PegAgente: Modelagem de Agentes por Aprendizado de Reforço em Jogos Educacionais

Thiago da Silva Teixeira, UTFPR, tteixeira@alunos.utfpr.edu.br
Helyane Bronoski Borges, UTFPR, helyane@utfpr.edu.br
Simone Nasser Matos, UTFPR, snasser@utfpr.edu.br
Vinicius Schultz Garcia da Luz, UTFPR, viniciusschultz@gmail.com
Tamara Liz Schwab Ribeiro, UTFPR, tamararibeiro@alunos.utfpr.edu.br

Resumo: Jogos educacionais demonstram um modo diferente e divertido de aprender, que pode ser potencializado pela utilização de Inteligência Artificial (IA), tornando a atividade mais dinâmica. Este artigo apresenta um jogo educacional utilizando uma técnica da IA conhecida como aprendizado por reforço, aplicada na modelagem de um agente inteligente. A metodologia usada para desenvolvimento do jogo denominado de PegAgente abrangeu quatro etapas: definição de ferramentas, modelagem do ambiente, modelagem do agente e a simulação. Conforme o nível do jogo aumenta, o agente se torna mais inteligente o que dificulta para o jogador, que precisa fugir e coletar itens que compõem o cenário do jogo. O tema do jogo foi a prevenção contra o vírus COVID-19, em que cada item coletável representa um método preventivo, e o agente inteligente é representado em formato de um vírus. O jogo demonstrou que a modelagem de agentes em jogos educacionais por meio do aprendizado por reforço permite a criação de um jogo com dificuldade ideal ao jogador, com o objetivo de gerar maior engajamento.

Palavras-chave: aprendizado por reforço, jogos educacionais, COVID-19.

PegAgente: Agent Modeling by Reinforcement Learning in Educational Games

Abstract: Educational games demonstrate a different and fun way to learn, which can be strengthened by the use of Artificial Intelligence (AI) making the activity more dynamic. This paper presents an educational game using an AI technique known as reinforcement learning to model an intelligent agent. The methodology used to develop the game called PegAgente included four steps: definition of tools, environment modeling, agent modeling and the simulation. As the game levels increase, the agent becomes more intelligent making it harder for the player who needs to escape and collect items that compose the game scenario. The game theme was the prevention for the COVID-19 virus, where the intelligent agent is represented by the format of a virus and each collectable item represents a preventive method. The game proved that modeling agents in educational games using reinforcement learning allows the creation of a game with ideal difficulty level for players, in order to generate greater engagement.

Keywords: reinforcement learning, educational games, COVID-19.

1. Introdução

O uso de jogos educacionais no processo de ensino e aprendizagem proporciona motivação, estimula a criatividade e a persistência. Esses tipos de jogos revelam uma alternativa à educação tradicional, tendo como propósito a relação dinâmica entre o conteúdo educativo e o jogador, proporcionando diversão e aquisição de conhecimento (Albuquerque e Fialho, 2009).

Um dos fatores que influenciam no engajamento do jogador é a dificuldade do jogo (Qin; Rau e Salvendy, 2010). Atividades muito difíceis requerem uma resiliência grande por parte do jogador além de produzirem muita ansiedade. Em contraste, jogos

muito fáceis podem parecer tediosos por não apresentarem desafios (Chanel *et al.*, 2011). Espera-se que um jogo que se adapte a dificuldade ideal de cada jogador, desperte um engajamento maior, assegurando a transmissão da parte educacional que aquele jogo trás.

O desenvolvimento de jogos educacionais pode ser realizado com a aplicação de várias áreas da tecnologia, dentre elas, destaca-se o uso da Inteligência Artificial (IA). Para McCarthy (1998), criador do termo IA, o campo trata-se da ciência e engenharia da criação de máquinas inteligentes.

Tais sistemas inteligentes podem ser compreendidos utilizando o conceito de agente orientado a objetivo, em que tal sistema percebe o ambiente através de sensores, e atua por meio de atuadores, de modo que suas ações levem a solução de um problema determinado (Russel e Norvig, 2004). Tal racionalização pode ser programada explicitamente ou por meio de Aprendizado de Máquina (AM), onde o agente aprende de forma automática como tomar boas decisões com base em sua experiência (Monard e Baranauskas, 2003).

