UNIVERSIDADE PRESBITERIANA MACKENZIE
PROGRAMA DE PÓS-GRADUAÇÃO EM
ENGENHARIA ELÉTRICA E COMPUTAÇÃO

BRUNO OMELLA MAINIERI

DIFICULDADE ADAPTATIVA EM JOGO PARA O ENSINO DA MATEMÁTICA

Orientador: Prof Dr Nizam Omar

São Paulo
2019

BRUNO OMELLA MAINIERI

DIFICULDADE ADAPTATIVA EM JOGO PARA O ENSINO DA MATEMÁTICA

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica e Computação da Universidade Presbiteriana Mackenzie para a obtenção do grau de Mestre em Engenharia Elétrica e Computação.

Orientador: Prof Dr Nizam Omar

São Paulo
2019

M224d

Mainieri, Bruno Omella

Dificuldade adaptativa em jogo para o ensino da matemática. / Bruno Omella Mainieri – São Paulo, 2019.

88 f. : il., 30 cm.

Dissertação (Mestrado em Engenharia Elétrica e Computação) - Universidade Presbiteriana Mackenzie - São Paulo, 2019.

Orientador: Nizam Omar.

Bibliografia: f. 75-80.

1. Dificuldade adaptativa. 2. Jogos digitais. 3. Jogos educativos. 4. Objeto de aprendizagem. 5. Aprendizado de máquina. I. Omar, Nizam, *orientador*. II.Título.

CDD 371.397

Bibliotecária Responsável: Marta Luciane Toyoda – CRB 8/ 8234

BRUNO OMELLA MAINIERI

DIFICULDADE ADAPTATIVA EM JOGO PARA O ENSINO DA MATEMÁTICA

Dissertação apresentada ao Programa de Pós Graduação em Engenharia Elétrica e Computação da Universidade Presbiteriana Mackenzie para a obtenção do grau de Mestre em Engenharia Elétrica e Computação.

Aprovado em 13 de Fevereiro de 2019

BANCA EXAMINADORA

Prof. Dr. Nizam Omar
Universidade Presbiteriana Mackenzie

Prof. Dr. Pedro Henrique Cacique Braga
Universidade Presbiteriana Mackenzie

Prof.ª Dr.ª Juliana Cristina Braga
Universidade Federal do ABC

RESUMO

Em 2015, 53,1% dos alunos de terceiro ano do ensino fundamental público brasileiro demonstraram conhecimento inadequado em matemática. O emprego de dificuldade adaptativa em jogos educativos é proposto com a finalidade de manter o desafio da experiência adequado ao nível de habilidade de cada jogador, em constante evolução, a fim de promover engajamento e permitir o aprendizado de matemática, em particular operações aritméticas básicas de adição, subtração e multiplicação, abordadas nesta etapa escolar. Nesta pesquisa foram desenvolvidas duas versões de um jogo para ensino da matemática, uma dotada desta capacidade adaptativa e outra não. O objeto foi implementado com base em definições de características motivadoras e avaliação de algoritmos de aprendizado de máquina, e levado a testes com alunos de terceiro ano do ensino fundamental. Dados deste experimento sugerem que a versão adaptativa provocou maior engajamento nos usuários, contando com tempo de interação 17,9% mais longo que a versão não adaptativa. Com base na análise do desempenho dos jogadores, a versão adaptativa também foi capaz de oferecer desafios mais adequados à habilidade dos usuários.

Palavras-chave: *dificuldade adaptativa, jogos digitais, jogos educativos, objeto de aprendizagem, aprendizado de máquina*

ABSTRACT

In 2015, 53,1% of brazilian public schools thrid graders had inadequate knowledge of maths. Adaptive difficulty in educational games is proposed with the goal of maintaining adequate challenge for each player's constantly evolving skill levels, thus promoting engagement and learning of the arithmetical operations of addition, subtraction and multiplication which are studied in this grade. In this research, two versions of an educational mathematics game were created - one of which possesses adaptive difficulty capabilities, and another which does not. The object was implemented according to definitions of motivational characteristics of games and assessment of different machine learning algorithms before being tested with third grade students. Results from this experiment suggest that the adaptive version of the game created greater engagement in users, with a 17,9% longer average interaction time compared to the non-adaptive version. Analysis of the players' performance also indicates that the adaptive version was capable of offering a level of challenge more adequate to the users' skills.

Keywords: *adaptive difficulty, digital games, educational games, learning object, machine learning*

Sumário

1 INTRODUÇÃO

Em 2015, apenas 42,9% dos alunos do terceiro ano do ensino fundamental da rede pública brasileira demonstraram conhecimento adequado em matemática (CRUZ, 2015). Vista esta carência, identifica-se espaço para o emprego de jogos eletrônicos como objetos de ensino auxiliares (COHEN, 1969; MALONE; LEPPER, 1987; CORDOVA; LEPPER, 1996; MICHAEL; CHEN, 2006; VOGEL et al., 2006). Estes tem como principal característica facilitadora do aprendizado a capacidade de promover a motivação dos usuários (GARRIS; AHLERS; DRISKELL, 2002), parte da qual é decorrente do desafio proposto por estes (CORDOVA; LEPPER, 1996).

Csikszentmihalyi (1975) argumenta que, para que o desafio seja de fato engajador, este deve ser apropriado para o nível de habilidade do usuário: nem superior a este, a ponto de causar frustração, nem inferior, provocando desinteresse. Além disso, o autor destaca a necessidade de contemplar a evolução da competência do indivíduo, cujo ritmo varia para cada usuário. Especificamente na aplicação de jogos ao ensino, Cardoso, Ghelli e Oliveira (2017) defendem a necessidade de garantir um nível adequado de dificuldade destes objetos, sendo suficiente para gerar incerteza, necessária para o aprendizado, mas não chegando ao ponto de provocar ansiedade ou levar o aluno a crer que é incapaz de superar o desafio. Para garantir a manutenção o nível adequado de desafio às habilidades de cada jogador e assim provocar efetivamente o engajamento, surge a proposta de emprego de técnicas de dificuldade adaptativa para estes jogos (SAMPAYO-VARGAS et al., 2013). Com a implementação destas, um jogo seria capaz de identificar as competências de cada jogador bem como suas dificuldades, adaptando dinamicamente elementos a fim de facilitar ou dificultar a experiência.

A implementação de técnicas de adaptabilidade é custosa no desenvolvimento de um jogo, fazendo necessário que sua adição seja justificada com evidência formal de sua eficácia para fins, no caso, de aprendizado (SAMPAYO-VARGAS et al., 2013). Embora outros estudos existam com este objetivo, não há consenso entre os resultados obtidos, e novos experimentos e abordagens são solicitadas (SAMPAYO-VARGAS et al., 2013; SILVA; SILVA; CHAIMOWICZ, 2017). A escolha dentre as alternativas de tecnologias disponíveis para a implementação desta característica também constitui uma preocupação (ANDRADE et al., 2005). A definição de um modelo baseado em inteligência artificial,

consideradas as arquiteturas e paradigmas possíveis dentro desta área, para atingir esta finalidade seria, desde que comprovadamente eficaz, uma facilitadora no processo de desenvolvimento de jogos digitais.

Um mesmo sistema capaz de identificar o desempenho do jogador para adaptar a dificuldade e manter o desafio seria também capaz de atuar como avaliador, auxiliando professores na tarefa de identificar as competências e carências de cada aluno com relação ao conteúdo abordado (STEVENS et al., 1999).

Finalmente, técnicas para a manutenção do engajamento em um jogo proporcionam não só oportunidades para a criação de jogos educativos, como também de simulações instrucionais e outras formas de treinamento virtual.

Esta pesquisa tem como objetivo principal avaliar a empregabilidade de técnicas de dificuldade adaptativa em jogos educativos para ensino da matemática. Mais especificamente, o trabalho busca analisar diferentes alternativas para a implementação de adaptabilidade em jogos, bem como o emprego de diferentes arquiteturas de redes neurais artificiais para este propósito. A partir dos resultados desta análise, toma-se como meta o desenvolvimento de um protótipo de jogo educativo para ensino da matemática para uso em sala de aula, com duas versões, uma empregando dificuldade adaptativa e a outra não. Por fim, é proposta a realização de testes com alunos cursando série escolar condizente com o conteúdo proposto pelo jogo, a fim de comparar os resultados obtidos por cada versão do jogo.

Sobre estes objetivos, a pesquisa apresenta duas hipóteses relacionadas à empregabilidade de dificuldade adaptativa em jogos educativos. A primeira diz respeito ao aprendizado proposto: dados dois jogos idênticos, exceto pela presença de dificuldade adaptativa em apenas um deles, o jogo adaptativo promoverá maior aprendizado nos alunos que o jogam, conforme medido por prova escrita daquele conteúdo. A segunda diz respeito à motivação do jogador: dados dois jogos idênticos, exceto pela presença de dificuldade adaptativa em apenas um deles, o jogo adaptativo promoverá maior engajamento dos jogadores, conforme medido por questionário de opinião aplicado aos usuários. Estas hipóteses são complementares, uma vez que o maior aprendizado será promovido, ao menos em parte, pelo maior engajamento dos jogadores.

Este projeto de pesquisa é organizado da seguinte maneira: a segunda seção trata

do referencial teórico, apresentando os conceitos cujo entendimento se faz necessário para o desenvolvimento da proposta. A seção seguinte, a terceira do documento, descreve a proposta de trabalho a ser realizada, incluindo os testes de implementação e o desenvolvimento do protótipo. A quarta e última seção traz o cronograma do desenvolvimento desta pesquisa, contemplando a disposição temporal dos passos explicitados na metodologia.

2 REFERENCIAL TEÓRICO

Neste referencial são relacionados os conceitos fundamentais para o entendimento da pesquisa elaborada. Primeiro, são apresentadas definições de jogos digitais e suas características, bem como de *Design* e Mecânicas de jogos. Em seguida, uma análise das dinâmicas de aprendizagem permite relacionar as necessidades deste processo com valores que podem ser oferecidos pelos jogos. É apresentado então o conceito de dificuldade adaptativa, e em seguida são introduzidos os algoritmos de redes neurais que podem ser empregados para obter esta característica em um jogo digital.

2.1 Jogos e o ato de jogar

A década de 1970 viu o surgimento de jogos eletrônicos como uma forma de entretenimento e cultura. Em seus primeiros anos, o entendimento dos *videogames* era indissociável do estudo das formas de jogo mais tradicionais, como jogos de carteado, tabuleiro ou esportes (CRAWFORD, 1982).

Huizinga (1949) descreve o conceito de jogar, mais de três décadas antes do advento dos jogos eletrônicos, porém aplicável também a estes, como sendo um ato fundamental para o ser vivo, não necessariamente subordinado à cultura ou à sociedade. O autor descreve características do ato de jogar que considera fundamentais: jogar é um ato livre, uma expressão pura de liberdade; aqueles que jogam tem a consciência de que o jogo não é real, e sim imaginário (algo que Zimmerman (2004) viria a chamar de "Artificialidade"); jogar não é necessariamente oposto à seriedade, e pode contemplar aspectos e propósitos práticos e aplicáveis a situações externas ao jogo.

Acrescentando à definição de jogar, Zimmerman (2004) apresenta este como a varia-

bilidade possível dentro dos limites e restrições de algum sistema. Segundo o autor, jogar é ter liberdade e agência, ao mesmo tempo que respeita e depende das regras existentes. Desta forma, caracteriza-se como papel do jogador fazer uso das ferramentas e contexto fornecidos pelo jogo para criar resultados e experiências diversas e imprevisíveis. O ato de jogar, então, só ocorre dentro de um conjunto de regras estabelecidas que permitam liberdade em sua execução ou interpretação.

Esta dependência do jogar das regras do sistema é uma das características primárias da área de estudo conhecida como *Design* de Jogos. Ao definir o que são jogos e por consequência, quais aspectos são compreendidos pelo *Design* de Jogos, Crawford (1982) encaixa este ponto no aspecto que nomeia como Representação: um jogo representa um sistema definido por regras e objetivos, um espaço no qual o jogador pode agir com liberdade de igual extensão à variabilidade permitida por estas regras. Trata-se de uma representação que o autor define como subjetiva, uma vez que a validade desta depende da interpretação e suspensão de descrença do jogador. É este aspecto que, de acordo com o autor, permite ao jogador abstrair o aspecto irreal do jogo, e vivenciá-lo como algo autêntico.

Crawford (1982) elenca ainda três outros aspectos indispensáveis à caracterização de um jogo: Interação, Conflito e Segurança. Ao tratar de Interação, o autor defende que para uma atividade ser considerada um jogo, ela precisa permitir ao praticante um grau de agência real, isto é, suas ações, decisões e competências devem promover resultados correspondentes - proposta que é reforçada por Zimmerman (2004). Tradicionalmente, isto é obtido através da participação de outros usuários: ao jogar xadrez, as decisões de um jogador influenciam a forma como o outro joga. No contexto de jogos digitais, o autor nota a possibilidade de usar algoritmos de heurísticas e inteligência artificial para gerar este tipo de reação em um jogo sem depender unicamente de outro participante humano.

Conflito é, então, um produto natural da interação: é a oposição de vontades ou forças na busca de um objetivo. Dois jogadores em uma partida de xadrez estão em constante conflito, ambos procurando derrotar o outro. Em um jogo digital individual, o jogador entra em conflito com elementos reativos do próprio jogo, inimigos controlados por computador e sistemas de inteligência artificial. Neste ponto, Zimmerman (2004) apresenta uma visão diferente, ampliando a definição de conflito em um jogo para também

compreender desafios estáticos como quebra-cabeças e não interativos como limites de tempo.

Tendo em vista esta necessidade por conflito, e a forma exagerada como este pode ser expresso em jogos, particularmente os digitais, Segurança é o aspecto que permite ao jogador vivenciar estas situações sem medo ou preocupação. Corroborado pela proposição de Huizinga (1949) de que os jogadores estão cientes de que o jogo não é real, Crawford (1982) defende que a segurança oferecida pelos jogos é um importante fator atrativo desta forma de mídia, apelando à vontade dos usuários de ter experiências que, de outra forma, seriam demasiadamente perigosas ou estressantes.

Em estudo com o objetivo de identificar e descrever as características que constituem um jogo, Järvinen (2009) propõe uma relação de nove elementos, separados em três categorias: elementos de sistema, elementos compostos e elementos comportamentais.

Elementos de sistema dizem respeito à forma como se representa o modelo do jogo, e englobam componentes e ambiente. Componentes são objetos com formas ou aspectos físicos e características que ilustram conceitos do jogo, como peças de um tabuleiro, cartas, modelos de personagens e objetos. Um componente pode pertencer a um jogador, sendo um personagem, pertence ou o próprio jogador, ou ao sistema do jogo, como no caso de uma bola em um jogo de futebol. Ambiente se refere ao espaço, físico ou simulado em meio digital, onde ocorre o jogo, e caracteriza e limita as opções de movimentação dos componentes. Sendo um fator limitante da liberdade de ação dos jogadores, está diretamente relacionado às regras do jogo.

Elementos compostos regulam e caracterizam a forma como jogadores interagem com o sistema do jogo. São elementos desta categoria regras, mecânicas, informação, tema e interface. A definição do autor de regra se assemelha à de Crawford (1982), com a notável diferença de que enfoque é dado às regras como determinantes das condições de vitória do jogo, de seus objetivos. Mecânicas descrevem as formas como um jogador pode agir dentro do modelo do jogo, podendo ser comparadas, em casos de jogos digitais, às opções de *input* disponíveis ao jogador. A extensão destas opções de ação é definida pelo conjunto de regras do jogo, e cada tomada de decisão pode ser avaliada como resultando em sucesso ou não com base na aproximação do jogador ao estado de vitória no jogo.

Todos os resultados de interações, expressões de regras e mecânicas e dados relativos

ao estado do jogo se encaixam na definição de informação, como o elemento que situa o jogador e lhe dá propriedade sobre a situação do jogo. As características de um jogo que vão além de suas regras e mecânicas e expressam a fantasia sobreposta à experiência prática constituem o tema de um jogo, e podem ser empregadas para aumentar a imersão e motivação do jogador. Interface é o elemento que permite ao usuário interagir com os demais componentes de um jogo quando o acesso direto a estes é impossível. Sempre presente em jogos digitais, esta é fator determinante para a definição das mecânicas disponíveis.

Elementos comportamentais são jogadores e contexto - externos ao sistema do jogo, mas ditam a forma como ocorre o uso deste. Jogadores são considerados elementos deste tipo pois seu comportamento ao se relacionar com o sistema do jogo tem influência na experiência resultante. As capacidades, motivações e decisões de cada jogador, bem como a forma que múltiplos jogadores se relacionam, geram variabilidade no funcionamento do sistema que é impossível de prever com base apenas nos demais elementos. O contexto atua de forma similar, caracterizando a influência do meio em que ocorre o ato de jogar sobre a experiência do jogo.

Abordando em particular o elemento de mecânicas de jogo, Sicart (2008) caracteriza estas como aspectos inerentemente interativos da experiência de um jogo por meio dos quais o jogador pode expressar sua agência. Especificamente, o autor sugere que mecânicas podem ser aproximadas às formas de *input* de um jogo, solidificando a proposta de que estas representam a totalidade da agência do jogador.

Além de caracterizá-las como elementos compostos, Järvinen (2009) aborda o conceito de mecânicas de jogo descrevendo a possibilidade de categorizar e hierarquizar estas. A primeira categorização de mecânica diz respeito ao escopo da influência desta no contexto do jogo. Uma mecânica que está presente a todo momento de um jogo, ou na maior parte de seus estados, é considerada global. Em contraste, uma mecânica que só se torna disponível dentro de um contexto particular no jogo é dita local, ou modificadora. Uma segunda categorização é feita com relação aos objetivos ligados a cada mecânicas. Mecânicas que estão diretamente relacionadas à realização dos objetivos principais do jogo são ditas primárias, enquanto aquelas que possibilitam a realização de objetivos locais ou de partes menores de um objetivo maior são chamadas submecânicas.

O autor ainda constrói uma relação de 40 mecânicas identificadas em jogos em sua pesquisa. Cada mecânica listada pelo autor pode exercer a função de mecânica global ou local, primária ou submecânica, dependendo da forma como é empregada no jogo. Cada mecânica esta aprensentada na forma de um cartão contendo nome, definição, exemplos e, em vezes, notas, conforme exemplificado no Figura 1, com o cartão da mecânica *Accelerating / Decelerating*.

Figura 1: Cartão de Mecânica

Mechanic: Accelerating / Decelerating

Definition: The players are allowed to change the speed of the game element (often component-of-self) they are manoeuvring.

Examples: Mario Kart, SSX.

Notes: Often a submechanic to **Manoeuvring.**

Fonte: Järvinen (2009), p.385

Ao abordar a questão de elementos de jogo, Djaouti et al. (2008) fazem uso de conceitos que chamam de *Gameplay Bricks* para descrever ações que um jogador ou outra entidade em um jogo digital pode tomar. Estes *bricks* podem ser vistos como uma combinação de elementos diretamente referentes ao *input* do usuário, sendo assim mecânicas de jogo, de acordo com Sicart (2008), nomeados *Play Bricks*, e elementos derivados do resultado de ações dentro do contexto do jogo, nomeados *Game Bricks*. Os autores apresentam a proposta de que a combinação destes *bricks* das duas categorias formam pares que representam outro conceito importante de jogos eletrônicos, o *Gameplay*. De acordo com o estudo realizado, eles teorizam que o *Gameplay*, definido como a experiência que um jogador vivencia ao participar de um jogo, é composto por ciclos de ação e resultado, onde o usuário toma decisões e vê as consequências destas.

Conhecidas as características dos jogos digitais e as suas realizações na forma de elementos de jogo, pode-se explorar a viabilidade de seu emprego para fins educativos. Esta proposta é ainda fortalecida por Crawford (1982), ao afirmar que o objetivo mais

primitivo por trás do ato de jogar é o de aprender.

2.2 Jogos no processo de aprendizagem

Como instrumento lúdico interativo capaz de abordar diferentes temáticas e conceitos através de mecânicas, o jogo pode ser analisado sob o mérito de objeto facilitador do ensino. Esta mídia permite que o indivíduo atue sobre os objetos sendo estudados, representados por elementos de jogo, de forma limitada pelas regras estabelecidas e com a garantia de receber como resposta a consequência de sua interação, processo este que permite o desenvolvimento de conhecimento (CARDOSO; GHELLI; OLIVEIRA, 2017).

Embora não exista consenso na literatura acerca da definição precisa do termo objeto de aprendizagem (WILEY et al., 2000; KURILOVAS; KUBILINSKIENE; DAGIENE, 2014; GUEVARA; AGUILAR; GONZALEZ-ERAS, 2017), neste trabalho é adotado o significado proposto por Wiley et al. (2000) de objeto de aprendizagem como qualquer recurso de natureza digital que auxilie no processo de aprendizado e possa ser reutilizado em diferentes contextos, capacidades e momentos, definição na qual os jogos digitais se enquadram.

Thomaz e Megid (2017), em argumentação construída para justificar o emprego de jogos no ensino da matemática porém aplicável ao ensino tradicional como um todo, apontam como qualidade desta forma de mídia a sua ubiquidade na rotina dos alunos, sendo a interação com estes objetos não apenas natural, como desejada por estes. O jogo, tenha este um fim educacional ou não, é divertido e atraente para os alunos - características essas que são favoráveis ao aprendizado (GRANDO, 2004).

Enquanto objetos de auxilio e promoção do aprendizado, particularmente da matemática, Corbálan (1996) constrói uma categorização dos jogos empregada também por Thomaz e Megid (2017). Nesta, os jogos são descritos como sendo de conhecimento ou de estratégia, de acordo com sua relação com o conteúdo educacional. Jogos tidos como de conhecimento abordam os conceitos estudados de maneira direta, introduzindo o aluno a tópicos novos de forma lúdica. Nesta categoria encontram-se jogos em que o teor educativo se coloca de forma explícita, como jogos numéricos para o ensino de operações matemáticas. Jogos de estratégia, por sua vez, são aqueles que dependem da resolução

de problemas e superação de desafios para alcançar a vitória - fator engajador principal destes. Jogos nesta categoria levam o aluno, já introduzido ao conteúdo relacionado, a praticar e expandir seu domínio sobre o tópico, sendo recompensado com vitória. Estes jogos comumente envolvem dois ou mais jogadores competindo a fim de obter o melhor desempenho. Esta competição, por sua vez, pode tanto ser um aspecto negativo para o aproveitamento da experiência, resultado em problemas de relacionamento e perda de auto estima, quanto positivo, gerando maior motivação para aprimoramento (MALONE; LEPPER, 1987).

Independentemente da categorização do objeto, o sucesso de qualquer jogo como ferramenta para aprendizado depende de seu emprego no contexto de ensino. De maneira análoga à forma como a presença de regras e estruturação da atividade definem esta como um jogo (HUIZINGA, 1949), é o planejamento de sua aplicação levando em conta o conhecimento prévio dos alunos e sua disposição e preferências, sua inclusão num processo mais abrangente de ensino e o objetivo educacional claramente declarado e tornado explicito que permitem que um jogo obtenha êxito enquanto objeto de aprendizagem (CARDOSO; GHELLI; OLIVEIRA, 2017).

