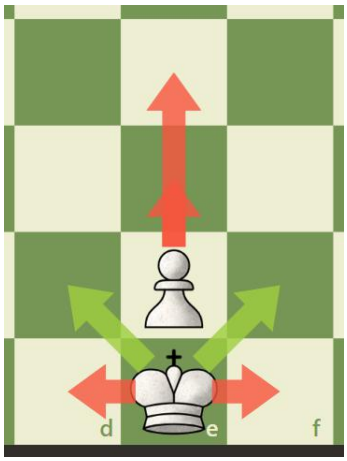
Essa configuração representa uma posição teórica onde toda pessoa que começa no xadrez precisa aprender, pois ela mostra como, com uma simples vantagem de um peão, fazer a promoção do peão numa rainha e ganhar a partida. O preto por sua vez precisa fazer o impossível para empatar pois ele não conseguirá ganhar de jeito nenhum caso o branco realize as jogadas boas.

Na teoria estudada no xadrez, nessa posição o branco ganha o 100% das vezes se sabe jogar, se erra só uma vez, o preto tem a chance de empatar, e para isso mostramos uma análise no chess.com. Pelo simples fato do branco começar jogando nessa posição ele já tem uma vantagem de +48 (valor calculado pelo algoritmo do chess.com), o que significa uma vantagem de mais de 5 rainhas (cada rainha vale 9 pontos).

Assim, o branco tem 6 possibilidade de jogadas, 4 com o rei e 2 com o peão como pode ser visto na Figura 15. Duas dessas jogadas mantém a vantagem de +48 e as outras 4

levam a vantagem para +0, permitindo que o jogador de peças pretas tenha a opção de empatar instantaneamente.

Figura 15 – Possibilidade de jogadas iniciais na Princesa de Gales



Jogadas boas (verdes) e ruins (vermelhas) no começo da configuração.

A estratégia do jogador preto por sua vez, é se manter na coluna do peão para evitar que ele chegue a outra ponta do tabuleiro e seja promovido, esperando que o branco erre alguma jogada e empatar assim a partida. Nossa busca é para obter um agente branco que jogue as sequências de jogadas boas quando estiver com as peças brancas levando a vitória e quando estiver com as peças pretas, consiga o empate se o jogador branco fizer uma jogada ruim.

Nessa configuração, ainda temos a possibilidade de analisar o aprendizado de uma técnica especifica do xadrez chamada Oposição. Essa técnica deve ser utilizada pelo jogador de peças brancas para tirar o rei preto da linha do peão e permitir a promoção. A oposição consiste em relacionar a posição que os reis mantêm entre si, especialmente nos finais de partidas. Quando a distância entre os reis é de um número ímpar de casas, o jogador que jogou por último "tem a oposição", como mostrado na Figura 16.

5.2.1 Caso 1

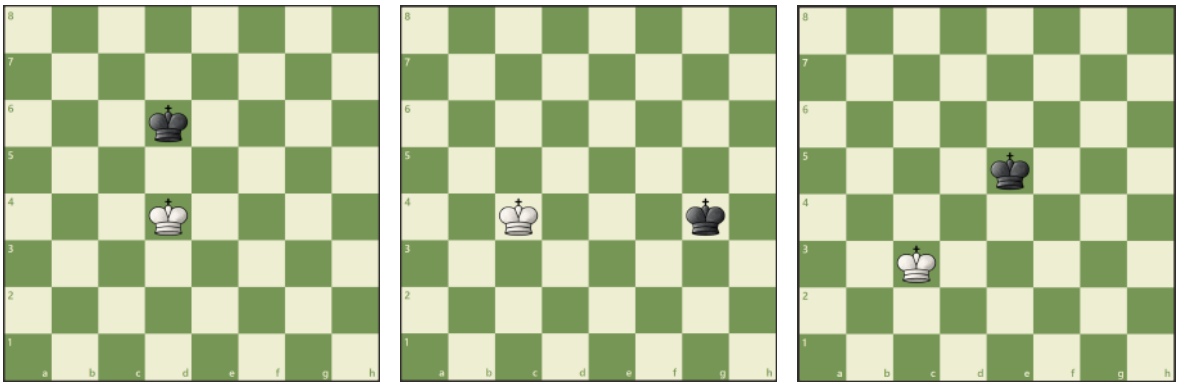
Em nossa primeira tentativa, utilizamos as seguintes listas de valores para os parâmetros:

Tabela 1 – Valores dos parâmetros na 1ª tentativa da Princesa de Gales

Games	100, 200, 500, 1.000, 2.000, 5.000, 10.000, 20.000, 50.000
StepSize	0,1, 0,2, 0,3, 0,5, 0,7
ExplorationRate	0,1, 0,2, 0,3, 0,5, 0,7

Conjuntos de valores para ExplorationRate, StepSize e Games, utilizados no primeiro experimento do trabalho.

Figura 16 – Configurações de oposição



(a) Oposição vertical. (b) Oposição horizontal. (c) Oposição diagonal.

Por ser nosso primeiro experimento no trabalho, utilizamos muitos valores diferentes. São treinados agentes utilizando combinações dos valores da tabela 1, gerando 225 agentes. Iniciamos em uma quantidade de jogos muito pequena até uma quantidade mais razoável para o aprendizado, com o objetivo de encontrar quais seriam os melhores valores. Com os resultados dessa primeira tentativa, queremos encontrar os melhores intervalos de valores para utilizar nos próximos experimentos e também mostrar que mesmo em algoritmos simples de aprendizado por reforço é necessário bastante tempo de treino.

Quando foram obtidos agentes com treinamento de 50.000 jogos, vimos como o jogador de peças brancas não escolhia as jogadas boas a partir do meio da partida e como o jogador de peças pretas não aproveitava os erros do adversário para levar o empate. Além disso, quando fizemos o teste de uma pessoa vs "Melhor Agente"(seção 5.1.3), tivemos que a pessoa ao jogar com as peças brancas ganhou facilmente (principalmente por já ter uma ideia de como são os melhores movimentos) e com as peças pretas não precisou de muito esforço para empatar, pois o agente não tinha aprendido completamente. Sendo assim, o agente treinado ainda fazia jogadas pseudo aleatórias que levavam ao empate ou derrota.

5.2.2 Caso 2

Como os resultados do caso 1 não foram satisfatórios pela avaliação do agente contra uma pessoa, passamos para uma nova tentativa adicionando novos valores nos parâmetros. Dado que o jogador treinado com 50.000 jogos ainda parecia realizar movimentos pseudo aleatórios, aumentamos a quantidade de jogos no treino. A partir do ranking obtido pelo torneio entre os agentes, vimos que as taxas de exploração menores obtiveram melhores colocações, então colocamos valores menores na taxa de exploração. Além disso, para acompanhar os valores da taxa de exploração, também incluímos valores menores no StepSize. Com isso, os valores adicionados à Tabela 1 foram:

- 100.000 e 200.000 na quantidade de partidas no treino

- 0,05 na taxa de exploração

- 0,05 no StepSize

Ao adicionar esses novos parâmetros, o tempo para treinamento aumentou consideravelmente para cerca de 30 minutos para cada agente, devido aos valores adicionados para quantidade de partidas ser muito maior do que as que estavam anteriormente. Contudo, ao final do treinamento os jogadores com 200.000 jogos no treino e baixa taxa de exploração obtiveram um bom aprendizado e conseguiram descobrir as jogadas boas para a configuração. Além disso, a média de movimentos por jogo realizado entre os agentes foi de 20.

5.2.3 Resultados quantitativos

Nesta subseção levantamos alguns resultados interessantes obtidos pelo Teste (seção 5.1.1, o agente jogando contra ele mesmo) e pelo resultado do torneio baseado em cores (seção 5.1.2, todos os jogadores de peças brancas joga contra todos os jogadores de peças pretas) realizado com os agentes obtidos no caso 2. Vamos então explorar gráficos para identificar valores mais relevantes utilizados nos parâmetros e quantidade de vitórias dos jogadores de peças brancas tanto na avaliação 1 quanto no torneio.

Nos resultados do Teste do caso 1 tivemos 16 jogadores de peças brancas, de um total de 225, que ganharam 100% dos jogos contra seu simétrico (jogador de cor oposta treinado com os mesmos valores para os parâmetros). Um ponto interessante é que após o treino, na avaliação 1, jogamos 1.000 partidas entre o jogador branco e o jogador preto e podemos levantar a hipótese de que esses 16 jogadores podem estar sempre jogando com os mesmo movimentos.