Este artigo apresenta um jogo educacional denominado de PegAgente, voltado ao público infantil. O jogo possui um agente inteligente que aprende por meio de uma técnica de Aprendizagem por Reforço (AR), o que permite um fator adaptativo para a dificuldade do jogo proposto.

O tema do jogo está relacionado a pandemia COVID-19 (Wu; Chen e Chan, 2020) a qual possui alguns desafios, sendo um deles a educação e conscientização da população para os métodos de prevenção do vírus, de modo a diminuir a taxa de contagio no local. A utilização de máscaras, lavar as mãos e realizar distanciamento social são exemplos de modos de prevenção (Eikenberry *et al.*, 2020).

No cenário do jogo cada forma de prevenção é representada através de itens, que o jogador tem que coletar para ganhar o jogo. Além disso, é necessário que o jogados fuja do inimigo, o qual é representado em formato de vírus. Tal vírus é controlado por um agente com aprendizado por reforço, que fica cada vez mais inteligente e desafiador para o jogador, até atingir o nível ideal para o usuário.

2. Aprendizado por reforço

A aprendizagem por reforço permite que um agente aprenda por meio de informação do ambiente, interagindo através de ações a, e recebendo um estado s e uma recompensa R. Com base na interação do agente com o ambiente, ele aprende quais são as melhores ações a a serem tomadas em um estado s, de modo a maximizar sua recompensa.

Pelo reforço das ações que levam a uma maior recompensa, o agente cria uma política de ações ótimas por meio de uma tabela de ações, chamada tabela Q, com a recompensa esperada em cada ação, avaliando a qualidade de suas ações no ambiente em um determinado estado e executando a melhor ação possível esperada. Este método é conhecido como Q-learning (Watkins, 1989). Inicialmente o agente, em suas primeiras interações com o ambiente, não tem conhecimento de quais são as melhores ações. Está fase é conhecida como exploração. Nesta etapa o agente coleta amostras estatísticas do ambiente, executando ações aleatoriamente e com base na recompensa adquirida, atualiza uma tabela com a qualidade das ações, de acordo com a Equação 1:

$$s, a : Q(s, a) \leftarrow (1 - \alpha)Q(s, a) + \alpha(R(s) + \gamma Q(s', a')) \quad (1)$$

A função Q, Equação 1, atualiza a qualidade da ação a no estado s. Onde s, a é o conjunto estado-ação, α corresponde a taxa de aprendizagem, γ o fator de desconto

temporal e R a função de recompensa baseado no estado s.

$$a(s) = argmaxQ(s, a) \tag{2}$$

Considerando um par de estado-ação visitado de forma a tender ao infinito, a função $Q(s, a)$ irá convergir com probabilidade 1 para a ação $a*$ ótima, considerando um valor α suficientemente pequeno (Watkins e Dayan, 1992), selecionando a ação $a*$ pela Equação 2.

Na fase de treinamento do agente, é necessário realizar a exploração e a explotação. A explotação é uma fase em que o agente explora a ações já aprendidas, com o objetivo de maximizar seu ganho de recompensa. Por meio da politica e-gulosa, em que as ações correspondentes a exploração ou explotação do agente pelo ambiente é determinado pela probabilidade $1 - e + \frac{e}{A(s)'}$, em que $A(s)$ corresponde ao número de ações possíveis no estado s, e e o parâmetro de controle entre gula e aleatoriedade.

3. Trabalhos relacionados

Foi realizada uma pesquisa a fim de identificar jogos que utilizam em sua aplicação algoritmo de aprendizagem de máquina. Essa pesquisa foi realizada em cinco bibliotecas digitais: “ACM Digital Library”, “Science Direct”, “Google Scholar”, “IEEE Xplore Digital Library” e “Scopus” no período de 2003 a 2019.

