DAVI HAGAP EMANUEL DA SILVA

NOVA ABORDAGEM PARA PREVISÃO DE ERROS EM PARTIDAS DE XADREZ

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae*.

Orientador: Giovanni Ventorim Comarela

VIÇOSA - MINAS GERAIS
2022

Ficha catalográfica elaborada pela Biblioteca Central da Universidade Federal de Viçosa - Campus Viçosa

T
S586n
2022

Silva, Davi Hagap Emanuel da, 1997-
Nova abordagem para previsão de erros em partidas de xadrez / Davi Hagap Emanuel da Silva. – Viçosa, MG, 2022.
1 dissertação eletrônica (54 f.): il. (algumas color.).

Orientador: Giovanni Ventorim Comarela.
Dissertação (mestrado) - Universidade Federal de Viçosa, Departamento de Informática, 2022.
Referências bibliográficas: f. 53-54.
DOI: https://doi.org/10.47328/ufvbbt.2022.261
Modo de acesso: World Wide Web.

1. Xadrez - Jogos para computador. 2. Erros. 3. Teoria dos grafos. I. Comarela, Giovanni Ventorim. II. Universidade Federal de Viçosa. Departamento de Informática. Programa de Pós-Graduação em Ciência da Computação. III. Título.

CDD 22. ed. 794.172

Bibliotecário(a) responsável: Alice Regina Pinto CRB6 2523

DAVI HAGAP EMANUEL DA SILVA

NOVA ABORDAGEM PARA PREVISÃO DE ERROS EM PARTIDAS DE XADREZ

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae*.

APROVADA: 23 de fevereiro de 2022.

Assentimento:

Davi Hagap Emanuel da Silva
Autor

Giovanni Ventorim Comarela
Orientador

Ao meu futuro eu, que vai me agradecer por tudo isso.

AGRADECIMENTOS

Agradeço as pessoas que fizeram parte dessa conquista e que me apoiaram no decorrer do mestrado. Um agradecimento especial ao meu orientador, Giovanni, que teve compreensão para entender os momentos difíceis, e também conhecimento e sabedoria, necessários para me guiar por essa jornada. À minha namorada, Laura Pérez, que me sempre me incentivou a continuar e que sem ela eu provavelmente não teria conseguido. Ao meu amigo, Marcus Albino, que mesmo de longe me apoiou nesse desafio e tornou o mestrado uma realidade para mim. Aos meus orientadores da graduação, Karina Dutra e Bruno Toledo que me incentivaram a continuar estudando e sempre acreditaram no meu êxito. À Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), pela concessão da bolsa de estudos.

É sempre melhor sacrificar as peças do seu adversário.

RESUMO

DA SILVA, Davi Hagap Emanuel, M.Sc., Universidade Federal de Viçosa, fevereiro de 2021. **Nova abordagem para previsão de erros em partidas de xadrez**. Orientador: Giovanni Ventorim Comarela.

O jogo de xadrez é amplamente utilizado como objeto de estudo em diversas áreas do conhecimento, incluindo a computação. Um dos problemas abordados na literatura recente é a análise de tomada de decisão em partidas de xadrez. Dentre as diferentes estratégias de representação de posições do tabuleiro adotadas nos trabalhos analisados, nenhuma utilizou a representação através de grafos no problema de previsão de erros. O objetivo desse trabalho é desenvolver modelos de aprendizado capazes de prever se um jogador cometerá ou não um erro em uma posição de tabuleiro. Para tal, são propostos atributos baseados em grafos capazes de representar conceitos do jogo de xadrez de forma eficiente, atributos extraídos de metadados das partidas também são propostos. A qualidade dos atributos propostos foi validada através de testes empíricos, da condução de análises exploratórias, e do treinamento de modelos de previsão que utilizam tais atributos como entrada no processo de treinamento. Os resultados dos modelos treinados também foram comparados com resultados obtidos através de modelos de referência, propostos por um trabalho da literatura recente. A comparação mostrou melhoria geral na acurácia obtida pelos modelos treinados com os atributos desenvolvidos, validando a estratégia proposta de utilização de grafos para representação de posições de xadrez.

Palavras-chave: Xadrez. Predição de erros. Grafos.

ABSTRACT

DA SILVA, Davi Hagap Emanuel, M.Sc., Universidade Federal de Viçosa, February, 2021. **A new approach for predicting errors in chess matches**. Advisor: Giovanni Ventorim Comarela.

Chess is widely used as an object of study in many areas of knowledge, including computing. One of the problems addressed in recent literature is decision-making analysis in chess matches. Among the different strategies of representation of board positions adopted in the works analyzed, none used the representation through graphs in the problem of prediction of errors. The objective of the present work is to develop learning models capable of predicting whether or not a player will make a mistake in a board position. For this, graph-based attributes capable of efficiently representing chess game concepts are proposed, attributes extracted from game metadata are also proposed. The quality of the proposed attributes was validated through empirical tests, the conduct of exploratory analyses, and the training of prediction models that use these attributes as input into the training process. The results of the trained models were also compared with results obtained through reference models, proposed by a recent literature study. The comparison showed an overall improvement in the accuracy obtained by the models trained with the developed attributes, validating the proposed strategy of using graphs to represent chess positions.

Keywords: Chess. Error prediction. Networks.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1. INTRODUÇÃO

O xadrez é um jogo tático e estratégico para dois jogadores que competem para capturar o rei inimigo. Apesar de muito antigo, o xadrez vem se popularizando até nos dias de hoje. Mesmo com a grande oferta de diferentes gêneros de jogos, existem diversas plataformas *online* de xadrez que recebem um grande volume de partidas. Por ser um jogo de natureza estratégica, balanceado e também por ter uma árvore de possibilidades que, apesar de finita, possui tantas ramificações que para humanos é impensável sua total compreensão, o xadrez é usado como modelo de estudo em diversos campos do conhecimento, incluindo a computação.

O desenvolvimento de *engines*[1] de computadores capazes de jogar o xadrez em patamar superior aos humanos é um dos motivos pelos quais o jogo é amplamente estudado nas áreas da computação (Lai, 2015). Existem diversas *engines* de xadrez. Em um passado recente, a *engine Stockfish*[2] exercia domínio sobre as demais, usando cálculo bruto para avaliar posições e escolher jogadas; entretanto, Silver et al. (2017) apresentaram o *AlphaZero*, uma rede neural que aprendeu o xadrez jogando consigo mesma uma grande quantidade de partidas, começando o processo de aprendizagem somente com as regras básicas do jogo. O *AlphaZero* dominou o *Stockfish* em confronto direto mostrando haver conceitos no xadrez que somente força bruta não é suficiente para aprender (Strogatz, 2018).

A medida que a tecnologia da informação progrediu e através do surgimento de novas *engines* melhores no jogo de xadrez, o desafio de superar os humanos foi alcançado e deixou de ser interessante. A primeira vez que um computador venceu um grande-mestre, maior título concedido a um profissional do jogo de xadrez, foi em 1997, quando Garry Kasparov, o então campeão e um dos melhores jogadores da história, foi derrotado pelo *DeepBlue*, um computador que avaliava 200 milhões de posições por segundo (Lai, 2015). Atualmente as *engines* tem pontuações em torno de 3500 pontos de Elo[3], enquanto o campeão mundial chega a 2850. Essa diferença é suficiente para considerar superioridade total das *engines* em uma partida contra

[1] No contexto do xadrez *online* as *Engines* são *softwares* construídos através de algoritmos de aprendizagem de máquina e que conseguem analisar posições e jogar partidas

[2] https://stockfishchess.org

[3] Sistema de ranqueamento de jogadores utilizado no xadrez

jogadores humanos (Biswas and Regan, 2015a).

A superioridade das *engines* de computador sobre os humanos é particularmente interessante se usada no contexto de análise de tomada de decisão, pois apesar de que o conhecimento de jogo dessas *engines* ainda não poder ser transmitido completamente para humanos, as avaliações providas por elas podem auxiliar no processo de extração do conhecimento do jogo.

O uso de *engines* no contexto de identificação de erros em partidas de xadrez é explorado pelas plataformas de jogo *online*, nas quais existem ambientes de análises de partidas e posições jogadas onde os jogadores podem, através do auxílio dessas *engines*, identificar os erros cometidos e tentar encontrar melhores jogadas na posição original. Cada plataforma explora de maneira diferente o conceito de erro nas partidas; algumas usam os erros cometidos pelos jogadores para comparar o desempenho deles com o desempenho de uma *engine*, e assim pontuam no final da partida a atuação de cada jogador com uma precisão que é alta quando o jogador faz as jogadas propostas pela *engine*, e baixa quando comete muitos erros. Também existem nessas plataformas ambientes de treinamento nos quais posições são apresentadas para o jogador e ele deve decidir qual é a melhor jogada naquela posição, a pontuação obtida é comparada com a solução proposta pelo computador.

1.1. Problema e hipóteses da pesquisa

Apesar de o jogo de xadrez ser bastante explorado na literatura, principalmente com o estudo e desenvolvimento das *engines* de jogo, o problema de previsão de erros foi o foco de poucos estudos, e nos trabalhos encontrados a representação das posições de xadrez se deu para considerar o tabuleiro em sua totalidade, através de tensores, ou por métricas desenvolvidas. Nesse trabalho será abordada a aplicação de atributos baseados em grafos extraídos de posições de tabuleiro, como uma nova aplicação da teoria de grafos na previsão de erros em partidas de xadrez. Levando-se em conta a relevância do estudo de análises de tomada de decisão para extração do conhecimento utilizando o xadrez como modelo de estudo, as seguintes hipóteses serão investigadas nesse trabalho:

1. H1: É possível prever com boa acurácia quando um jogador cometerá um erro em uma partida de xadrez utilizando atributos baseados em grafos?

2. H2: É possível obter melhores tempos de processamento com a utilização de atributos baseados em grafos?

Considerando-se os resultados já obtidos por (McIlroy-Young et al., 2020a) e (Anderson et al., 2017) nos estudos relacionados a previsão de erros, a hipótese H1 será

investigada como uma alternativa às estratégias já existentes, como a utilização de métricas ou a representação do tabuleiro em estruturas de dados. Além do desenvolvimento de modelos acurados na previsão de erros, a hipótese H2 também será investigada para que seja estudada a relação de custo computacional entre as estratégias presentes na literatura recente e a abordagem proposta.

1.2. Objetivo

Considerando a relevância do estudo de análise de tomada de decisão no contexto de partidas de xadrez, o objetivo principal desse trabalho é desenvolver modelos de predição utilizando atributos baseados em grafos e atributos de metadados. De modo a atingir esse objetivo, os seguintes objetivos específicos são considerados:

1. O1: Obtenção de uma base de partidas representativa para estudo.
2. O2: Gerar os grafos propostos por Farren et al. (2013).
3. O3: Extrair atributos baseados nos grafos gerados.
4. O4: Validar a qualidade dos atributos desenvolvidos no contexto de previsão de erros.
5. O5: Treinar modelos que utilizem os atributos desenvolvidos e obtenham bons resultados na previsão de erros.
6. O6: Reproduzir o trabalho de McIlroy-Young et al. (2020a) para comparação.

1.3. Relevância

Existem diferentes aplicações para o problema de predição de erros em partidas de xadrez. Uma solução para esse problema pode ser usada nas plataformas de jogo como um diferencial, com treinamentos personalizados para cada jogador. Além do valor comercial de um modelo que prevê erros de jogadores, o estudo das hipóteses formuladas e o cumprimento dos objetivos dessa pesquisa podem ser usados em outros sistemas onde a tomada de decisão também possa ser prevista e avaliada. Os atributos desenvolvidos também podem ser utilizadas em outros estudos que tem o xadrez como objeto de estudo, utilizando-as como uma representação para as posições de tabuleiros e servindo como entrada para modelos de aprendizagem de máquina com diferentes objetivos. A aplicação bem sucedida de atributos baseados em grafos também impacta o estado da arte, podendo ser escolhida como abordagem em trabalhos futuros. Este trabalho originou a publicação do artigo *A lightweight approach for predicting errors in chess matches* no evento ENIAC 2021.

1.4. Organização da Dissertação

O trabalho está organizado da seguinte forma: no Capítulo 2 são explicados os principais termos e conceitos utilizados durante o trabalho. O Capítulo 3 aborda os trabalhos relacionados à previsão de erros em partidas de xadrez e também à utilização de *engines* como ferramenta para resolver diversos tipos de problemas relacionados ao jogo. No Capítulo 4 é apresentada a forma de aquisição dos dados utilizados, bem como algumas etapas de preparação realizadas. O Capítulo 5 descreve os atributos propostos e os meios que foram extraídos, bem como a construção dos grafos utilizados. O Capítulo 6 é destinado aos processos de treinamento dos modelos utilizados no trabalho, e também a configuração e utilização desses modelos. Os resultados obtidos estão apresentados no Capítulo 7, e por fim as conclusões e sugestões de trabalhos futuros estão dispostos no Capítulo 8.