2.3 Motivação e aprendizado

A promoção de experiências fantasiosas, o desafio e o engajamento propostos por jogos digitais tornam estes uma mídia adequada para objetivos educacionais. Essas características intrínsecas os tornam motivadores (GARRIS; AHLERS; DRISKELL, 2002), e a motivação, por sua vez, é fator fundamental para o aprendizado (MALONE; LEPPER, 1987; CORDOVA; LEPPER, 1996).

Malone e Lepper (1987) elencam quatro motivações que chamam de individuais e que são necessárias no processo de aprendizado. Já trabalhando no contexto de jogos digitais, é ainda de valia notar os paralelos possíveis entre as propostas dos autores com definições de Zimmerman (2004) e Crawford (1982) para características de *Design* de Jogos.

A primeira motivação apresentada é o desafio. De acordo com os autores, e tomando como base teorias como a do Fluxo, de Csikszentmihalyi (1975), desafio é a condição engajadora de uma atividade que possui objetivos cuja realização não seja garantida. A

vontade de obter êxito e o efeito positivo na auto-estima que isto gera são, para os autores, facilitadores do aprendizado. Paralelos para esta motivação podem ser encontrados nas teorias de *Design* de Jogos quando estas dizem respeito à Interação e Conflito - quando a oposição de outros jogadores ou de elementos controlados por computador tornam incerto o sucesso do jogador (CRAWFORD, 1982; ZIMMERMAN, 2004).

Em seguida, os autores apresentam a motivação da curiosidade. Definida como o resultado da incerteza, da quebra de expectativas e da possibilidade de descoberta, a curiosidade é tida como impulso primitivo para o ato de descobrir e aprender. Malone e Lepper (1987) separam esta em duas categorias, a curiosidade sensorial, onde há estímulo aos sentidos por meio, principalmente, de imagens e sons, e a curiosidade cognitiva, onde o estímulo é direcionado ao modelo mental do indivíduo e causado por incongruências entre expectativas e evidências. Num contexto de jogos digitais, tecnologias de *output*, como animações, processamento gráfico e áudio podem promover a curiosidade sensorial, enquanto a variabilidade promovida pela interatividade e não-linearidade oriundas aos jogos geram curiosidade cognitiva. Na abordagem de Järvinen (2009), a curiosidade está relacionada ao elemento de informação: limitar o conhecimento do jogador acerca de eventos e fatos do contexto do jogo estimula a vontade de descoberta.

O terceiro tipo de motivação intrínseca apresentado por Malone e Lepper (1987) é o controle. Nesta proposta, controle é caracterizado como a realização da agência do indivíduo sobre a experiência que este vivencia. Indo além, os autores argumentam que mais do que o real poder sobre a situação, é a percepção deste que estimula o usuário a ter mais investimento e imersão na atividade. Poder ver os resultados de suas ações e esforços na forma como os eventos se desenrolam em um dado contexto gera um efeito cíclico de motivação, em que o indivíduo passa a tomar propriedade da situação e busca promover mais mudanças. Este tipo de *feedback* dinâmico para a tomada de decisões é previsto como aspecto integrante de jogos digitais no contexto de Interação (ZIMMERMAN, 2004).

A última categoria de motivação apresentada é a fantasia. Além de proporcionar engajamento do usuário, apelando para interesses que a realidade não é capaz de suprir, a fantasia também auxilia no processo de aprendizado, permitindo ao indivíduo abstrair conceitos técnicos através de analogias com elementos ficcionais. Crawford (1982) já explicitava a fantasia como motivação por trás do ato de jogar, e Zimmerman (2004)

elenca esta como fator importante para a experiência de jogo. No modelo de Järvinen (2009), a fantasia é promovida principalmente pelo elemento do tema.

Dentro do conceito de fantasia, Malone e Lepper (1987) fazem uma distinção entre dois tipos de fantasia. Um tipo é a fantasia endógena, onde os conceitos ficcionais estão diretamente ligados ao aspecto prático ou técnico da atividade, como, por exemplo, em um jogo onde frações são praticadas ao partir pedaços de tortas. A outra variedade é a fantasia exógena, na qual a ficção e a prática não se relacionam, como no jogo da forca, onde a dedução de palavras influencia o enforcamento de um personagem, apesar de não ter qualquer relação lógica com este processo. De acordo com os autores, fantasias endógenas são mais favoráveis à criação e manutenção de motivação do que as exógenas.

Consideradas as características que tornam possível o aprendizado, Prensky (2001) afirma, abordando a questão a partir de um ponto de vista sociológico, que os jovens dos anos 2000 não tem interesse pelas formas tradicionais de ensino. O autor argumenta que, acostumados a um mundo de interatividade e tecnologia, estes indivíduos não encontram motivação em atividades estáticas, como treinamentos e aulas. Este conclui, então, que mídias digitais, em particular jogos, devem se tornar materiais de ensino fundamentais no futuro, para que possa ocorrer aprendizado real, conclusão que vai de encontro à proposta de McGonigal (2011) acerca da necessidade de introduzir tecnologias digitais e particularmente jogos em contextos sociais e educacionais.

Para corroborar as propostas de Prensky (2001) e McGonigal (2011), o estudo de Cohen (1969) possui valor histórico ao demonstrar que a indisposição de alunos com relação a modelos clássicos de aulas não é exclusiva ao novo milênio. Em seu estudo foram comparados jogos, no caso não digitais, com métodos de aula clássicos. Entre os resultados demonstrando a preferência dos alunos pelos jogos, destaca-se que 87% dos participantes afirmam que os jogos são mais interessantes que atividades de aula tradicionais.

2.4 Adaptatividade em Jogos Digitais

Dentre as motivações elencadas que promovem o engajamento e aproveitamento do jogo por parte do usuário, destaca-se aqui a denominada Desafio. Ao mesmo tempo em que afirma que o desafio é fundamental para que haja motivação, Csikszentmihalyi

(1975) aponta que um indivíduo, ao investir tempo e esforço em uma dada atividade, se torna mais habilidoso e apto a realizá-la, inevitavelmente deixando o estado de fluxo e passando para o tédio e o desinteresse. Desta forma, uma atividade como um jogo que não se adapte à evolução do jogador irá deixar de motivá-lo. A simples evolução linear do desafio proposto pelo jogo não seria suficiente para manter o engajamento: de acordo com Andrade et al. (2005), diferentes indivíduos se desenvolvem em ritmos diferentes, sendo impossível prever um modelo que se adeque a todos os possíveis jogadores.

Para manter o jogador engajado e experienciando um nível de desafio sempre adequado, se faz necessário um modelo adaptativo de evolução de dificuldade (HUNICKE, 2005). Ajuste dinâmico de dificuldade (DDA, *Dynamic Difficulty Adjustment*) é uma solução proposta para este cenário, que propõe a alteração de aspectos de um jogo de acordo com o desempenho do jogador até então. Nesta abordagem, pode-se alterar, dependendo do tipo do jogo, elementos como o número de inimigos, a competência destes ou os limites de tempo impostos ao jogador.

A aplicação de soluções como o DDA requer, além da capacidade de alteração de elementos do jogo de maneira dinâmica, um modelo a partir do qual possam ser decididas quais mudanças devem ser realizadas. Hunicke (2005) propõe um modelo no qual são realizadas mudanças em elementos de jogo de maneira proporcional à competência medida do jogador: se o jogador tem desempenho menor que o esperado, o dano causado por inimigos é reduzido proporcionalmente, por exemplo. A Figura 2 exemplifica um modelo linear, no qual o desempenho do jogador é medido em um valor único p, que sintetiza todos os aspectos da competência no jogo.

Figura 2: Modelo de ajuste linear

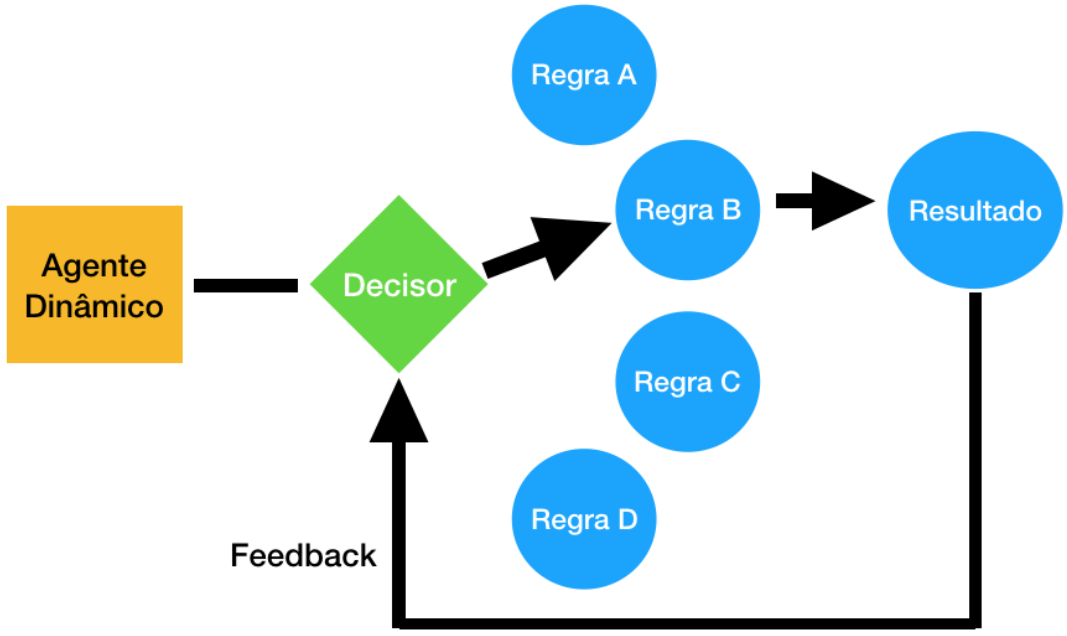
Fonte: Adaptado de Silva, Silva e Chaimowicz (2017), p.20

Se o desempenho p do usuário na atividade proposta pelo jogo for superior a β, uma constante definida em 20% acima do desempenho médio, será ajustada a dificuldade para

o nível imediatamente superior. Se o desempenho for inferior a $-\beta$, será ajustada a dificuldade para o nível imediatamente inferior (SILVA; SILVA; CHAIMOWICZ, 2017). Este modelo não leva em consideração a natureza dos desafios e competências, sendo incapaz, então, de discernir entre dois usuários que apresentem baixo desempenho por motivos diferentes.

A proposta de Spronck, Sprinkhuizen-Kuyper e Postma (2004) consiste em um modelo baseado em regras, representado na Figura 3, no qual cada agente do jogo, ou componente inteligente capaz de tomar decisões, escolhe um curso de ação a partir de um conjunto previamente definido de possibilidades de acordo com o as ações realizadas pelo jogador. Diferentemente do modelo linear, este é capaz avaliar separadamente cada aspecto que compõe o desempenho de um jogador, podendo então propor alterações à dificuldade específicas às competências do usuário.

Figura 3: Modelo de ajuste baseado em regras



Fonte: elaborado com base em Spronck, Sprinkhuizen-Kuyper e Postma (2004)

A cada iteração de tomada de decisões o Agente Dinâmico, por meio do seu componente Decisor, escolhe uma regra dentre todas as existentes (no caso ilustrado, a Regra B é a escolhida) e então realiza as alterações na dificuldade do jogo descritas naquela regra. A escolha feita pelo Decisor é baseada no *feedback* que este recebe ao final de cada iteração

sobre o desempenho do jogador. Caso este *feedback* indique melhoria no desempenho, o Decisor passa a considerar a regra escolhida como sendo mais fácil, ou contendo elementos de maior domínio do jogador. A cada nova iteração, uma nova regra pode ser escolhida, de acordo com as novas observações obtidas.

Andrade et al. (2005) apresentam um modelo baseado em aprendizado de máquina no qual agentes aprendem através de iterações do jogo, isto é, os agente jogam contra outros agentes ou contra jogadores humanos a fim de explorar diferentes estratégias e criar padrões de comportamento emergentes, moldados pelo sucesso obtido em tentativas anteriores.

2.5 Aprendizado de Máquina

Algoritmos de dificuldade adaptativa não lineares, como o proposto por Andrade et al. (2005), necessitam da capacidade de tomar decisões com base em observações feitas sobre diferentes elementos e métricas do jogo, como pontuação, tempo e taxa de acertos (HUNICKE, 2005; SILVA; SILVA; CHAIMOWICZ, 2017). Esta capacidade de decisão pode ser dada a um sistema de jogo digital por meio de técnicas de Aprendizado de Máquina.

Aprendizado de Máquina ou *Machine Learning* (ML) é uma área de Inteligência Artificial que tem como objetivo dar a sistemas computacionais a habilidade de desenvolver novas técnicas e melhorar seu desempenho na realização de dadas tarefas sem a necessidade de serem diretamente programados para obter estas competências. Esta habilidade é descrita pelo termo *aprender*, assemelhando o processo ao realizado por seres humanos (MICHALSKI; CARBONELL; MITCHELL, 2013).

Historicamente, sistemas computacionais foram empregados para tratar problemas cujas soluções podiam ser adequada e completamente descritas por meio de algoritmos, ou sequências de instruções finitas e objetivas, não sendo possível lidar com problemas que não se enquadrem nesta descrição. Ao dispensar a necessidade de um algoritmo programado aplicações em Aprendizado de Máquina são, então, capazes de explorar tarefas cujas soluções não podem ser descritas por algoritmos tradicionais. Esta capacidade é garantida a este tipo de aplicação pois o processo empregado para encontrar soluções é estatístico:

um sistema de Aprendizado de Máquina busca em padrões em conjuntos de dados, inferências a partir de exemplos de soluções válidas e inválidas para o problema e *feedback* para tentativas de resolução o conhecimento necessário para resolver de forma adequada o problema; ao invés de instruções, estes sistemas trabalham sobre dados (ALPAYDIN, 2009).

Comumente, sistemas de Aprendizado de Máquina apresentam dois momentos distintos no seu ciclo de vida: o treinamento, ou aprendizado, e a aplicação. Durante o aprendizado, o sistema recebe dados relativos ao problema que foi construído para resolver e opera sobre estes a fim de encontrar padrões, ou, em analogia ao processo de aprendizado humano, "se familiarizar"com o problema. Na aplicação, o sistema utiliza deste aprendizado para fazer inferências, categorizações ou predições em novos dados, diferentes daqueles encontrados durante o treinamento (MICHALSKI; CARBONELL; MITCHELL, 2013; ALPAYDIN, 2009).

Dentro do campo de Aprendizado de Máquina, diferentes abordagens podem ser classificadas de acordo com o tipo de aprendizado que ocorre e as técnicas empregadas. As próximas sessões introduzem três tipos de aprendizado existentes nesta área de estudos, categorizados de acordo com o tipo de *feedback* fornecido ao sistema durante seu treinamento (RUSSELL; NORVIG, 2016; JAIN; MAO; MOHIUDDIN, 1996).

2.5.1 Aprendizado Supervisionado

O paradigma de Aprendizado Supervisionado é caracterizado pelo fornecimento de *feedback* para cada valor de treinamento alimentado ao sistema. Neste modelo, o sistema computacional recebe durante a etapa de aprendizado um conjunto de dados de treinamento na forma de n pares de valores de entrada x e saída y:

$$(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$$

onde cada y_i é resultado da função, até então desconhecida, $y = f(x)$. O sistema deve então fazer uso destes dados para obter uma função-hipótese h que aproxime f (RUSSELL; NORVIG, 2016).

O aprendizado ocorre então conforme o sistema testa diferentes possíveis funções para

h. O caráter supervisionado deste paradigma é dado pela possibilidade de comparação dos valores de saída $y_{hipotese}$ obtidos em cada iteração do treinamento com os valores de saída conhecidamente corretos y.

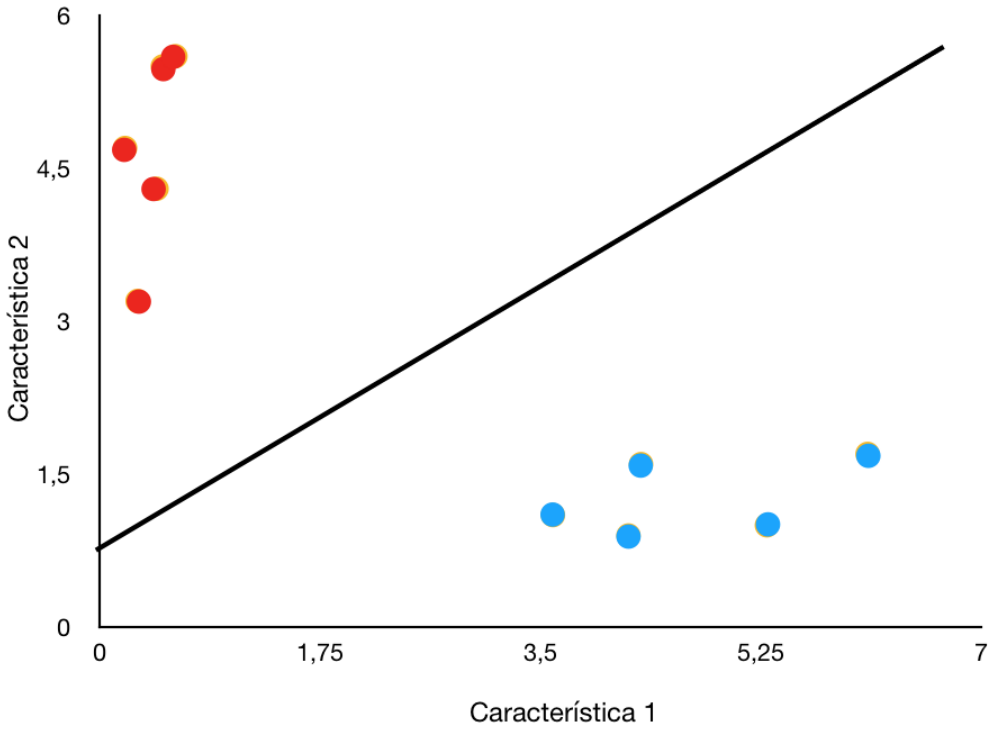
Problemas de Aprendizado Supervisionado são divididos em duas categorias, de acordo com a natureza dos valores de saída y. Se estes forem discretos e finitos, trata-se de um problema de classificação, em que se busca enquadrar cada valor de entrada x para um dos possíveis valores de saída. A classificação binária é um caso particular em que y assume apenas dois valores. Caso y assuma valores numéricos contínuos e, assim, infinitos, trata-se de um problema de regressão, em que o objetivo é aproximar uma estimativa para y dado x, com base na semelhança entre a função encontrada h e a função real, desconhecida, f (RUSSELL; NORVIG, 2016).

Em termos de implementações, o Perceptron, desenvolvido por Rosenblatt (1957), é um algoritmo de Aprendizado Supervisionado que serve de base para novas implementações até hoje. O algoritmo do Perceptron é um classificador binário construído como uma analogia digital ao neurônio humano, capaz de categorizar valores de entrada em uma de duas possíveis classes linearmente separáveis. O aprendizado do Perceptron, sendo Supervisionado, se dá através da iterativa estimativa de uma função h que seja capaz de corretamente separar os valores de entrada do conjunto de teste, x, nas duas dadas categorias para valores de saída, y (MINSKY; PAPERT, 2017; LIOU; LIOU; LIOU, 2013).

A Figura 4 ilustra um exemplo de classificação linear realizada por meio de um algoritmo do tipo Perceptron. Cada elemento classificado é descrito por um vetor de características - no caso, de apenas duas dimensões, para fins de simplificar a visualização, mas podendo conter qualquer quantidade de dimensões - e disposto no espaço de acordo com estas características. O Perceptron então busca uma função linear que consiga dividir os elementos de acordo com sua classe, o valor de saída y, representada na figura pela cor dos elementos. A expectativa então criada é de que o Perceptron, uma vez treinado para ser capaz de adequadamente classificar os dados no conjunto de treinamento, consiga também corretamente classificar novos dados de mesma natureza, ou seja, para os quais a categoria correta y seja dada por $y = f(x)$, para um mesmo f empregado nos dados do conjunto de treinamento.

Para realizar tanto aprendizado quanto inferência, o Perceptron emprega a estrutura

Figura 4: Exemplo de classificação linear via Perceptron



Fonte: elaborado pelo autor

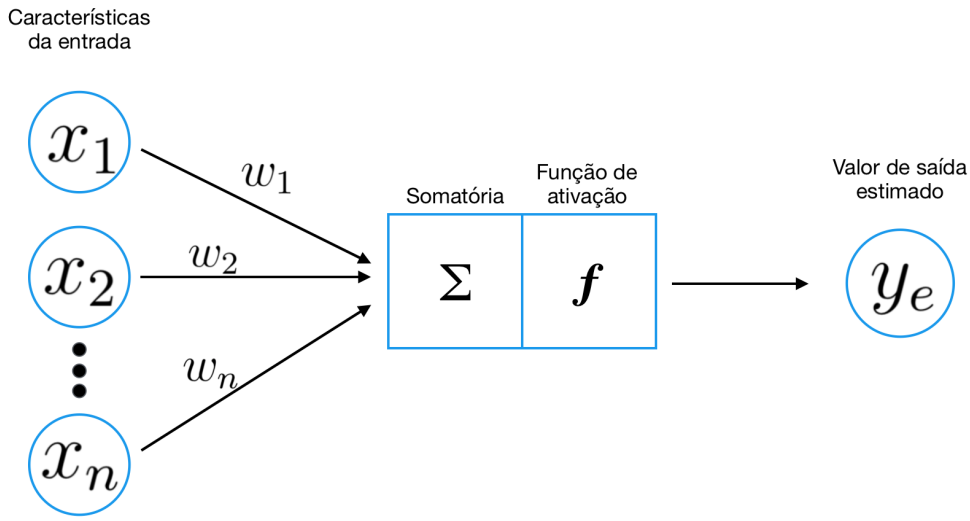
de neurônio artificial ilustrada na Figura 5. Para cada elemento, por exemplo, no conjunto de treinamento, é calculado seu potencial de ativação u a partir da somatória dos valores de todas as suas características x multiplicados por pesos w:

$$u = \sum_{k=1}^{n} x_k . w_k$$

Este potencial de ativação é então fornecido como entrada para a função de ativação do Perceptron, que tem como saída uma das duas possíveis categorias. O resultado é então a classificação do elemento de acordo com a função estimada pelo Perceptron.