O trabalho de Cowley *et al.* (2014) realizado na área de psicofisiologia, descreve um novo método para integrar preferências de jogadores, dados experimentais e padrões de desenhos de jogos em um único *framework*, chamado *Play Patterns And eXperience* (PPAX). O framework explorou os padrões de jogabilidade e reações fisiológicas das faces dos jogadores obtendo como resultado informações padrão de reação, jogadas e a personalidade de cada jogador.

No estudo realizado por Barata *et al.* (2016), o experimento caracterizou dados para prever o perfil de aluno no início do curso de mestrado em ciências da computação da universidade de Lisboa. Para isso, utilizou algoritmos de aprendizado de máquina para classificar dados de alunos de um período e prever o perfil de aluno em outro período. Por meio da inclusão de jogos na grade de ensino, foi criado os jogos *MCP Quest* e *Skill Tree*. Para ampliar interatividade criaram um *ranking* para premiar a passagem de níveis e as experiências adquiridas pelos alunos.

Outro estudo realizado por Bharathi *et al.* (2016), diversos jogos e aplicativos para *smartphones* foram analisados para levantamento de características de *design* e elementos de gamificação, como desafios, *feedbacks*, recompensas, objetivos, personagens, insígnias, pontuação, níveis, *ranking* e dinâmica de estados dos jogos. Tais elementos de gamificação resultaram em um aumento na motivação ao aluno, como um incentivo para continuar os estudos.

O sistema *LudifyMe*, criado por Llorens-Largo *et al.* (2016) para avaliar o potencial da gamificação como meio de aprimorar a aprendizagem. A principal contribuição é na aplicação de jogos no ensino de inteligência artificial a fim de aumentar a motivação, desempenho e satisfação do aluno.

Siu *et al.* (2018) desenvolveram uma plataforma de jogos baseada na história de Romeu e Julieta para auxiliar a aprendizagem na língua inglesa. Nesta plataforma é realizado uma análise das pontuações obtidas pelos alunos, rastreando sua evolução e domínio da língua inglesa, com o objetivo de prever se o aluno atingirá a pontuação necessária para as provas.

4. Metodologia

O desenvolvimento do jogo PegAgente foi realizado em 4 (quatro) etapas: Definição de Ferramentas, Modelagem do Ambiente, Modelagem do Agente e Simulação, conforme ilustrada a Figura 1. Ressalta-se que as etapas Modelagem do Agente e Simulação são iterativas, de forma a realizar o treinamento do agente.

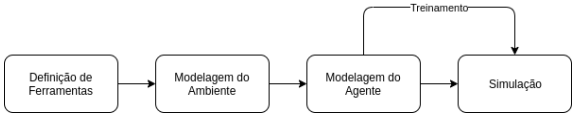

Figura 1. Etapas da metodologia de desenvolvimento do PegAgente

A etapa de Definição de Ferramentas foi realizada por meio de uma busca na literatura identificando as ferramentas mais adequadas. Dentre elas cita-se: Python (Rossum e Drake, 2009), Pygame (Shinners *et al.*, 2011), NumPY (Oliphant, 2006) e Godot (Engine, 2017).

Para a implementação do PegAgente usou a linguagem de programação Python (Rossum e Drake, 2009). Esta é uma linguagem de alto nível e utilizada na área de inteligencia artificial por possuir várias bibliotecas direcionadas ao aprendizado de máquina, como Scikit-learn (Pedregosa *et al.*, 2011) e Tensorflow (Abadi *et al.*, 2016).

Para o ambiente de treinamento do PegAgente usou a biblioteca Pygame (Shinners *et al.*, 2011). Essa biblioteca é voltada para o desenvolvimento de jogos em Python e facilita a integração do ambiente e o jogo por utilizarem a mesma linguagem de programação. O Pygame facilita a análise visual de como o PegAgente desempenha seu papel de perseguir o jogador.