2. REFERENCIAL TEÓRICO

Neste capítulo são apresentados os aspectos fundamentais envolvidos na tarefa de predição de erros em partidas de xadrez. Primeiro será realizada na Seção 2.1 uma breve introdução as nomenclaturas e conceitos usados no âmbito do jogo, logo após, serão abordadas as principais representações computacionais no contexto do xadrez, como são representadas as posições, partidas e outras notações comuns não triviais (Seção 2.2). Em seguida, na Seção 2.3, são descritas as características da análise de erros através de *engines*. Na mesma seção é normalizada a definição de erro adotada neste trabalho. Logo após, são descritos na Seção 2.4 os modelos de aprendizagem utilizados neste trabalho, e por fim as métricas adotadas para os resultados obtidos são apresentadas na Seção 2.5.

2.1. Xadrez *online*

Atualmente existem diversas plataformas de xadrez *online* disponíveis como o *Lichess*[1] e o *Chess.com*[2]. Nelas é possível encontrar jogadores de mesmo nível além de jogar nos diversos controles de tempo disponíveis. Nesta seção serão descritos os elementos mais importantes do jogo para a compreensão deste trabalho.

2.1.1. Elo

Os níveis dos jogadores nas plataformas *online* são mensurados através do sistema Elo (Glickman and Jones, 1999). Nesse sistema, um número é atribuído a uma pessoa para representar o quão forte ela é no jogo, de forma com que se duas pessoas com Elos muito diferentes se enfrentam, a que tem maior pontuação tem uma probabilidade muito maior de vitória (Glickman and Jones, 1999). Esse sistema de ranqueamento de jogadores é utilizado pela Federação Internacional de Xadrex (FIDE) em competições oficiais e também nas plataformas de xadrez *online* para ranquear jogadores amadores.

Nas plataformas *online*, os jogadores se cadastram e recebem uma pontuação inicial, de 800 pontos, por exemplo. À medida que partidas são jogadas, os novos jogadores vão ganhando ou perdendo pontos de Elo e com o passar dos jogos, ficam

[1] https://lichess.org/
[2] https://www.chess.com/

posicionados em uma faixa adequada de pontuação. Nos primeiros jogos, ganha-se e perde-se uma quantidade maior de pontos, essa variação vai diminuindo até um número que normalmente é entre seis e oito pontos de Elo ganhos ou perdidos por partida. O número de pontos ganhos/perdidos em uma partida depende de alguns fatores; o principal deles é a diferença de Elo entre os enxadristas. Se, por exemplo, um jogador de 1400 pontos ganha de um jogador com 400, ele não ganhará pontos, mas se perder, a derrota vai lhe custar de 15 a 20 pontos (Chess.com, 2021).

2.1.2. Controle de Tempo

O tempo é um fator muito importante no xadrez, e existem diversos controles de tempo disponíveis. Enquanto as partidas clássicas, nas quais cada jogador tem uma hora e quarenta minutos para realizar os lances, tem prevalência nos torneios oficiais, no xadrez amador os modos jogos mais rápidos são mais populares, como as partidas rápidas, por exemplo, que dispõe de cinco a vinte minutos no relógio, ou até mesmo *bullet*, onde cada jogador não tem mais de cinco minutos para completar a partida.

A importância do ritmo da partida é tão expressiva, que existem Elos separados para um jogador nas diferentes modalidades, alguns jogadores se dão muito bem em partidas rápidas enquanto não aproveitam o tempo extra nos ritmos mais lentos.

2.2. Representações computacionais do jogo de Xadrez

As partidas de xadrez e seus movimentos têm notações específicas nas quais os dados são organizados. Neste capítulo serão abordadas as formas de representação de partidas, movimentos e também das posições de tabuleiro.

2.2.1. Partidas e Movimentos

As partidas de xadrez são armazenadas em *Portable Game Notation* (PGN), que é uma notação de texto simples. Através da PGN é possível armazenar não apenas os movimentos que compuseram a partida, mas também metadados, como o nome dos jogadores, o torneio pelo qual ela foi realizada e os Elos dos jogadores (Brown et al., 2017). Começa-se gravando todos os metadados pertinentes da partida através de etiquetas, depois os movimentos são gravados através da *Standard Algebric Notation* (SAN).

Na SAN os movimentos são compostos por um caractere que representa a peça movida e outro que representa a casa de destino da peça. As peças no xadrez são representadas com as seguintes letras: `K` - rei (*king*), `Q` - dama (*queen*), `B` - bispo (*bishop*), `N` - cavalo (*knight*), `R` - torre (*rook*), `P` - peão (*pawn*). As casas do tabuleiro de xadrez são nomeadas por colunas, de `"a"` a `"h"`, e fileiras, de `1` a `8` (Brown et al., 2017).

Sendo assim, uma partida de xadrez pode ser representada por uma série de pares de movimentos efetuados de forma intercalada entre os jogadores, a sequência `"1.e4 e5 2.Nf3 Nc6"` por exemplo, representa a seguinte sequência de movimentos:

- `"1. e4"` - brancas jogam peão de `e2` para `e4` na primeira jogada conforme Figura 2.1. Quando não há caractere indicando o tipo da peça, significa que é um peão.

Figura 2.1: Brancas movem peão de `"e2"` para a casa de `"e4"`

- `"1. ... e5"` - em seguida, pretas jogam peão de `e7` para `e5` conforme Figura 2.2.

Figura 2.2: Pretas movem peão de `"e7"` para a casa de `"e4"`

- `"2. Nf3"` - no segundo movimento, as brancas desenvolvem o cavalo de `g1` movendo-o para `f3` (Figura 2.3).
- `"2. ... Nc6"` - as pretas também desenvolvem seu cavalo de `b8` para `c6` conforme Figura 2.4.

Figura 2.3: Brancas movem cavalo de "g1" para a casa de "f3"

Figura 2.4: Pretas movem cavalo de "b8" para a casa de "c6"

Através da notação SAN a sequência de movimentos da partida é então gravada no arquivo PGN depois das etiquetas de metadados, e assim é possível recriar todas as posições de tabuleiro e os movimentos realizados a partir delas.

2.2.2. Posições de tabuleiro

As posições do jogo de xadrez podem ser representadas de diversas formas, mas a mais comum é a *Forsyth-Edwards notation* (FEN). As *strings* FEN são compostas da seguinte maneira (Brown et al., 2017):

- 1 - Oito sequências de até oito caracteres que representam cada sequência, uma fileira do tabuleiro; essas sequências são separadas por barras e cada caractere representa um tipo de peça ou um número de casas vazias. As oito fileiras do tabuleiro da Figura 2.5 estão representados pela *string* `"rnbqkbnr/pppppppp/8/8/4--P3/8/PPPP1PPP/RNBQKBNR"`.

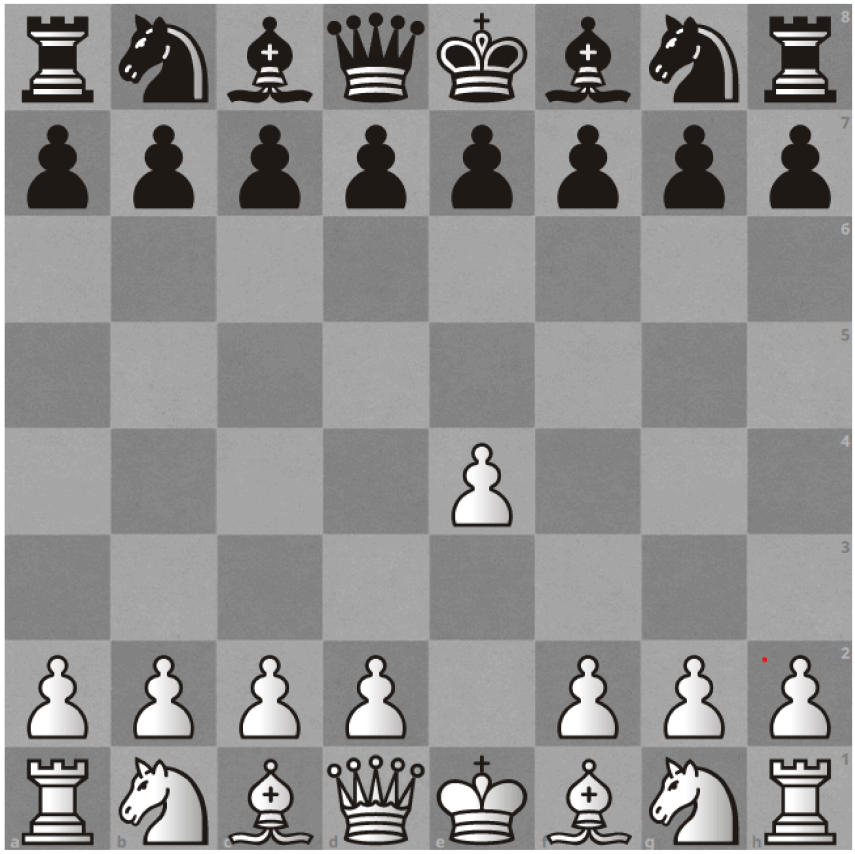

Figura 2.5: Tabuleiro para FEN `"rnbqkbnr/pppppppp/8/8/4P3/8/PPPP1PPP/RNBQKBNR"`

Os caracteres presentes nessa sequência representam os tipos de peças presentes no jogo: `P` - peão, `R` - torres, `B` - bispo, `K` - rei, `N` - cavalo, `Q` - rainha. Os caracteres que representam peças podem estar em caixa alta ou baixa representando, respectivamente, peças brancas e pretas. Sendo assim, os primeiros oito caracteres `"rnbqkbnr/..."` representam a primeira fileira do tabuleiro da Figura 2.5.

Também estão presentes na *string* FEN sequências com números que representam casas vazias no tabuleiro. A sequência `.../4P3/...`, por exemplo, representa quatro casas vazias seguidas por um peão branco e por fim três casas vazias. Essa sequência é a quinta fileira (de cima para baixo) do tabuleiro da Figura 2.5;

- 2 - O caractere `"w"` ou `"b"` representando de quem é a vez de jogar, brancas ou pretas, respectivamente. Na FEN `"...RNBQKBNR w KQkq - 0 1"` por exemplo, o próximo a jogar é o jogador com as peças brancas;

- 3 - As possibilidades de realizar o movimento `"roque"` com os caracteres `"K"`, `"Q"`, `"k"` e `"q"`. Novamente, os maiúsculos representam as possibilidades das brancas `"rocarem"` pelo lado do rei (`"K"`) ou pelo lado da rainha (`"Q"`) e os minúsculos as possibilidades das pretas. Quando nenhum jogador pode `"rocar"` para nenhum dos lados, um traço é mostrado no lugar dos caracteres; Na FEN

`"...RNBQKBNR w KQkq - 0 1"` tanto brancas quanto pretas tem a possibilidade de realizar o movimento para ambos os lados;

- 4 - Possibilidade do movimento *en-passant* com a casa no tabuleiro atrás do peão alvo ou um traço. Na FEN `"...RNBQKBNR w KQkq - 0 1"` o traço (-) significa que não há a possibilidade de efetuar o movimento especial na posição;
- 5 - Número de jogadas realizadas desde último avanço de peão ou captura de peças. Útil para saber quando um empate pode ser declarado após 50 movimentos. No nosso exemplo, como último movimento foi um avanço de peão, o penúltimo número da FEN `"...RNBQKBNR w KQkq - 0 1"` significa que não se passaram nenhum movimento sem avanço de peão ou captura de peças;
- 6 - Contador de rodadas. Esse contador se incrementa somente após jogada das pretas.

A representação em FEN da posição inicial de um jogo de xadrez clássico é `"rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR w KQkq - 0 1"`. Essa representação em *string* da posição do tabuleiro expressa a disposição das peças, além de outras informações necessárias para se reconstituir uma partida (possibilidade de roque, de quem é a vez), entretanto, essa notação não expressa as características da posição, úteis em problemas de predição. Sendo assim, outras formas de representação de posições de xadrez mais elaboradas são usadas para extrair aspectos desejados conforme o problema abordado.

2.2.3. Representação através de grafos

O uso de grafos no contexto do jogo de xadrez é particularmente útil devido à capacidade da estrutura de representar relacionamentos, e a representação das posições através de grafos é uma das soluções utilizadas na literatura recente. Essa estratégia traz vantagens e desvantagens e elas serão discutidas na Seção 3.3, com outras formas de representação, como o desenvolvimento de métricas (Anderson et al., 2017) e o uso de tensores (McIlroy-Young et al., 2020a).