Durante o processo de aprendizado, é calculado ainda o erro do valor de saída encontrado, comparando este ao valor de saída conhecidamente correto fornecido no conjunto de dados de treinamento. Este valor de erro é utilizado para alterar os valores de cada peso w_k, a fim de aumentar a acurácia das próximas estimativas, conforme descrito pela equação:

Figura 5: Estrutura do neurônio no Perceptron



Fonte: elaborado pelo autor com base em Minsky e Papert (2017)

$$w_k(t+1) = w_k(t) + r.(y - y_e(t)).x_k$$

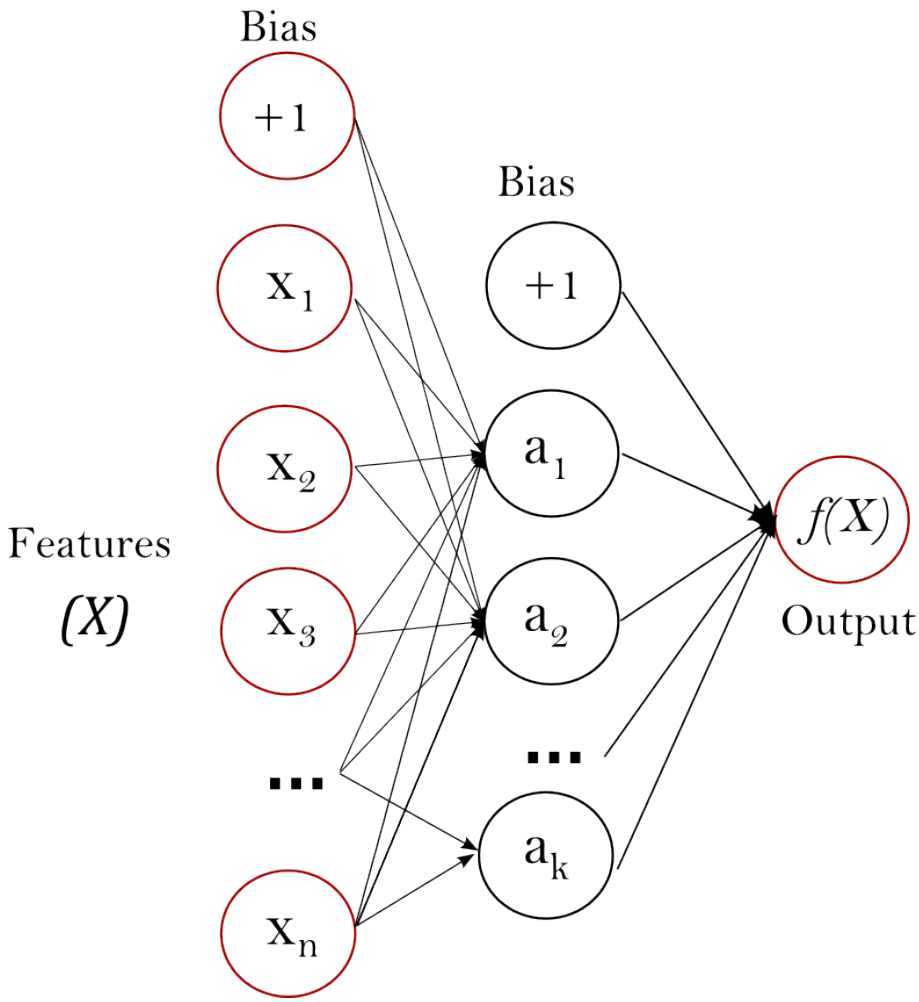
Onde t é o índice da iteração atual, y é o valor de saída conhecidamente correto, $y_e(t)$ é a saída encontrada pelo Perceptron nesta iteração do aprendizado e r é a taxa de aprendizado, valor que define o quão rapidamente ocorre a alteração dos pesos ao longo do treinamento. A evolução dos pesos de um neurônio pode ser vista como a representação do desenvolvimento de seu "aprendizado": é na combinação destes valores que existe a capacidade de inferir decisões sobre novos dados. Este processo de atualização dos pesos se repete até que o Perceptron apresente uma taxa de erro inferior à um valor dado como parâmetro para a rede ou que se chegue ao número máximo de iterações de treinamento estabelecido.

Outros algoritmos implementando Aprendizado Supervisionado existem, com características similares ao Perceptron, Como as SVM (*Support Vector Machines*) e as redes neurais no estilo MLP (*Multi Layer Perceptron*), com capacidades diferentes.

O MLP, enquanto modelo de Aprendizado Supervisionado, consiste em um tipo Rede Neural Artificial, uma estrutura computacional criada como analogia ao sistema nervoso central biológico. Em uma rede neural artificial, múltiplos neurônios artificiais como o

ilustrado na Figura 5 são dispostos em camadas ordenadas, de forma que o valor de saída calculado pelos neurônios de uma camada são passados como valores de entrada para os neurônios da camada seguinte, conforme ilustrado na Figura 6, que descreve um MLP contendo exatamente três camadas: a de entrada, que contém os valores das características $x_1, x_2, .., x_n$ de cada elemento x, uma camada chamada escondida, que recebe estes valores de entrada, processa em cada neurônio a, e repassa para a camada de saída, que também contém um ou mais neurônios com função de ativação capaz de associar ao elemento x um valor de saída (RUMELHART; HINTON; WILLIAMS, 1985). Cada círculo em preto na imagem bem como o círculo da camada de *output* representa um neurônio, conforme a Figura 5.

Figura 6: Estrutura genérica de MLP



Fonte: Pedregosa et al. (2011)

Um MLP pode conter qualquer número de camadas escondidas, bem como qualquer

número de neurônios em suas camadas escondidas. Este número de neurônios pode também variar entre as diferentes camadas escondidas de um mesmo MLP. O número de neurônios da camada de saída varia de acordo com o tipo de dado assumido pelos valores de saída.

Assim como o Perceptron, o aprendizado de uma rede MLP se dá na evolução dos pesos sinápticos que existem entre cada conexão de neurônios da rede, que ocorre a partir da medição do erro encontrado durante o treinamento. Diferentemente do Perceptron, porém, os pesos de um MLP não podem ser todos alterados a partir de um único fator baseado no erro, uma vez que cada peso em cada neurônio tem uma contribuição diferente para o erro final no valor de saída encontrado. Propõe-se então como alternativa o emprego do método do gradiente para propagar o erro para todos os pesos existentes na rede e executar sua atualização (RUMELHART; HINTON; WILLIAMS, 1985).

Este processo, conhecido também como *backpropagation*, incia-se com o cálculo da função de custo, ou do erro, de uma execução da rede neural para um dado elemento do conjunto de dados. Esta função assume, em sua iteração mais simples, a forma de $E = (y\_esperado - y\_encontrado)$, podendo também assumir diferentes outras formas de expressar o erro. Tanto para problemas de regressão quanto de classificação, Hastie, Tibshirani e Friedman (2009) sugere como função de custo a soma dos erros quadrados da rede:

$$E = \sum_{k=1}^{K} \sum_{i=1}^{N} (y\_esperado_{ik} - y\_encontrado_{ik}))^2$$

com $y\_encontrado_{ik}$ para o valor de saída encontrado pelo neurônio de saída k para o valor de entrada x_i e $y\_esperado_{ik}$ o valor conhecidamente correto para esta mesma saída, para todos os valores de entrada considerados em N e todos os neurônios da camada de saída K.

Uma vez conhecida a função de custo obtida ao final da execução da rede, este valor é usado no cálculo do valor de erro relativo à cada neurônio da camada de saída, valor conhecido como delta (δ):

$$\delta_k = E f^{'}(I_k)$$

com $f^{'}(I_k)$ sendo a derivada da função de ativação empregada pelo neurônio k sobre o *input* I_k recebido da camada anterior.

A partir do δ calculado para cada neurônio da camada de saída, pode-se então calcular o δ para cada neurônio h da camada escondida imediatamente anterior à esta, por meio de:

$$\delta_h = f^{'}(I_h) \sum_{k=1}^{K} \delta_k W_{hk}$$

com $f^{'}(I_h)$ sendo a derivada da função de ativação empregada pelo neurônio h sobre o *input* I_h recebido da camada anterior e W_{hk} o valor atual do peso entre os neurônios h, da camada escondida e k, da camada de saída. Esta mesma equação é empregada sucessivamente para cada camada anterior, até a camada de entrada. Para cada neurônio, o valor de δ é calculado em função da derivada de sua função de ativação e do δ de cada neurônio para o qual sua saída é passada como *input* multiplicado pelo peso desta conexão sináptica.

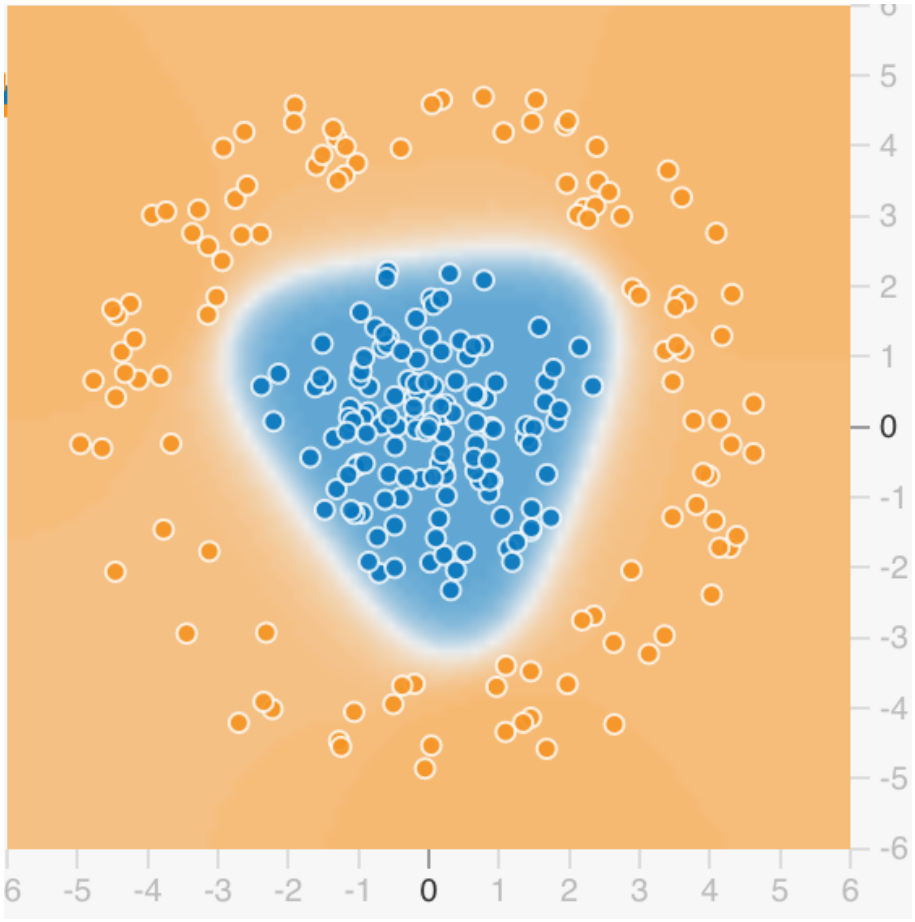
Uma vez definido o valor de δ para cada neurônio da rede, os pesos de todas as conexões são atualizados por conforme:

$$W_{ij} = W_{ij} + \gamma I_j \delta_j$$

para cada conexão entre neurônios i, de uma camada anterior, e j, da camada posterior, em que γ é a taxa de aprendizado e I_j é o *input* recebido por j da camada anterior, ou, no caso da camada de entrada, o valor do *input* passado à rede. Assim como no Perceptron, este processo é repetido até que se chegue a um valor de erro inferior à um parâmetro escolhido ou que se atinja o número máximo de iterações previstas.

Enquanto um Perceptron é capaz de adequadamente dividir elementos em classes linearmente separáveis, uma rede MLP consegue também dividir elementos em classes não linearmente separáveis graças à presença de camadas escondidas e dos pesos sinápticos nestas, conforme ilustrado na Figura 7, onde uma rede com uma única camada escondida consegue adequadamente dividir as duas classes (representadas por pontos azuis e laranjas).

Figura 7: Exemplo de classificação não linear com MLP de uma camada escondida



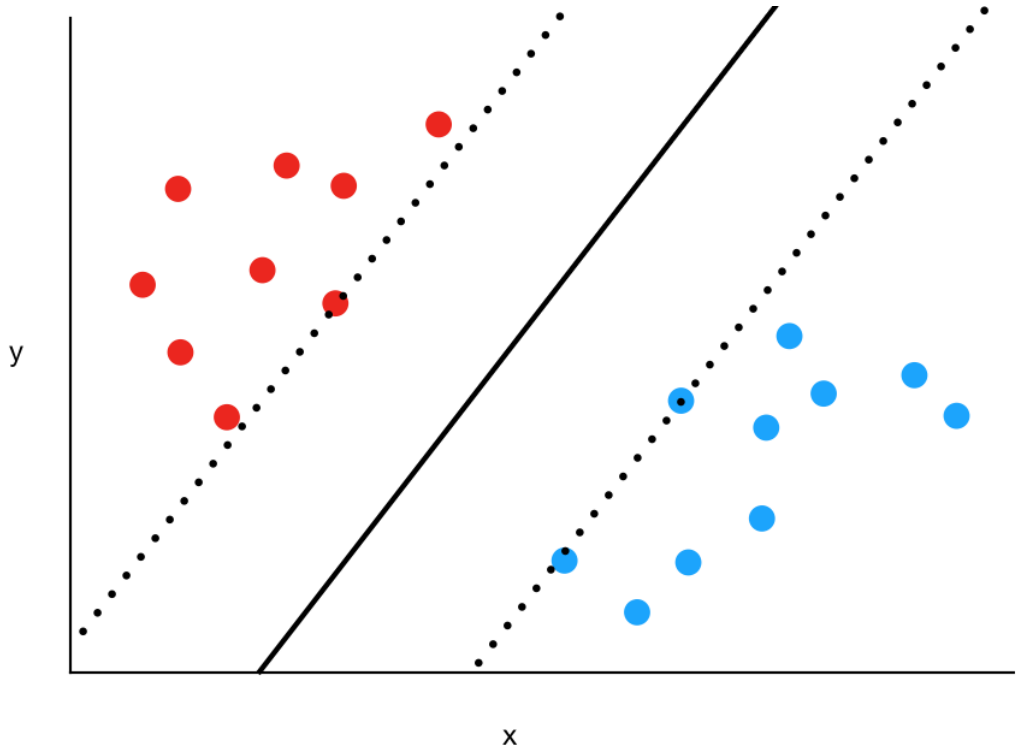
Fonte: Criado com TensorFlow (ABADI et al., 2015)

Uma outra alternativa de Aprendizado Supervisionado, um algoritmo SVM realiza a classificação de elementos em categorias encontrando a maior margem entre as classes no espaço definido pelas características dos elementos. Diferentemente de outros algoritmos como o Perceptron ou o MLP, uma SVM garante que, caso as classes do domínio do problema sejam linearmente separáveis, será encontrada a maior margem entre estas.

A Figura 8 ilustra um caso de uma SVM treinada com dados linearmente separáveis em duas classes distintas, representadas pelas cores dos pontos. A linha contínua entre as classes é chamada hiperplano ótimo ou de margem máxima, pois se encontra na máxima distância da envoltória convexa dos elementos das duas diferentes classes (HEARST et al., 1998; FREUND; SCHAPIRE, 1999).

Esta característica de SVM pode ser contrastada com a forma como outros algoritmos, como o Perceptron, realizam a separação linear de grupos de elementos, podendo estes

Figura 8: Separação de classes por SVM, encontrando a maior margem possível

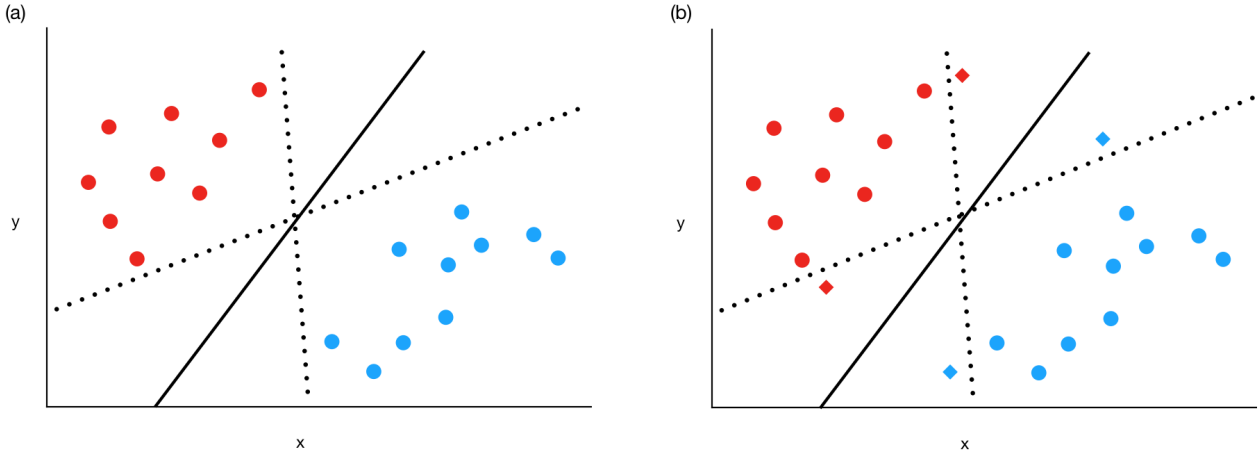


Fonte: elaborado pelo autor com base em Hearst et al. (1998)

encontrarem qualquer uma das infinitas margens possíveis entre as classes. A Figura 9 exemplifica um caso de margem encontrada por SVM (em linha contínua) e duas margens encontradas, por exemplo, por Perceptron (em linha pontilhada), a partir do conjunto de treinamento mostrado em (a). Em (b), simula-se uma situação em que os dados que os algoritmos encontram após o seu treinamento, ilustrados na forma de losangos, não se encaixam nas divisões feitas pelo Perceptron, apesar de estarem espacialmente próximos dos demais exemplares de suas classes. A divisão feita pela SVM, por outro lado, consegue adequadamente categorizar os novos pontos (FREUND; SCHAPIRE, 1999).

Embora inicialmente um algoritmo para classificação linear, uma SVM pode ser empregada em problemas de classificação não linear por meio do emprego do chamado truque do *kernel*, em que a dimensionalidade dos pontos de dado no espaço de características é aumentada (em vezes ao infinito) até que as classes se tornem linearmente separáveis (BOSER; GUYON; VAPNIK, 1992).

Figura 9: Comparação da margem encontrada por SVM (linha contínua) com possíveis margens encontradas por Perceptron (linhas pontilhadas)



Fonte: elaborado pelo autor

2.5.2 Aprendizado Não Supervisionado

No paradigma de Aprendizado Não Supervisionado, em contraste, não é fornecida qualquer informação a respeito da saída esperada durante o treinamento do sistema de Aprendizado de Máquina. Os algoritmos deste tipo tratam de identificação de *clusters* ou padrões em dados, ou de cálculo de densidade e distribuição de elementos num dado espaço. Diferentemente de métodos de Aprendizado Supervisionado, em que é possível identificar uma taxa de erro durante o treinamento e obter conclusões acerca da acurácia de uma dada implementação, técnicas de Aprendizado Não Supervisionado estão limitadas a análises heurísticas no que diz respeito a medir seu sucesso (HASTIE; TIBSHIRANI; FRIEDMAN, 2009).

Uma técnica de Aprendizado Não Supervisionado é o algoritmo de Lloyd (LLOYD, 1982), comumente chamado de algoritmo *k-means*. Este tem como objetivo agrupar todos os elementos fornecidos em um de k grupos, ou *clusters*, de acordo com a distribuição destes elementos no espaço definido por suas características. Este processo se dá de forma iterativa: inicialmente, são escolhidos k (valor este definido como parâmetro do algoritmo) elementos aleatórios do conjunto de dados para representarem o centro de cada grupo. A partir desta organização inicial aleatória, cada elemento do conjunto de dados é então associado ao *cluster* até cujo centro tenha a menor distância euclidiana

quadrada, conforme descrito pela equação:

$$c_x = \text{argmin}_{c_i \in C}\, distancia(c_i, x)^2$$

onde c_x é o centro do *cluster* escolhido para o elemento x, C é o conjunto de todos os k centros e $distancia(q, p)$ é a distância euclidiana $\sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$ entre pontos n dimensionais.

Uma vez que todos os elementos são associados ao *cluster* de centro mais próximo, a posição do centro c de cada *cluster* é recalculada como o centroide definido por todos os elementos associados a este, por meio de:

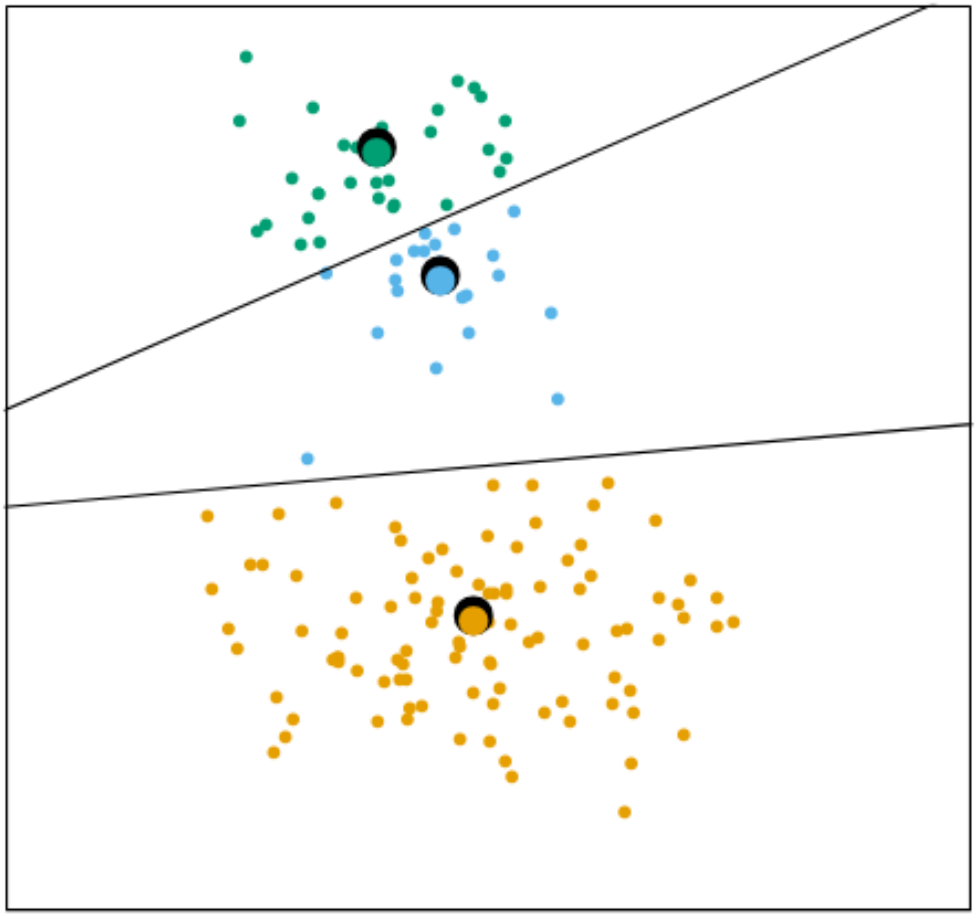
$$c_i = \frac{1}{|P_i|} \sum_{x_i \in P_i} x_i$$

onde P_i é o conjunto de todos os elementos associados ao *cluster* i.

Estas duas etapas formam uma iteração do algoritmo, e são repetidas até que não haja qualquer alteração na posição dos centroides dos grupos ou que um número máximo de iterações definido como parâmetro seja atingido. Caso o algoritmo termine por motivo de não haver nova alteração na posição dos centroides, diz-se que houve convergência.

A Figura 10 ilustra o resultado de 20 iterações do algoritmo de Lloyd sobre um conjunto de dados cujas características estão descritas como as duas dimensões do espaço. Os elementos do conjunto de dados estão então classificados em três grupos de acordo com a similaridade de suas características. A classificação realizada depende do fornecimento de um valor para k, a quantidade de grupos. Idealmente, este valor deve ser determinado com base nos dados do problema sendo abordado, estratégia viável em casos onde se conhece a natureza dos dados ou se busca segmentação em um número já conhecido de classes. Para outros casos, porém, onde não se conhece o valor ideal de k, é necessário fazer uso de alguma técnica para sua escolha. Hastie, Tibshirani e Friedman (2009) explica uma abordagem baseada em sucessivas aplicações do algoritmo para diferentes valores de k, analisando em cada caso o valor de dissimilaridade em cada *cluster*, nomeado w. Quanto menor o valor de w, mais similares são os elementos dentro de cada *cluster* encontrado pelo algoritmo, ou, em termos de sua representação visual no espaço, mais próximos são os pontos dentro de cada grupo.