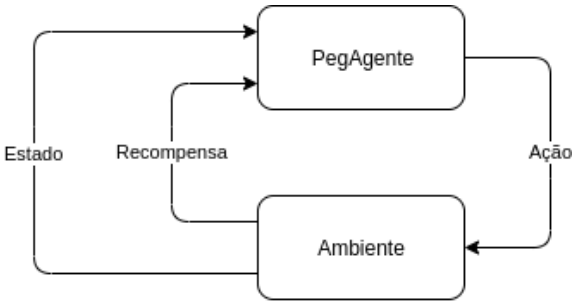

Figura 2. Interação entre o PegAgente e o ambiente de treino.

Na etapa de Modelagem de Ambiente, foi criado o mapa para o jogo, em que o agente inicia sempre na parte central e o jogador na parte inferior. A cada interação com o agente, é necessário que o ambiente retorne: o estado, definido como a localização do agente e o jogador; a recompensa, valor baseado em uma função de recompensa, que indica a qualidade de um estado. A função de recompensa $R(s)$ foi modelada conforme mostra a Equação 3, em que a função $distancia$ retorna a distância euclidiana entre o jogador e o agente, em um determinado estado s, e n_{piso} corresponde a quantidade de pisos no ambiente, o qual possui valor 200.

$$R(s) = \begin{cases} \frac{-distancia(s)}{distancia_{maxima}}, & \text{se distância } s > 0 \\ n_{piso}, & \text{se distância } s = 0 \end{cases} \quad (3)$$

Na etapa de Modelagem do Agente, o treinamento do PegAgente foi realizado usando aprendizagem por reforço, o que permite um incremento gradual de seu

desempenho no ambiente. É necessário que este ambiente responda as ações do agente, por meio de um estado e uma recompensa, representado pelo diagrama da Figura 2.

No aprendizado por reforço, o agente tem como objetivo conseguir o maior número de recompensa possível em um episódio de treinamento. Cada episódio é uma partida simulada do agente usada para retirar amostras de recompensa do ambiente e por meio do algoritmo Q-learning tenta convergir para ações ótimas. A tabela Q do agente foi implementada em uma matriz, usando a biblioteca de computação científica NumPy (Oliphant, 2006). Essa biblioteca permite um melhor desempenho computacional na fase de treinamento do agente quando comparado com a utilização de uma estrutura de lista padrão da linguagem Python (Walt; Colbert e Varoquaux, 2011).

O agente percebe o ambiente por meio do estado s, que contém sua localização e a localização do jogador. Cabe ao agente aprender a utilizar essas informações para se movimentar no ambiente. Ressalta-se que inicialmente o agente não reconhece esse ambiente, como por exemplo as paredes, bem como as ações que podem melhorar o seu desempenho. O agente interage com o ambiente pelas ações a, por meio de movimentos que podem ocorrer em quatro direções em um espaço bidimensional: cima, baixo, esquerda e direita.

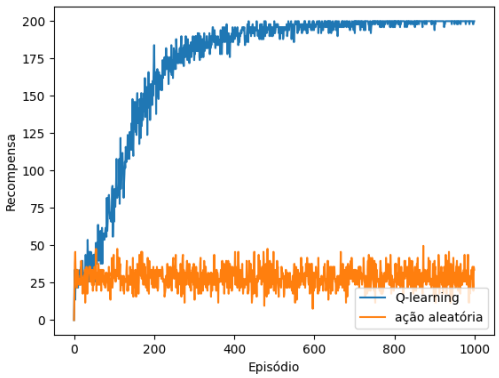

Figura 3. Curva de aprendizado do agente em busca de recompensa.

Na etapa de Simulação é feita uma análise comportamental do agente. A Figura 3 exibe a curva de aprendizagem do agente em um treino de 999 episódios. A medida que o agente simula esses jogos por episódios, esse adquire mais experiência do ambiente e obtém número maior de recompensa. Quando comparado com a ação aleatória nota-se que a recompensa tende a permanecer constante (aproxima 20 pontos). Sendo assim, observa-se que o agente melhorou seu desempenho durante o treinamento chegando a atingir 200 pontos que é o valor máximo a ser alcançado neste ambiente. O agente é recompensando de duas maneiras: quando chega perto do jogador ou quando alcança o jogador.