2.3. Análise de erros

Para identificar os erros em partidas de xadrez, as *engines* são utilizadas de modo a se rotular as posições em posições que originam erros e posição que não originam erros. Nesta seção serão descritas como essas análises são conduzidas e como os erros são definidos.

2.3.1. Avaliação de posições

As *engines* em sua essência analisam as posições através de uma busca iterativa de profundidade na árvore de possibilidades a partir da posição em análise. Partindo da posição inicial ($d = 1$), um valor é atribuído a cada movimento legal através de uma função de avaliação, em seguida, avança-se para a profundidade $d + 1$ e repete-se o processo. Essa busca pode ser encerrada previamente por tempo de análise da *engine*, ou por profundidade alcançada (Biswas and Regan, 2015b,a).

As avaliações das posições de xadrez realizadas pelas *engines* resultam em um valor inteiro denominado centésimo de peão[3]; esse valor representa vantagem para as brancas quando positivo, vantagem das pretas quando negativo, e igualdade quando zero. Nas notações, geralmente se divide esse valor por 100; desse modo, cada unidade representa um peão de vantagem ou desvantagem. Uma avaliação de `"-3.20"`, por exemplo, representa uma vantagem expressiva para as pretas, pois seriam mais três peões de vantagem sobre o oponente, ou um bispo/cavalo que valem 3 pontos ou trezentos centésimos de peão.

No xadrez cada peça tem seu valor, esse valor é subjetivo e não tem nenhuma influência no jogo, mas é útil ao avaliar posições. A Tabela 2.1 mostra o que seria o consenso atual adotado pelos livros de xadrez (Eade, 2016). Observe que o rei não tem valor, pois sem ele o jogo acaba.

Peça	**Valor**
Dama	9
Torre	5
Bispo	3
Cavalo	3
Peão	1

Tabela 2.1: Valor das peças

Também existem posições em que o computador encontra uma sequência de xeque-mate forçado, nesses casos a avaliação é composta por uma cerquilha seguida de um número inteiro que representa em quantos movimentos o xeque-mate pode ser aplicado, se após a cerquilha houver um `"-"`, a sequência é favorável para as pretas. Uma avaliação de `"#-2"`, por exemplo, significa que as pretas podem vencer forçadamente em dois lances, entretanto isso não significa que o jogador é obrigado a fazer os melhores lances e vencer.

[3]Unidade de valor relativo das peças, comumente utilizada no xadrez.

2.3.2. Definição de erro adotada

Há diferentes maneiras de se definir um erro em uma partida de xadrez, mas a adotada nesse trabalho foi a de que um erro é um movimento ruim que leva a uma posição pior do que a anterior para o jogador que efetuou esse movimento.

Um movimento ruim também pode ser entendido de diversas formas diferentes. Uma abordagem já utilizada na literatura (McIlroy-Young et al., 2020a) é a de se obter previamente na base de dados a probabilidade de vitória de cada avaliação de *engine* e assim, definir como erro os movimentos entre as posições nas quais a diferença das probabilidades empíricas de vitória observadas nas posições caíram drasticamente, da posição originária para a posição seguinte. O limite para que a diferença seja considerada um erro também pode variar.

2.4. Modelos de Aprendizagem de Máquina

Dentre os diversos modelos utilizados para problemas de predição no campo de aprendizagem de máquina, serão descritos os dois principais utilizados nas seções seguintes.

2.4.1. *Random Forest*

Uma *Random Forest* é um conjunto de árvores de decisão em que a predição da regressão ou classificação é dada conforme o resultado de cada um dos modelos base $h(x)_1, ..., h(x)_j$, onde j é a quantidade de elementos presentes na floresta e h(x) o resultado individual de uma árvore (Cutler et al., 2012). O resultado final da floresta f(x), em caso de regressão, é dado pela média dos resultados dos regressores:

$$f(x) = \frac{1}{J} \sum_{j=1}^{J} h_j(x) \tag{2.1}$$

Já na classificação, a classe final da floresta $f(x) = y$, com $y \in Y$, é dada por votação das árvores que compõem a floresta:

$$f(x) = argmax_{y \in Y} \sum_{j=1}^{J} I(y = h_j(x)) \tag{2.2}$$

As árvores de decisão das *Random Forests* são treinadas utilizando um algoritmo recursivo de divisão dos dados de entrada, visando a cada iteração, dividir o conjunto o melhor possível, ou seja, que a cada divisão nos dados haja mais "pureza" nos novos nós que passam a constituir a árvore. O critério de pureza pode ser, por exemplo, o erro quadrático médio dos nós resultantes no caso de regressão (Cutler et al., 2012).

Considerando $y_1, ..., y_n$ como os valores da predição em um determinado nó, e $\bar{y}$ como a média desses valores, o erro médio quadrático residual pode ser obtido com:

$$Q = \frac{1}{n} \sum_{i=1}^{n} (y_i - \bar{y})^2 \tag{2.3}$$

Já nos problemas de classificação, tendo $1, ..., K$ como as classes existentes, e $\hat{p}_k$ a proporção de observações da classe k no nó, um critério típico de divisão é:

$$Q = \frac{1}{n} \sum_{k \neq k'}^{K} \hat{p}_k \hat{p}_{k'} \tag{2.4}$$

O processo de divisão considera todas as possibilidades e usa os critérios para definir o que melhor separa os dados recebidos em outros dois nós. Essa divisão iterativa é repetida até que algum critério de parada seja alcançado, como, por exemplo: parar somente quando houverem nós puros (pertencentes a uma classe apenas no caso de classificadores); até que uma quantidade mínima de dados restem; até que uma certa profundidade da árvore seja alcançada.

As predições nas árvores de decisão então são feitas através dos valores encontrados nas etapas de divisão durante o treino, do nó raiz até que alcance algum nó terminal (Cutler et al., 2012).

Cutler et al. (2012) cita outras vantagens que fazem as *Random Forests* interessante: é fácil de se implementar paralelismo; tem relativa rapidez no treinamento e predição; não tem muitos hiper-parâmetros para se configurar; não sofre muito com entradas com alta dimensionalidade.

2.4.2. *Multilayer Perceptron*

Um *Multilayer Perceptron* (MLP) é uma rede neural do tipo *feedfoward* com no mínimo três camadas, uma camada de entrada, uma de saída e pelo menos uma camada oculta. As redes neurais do tipo *feedfoward* são compostas por neurônios que se conectam aos neurônios da camada seguinte que possuem funções de ativação não lineares, essas conexões possuem pesos ajustados na etapa de treino da rede neural através de retro-propagação (Han et al., 2011). As redes neurais conseguem classificar entradas de dados não lineares, e apesar de suas camadas ocultas atribuírem-lhes a característica de pouco interpretáveis, existem estudos na literatura que buscam extrair conhecimento de redes neurais treinadas.

A topologia de um rede neural é normalmente definida empiricamente e o número de neurônios em cada uma das camadas também não é fixo, mas para a camada de saída, em problemas de classificação se usa o número de classes a serem mensuradas. As funções de ativação também podem variar de rede para rede, e dependem muito do problema e conjunto de dados.

Como o processo de se definir a topologia da rede neural é baseado em tentativa e erro, existem métricas usadas para se avaliar a rede treinada. Se ela não for considerada boa, um novo conjunto de configurações é usado e submetido aos testes.

2.5. Métricas para modelos de previsão

Para avaliação dos modelos desenvolvidos foram utilizadas métricas que serão descritas nesta seção.

2.5.1. Matriz de confusão

Nessa seção usaremos classe positiva e classe negativa na introdução de alguns conceitos a serem apresentados, incluindo a matriz de confusão. No contexto desse trabalho, onde o objetivo é a previsão de erros em partidas de xadrez, a classe positiva diz respeito as posições de tabuleiro que precedem um movimento considerado errado, enquanto a classe negativa diz respeito as posições que não precederam erros. Outros termos também são usados em diversas métricas e seus entendimentos são necessários para compreensão da matriz de confusão:

- ***True Positives*** (TP): Quantidade de instâncias com a classe positiva e previstas corretamente;
- ***False positives*** (FP): Quantidade de instâncias com a classe negativa e previstas como classe positiva;
- ***True negatives*** (TN): Quantidade de instâncias com a classe negativas e previstas corretamente;
- ***False negatives*** (FN): Quantidade de instâncias com a classe positiva e previstas como classe negativa.

A matriz de confusão é uma ferramenta que sumariza essas contagens e ajuda a analisar o quão bem o modelo previu as classes (TP e TN) e também quantos erros foram observados em suas previsões (FP e FN) (Han et al., 2011). Em um problema com duas classes de predição, a matriz de confusão pode ser construída como na Tabela 2.2.

	Previsão	
	Positivo	**Negativo**
Real Positivo	TP	FN
Real Negativo	FP	TN

Tabela 2.2: Exemplo de matriz de confusão para 2 classes

Em suma, dado m classes, a matriz de confusão é uma matriz de no mínimo ordem m e cada entrada ij indica quantas instâncias da classe i foram previstas pelo modelo como classe j. Também podem-se adicionar linhas ou colunas de totalizadores na matriz como, por exemplo, P que é a soma de TP + FN, e N que é a soma de FP + TN.

2.5.2. Métricas

Dado a matriz de confusão e os valores TP, FN, FP, TN, P e N pode-se calcular as seguintes métricas:

- **Acurácia**: É a porcentagem das instâncias classificadas corretamente:

$$acurácia = \frac{TP+TN}{P+N}$$

- **Precisão**: É a porcentagem das instâncias classificadas como positivas e são de fato positivas:

$$precisão = \frac{TP}{TP+FP}$$

  Essa métrica pode ser interpretada como quão exato o modelo é ao identificar uma instância como positiva. Essa métrica também pode ser calculada para outras classes.

- **Revocação**: É a porcentagem das instâncias positivas previstas como tal:

$$revocação = \frac{TP}{TP+FN}$$

  Essa métrica pode ser interpretada como o quão eficiente o modelo é em identificar as instâncias de classe positiva de todas que realmente são positivas. Essa métrica também pode ser calculada para outras classes.

- ***F-score***: É uma métrica que captura harmonicamente precisão e revocação:

$$revocacao = \frac{2xprecisaoxrevocacao}{precisao+revocacao}$$

  Essa métrica é útil quando as classes estão desbalanceadas, pois nesse caso a acurácia pode não ser a melhor forma de se medir o desempenho do modelo, visto que neste caso, o modelo pode possuir acurácia grande assumindo todas as entradas como pertencentes a classe predominante, comportamento que não é interessante para prever casos reais. Diferentemente da acurácia, o *F-score* é interessante em bases de dados desbalanceados, pois considera a precisão e a revocação.

3. TRABALHOS RELACIONADOS

Por se tratar de um jogo estratégico, com regras bem definidas, e que já há algum tempo pode ser satisfatoriamente aprendido pelos computadores, o xadrez é objeto de estudo em diversos trabalhos na literatura. Outra característica que facilita o estudo do jogo, principalmente em trabalhos na área da computação, é a alta disponibilidade de bases de partidas nas diversas plataformas de xadrez *online*.

Na Seção 3.1 são descritos os trabalhos relacionados que utilizaram os dados de partidas disponíveis para enriquecer o processo de extração de conhecimento do jogo de xadrez através de técnicas de mineração de dados e/ou aprendizagem de máquina.

Também há na literatura trabalhos que exploram a superioridade das *engines* com relação a jogadores humanos para estudar métricas de dificuldade e outros aspectos do jogo (Seção 3.2) aprimorando, portanto, o processo de extração de conhecimento no xadrez. Algum desses trabalhos tiveram como foco a tarefa de previsão de erros nas partidas de xadrez e são detalhados na Seção 3.3.

3.1. Extração de conhecimento em partidas de xadrez

Foram encontrados trabalhos na literatura recente que tiveram o xadrez como objeto de estudo na área de mineração de dados. Brown et al. (2017) utilizou partidas de diferentes jogadores profissionais para construir um sistema de aprendizagem de máquina de extração de conhecimento do jogo de xadrez. O resultado do trabalho foi o desenvolvimento de dois modelos; o primeiro, utilizando aprendizagem não-supervisionada, obteve sucesso na tarefa de agrupar as partidas parecidas. O objetivo desse primeiro modelo é diminuir a quantidade de partidas a serem estudadas de modo a tornar o aprendizado do jogo mais eficiente. O segundo modelo, desenvolvido com aprendizagem supervisionada, conseguiu extrair regras de associação das partidas dos grande-mestres, encontrando configurações específicas de disposições de peças e/ou movimentos que levavam os jogadores a vitória.