Figura 10: Definição de *clusters* via *k-means*



Fonte: Hastie, Tibshirani e Friedman (2009)

O objetivo neste método, porém, não é minimizar o valor de w. Intuitivamente tem-se que quanto maior o valor de k, menor o valor de w, ao extremo em que o valor de w tende a zero conforme k se aproxima do número de elementos no conjunto de dados. A fim de obter agrupamentos relevantes, este método busca então aproximar k de um valor desconhecido k^*, que represente a distribuição natural ideal dos elementos. Para isso, o autor apresenta duas considerações acerca do algoritmo *k-means*:

- Para $k < k^*$, todos os k^* grupos naturalmente formados nos elementos são mantidos; a perda de acurácia se dá inteiramente pelo agrupamento de dois ou mais *clusters* em um único no resultado do algoritmo
- Para $k > k^*$, pelo menos um dos k^* grupos naturalmente formados é separado entre pelo menos dois dos *clusters* resultantes do algoritmo

Com base nisso, o autor afirma que é valido:

$$\{w_k - w_{k+1} | k < k^*\} \gg \{w_k - w_{k+1} | k \geq k^*\}$$

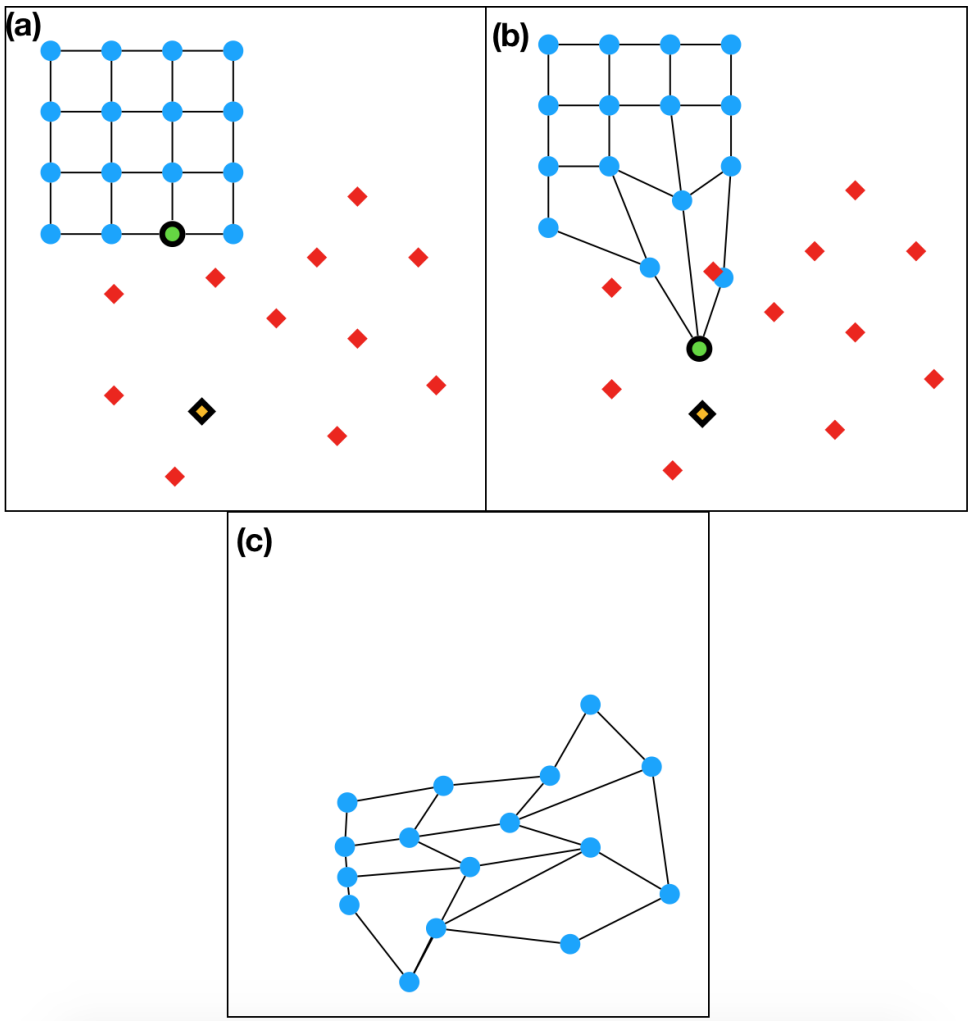
ou seja, enquanto $k < k^*$, aumentar o valor de k provoca uma redução em w significativamente maior do que aumentar o valor de k quando $k > k^*$, uma vez que separar dois grupos naturais em seus próprios *clusters* resulta em maior redução de w do que subdividir grupos naturais em mais *clusters*. De maneira heurística, é então possível identificar k^* como o ponto onde, numa sequência de execuções do algoritmo *k-means* com valores cada vez maiores de k, o valor de w é reduzido significativamente menos do que nas iterações anteriores. A fim de eliminar a necessidade desta identificação subjetiva de "redução significativamente menor"de w, são sugeridas técnicas como o *Gap* (TIBSHIRANI; WALTHER; HASTIE, 2001).

Uma outra abordagem dentro do paradigma de Aprendizado Não Supervisionado é o Mapa Auto Organizado, ou Rede de Kohonen (KOHONEN, 1990). Este tipo de rede neural artificial emprega princípios similares ao algoritmo de Lloyd para organizar elementos de um conjunto de dados em um espaço bidimensional, onde a posição dos elementos é definida por suas características e a proximidade entre estes é equivalente a sua similaridade nestas mesmas características.

Um Mapa Auto Organizado consiste em uma rede de neurônios conectados dispostos em um espaço bidimensional. Estes neurônios, apesar de também fazerem analogia à estrutura neural biológica, são distintos daquele descrito na Figura 5. Ao invés de receber estímulos ponderados por pesos e identificar um valor de saída através de uma função de ativação, estes neurônios realizam o aprendizado por meio de sua posição no espaço bidimensional, seu peso sináptico sendo representado pelo vetor de suas coordenadas. Desta forma, o aprendizado de uma rede deste tipo pode ser descrito como a deformação de seu formato ao longo das iterações de treinamento.

Inicialmente, os neurônios de um Mapa Auto Organizado são dispostos de forma aleatória, ou seja, inicializados com pesos aleatórios, conforme representado no painel (a) da Figura 11, onde os círculos azuis são neurônios e os losangos vermelhos são elementos do conjunto de dados. A cada iteração, um dos elementos do conjunto de dados é selecionado, representado na imagem como o losango amarelo com contorno destacado. Busca-se então aquele neurônio dentre todos os da rede que apresente menor distância euclidiana do

Figura 11: Aprendizado de um Mapa Auto Organizado



Fonte: elaborado pelo autor

elemento escolhido. Este neurônio é denominado BMU, do inglês *best matching unit*. A BMU na ilustração é representada pelo círculo verde com contorno destacado.

Uma vez encontrada a BMU, é realizada a deformação da rede por meio da alteração dos pesos da BMU, bem como de seus neurônios vizinhos. Esta alteração de pesos w se dá por:

$$w_v(t+1) = w_v(t) + \theta(u, v, t).\alpha(t).(D(x) - w_v(t))$$

para cada peso w de cada neurônio v, dada a iteração atual t, uma função de vizinhança θ, a BMU encontrada u, uma taxa de aprendizado α e o elemento do conjunto de

dados escolhido $D(x)$.

Esta operação é realizada sobre todos os neurônios da rede, mas, conforme representado pelo painel (b) da Figura 11, nem todos os neurônios são movidos em cada iteração. Isto se deve à função de vizinhança θ, responsável por propagar a alteração realizada na BMU aos seus neurônios vizinhos, a fim de manter as características topográficas da rede. Para neurônios v que não sejam vizinhos da BMU, θ será 0, e não ocorrerá qualquer mudança em suas posições.

Tanto a função de vizinhança θ quanto a taxa de aprendizado α são funções de t, a iteração atual. Tipicamente, estas duas funções diminuem em valor conforme t aumenta, a fim de obter convergência (HASTIE; TIBSHIRANI; FRIEDMAN, 2009).

Este processo é então repetido para todos os elementos do conjunto de dados, e então novamente para todos os elementos, sucessivamente, até que se atinja uma condição de parada ou um limite de iterações. O painel (c) da Figura 11 ilustra um possível estado final do treinamento da rede, em que os neurônios se organizaram sobre as observações do conjunto de dados. Uma vez atingido este estado, cada novo dado apresentado à rede será associado ao neurônio que mais se assemelhe às suas características, posicionando este com relação a todas as demais observações de dados de acordo com a similaridade de suas características.

Enquanto no algoritmo de Lloyd é necessário escolher um valor k para o número de grupos a serem formados, para um Mapa Autor Organizável é necessário escolher a quantidade de neurônios da rede. Existem diferentes recomendações para esta escolha: Kohonen (1990), por exemplo, sugere um neurônio para cada 50 observações no conjunto de dados, enquanto Vesanto e Alhoniemi (2000) propõe um número k de neurônios tal que $k = 5\sqrt{n}$, onde n é a quantidade de observações no conjunto de dados.

2.5.3 Aprendizado Por Reforço

Um terceiro paradigma é o chamado de Aprendizado Por Reforço. Sutton e Barto (1998) definem este como o processo de aprender o que fazer numa dada situação, mapeando uma ação a esta. Neste contexto, o aprendizado é descrito em termos de competências de um agente, entidade de software que realiza a tomada de decisões. Este agente

existe em um ambiente descrito em estados, e pode tomar ações que resultam em alterações de estado. Diferentemente do Aprendizado Não Supervisionado, onde não existe *feedback* e do Aprendizado Supervisionado onde é fornecido o valor de saída correto para cada valor de entrada durante o treinamento, um agente no Aprendizado Por Reforço recebe, a cada iteração de treinamento, um valor de recompensa, que mede o nível de adequação da ação tomada com o resultado ideal. Ações que resultam em estados finais mais desejáveis recebem recompensas maiores do que ações que resultam em estados finais menos desejáveis. O aprendizado se dá então no desenvolvimento da metodologia que o agente emprega para mapear ações para estados de seu ambiente a fim de maximizar a recompensa obtida.

Esta metodologia é chamada de *policy*, ou, em tradução grosseira, a "política"empregada pelo agente para tomar decisões. Esta pode tomar a forma de uma função, uma tabela, ou qualquer outro método que, tendo como entrada o estado atual do ambiente, retorne como saída uma ação a ser tomada. Esta *policy*, conforme dito, evolui ao longo do treinamento do agente de acordo com as recompensas que este recebe.

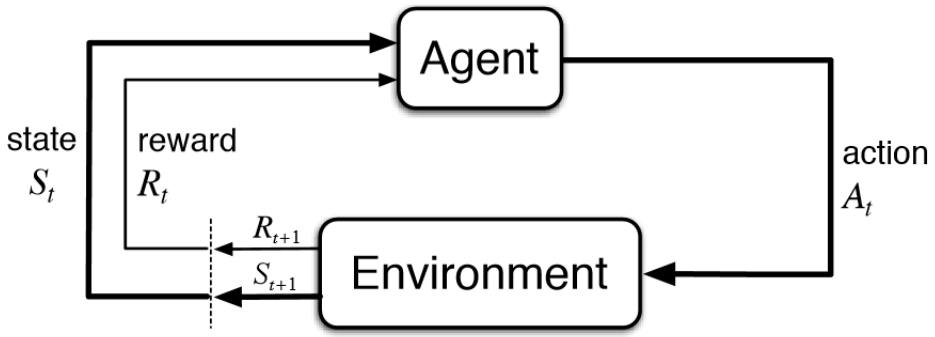
A recompensa dada ao agente depende de uma função de recompensa imutável que mapeia, para cada estado, um valor de recompensa. Para um agente aprendendo a jogar xadrez, por exemplo, uma função de recompensa adequada deve fornecer uma recompensa baixa para estados em que este é colocado em xeque, uma recompensa mais alta para estados em que este coloca o oponente em xeque e a recompensa máxima para estados em que este coloca o oponente em xeque-mate.

A fim de garantir ao agente de Aprendizado por Reforço a capacidade de planejar estados futuros ao invés de simplesmente buscar o estado que imediatamente pareça mais adequado a cada instante, existe também uma função de valor estimado. Esta mapeia para cada estado não a recompensa imediata que este gera, mas sim o potencial de recompensa esperada a longo prazo a partir deste estado (SUTTON; BARTO, 1998). No exemplo do agente enxadrista, um estado em que este é colocado em xeque pode ter uma recompensa instantânea (conforme dada pela função de recompensa) baixa. Se este mesmo estado, porém, possibilitar ao agente sair de xeque e dar xeque-mate no oponente na próxima jogada, então este tem um valor estimado alto.

O comportamento da função de recompensa para cada estado é imutável e escolhido

antes do início do aprendizado, e depende do domínio do problema, como no caso do jogo de xadrez, em que dar xeque-mate é o estado de maior recompensa possível. A função de valor estimado, por outro lado, evolui ao longo das iterações de treinamento, e seus valores iniciais muitas vezes são escolhidos aleatoriamente ou com base em estimativas informadas acerca do domínio do problema.

Figura 12: Modelo de tomada de decisões agente-ambiente



Fonte: Sutton e Barto (1998)

A evolução do valor estimado de um estado ocorre quando o agente explora este estado durante o aprendizado e verifica empiricamente a recompensa que eventualmente recebe, conforme ilustrado na Figura 12. Mais especificamente, quando o agente passa de um estado para outro, este atualiza o valor estimado do estado anterior $V(s)$ para que se aproxime do valor estimado do estado atual $V(s^{'})$, seguindo o modelo geral:

$$V(s) < -V(s) + \alpha(V(s^{'}) - V(s))$$

onde s é o estado anterior, $s^{'}$ é o estado atual e α é um parâmetro análogo à taxa de aprendizado de outros modelos de Aprendizado de Máquina.

O Aprendizado por Reforço consiste, então, em oferecer ao agente a oportunidade de praticar a escolha de ações e a navegação por estados para descobrir quais decisões levam às melhores recompensas (SUTTON; BARTO, 1998).

Quando um agente, de acordo com sua *policy*, escolhe a próxima ação a ser tomada como aquela que leva ao estado de maior valor estimado, diz-se que este *tomou vantagem* da situação, em tradução do inglês *exploit*. Este tipo de decisão leva em consideração

apenas a maximização do valor estimado e, em consequência, da recompensa. Entretanto, como a função de valor estimado não é perfeita e depende de sucessivas iterações diferentes para ser aprimorada, um agente que sempre toma decisões do tipo *exploit* corre o risco de encontrar um máximo local, ignorando alternativas não exploradas que levariam ao máximo global.

Quando, por outro lado, um agente escolhe a próxima ação como aquela que leva a um estado ainda pouco conhecido a fim de aprimorar a função de valor estimado, este está *explorando*, do inglês *explore*. Este tipo de decisão permite ao agente aprimorar a função de valor estimado para diferentes estados, enriquecendo sua capacidade de prever recompensas em diferentes momentos e situações do ambiente. Um agente que sempre toma decisões do tipo *explore*, porém, não faz proveito deste aprendizado, e pode nunca obter uma boa recompensa por estar sempre buscando novas experiências.

Esta dicotomia entre *exploit* e *explore* é um dos desafios de abordagens de Aprendizado por Reforço apontados por Sutton e Barto (1998). Os autores sugerem o uso de uma taxa variável monótona descendente de exploração: para a tomada de decisões de cada iteração t, a probabilidade do agente escolher uma opção exploratória é dada por ϵ, cujo valor decresce conforme t aumenta. Desta forma, o agente explora diferentes decisões nas primeiras iterações e converge para aquelas mais promissoras com o tempo.

2.6 Trabalhos correlatos

Vistas as motivações para o emprego de técnicas de adaptabilidade de dificuldade para acompanhar a evolução de competência do jogador e garantir maior engajamento, casos de desenvolvimento de jogos adaptativos foram estudados. Com base na hipótese de que os fatores que aumentam o aproveitamento de um jogo puramente de entretenimento sejam similares aos que proporcionam eficácia para um jogo educacional, foram considerados também jogos não educacionais neste levantamento. Com o objetivo também de estabelecer comparações ou aproximações com outras soluções existentes para o ensino da matemática por meio de jogos, foram considerados jogos digitais não adaptativos dentro desta área.

Sampayo-Vargas et al. (2013) realizam um estudo no qual três objetos de ensino para

vocabulário em espanhol são comparados: um jogo que implementa dificuldade adaptativa, um jogo com dificuldade não adaptativa, e um exercício textual de aula. A comparação entre objetos adaptativos e não adaptativos assemelha o trabalho dos autores ao presente estudo, enquanto a comparação também com uma atividade textual tradicional é ponto de divergência. A variação da dificuldade nas atividades dos autores é dada pela quantidade de opções dentre as quais o aluno deve escolher a resposta correta. Testes com estudantes indicam que não houve diferença significativa entre os jogos com e sem dificuldade adaptativa em relação à motivação percebida pelos alunos, mas que o jogo adaptativo foi eficaz em termos de objetivos de aprendizado: alunos que jogaram esta versão demonstraram ter aprendido 7 termos, em média, comparados aos 3.3 da versão não adaptativa. Os autores ressaltam que a ausência de efeito mensurável na motivação possa se dever a forma como este exemplo foi implementando, argumentando que, de acordo com os alunos, a versão adaptativa não foi capaz de oferecer o nível de dificuldade desejado.

O estudo realizado por Silva, Silva e Chaimowicz (2017) traz como objetivo a implementação de um agente de inteligência artificial capaz de jogar o jogo *DotA* com um nível variável de competência, de forma a oferecer sempre uma experiência de desafio adequada ao nível de habilidade do jogador, objetivo alinhado com o do presente estudo, visando garantir engajamento por meio do ajuste da dificuldade. O trabalho se distancia do presente ao não contemplar em seu escopo um jogo como objeto de ensino. Testes realizados comparando um agente dinâmico com um agente estático apontaram, com base na opinião expressa pelos jogadores, uma preferência pela dificuldade apresentada pelo agente dinâmico em 85% dos casos. Os autores notam ainda que, para jogadores de nível de habilidade considerado 'experiente', o agente dinâmico foi incapaz de desenvolver estratégias suficientemente sofisticadas para apresentar real desafio.

Andrade et al. (2005) empregam o paradigma de Aprendizagem por Reforço para criar um agente dinâmico capaz de jogar um jogo de luta com nível variável de habilidade. Este agente é treinado contra oponentes de diferentes níveis de competência e, ao lutar contra um jogador, determina o nível de habilidade deste, e limita sua tomada de decisões para que represente um desafio condizente. Este trabalho também apresenta objetivo similar ao presente estudo, no que diz respeito à manutenção de um nível adequado de desafio para cada jogador, mas distinto ao não analisar um jogo enquanto facilitador do processo

de aprendizado.

Liu et al. (2009) propõe uma abordagem de DDA que leva em consideração não só o desempenho do jogador, mas também seu estado emocional. Empregando sensores para detectar respostas fisiológicas de ansiedade, os autores defendem que a resposta emocional do jogador é tão importante para o ajuste de dificuldade quanto a capacidade de obter sucesso no jogo. Este tipo de abordagem, embora não contemplada no escopo do atual trabalho, corrobora para o entendimento da variação da experiência do jogador provocada pela dificuldade do jogo.

Missura e Gärtner (2009) oferecem uma proposta de adaptatividade em jogos baseada na adequação do perfil de competência de cada jogador à categorias pré-estabelecidas a partir da análise dos padrões de interação de vários jogadores-teste. Este modelo, de acordo com a avaliação realizada pelos autores, é capaz de prever e promover a experiência de jogo adequada para o perfil de um novo jogador em tempo real, conforme este interage pela primeira vez com o sistema. Esta proposta se alinha com os objetivos do presente estudo, apesar da implementação feita pelos autores ser inviável no contexto deste, devido à indisponibilidade dos colégios para a coleta de dados de interação dos alunos antes do experimento.

Sreenivas (2007) apresenta uma proposta de sistema tutor inteligente empregando Aprendizado por Reforço para se adequar as necessidades de cada estudante, sendo capaz de desenvolver abordagens de instrução diferenciadas de acordo com as interações com o usuário. Embora o escopo da aplicação proposta pelo autor seja diferente do que é contemplando no presente trabalho, existe convergência de objetivos no que diz respeito à pesquisa acerca da empregabilidade de técnicas de Aprendizado por Reforço para nortear a experiência do usuário em um sistema de aprendizado, enquanto a principal diferença se dá pela natureza do objeto proposto, que não é um jogo.

3 METODOLOGIA

O estudo realizado teve como objetivo construir um jogo para auxiliar no ensino da matemática no contexto escolar, analisando a empregabilidade de técnicas de dificuldade adaptativa.

A primeira etapa deste processo consistiu na decisão acerca dos conteúdos a serem abordados e forma como estes devem ser apresentados. A partir desta escolha, foi projetado e desenvolvido um primeiro protótipo do jogo, para fins de validação da forma de abordagem do conteúdo. Esta validação foi realizada com educadores da área de Matemática, e o *feedback* obtido foi empregado nas etapas seguintes. O passo seguinte foi a investigação de alternativas para a implementação de dificuldade adaptativa, incluindo testes com diferentes paradigmas e algoritmos de Aprendizado de Máquina. A partir desta investigação, foi decidida a abordagem a ser empregada para adaptatividade e, então, desenvolvida uma nova versão do jogo, funcionalmente completa e adequada para teste com alunos do público alvo, que ocorreu ao final do processo. Os artefatos gerados ao longo do processo, bem como aqueles resultantes deste teste com alunos, formam a base da contribuição deste trabalho.

3.1 Escolha dos conteúdos

Como primeiro passo, foi definido o conteúdo a ser abordado pelo jogo em questão. A matemática foi escolhida por ser uma área de carência no ensino brasileiro: dados do levantamento De Olho nas Metas de 2015 apontam que apenas 42,9% dos alunos no terceiro ano do ensino fundamental da rede pública de ensino detém conhecimento de matemática avaliado como adequado (CRUZ, 2015). Em artigo de 1918 sobre esta tendência, que não se restringe ao ensino público brasileiro ou mesmo ao tempo presente, Merrill (1918) afirma que parte da dificuldade no aprendizado de matemática vem da frustração inerente à complexidade dos conceitos, permitindo a conclusão de que métodos de ensino que proporcionem motivação obtenham sucesso. Ainda abordando a empregabilidade de jogos como objetos de ensino de matemática, Thomaz e Megid (2017) defendem a necessidade de aproximar o conteúdo abstrato desta área de maneira lúdica e atrativa para o aluno, atenuando riscos de desinteresse e tédio.

Dentro da matemática, foram escolhidos como tópicos específicos para ensino neste estudo as operações básicas de aritmética, por serem os estudados no segundo ciclo (que inclui terceira, quarta e quinta séries), etapa escolar da qual se tem os dados comprovando a carência no aprendizado. Mais especificamente, foi escolhido como público alvo deste trabalho o conjunto dos alunos de 7 e 8 anos de idade, estudantes do segundo e terceiro

anos. Para este público específico, as operações trabalhadas foram restritas a adição, subtração e, de forma limitada, multiplicação.