A tabela Q do PegAgente é salva a cada 33 episódios, permitindo a criação de 30 níveis no jogo. Cada nível do jogo está associado a uma fase do treino do PegAgente o que lhe permite uma dificuldade progressiva.

Após a etapa de Modelagem do Ambiente e Modelagem do Agente, o jogo é implementado utilizando o motor de jogo Godot (Engine, 2017), onde será feito a animação dos personagens e a criação do pacote para os sistemas Windows (Bott e Stinson, 2019) e Ubuntu Linux (Sobell, 2015).

O agente é representado como o vírus COVID-19 (Wu; Chen e Chan, 2020), e os itens no jogo são objetos ou ações que previnem contrair o vírus, como o uso de máscaras

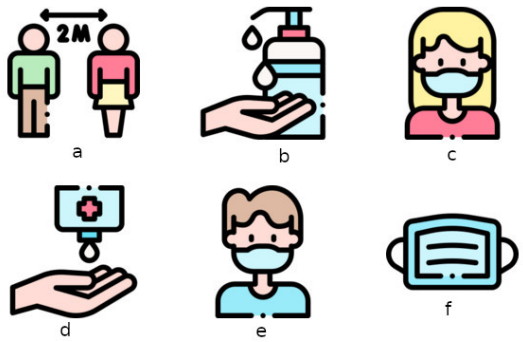

Figura 4. Itens coletáveis no jogo, onde: (a) representa o distanciamento social;(b) e (d) higienização das mãos; (c),(e) e (f) o uso de máscara (Repeat, 2014)).

(Eikenberry *et al.*, 2020), distanciamento social, uso de álcool em gel e a higiene correta das mãos (Pradhan *et al.*, 2020). Isso reforça as principais medidas preventivas para combate ao vírus COVID-19, e a saúde como um todo, já que as medidas preventivas são importantes para combater diversas outras doenças, além do COVID-19 (Cairncross *et al.*, 2010).

Cada item na Figura 4 representa uma forma de prevenção o vírus, reforçando o caráter educativo do jogo, ensinando a importância do distanciamento social, higienização das mãos e o uso de máscara. Só por meio da coleta desses itens é possível vencer o jogo, fazendo uma alusão de como deve ser a prevenção para vencer o vírus na vida real.

Figura 5. Arte do personagem usado no jogo (Montero, 2017).

O jogo PegAgente possui uma inspiração em jogos do tipo Pac-Man (Namco, 1980), em que o jogador precisa coletar itens em um mapa fixo e fugir dos inimigos. Caso o jogador colete todos os itens do mapa, ele vence o jogo. Se ele for alcançado pelo inimigo neste processo, o jogador perde a jogada.

O visual do protagonista foi extraído do site opengameart.org (Montero, 2017), que disponibiliza artes de domínio público. A Figura 5 mostra a arte do personagem do jogo, o qual é baseado em jogos de aventura inspirado em Zelda (Nintendo, 1991), em que a visão do jogador é de cima para baixo.

5. Resultados

Os resultados obtidos com o desenvolvimento do PegAgente são apresentados em termos de funcionamento, simulação e discussão em relação aos trabalhos relacionados.

5.1. Funcionamento do jogo

O jogo é dividido em 30 níveis, quanto maior o nível, mais o agente aprende. O PegAgente realiza centenas de simulações em que ele joga sozinho e ganha experiência de forma que seu desempenho é melhorado. Conforme o avanço de níveis do jogo, o agente fica mais inteligente, consequentemente aumentando a dificuldade do jogo, visto que é mais difícil para o jogador fugir do agente vírus, conforme ilustra a Figura 6.

Figura 6. Personagem do jogo a esquerda fugindo do PegAgente (vírus) a direita.

O jogo começa em um mapa fixo, representado na Figura 7, onde o personagem do jogador esta posicionado na parte inferior e o vírus na parte central, acima do personagem. Cabe ao jogador, por meio da utilização das teclas direcionais do teclado, mover o personagem e coletar os 6 itens, espalhados aleatoriamente pelo mapa, como apresenta a Figura 4.