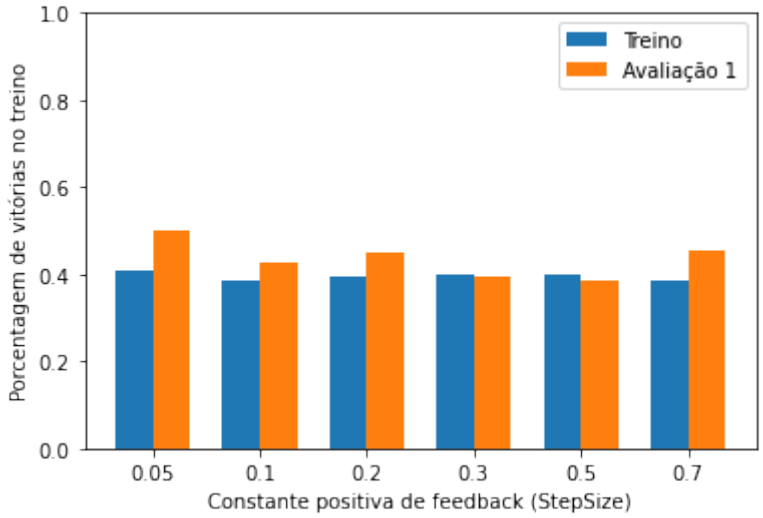
Com o surgimento dessa hipótese, em uma das execuções desse experimento, incluímos um log que contabiliza a quantidade de vezes que a sequência de jogadas é repetida. Com isso, encontramos que em algumas situações realmente os movimentos estavam sendo repetidos em todos os jogos, ou seja, era sempre a mesma partida. Porém, apenas 5 dos 16 resultados com 100% de vitórias seguiam essa hipótese e os demais mostraram pelo menos duas sequências de jogadas diferentes que o levaram a vitória. Sendo assim, concluímos que os agentes aprenderam mais de uma estratégia para chegar à vitória e que ao jogar duas vezes uma partida entre dois jogadores, eles podem utilizar estratégias diferentes em cada partida. Além disso, como esses jogadores apresentaram bons resultados levando em consideração a configuração do tabuleiro, onde esperamos que o jogador branco sempre ganha se ele realmente aprende a jogar, começamos a ter evidências de que o jogador poderia estar aprendendo.

Outra situação possível que também ocorreu foram os casos em que o jogador branco apenas perdeu. 38 jogadores brancos tiveram 100% das partidas terminadas em empate (nessa configuração, no caso de empate é contabilizado vitória para o jogador preto).

Porém, diferente do caso mostrado anteriormente, a maioria apresentou uma repetição na sequência de movimentos das partidas, ou seja, foram partidas repetidas (apenas 3 jogadores apresentaram mais de uma sequência de movimentos diferente) e concluímos que os agentes não aprenderam estratégias vencedoras jogando com as peças brancas.

Na Figura 17 mostramos a porcentagem de vitórias média dos jogadores de peças brancas na fase de treino e na avaliação 1 para cada quantidade de jogos utilizada. Nesse caso temos que apenas as alterações nos valores do StepSize não causaram grande variação na quantidade de vitórias. Com isso, podemos começar a diminuir a quantidade de valores utilizados, gerando menos agentes e resultando em um treino mais rápido.

Figura 17 – Variação do StepSize na 1ª configuração



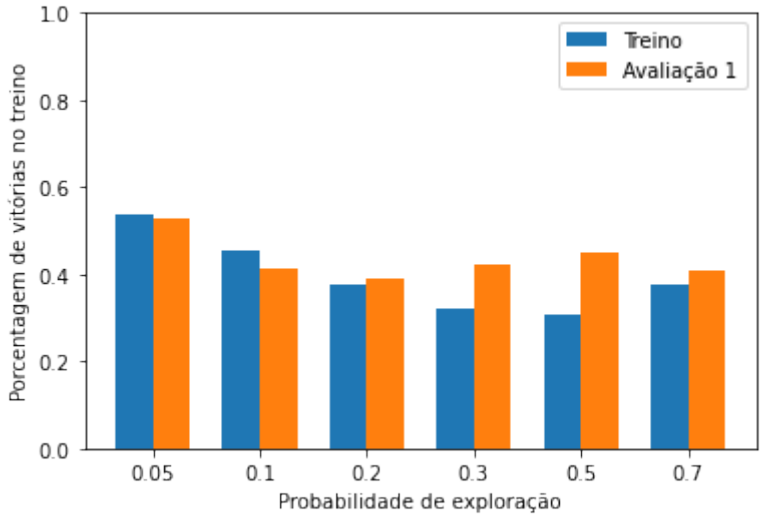
Porcentagem de vitórias conjunta dos jogadores de peças brancas para cada StepSize utilizado no treino.

Diferente do caso anterior, na Figura 18 conseguimos observar maiores variações na quantidade de vitórias dado apenas o valor utilizado na taxa de exploração. O principal impacto visível é na fase de treino, em que os agentes com menores taxas obtiveram uma maior quantidade de vitórias, enquanto valores maiores como 0,3 e 0,5 tiveram uma queda considerável.

Após as análises feitas a partir da avaliação 1, passamos a observar o torneio. Nessa fase estávamos procurando o melhor jogador de peças brancas para posteriormente realizar uma análise qualitativa (subseção 5.2.4) para determinar se foi apresentado um aprendizado significativo. Além disso, junto dos gráficos das Figuras 17 e 18, buscamos os melhores valores para os parâmetros observando os jogadores melhores ranqueados. Desse modo, para os próximos experimentos poderíamos começar a descartar alguns valores utilizados que não demonstram bons resultados.

O ranking do torneio (Tabela 2) nos mostrou que a lista de valores para os parâmetros utilizados na 1ª tentativa realmente não estavam adequados, pois a maioria dos melhores

Figura 18 – Variação da taxa de exploração na 1ª configuração



Porcentagem de vitórias conjunta dos jogadores de peças brancas para cada probabilidade de exploração.

Tabela 2 – Resultado dos melhores jogadores no torneio

Ranking	ExplorationRate	StepSize	Games	2-0	2-1	1-2	0-2
1	0,05	0,2	200.000	395	0	0	1
2	0,05	0,5	100.000	394	0	0	2
3	0,1	0,3	200.000	392	2	1	1
4	0,05	0,5	200.000	393	0	0	3
5	0,05	0,5	50.000	391	1	2	2

Ranking dos 5 melhores jogadores de peças brancas no torneio baseado em cores para a configuração Princesa de Gales após o treino da 2ª tentativa.

colocados utilizou ou uma quantidade de jogos maior (100.000 e 200.000) ou uma menor taxa de exploração (0,05). Esse resultado nos mostrou que o desejável é permitir baixas taxas de exploração por bastante tempo enquanto que o StepSize demonstrou uma boa variabilidade (situação também observada na Figura 17).

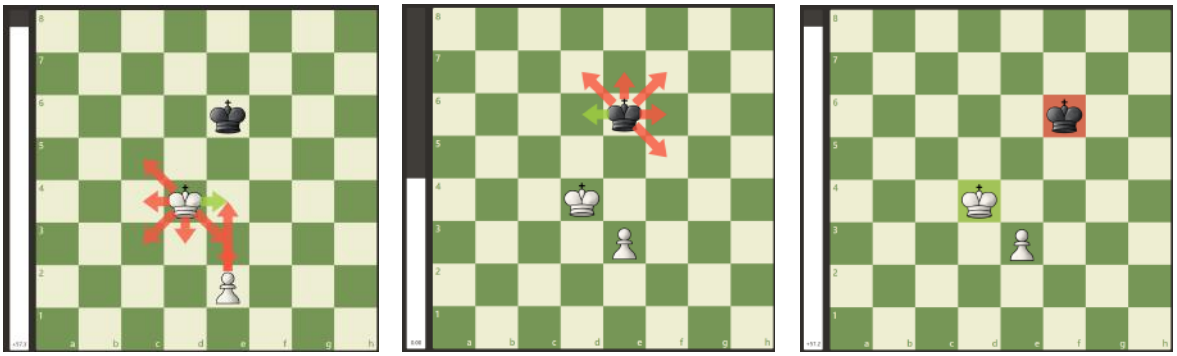
5.2.4 Resultados qualitativos

Após conseguirmos os agentes gerados na 1ª tentativa, tentamos observar como o agente que treinou com 5.000 jogos estava se comportando. Porém, as jogadas feitas pelo agente, em sua maioria, não faziam sentido e pareciam ainda ser parcialmente aleatórias. Com isso, seguimos ao melhor agente do torneio entre os agentes da 1ª tentativa e observamos um aumento considerável no aprendizado.

O melhor agente com treinamento de 50.000 jogos no lado branco conseguia progredir no começo com melhores jogadas, porém chegava a posições onde escolhia jogadas que produziam o empate e não a única jogada que o mantinha na vantagem. Além disso, no

lado de peças pretas, quando tinha o empate na posição, ele não jogava a jogada "boa" e devolvia a vantagem para o jogador de peças brancas (Figura 19).

Figura 19 – Agente semi-inteligente perdendo a vantagem durante o jogo



Sequência de jogadas do agente com treino de 50.000 jogos jogando contra si mesmo, onde inicialmente há uma vantagem para o lado preto, mas ela é perdida por uma jogada ruim.

Na Figura 19 o jogador de peças brancas tem a vantagem na posição (vantagem indicada pela barra lateral esquerda), mas ele joga e3 quando tinha que jogar Re4, fazendo que a posição permita um empate. Porém o jogador de peças pretas joga Rf6 quando tinha que jogar Rd6, devolvendo a vantagem para o jogador branco. Com isso, observamos que o jogo passa a ser um estilo de "quem errar menos ganha" numa configuração em que, teoricamente, fazer a melhor jogada é fácil.