O trabalho de Farren et al. (2013) propôs a utilização de grafos como forma de representação das posições de xadrez. Os autores utilizaram atributos extraídos através desses grafos com o objetivo final de prever o resultado da partida. No trabalho dos autores, foi utilizada uma base de 40000 partidas, e os modelos treinados com esses atributos extraídos de grafos obtiveram acurácia de até 90% na predição do resultado

da partida.

Neste trabalho será também usada uma representação em grafos, porém diferente da apresentada pelos autores. Essa distinção se dá porque os objetivos dos trabalhos são diferentes. Para este trabalho é mais interessante capturar características gerais de cada posição do que saber que lado tem mais oportunidades de vitória. Dessa forma, os atributos extraídos dos grafos construídos diferirão, para se adaptar ao objetivo de predição de erros.

Ao se analisar os atributos desenvolvidos por Farren et al. (2013), pode-se observar a estratégia dos autores na elaboração e extração dos atributos desenvolvidos, boa parte deles é voltado a identificar vantagens de um jogador sobre o outro em uma determinada posição, e essa estratégia se mostrou eficaz na tarefa de previsão do resultado da partida. Para a tarefa de predição de erros, a vantagem ou desvantagem na partida não é necessariamente indicador de erro, por isso, os atributos serão elaboradas para abranger características do tabuleiro na totalidade, bem como das peças e suas funções no jogo, isso não exclui atributos que sinalizam vantagens dos jogadores, que também podem auxiliar nesse processo.

3.2. Análises através de *engines*

O uso de *engines* no estudo do xadrez é amplamente empregado na literatura. Biswas and Regan (2015a) utilizou as análises das *engines* por profundidade para comparar os movimentos escolhidos pelo jogador e o melhor movimento segundo a *engine*. Ao analisar as posições nas quais se exige uma profundidade de análise para se obter o melhor movimento, os autores encontraram correlação entre o Elo dos jogadores e até que profundidade eles analisavam uma posição para escolher um movimento. Quanto melhor o nível de jogador, mais profunda a análise da *engine* deve ir para refutar aquele movimento.

McIlroy-Young et al. (2020b) modificou a *engine* Maia que foi criada inicialmente para predizer movimentos e também probabilidades de vitória, para treinar modelos individuais de predição de comportamento que superaram a acurácia da *engine* original. Zegners et al. (2020) também estudou o comportamento no jogo de xadrez, mas em dados de torneios oficiais, também com o uso de *engines*. Os autores estudaram desvios no padrão de movimento dos jogadores e os resultados mostraram que nem sempre esses desvios estão associados a um desempenho pior.

Chabris (2017) sugere que através de ferramentas computacionais pode-se não só apontar os erros cometidos, mas também analisar que condições levam a eles, e isso pode ser usado para prover treinos que auxiliem no desenvolvimento do jogo. O autor sugere que estratégia pode ser levada a outros ambientes complexos, podendo ser útil na tomada de decisão e aprimoramento de habilidades ao facilitar o entendimento dos

erros e o que os provoca.

3.3. Predição de erros

Há também, na literatura, trabalhos que buscam a extração de conhecimento no jogo de xadrez através do estudo dos erros nas partidas. Anderson et al. (2017) teve como objetivo prever os erros nos finais das partidas. Os autores utilizaram somente finais de jogos com seis ou menos peças, o que possibilitou o uso de tabelas de finais de jogo. As tabelas de finais de jogo são bases de dados de todas as possíveis posições no xadrez, mas com um número limitado de peças no tabuleiro. Essa estratégia difere de estratégias apresentadas em outros trabalhos, nas quais são utilizadas avaliações de *engines* onde, apesar de não haver garantia de que essas avaliações das *engines* sejam corretas, possibilita o estudo dos erros em posições com qualquer número de peças no tabuleiro.

Anderson et al. (2017) consideraram três aspectos nos modelos desenvolvidos: tempo disponível, dificuldade da tomada de decisão e a habilidade do tomador de decisões. Esses aspectos trazem a vantagem de estarem presentes em outros cenários reais onde o estudo de tomada de decisão pode ser interessante. Esses aspectos foram explorados através do desenvolvimento de métricas que foram eficientes em prever os erros. Os autores obtiveram acurácia de 75% nos resultados obtidos.

O trabalho de McIlroy-Young et al. (2020a) abordou duas tarefas: a previsão de movimentos através da *engine* desenvolvida *MaiaChess* e a previsão de erros. Os autores utilizaram uma adaptação da *engine AlphaZero* para extrair um modelo que atingiu acurácia de 71,7% no melhor caso. Diferentemente da estratégia de (Anderson et al., 2017), o trabalho compreendeu posições a partir da décima jogada da partida, isso torna a tarefa de previsão mais difícil. Os autores também exploraram a previsão de erros em posições que se repetiram na base de dados.

Os trabalhos relacionados à previsão de erros em jogos de xadrez citados utilizam diferentes entradas de dados para os modelos de previsão escolhidos. Anderson et al. (2017) desenvolveu métricas eficientes na representação das posições, mas que tem um custo computacional alto para serem calculadas e limitam o trabalho dos autores a posições simplificadas. Já McIlroy-Young et al. (2020a) utilizaram modelos mais complexos e tensores como formas de representação das posições.

O uso dos grafos no contexto de representação de posições de xadrez como proposto por Farren et al. (2013) pode solucionar alguns dos problemas citados anteriormente. Essa estratégia não limita as análises a posições simplificadas, ou menos complexas, podendo ser utilizada em qualquer posição do jogo de xadrez. O uso dos grafos também pode solucionar o problema de se adotar a representação pura do tabuleiro como a utilizada por McIlroy-Young et al. (2020a), que gera dados comple-

xos e de alta dimensionalidade para serem usados como entrada para os modelos de precisão. Neste trabalho, as hipóteses formuladas serão investigadas de modo a saber se essa representação pode ser eficiente também na tarefa de predição de erros em partidas de xadrez.

4. CONJUNTO DE DADOS

Neste trabalho foi utilizada a base de partidas da plataforma de xadrez *online* Lichess (Lichess, 2021). Neste capítulo são apresentados os passos realizados para obtenção do conjunto final de dados utilizado. A Seção 4.1 apresenta como os dados foram obtidos e as etapas de limpeza e pré-processamento empregados nas partidas são elucidadas na Seção 4.2.

4.1. Aquisição dos dados

O *site Lichess* disponibiliza todas as partidas realizadas em sua plataforma em uma página *web*, que pode ser acessada de forma gratuita e aberta através do *Open Database* (Lichess, 2021). Esta página contém *links* onde é possível realizar o *download* das bases de partidas, que se encontram separadas por mês e ano. Foram utilizadas neste trabalho as partidas da base de dados do mês de novembro de 2019, que totalizam 40357832 partidas. O arquivo obtido possui formato PGN, e além de possuir o registro dos movimentos realizados na partida, também possui para cada partida, informações dos jogadores como Elo e os nomes de usuário, e também informações do jogo; como tempo disponível no relógio para cada jogador, a data de realização, se ele fez parte de algum torneio *online* e o resultado. Para cada movimento o arquivo também possui o tempo restante no relógio para o jogador antes da realização da jogada.

4.2. Pré-processamento

Com o objetivo de tornar o desenvolvimento do trabalho viável, algumas etapas de pré-processamento foram realizadas. Como foi obtida uma quantidade grande de partidas na etapa de aquisição, seria inviável realizar algumas das etapas de pré-processamento devido ao alto custo computacional, logo, uma etapa de amostragem foi realizada. Após amostradas, as partidas a serem utilizadas no trabalho foram avaliadas em cada posição pela *engine Stockfish* (StockFish, 2021), escolhida por ser de código aberto e por prévia familiaridade de uso do autor. Nessa fase também foram adicionadas informações a serem utilizadas posteriormente para construção de atributos. Mais detalhes dessas fases são descritos a seguir.

4.2.1. Amostragem de partidas

Para tornar possível as análises desejadas para o desenvolvimento do trabalho, as partidas da base original obtida foram amostradas aleatoriamente, e separadas em nove faixas de Elo. As bases foram separadas agrupando partidas de jogadores por faixas de Elo de 100 pontos, começando com a faixa dos 1100 pontos até 1199, em seguida 1200 até 1299 e assim por diante, até a última faixa de Elo que contempla partidas de jogadores entre 1900 e 1999 de Elo. A amostragem foi realizada de acordo com os seguintes passos:

1. **P1**: A base de partidas original foi percorrida, computando-se em um arquivo local os *indexes* de todas as partidas da base original, agregando a informação de Elo dos jogadores;

2. **P2**: O arquivo com os *indexes* das partidas foi lido e obteve-se uma lista com todos os *indexes* para cada faixa de Elo;

3. **P3**: Das nove listas obtidas, foram sorteados aleatoriamente 100000 indexes de cada;

4. **P4**: Percorreu-se novamente a base original e, através dos *indexes* sorteados, cada partida foi armazenada em uma nova base, de acordo com os Elos dos jogadores.

Esses passos geraram nove arquivos, cada um com 100000 partidas de jogadores da faixa de Elo correspondente. É importante ressaltar que no passo P1, as partidas realizadas nos controles de tempo *Bullet* e Correspondência foram ignoradas. As partidas *Bullet* são realizadas com pouco tempo disponível para que cada jogador realize suas jogadas, isso acaba levando essas partidas a ter uma grande quantidade de erros e jogadas aleatórias causadas exclusivamente pelo fator do tempo. Já as partidas por correspondência não tem nenhum tipo de controle de tempo, o que também não é desejável para este trabalho, visto que o tempo é um fator explorado nas análises de tomadas de decisão no xadrez (Anderson et al., 2017).

É possível observar que na amostragem adotada neste trabalho, podem existir partidas repetidas em diferentes faixas de Elo. Isso acontece porque um jogador de Elo entre 1590 e 1599, por exemplo, enfrentará, eventualmente, jogadores de Elo entre 1600 e 1610, e isso possibilita com que o mesmo *index* computado e lido nos passos P1 e P2, possa ser sorteado duas vezes no passo P3. Como a base original é relativamente grande, esses casos específicos não comprometem a diversidade das bases de partidas resultantes do passo P4.

4.2.2. Avaliação da *engine* e computação de dados das posições

Com as bases de partidas separadas por Elo, a avaliação das posições foi realizada. Essa etapa é imprescindível para o desenvolvimento do trabalho, pois é através dessas avaliações que os erros serão posteriormente identificados e analisados.

Percorreu-se então, cada uma das nove bases de partidas, dessa vez posição por posição, e em cada uma delas adicionou-se no comentário as seguintes informações:

1. **I1**: Avaliação da *engine*;
2. **I2**: Média de avaliação das 5 melhores variantes na profundidade 1;
3. **I3**: Tempo levado pela *engine* para avaliar a posição;
4. **I4**: Tipo de peça a ser movida na melhor jogada segundo a *engine*.

As informações I2, I3 e I4 serão melhor explicadas nas seções posteriores deste trabalho. A informação I1 é a avaliação da *engine* na posição atual do tabuleiro. Todas as avaliações foram obtidas com o *Stockfish* na versão 13 e limitado a profundidade 12 em suas análises. Esse fato possibilitou, além de avaliações mais confiáveis, a extração da informação I3, pois isolando-se a variável de profundidade atingida pela *engine*, é possível utilizar o tempo gasto por ela para realizar tal análise como métrica de complexidade da posição.

4.3. Considerações finais

As etapas de pré-processamento realizadas resultaram em nove arquivos com 100000 partidas cada, totalizando 900000 partidas. Nos arquivos finais consta cada posição de cada partida avaliada uniformemente pela *engine Stockfish* e com as informações necessárias para desenvolvimento dos próximos passos. São essas informações: avaliação da *engine*; média de avaliação das 5 melhores jogadas na profundidade 1; tempo levado pela *engine* para avaliar a posição até a profundidade determinada; símbolo que representa a peça a ser movimentada na melhor jogada segundo o computador. Além das informações de cada movimento, os dados da partida e dos jogadores foram mantidos no arquivo final, são eles: evento, ou seja, a categoria do jogo (se vale *ranking* ou se é partida de torneio, por exemplo); *link* da partida, que leva diretamente a plataforma de análise do *Lichess*; data; nomes de usuário dos jogadores; Elo dos jogadores; resultado da partida; controle de tempo; abertura jogada.

5. EXTRAÇÃO DE ATRIBUTOS

Com as bases de partidas já obtidas e as etapas de pré-processamento efetuadas, é possível extrair os atributos que serão utilizadas como entrada de dados para o modelo de previsão. Na Seção 5.1 são explicadas os atributos que foram extraídos dos metadados das partidas e em seguida, a construção dos grafos e extração dos atributos baseados neles são descritas na Seção 5.2.