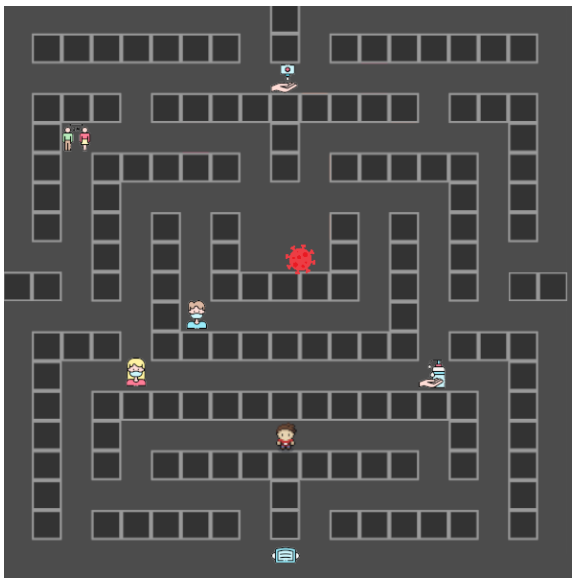

Figura 7. Mapa do jogo.

Se o jogador conseguir coletar todos os itens do mapa, esse passa para o próximo nível de dificuldade. Nesse caso, o agente possuirá maior conhecimento acerca do ambiente. Caso o PegAgente consiga alcançar o jogador, o jogo reinicia. O jogador e o agente voltam para suas posições iniciais, os itens do jogo são realocados aleatoriamente no mapa reiniciando a jogada sem avanço de nível.

O jogo distribui de maneira aleatória os itens no mapa e permite que o jogador explore o ambiente de maneira diferenciada a cada jogada, deixando o jogo mais dinâmico.

O PegAgente não sabe da existência desse itens coletáveis do jogo, pois no ambiente de treinamento esse itens não estão presentes. Isso ocorre para evitar que o agente não escolha estratégias que possam finalizar o jogo, como ficar em cima de um item, tornando impossível que o jogador vença.

A temática do jogo reforça os principais métodos de prevenção contra o vírus COVID-19 (Wu; Chen e Chan, 2020), por meio dos itens coletáveis. Sendo eles o distanciamento social, higienização das mãos (Pradhan *et al.*, 2020) e o uso de máscara (Eikenberry *et al.*, 2020).

5.2. Simulação do PegAgente

A simulação realizada para o PegAgente ocorreu em três níveis. A Figura 8 mostra a progressão do agente em relação a experiência adquirida em cada intervalo de nível do jogo. Nessa figura a cor vermelho representa a área dominada pelo agente.

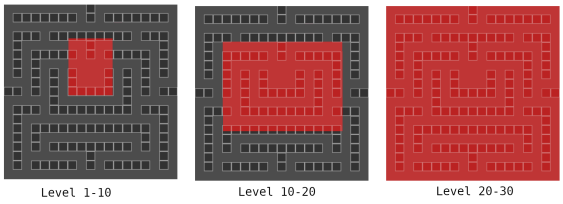

Figura 8. Progressão de domínio do PegAgente sobe a área do mapa do jogo.

Nos primeiros níveis, de 1 a 10, o PegAgente praticamente não consegue sair da parte central do mapa, pois não conhece ainda onde estão as paredes. Dessa forma, a maior parte de suas ações resulta em colisões.

Já nos níveis intermediários, 10 a 20, o PegAgente consegue se movimentar pelo mapa sem muitas colisões. Nessa fase, ele já percebe a presença do jogador quando esse se aproxima iniciando a perseguição.

Por fim, nos últimos níveis, de 20 a 30, o PegAgente, logo no inicio da jogada, vai diretamente de encontro com o jogador. Neste estagio ele já conhece o ambiente, não realiza nenhuma colisão com as paredes e sabe quais são as melhores rotas em qualquer condição do jogo.

5.3. Discussões

Alguns trabalhos da literatura usam o aprendizado de máquina em jogos educacionais para classificar padrões do usuário, como os trabalhos de Cowley *et al.* (2014) e Barata *et al.* (2016). Estes estudos utilizam aprendizado não supervisionado em que a inteligência artificial gera um modelo para minerar dados já armazenados em um repositório.