Além disso, vimos comportamentos pouco comuns como, por exemplo, dar voltas pelo tabuleiro com os reis sem estratégias claras mostrando que o treinamento não foi suficientemente bom. Sendo assim, passamos para a análise do melhor agente do caso 2 (1º no ranking da Tabela 2). O agente que treinou com 200.000 jogos teve a melhor pontuação no torneio e tivemos um resultado completamente superior, mantendo sempre a vantagem, jogando sempre as jogadas boas e ganhando consequentemente com as peças brancas, como é visto na figura 20.

Após vermos o agente jogando contra si mesmo, colocamos o agente contra uma pessoa com conhecimento técnico da configuração. As partidas mostraram que o agente conseguiu manter a vantagem jogando com as peças brancas, como no seguinte exemplo:

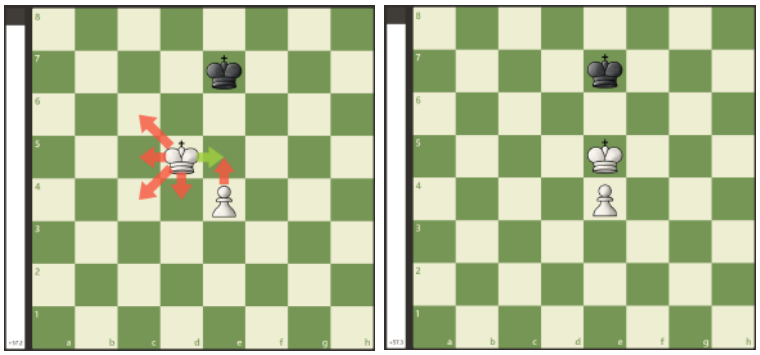
1. Rd2 Re7 2. Rd3 Re6 3. Re4 Rf6 4. Rd5 Re7 5. e3 Rd7 6. e4 Re7 7. Re5 Rf7 8. Rd6 Re8 9. e5 Rf8 10. e6 Re8 11. e7 Rf7 12. Rd7 Rf6 13. e8=Q 1-0

Agora, quando esse mesmo agente joga com as peças pretas, se a pessoa começa com uma jogada ruim (Rd1), o agente joga todas as jogadas plausíveis que mantém o empate na posição, como é mostrado no seguinte jogo:

1. Rd1 Rf7 2. Rd2 Re6 3. Rd3 Rd5 4. Rc3 Re5 5. Rc4 Re4 6. Rc3 Re5 7. Rd3 Rd5 8. Re3 Re5 9. Rf3 Rf5 10. Rg3 Re5 11. Rg4 Re4 12. Rg5 Re3 13. Rf5 Rxe2 1/2-1/2

Com essas avaliações, concluímos que o agente aprendeu a ganhar com peças brancas e aprendeu a empatar com peças pretas quando o oponente erra em algum momento. De

Figura 20 – Agente inteligente mantendo a vantagem durante o jogo

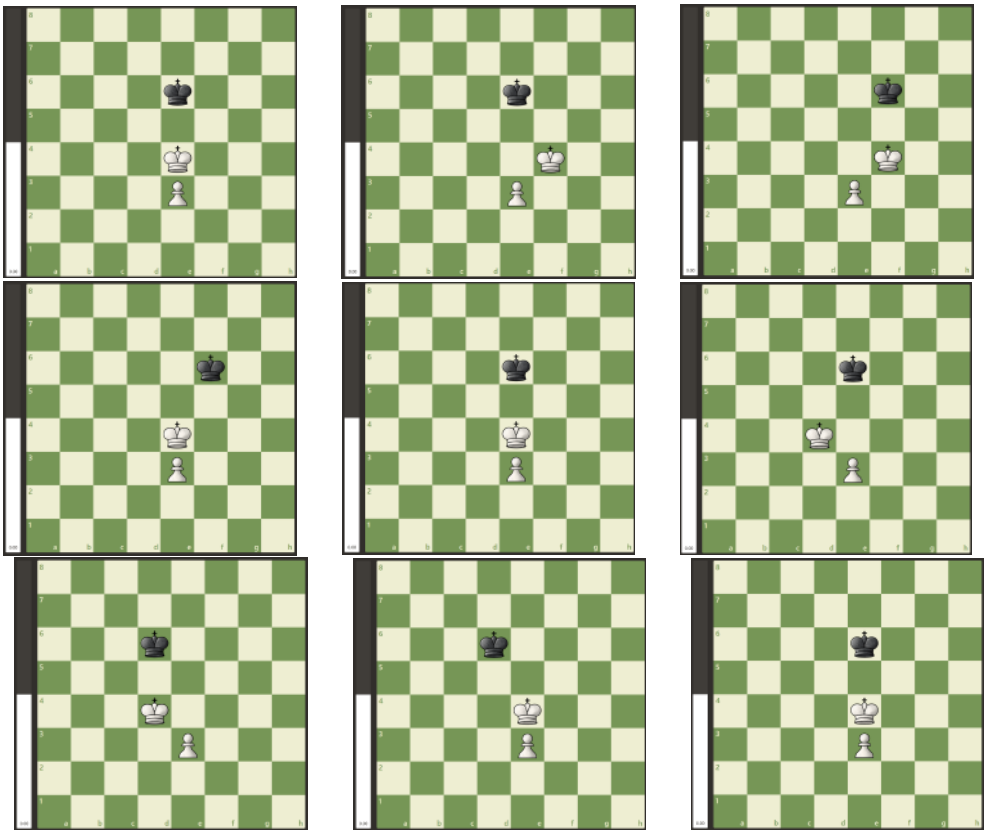


Sequência de jogadas do agente com treino de 200.000 jogos jogando contra si mesmo, onde inicialmente há uma vantagem para o lado preto e ela é mantida realizando a melhor jogada possível.

forma mais técnica, podemos dizer que o agente entendeu o conceito de oposição (Figura 21) como é mostrado no seguinte jogo:

1. Rf2 Rf7 2. Rf3 Rf6 3. Rf4 Re6 4. e3 Rf6 5. Re4 Re6 6. Rd4 Rd6 7. Re4 Re6 8. Rf4 Rf6 9. Re4 Re6 1/2-1/2

Figura 21 – Demonstração da Oposição no Princesa de Gales

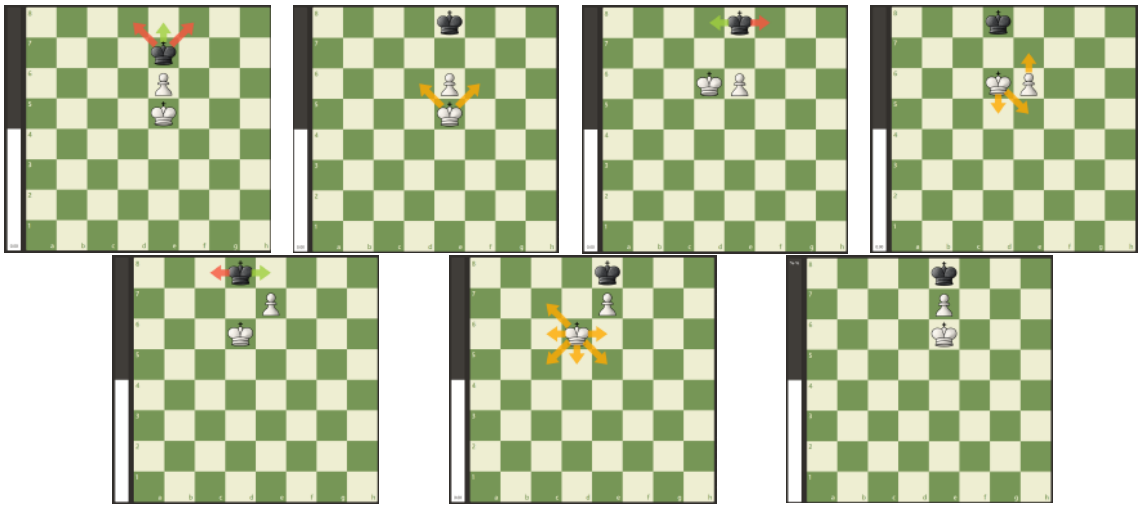


Na oposição o jogador de peças pretas se mantém alinhado com o jogador de peças brancas sem permitir o peão avançar.

Além disso, o jogador de peças pretas aprendeu o truque de afogado (Figura 22) quando o peão está chegando na sétima linha, permitindo assim o empate por falta de movimentos. É importante adicionar que nessa posição quase no final do jogo, uma jogada do jogador de peças pretas concede o empate e a outra concede a derrota.

1. Rd2 Re7 2. Rd3 Rf7 3. e4 Re6 4. Rd4 Rd6 5. e5 Re6 6. Re4 Re7 7. Rd5 Rd7 8. e6 Re8 9. Rd6 Rd8 10. Re5 Re8 11. Rd5 Re7 12. Re5 Re8 13. Rd6 Rd8 14. e7 Re8 15. Re6 1/2-1/2

Figura 22 – Demonstração do Afogamento no Princesa de Gales



No afogamento, o rei de um lado fica sem possibilidade de movimento e é forçado o empate.