5.1. atributos de metadados

Os Atributos de Metadado (AM) contém aspectos relevantes no problema de previsão de erros em partidas de xadrez. Essas informações não dizem respeito à posição das peças dispostas no tabuleiro, mas contribuem indiretamente para a caracterização da posição. Os AMs extraídos das bases de partidas nesse trabalho foram as seguintes:

1. **AM1 - avaliação profundidade 1**: Média da avaliação da *engine* das cinco melhores variantes na profundidade 1. Esse atributo tem o objetivo de capturar a dificuldade da posição, normalmente é mais fácil realizar boas jogadas no xadrez quando elas te dão uma vantagem muito grande, pois são mais fáceis de se observar. Se o jogo está equilibrado, provavelmente esse atributo vai assumir valores próximos de zero, e se não há movimentos que deem vantagem decisiva ao jogador, a posição é considerada mais difícil.

2. **AM2 - relógio**: Tempo restante (em segundos) no relógio do jogador ativo.

3. **AM3 - proporção relógio**: Porcentagem de tempo restante para o jogador ativo. É dado por:

$$proporcao\_relogio = \frac{relogio}{tempo\_inicial\_relogio}$$

4. **AM4 - horário partida**: Horário do dia em que o jogo foi realizado (em milisegundos).

5. **AM5 - *rating* inimigo**: Elo do oponente.

6. **AM6 - tempo *engine***: Tempo em segundos que a *engine* precisou para chegar a profundidade especificada durante a avaliação. Como todas as avaliações foram

realizadas no mesmo hardware e até a mesma profundidade, esse característica pode ajudar a mensurar a dificuldade da posição. Se a *engine* leva mais tempo para chegar a uma mesma profundidade em posições diferentes, esse é um indicativo de diferente complexidade das posições.

7. **AM7 - avaliação**: Avaliação da *engine* na posição atual.
8. **AM8 - avaliação -1**: Avaliação da *engine* na posição anterior.
9. **AM9 - avaliação -2**: Avaliação da *engine* duas posições atrás.
10. **AM10 - avaliação -3**: Avaliação da *engine* três posições atrás.
11. **AM11 - avaliação -4**: Avaliação da *engine* quatro posições atrás.
12. **AM12 - *rating* ativo**: Elo do jogador ativo.
13. **AM13 - contador**: Contador de movimentos da partida.

5.2. Atributos de tabuleiro

Para representação do estado do tabuleiro, a estratégia utilizada foi a representação através de grafos. Essa estratégia, proposta por Farren et al. (2013), consiste na prévia construção de diferentes grafos através da posição do jogo, e em seguida, da extração de atributos.

Neste trabalho, dois dos tipos de grafos propostos por Farren et al. (2013) foram utilizados, o denominado pelos autores como grafo de posição, neste trabalho será referido por grafo de mobilidade, e o de suporte, que será chamado de grafo de atividade. A seguir será descrito como cada um dos grafos são extraídos das posições do tabuleiro.

5.2.1. Grafo de Atividade

A partir da representação FEN de cada posição nas bases de partidas, o Grafo de Mobilidade (GA) é construído através do algoritmo iterativo descrito a seguir:

1. **P1**: Cria-se um nó para cada peça de determinada cor, além disso, obtém-se as possibilidades de captura e as peças de mesma cor que estão ao alcance dela, ou seja, as peças que estão sendo defendidas por ela;
2. **P2**: Cria-se um nó para cada peça atacada ou defendida obtida no P1. Os nós criados possuem um atributo "cor" de acordo com a cor da peça representada;

3. **P3**: Cria-se uma aresta entre o nó da peça iterada e os nós criados no P2. Essas arestas recebem um atributo "tipo" com o valor "ataque" quando as peças são de cor diferente e "defesa" caso contrário;

Cada nó do grafo direcionado resultante representa uma peça no tabuleiro, com o atributo "cor" definido e com arestas ligando aos nós das peças que são atacadas ou defendidas.

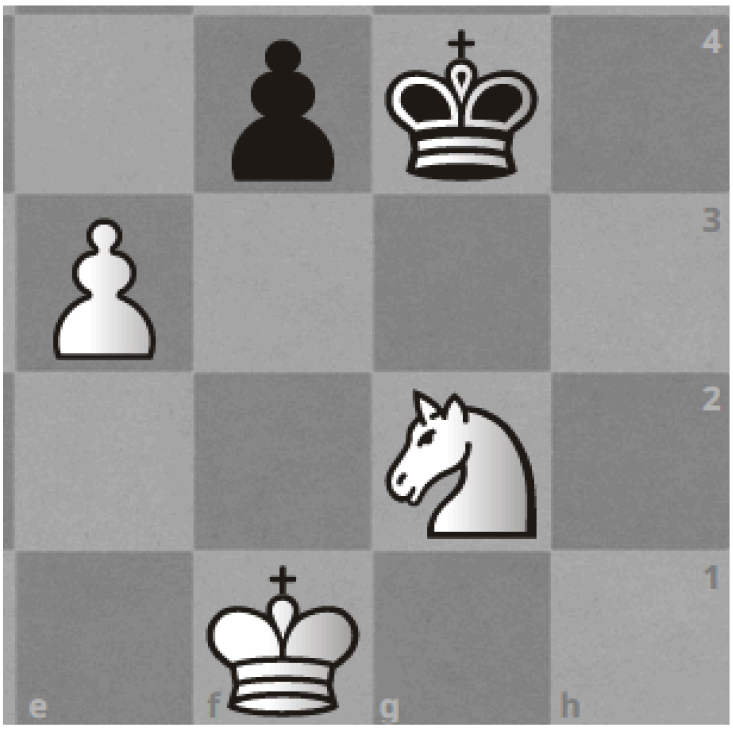

Figura 5.1: Posição hipotética, não existem peças nas casas ocultas.

A Figura 5.1 é uma posição hipotética que ajuda a esclarecer a criação dos grafos. A Figura 5.2 ilustra o GA gerado dessa posição hipotética; nela cada nó contém na primeira letra do nome, o tipo da peça que ele representa, e além do atributo "cor", as peças também podem ser distinguidas por essa primeira letra ser maiúscula (peça branca) ou não (peça preta). O resto do nome do nó representa a casa em que a peça se encontra, seguindo a nomenclatura padrão das casas no tabuleiro de xadrez, na qual as casas recebem um nome de duas letras composto pela coluna, que começa por `"a"` e vai até `"h"`, e pela fila, que começa por `"1"` e vai até `"8"`.

As arestas do grafo direcionado representado na Figura 5.2 contém duas propriedades. A propriedade "cor" é sempre a mesma do nó de origem, enquanto "tipo" varia de acordo com a cor da peça de destino, se é de mesma cor, significa que a peça de origem está defendendo uma peça aliada, recebendo então o tipo "defesa". No caso das cores entre os nós da aresta diferirem, essa aresta recebe o tipo "ataque".

O grafo gerado materializa em suas arestas e nós as relações entre as peças que nos auxiliam a entender um dos conceitos mais importantes e presentes no jogo de

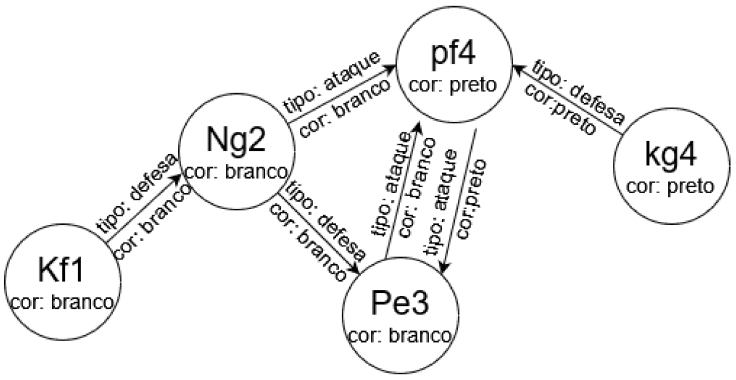

Figura 5.2: Grafo de Atividade gerado através da posição hipotética

xadrez, a captura. Na posição hipotética, por exemplo, o peão das pretas em `f4` é alvo de dois ataques, o peão em `e3` e o cavalo em `g2`, e isso é representado no grafo gerado através das arestas do tipo ataque que apontam para o nó `"pf4"`. Se nessa posição a vez é das brancas, o peão em `f4` pode ser capturado sem ônus, pois pode-se observar que enquanto como constatado, essa peça possui dois ataques, ela só possui uma peça defensora, que é o rei em `g4`. Esse exemplo ajuda a entender como em posições mais complexas essa representação através de grafos pode materializar conceitos importantes do jogo.

5.2.2. Grafo de Mobilidade

O processo de criação do Grafo de Mobilidade (GM) é semelhante ao do GA, mas o objetivo é representar as possibilidades de jogada de cada uma das peças no tabuleiro. O algoritmo para construção desse grafo é descrito a seguir:

1. **P1**: Para cada peça de determinada cor, cria-se um nó para a casa que a peça ocupa, além disso, obtém-se as possibilidades de captura ou movimento da peça;

2. **P2**: Cria-se um nó para cada casa atacada (possibilidade de captura) ou com possibilidade de movimento obtida no P1. Os nós criados possuem um atributo "cor" que é nulo em caso de possibilidade de movimento ou preto/branco dependendo da cor da peça atacada;

3. **P3**: Cria-se uma aresta entre o nó da casa de origem (nó da peça iterada) e os nós criados no P2. Essas arestas recebem um atributo "tipo" com o valor ataque;

O resultado do algoritmo descrito é um grafo direcionado onde cada nó representa uma casa no tabuleiro com um atributo "cor" que pode representar casas vazias no tabuleiro ou que alguma peça ocupa. As arestas do grafo formado sempre se originam de uma casa com "cor" definida (não nula) e representam possibilidades de movimento daquela peça na posição analisada. Note que as arestas originadas de

possibilidades de captura também representam possibilidades de movimentos, pois no xadrez, quando há uma captura, a peça que capturou ocupa o lugar da peça capturada, com a rara exceção do movimento especial *en-passant*[1].

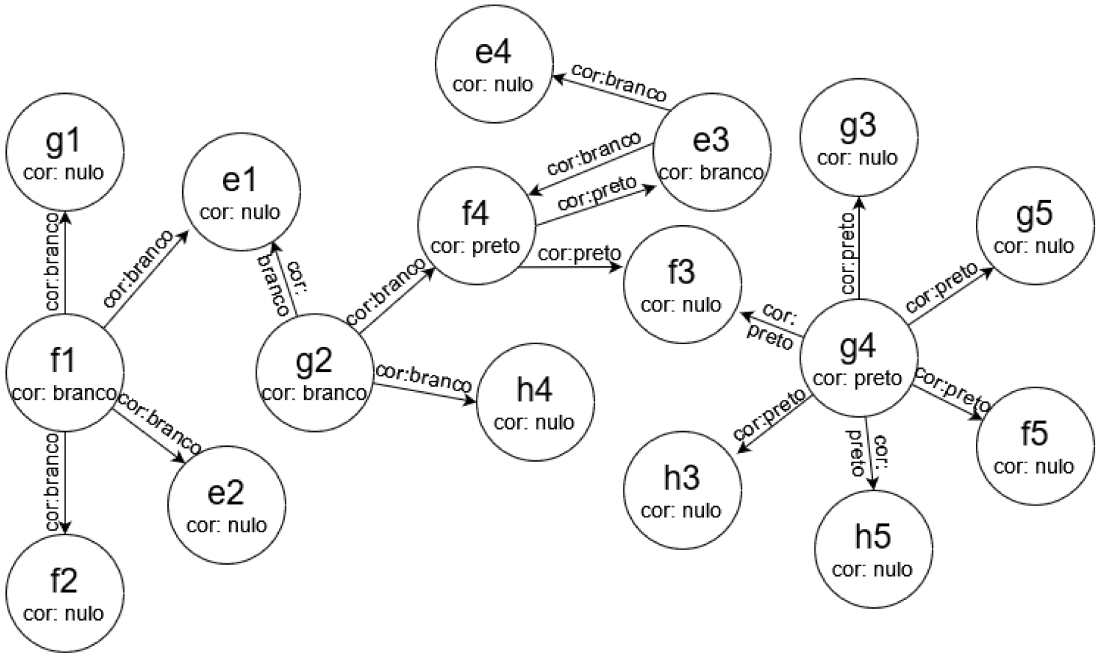

Figura 5.3: Grafo de mobilidade gerado através da posição hipotética

A Figura 5.3 exemplifica um GM gerado a partir da Figura 5.1. Nesse grafo nota-se que os nomes dos nós não se iniciam com o tipo da peça, pois diferentemente do GA, os nós no GM representam casas do tabuleiro, por isso os dois caracteres representando coluna e fila são suficientes. A propriedade "cor" dos nós pode possuir o valor "nulo" no GM e nesse caso, nenhuma peça ocupa a casa em questão. As arestas nesse grafo direcionado representam uma possibilidade de movimento e possuem duas propriedades. A propriedade "cor" será a mesma do nó de origem, assim como no GA. A propriedade "tipo" é sempre "ataque" e foi ocultada da imagem para fins de clareza. No entanto, essa propriedade foi mantida na implementação por conveniência.