Outros estudos como Bharathi *et al.* (2016), Llorens-Largo *et al.* (2016) e Siu *et al.* (2018), apesar de utilizarem algoritmos diferentes dos de clusterização, necessitam que os dados já estejam gravados para que o processo de mineração seja realizado. Isto pode acarretar um tempo maior para o levantamento dos dados de várias pessoas, pois precisa atingir a quantidade de dados necessários para a viabilidade do uso de técnicas de aprendizado de máquina.

Neste trabalho foi utilizado aprendizado por reforço, que permite ao agente aprender de forma automática, sem intervenção humana. Sua fase de treinamento requer um custo computacional alto, devido ao numero de iterações necessárias para reconhecer o ambiente. Porém, uma vez treinado, o desempenho do agente no ambiente é satisfatório, pois o algoritmo Q-learning garante a convergência em ações ótimas (Watkins e Dayan, 1992). Isto possibilita usar sua curva de aprendizado no treinamento para gerar níveis de dificuldades.

O foco da aprendizagem de máquina por reforço aplicado no PegAgente permite que o agente aprenda, por meio da sua interação com o ambiente. Portanto, não necessita

previamente de dados para seu aprendizado ou a programação de uma máquina de estado finito.

6. Conclusão

O jogo proposto ensina métodos de prevenção para o vírus COVID-19 por meio de sua temática, o que permite um ensino de conceitos atuais importantes para o público infantil em um contexto de pandemia. Como por exemplo, lavar as mãos e utilizar máscaras, que são importantes para a higiene em geral.

A aplicação de aprendizado por reforço para modelagem de agentes em jogos educativos, demostrou ser uma ferramenta efetiva e versátil, em que o PegAgente aprendeu os princípios do jogo somente interagindo com o ambiente, sem nenhum conhecimento prévio ou intervenção humana.

Este trabalho contribuiu para a exploração de métodos de aprendizado de máquina na criação de jogos educacionais, com dificuldade dinâmica. Tal fator proporciona ao o jogador permanecer em sua dificuldade ideal, de modo que aumente seu engajamento e incentive seu aprendizado do conteúdo educativo.

A realização de testes com o público infantil ainda precisa ser executada, com o objetivo de analisar a hipótese de maior engajamento do jogador pelo uso da dificuldade dinâmica, bem como avaliar a efetividade do aprendizado dos métodos de prevenção e fatores de aceitação do jogo.

Como trabalhos futuros, a modelagem do agente desenvolvida pode ser aplicada em outros jogos educativos de diferente temáticas e gêneros. Além disso, a exploração de novas tecnologias com aprendizado de reforço em aplicações educacionais, devido ao grande potencial adaptativo do agente, como demonstrado neste artigo.

Referências

Abadi, M. *et al.* Tensorflow: A system for large-scale machine learning. In: **12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)**. [S.l.: s.n.], 2016. p. 265–283.

Albuquerque, R. d.; Fialho, F. Concepção de jogos eletrônicos educativos: Proposta de processo baseado em dilemas. **VIII Simpósio Brasileiro de Jogos e Entretenimento Digital**, v. 8, 2009.

Barata, G.; Gama, S.; Jorge, J.; Goncalves, D. Early Prediction of Student Profiles Based on Performance and Gaming Preferences. **IEEE Transactions on Learning Technologies**, v. 9, n. 3, p. 272–284, 2016. ISSN 19391382.

Bharathi, A.; Singh, A.; Tucker, C.; Nembhard, H. Knowledge discovery of game design features by mining user-generated feedback. **Computers in Human Behavior**, v. 60, p. 361–371, 07 2016.

Bott, E.; Stinson, C. **Windows 10 inside out**. [S.l.]: Microsoft Press, 2019.

Cairncross, S. *et al.* Water, sanitation and hygiene for the prevention of diarrhoea. **International journal of epidemiology**, Oxford University Press, v. 39, n. suppl_1, p. i193–i205, 2010.