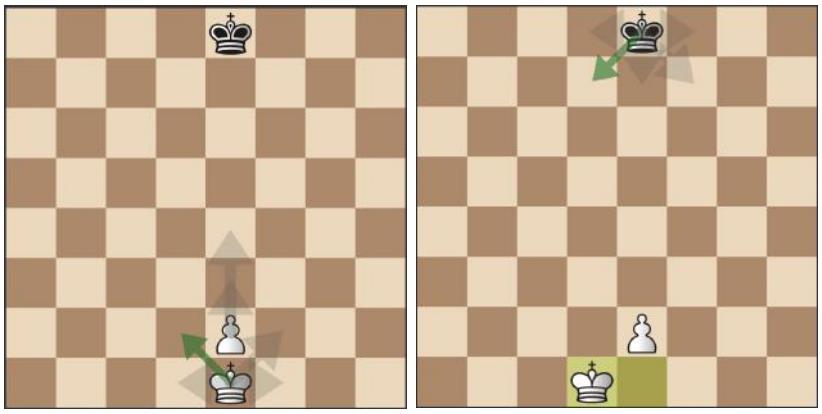
Na Figura 23 vemos a análise feita usando a ferramenta do Opening Tree depois de carregar todos os jogos feitos durante o treinamento do agente. Com essa análise identificamos que o agente jogando com as peças brancas aprende certo como começar o jogo e o preto como responder caso o branco comece com uma jogada ruim. Além disso podemos ver as partidas percorrendo todas as jogadas feitas.

5.2.5 Resultados contra Pessoas

Depois de testar o aprendizado do agente jogando contra crianças que estão começando a aprender as regras do xadrez e crianças com experiencia previa no xadrez, obtivemos os resultados da tabela 3 e os seguintes comentários dos especialistas em xadrez:

- A maquina consegue jogar muito bem com as peças brancas, seguindo as estrategias ótimas na posição e ensinando às crianças como deve ser jogado com as peças brancas para ganhar na posição.
- Com as peças pretas faz as jogadas normais para criar uma defesa, que ajuda no ensinamento da oposição.

Figura 23 – Avaliação no opening tree da primeira configuração



O jogador de peças brancas identifica a jogada boa no começo da partida e o jogador de peças pretas identifica a jogada de resposta boa caso o branco erre.

- A maquina sempre repete os mesmos movimentos se a criança responde com as mesmas jogadas em diferentes partidas, gerando dois partidas iguais, com isso, depois de ganhar uma partida, só basta repetir os movimentos que a maquina vai responder a mesma coisa sempre.
- Quando as crianças não conseguiram ganhar da maquina com as peças brancas, jogaram com as peças pretas para aprender como a maquina ganhava delas e assim aprenderam a estrategia para ganhar da maquina com as peças brancas

Tabela 3 – Resultados de jogos contra crianças de diferentes níveis

Nível de Jogo	Jogos ganhos com brancas	Jogos ganhos com pretas
Iniciantes	9 de 31	0 de 37
Avançados	14 de 14	0 de 13

Quantidade de jogos ganhos pelas crianças com peças brancas e pretas respectivamente

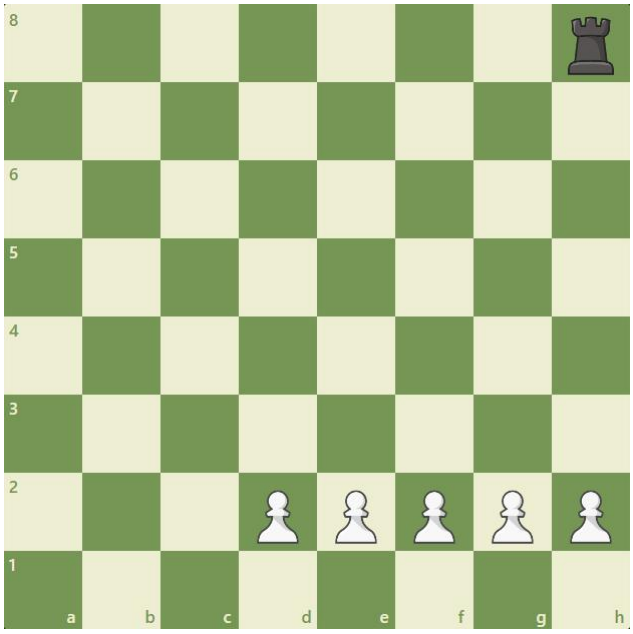
5.3 CONFIGURAÇÃO 2 - ROMA

A configuração Roma (Figura 24) possui características diferentes da anterior (Princesa de Gales). Com isso, vamos poder explorar mais o método de aprendizado que estamos utilizando e entender melhor seu comportamento e limitações.

Uma característica importante dessa configuração que faz necessário ajustes de validação do jogo tradicional é que não temos a peça do rei em nenhum dos lados. Sendo assim, a ferramenta que utilizamos para identificar vantagens para algum dos lados não funciona (o chess.com apresentado em 3.2.2). Vamos definir que um jogador vence a partida caso ele elimine todas as peças do adversário ou possua algum peão promovido à rainha. Sendo assim, tanto o lado branco quanto o lado preto podem vencer, diferente do experimento anterior onde apenas um lado poderia vencer e o outro só poderia empatar. Com isso, temos uma visão inicial de que não há grandes vantagens aparentes para um dos jogadores.

Como a configuração Roma possui mais peças e em particular uma das peças sendo a torre, com grande possibilidade de jogadas, a quantidade de estados possíveis é muito maior. Isso é importante para podermos tentar identificar impactos ou limitações no aprendizado.

Figura 24 – Configuração Roma



Configuração do jogo Roma com 5 peões brancos de d2 até h2 e 1 torre preta em h8.

Outro aspecto interessante é que o jogador branco possui apenas peões e ele deve conseguir avançá-los até realizar uma promoção. Porém, o jogador branco não pode apenas avançar um mesmo peão várias vezes seguidas pois ele ficará desprotegido e se torna um alvo fácil para a torre. Com isso, o agente terá que descobrir alguma forma

de ir avançando os peões e mantendo suas peças protegidas. Já no lado preto, como a torre possui muito mais possibilidades de movimento, ela só precisa ir removendo peões desprotegidos. Nesse caso, não temos uma estratégia clássica especifica a ser aprendida por algum dos jogadores, mas nosso objetivo é identificar se o agente consegue aprender a importância de manter sua peças protegidas e atacar peças desprotegidas com consistência durante toda a partida.

5.3.1 Caso 1

Em nossa primeira tentativa na configuração Roma, utilizamos as seguintes listas de valores para os parâmetros:

Tabela 4 – Valores dos parâmetros no 1º caso na configuração Roma

Games	10.000, 20.000, 50.000, 100.000, 200.000, 400.000, 800.000
StepSize	0,05, 0,1, 0,2, 0,3, 0,5, 0,7
ExplorationRate	0,05, 0,1, 0,2, 0,3, 0,5, 0,7

Conjuntos de valores para ExplorationRate, StepSize e Games, utilizados na primeira tentativa de aprendizado da configuração Roma.

Como já tínhamos o ambiente configurado para os valores de StepSize e taxa de exploração utilizados no caso 2 da configuração Princesa de Gales, apenas alteramos os valores na quantidade de jogos. No último experimento, vimos que mesmo em configurações simples, são necessários uma quantidade grande de jogos para que o aprendizado ocorra de forma eficiente. Logo, partimos de valores iniciais maiores até o limite de 800 mil jogos.

Assim como nos experimentos realizados na configuração anterior, ao analisar os agentes que treinaram neste caso, encontramos diversas situações em que os jogadores realizavam movimentos ruins. Com isso, concluímos da mesma forma que os agentes ainda não tinham aprendido completamente e eram necessários mais jogos no treino para melhorar o resultado dos agentes.

5.3.2 Caso 2

Como visto em 5.2.3, a variação nos valores para o StepSize não trouxe diferenças significativas na quantidade de partidas ganhas e valores menores para a taxa de exploração obtiveram os melhores resultados. Sendo assim, em nossa nova tentativa, além de incluir uma quantidade de jogos muito maior, mantivemos apenas alguns dos valores utilizados nos demais parâmetros. Este caso já começa a mostrar um pouco da dificuldade no aprendizado de configurações maiores pois a quantidade de jogos necessária é muito maior e consequentemente precisamos de mais tempos de processamento e memória.

Utilizando os novos valores:

- 5.000.000 na quantidade de partidas no treino

- 0,05, 0,1 e 0,2 na taxa de exploração

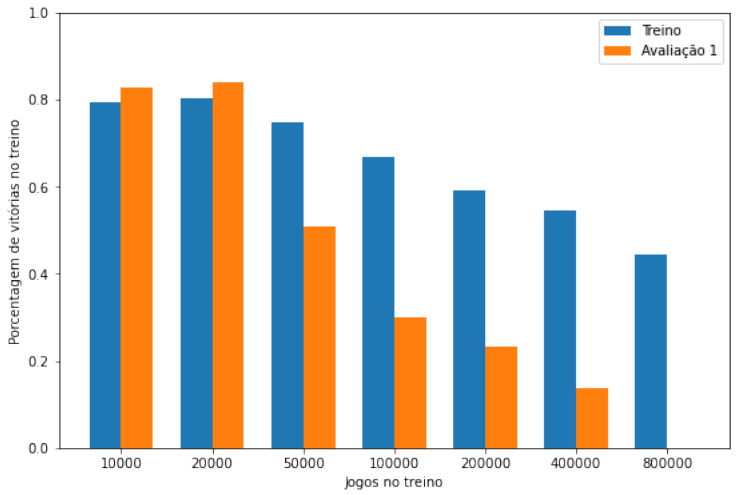
- 0,2 no StepSize

conseguimos chegar em agentes que aprenderam as melhores jogadas. Esses agentes treinaram por volta de 2 horas cada e as partidas realizadas entre si terminavam em média com 16 movimentos.