As competências abordadas neste trabalho estão descritas nas habilidades (EF02MA05), (EF03MA06) e (EF03MA07) definidas pela Base Nacional Comum Curricular de 2018 para o terceiro ano do ensino fundamental, todas referentes à Unidade Temática "Números", conforme elencadas na Tabela 1.

Tabela 1: Habilidades abordadas no projeto

Objetos de Conhecimento	Habilidades
Procedimentos de cálculo (mental e escrito) com números naturais: adição e subtração	(EF03MA05)Utilizar diferentes procedimentos de cálculo mental e escrito para resolver problemas significativos envolvendo adição e subtração com números naturais.
Problemas envolvendo significados da adição e da subtração: juntar, acrescentar, separar, retirar, comparar e completar quantidades	(EF03MA06)Resolver e elaborar problemas de adição e subtração com os significados de juntar, acrescentar, separar, retirar, comparar e completar quantidades, utilizando diferentes estratégias de cálculo exato ou aproximado, incluindo cálculo mental.
Problemas envolvendo diferentes significados da multiplicação e da divisão: adição de parcelas iguais, configuração retangular, repartição em partes iguais e medida	(EF03MA07)Resolver e elaborar problemas de multiplicação (por 2, 3, 4, 5 e 10) com os significados de adição de parcelas iguais e elementos apresentados em disposição retangular, utilizando diferentes estratégias de cálculo e registros.

Fonte: Base Nacional Curricular 2018 (BRASIL, 2018)

Apesar de estar presente, de maneira introdutória, no conteúdo previsto para este público alvo, a divisão se ausenta do escopo deste trabalho, que foca nas demais operações com as quais um aluno desta etapa, no geral, tem maior familiaridade. Desta forma, o jogo desenvolvido se enquadra como um de estratégia, dentro da definição de Corbálan (1996), por focar em exercitar habilidades e conhecimentos já obtidos pelos alunos, ao

invés de expor estes a novos aprendizados.

3.2 Desenvolvimento do Primeiro Protótipo

Uma vez escolhido o domínio de conteúdos a serem abordados, foi criado um primeiro protótipo do jogo com o objetivo de validar esta escolha, bem como a forma de apresentação dos conteúdos. Este protótipo ainda visava obter uma análise preliminar dos impactos da dificuldade adaptativa em um jogo educativo para a matemática.

Este primeiro protótipo foi desenvolvido ao longo de um perído de quatro semanas durante os meses de maio e junho de 2018, utilizando a linguagem C# com a plataforma Unity. A escolha da plataforma se deve à possibilidade de gerar versões executáveis em qualquer tipo de dispositivo móvel, computador ou web a partir de um mesmo projeto, capacidade desejável uma vez que não se sabia qual dispositivo estaria disponível para alunos no contexto dos testes que se buscava realizar. Todos os elementos do jogo, incluindo o projeto de software, o código e os elementos gráficos e de interface foram criados pelo autor.

A fim de criar um objeto capaz de proporcionar engajamento aos estudantes que interagem com este, o desenvolvimento do jogo foi guiado pelas definições de motivações intrínsecas de Malone e Lepper (1987). Outros requisitos no desenvolvimento deste objeto foram a minimização de elementos textuais na interface, visto que os alunos do grupo alvo estão ainda em idade de alfabetização; a restrição da interação à comandos de toque apenas, a fim de possibilitar a disponibilização do objeto em plataformas móveis; e a simplicidade do entendimento de seu objetivo, visto que não seria construído um tutorial. Estes requisitos foram levantados com a ajuda dos educadores da área de matemática da Universidade Presbiteriana Mackenzie tendo em mente o público alvo de alunos de terceiro ano do ensino fundamental.

O protótipo desenvolvido faz uso da fantasia e contextualização do conteúdo matemático para promover motivação e interesse nos estudantes (CRAWFORD, 1982; MALONE; LEPPER, 1987). O jogo representa um cenário antártico[1] pelo qual o personagem, um

[1]Neste protótipo inicial, o ambiente do jogo apresentava algumas árvores como elementos de cenário. A fim de proporcionar uma ilustração mais correta do ambiente antártico, estas foram removidas na versão seguinte do jogo.

pinguim, avança. Esta composição representativa do cenário do jogo promove a fantasia, caracterizando o elemento tema de acordo com Järvinen (2009). O personagem se encontra inicialmente em uma margem de um corpo de água, e precisa chegar até a margem oposta. Se conseguir chegar ao outro lado, o personagem avança até novamente encontrar outro corpo de água, e a situação se repete indefinidamente. Cada travessia de água é um desafio do jogo, e a interação com este se dá pelo ciclo destes desafios.

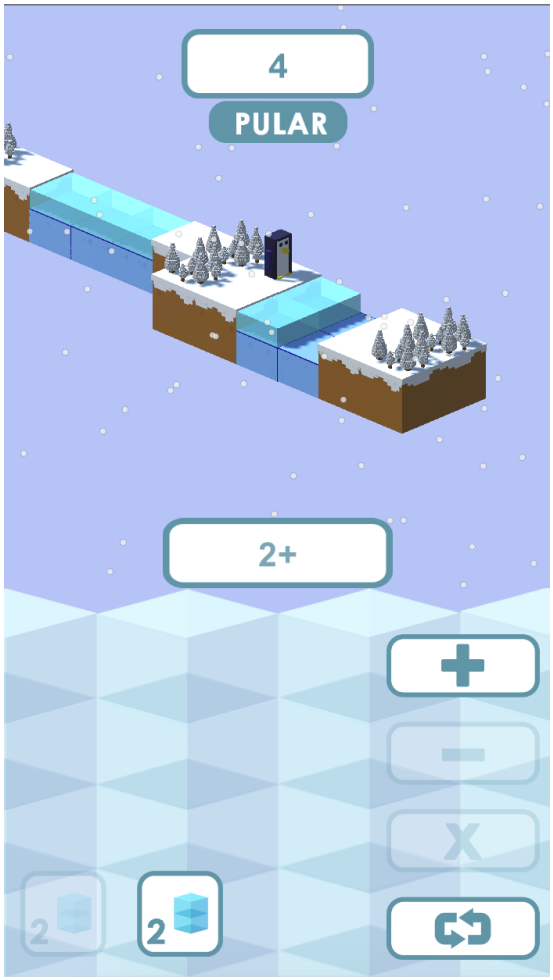
O papel do jogador é auxiliar o personagem a avançar, realizando operações matemáticas para colocar a quantidade precisa de blocos de gelo sobre a água para permitir a passagem segura sobre estes.

Ao criar cada novo desafio como diferente do anterior, crescendo em complexidade matemática de acordo com o modelo de progressão, o jogo busca promover a motivação da curiosidade, privando do jogador o conhecimento específico da próxima situação e promovendo o engajamento. O controle é caracterizado pela agência que o jogador tem sobre o sucesso do pinguim em atingir seu objetivo de cruzar cada corpo de água e da forma como as decisões do jogador provocam alterações imediatas e visíveis no ambiente do jogo, colocando ou retirando blocos de gelo do cenário (MALONE; LEPPER, 1987).

Em termos de fantasia e representação do conteúdo, cada desafio é apresentado como uma quantidade de blocos de água formando um obstáculo que o personagem precisa atravessar. Para auxiliá-lo neste processo, o jogador precisa cobrir toda a água com blocos de gelo. O jogador coloca estes blocos no cenário ao realizar operações aritméticas, utilizando um conjunto dado de números e operadores. O resultado obtido de cada operação é refletido na quantidade de blocos posicionados sobre a água, e o jogador obtém sucesso ao cobrir todos os blocos de água com gelo, sem excesso de blocos: o jogador precisa obter como resultado de sua expressão um valor igual à quantidade de blocos de água presentes. A Figura 13 demonstra o cenário do jogo em uma situação de desafio, bem como a interface por meio da qual o jogador interage com o objeto.

A quantidade de blocos de água no desafio atual, dado conhecido como Valor Alvo do desafio, está sempre visível em uma caixa no topo da tela. Diretamente abaixo deste valor existe o botão *Pular*, que permite ao jogador automaticamente completar o desafio atual e avançar para o próximo. Os números e operadores disponíveis para o jogador no desafio atual estão dispostos na interface de calculadora, ocupando a parte inferior

Figura 13: Visão geral do jogo desenvolvido



Fonte: Mainieri et al. (2018)

da tela, abaixa da visão do jogo. Elementos da calculadora com menor opacidade estão desabilitados, e não podem ser usados. Cada botão de número pode ser utilizado apenas uma vez, sendo desativado após o uso. Embora todos os botões de operadores sempre estejam presentes na calculadora, só aqueles correspondentes às operações disponíveis no desafio atual são ativados. O botão com duas setas, abaixo dos operadores é o botão *Retroceder*, que permite ao jogador remover todos os blocos colocados, reativar todos os botões pressionados e redefinir o resultado atual da expressão como 0 - efetivamente, retornar ao estado inicial do desafio atual. Diretamente acima da interface de calculadora, no meio da tela, encontra-se o campo que descreve a expressão atual que o jogador está construindo. Sempre que um botão de numeral ou operador é pressionado, o elemento correspondente é adicionado à expressão.

Além de ser adicionado à expressão em construção, cada elemento selecionado na interface de calculadora resulta em um efeito visual na visão do jogo. Quando o primeiro número é escolhido, uma quantidade equivalente de blocos de gelo é acrescentada ao cenário, cobrindo blocos de água. Após escolher uma operação e selecionar o segundo operando, a resolução do operação é refletida no cenário. No caso de uma adição, uma quantidade de blocos de gelo igual ao valor do segundo operando é adicionada. Na subtração, esta quantidade de blocos é removida. Não é possível remover mais blocos de que estão presentes, e tentar fazer isso resulta na operação sendo cancelada - não são abordados números negativos no jogo. Multiplicações também resultam no acréscimo de blocos, de acordo com o resultado da operação. Se em qualquer momento forem acrescentados blocos em excesso do Valor Alvo, isto é, se forem adicionados blocos de gelo que excedem a quantidade de blocos de água que necessitam ser cobertos, os blocos excedentes são empilhados naqueles previamente adicionados. A presença de blocos em excesso impede o sucesso do jogador - é preciso atingir exatamente o Valor Alvo, situação que é representada por ter todos os blocos de água cobertos por uma única camada de gelo.

Adotando as definições de Järvinen (2009), podem ser então identificadas as mecânicas presentes no jogo desenvolvido. Estas estão listadas na Tabela 2, que contém, para cada uma, seu nome (em inglês, diretamente retirado da obra original), uma descrição (adaptada para o português a partir do original) e uma explicação de sua aplicação no jogo em questão.

Dentre as mecânicas identificadas, *Allocating* se destaca como a mecânica primária do jogo, responsável por permitir ao jogador atingir o objetivo principal de superar cada um dos desafios propostos. As demais mecânicas se configuram como submecânicas desta, auxiliando no processo executado pelo jogador para montar corretamente cada expressão matemática. As mecânicas do jogo não se alteram ao longo da experiência ou de acordo com a variação da dificuldade oferecida pela jogo: os diferentes níveis de desafio são dados apenas pela variação da complexidade matemática das questões propostas. Desta forma, todas são mecânicas globais do jogo, estando presentes durante toda a interação.

Quando o jogador conclui um desafio, seja por atingir o Valor Alvo com o resultado da expressão montada ou pelo uso do botão *Pular*, o jogo oferece um novo desafio para dar continuidade a experiência do usuário. Cada uma destas fases está na lista de desafios do

Tabela 2: Mecânicas conforme Järvinen (2009) presentes no jogo desenvolvido

Mecânica	Descrição	Aplicação
Allocating	Alocar componentes de jogo para fins específicos	Empregar os números e operadores disponíveis na resolução do desafio
Arranging	Organizar a ordem, composição ou posição de componentes em conjuntos	Ordenar corretamente os operadores e operandos de cada expressão
Choosing	Realizar uma escolha entre múltiplas possíveis opções	Escolher o número ou operação a utilizar, em particular nos desafios onde existem números redundantes
Submitting	Submeter uma informação à avaliação do jogo ou outro jogador	Executar cada passo da expressão sendo construída, ou selecionar o opção de pular

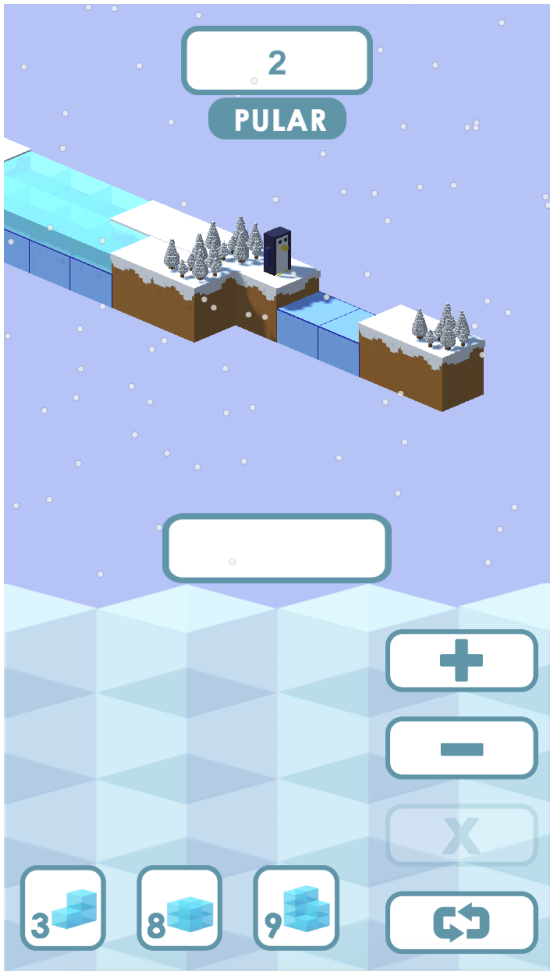
jogo (APÊNDICE A), que contém, em ordem crescente de complexidade matemática, 38 diferentes questões. Esta lista foi construída com o auxilio de alunos e professores do curso de Matemática da Universidade Presbiteriana Mackenzie. O aumento da complexidade de cada desafio se dá pela alteração das operações disponíveis, das operações cujo uso é necessário para atingir o Valor Alvo, e da presença e quantidade de números adicionais, não necessários para a resolução do desafio, na calculadora.

A Figura 14 demonstra um desafio mais complexo do que aquele proposto na Figura 13. Além do desafio disponibilizar tanto a operação de adição quanto a de subtração, sua resolução depende do emprego de ambas as operações em ordem apropriada sobre os números fornecidos.

A progressão de complexidade dos desafios ao longo da lista se dá de acordo com os fatores apontados na Tabela 3.

Com base na metodologia empregada por Sampayo-Vargas et al. (2013) foram cri-

Figura 14: Exemplo de desafio requerindo múltiplas operações



Fonte: Imagem elaborada pelo autor

adas e testadas duas versões de um objeto de aprendizagem, abordando exatamente o mesmo conteúdo (MAINIERI et al., 2018). Uma destas possui um sistema de dificuldade adaptativa, que ajusta o desafio proposto de acordo com a performance medida do jogador, enquanto a outra emprega um modelo estático de dificuldade, no qual o desafio segue uma progressão pré-determinada e idêntica em todas as sessões. O desenvolvimento e teste deste primeiro protótipo está mais detalhadamente documentado na obra de Mainieri et al. (2018).

As duas versões do jogo utilizam a mesma lista de desafios. A diferença entre as versões se dá na forma como a lista é navegada.

Tabela 3: Complexidade ao longo da lista de desafios

Número do Desafio	Operações Disponíveis	Operações Necessárias	Números Adicionais
01-05	Adição	uma Adição	Nenhum
06-13	Adição	uma Adição	[1,2]
14-17	Subtração	uma Subtração	Nenhum
18-20	Subtração	uma Subtração	[1,2]
21-23	Adição, Subtração	uma Adição	Nenhum
24-25	Adição, Subtração	uma Adição	[1,3]
26-28	Adição, Subtração	uma Adição ou Subtração	[1,3]
29-36	Adição, Subtração	múltiplas Adiçôes e Subtraçôes	[1,3]
37-38	Adição, Subtração, Multiplicação	múltiplas Adiçôes, Subtraçôes e Multiplicaçôes	[1,3]

Fonte: Mainieri et al. (2018)

3.2.1 Versão Estática

Na versão estática do jogo, o primeiro desafio apresentado é o de número 1 na lista. Sempre que o jogador avançar, seja por êxito em atingir o Valor Alvo ou por usar o botão *Pular*, o próximo desafio apresentado é o que se encontra duas posições à frente na lista. Desta forma, a sequência de desafios encontrados em todas as sessões da versão estática é a mesma: 1,3,5,...,37. Esta escolha de "pular"metade dos desafios se deu por razões relacionadas a duração da interação com o objeto durante o processo de validação: caso não houvesse este salto e todos os desafios fossem apresentados em ordem, não seria possível aos participantes do processo de validação interagirem com desafios de todos os tipos durante o tempo disponível para a avaliação.

3.2.2 Versão Dinâmica

Na versão dinâmica, por outro lado, a lista é percorrida de acordo com o desempenho medido do jogador durante a sessão atual. A forma como o usuário lida com cada questão apresentada determina a progressão, ou regressão, na lista. Os possíveis eventos que influenciam a ordem com que se encontram os desafios são três:

- Sucesso sem uso do botão *Retroceder*: cada vez que o usuário obtém sucesso em um desafio sem fazer uso do botão *Retroceder*, uma variável chamada *sucessosEmOrdem*, invisível para o jogador, inicialmente com o valor 0, é incrementada em 1. O próximo desafio apresentado ao jogador é o que se encontra na posição da lista dada pela seguinte expressão:

$$proxPosicao = posicaoAtual + 1 + (sucessosEmOrdem)^{0,5}$$

- Sucesso com uso do botão *Retroceder*: ao fazer uso do botão *Retroceder* durante a resolução de um desafio, o valor da variável *sucessosEmOrdem* é alterado para 0. A próxima questão oferecida ao jogador se encontra na posição seguinte da lista à atual.

- Uso do botão *Pular*: ao fazer uso do botão *Pular* para avançar um desafio, o valor da variável *sucessosEmOrdem* é alterado para 0. A próxima questão oferecida ao jogador se encontra 4 posições atrás da atual na lista. Se está posição não existir (ou seja, o número da questão for menor que 1), o primeiro desafio da lista é apresentado.

O uso destas alternativas para a progressão na lista de desafios tem como objetivo fornecer uma experiência adequada e personalizada para cada jogador, em cada sessão do jogo. Usuários que resolvem os desafios sem uso das opções de *Retroceder* ou *Pular* são interpretados pelo sistema como tendo domínio dos conteúdos apresentados pelos desafios enfrentados. A fim de evitar o tédio da resolução de exercícios simples, o sistema de dificuldade então acelera a progressão do jogador pela lista de desafios, oferecendo questões mais complexas. No outro extremo do espectro, quando um jogador faz uso do botão *Pular*, é considerado que este desistiu de tentar resolver o desafio oferecido. A fim de evitar a frustração de uma sequência de desafios percebidos como impossíveis, o

sistema de dificuldade oferece questões mais simples, dando ao jogador a oportunidade de praticar antes de encarar novamente o desafio problemático. Estas práticas vão de encontro com a proposição de Csikszentmihalyi (1990) para a manutenção de um nível adequado de desafio e o favorecimento do engajamento dos usuários.

Por medir a competência do jogador e promover alterações na dificuldade ao longo de um único eixo, a técnica de dificuldade adaptativa empregada pode ser mais bem qualificada como linear, similar àquela proposta por Silva, Silva e Chaimowicz (2017).

Este método empregado não faz uso de qualquer algoritmo de Aprendizado de Máquina, limitando a implementação de dificuldade adaptativa neste protótipo a uma simples prova de conceito.

3.2.3 Validação do Primeiro Protótipo

O protótipo desenvolvido foi levado pelo autor a testes com alunos de graduação e professores do curso de Matemática da Universidade Presbiteriana Mackenzie com o propósito de medir a percepção dos educadores com relação a aplicabilidade do jogo como objeto de aprendizagem de aritmética, bem como sondar possíveis reações ao uso de dificuldade adaptativa no objeto. O protótipo foi disponibilizado para teste na plataforma iOS, em dispositivos iPhone 6S. A escolha destes se deu pela disponibilidade dos aparelhos e por apresentarem interface de interação por toque igual à que seria usada pelos alunos na sala de aula.

O teste de Mainieri et al. (2018) foi conduzido com base na metodologia proposta por Whitton (2007) para avaliar o uso de jogos na educação, adaptada para a aplicação com educadores. Um total de 10 alunos e professores interagiram com ambas as versões do objeto, tendo recebido explicação do objetivo deste objeto e da diferença de funcionamento entre versões. Resultados foram coletados através de um questionário de 16 itens empregando uma escala Likert de 5 pontos, na qual, para cada afirmação, o participante deveria responder o quão de acordo estava (Discordo Totalmente, Discordo, Não Concordo Nem Discordo, Concordo, Concordo Totalmente). Esta escala foi escolhida devido a seu uso em aplicações similares na literatura (GREENE; D'OLIVEIRA, 2005; WHITTON, 2009). A Tabela 4 contém as afirmações presentes no questionário aplicado. A versão A se refere à estática, enquanto versão B é a dinâmica.

Tabela 4: Questionário de teste

	Afirmação
1	O conteúdo abordado pelo jogo é adequado para a faixa etária a que se destina (7-8 anos).
2	A forma como o conteúdo é abordado no jogo condiz com a forma como é abordado em aula.
3	Os elementos visuais do jogo consistem em uma representação concreta adequada para os conceitos abstratos trabalhados.
4	O jogo incentiva mais que o aluno decore do que entenda a operações.
5	Os elementos visuais acrescentam confusão, dificultando o entendimento do conteúdo.
6	De maneira geral, o jogo promove a prática e o entendimento da aritmética básica.
7	A versão A do jogo propõe desafios similares aos apresentados em sala de aula.
8	Existem alunos para os quais a progressão de desafios da versão A seria muito extrema.
9	Existem alunos para os quais a progressão de desafios da versão A seria muito branda.
10	O fato de a sequência de desafios da versão A ser sempre a mesma reduz a aplicabilidade do jogo como objeto de ensino.
11	É possível notar a adaptabilidade da dificuldade da versão B em testes.
12	Desempenho reduzido na versão B resultou em desafios perceptivelmente mais simples.
13	Desempenho elevado na versão B resultou em desafios perceptivelmente mais complexos.
14	O aumento de dificuldade resultante do bom desempenho é excessivo; o jogo rapidamente se torna difícil demais.
15	A redução de dificuldade resultante do mal desempenho é excessiva; o jogo rapidamente se torna fácil demais.
16	A adaptabilidade da versão B torna o objeto mais adequado para uso em sala de aula do que a versão A.

Fonte: adaptado de Mainieri et al. (2018)

As afirmações 1 até 6 dizem respeito ao jogo no geral, sem fazer especificação a uma versão ou outra, pois abordam características comuns entre as duas. O objetivo com estas afirmações era medir a aplicabilidade educacional do jogo. As afirmações 7 até 10 tratam da versão A, estática, e buscam identificar as limitações levantadas pelas hipóteses sobre este tipo de objeto. As afirmações 11 até 16 abordam a versão B, dinâmica, e buscam sondar a percepção dos educadores quanto aos efeitos da dificuldade adaptativa no objeto de aprendizagem.

