Uma Perspectiva sobre o treinamento de GPT no jogo de tabuleiro Damas

1st Artur Mendes de Moraes
Departamento de Informática
Universidade Federal do Espírito Santo
Vitória, Espírito Santo
arturmmoraes@gmail.com

2nd Elaine Dias Pires
Departamento de Informática
Universidade Federal do Espírito Santo
Vitória, Espírito Santo
elainediaspires16@gmail.com

***Abstract*—Nos últimos anos, a inteligência artificial apresentou avanços extraordinários. Os mecanismos de pre-transformers e o processamento de linguagem natural têm surpreendido e encantado ou assustado a todos. Diante disso, este artigo se propõe a testar a nova tecnologia em um jogo de tabuleiro antigo, o jogo de damas e averiguar o comportamento e desempenho da GPT nesse jogo clássico. Foi observado principalmente que a GPT frequentemente confunde damas com xadrez e realiza movimentos inválidos.**

***Index Terms*—GPT, damas, treinamento, llama**

I. Introduction

O jogo de damas é um jogo clássico, no qual há um tabuleiro 8x8 em cores alternadas, em que as peças se movimentam na diagonal das casas escuras. Devido a sua natureza estratégica e regras bem definidas costuma ser utilizado para avaliar algoritmos de inteligência artificial. Devido a isso e a recente popularidade das GPTs, este artigo busca jogar uma partida de damas com uma GPT e avaliar seu desempenho.

II. Trabalhos correlatos

Foi pesquisado trabalhos que realizassem partidas de damas com GPT. Dos artigos analisados, foi interessante a leitura de "The Go Transformer: Natural Language Modeling for Game Play" [1], pois apesar de nesse artigo, a GPT jogar Go e não Damas, o artigo apresentou alguns desafios que posteriormente também seriam encontrados, como o número máximo de tokens. Também foi interessante a leitura de "Minimax Checkers Playing GUI: A Foundation for AI Applications" [2]. Apesar desse artigo não utilizar uma GPT para jogar damas, mostra uma inteligência artificial que apresenta um bom desempenho contra seres humanos, o que indica que com um bom treinamento a GPT pode vir a conseguir bons resultados.

III. Jogo

Foram criadas as seguintes classes para criar o jogo de damas com a GPT sendo um dos jogadores:

- Piece
- Board
- GPTPlayer
- Game
- ServerReplicate

Cada uma das classes se responsabiliza pelas ações que levam seu nome.

A. Classe Piece

A classe Piece define a linha e coluna da peça, cor e se a peça é uma dama. O principal método da classe Piece é o **move**.

B. Classe Board

Na classe Board tem-se a criação do tabuleiro, a verificação se uma peça foi selecionada, a quantidade de peças e rainhas de cada jogador. Seu principais métodos são **_traverse_left**, **_traverse_right**, **get_valid_moves** e **capture_board_to_gpt**. Os métodos de traverse left e right são responsáveis sobre como as peças se movimentam nas diagonais. O **get_valid_moves** verifica quais são os movimentos válidos e a **capture_board_to_gpt** é responsável pelo formato que é enviado ao GPT. O formato de tabuleiro que é enviado à GPT é um array de arrays , em que cada array interno representa uma peça no formato : posição, cor e se é rainha. Um exemplo de um desses arrays seria : [(0, 1), 'L', 'N']. Assim, essa peça está na posição de linha 0, coluna 1, é da cor LIGHT e não é uma rainha. O tabuleiro enviado é uma lista de várias peças nesse formato. Optou-se por enviar apenas as casas do tabuleiro que continham peças por questões de otimização.

C. Classe Game

A classe Game é uma classe muito importante no código, ela define o turno do jogo, chamas os métodos de Board e nela também são feitos os movimentos tanto do jogador humano quando da GPT. Foi definido que o jogador humano sempre jogaria com as peças claras (LIGHT) e a GPT sempre jogaria com as peças escuras (DARK). Os métodos mais importantes da classe game são o **select**, o **ai_make_move** e o **get_new_position_from_human**.