5.3.3 Resultados quantitativos

Na Figura 25 temos uma tendencia do jogador de peças brancas ganhar quando há pouco treino e, ao aumentar a quantidade de jogos, o jogador de peças pretas aprende as jogadas boas que o levam a vitória em todas as partidas da Avaliação 1. Além disso, como a porcentagem de jogos vencidos pelo jogador de peças brancas diminui consideravelmente aumentando o tempo de treino, temos que a configuração fornece uma vantagem para o lado de peças pretas.

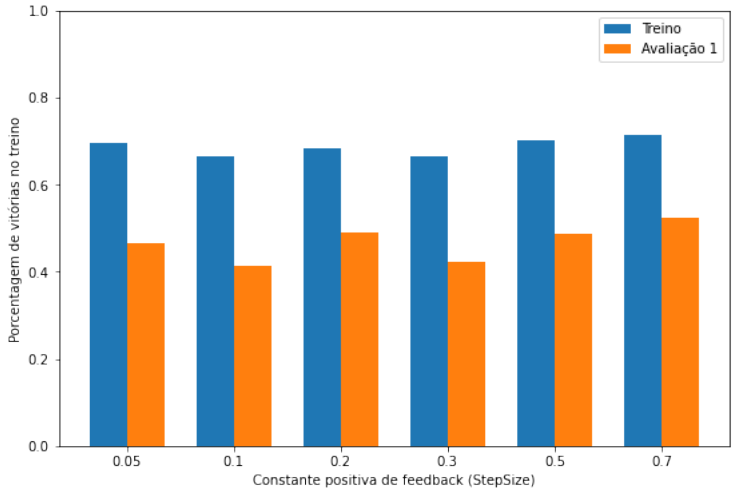
Figura 25 – Variação da quantidade de jogos no treino na 2ª configuração



Porcentagem de vitórias conjunta dos jogadores de peças brancas entre os agentes da 1ª tentativa para cada quantidade de jogos no treino.

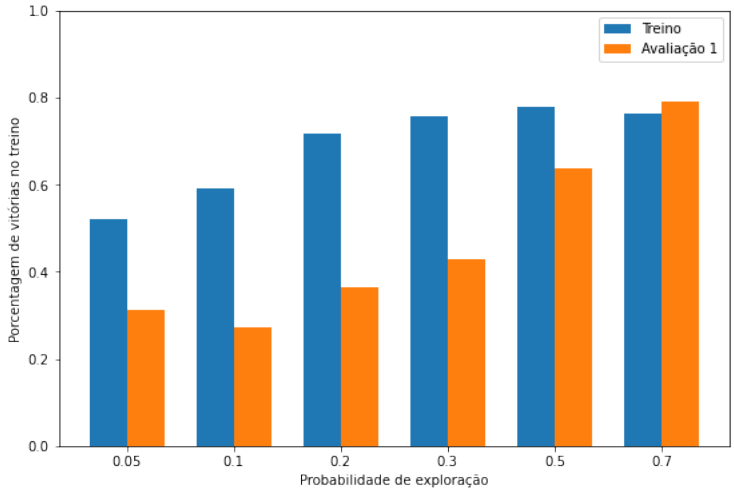
Assim como no experimento anterior, a variação no StepSize não resultou em diferenças significativas na quantidade de jogos vencidos por um dos lados no tabuleiro (Figura 26). Porém, ao variar a taxa de exploração (Figura 27), percebemos uma grande variação na quantidade de vitórias. Ao associar o gráfico da taxa de exploração com a quantidade de jogos no treino, podemos concluir que as menores taxas resultaram no melhores jogadores, pois vimos que há uma vantagem para o lado de peças pretas. Como o comportamento sobre a taxa de exploração foi similar em ambos experimentos que realizamos, podemos parar de considerar as taxas acima de 0.2 nas próximas configurações por não refletirem em um aprendizado efetivo.

Figura 26 – Variação do StepSize na 2ª configuração



Porcentagem de vitórias conjunta dos jogadores de peças brancas entre os agentes da 1ª tentativa para cada StepSize utilizado no treino.

Figura 27 – Variação da taxa de exploração na 2ª configuração



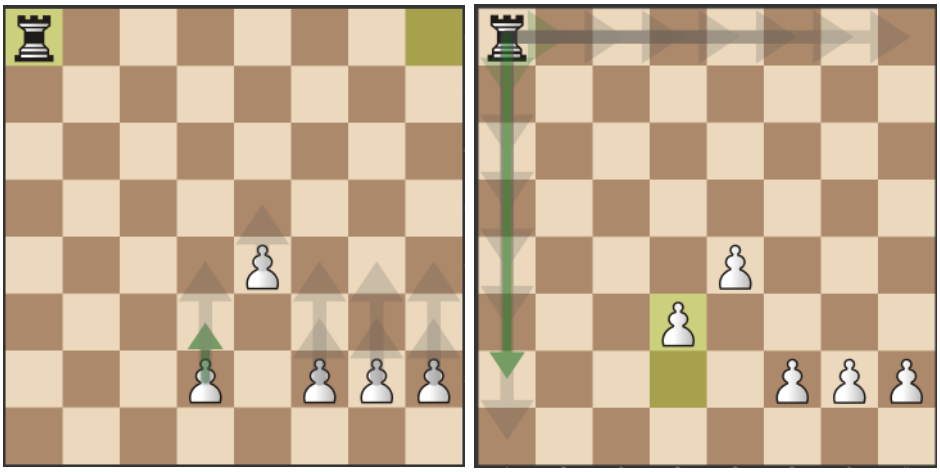
Porcentagem de vitórias conjunta dos jogadores de peças brancas entre os agentes da 1ª tentativa para cada probabilidade de exploração.

Além da análise feita sobre a variação dos parâmetros, temos que o aumento da quantidade de partidas no treino foi muito maior comparada à configuração Princesa de Gales. Verificamos a quantidade de estados (configurações do tabuleiro) explorados em cada aprendizado do melhor agente das configurações. Com 5.000.000 de jogos, o agente da configuração Roma explorou cerca de 27 vezes mais estados do que o agente da configuração Princesa de Gales treinado com 200.000 jogos. Com isso, temos que o tamanho dos arquivos para cada um desses agentes também apresentou uma diferença maior que 30 vezes. Sendo assim, percebemos que a quantidade de memória necessária para o aprendizado em configuração maiores é um limitante para o método que estamos utilizando.

5.3.4 Resultados qualitativos

Para começar as análises qualitativas primeiro nos apoiamos na ferramenta de opening tree (Figura 28) para observar a quantidade de movimentos disponíveis em cada jogada e a escolha da melhor decisão. Podemos ver que os peões vão ter mais jogadas no começo, e assim que forem ficando menos peões as jogadas vão diminuir. Porém, para o jogador de peças pretas sempre haverá mais de 10 jogadas possíveis em cada turno, o que reflete a lentidão na qual o jogador de peças pretas acha a estrategia ganhadora (Mais de 100000 jogos)

Figura 28 – Avaliação no opening tree da segunda configuração



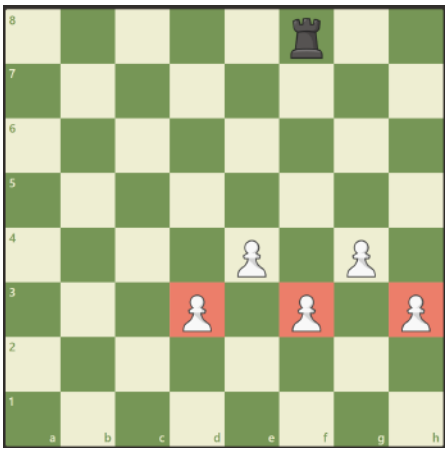
Possibilidades de movimento entre os dois jogadores numa posição.

Agora, uma análise estratégica sobre a posição nos leva a acreditar que o jogador de peças pretas e o jogador de peças brancas têm as mesmas oportunidades de ganhar a partida, pois a torre é equivalente em pontos a 5 peões. Porém, como foi visto nos resultados quantitativos, o jogador de peças pretas consegue achar as jogadas perfeitas para conseguir uma vantagem sobre os peões brancos aproveitando as seguintes debilidades da posição:

- Os peões, depois da sua primeira jogada, só podem avançar uma casa por turno, limitando a quantidade de jogadas possíveis do branco em um turno.
- A separação de ilhas de peões na posição permite à torre atacar duas ilhas ao mesmo tempo.
- Peões isolados não representam uma ameaça para a torre.