Chanel, G.; Rebetez, C.; Bétrancourt, M.; Pun, T. Emotion assessment from physiological signals for adaptation of game difficulty. **IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans**, IEEE, v. 41, n. 6, p. 1052–1063, 2011.

Cowley, B. *et al.* Experience Assessment and Design in the Analysis of Gameplay. **Simulation and Gaming**, SAGE Publications Sage CA: Los Angeles, CA, v. 45, n. 1, p. 41–69, 2014. ISSN 1552826X.

Eikenberry, S. E. *et al.* To mask or not to mask: Modeling the potential for face mask use by the general public to curtail the covid-19 pandemic. **Infectious Disease Modelling**, Elsevier, 2020.
Engine, G. Godot. **Motor de Jogo 2d e 3d de Código Aberto**, 2017.
Llorens-Largo, F. *et al.* LudifyME: An Adaptive Learning Model Based on Gamification. In: Caballé, S.; Clarisó Learning Data Analytics and Gamification, R. B. T. F. A. (Ed.). **Formative Assessment, Learning Data Analytics and Gamification: In ICT Education**. Boston: Academic Press, 2016. p. 245–269. ISBN 9780128036679.
McCarthy, J. What is artificial intelligence? 1998.
Monard, M. C.; Baranauskas, J. A. Conceitos sobre aprendizado de máquina. **Sistemas inteligentes-Fundamentos e aplicações**, Manole Ltda, v. 1, n. 1, p. 32, 2003.
Montero, A. **Zelda-like tilesets and sprites**. 2017. Disponível em: ⟨https://opengameart.org/content/zelda-like-tilesets-and-sprites⟩.
Namco, P.-M. **Game [Arcade]**. [S.l.]: Namco, 1980.
Nintendo, E. The legend of zelda: A link to the past. **Nintendo. Super Nintendo Entertainment System**, 1991.
Oliphant, T. E. **A guide to NumPy**. [S.l.]: Trelgol Publishing USA, 2006. v. 1.
Pedregosa, F. *et al.* Scikit-learn: Machine learning in python. **the Journal of machine Learning research**, JMLR. org, v. 12, p. 2825–2830, 2011.
Pradhan, D.; Biswasroy, P.; Ghosh, G.; Rath, G. *et al.* A review of current interventions for covid-19 prevention. **Archives of medical research**, Elsevier, 2020.
Qin, H.; Rau, P.-L. P.; Salvendy, G. Effects of different scenarios of game difficulty on player immersion. **Interacting with Computers**, Oxford University Press Oxford, UK, v. 22, n. 3, p. 230–239, 2010.
Repeat, A. by flaticon. **Freepik. com. Date**, 2014.
Rossum, G. V.; Drake, F. **Python 3 Reference Manual; CreateSpace**. [S.l.]: Scotts Valley: CA, 2009.
Russel, S.; Norvig, P. Inteligência artificial. 2ª. edição. **Rio de Janeiro: Campus**, 2004.
Shinners, P. *et al.* Pygame. **Dostupné z: http://pygame. org/[Online**, 2011.
Siu, W. L. *et al.* Using an English language education APP to understand the English level of students. In: **2018 27th Wireless and Optical Communication Conference, WOCC 2018**. [S.l.: s.n.], 2018. p. 1–3. ISBN 9781538649596.
Sobell, M. G. **A practical guide to Ubuntu Linux**. [S.l.]: Pearson Education, 2015.
Walt, S. v. d.; Colbert, S. C.; Varoquaux, G. The numpy array: a structure for efficient numerical computation. **Computing in science & engineering**, IEEE Computer Society, v. 13, n. 2, p. 22–30, 2011.
Watkins, C. J.; Dayan, P. Q-learning. **Machine learning**, Springer, v. 8, n. 3-4, p. 279–292, 1992.
Watkins, C. J. C. H. Learning from delayed rewards. King's College, Cambridge, 1989.
Wu, Y.-C.; Chen, C.-S.; Chan, Y.-J. The outbreak of covid-19: An overview. **Journal of the Chinese Medical Association**, Wolters Kluwer Health, v. 83, n. 3, p. 217, 2020.