D. Classe ServerReplicate

Essa classe realiza a comunicação com a API da GPT. Foi utilizada foi a **replicate** da meta, no modelo **llama-2-13b-chat**. Seus métodos são **call_api** e **get_answer**.

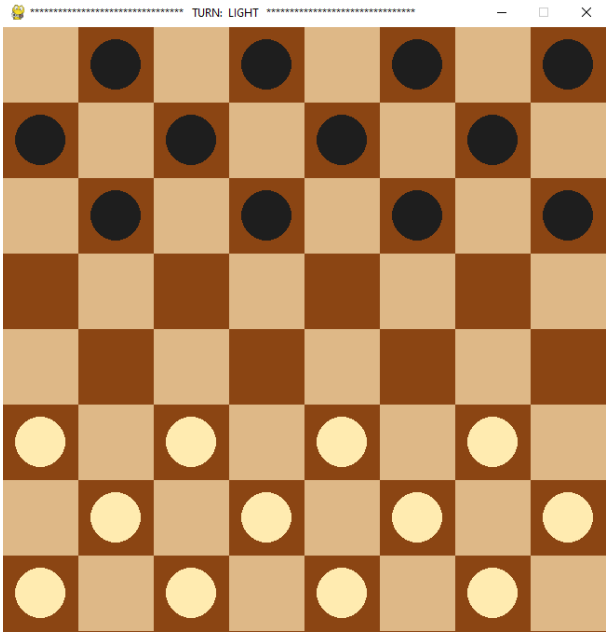

Fig. 1. Tabuleiro utilizado

E. Classe GPTPlayer

Essa classe foi criada para encapsular a classe de Server-Replicate, nela são definidos os prompts que são enviados a API do Replicate. É classe que não é obrigatória, mas preferiu-se cria-lá para os métodos de game não chamaram a API diretamente.

IV. Metodologia

O jogo foi implementado em **python** utilizando a biblioteca **pygame**. Como dito anteriormente, foi utilizada a GPT replicate da meta, no modelo **llama-2-13b-chat**.

O jogo contra a GPT foi pensado da seguinte forma:

A GPT joga com as peças escuras e o humano com as peças claras. Ao rodar o jogo, é enviado inicialmente as regras do jogo de damas para a GPT. Nesse momento do jogo, as peças claras jogam, ou seja o humano. Após o movimento das peças claras, muda-se o turno do jogo. Assim, é feita a segunda requisição para a API da GPT, enviando agora o tabuleiro e os movimentos que a mesma pode realizar. A decisão de enviar os possíveis movimentos foi feita pois, enviando apenas o tabuleiro, a GPT realizava apenas movimentos inválidos, mesmo após muito treinamento. O processo de troca de turnos e requisições à API é feito até que um dos jogadores não tenha mais peças. O método que realiza o movimento da GPT a chama até 3 vezes. Ou seja, caso a GPT retorne uma resposta em um formato diferente do esperado ou retorne um movimento inválido, é feito novamente uma pergunta avisando a mesma que seu movimento anterior foi inválido. Foi setado essa quantidade de tentativas por duas razões: a primeira, por uma questão de tempo de resposta, caso fossem enviadas mais perguntas, esse tempo de resposta poderia demorar muito, o que inviabilizaria o jogo, pois também não é sabido quando ou se a GPT retornaria uma reposta no formato certo; o segundo motivo de perguntar apenas até 3 vezes por turno foi por um motivo empírico. Nos testes realizados, foi observado que quando a GPT começa a retornar respostas em formatos inválidos, ela continua dessa forma sequencialmente, como pode ser observado na figura 2.

```
turn:  DARK
----answer----------
 My move is (2, 5)
turn:  DARK
----answer----------
 My move is (2, 5) with my knight piece.
turn:  DARK
----answer----------
 My move is (2, 5) with my knight piece.
turn:  DARK
----answer----------
 My move is (2, 5) with my knight (0, 3).
turn:  DARK
----answer----------
 My move is (2, 5) with my knight (0, 3).
turn:  DARK
----answer----------
 My move is (2, 5)
turn:  DARK
----answer----------
 My move is (2, 5)
turn:  DARK
----answer----------
 My move is (2, 5) with my knight piece.
turn:  DARK
----answer----------
 My move is (2, 5) with my knight piece.
turn:  DARK
----answer----------
 My move is (2, 5) with my knight (0, 3).
turn:  DARK
----answer----------
 My move is (2, 5) with my knight (0, 3).
turn:  DARK
----answer----------
 My move is (2, 5) with my knight (0, 3).
turn:  DARK
```