Para cada uma das afirmações foi estabelecida uma expectativa com relação as hipóteses do trabalho. Para estar plenamente alinhados com estas hipóteses, os resultados para as afirmações 1,2,3,6,7,8,9,10,11,12,13 e 16 seriam em maioria *Concordo* ou *Concordo Totalmente*, enquanto os resultados para as afirmações 4,5,14 e 15 seriam em maioria *Discordo* ou *Discordo Totalmente*. Os resultados obtidos estão relacionados na Tabela 5. A coluna *Adequação a Hipótese* indica a porcentagem das respontas que condizem com estas

expectativas descritas acima.

Tabela 5: Resultados do Questionário

Afirmação	Discordo Totalmente	Discordo	Nem Concordo Nem Discordo	Concordo	Concordo Totalmente	Adequação à Hipótese
1	0	0	1	3	6	90%
2	0	0	0	6	4	100%
3	0	0	0	2	8	100%
4	4	6	0	0	0	100%
5	5	5	0	0	0	100%
6	0	0	0	3	7	100%
7	0	0	2	3	5	70%
8	0	2	0	7	1	80%
9	0	0	2	6	2	80%
10	0	2	5	2	1	30%
11	0	0	0	4	6	100%
12	0	2	0	4	4	80%
13	0	1	0	5	4	90%
14	1	8	0	1	0	90%
15	0	7	3	0	0	70%
16	0	0	4	2	4	60%

Fonte: adaptado de Mainieri et al. (2018)

Realçados em azul na tabela estão os resultados mais frequentes para cada afirmação.

Embora os testes sofram de limitações como o número de participantes, o estado ainda inicial de desenvolvimento em que se encontra o protótipo e a implementação apenas de um modelo linear para a dificuldade adaptativa, os resultados preliminares obtidos apontam aprovação dos educadores quanto ao jogo desenvolvido até então. 100% dos participantes afirmaram Concordar ou Concordar Totalmente com as afirmações de que o jogo aborda de forma adequada os conteúdos estudados em sala de aula, os elementos visuais colaboram com o entendimento do conteúdo e a interação com o jogo estimula a prática das operações aritméticas abordadas.

Com relação à percepção dos participantes do emprego da dificuldade adaptativa, as respostas obtidas são insuficientes para corroborar ou refutar a hipótese de que o emprego desta técnica corresponde a um ganho no desempenho do jogo enquanto objeto de apren-

dizagem. Embora 60% dos participantes concordem em algum grau com a afirmação 16, que diz que a versão dinâmica é mais adequada para uso em sala de aula do que a versão estática, apenas 30% responderam à afirmação 10 de forma adequada à hipótese. Este resultado aponta para a possibilidade de objetos de ensino estáticos, sem qualquer capacidade de adaptação, possuírem empregabilidade no contexto da educação (MAINIERI et al., 2018).

Consideradas as limitações dos testes na avaliação dos resultados obtidos, pode-se derivar apenas confirmação de que o jogo em desenvolvimento é considerado pelos educadores participantes como adequado para emprego educacional.

3.3 Avaliação de Técnicas para Adaptabilidade

Os resultados da validação do protótipo inicial nortearam então a continuidade do desenvolvimento do projeto. Com a aprovação dos educadores e alunos do curso de Matemática do jogo enquanto objeto de aprendizagem, a próxima etapa do processo consistiu no estudo de diferentes abordagens para a implementação de dificuldade adaptativa mais sofisticada numa próxima iteração do jogo. Esta etapa foi desenvolvida ao longo de dois meses, e as avaliações foram realizadas pelo autor.

Independentemente da técnica escolhida, esta precisa ser capaz de realizar duas tarefas dentro do jogo para ser considerada efetiva:

- Identificar o desempenho do jogador com base na sua interação com o jogo
- Escolher a alteração a ser feita na dificuldade do jogo a fim de aproximar o desafio da habilidade do jogador

Existem duas maneiras em que se podem enquadrar estes requisitos numa estrutura de algoritmo de Aprendizado de Máquina. Cada ponto pode ser abordado separadamente, de tal forma que o sistema primeiro gera como resultado uma categorização do jogador de acordo com os dados de desempenho que receber de entrada e, então, utiliza esta categorização como entrada para uma nova inferência, buscando como saída a estratégia de alteração de dificuldade. Alternativamente, as duas etapas podem ser combinadas, de

tal forma que o sistema receba como entrada os dados de desempenho do jogador e associe estes diretamente à uma estratégia de alteração de dificuldade.

As seguintes seções apresentam a investigação realizada acerca de cada estratégia contemplada, e as conclusões resultantes sobre a aplicabilidade destas ao objetivo do projeto. Os algoritmos avaliados forma escolhidos de acordo com sua presença na literatura relativa ao desenvolvimento de objetos similares (ANDRADE et al., 2005; SREENIVAS, 2007; MISSURA; GÄRTNER, 2009) e por representarem uma variedade de técnicas empregadas no campo de Aprendizado de Máquina (RUSSELL; NORVIG, 2016).

3.3.1 Algoritmos de Aprendizado Supervisionado

Qualquer implementação em Aprendizado Supervisionado necessita, por definição, de um conjunto de dados de entrada para os quais já se conhecem as saídas desejadas. No caso do problema abordado, seria necessário então obter dados de interação dos alunos do público alvo com o jogo já associados às suas categorizações corretas de competência. Devido ao cronograma dos colégios contatados foi impossível realizar sessões com os alunos para a coleta destes dados antes do teste da aplicação, motivo este também que tornou inadequada a escolha de um algoritmo de Aprendizado Supervisionado para a versão final do jogo. Apesar disto, por motivos de completude da investigação, foi realizado um teste com dados gerados artificialmente para medir a aplicabilidade de técnicas de Aprendizado Supervisionado em sistemas de dificuldade adaptativa, particularmente no que diz respeito à tarefa de categorizar o jogador de acordo com suas interações.

Nesta prova de conceito, foi criado um conjunto de dados artificial que representa o desempenho de 100 alunos numa hipotética atividade que consiste em 12 desafios, três para cada uma das quatro operações aritméticas. O desempenho dos alunos é medido como uma nota numérica entre 0 e 10 para cada uma das atividades. Cada um dos alunos simulados pertence a uma das 9 categorias descritas na Tabela 6.

Os dados deste conjunto foram gerados de maneira pseudo-aleatória, buscando uma distribuição de competências dos alunos simulados que cobrisse todo o espaço de possibilidades: desde alunos que não detém qualquer conhecimento até alunos que dominam todas as categorias. Este conjunto foi gerado com o objetivo de testar o funcionamento dos algoritmos avaliados no vácuo, não servindo então para prever seu funcionamento

Tabela 6: Categorias de Desempenho

Classe	Descrição
1	Nenhum domínio
2	Domínio apenas sobre adição
3	Domínio sobre adição e subtração
4	Domínio sobre adição e multiplicação
5	Domínio sobre adição, subtração e multiplicação
6	Domínio sobre adição, subtração, multiplicação e divisão
7	Domínio sobre subtração e multiplicação
8	Domínio sobre subtração e divisão
9	Domínio sobre multiplicação e divisão

Fonte: Elaborado pelo autor

para qualquer grupo específico de alunos reais.

Uma rede MLP de duas camadas escondidas de 12 neurônios cada foi construída na linguagem Python com uso da biblioteca Scikit-learn (PEDREGOSA et al., 2011) e treinada ao longo de 700 iterações com este conjunto de dados, processo este que em um MacBook Pro 2015 com processador de 2,7 GHz e 8 GB de memória RAM durou em média dois segundos. Separando 20% dos elementos do conjunto de treinamento para teste, a rede obteve precisão média de 99% sobre estes dados. Esta precisão foi calculada por meio da expressão:

$$\frac{inferenciasCorretas}{totalDeInferencias}$$

sendo uma inferência considerada correta quando uma questão é adequadamente categorizada de acordo com o valor esperado no conjunto de testes.

Uma vez treinada, a rede era capaz de receber novas entradas de dados no formato de 12 valores de desempenho para os desafios das operações aritméticas, e associar esta nova entrada a uma das categorias. Cada inferência demandou cerca de 0,1 segundos. Visto que foram empregados dados artificiais para o treinamento e também para o teste, é impossível fazer qualquer afirmação acerca da acurácia do algoritmo em casos reais. O que se pode

afirmar sobre a utilidade do algoritmo é a capacidade de categorizar adequadamente dados novos que cujas características estão representadas nos elementos do conjunto de treinamento e de fornecer resultados informativos nos demais elementos: caso um valor de entrada demonstre características similares às daqueles de duas categorias diferentes, por exemplo, o sistema é capaz de fornecer como saída uma porcentagem de adequação à cada categoria.

Este teste, naturalmente básico e limitado em aplicabilidade devido ao uso de um conjunto de dados artificial, não justifica o uso de um algoritmo do tipo MLP, ou de Aprendizado Supervisionado em geral, para a aplicação desenvolvida neste trabalho, visto que sua utilidade depende da disponibilidade de dados reais para treinamento.

3.3.2 Algoritmos de Aprendizado Não Supervisionado

O uso de um algoritmo de Aprendizado Não Supervisionado é sugerido na literatura por Stevens et al. (1999). Nesta abordagem um SOM, ou Mapa Auto Organizável (KOHONEN, 1990), é alimentado com os dados de desempenho dos alunos e, então, organiza estes de acordo com sua similaridade. Em contraste com a abordagem Supervisionada em que um MLP, por exemplo, classifica cada aluno em uma das categorias pré determinadas, o SOM agrupa os alunos em um número de grupos determinado pelo próprio algoritmo de acordo com a similaridade de seus valores de desempenho em cada uma das dimensões medidas.

O mesmo conjunto de dados gerados artificialmente para a avaliação do MLP foi empregado para o teste do SOM, removidos os valores de saída esperados. Novamente, esta simulação se faz necessária devido à dificuldade de obter dados para treinamento em um sessão com os alunos antes do teste da aplicação.

A rede foi construída e treinada utilizando a linguagem Python e a biblioteca NumPy (OLIPHANT, 2006), processo este que levou em média 4 minutos e 42 segundos em um MacBook Pro 2015 com processador de 2,7 GHz e 8 GB de memória RAM. Para este conjunto de dados, a rede encontrou 6 *clusters*, em contraste às 9 categorias definidas na construção do conjunto de dados que podem ser vistas na Tabela 6.

O teste realizado aponta para a capacidade de um SOM realizar agrupamentos de

dados diferentes dos previstos durante a elaboração do problema. Esta capacidade pode ser útil em situações onde não se conhece a quantidade de categorias e as características de cada uma. Caso a rede seja treinada de maneira *off-line*, isto é, antes de ser utilizada na aplicação, o tempo de treinamento não inviabiliza seu emprego em um sistema de dificuldade adaptativa. Para a aplicação atual, porém, a dificuldade de coletar dados reais para o treinamento da rede torna esta alternativa inadequada.

Foi testado também um algoritmo de Lloyd (LLOYD, 1982), ou *k-means*, dentro do paradigma Não Supervisionado. Novamente empregando o mesmo conjunto de dados, foi definido como k o valor 9, igual ao número de classes contempladas no teste do algoritmo de Aprendizado Supervisionado.

O algoritmo foi implementado em Python através da biblioteca Scikit-learn (PEDREGOSA et al., 2011) com k definido como 9, número de categorias definidas na construção do conjunto de dados. A categorização dos 100 exemplares do conjunto de dados gerado demorou em média 1.1 segundos em um MacBook Pro 2015 com processador de 2,7 GHz e 8 GB de memória RAM, e obteve-se 85% de precisão, calculada como a taxa de questões agrupadas na categoria correta dentre o total de questões agrupadas realizadas.

O algoritmo de Lloyd demonstrou, neste teste limitado com base em um conjunto de dados gerados artificialmente, precisão inferior ao MLP na tarefa de categorizar usuários de acordo com medidas de seu desempenho.

3.3.3 Algoritmos de Aprendizado Por Reforço

Vistas as limitações na aplicabilidade tanto de algoritmos de Aprendizado Supervisionado quanto de Aprendizado Não Supervisionado, principalmente devido à dificuldade de se coletar dados de treinamento a partir de interações dos alunos com o jogo, técnicas de Aprendizado Por Reforço surgem como uma alternativa viável para implementação de dificuldade adaptativa em um objeto de aprendizagem, conclusão a que chega também Sreenivas (2007).

Uma maneira com a qual algoritmos de Aprendizado Por Reforço evitam a limitação encontrada pelas demais alternativas é a possibilidade de realizar seu treinamento *on-line*, isto é, durante a própria execução do sistema na sua situação de uso real, neste caso, na

interação dos alunos. Um algoritmo deste tipo poderia, então, aprender durante o teste final da aplicação, sem necessitar treinamento prévio. Hester, Quinlan e Sonte (2011) descrevem uma arquitetura de Aprendizado por Reforço deste tipo em sua obra, citando como requerimento para esta solução a capacidade de aprender em poucas iterações ao mesmo tempo em que está sendo utilizada.

O algoritmo escolhido para avaliação é uma variação do problema do "bandido de k braços", ou *k-armed bandit* (SUTTON; BARTO, 1998), nomeado em analogia à uma hipotética máquina de caça niqueis com k alavancas. Este problema consiste em um agente que pode escolher uma dentre k opções de ação, cada uma com uma probabilidade diferente e imutável de ser a escolha correta. Cada vez que o agente escolhe a opção correta, este recebe um sinal de recompensa positivo, nos demais casos recebe um sinal de recompensa negativo. Conforme o agente explora estas opções ao longo de múltiplas iterações, este tende a adequar sua política de tomada de decisões para favorecer a opção com maior probabilidade de resultar em recompensa positiva. O algoritmo para avaliação foi desenvolvido em linguagem C#, e teve como base a implementação de Juliani (2017). Diferentemente dos demais algoritmos testados, o objetivo desta alternativa não é categorizar o usuário de acordo com sua competência, e sim estimar, a partir dos dados coletados da interação com este, o desafio adequado a ser apresentado.

O experimento realizado com algoritmo de Aprendizado Por Reforço consistia em um sistema que podia escolher, como sua ação, uma dentre cinco opções de tipo de desafio matemático e recebia, como recompensa, o resultado da interação do usuário com o desafio: acerto ou erro. Assim como na implementação de Juliani (2017), este sistema aprendia de forma *on-line*, durante a própria interação com o usuário. Foi definido para o sistema que, caso o resultado de um desafio seja uma certo, este recebe recompensa 1 e, case seja erro, uma recompensa -1: quanto maior a recompensa associada a um tipo de desafio, maior a competência do jogador para a resolução destes. A função de valor estimada do sistema foi construída de tal forma que considerasse como mais adequada a ação com valor de recompensa mais próxima de 0, ou seja, para a qual não se tenha medido no jogador nem domínio (valores de recompensa acima de 0) nem carência (valores de recompensa inferiores a 0).

Apesar da avaliação do sucesso deste sistema se limitar a observações heurísticas

(HASTIE; TIBSHIRANI; FRIEDMAN, 2009), este mostrou capacidades úteis para a aplicação desenvolvida neste trabalho. O sistema baseado em Aprendizado Por Reforço não realiza a tarefa de categorizar o usuário explicitamente, e foca em mapear os dados coletados da interação diretamente para decisões de escolha de desafio. Em simulações onde o usuário era instruído a acertar todos os desafios de uma dada categoria, o sistema decidia por não apresentar novos desafios desta categoria após poucas iterações: para uma taxa de aprendizado definida em 0.2, três acertos consecutivos em uma dada categoria eram suficientes para que não fossem apresentados novos desafios desta. De maneira similar, três erros consecutivos tinham o mesmo efeito, ao reduzir o valor de recompensa associado a esta ação pelo algoritmo. Alterações na taxa de aprendizado permitiam regular a quantidade de acertos ou erros necessários para que uma categoria fosse considerada inadequada em níveis de desafio. Em princípio, este resultado é útil no que diz respeito à obtenção de um nível adequado de dificuldade, rapidamente excluindo a possibilidade de serem apresentados desafios excessivamente fáceis ou difíceis para o usuário, promovendo o desejado engajamento descrito por Csikszentmihalyi (1990).

Este teste, apesar de simples, demonstrou a capacidade de treinamento *on-line* deste tipo de algoritmo, e a possibilidade de empregar estes no processo de escolha de desafios propostos no jogo. Uma implementação baseada neste princípios surge, então, como alternativa viável para a dificuldade adaptativa na aplicação em desenvolvimento.

3.4 Desenvolvimento do Jogo para Teste

Tendo como recursos os resultados da validação do primeiro protótipo e as avaliações de diferentes alternativas para implementação da dificuldade adaptativa, a última etapa do processo de desenvolvimento foi a criação da iteração final do jogo que seria levada a testes com alunos. Esta nova versão foi construída a partir do protótipo anterior, mantendo a mesma estrutura básica de mecânicas de jogo.

A nova versão foi desenvolvida pelo autor ao longo de cinco semanas durante os meses de Setembro e Outubro de 2018, também empregando a linguagem *C#* na plataforma Unity. Neste momento do desenvolvimento já se sabia que durante o teste os alunos teriam acesso à *iPads*, então a nova iteração do jogo foi construída para as especificações de tela e tecnologias desta plataforma.

Além de mudanças em seu funcionamento e no modelo de dificuldade adaptativa, esta iteração do jogo recebeu melhorias em sua apresentação e interface, visto que este seria levado a testes diretamente com os alunos. Enquanto o protótipo era puramente funcional e continha apenas a tela do jogo, esta nova iteração apresenta um menu principal que pode ser visto na Figura 15.

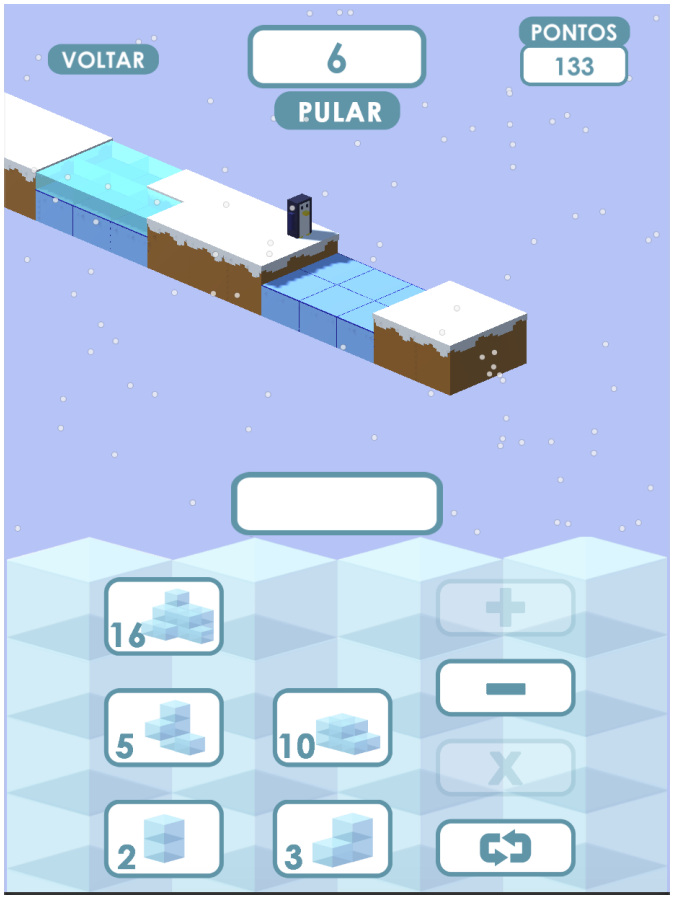
Figura 15: Interface do menu principal do jogo desenvolvido

Fonte: Imagem elaborada pelo autor

Sendo a primeira tela com que o usuário tem contato, este menu faz o papel de introduzir o jogador à fantasia do jogo, ilustrando tanto o personagem principal quanto a temática glacial, fazendo também referência à matemática. O botão Iniciar leva o jogador à tela do jogo, enquanto o botão Opinar leva a uma interface de coleta de *feedback*.

Na Figura 16, da tela de jogo, é possível ver algumas das mudanças desta iteração em comparação com o protótipo. No que diz respeito à ilustração da fantasia do jogo, as árvores anteriormente presentes no cenário do jogo foram removidas a fim de tornar mais

Figura 16: Tela do jogo da iteração final



Fonte: Imagem elaborada pelo autor

fiel a representação do ambiente antártico em que este se passa.

Não mais sendo a única tela da aplicação, a interface da tela de jogo agora contém um botão de Voltar no canto superior esquerdo para permitir a navegação até o menu principal. Oposto a este encontra-se um campo para registro da pontuação do jogador. Esta característica nova do jogo foi introduzida a partir de recomendações dos educadores após o processo de validação do primeiro protótipo e encontra seu embasamento teórico na obra de Malone e Lepper (1987), dentro do que categorizam como motivação interpessoal de reconhecimento. Os autores propõem que a possibilidade de ter sua competência e sucesso exposto aos indivíduos ao seu redor é um fator de motivação social que pode ser abordado pelos jogos por meio de valores de pontuação. A fim de evitar a emergência de competição negativa, onde o indivíduo que tem menor pontuação é desmotivado de continuar participando, os autores recomendam que esta métrica não seja utilizada como

fator principal de desempenho do jogo, ou tida como foco da atividade.

Outra diferença visível na Figura 16 é a presença de números de dois algarismos na interface de calculadora. A limitação à números de um único algarismo no protótipo restringia a variedade possível de expressões a serem construídas no jogo, o que motivou o alargamento dos botões e a criação de botões contendo valores até 20. Este novo valor máximo foi escolhido com base em limitações de visualização na tela do jogo: em expressões que contém números maiores, os blocos de água e gelo no cenário do jogo são reduzidos a fim de possibilitar a visualização do cenário inteiro na tela. Para operadores de valor superior a 20, a redução necessária para que todos os blocos coubessem em tela resultaria em blocos de difícil visualização e contagem.

Além das mudanças visíveis na interface do jogo e em sua navegação, a nova iteração recebeu também alterações nos desafios propostos. Com base nas respostas obtidas dos educadores durante o processo de validação e nas definições de habilidades da Base Nacional Comum Curricular (BRASIL, 2018), foram definidas 7 categorias de desafios. A Tabela 7 identifica estas 7 categorias, descrevendo a natureza dos desafios de cada uma bem como o número de questões existentes no jogo para cada categoria. Esta iteração do jogo apresenta um total de 80 questões (APÊNDICE B), mais do que o dobro das 38 presentes no protótipo.

Dentro de cada categoria foram definidas questões com níveis de complexidade diferentes, com base nos mesmos fatores usados para determinar o nível dos desafios do protótipo anterior: questões que apresentam números adicionais são mais complexas do que aquelas que não apresentam, questões com operandos de dois algarismos são mais complexas que aquelas com apenas operandos de um algarismo, questões que requerem múltiplas operações encadeadas são mais complexas do que aquelas que requerem apenas uma operação. Um exemplo de duas questões na mesma categoria mas de níveis distintos de complexidade pode ser visto na Figura 17, em que (a) não contém números adicionais, enquanto (b) contém.