Então, em um jogo perfeito por parte dos dois jogadores, que leva a posições como na figura 29, a primeira debilidade é explorada com os peões da coluna H, F e E sem proteção de um ataque da torre, assim o branco perde o primeiro peão na jogada 6.

Figura 29 – Perdida do primeiro peão



O jogador de peças pretas identifica que o peão mais atrasado é o mais débil.

Depois disso, o jogador de peças pretas aproveita a segunda debilidade dos peões, atacando ao mesmo tempo as duas ilhas criadas anteriormente, como na Figura 30, resultando na perda do segundo peão na jogada 7 ou 8 da partida e deixando um peão isolado (terceira debilidade da posição). Com isso o jogador de peças pretas procura outra posição de vantagem como na Figura 31, forçando assim a perda do terceiro peão branco e deixando os últimos dois peões isolados.

Com os últimos dois peões das peças brancas em colunas separadas as possibilidades de vitoria do branco ficam reduzidas a 0, pois com uma posição como na figura 32, a promoção dos dois peões é evitada ao mesmo tempo, e sem importar o que jogue o branco, vão ser capturados os peões.

Assim, nosso agente mostra que aprendeu as melhores estrategias para jogar com a Torre preta forçando a vitoria da posição entre as jogadas 15 à 17 e também reforçando a ideia de que, mesmo com o valor das peças iguais para os dois jogadores, o jogador de peças pretas sempre vai ganhar se jogar perfeitamente.

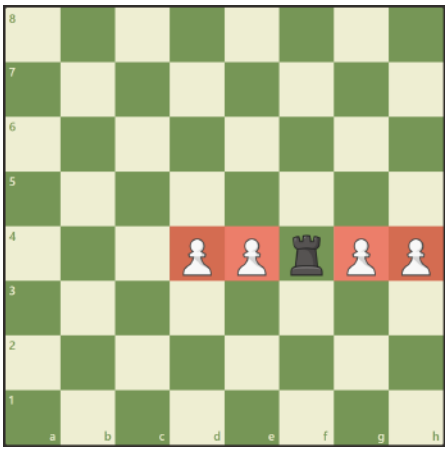
5.3.5 Resultados contra Pessoas

Depois de testar o aprendizado do agente jogando contra crianças que estão começando a aprender as regras do xadrez e crianças com experiencia previa no xadrez, obtivemos os resultados da tabela 5 e os seguintes comentários dos especialistas em xadrez:

- As peças pretas conseguem ganhar em todos os casos mostrando varias jogadas ótimas que não foram consideradas antes, com estrategias de ataque a posições fracas perfeitas para ser ensinadas no começo do aprendizado em xadrez.

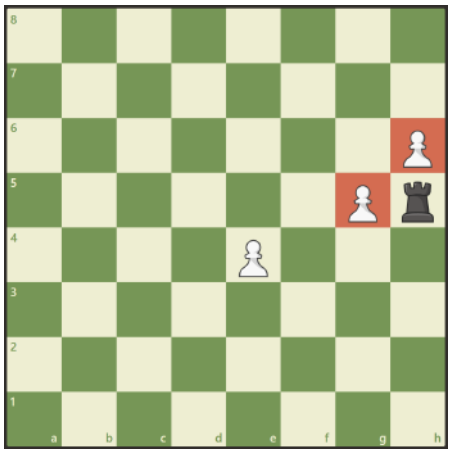
- Depois que as crianças repararam que as peças brancas nunca conseguiam ganhar, desistiram de aprender a jogar com os peões e foram atrás de entender como ganhava

Figura 30 – Ataque simultâneo da torre



O jogador de peças pretas ataca a ilha da esquerda e a ilha da direita ao mesmo tempo.

Figura 31 – Terceiro peão perdido



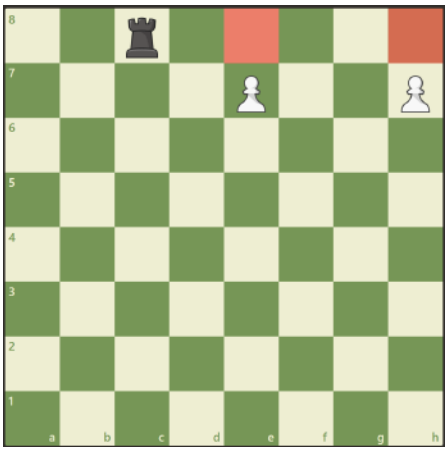
O jogador de peças pretas ataca os dois peões da ilha.

a torre.

- O professor de xadrez que utiliza esta configuração em suas aulas, tem como objetivo dar possibilidades iguais de vitórias para o jogador de peças brancas e o jogador de peças pretas, dado a equivalência de pontos e acreditando que os peões juntos conseguiam vencer a torre. Porém, vimos que o jogador de peças pretas sempre ganhava se realizasse bons movimentos. Com isso, passamos a acreditar que a torre pode chegar a valer mais que 5 peões juntos e motivamos a alteração do jogo para uma torre contra 6 peões. Além disso, esse resultado surpreendeu tanto o professor Luís quanto os jogadores avançados que tivemos contato.

- A maquina sempre repete os mesmos movimentos se a criança responde com as mesmas jogadas em diferentes partidas, gerando dois partidas iguais, com isso, depois

Figura 32 – Evitando a promoção



O jogador de peças pretas evita a promoção simultânea dos dois peões.

de ganhar uma partida, só basta repetir os movimentos que a maquina vai responder a mesma coisa sempre.

- Quando as crianças não conseguiram ganhar da maquina com as peças pretas, jogaram com as peças brancas para aprender como a maquina ganhava delas e assim aprenderam a estrategia para ganhar da maquina com as peças pretas

Tabela 5 – Resultados de jogos contra crianças de diferentes níveis

Nível de Jogo	Jogos ganhos com brancas	Jogos ganhos com pretas
Iniciantes	0 de 45	5 de 33
Avançados	0 de 21	14 de 19

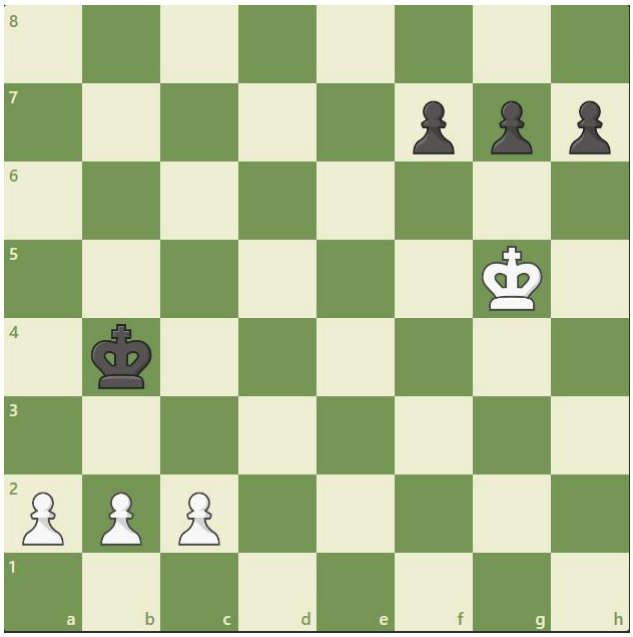
Quantidade de jogos ganhos pelas crianças com peças brancas e pretas respectivamente

5.4 CONFIGURAÇÃO 3 - HÉRCULES

Em nosso último experimento, vamos utilizar a configuração inicial Hércules (Figura 33). Nesse caso, os dois jogadores possuem as mesmas peças, que estão dispostas de forma simétrica. Pela simetria da configuração, ambos os jogadores possuem a mesma chance de vitória.

Em cada jogada, um jogador tem a possibilidade de avançar um dos peões ou movimentar seu rei. Com isso, pela quantidade total de peças, chegamos em uma configuração com mais estados possíveis comparados às configurações que vimos anteriormente. A partir dos resultados da configuração Roma, observamos que a quantidade de estados explorados foi muito maior que na Princesa de Gales e a quantidade de memória necessária foi proporcionalmente maior. Sendo assim, vamos explorar a limitação e a viabilidade do método de aprendizagem dada pela quantidade de estados que serão explorados e o resultado que teremos em um jogo inicialmente simétrico com chances de vitória iguais.

Figura 33 – Configuração Hércules



Configuração do jogo Hércules com 3 peões e 1 rei para cada lado.

Em nosso método de aprendizado, cada jogador recebe maior recompensa ao chegar em uma vitória e em nossos outros experimentos sempre temos apenas duas possibilidade de resultado (na configuração 1 temos vitória ou empate e na configuração 2 temos vitória ou derrota). Na configuração Hércules, em um teste feito com uma partida do modelo disponível no Chess.com contra si mesmo, tivemos que cada jogador não conseguiu encontrar vantagens e terminaram em empate. Esse caso é interessante pois como temos as possibilidades de vitória, derrota e empate para cada jogador onde o modelo ideal mostra que o melhor resultado é o empate, o foco da busca na vitória pode gerar dificuldades no

aprendizado. Durante o treinamento, podemos encontrar um caminho que leve a vitória contra um conjunto de jogadas mas leva a derrota contra outro conjunto. Pelo resultado do modelo do Chess.com, temos que não há um caminho para nenhum dos lados que sempre leve a vitória e sempre há um conjunto de jogadas melhores para o outro jogador. Esse comportamento que é explorado nessa configuração é utilizado para tentar identificar limitantes sobre a lógica do método de aprendizado utilizado.