Pode-se notar que o GM não possui arestas ligando casas ocupadas por peças de mesma cor, que seria o caso do rei das brancas em `f1` que defende o cavalo em `g2`, isso acontece porque o objetivo do grafo é capturar as possibilidades de jogadas, e no xadrez não se pode capturar peças de mesma cor e, portanto, o rei em `f1` não pode mover-se para a casa `g2`, portanto seu movimento a essa casa está restrito pela ocupação do cavalo. Essa propriedade do GM é complementada pelo GA com suas arestas do tipo "defesa", e possibilita representar a restrição das peças, que é um conceito importante do jogo de xadrez.

[1]Tipo de captura especial no xadrez que acontece quando um peão avança duas casas e há um peão adversário na casa adjacente à casa de destino. O peão adversário pode então capturar o peão movido e ocupar a casa que ele pulou.

Outra propriedade do GM é que ele representa as casas do tabuleiro, mas não todas. Nesse grafo somente existirão nós que representam peças ocupadas ou que podem ser ocupadas no próximo movimento de uma das peças presentes no estado atual do tabuleiro. Isso exclui casas vazias que não são "tocadas", ou seja, não podem ser destino de movimento de nenhuma peça.

5.2.3. Atributos extraídos

A criação dos GM e GA possibilitam a extração de uma série de atributos que podem auxiliar o objetivo final de predição de erros. A hipótese elaborada é de que esses atributos expressam aspectos do jogo que são difíceis de serem extraídas através da representação FEN das posições, além disso, o processo de criação desses grafos não representa complexidade tal que inviabilizaria o processamento de uma grande quantidade de partidas. Essa hipótese será retomada nos resultados desse trabalho.

O conjunto de Atributos de Tabuleiro (AT) desenvolvidos é grande, a seguir são as descritas os ATs que obtiveram melhores resultados e fizeram parte do conjunto final de atributos:

1. **AT1 - peões atacados**: Representa os ataques aos peões. É dada pela soma dos graus de entrada dos nós que representam peões no GA;

2. **AT2 - peças atacadas**: Representa os ataques em todas as peças da cor analisada e é calculada somando-se os graus de entrada de todos os nós da cor analisada no GA.

3. **AT3 - mobilidade bispos**: Soma dos graus de saída dos nós que representam bispos no GM.

4. **AT4 - peões em jogo**: Soma dos nós que representam peões no GA de ambas as cores. Representa o quão restrito o jogo está. No xadrez geralmente as peças ficam muito restritas com muitos peões no tabuleiro. Esse atributo também pode indicar o andamento do jogo (abertura, meio-jogo e final).

5. **AT5 - influência central**: Captura a influência da cor analisada nas casas centrais do tabuleiro (`d4`, `d5`, `e4`, `e5`). É a soma dos nós da cor analisada que ocupam essas casas e do número de arestas originadas de peças da cor analisada das quais o destino é um nó que ocupa qualquer uma das casas centrais no GA.

6. **AT6 - atividade cavalos**: Representa a participação dos cavalos da cor analisada no jogo. É a soma dos graus de saída dos nós que representam cavalos no GA.

7. **AT7 - defensores do rei**: Captura a proteção do rei. Soma das arestas do tipo defesa que se originam ou se destinam a peça que representa o rei no GA.

8. **AT8 - peças sobrecarregadas**: No xadrez, uma peça está sobrecarregada se ela sozinha protege mais de uma outra peça. É dada pela quantidade de nós que possuem mais de uma aresta do tipo defesa para nós que não possuem outra aresta do tipo defesa no GA.

9. **AT9 - atividade peões**: Captura a atividade dos peões. Soma dos graus de saída dos nós que representam peões no GA.

10. **AT10 - mobilidade peões**: Captura as possibilidades de jogadas de peões. Dada pela soma dos graus de saída dos nós que representam peões no GM.

11. **AT11 - atividade peças**: Captura os relacionamentos das peças da cor analisada. Soma dos graus de saída dos nós da cor analisada no GA.

12. **AT12 - mobilidade peças**: Representa as possibilidades de movimentos existentes para determinado jogador. É a soma dos graus de saída dos nós da cor analisada no GM.

13. **AT13 - peças sem defensores**: Representa o conceito de "peça no ar". É calculada realizando-se uma cópia do GA e removendo as arestas do tipo ataque e em seguida, somando-se a quantidade de nós com grau de entrada igual a zero.

14. **AT14 - atividade rainha**: Representa as relações da rainha. É a soma dos graus de saída dos nós que representam rainha no GA para a cor analisada.

15. **AT15 - mobilidade rainha**: Representa as possibilidades de movimento da(s) rainha(s). Soma dos graus de saída dos nós que representam rainhas no GM.

16. **AT16 - mobilidade torres**: Restrição de movimento com as torres. Calculada pela soma dos graus de saída dos nós que representam torres no GM.

17. **AT17 - peças sem desenvolvimento**: Uma das primeiras lições aprendidas no xadrez é o desenvolvimento das peças. Esse atributo captura o quão atrasado no desenvolvimento o jogador analisado está. É dado pela soma dos nós que representam peças que permanecem nas casas de origem no GA.

É importante ressaltar que o processo de extração de atributos do tabuleiro é centrada no jogador ativo, ou seja, na construção do GM e GA, as peças analisadas para construção dos grafos são as peças do jogador que vai realizar o movimento. Com objetivo de abranger também as peças do oponente, também foram criados os grafos GM e GA para o jogador oponente, além disso, os atributos listados acima foram extraídos desses grafos e adicionadas a lista de atributos com o prefixo "oponente". Dessa forma, a entrada de dados do modelo também contempla a disposição das peças do oponente. Testes preliminares foram realizados nos quais a entrada final

dos modelos de previsão seria a diferença entre os atributos do jogador ativo e as do jogador oponente, mas a separação deles nos sufixos "ativo" (do jogador ativo) e "oponente" (do adversário que espera o movimento) se mostrou ser uma estratégia mais eficaz. Como exceção à separação dos atributos está somente o "AT4 - peões em jogo" que diz respeito ao número total de peões no jogo, e não somente a peões de uma determinada cor.

6. TREINAMENTO DOS MODELOS

Neste capítulo são descritos os dois modelos utilizados no desenvolvimento deste trabalho. A Seção 6.1 explica como *Random Forests* foram utilizadas no processo de seleção dos atributos desenvolvidos, enquanto a Seção 6.2 descreve as configurações da rede neural utilizada na obtenção dos resultados.

6.1. Seleção de Features

Após o processo de desenvolvimento dos atributos, tornou-se necessário avaliar a sua eficiência, e para tal, foram utilizados classificadores. O processo de avaliação dos atributos desenvolvidos ocorreu de forma empírica, testando-se diferentes conjuntos de atributos em uma mesma configuração de *Random Forest*, a fim de se identificar o subconjunto de atributos com os melhores resultados. Essa estratégia foi posteriormente validada conforme descrito no próximo capítulo.

A escolha da *Random Forest* se deu devido a seu baixo custo computacional, necessário para se realizar esse tipo de avaliação empírica, além disso, as *Random Forests* também facilitam análises dos atributos usados no processo de treinamento dos modelos. A configuração utilizada foi a de 1000 árvores de decisão na floresta e uma profundidade máxima de 25 em cada uma delas. Essa configuração foi suficiente para que o classificador aprendesse com conjunto de atributos a ser testado de forma satisfatória, e, ao mesmo tempo, não demorou mais do que 3 minutos na etapa de treinamento, executada no ambiente gratuito do Google Colab.

Para o treinamento desses modelos, foram utilizados dois subconjuntos do conjunto de dados obtidos. Foram utilizadas dez mil partidas do conjunto de partidas de jogadores de Elo 1600-1699 e dez mil partidas do conjunto de partidas dos jogadores com Elo 1100-1199. A amostra de partidas utilizadas nessa etapa de seleção de atributos foi extraída da base inicial através de amostragem aleatória e foi necessária para que o tempo de treinamento das *Random Forests* fosse menor.

Os resultados dos modelos treinados com os diferentes conjuntos de atributos foram comparados e obteve-se um conjunto inicial, do qual os atributos que não se mostraram eficientes foram descartados. Esses atributos desenvolvidos que não se mostraram eficientes foram úteis para melhor entendimento de que tipo de característica era melhor aproveitada pelos modelos. Esse processo inspirou o desenvolvi-

mento de novos atributos que também foram testados através de uma nova iteração do processo de seleção descrito. Alguns deles se mostraram eficientes e passaram a compor o conjunto final de atributos.

6.2. Obtenção dos Resultados

Após a definição do conjunto final de atributos, os resultados finais descritos neste trabalho foram obtidos através do treinamento de *Multilayer Perceptrons*. Esses modelos obtiveram resultados um pouco melhores que os resultados obtidos na fase de seleção dos atributos, porém exige mais capacidade computacional para ser treinado.

A configuração da rede utilizada é descrita a seguir:

- **Camada de entrada**
- **Camada escondida**: 1024 neurônios, com *dropout* de 20%
- **Camada escondida**: 512 neurônios, com *dropout* de 20%
- **Camada escondida**: 128 neurônios, com *dropout* de 20%
- **Camada saída**: 1 neurônio.

A taxa de aprendizado inicial é de .002, mas foi utilizado decaimento programado dessa taxa, que diminui em .01 a cada 30 passos no processo de treinamento. Essa estratégia permite que o processo de treinamento do modelo seja mais rápido no início, mas que ele ainda tenha a precisão necessária para diminuir a função de perda a medida que o processo avança.

A utilização de *dropout* nas camadas foi necessária para que o modelo não sofresse super-adaptação. O valor de 20% nas camadas foi obtido empiricamente. Outra estratégia utilizada para prevenção de super-adaptação foi o uso de parada antecipada; nessa estratégia foi definida uma tolerância de 50 passos para que a acurácia no conjunto de validação melhore. Caso isso não ocorra, o processo de treinamento é interrompido precocemente e os pesos da rede são restaurados à melhor acurácia obtida.

7. RESULTADOS E DISCUSSÃO

Neste capítulo são apresentados os resultados obtidos com os atributos desenvolvidos. A Seção 7.1 apresenta as validações realizadas no conjunto de atributos extraídos dos grafos gerados. Em seguida, na Seção 7.2, são apresentados os resultados obtidos através do treinamento de modelos que utilizam os atributos gerados, e por fim, na Seção 7.3, são apresentadas algumas limitações encontradas durante a obtenção dos resultados apresentados.

7.1. Avaliação de atributos

Em conjunto com os testes empíricos descritos no capítulo anterior, foram conduzidas avaliações dos atributos desenvolvidos de modo a validar os resultados dos experimentos e atestar sua qualidade.

7.1.1. Informação Mútua

Como primeiro experimento, a Informação Mútua (MI) entre os atributos e as classes "erro" e "não erro" foi calculada. Os resultados apresentados a seguir foram obtidos utilizando-se somente a base de partidas de jogadores de Elo 1600 – 1699, mas as bases de dados de todas as faixas de Elo foram analisadas durante o desenvolvimento da pesquisa. Além disso, como o método escolhido para a obtenção desses resultados tem uma propriedade aleatória que diverge ligeiramente os valores obtidos, mesmo em execuções utilizando os mesmos dados. Devido a essa propriedade aleatória, múltiplos testes foram efetuados e os resultados apresentados a seguir são obtidos de uma dessas execuções realizadas durante o processo de avaliação dos atributos. É importante salientar também que nessa etapa foi utilizado o conjunto inicial de atributos e não o conjunto final apresentado anteriormente nas Seções 5.1 e 5.2. Os atributos que não foram apresentados anteriormente podem ser identificados pela ausência de numeração AT ou AM.

Os resultados apresentados nas Tabelas 7.1 e 7.2 evidenciam a importância dos Atributos de Metadados, mais especificamente, os que dizem respeito a avaliação da *engine* na posição e o histórico dessas avaliações. A "AM1 - avaliação profundidade 1" é um Atributo de Metadado importante segundo o que se observa na Tabela 7.1, e é outro exemplo do uso da *engine*, que nesse caso avalia as melhores posições possíveis.