Assim como no protótipo, a iteração final do jogo consiste em duas versões idênticas exceto pela maneira como são definidos os desafios oferecidos ao jogador: uma versão, chamada estática, apresenta os desafios em uma ordem pré determinada e imutável, enquanto outra versão, chamada dinâmica, emprega um algoritmo de dificuldade adaptativa

Tabela 7: Categorias de Desafios da iteração final do jogo

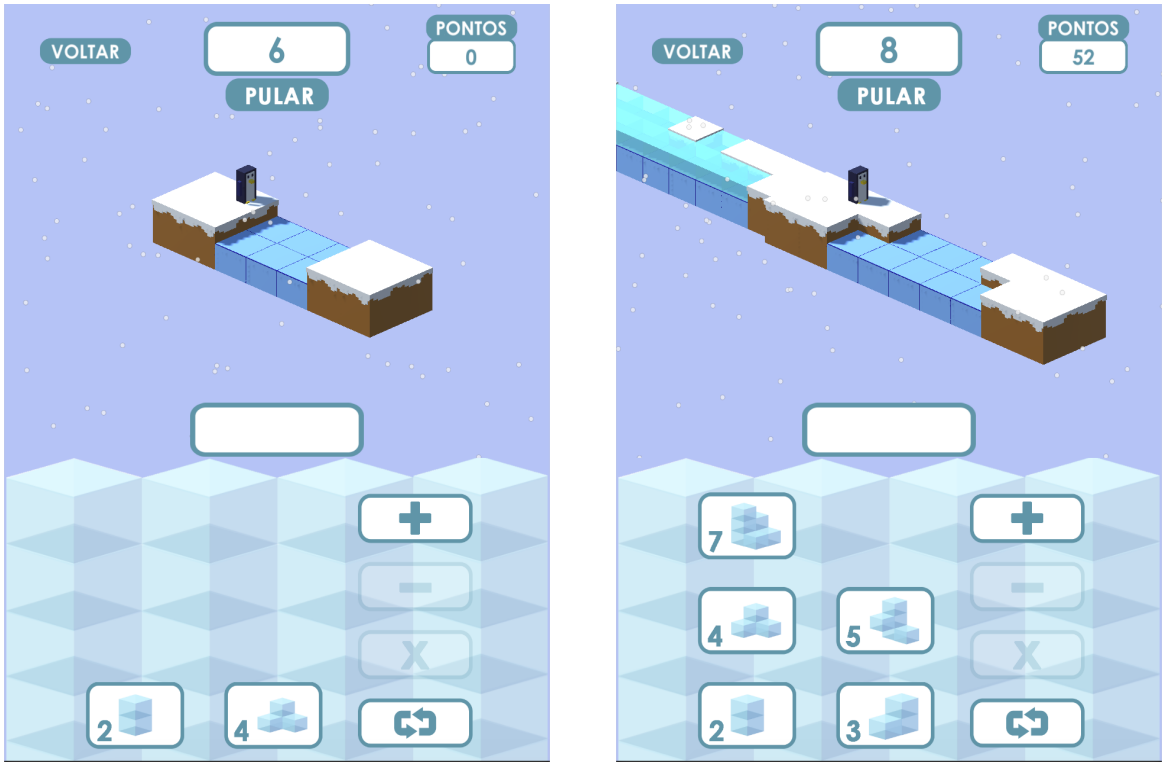
# Questões	Classe	Descrição
15	Adição Simples	Desafios contendo apenas adições com dois operandos, podendo conter números adicionais
12	Subtração Simples	Desafios contendo apenas subtrações com dois operandos, podendo conter números adicionais
13	Adição composta	Desafios contendo adições com mais de dois operandos, podendo conter números adicionais
11	Subtração composta	Desafios contendo subtrações com mais de dois operandos, podendo conter números adicionais
9	Adição ou subtração	Desafios que requerem uso ou de adição ou subtração, mas não ambos, podendo requerer múltiplas operações e conter números adicionas
12	Adição e subtração	Desafios que requerem uso tanto de adição quanto subtração, podendo requerer mais de duas operações e conter números adicionais
9	Multiplicação	Desafios contendo apenas multiplicações com dois operandos, podendo conter números adicionais

Fonte: Elaborado pelo autor

para a definição dos desafios apresentados.

A versão estática apresenta uma sequência contendo 53 dos 80 desafios presentes no jogo, incluindo desafios de todas as categorias descritas na Tabela 7. Os desafios são ordenados por categoria, na mesma ordem descrita na Tabela 7 e, dentro de cada categoria, em ordem crescente de complexidade, utilizando os mesmos fatores utilizados para definir a ordem dos desafios no primeiro protótipo.

Na versão dinâmica, em contraste, não existe uma ordem pré definida de desafios; ao invés disto, cada questão é escolhida por um sistema de dificuldade adaptativa.



(a) Desafio de adição que não contém números adicionais

(b) Desafio de adição que contém números adicionais

Figura 17: Comparação de complexidade entre dois desafios de adição

3.4.1 Implementação de Dificuldade Adaptativa

Com base na avaliação de diferentes alternativas para a implementação de Dificuldade Adaptativa, foi escolhida uma solução baseada em Aprendizado Por Reforço com treinamento *on-line*, visto que não seria possível coletar dados de interação com os estudantes antes do teste para realizar o treinamento de modelos de Aprendizado Supervisionado ou Não Supervisionado.

A implementação escolhida (APÊNDICE C) foi baseada no problema do bandido de k braços, consistindo em um sistema de Aprendizado Por Reforço em que existe um único estado. O objetivo do sistema é escolher, com base no histórico de interações do jogador durante a sessão do jogo, qual desafio apresentar a cada momento. Neste contexto, o agente é um componente do jogo, o ambiente é o próprio jogo, as ações possíveis são as diferentes possibilidades de escolha para o próximo desafio e a recompensa vem na forma do resultado da interação do usuário com a questão escolhida.

O algoritmo detém uma estimativa de valor de competência para cada categoria, que representa a habilidade medida do jogador para resolver questões da categoria. Estes valores podem variar de 0, significando que o usuário não detém qualquer habilidade para resolver questões desta categoria, até 1, representado domínio completo das questões desta categoria. Os valores para cada categoria são inicializados com base na expectativa do nível de conhecimento médio de um aluno do terceiro ano do ensino fundamental, definida a partir dos dados da Base Nacional Comum Curricular com o auxílio de educadores da matemática. Os valores iniciais definidos podem ser vistos na Tabela 8.

Tabela 8: Valores iniciais para estimativa de competência de cada categoria

Categoria	Valor inicial
Adição Simples	0,70
Subtração Simples	0,60
Adição composta	0,40
Subtração composta	0,30
Adição ou subtração	0,20
Adição e subtração	0,10
Multiplicação	0,10

Fonte: Elaborado pelo autor

Estes valores são atualizados sempre que o algoritmo recebe uma recompensa, e utilizados para selecionar uma nova questão quando necessário. Estes valores não são, porém, uma expectativa de recompensa a ser maximizada, como seria no caso do problema do bandido de k braços. Como estes medem a habilidade do jogador, escolher sempre a opção com maior valor de estimativa de competência resultaria num jogo fácil: seriam escolhidas as categoria nas quais o jogador demonstra maestria.

A principal diferença desta implementação com relação ao problema original da literatura se dá na forma como é feita a escolha da próxima ação. A fim de evitar a escolha de categorias que ofereçam desafio desproporcional à habilidade do jogador, o algoritmo descarta as opções de categoria cujo valor de competência seja superior a 0,85 , limite definido via testes com educadores e voluntários. Caso restem ainda quatro ou mais categorias válidas, são descartadas também as duas categorias com menor valor de com-

petência. Esta primeira etapa elimina os extremos da seleção de ação, escolha necessária devido à natureza do sistema implementado: sendo um sistema com treinamento *on-line* ao longo de poucas interações, a exploração das opções extremas tem maior probabilidade de causar frustração no usuário.

Para a escolha da categoria dentre as alternativas restantes, faz-se uso de uma função *softmax* para determinar a probabilidade P_k de escolha de cada categoria em função de seu valor esperado:

$$P_k = \frac{exp(V_k/\tau)}{\sum_{i_1}^{n} exp(V_i/\tau)}$$

para todas as n categorias válidas para escolha e com τ como o valor de temperatura. Este valor determina o peso do valor esperado V_k de cada categoria no cálculo da sua probabilidade de ser escolhido. Quanto maior a temperatura, menor influência o valor esperado de cada categoria tem na probabilidade. Para $\tau \longrightarrow 0$, a probabilidade de escolher a opção com maior valor esperado tende a 1. O valor de τ escolhido para a aplicação foi 0,08 , valor também definido empiricamente através de testes com educadores e voluntários, com o objetivo de minimizar casos de categorias com baixo valor estimado sendo escolhidas repetidamente.

Uma vez definida a probabilidade de escolha de cada categoria, o sistema sorteia a categoria escolhida a partir destes valores de probabilidade. Esta técnica é usada para garantir que a categoria escolhida não seja sempre a com maior valor esperado de competência, permitindo ao sistema avaliar o aluno também nas outras categorias, mas ainda favorecendo a escolha de categorias com alto valor estimado, a fim de evitar frustração do jogador.

É importante ressaltar que este sistema, ao não possuir qualquer treinamento *off-line* anterior à interação com os alunos, precisa aprender sobre as competências dos alunos durante o próprio teste. Desta forma, é razoável supor que as primeiras iterações de escolha de desafio, baseadas inteiramente nos valores iniciais de estimativas gerais de conhecimento dos alunos, estarão menos adequadas à competência particular de cada aluno do que as iterações posteriores, baseadas nas observações das interações do aluno.

Em termos de *exploration* e *exploitation* (SUTTON; BARTO, 1998), o algoritmo de

escolha de categorias implementado tem foco em *exploitation* devido ao curto tempo disponível para interação dos estudantes com o jogo, mas apresenta também *exploration* dentro das alternativas que não foram cortadas inicialmente.

Uma vez escolhida a categoria de questões a apresentar ao jogador, são propostos três desafios desta categoria ao jogador, um após o outro, antes de ser escolhida a próxima categoria de questões. Estes três desafios são escolhidos dentre todos os da categoria como aqueles cujo nível de complexidade mais se aproxime do valor estimado de competência do jogador para a categoria. Com este agrupamento, busca-se evitar a aparência de aleatoriedade na escolha de questões que poderia ser passada caso fosse escolhida uma categoria diferente após cada desafio. Outro motivo para esse agrupamento de questões diz respeito ao processo de recompensa do algoritmo: o resultado da interação do jogador com as três questões é combinado em um único valor de recompensa. Desta forma, um jogador que detém conhecimento em uma categoria mas comete um engano na primeira questão deste tipo pode ainda mostrar ao sistema sua competência ao acertar as duas próximas questões. Caso não houvesse este agrupamento, ao errar a primeira questão o jogador forneceria ao sistema uma recompensa que indica que este não possui competência para esta categoria.

Tendo como ideal uma situação onde o desafio escolhido pelo sistema condiz com o nível de competência do jogador (CSIKSZENTMIHALYI, 1975), foi definida a seguinte função de recompensa, resultante da interação do jogador com cada desafio escolhido:

$$R_k = \begin{cases} 1 + (c - V_k), & \text{caso Sucesso sem uso de Retroceder.} \\ min(V_k, V_k + (V_k - c)), & \text{caso Sucesso após Retroceder.} \\ max(0, min(V_k - 0.1, -V_k + c)), & \text{caso uso de Pular} \end{cases} \quad (1)$$

em que k é a categoria do desafio escolhido, V_k é o valor de recompensa estimado atual para questões da categoria escolhida e c é o valor de complexidade do desafio escolhido, cujo valor varia entre 0,3 e 0,9. Este valor é calculado para cada uma das três questões para uma dada iteração de escolha de categoria, e a média destes é a recompensa dada ao algoritmo de aprendizado por reforço.

Em casos de sucesso sem uso da opção *Retroceder*, o valor de recompensa aproxima-se

de, ou excede, 1. Considerada no modelo como a demonstração de maior competência possível, este resultado provoca também o maior aumento de valor estimado, a fim de evitar que o aluno continue recebendo desafios desta categoria.

Em casos de sucesso após uso da opção *Retroceder*, independentemente da quantidade de usos desta opção, o valor de recompensa tende a se aproximar do valor de competência atual do aluno para a categoria, sem nunca excedê-lo. Considerada no modelo como demonstração de um nível de competência similar ao nível do desafio proposto, este resultado gera a menor variação do valor estimado, a fim de promover a escolha de mais desafios desta categoria.

Caso o aluno opte pelo uso do botão *Pular*, o valor de recompensa será sempre menor que a sua competência atual na categoria. Considerada no modelo como situação em que o nível de desafio é superior à competência do aluno, este resultado provoca a maior redução de valor estimado, reduzindo a probabilidade de subsequente escolha desta categoria.

O valor de recompensa R_k definido é então utilizado para atualizar o valor de competência estimado para a categoria da questão V_k da seguinte maneira:

$$V_k(t+1) = V_k(t) + \alpha(R_k - V_k(t))$$

com o valor da alteração realizada dependendo da taxa de aprendizado α, definida na aplicação como 0.35.

Outra diferença da implementação realizada com relação ao problema básico do bandido de k braços é o uso de um modelo que correlaciona diferentes opções de ações de acordo com sua similaridade semântica. Trabalhando com uma abordagem simplifica proposta de Hester, Quinlan e Sonte (2011) de usar um modelo para agilizar o aprendizado de um sistema com poucas iterações de seleção de ação, o sistema desenvolvido realiza alterações nos valores de competência estimados do jogador não só para a categoria específica da questão que acabou de resolver, mas também de categorias que estão semanticamente relacionadas. Realizar corretamente uma questão de Adição Composta, por exemplo, permite ao sistema inferir que o usuário também seria capaz de realizar corretamente uma questão de Adição simples. Da mesma forma, sucesso em uma atividade de Adição Composta indica uma maior probabilidade de sucesso em uma atividade de

Multiplicação, dado que o princípio matemático aplicado em uma é similar ao da outra.

A implementação desta capacidade se dá por meio de uma tabela de correlação, que indica qual fator da alteração do valor estimado para a categoria escolhida deve ser transferida também para as demais categorias. A Tabela 9 indica os fatores escolhidos na aplicação.

Tabela 9: Tabela de valores de correlação de categorias

Categoria Escolhida	Categorias Influenciadas						
	A	S	Ac	Sc	A ou S	A e S	M
Adição	1	0	0.4	0	0.3	0.3	0.2
Subtração	0	1	0	0.4	0.3	0.3	0
Adição Composta	1	0	1	0	0.4	0.4	0.35
Subtração Composta	0	1	0	1	0.4	0.4	0
Adição ou Subtração	0.6	0.6	0.3	0.3	1	0.45	0.15
Adição e Subtração	1	1	0.6	0.6	0.8	1	0.3
Multiplicação	0.8	0	0.6	0	0	0	1

Fonte: Elaborado pelo autor.

A alteração do valor de competência estimado para cada categoria j após um desafio da categoria k se dá por meio de:

$$V_j(t+1) = V_j(t) + c_{kj} m \alpha (R_k - V_k(t))$$

com base no fator de correlação c_{kj} da categoria escolhida do desafio com a categoria afetada e taxa de correlação m, que assume o valor de 1 caso a mudança de valor estimado seja positiva ou zero e 0.65 caso a mudança seja negativa. O propósito desta taxa é reduzir o impacto que um erro em uma questão tem sobre a medida de competência em outras categorias.

Este modelo de alteração de valor estimado baseado na tabela de correlação tem como objetivo acelerar o processo de aprendizado do algoritmo, fazendo uso do relacionamento entre as competências necessárias para cada categoria de questões para atualizar múltiplos

valores simultaneamente.

Ao implementar este modelo, o algoritmo adquire ainda a capacidade de voltar a apresentar questões de categorias que foram, em um momento anterior, consideradas inválidas para escolha, graças ao alto nível de competência demonstrado pelo jogador. Caso o jogador tenha sua competência em questões de Adição Simples medida como alta o suficiente para que o algoritmo considere esta categoria inválida para escolha, mas demonstre dificuldade na resolução de desafios de Adição Composta, por exemplo, o algoritmo é capaz de reduzir o valor de competência para Adição Simples e voltar a apresentar questões desta categoria, permitindo ao aluno praticar a adição em questões mais simples.

3.5 Teste com Alunos

Esta iteração final do jogo em suas duas versões, estática e dinâmica, foi levada a testes com alunos do terceiro ano do ensino fundamental do Colégio Presbiteriano Mackenzie. Os testes ocorreram em seis sessões, cada uma com turmas diferentes de alunos, durante a semana do dia 5 de Novembro de 2018. Todas as atividades foram realizadas em sala de aula dentro do horário regular de aula, guiadas pelo autor com o acompanhamento da professora responsável pela turma. Estes alunos, durante as suas aulas regulares, já interagiam com objetos interativos digitais na plataforma *iPad*, o que facilitou a aplicação deste teste ao remover a possibilidade dos estudantes não estarem familiarizados com o uso da tecnologia. A participação nos testes era opcional e dependia da prévia autorização dos responsáveis pelos alunos. No total, 150 alunos participaram das atividades em alguma capacidade, mas 8 destes não foram contabilizados nos resultados pois não completaram a interação com o objeto - 1 destes não chegou a realizar qualquer desafio no jogo e os outros 7 não responderam ao questionário sobre a experiência.

Cada sessão de testes era iniciada com a apresentação do objeto aos alunos na sala de aula. A existência das duas versões distintas não era informada aos alunos, e não era passada qualquer informação sobre a forma como ocorria a seleção de questões. Cada aluno então recebia um *iPad* com o jogo instalado. Metade dos participantes recebia cada versão do jogo, sendo a distribuição destes dois grupos pela sala de aula aleatória; não havia correlação entre a proximidade dos alunos e a chance de interagirem com a mesma

versão.

Ao curso de um minuto eram então passadas instruções para o jogo e, a partir deste momento, os alunos estavam livres para selecionar a opção Iniciar no menu principal e interagir com o objeto. Foi explicado que estes podiam encerrar a atividade a qualquer momento, sem precisar continuar interagindo até o término do horário da aula, caso desejassem. Os alunos também foram encorajados a expressar suas opiniões sobre o jogo e eram convidados a fazer perguntas e sugestões a qualquer momento da atividade.

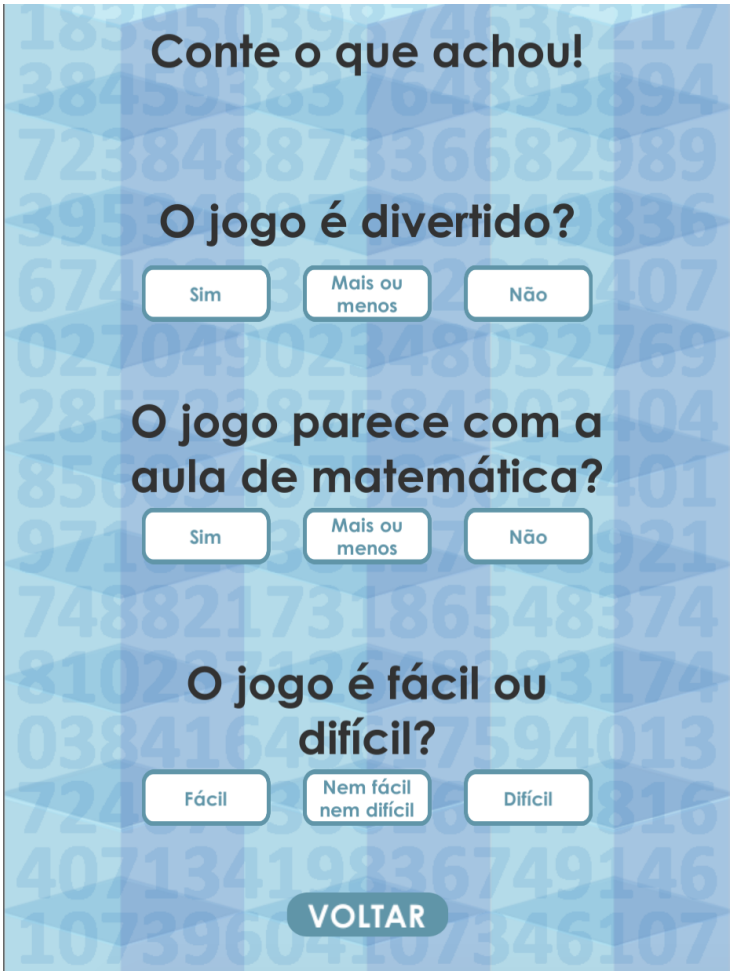
Durante o teste, o jogo coletava dados anônimos da interação de cada jogador. Estes dados eram o tempo total de interação, o tempo em cada questão, o resultado de cada questão (acerto, acerto após n tentativas, pulo), a categoria e complexidade de cada questão e a evolução dos valores estimados de competência, no caso da versão dinâmica.

Ao final da interação dos alunos com o objeto, estes eram orientados a retornar ao menu principal do jogo e selecionar a opção Opinar. Esta opção leva à interface de *feedback* de opinião, que pode ser vista na Figura 18.

Nesta tela são apresentadas três perguntas ao jogador, cada uma com três opções de respostas. Este questionário foi elaborado com base na obra de Whitton (2007) com o objetivo de coletar informações a respeito da experiência dos jogadores, bem como opiniões que auxiliassem na medição de sua motivação. As respostas de cada pergunta se assemelham a um modelo da escala Likert com apenas três opções, equivalentes às opções Concordo, Não Concordo Nem Discordo e Discordo. A escolha de uso de apenas três opções, em contraste às cinco vistas na literatura (GREENE; D'OLIVEIRA, 2005; WHITTON, 2009), foi com base em recomendações dos educadores que trabalham com os alunos do grupo de teste, com o objetivo de simplificar a escolha para os alunos.

A primeira pergunta diz respeito à opinião dos alunos com relação à diversão promovida pelo jogo. Naturalmente subjetiva, o objetivo existente na inclusão desta pergunta era de comparar as respostas de alunos que interagiram com a versão estática e dinâmica, a fim de verificar se houve diferença na percepção de diversão dos alunos. A segunda pergunta busca medir a familiaridade dos conteúdos e atividades propostas pelo jogo, vista a necessidade de contextualização do jogo enquanto objeto de aprendizagem no ambiente da sala de aula, exposta por Cardoso, Ghelli e Oliveira (2017). A terceira pergunta estava diretamente relacionada ao objetivo da pesquisa de promover um nível adequado de

Figura 18: Interface de *feedback*



Fonte: Imagem elaborada pelo autor

dificuldade no jogo. As respostas a esta pergunta seriam comparadas aos dados de desempenho dos alunos a fim de verificar se a percepção de nível de dificuldade dos alunos condiz com o seu desempenho na atividade, além de identificar diferenças entre respostas de alunos que interagiram com as diferentes versões do jogo.

4 Resultados

Foram coletados resultados válidos da interação de 142 alunos com o objeto, tendo 70 destes interagido com a versão estática e 72 com a versão dinâmica do jogo.

A primeira informação comparada foi a de tempo de interação dos alunos com o objeto; foi explicado aos participantes que estes poderiam encerrar a atividade a qualquer

momento caso desejassem. A Tabela 10 contém os dados de tempo médio de interação de cada aluno, divididos pela versão com a qual interagiram. Alunos que interagiram com versão dinâmica o fizeram por, em média, 24 minutos e 18 segundos, enquanto aqueles que interagiram com a versão estática o fizeram por uma média de 20 minutos e 36 segundos. A versão dinâmica obteve então uma média de 3 minutos e 42 segundos de interação a mais do que a estática, ou 17,9% a mais.