5.4.1 Caso 1

Como nos experimentos anteriores, iniciamos treinando diversos agente a partir de uma tabela de valores para os parâmetros. Nosso objetivo foi identificar qual seria o resultado obtido utilizando a quantidade de jogos com valor entre os casos ótimos das outras configurações.

Tabela 6 – Valores dos parâmetros no 1º caso na configuração Hércules

Games	500.000, 1.000.000, 2.000.000
StepSize	0,2
ExplorationRate	0,05, 0,1, 0,2

Conjuntos de valores para ExplorationRate, StepSize e Games, utilizados no aprendizado da configuração Hércules.

Como esperado nenhum agente teve qualquer avanço significativo no aprendizado. Porém, como comentado anteriormente, ficou claro a limitação de memória necessária para configurações com espaço de estado grande. Tivemos que os agentes treinados com 500.000 jogos resultaram em arquivos com 600 KB, com 1.000.000 de jogos foram a 1.2 GB e com 2.000.000 a 1.8 GB.

5.4.2 Caso 2

Dado que a quantidade de memória necessária para treinar cada agente ficou muito grande, passamos a treinar um único agente com os parâmetros ótimos das configurações anteriores (0,05 de taxa de exploração e 0,2 de StepSize). Além disso, como não sabemos uma quantidade miníma de partidas para realizar o aprendizado, realizamos um treinamento gradativo. Nosso objetivo nesse caso foi uma tentativa de dar "tempo"para que o agente explorasse muitos estados e depois conseguisse ajustar o valor de cada estado para realizar boas jogadas.

Inicialmente o agente trainou com 6.000.000 de partidas e então avaliamos o resultado jogando contra o agente. Como não observamos um bom desempenho, continuamos o treinamento colocando mais 1.000.000 de partidas e realizávamos mais uma avaliação. Seguimos com esse método até obter um agente que treinou com 12.000.000 de jogos e ainda não mostrava os resultados desejáveis.

O tamanho do agente final desse caso foi cerca de 500 KB. Essa observação é interessante pois, no caso anterior, os agentes que treinaram com 0,05 de taxa de exploração, 0,2 de StepSize e apenas 500.000 jogos possuem tamanho equivalente. Com isso, levantamos a hipótese de que o uso de uma taxa de exploração nessa configuração limitava a quantidade de estados explorados e o agente poderia não estar encontrando as jogadas boas.

5.4.3 Caso 3

Para verificar a hipótese levantada, tínhamos que obter um agente que explorasse mais jogadas mas que não chegasse a um tamanho muito grande para que os valores associados aos estados pudessem convergir. Sendo assim, utilizamos uma estratégia de iniciar o treinamento com uma taxa de exploração e, depois de jogar metade das partidas, alterar essa taxa.

Realizamos esse treino com dois agentes, ambos com StepSize de 0,2. Inicialmente treinamos um agente com taxa de exploração de 0,05 e o outro com taxa de 0,2. Após treinarmos cada modelo com 5.000.000 de partidas, alteramos a taxa de exploração do modelo com 0,05 para 0,2 e o modelo com 0,2 para 0,05.

Os agentes obtidos nesse caso possuem tamanho de 1.2 GB e também não apresentaram um bom desempenho jogando contra pessoas. Sendo assim, concluímos que nossa hipótese era falsa e mesmo explorando uma boa quantidade de estados com bastante tempo, não tivemos sucesso no treinamento. Além disso, a média de movimentos em cada jogo foi de 45, muito maior que nas configurações anteriores. Como a quantidade de estados e a média de jogadas era maior que nos demais experimentos, apenas o treinamento de cada agente levou mais de 24 horas para ser finalizado.

5.4.4 Resultados quantitativos

Devido a limitação de memória e o tempo necessário para gerar cada modelo, não pudemos realizar o treinamento de uma quantidade muito grande de agentes. Sendo assim, não realizamos o torneio e avaliamos os agentes individualmente.

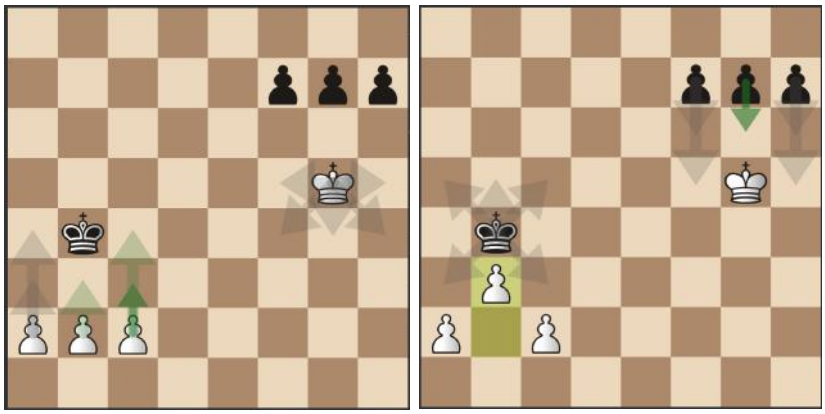
As analises individuais dos modelos se basearam em observar as jogadas realizadas durante os testes e como era o desempenho contra pessoas. Logo, nessas condições, focamos nos resultados qualitativos.

5.4.5 Resultados qualitativos

Para analisar os resultados desse experimento desde uma visão qualitativa, começamos com o uso da ferramenta de opening tree (Figura 34), entendendo assim que para cada posição vamos ter entre 8 e 14 jogadas possíveis para analisar por parte do jogador de peças brancas e do jogador de peças pretas. Adicionalmente, a média da quantidade de jogadas das partidas dessa configuração é bem maior comparada com as outras duas

configurações (Tabela 7), o que permite que a quantidade de configurações possíveis das 8 peças nesse jogo seja bem maior.

Figura 34 – Avaliação no opening tree da terceira configuração



Possibilidades de movimento entre os dois jogadores numa posição.

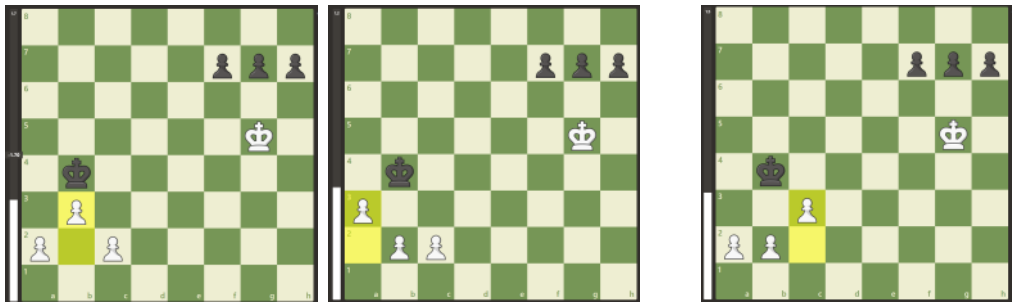
Tabela 7 – Media de jogadas por configuração

Configuração	Média de Jogadas
Princesa de Gales	20
Rome	16
Hércules	45

Quantidade média de jogadas para o fim de uma partida

Além da quantidade de jogos possíveis, temos que nessa configuração não ganha o jogador que melhor manipula as peças, mas sim o jogador que não erra o movimento dos peões. Para demonstrar isso utilizamos a ferramenta de chess.com (Figura 35), onde é exemplificado na pratica, um jogador com peças brancas que após fazer movimentações com seus respectivos peões (a3,b3 ou c3), dá uma certa vantagem que pode levar à vitoria o jogador de pecas pretas.

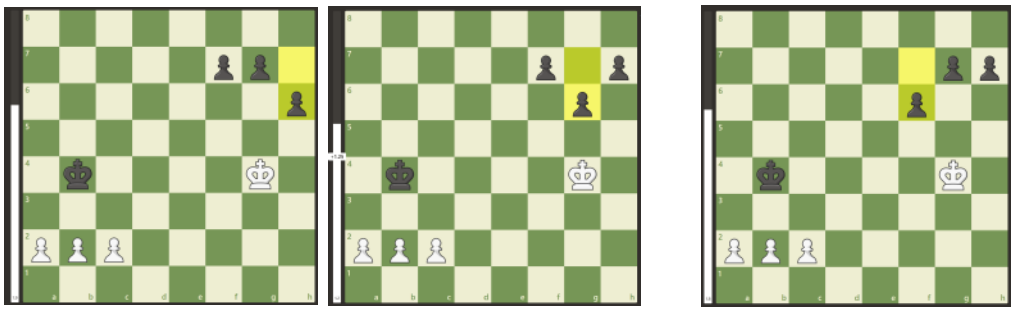
Figura 35 – Avaliação no chess.com da terceira configuração



Jogar com o peão da vantagem para o inimigo.