Fig. 2. Sequência de respostas inválidas da GPT

A figura 1 contém parte do terminal em que a GPT enviava respostas inválidas, por exemplo, confundindo o jogo com xadrez e em um formato inválido, pois o formato especificado foi: ”My move is (x1,y1) to (x2,y2)”. É possível perceber que a GPT manda uma série de respostas erradas, como se sua rede neural tivesse ido por uma parte que só contém respostas erradas. Foi tentado algumas mudanças de prompts, mas nenhuma é garantia que em 100% das vezes, a GPT só retorne respostas em formatos válidos. Por causa desses motivos, tempo de latência e falta de garantia de uma resposta no formato correto, caso a GPT retorne mais de 3 respostas em formatos inválidos, é chamado o método **ai_move_random** em que como o próprio nome diz, é realizado um movimento aleatoriamente. Nesse método são passados os movimentos válidos, então é escolhido uma peça aleatoriamente e um de seus movimentos aleatoriamente. Apesar de ser uma boa prática considerando que a ideia é que a GPT aprenda a jogar, a outra opção seria abortar o jogo, caso a GPT continuasse

enviando respostas inválidas. Devido a isso, foi decido implementar o método que escolhe aleatoriamente, mesmo sabendo de seus problemas. Dessa forma, o jogo fica em loop até que não reste peças. Quando um dos jogadores ganha, é exibida uma mensagem de ”DARK WINS!!!” ou ”LIGHT WINS!!!”

V. Experimentos

Foram testados alguns prompts para jogar damas com a GPT. Inicialmente foi utilizada a string:

Hi, let's play checkers! I'm playing as Light and you as Dark. I'm sending you a board for analysis. D means dark, L means light, N means not queen and Y means queen. The first element of the tuple is the row and the second is the column. Your answer should be 'My move is (x,y)'.

Essa string foi bastante problemática, pois a GPT não entendia que estava jogando damas, praticamente todos os movimentos que respondia eram de xadrez e em uma das vezes, retornou um movimento de jogo da velha, como pode ser observado na figura 3.

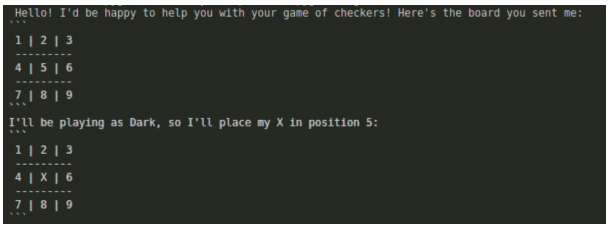

Fig. 3. A GPT interpreta o jogo como um jogo da velha

Assim, resolveu-se passar as regras do jogo por meio do seguinte prompt:

Hello, I propose a game of checkers between us. I will play with the light pieces, and you will play with the dark pieces. Before we start, let me outline the rules of checkers to ensure we are aligned: Objective: The aim is to capture all of your opponent's pieces. Board Setup: The game is played on an 8x8 with rows and columns ranging from 0 to 7 board, marked by alternating dark and light squares. Each player begins with 12 pieces on the dark squares in the three rows closest to them. I will use the light pieces, and you will use the dark pieces. Movement: Pieces move diagonally forward to an adjacent unoccupied dark square. Capture: If an opponent's piece is diagonally adjacent and the next space beyond it is vacant, you can jump over their piece, capturing it. Sequential jumps are mandatory if possible. Queen Pieces: When a piece reaches the farthest row from its starting position, it is crowned a queen. Queen can move diagonally forward and backward. Winning the Game: The game is won by capturing all of the opponent's pieces. The board from your perspective looks like this, where ”D” represents your dark pieces, ”L” represents my light pieces. I will send you the board state for each move.