Feature	Informação Mútua (%)
AM7 - avaliação	10,57
AM1 - avaliação profundidade 1	9,14
AM8 - avaliação -1	7,18
AM9 - avaliação -2	6,47
AM10 - avaliação -3	5,77
AM11 - avaliação -4	5,54
AM4 - horário partida	4,59
AM6 - tempo *engine*	2,91
AT11 - atividade peças ativo	2,79
AT12 - mobilidade peças ativo	2,76
AT11 - atividade peças oponente	2,54
AT14 - atividade rainha ativo	2,50

Tabela 7.1: Doze melhores atributos ordenados por Informação Mútua

Feature	Informação Mútua (%)
AT6 - atividade cavalos ativo	0,03
melhor movimento rainha	0,05
torres atacadas ativo	0,06
rei atacado oponente	0,10
torres atacada oponente	0,10
AT13 - peças inimigas sem defensores	0,10
peões isolados oponente	0,10
bispos atacados oponente	0,10
raio-x torre-rei oponente	0,11
defensores do rei atacados oponente	0,12
raio-x bispo-rainha oponente	0,12
defensores do rei atacados ativo	0,14

Tabela 7.2: Doze piores atributos ordenados crescentemente por Informação Mútua (atributos excluídos do conjunto final não possuem numeração)

Esses atributos podem ser usados como referência ao se avaliar o desempenho dos Atributos de Tabuleiro desenvolvidos.

A Tabela 7.2 mostra os piores atributos nessa execução do classificador de MI. Muitos desses atributos foram descartados da lista final dos atributos usados nos resultados apresentados, portanto não possuem uma numeração AT; já outros obtiveram melhores valores de MI em execuções em outras bases de partidas e foram mantidos, como o AT6 e AT13, que desempenharam mal na base de dados de jogadores de Elo 1600 – 1699, mas obtiveram melhores resultados em outras execuções.

O processo de seleção de atributos foi iterativo e a comparação da Tabela 7.1 com a Tabela 7.2 foi importante para entender quais tipos de atributos eram mais eficientes nas diferentes execuções. Notou-se ao decorrer desse processo que atributos complexos como "peões isolados" e os atributos de "raio x" não desempenharam bem em

nenhuma das bases de partidas utilizadas. A hipótese elaborada foi a de que esses atributos representam conceitos específicos, e que podem não ocorrer com frequência durante uma partida de xadrez, não sendo úteis na maioria das posições. Por outro lado, os atributos que representam conceitos presentes em qualquer posição de tabuleiro, como os de mobilidade e atividade (AT11, AT12, AT13, AT14), obtiveram resultados significativamente melhores, o que corrobora com a hipótese formulada.

7.1.2. Análise de densidade

De modo a validar o desempenho dos atributos desenvolvidos, também foi conduzida uma análise exploratória do conjunto final de atributos, apresentado nas Seções 5.1 e 5.2. A Figura 7.1 apresenta a densidade de alguns dos atributos extraídos dos grafos, separando o conjunto em posições "erro" e "não erro".

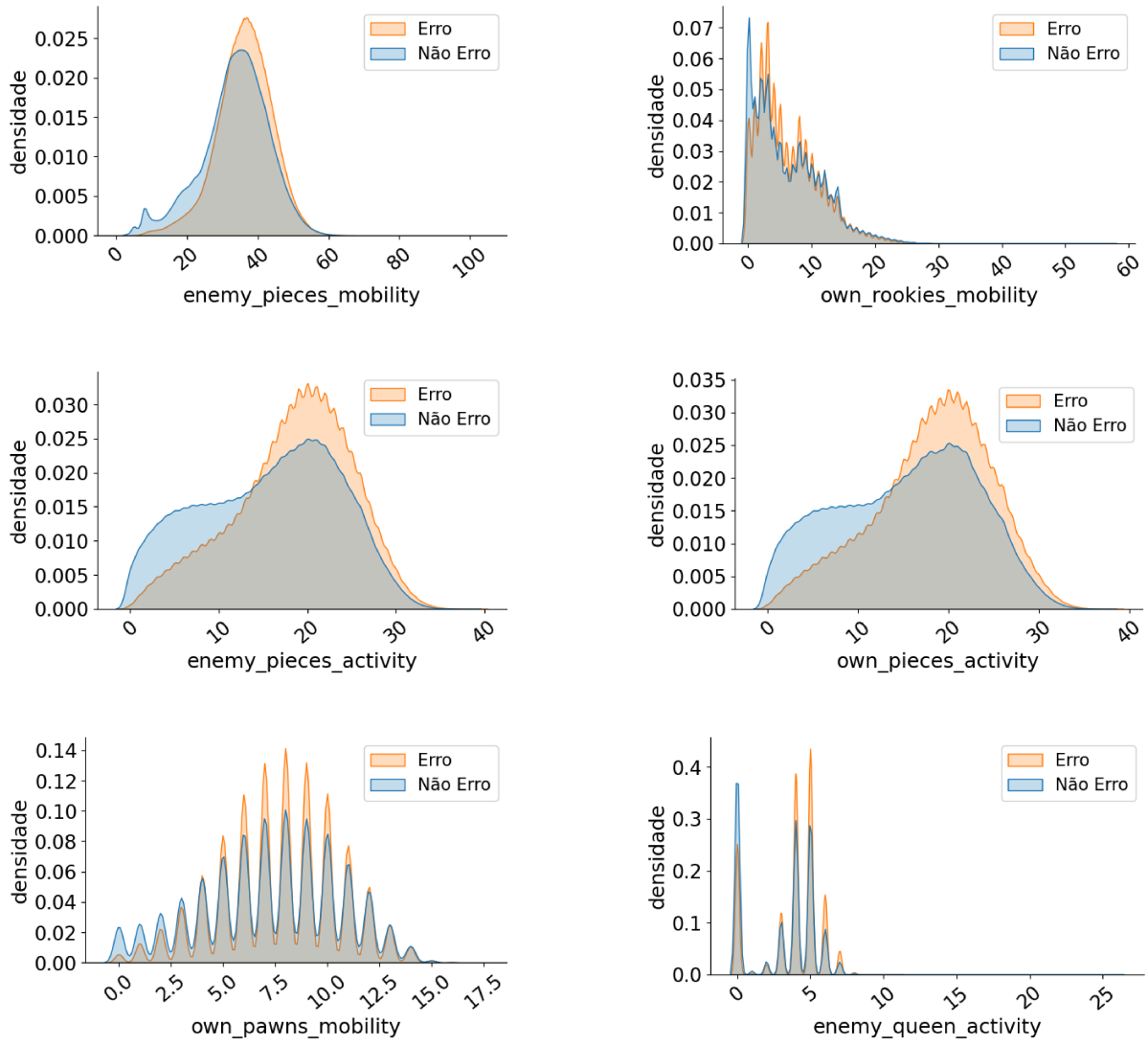

Figura 7.1: Densidade dos atributos extraídos dos grafos

Pode-se observar através da Figura 7.1 a diferença na distribuição de alguns dos atributos desenvolvidos. Alguns dos atributos possuem diferença mais evidente na densidade dos conjuntos "erro" e "não erro", mas em nenhuma delas não se pode observar essa diferença. A Figura 7.2 apresenta o resultado do mesmo experimento, mas dessa vez com os atributos de metadados.

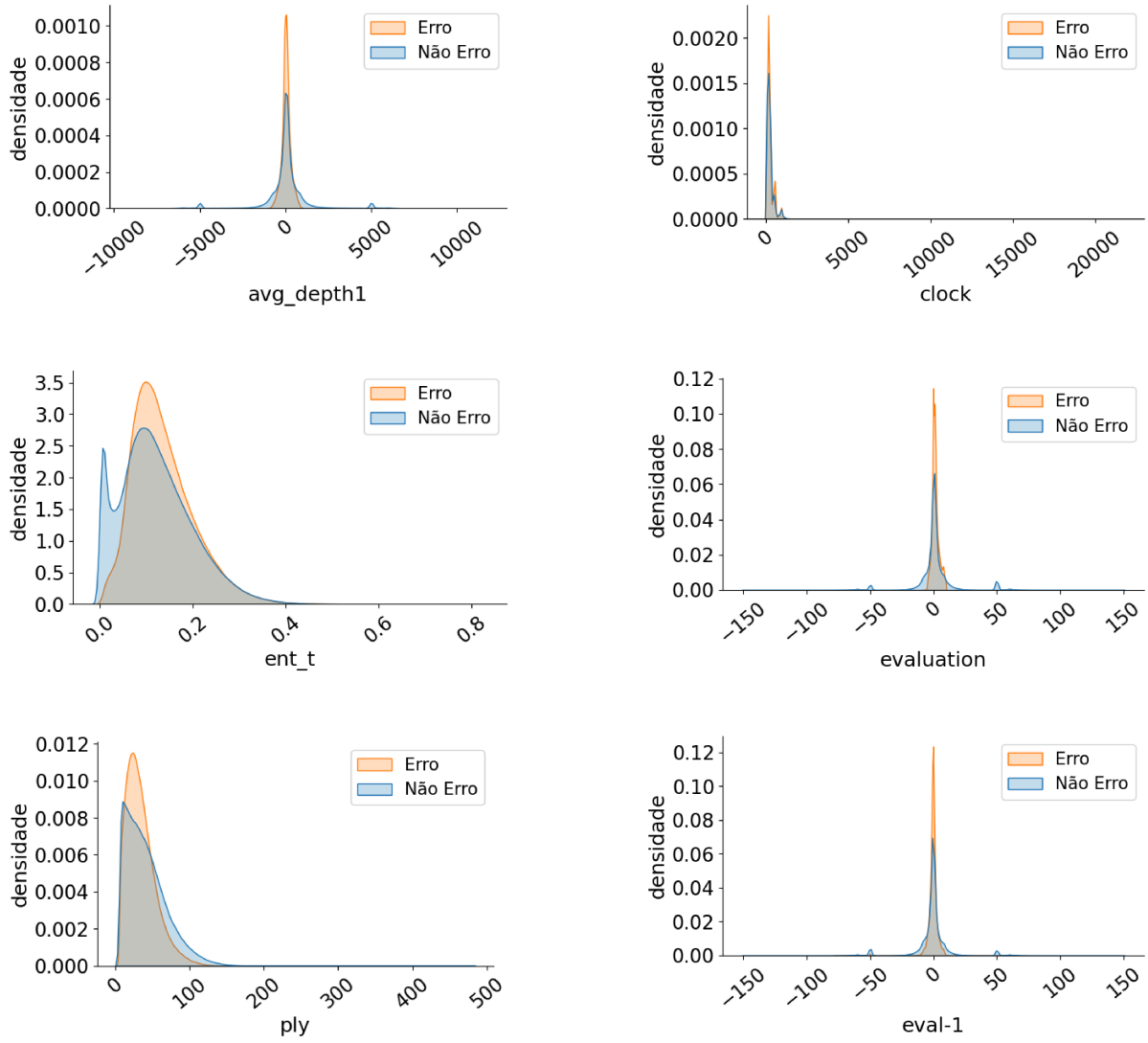

Figura 7.2: Densidade dos atributos de metadados

Os atributos de metadados são válidos como ponto de referência como mostrado nos experimentos de MI, e comparando-se as Figuras 7.1 e 7.2 é possível verificar que os atributos desenvolvidos, semelhantemente aos Atributos de Metadado, também possuem diferenças nas distribuições dos conjuntos "erro" e "não erro". Pode-se observar também, pela Figura 7.2, que os atributos relacionados a avaliação da *engine*, como os AM7, AM8, AM9 e AM10, são os que apresentam a maior diferença nas densidades dos conjuntos, já os AM2 e AM13 apresentam uma diferença significativamente menor.

7.2. Avaliação dos modelos

Nesta sessão serão apresentados os resultados obtidos com o conjunto final de atributos. As métricas acurácia, precisão, revocação e f-score serão utilizadas para a avaliação do desempenho dos modelos utilizados.