Tendo como base a análise do paragrafo anterior, observamos que a melhor jogada no inicio da partida é o jogador de pecas brancas mover o rei, e o jogador de peças pretas fazer o mesmo. Com isto, podemos observar que a melhor jogada é mover o rei em ambos os casos (Figura 36). Isso configura uma situação na qual ambos jogadores não devem mover seus respectivos peões para não cair em desvantagem durante a partida, impossibilitando assim o aprendizado do agente para obter uma estrategia vitoriosa que, neste caso, não vai existir.

Figura 36 – Avaliação no chess.com da terceira configuração



Jogar com o peão da vantagem para o inimigo.

Com isso as memorias do agente nessa configuração nunca mostraram aprendizados significativos que explorarem as jogadas ruins do inimigo, pois ambos os jogadores iam convergir a jogar sempre com os reis nas jogadas normais

5.4.6 Resultados contra Pessoas

Depois de testar o aprendizado do agente jogando contra crianças que estão começando a aprender as regras do xadrez e crianças com experiencia previa, obtivemos os resultados da Tabela 5 e os seguintes comentários dos especialistas em xadrez:

- O agente não aprendeu a jogar nessa configuração. Até a estrategia mais simples consegue ganhar sem dificuldade com peças brancas ou com peças pretas, perdendo a atenção das crianças rapidamente.
- Das 3 configurações, foi a de pior desempenho e virou uma posição de brinquedo para saber se a criança sabe mover as peças sem precisar de alguma estrategia.

Tabela 8 – Resultados de jogos contra crianças de diferentes níveis

Nível de Jogo	Jogos ganhos com brancas	Jogos ganhos com pretas
Iniciantes	9 de 9	5 de 5
Avançados	6 de 6	5 de 5

Quantidade de jogos ganhos pelas crianças com peças brancas e pretas respectivamente

5.4.7 Conclusão

Em nossos experimentos conseguimos verificar a eficiência do método de aprendizado quando temos o objetivo de utilizar técnicas especificas do jogo de xadrez (como a Oposição na configuração Princesa de Gales). Além disso, analisamos como os agentes se comportam para realizar jogadas que o levam a vitória e como exploram vantagens oferecidas por seus oponentes. Com a ajuda de um professor de xadrez, realizamos jogos de alunos iniciantes e avançados com os agentes para avaliar a eficiência de seu uso como um auxiliar no ensino. Por fim, observamos o impacto do tamanho do espeço de estados possíveis no jogo sobre o método de treinamento. Conseguimos agentes bons para as configurações Princesa de Gales e Roma, demonstrando a capacidade do treinamento em gerar jogadores que identificam as jogadas ótimas e quais são as sequências necessárias para chegar a vitória.

Na configuração Hércules, não conseguimos agentes inteligentes e evidenciamos limitações no método utilizado. Na Tabela 9, vemos que com o aumento da quantidade de estados explorados, o tempo e memória necessários para o treinamento passam a ser fatores importantes para determinar a viabilidade do método. Com a necessidade de uma quantidade de memória muito grande, torna-se difícil encontrar disponibilidade de máquinas robustas por tanto tempo. Sendo assim, em configurações com espaços de estados maiores, é necessário utilizar outros métodos de aprendizado que evitem realizar uma exploração profunda nos estados e que possua uma convergência mais rápida.

Tabela 9 – Resultado do treinamento das configurações

Configuração	Dificuldade	Jogos no treino	Estados	Tamanho	Tempo
Princ. Gales	fácil	200.000	35.340	1,4 KB	30 minutos
Roma	médio	2.000.000	967.016	33,7 KB	2 horas
Hércules	difícil	10.000.000	32.834.069	1,2 GB	24 horas

No endereço https://ajedrez.aplicando.com.co, usuários podem jogar contra os melhores agentes obtidos no treinamento das configurações Princesa de Gales e Roma. Os agentes estão disponíveis online para que pessoas interessadas possam interagir e avaliar os jogadores resultantes deste trabalho e comprovar a eficiência do método de aprendizado por reforço nesses casos.

6 CONCLUSÃO

Neste trabalho aplicamos o aprendizado por reforço em configurações do jogo de xadrez, avaliamos a performance dos agentes obtidos dado as características de cada configuração e disponibilizamos os jogadores autônomos para partidas contra pessoas com diferentes níveis de conhecimento do jogo. Para isso, utilizamos um método de aprendizado por reforço em jogos que é apresentado no Capítulo 4. Dado que o método se baseia em explorar o espaço de estados do jogo e atualizar a nota de cada estado conforme o resultado das partidas a partir de feedbacks para o agente, é muito importante incluir uma forma eficiente de representar o tabuleiro e suas jogadas. Esse desafio nos levou a utilização de uma tabela hashing com uma construção que facilitasse a transição de estados (ZOBRIST, 1970), como apresentado em 3.1. Além disso, utilizamos e adaptamos ferramentas para visualização das jogadas em partidas feitas pelos agentes para realizar as avaliações qualitativas e identificar o comportamento dos jogadores diante as características particulares de cada configuração.

Cada configuração escolhida possui um grau de dificuldade diferente de modo que obtemos experimentos com dificuldade fácil, média e alta. Na configuração Princesa de Gales exploramos a capacidade do agente aprender a aplicação do conceito de oposição. Na configuração Roma colocamos ambos os jogadores com possibilidade de vitória para observar como seriam feitos os ataques e defesas das peças. Por fim, na configuração Hércules, com maior nível de dificuldade, temos lados simétricos de modo que não há jogadas vencedoras se nenhum jogador realizar um movimento ruim e portanto deveríamos obter um empate.

Com o método de aprendizado e os agentes treinados no experimentos, pudemos verificar a eficiência do treinamento e avaliar quais foram as respostas para as peculiaridades de cada configuração. Além disso, com o auxilio de jogadores experientes e iniciantes, foram levantados opiniões sobre o potencial do uso dos agentes jogando contra pessoas.

Conseguimos ótimos resultados nas configurações Princesa de Gales e Roma. Porém, no desenvolvimento dos experimentos encontramos limitações de memória e tempo baseado no tamanho do espaço de estados do jogo. Como resultado da limitação, tivemos agentes salvos em arquivos ocupando mais de 1 GB de memória após dezenas de horas de treinamento. Além da quantidade de memória tornar o processo de aprendizado desafiador, impossibilita a disponibilização do agente online para jogar contra demais usuários (caso da configuração Hércules). Fora as limitações tecnológicas associadas ao aprendizado, em nosso último experimento não conseguimos agentes que realizassem boas jogadas e fossem capazes de vencer uma pessoa.

Devido ao resultado encontrado no treinamento a partir da configuração Hércules, deixamos como trabalhos futuros a tentativa de ajuste dos valores dos resultados das par-

tidas para tentar uma melhor convergência das notas dos estados. Como desconhecemos o motivo exato para o problema no aprendizado dessa configuração, novas tentativas não garantem o sucesso. Vimos que o empate é um bom resultado se ambos os jogadores não realizam jogadas ruins. Sendo assim, poderíamos tentar examinar um treinamento começando com maior valor para vitória, de modo que as boas jogadas fossem consideradas mais importantes e depois que temos algumas modificações nos valores estabelecido para os estados, invertemos a importância da vitória e do empate para dar maior foco ao empate no caso em que ambos os jogadores não realizarem jogadas ruins (empate $=$ 1, vitoria $<$ 1). Além disso, podemos verificar métodos de redução do espaço de estados para que não seja necessário armazenar todos os estados possíveis e suas respectivas notas na memória durante todo o treinamento e posteriormente ao salvar o agente. Existem muitos estados que são visitados poucas vezes em tentativas de exploração e que não são relevantes.

REFERÊNCIAS

AREL, I. et al. Reinforcement learning-based multi-agent system for network traffic signal control. **IET Intelligent Transport Systems**, IET, v. 4, n. 2, p. 128–135, 2010.

CAMPBELL, M.; JR, A. J. H.; HSU, F.-h. Deep blue. **Artificial intelligence**, Elsevier, v. 134, n. 1-2, p. 57–83, 2002.

HSU, F.-h.; CAMPBELL, M. S.; JR, A. J. H. Deep blue system overview. In: **Proceedings of the 9th international conference on Supercomputing**. [S.l.: s.n.], 1995. p. 240–244.

KOBER, J.; BAGNELL, J. A.; PETERS, J. Reinforcement learning in robotics: A survey. **The International Journal of Robotics Research**, SAGE Publications Sage UK: London, England, v. 32, n. 11, p. 1238–1274, 2013.

MORALES, E. F.; ZARAGOZA, J. H. An introduction to reinforcement learning. In: **Decision Theory Models for Applications in Artificial Intelligence: Concepts and Solutions**. [S.l.]: IGI Global, 2012. p. 63–80.

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.

SUTTON, R. S.; BARTO, A. G. **Reinforcement Learning**. 2. ed. Cambridge: Bradford Book, 2018.

ZHENG, G. et al. Drn: A deep reinforcement learning framework for news recommendation. In: **Proceedings of the 2018 World Wide Web Conference**. [S.l.: s.n.], 2018. p. 167–176.

ZOBRIST, A. L. **A new hashing method with application for game playing**. [S.l.], 1970.