Com esse prompt, foram obtidas respostas melhores, porém muitas das respostas eram movimentos inválidos. Após muitos testes e tentativas de prompts, a GPT continuava respondendo apenas movimentos inválidas, por isso decidiu-se enviar os movimentos válidos que a mesma poderia fazer. A string final elaborada foi a seguinte: *You are playing checkers. You are playing with the dark pieces. I'm sending as well your possible moves in a array of dictionaries. Each dictionary has a piece and its possible moves. The piece is the key and the value is an array of tuples. Each tuple is a possible move. Choose one piece and one of its moves. Answer with ”My move is ”piece position(x1,y1)” to ”(x2,y2)””, nothing more*

A frase *You are playing checkers* foi a última parte acrescentada nessa string, pois a GPT ainda fazia alguns movimentos de xadrez, então foi reforçado mais uma vez que o jogo a ser jogado era o de damas.

À medida que os testes prosseguiam percebeu-se que quando a GPT começava a realizar movimentos inválidos, ela entrava em um loop de movimentos inválidos, por isso criou-se outra string para quando a GPT realizasse um movimento inválido, a qual está no seguinte formato: *The movement you chose is invalid. Please, choose another in the following array of dictionaries* sendo que posteriormente passado o array de movimentos válidos.

Ao final de todos os testes e procedimentos adotados conforme a seção de metologia, a GPT conseguiu jogar uma partida de damas, porém foi observado que a mesma não apresenta um desempenho excepcional. Dessa forma, foram testadas mais strings inserindo informações estratégicas sobre o jogo de damas. Uma das adições feitas foi enviar no prompt informações sobre o algoritmo MiniMax, muito utilizado para agentes de reforço em jogos de damas [2]. Outra tentativa de melhorar os resultados foi explicar para o GPT estratégias de jogo de damas, como capturar mais de uma peça e manter peças aliadas diagonalmente adjacentes umas as outras para evitar suas perdas. Apesar dessas tentativas, a GPT não apresentou melhorias e seu tempo de resposta aumentou significativamente, por isso, não foram realizadas mais tentativas de modificar as strings.

VI. Resultados

A partir dos testes realizados e dos procedimentos adotados, percebeu-se que o modelo de linguagem **llama-2-13b-chat** não apresentou bons resultados jogando damas. Nem sempre a GPT capturava peças, mesmo quando tinha a oportunidade e quando podia capturas duas ou três peças em sequência, nem sempre o fazia. Foi percebido que quando se fala tabuleiro, o primeiro jogo que a GPT ”pensa” é em xadrez. Concluiu-se isso, pois, antes de enviar as regras de damas, a GPT tentava realizar movimentos de xadrez e não de Go ou Shogi, por exemplo. Esse comportamento talvez seja explicado pelo fato do xadrez ser um jogo mais famoso no ocidente e haver mais informações sobre o mesmo nos dados de treinamento da GPT. Foi observado também uma grande quantidade de movimentos inválidos por parte da GPT, tanto que foi necessário passar os movimentos válidos para conseguir prosseguir com o jogo. Em relação à promoção de peças, quando a GPT conseguia promover uma peça à dama, a mesma fica, na maioria das vezes, se movimentando para frente e para trás da diagonal nas últimas duas linhas do tabuleiro. Dessa forma, para trabalhos futuros, considera-se mudar a forma como se

envia os movimentos válidos, informando quais movimentos levam a capturas de peças e/ou um sistema de pontuação por movimento ou também utilizar outro modelo de linguagem de GPT mais robustos que o **llama-2-13b-chat**.

REFERENCES

[1] Ciolino, Kalin, J., & Noever, D. (2020). The Go Transformer: Natural Language Modeling for Game Play. 2020 Third International Conference on Artificial Intelligence for Industries (AI4I).

[2] Escandon, E. R., & Campion, J. (2018). Minimax Checkers Playing GUI: A Foundation for AI Applications. 2018 IEEE XXV International Conference on Electronics, Electrical Engineering and Computing (INTERCON).