Foram treinados no total 18 *Multilayer Perceptrons*, sendo nove treinados com o uso somente dos Atributos de Tabuleiro desenvolvidos e nove com atributos de tabuleiro e metadados. Os resultados dos modelos treinados estão separados por faixa de Elo das partidas utilizadas no processo de treinamento e dispostos na Tabela 7.3. A prática de separação por faixa de Elo é usada na literatura e é interessante no contexto de análise de erro para analisar o comportamento dos modelos em diferentes níveis de habilidade dos jogadores,

Tabela 7.3: Resultados do conjunto final de atributos desenvolvidos

	Atributos de Tabuleiro + Atributos de Metadado				**Atributos de Tabuleiro**			
ELO	**Acurácia**	**Error True**			**Acurácia**	**Error True**		
		Precisão	**Revocação**	**F-score**		**Precisão**	**Revocação**	**F-score**
1100	0,76	0,73	0,82	0,77	0,68	0,64	0,79	0,71
1200	0,75	0,72	0,82	0,77	0,67	0,64	0,77	0,70
1300	0,75	0,72	0,81	0,76	0,66	0,63	0,78	0,70
1400	0,75	0,72	0,80	0,76	0,66	0,63	0,75	0,70
1500	0,74	0,71	0,81	0,76	0,65	0,63	0,76	0,69
1600	0,74	0,71	0,82	0,76	0,65	0,62	0,74	0,68
1700	0,75	0,71	0,79	0,75	0,65	0,62	0,74	0,68
1800	0,73	0,71	0,80	0,75	0,64	0,62	0,73	0,67
1900	0,73	0,71	0,79	0,75	0,64	0,62	0,75	0,68

Observa-se na Tabela 7.3 que a acurácia dos modelos varia de 64% a 68% nos modelos treinados com Atributos de Tabuleiro (AT) e de 73% a 76% nos treinados com ATs e Atributos de Metadado (AM). A diferença de acurácia dos modelos treinados com AMs ressalta a importância desses atributos na previsão de erros, os valores de precisão também apresentam diferenças, variando de 62% a 64% nos modelos de AT e entre 71% a 73% nos modelos AT+AM. A revocação é a métrica que apresentou menor variação entre os modelos AT e AT+AM, obtendo variação de 73% a 79% nos modelos AT e de 79% a 82% nos modelos AT+AM.

Pode se observar também, na Tabela 7.3, que o conjunto de dados que obteve o melhor resultado foi o conjunto de partidas de jogadores entre 1100 e 1199 de Elo, no geral os valores das métricas pioram conforme o nível de habilidade dos jogadores aumenta. Esse comportamento indica que os atributos desenvolvidos foram menos eficientes quando o erro é menos óbvio, ou as posições são mais complexas, que são os casos no xadrez em que jogadores de nível mais alto erram mais. Esse comportamento será melhor discutido nas sessões seguintes.

7.2.1. Comparação com estado da arte

Para validar os resultados obtidos, parte do trabalho de McIlroy-Young et al. (2020a) foi reproduzido, especificamente o modelo de previsão de erros. O modelo consiste em uma *Residual Convulational Neural Network* (*Residual* CNN) e para cada faixa de Elo, esse modelo foi treinado com a mesma base de dados que os modelos cujos resultados foram apresentados na Tabela 7.3. A Tabela 7.5 mostra as acurácias obtidas dos nove modelos treinados conforme a estratégia de McIlroy-Young et al. (2020a). As acurácias obtidas na estratégia desenvolvida foram repetidas para melhor comparação.

Tabela 7.4: Comparação de acurácia com o estado da arte

	Atributos de Tabuleiro + Atributos de Metadado		**Atributos de Tabuleiro**	
ELO	**Estratégia Desenvolvida**	**McIlroy-Young et al. CNN Residual**	**Estratégia Desenvolvida**	**McIlroy-Young et al. CNN Residual**
1100	**0,76**	0,65	**0,68**	0,62
1200	**0,75**	0,65	**0,67**	0,63
1300	**0,75**	0,65	**0,66**	0,63
1400	**0,75**	0,66	**0,66**	0,63
1500	**0,74**	0,66	**0,65**	0,63
1600	**0,74**	0,66	**0,65**	0,64
1700	**0,75**	0,67	**0,65**	0,64
1800	**0,73**	0,67	0,64	**0,65**
1900	**0,73**	0,67	0,64	0,64

A Tabela 7.5 mostra que o comportamento identificado na estratégia desenvolvida é o oposto ao observado na Residual CNN da estratégia de McIlroy-Young et al. (2020a). Enquanto os modelos treinados com os atributos desenvolvidos apresentam melhores resultados em faixas de Elos mais baixas, o modelo de comparação desempenha melhor nas faixas de Elo mais altas.

As acurácias das MLPs desenvolvidas superaram a Residual CNN em 16 dos 18 modelos, as diferenças variaram entre 6% e 9% nos modelos AT+AM e entre 1% e %6 nos modelos treinados somente com atributos de tabuleiro.

Tabela 7.5: Comparação de tempo de execução com o estado da arte

	Construção de grafos e extração de atributos	Treinamento do modelo
Estratégia desenvolvida	90 minutos	8 minutos
Estado da arte	-	180 minutos

É importante ressaltar que o tempo levado para a obtenção dos resultados também diferiu, como pode-se observar na Tabela **??**. Na estratégia desenvolvida, o processo de geração dos grafos, extração dos atributos, treinamento dos modelos e predição dos testes variou entre as faixas de Elo e foi de uma hora e meia a no máximo duas horas. Para as etapas de treinamento e predição da Residual CNN levou-se cerca de três horas para o treinamento do modelo, chegando em alguns conjuntos de dados a cerca de cinco horas e meia. Nessas comparações, todos os modelos haviam a estratégia de *early stop* habilitada e os processos foram realizados no mesmo *hardware*.

7.3. Limitações

Nesta sessão serão apresentadas algumas limitações encontradas durante o processo de obtenção dos resultados. A reprodução do trabalho de McIlroy-Young et al. (2020a) foi possível através do código disponibilizado pelos autores. As configurações do modelo foram ao máximo mantidas conforme as especificações dos autores, mas nem todas estão descritas explicitamente no trabalho, sendo assim, não há como garantir que as configurações utilizadas nos testes são as mesmas dos resultados que os autores apresentam no trabalho. Este fato pode explicar os valores das acurácias apresentadas neste trabalho estarem diferentes aos valores apresentados pelos autores. Outro fato que pode influenciar os resultados obtidos é que os dados utilizados diferiram dos utilizados pelos autores, pois se optou por utilizar os mesmos dados utilizados para treinar os modelos com os atributos propostos. Além de diferentes, os dados utilizados nesse trabalho para treinar o modelo proposto por McIlroy-Young et al. (2020a) tem menor volume, pois, devido ao custo computacional e demora no processo de treinamento, seria inviável reproduzir todos os testes com a massa de dados utilizada pelos autores.

8. CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho apresentou uma nova abordagem para o problema de previsão de erros em partidas de xadrez com a utilização de atributos baseadas em grafos. Foram propostas no total 47 atributos extraídos tanto dos grafos gerados, cumprindo os objetivos O2 e O3 propostos, quanto de metadados da partida. Os atributos desenvolvidos foram selecionadas e avaliadas empiricamente, com testes utilizando o modelo *Random Forest*, e validadas posteriormente com análises exploratórias nas bases de partidas e através do treinamento de um classificador de Informação Mútua.

Os atributos desenvolvidos se mostraram, em sua maioria, úteis na tarefa de separar posições que precedem e que não precedem erro, alcançando sucesso também no objetivo O4. Alguns atributos falharam em prover informação para os modelos de previsão e foram descartados do conjunto final, entretanto, esses atributos ajudaram no direcionamento do desenvolvimento de novos atributos que apresentaram melhores resultados.

Na etapa de obtenção dos resultados, foram treinados dezoito modelos diferentes, separados de acordo com o nível de habilidade dos jogadores presentes no conjunto de dados utilizado no treinamento e pelo tipo de atributo utilizada, somente atributos de tabuleiro, ou também com metadados. Esses modelos foram treinados utilizando os atributos propostos, e alcançou-se o objetivo O5. Além disso, outros dezoito modelos baseados no trabalho de McIlroy-Young et al. (2020a) foram treinados para a realização das comparações com estado da arte. Essa comparação cumpriu o objetivo O6, reproduzindo-se os resultados obtidos por McIlroy-Young et al. (2020a), mas com os mesmo dados utilizados nos modelos desenvolvidos. Isso tornou possível a comparação do desempenho dos modelos treinados com os atributos propostos e os modelos utilizados em estudos recentes.

Os dados utilizados em toda a pesquisa são de 2019 e foram obtidos de uma plataforma de xadrez *online*. No total foram 900 mil partidas de xadrez analisadas, que totalizaram cerca de 4,5 milhões de posições no conjunto já balanceado. Esse volume de dados se mostrou suficiente para os experimentos realizados no desenvolvimento da pesquisa, cumprindo o objetivo O1.

O trabalho de Farren et al. (2013) foi utilizado como ponto de partida para o desenvolvimento dos novos atributos, utilizou-se dois dos grafos propostos pelos autores e foram desenvolvidos novos atributos voltados especificamente para o problema de

previsão de erro nas partidas. Os resultados obtidos foram comparados com os modelos baseados no trabalho de McIlroy-Young et al. (2020a) e comprovou-se a eficácia dos atributos baseados em grafos, comprovando a hipótese H1. Os resultados obtidos pelos atributos propostos se destacam nos modelos que utilizam partidas com níveis de Elo mais baixo. Além disso, comprovou-se a hipótese H2 ao constatar-se uma melhora significativa nos tempos de execução dos processos envolvidos no treinamento dos modelos, nos quais foram observados tempos três vezes menores em alguns conjuntos de partidas.

Durante o desenvolvimento do trabalho, observou-se que os atributos que apresentam melhores resultados são os que estão presentes de forma geral nas posições, mais simples do que atributos complexos que representam conceitos muito avançados do jogo. Como trabalho futuro pode-se avaliar o quanto esse fato pode limitar a quantidade de conhecimento específico do domínio que pode ser empregada no processo de desenvolvimento dos atributos, ou seja, se existem atributos que representam conceitos avançados do jogo de xadrez e que ainda sim, desempenham bem no problema de predição de erros.

Também como trabalho futuro, pode-se estudar mais profundamente o comportamento dos atributos propostos em conjuntos de partidas de jogadores profissionais ou de nível mais alto do que os usados nessa pesquisa, visto que os resultados obtidos indicaram uma queda no desempenho dos modelos treinados com partidas de jogadores de Elo mais alto. Outro fato a ser observado é que neste trabalho os ritmos de jogo não foram separados nos conjuntos de dados, pois se optou por apenas retirar os ritmos extremamente rápidos ou lentos, o que também pode ser investigado de modo a entender melhor os tipos de erro cometido em diferentes ritmos de jogo.

REFERÊNCIAS BIBLIOGRÁFICAS

Anderson, A., Kleinberg, J., and Mullainathan, S. (2017). Assessing human error against a benchmark of perfection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 11(4):1–25.

Biswas, T. and Regan, K. (2015a). Measuring level-k reasoning, satisficing, and human error in game-play data. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 941–947. IEEE.

Biswas, T. T. and Regan, K. W. (2015b). Quantifying depth and complexity of thinking and knowledge. In *ICAART (2)*, pages 602–607.

Brown, J. A., Cuzzocrea, A., Kresta, M., Kristjanson, K. D., Leung, C. K., and Tebinka, T. W. (2017). A machine learning tool for supporting advanced knowledge discovery from chess game data. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 649–654. IEEE.

Chabris, C. F. (2017). Six suggestions for research on games in cognitive science. *Topics in cognitive science*, 9(2):497–509.

Chess.com (2021). Sistema de rating elo - termos de xadrez. https://www.chess.com/pt-BR/terms/sistema-rating-elo-xadrez. Acessado em 10 out. 2021.

Cutler, A., Cutler, D. R., and Stevens, J. R. (2012). Random forests. In *Ensemble machine learning*, chapter 5, pages 157–175. Springer.

Eade, J. (2016). *Chess for dummies*. John Wiley & Sons.

Farren, D., Templeton, D., and Wang, M. (2013). Analysis of networks in chess. Technical report, Stanford University, Tech. Rep.

Glickman, M. E. and Jones, A. C. (1999). Rating the chess rating system. *CHANCE-BERLIN THEN NEW YORK-*, 12:21–28.

Han, J., Pei, J., and Kamber, M. (2011). *Data mining: concepts and techniques*. Elsevier.

Lai, M. (2015). Giraffe: Using deep reinforcement learning to play chess. *arXiv preprint arXiv:1509.01549*.

Lichess (2021). Open database. https://database.lichess.org/. Acessado em 08 set. 2021.

McIlroy-Young, R., Sen, S., Kleinberg, J., and Anderson, A. (2020a). Aligning superhuman ai with human behavior: Chess as a model system. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1677–1687.

McIlroy-Young, R., Wang, R., Sen, S., Kleinberg, J., and Anderson, A. (2020b). Learning personalized models of human behavior in chess. *arXiv preprint arXiv:2008.10086*.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al. (2017). Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*.

StockFish (2021). Open source chess engine. https://stockfishchess.org/. Acessado em 08 set. 2021.

Strogatz, S. (2018). One giant step for a chess-playing machine. *New York Times*, pages 1–6.

Zegners, D., Sunde, U., and Strittmatter, A. (2020). Decisions and performance under bounded rationality: A computational benchmarking approach.