Tabela 10: Duração de interação e quantidade de desafios por versão do jogo

	Estática	Dinâmica
Tempo médio de interação	20 minutos e 36 segundos	24 minutos e 18 segundos
Número médio de questões	43	44,3
Tempo médio por questão	29 segundos	33 segundos

A Tabela 10 denota ainda a média de questões resolvidas, seja por acerto ou pulo, por jogador para cada versão, e o tempo médio por questão. A diferença nestes valores foi menos expressiva, com um aumento de 3% no número de questões resolvidas na versão dinâmica em comparação à estática e de 13,7% no tempo gasto por questão.

Descritas na Tabela 11 estão as respostas dos alunos às perguntas de opinião feitas ao final da interação, categorizadas pela versão do jogo. Para a pergunta *O jogo é divertido?*, um único aluno respondeu *Mais ou menos*, enquanto todos os demais responderam *Sim*. Não houve diferença expressiva então na opinião de diversão do jogo entre as versões, dentro das limitações da pergunta apresentada. A pergunta *O jogo parece com a aula de matemática?* teve a maior disparidade entre respostas nas diferentes versões, com 84,51% de alunos que interagiram com a versão dinâmica opinando que *Sim*, o jogo parece com a aula, comparados aos 70% desta opinião entre aqueles que interagiram com a versão estática. A similaridade de respostas entre as versões para a pergunta *O jogo é fácil ou difícil?* sugere que não houve diferença na percepção de dificuldade do jogo entre as versões, dentro do escopo da pergunta realizada.

Os comentários e *feedback* aberto dos alunos, que foram convidados a fazer sugestões e críticas a qualquer momento durante a interação, e a opinião das educadoras que acompanharam o processo servem como possível explicação para a similaridade das respostas

Tabela 11: Respostas por versão

Pergunta	Resposta	Estática (%)	Dinâmica (%)	Total (%)
O jogo é divertido?	Sim	100	98,59	99,30
	Mais ou menos	0	1,41	0,70
	Não	0	0	0
O jogo parece com a aula de matemática?	Sim	70	84,51	77,25
	Mais ou menos	25,41	12,68	19,04
	Não	4,29	2,82	3,55
O jogo é fácil ou difícil?	Fácil	45,51	45,07	45,29
	Nem fácil nem difícil	51,43	52,11	51,77
	Difícil	2,86	2,82	2,84

entre as duas versões para a questão da diversão. De acordo tanto com os alunos participantes quanto com as educadoras, o objeto, independentemente da versão, era motivador o suficiente para ser considerado divertido pelos alunos. Os termos usados pelos estudantes para descrever as qualidades da experiência foram "interativo", "animado", "bonito"e "diferente". Estes relatos vão de encontro às propostas de Prensky (2001) e McGonigal (2011) da efetividade do emprego de jogos na sala de aula e apontam adequação do objeto desenvolvido às especificações de características motivadoras de Malone e Lepper (1987).

A seguir, foram observados os dados coletados pelo jogo a respeito do desempenho dos jogadores. A Tabela 12 contém os resultados da interação dos jogadores com os desafios propostos pelo jogo, categorizados por versão. Apesar da percepção dos alunos, conforme medida pelas perguntas de opinião, não apontar diferença de dificuldade entre as versões, os dados de desempenho mostram que 37,97% dos resultados para desafios dinâmicos foram do tipo *Sucesso após Retroceder*, comparados aos 29,98% para desafios estáticos. Considerando resultados *Sucesso sem uso de Retroceder* como demonstração de domínio do jogador sobre o conteúdo do desafio, resultados *Pular* como carência do jogador com relação aos conteúdos e resultados *Sucesso após Retroceder* como similaridade do nível de desafio apresentado com a competência do jogador, a versão dinâmica mostrou-se capaz de propor desafios de nível similar à competência do jogador com maior frequência do que

a versão estática.

Tabela 12: Resultados por versão

Resultado	Estática (%)	Dinâmica (%)
Sucesso sem uso de Retroceder	59,11	53,88
Sucesso após Retroceder	29,98	37,97
Pular	10,91	8,16

Continuado a investigação acerca do desafio oferecido por cada versão, foram comparadas as respostas de opinião dos jogadores sobre a dificuldade do jogo com o desempenho destes na interação. A Tabela 13 categoriza os resultados dos alunos de acordo com suas respostas à pergunta *O jogo é fácil ou difícil?*, com a resposta *Nem fácil nem difícil* transcrita como *Médio*.

Tabela 13: Resultados por opinião de dificuldade

	Estática			Dinâmica		
	Fácil (%)	Médio (%)	Difícil (%)	Fácil (%)	Médio (%)	Difícil (%
Sucesso sem Retroceder	53,14	65,93	53,95	50,42	58,12	42,9
Sucesso após Retroceder	29,86	27,30	46,05	40,99	35,67	50,7
Pular	15,49	6,07	0,00	8,59	6,20	6,2

Alunos que interagiram com a versão estática e descreveram o jogo como *Fácil* formaram o grupo com a maior taxa de resultados *Pular*, 15,49%. O grupo com a segunda maior taxa de resultados *Pular*, com 8,59%, foi o de alunos que interagiram com a versão dinâmica e descreveram o jogo como *Fácil*. De acordo com estas informações, alunos que avaliaram o jogo como *Fácil* foram aqueles que encontraram a maior quantidade de questões cujo desafio era superior à sua habilidade, apontando uma possível disparidade entre a percepção e opinião de dificuldade dos jogadores e a dificuldade medida pelo seu desempenho.

Outra característica analisada nos resultados coletados durante o teste foi a expecta-

tiva de que a versão dinâmica, devido à natureza do algoritmo de Aprendizado por Reforço *on-line* implementado, teria maior taxa de escolha inadequada de desafios durante as primeiras iterações, enquanto o algoritmo está avaliando a competência do jogador. A fim de validar esta hipótese, foram medidas as ocorrências de resultados *Pular* durante as 5 primeiras iterações do algoritmo, correspondentes aos 15 primeiros desafios apresentados ao jogador. Este intervalo equivale a 33% da interação total dos alunos com o jogo. A Tabela 14 contém os valores encontrados para ocorrências de *Pular* nas primeiras 5 iterações do algoritmo na versão dinâmica. Para motivos de comparação, foram também medidas as ocorrências de *Pular* nos 15 primeiros desafios para a versão estática.

Tabela 14: Resultados *Pular* nos 15 primeiros desafios

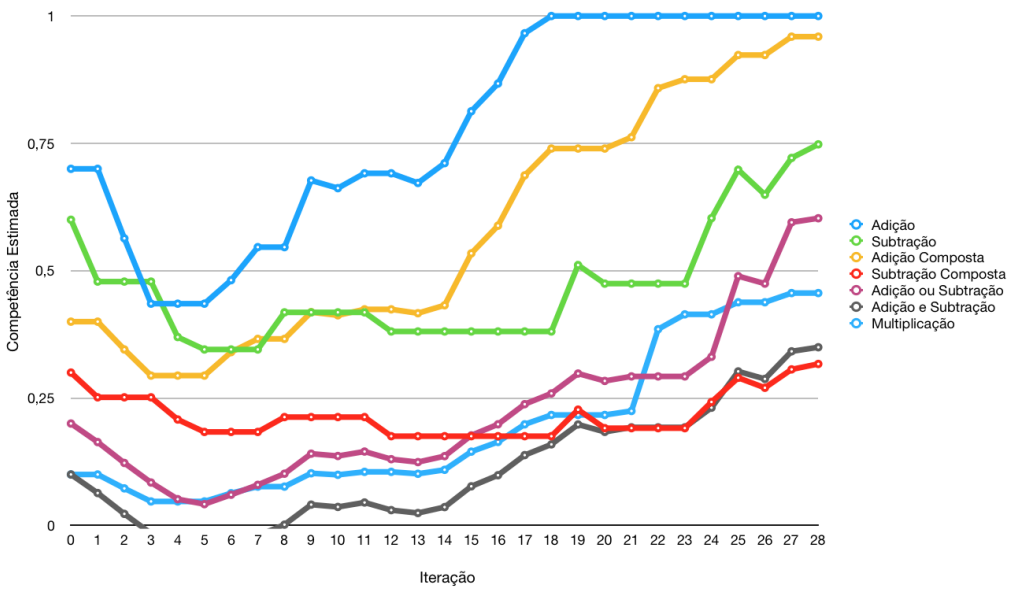
	Estática	Dinâmica
Total de resultados *Pular*	347	267
Resultados *Pular* nos 15 primeiros desafios	110	142
Porcentagem de resultados *Pular* nos 15 primeiros desafios	31,70%	53,18%

Conformando com a expectativa, a versão dinâmica concentra erros de escolha de desafio nas primeiras iterações. O primeiro um terço da interação dos jogadores com a versão dinâmica conteve 53,18% de todos os resultados *Pular* desta versão. Na versão estática as primeiras 15 questões, correspondendo à 34% do total de questões resolvidas desta versão, contiveram 31,70% dos resultados *Pular*.

Os dados coletados durante a interação com a versão dinâmica permitem ainda ilustrar a evolução da competência estimada dos alunos para cada categoria ao longo da iterações do algoritmo de Aprendizado por Reforço implementado. A Figura 19 contém um exemplo da evolução de estimativas de competência de um aluno ao longo dos 28 ciclos de escolha de desafio que ocorreram durante a interação deste aluno. Visto que o algoritmo desenvolvido apresenta questões em grupos de 3 antes de avaliar a próxima escolha, cada intervalo unitário no eixo das iterações corresponde a três desafios.

Notável neste gráfico é a tendência de redução da competência estimada para todas as categorias durante as primeiras iterações, condizente com os dados encontrados na Tabela 14 sobre a distribuição de resultados *Pular*. Além desta informação, este tipo de

Figura 19: Evolução da competência estimada de um jogador por categoria



Fonte: Imagem elaborada pelo autor

visualização pode ser de utilidade para professores ao expor as competências e carências de cada jogador com relação a cada categoria de desafio.

5 Conclusão e Trabalhos Futuros

Durante este trabalho foi desenvolvido um objeto de aprendizagem na forma de um jogo para o ensino de matemática abordando operações aritméticas de adição, subtração e multiplicação. O desenvolvimento do jogo se deu guiado pelas definições de motivações em jogos de Malone e Lepper (1987) e pelas definições de mecânicas de Järvinen (2009). Duas versões do jogo foram criadas: uma, estática, apresentando sempre uma mesma sequência de desafios; outra, dinâmica, empregado um algoritmo simples de Aprendizado por Reforço para escolher desafios que se adéquem ao nível de habilidade demonstrada pelo jogador, com o objetivo de promover mais engajamento e motivação no jogador com base nas teorias de Malone e Lepper (1987) e Csikszentmihalyi (1990).

O algoritmo empregado na versão dinâmica para a implementação de dificuldade adaptativa, estratégia esta baseada no trabalho de Hunicke (2005), foi escolhido após investigação de diferentes alternativas em Aprendizado de Máquina. A solução escolhida,

uma modificação do algoritmo de Aprendizado por Reforço usado no problema do bandido de k braços, descrito por Sutton e Barto (1998), foi criada com base na implementação de Juliani (2017). Limitada pela dificuldade de realização de treinamento prévio durante esta pesquisa, o algoritmo apresenta treinamento totalmente *on-line*, durante a própria interação com o aluno.

O objeto foi levado a teste com alunos de terceiro ano do ensino fundamental, e foram coletados dados de opinião dos participantes bem como de interação e desempenho no jogo. A análise destes resultados sugere possível validação da hipótese de que a versão adaptativa promove maior engajamento, principalmente na comparação de tempo de interação. Sendo uma atividade voluntária em que os participantes podiam encerrar a interação a qualquer momento, o tempo de interação 17,9% mais longo para a versão dinâmica sugere maior vontade de interação com o objeto por alunos que receberam esta versão, caracterizando maior motivação.

As respostas às perguntas de opinião dos jogadores a respeito da diversão e dificuldade do jogo foram similares entre as duas verões, com variações da ordem de 1%. A pergunta que dizia respeito à similaridade do jogo com a aula de matemática resultou em opiniões mais díspares entre as versões, com 84% dos alunos que interagiram com a versão dinâmica respondendo *Sim*, em contraste com os 70% na versão estática. Estes resultados não validam diretamente as hipóteses, não havendo indicação de que a versão dinâmica proporcionou experiência mais divertidas ou de dificuldade mais adequada à competência do jogador. Foi observado ainda nestas respostas de opinião um fenômeno que sugere uma disparidade entre a percepção dos alunos sobre a experiência e a realidade: alunos que descreveram o jogo como *Fácil* tiveram a maior taxa de resultados do tipo *Pular*, que indicam desafios de dificuldade superior à habilidade do jogador.

A hipótese de que a versão dinâmica promove maior aprendizado nos alunos não pôde ser comprovada pelo experimento realizado, visto que não foi possível medir especificamente o conhecimento de matemática dos alunos antes e depois da interação, mas os dados de desempenho dos jogadores indicam que a versão dinâmica promove desafios de nível mais adequado à habilidade de cada jogador do que a versão estática. Resultados do tipo *Sucesso após Retroceder*, considerados indicação de desafio com nível adequado de dificuldade, foram mais frequentes na versão dinâmica (37,97% de todos os resultados) do

que na versão estática (29,98% de todos os resultados). Esta mesma diferença se reflete em taxas menores de resultados *Sucesso sem uso de Retroceder*, considerados indicação de desafio de nível de dificuldade inferior à habilidade do jogador, e resultados *Pular*, considerados indicação de desafio de nível de dificuldade superior à habilidade do jogador.

Os resultados sugerem então que a versão dinâmica oferece vantagens com relação à estática no que diz respeito à engajamento dos jogadores e manutenção de nível adequado de desafio. A contribuição deste trabalho se dá ainda por meio da confirmação da empregabilidade de definições de características motivadoras e de engajamento como base para o desenvolvimento de jogos enquanto objetos de aprendizagem, elicitada pela quase unânime avaliação do jogo como divertido pelos alunos e da validação do objeto por educadores.

Com base na investigação realizada sobre motivação e técnicas de dificuldade adaptativa, acredita-se que resultados mais expressivos em termos de validação das hipóteses sejam possíveis em trabalhos futuros, livres das limitações encontradas por este. A implementação de um algoritmo de Aprendizado por Reforço mais sofisticado, como os apresentados por Sutton e Barto (1998) e Hester, Quinlan e Sonte (2011), com a possibilidade de treinamento com dados reais anterior à aplicação pode evitar a baixa acurácia exibida pelo algoritmo implementado neste trabalho nas primeiras iterações. De maneira similar, uma implementação de dificuldade adaptativa baseada em Aprendizado Supervisionado, dada a possibilidade de treinamento do sistema com dados reais anterior à aplicação, tem o potencial de apresentar resultados satisfatórios na categorização dos alunos e escolha de atividades a propor.

A hipótese de que a versão dinâmica promove melhor aprendizado pode ser melhor validada por meio da aplicação de um teste para medir o conhecimento dos alunos antes da interação do objeto e um novo teste após a interação, possivelmente ao longo de múltiplas sessões, a fim de medir a evolução do conhecimento destes estudantes.

Existe também a possibilidade de empregar um sistema baseado no desenvolvido neste trabalho para gerar relatórios do desempenho de cada aluno para o acompanhamento dos professores, usando dados como as métricas de competência estimadas como ilustrado no gráfico da Figura 19.

REFERÊNCIAS BIBLIOGRÁFICAS

ABADI, M. et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. Disponível em: <https://www.tensorflow.org/>.

ALPAYDIN, E. *Introduction to machine learning*. [S.l.]: MIT press, 2009.

ANDRADE, G. et al. Challenge-sensitive action selection: an application to game balancing. In: IEEE. *Intelligent Agent Technology, IEEE/WIC/ACM International Conference on*. [S.l.], 2005. p. 194–200.

BOSER, B. E.; GUYON, I. M.; VAPNIK, V. N. A training algorithm for optimal margin classifiers. In: ACM. *Proceedings of the fifth annual workshop on Computational learning theory*. [S.l.], 1992. p. 144–152.

BRASIL. *Base Nacional Comum Curricular (BNCC)*. Brasília, 2018.

CARDOSO, M. R. G.; GHELLI, K. G. M.; OLIVEIRA, G. S. O uso de jogos como metodologia de ensino de matemática na educação infantil. *Cadernos da Fucamp*, v. 16, n. 27, p. 12–30, 2017.

COHEN, K. C. The effects of two simulation games on the opinions and attitudes of selected sixth, seventh and eight grade students. Johns Hopkins University, Maryland, 1969.

CORBÁLAN, F. Estrategias utilizadas por los alumnos de secundaria en la resolución de juegos. *SUMA*, v. 23, p. 21–32, 1996.

CORDOVA, D. I.; LEPPER, M. R. Intrinsic motivation and the process of learning: Beneficial effects of contextualization, personalization and choice. *Journal of Educational Psychology*, v. 88, n. 4, p. 715–730, 1996.

CRAWFORD, C. *The Art of Computer Game Design*: Reflections of a master game designer. California: Osborne McGraw-Hill, 1982.

CRUZ, P. *De Olho nas Metas*. [S.l.], 2015. 42 p. Disponível em: <https://www-.todospelaeducacao.org.br//arquivos/biblioteca/olho_metas_2015_16_final.pdf>.

CSIKSZENTMIHALYI, M. *Beyond Boredom and Anxiety*. San Francisco: Jossey-Bass, 1975.

CSIKSZENTMIHALYI, M. *Flow: The Psychology of Optimal Experience*. [S.l.: s.n.], 1990. 74 p.

DJAOUTI, D. et al. A gameplay definition through videogame classification. *International Journal of Computer Games Technology*, 2008.

FREUND, Y.; SCHAPIRE, R. E. Large margin classification using the perceptron algorithm. *Machine learning*, Springer, v. 37, n. 3, p. 277–296, 1999.

GARRIS, R.; AHLERS, R.; DRISKELL, J. E. Games, motivation, and learning: A research and practice model. *Simulation & Gaming*, p. 441–467, 2002.

GRANDO, R. C. *O jogo e a Matemática no contexto da sala de aula*. São Paulo: Paulus, 2004.

GREENE, J.; D'OLIVEIRA, M. *Learning to use statistical tests in psychology*. [S.l.]: McGraw-Hill Education (UK), 2005.

GUEVARA, C.; AGUILAR, J.; GONZALEZ-ERAS, A. The model of adaptive learning objects for virtual environments instanced by the competencies. *Advances in Science, Technology and Engineering Systems Journal*, v. 2, p. 345–355, 05 2017.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. *The Elements of Statistical Learning*. New York: Springer, 2009.

HEARST, M. A. et al. Support vector machines. *IEEE Intelligent Systems and their applications*, IEEE, v. 13, n. 4, p. 18–28, 1998.

HESTER, T.; QUINLAN, M.; SONTE, P. A real-time model-based reinforcement learning architecture for robot control. *aeXiv preprint arXiv:1105.1749*, 2011.

HUIZINGA, J. *Homo Ludens*: A study of the play-element in culture. London: Touledge & Kegan Paul Ltd, 1949.

HUNICKE, R. The case for dynamic difficulty adjustment in games. In: ACM. *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology*. [S.l.], 2005. p. 429–433.

JAIN, A. K.; MAO, J.; MOHIUDDIN, K. M. Artificial neural networks: A tutorial. *Computer*, IEEE, v. 29, n. 3, p. 31–44, 1996.

JÄRVINEN, A. *Games without Frontiers*: Methods for game studies and design. [S.l.]: VDM Verlag, 2009.

JULIANI, A. *Unity AI-themed Blog Entries*. 2017. Disponível em: <https://blogs-.unity3d.com/2017/06/26/unity-ai-themed-blog-entries/>.

KOHONEN, T. The self-organizing map. *Proceedings of the IEEE*, IEEE, v. 78, n. 9, p. 1464–1480, 1990.

KURILOVAS, E.; KUBILINSKIENE, S.; DAGIENE, V. Web 3.0–based personalisation of learning objects in virtual learning environments. *Computers in Human Behavior*, Elsevier, v. 30, p. 654–662, 2014.

LIOU, D.-R.; LIOU, J.-W.; LIOU, C.-Y. *Learning Behaviors of Perceptron*. [S.l.], 2013.

LIU, C. et al. Dynamic difficulty adjustment in computer games through real-time anxiety-based affective feedback. *International Journal of Human-Computer Interaction*, Taylor & Francis, v. 25, n. 6, p. 506–529, 2009.

LLOYD, S. Least squares quantization in pcm. *IEEE transactions on information theory*, IEEE, v. 28, n. 2, p. 129–137, 1982.

MAINIERI, B. O. et al. Development and assessment of an adaptive difficulty arithmetic game based learning object. 2018.

MALONE, T. W.; LEPPER, M. R. Making learning fun: A taxonomy of intrinsic motivations for learning. In: SNOW, R. E.; FARR, M. J. (Ed.). *Aptitude, Learning and Instruction*. New Jersey: Lawrence Elrbaum Associates, 1987. cap. 10, p. 223–253.

MCGONIGAL, J. *Reality is Broken*: Why games make us better and how they can change the world. New York: The Penguin Press, 2011.

MERRILL, H. A. Why students fail in mathematics. *The Mathematics Teacher*, National Council of Teachers of Mathematics, v. 11, n. 2, p. 45–56, 1918. ISSN 00255769. Disponível em: <http://www.jstor.org/stable/27950169>.

MICHAEL, D.; CHEN, S. *Serious Games: Games that Educate, Train and Inform*. Thomson Course Technology, 2006. ISBN 9781592006229. Disponível em: <https://books.google.com.br/books?id=49kTAQAAIAAJ>.

MICHALSKI, R. S.; CARBONELL, J. G.; MITCHELL, T. M. *Machine learning: An artificial intelligence approach*. [S.l.]: Springer Science & Business Media, 2013.

MINSKY, M.; PAPERT, S. A. *Perceptrons: An introduction to computational geometry*. [S.l.]: MIT press, 2017.

MISSURA, O.; GÄRTNER, T. Player modeling for intelligent difficulty adjustment. In: SPRINGER. *International Conference on Discovery Science*. [S.l.], 2009. p. 197–211.

OLIPHANT, T. E. A guide to numpy. Trelgol Publishing, 2006.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011.

PRENSKY, M. *The Digital Game-Based Learning Revolution*. [S.l.]: McGraw-Hill, 2001.

ROSENBLATT, F. *The perceptron, a perceiving and recognizing automaton Project Para*. [S.l.]: Cornell Aeronautical Laboratory, 1957.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. *Learning internal representations by error propagation*. [S.l.], 1985.

RUSSELL, S.; NORVIG, P. *Artificial Intelligence: A Modern Approach*. Pearson, 2016. (Always learning). ISBN 9781292153964. Disponível em: <https://books.google.com.br-/books?id=XS9CjwEACAAJ>.

SAMPAYO-VARGAS, S. et al. The effectiveness of adaptive difficulty adjustments on students' motivation and learning in an educational computer game. *Computers & Education*, Elsevier, v. 69, p. 452–462, 2013.

SICART, M. Defining game mechanics. *The International Journal of Computer Game Research*, v. 8, n. 2, 2008.

SILVA, M. P.; SILVA, V. do N.; CHAIMOWICZ, L. Dynamic difficulty adjustment on MOBA games. *CoRR*, abs/1706.02796, 2017. Disponível em: <http://arxiv.org/abs-/1706.02796>.

SPRONCK, P.; SPRINKHUIZEN-KUYPER, I.; POSTMA, E. Difficulty scaling of game ai. In: *Proceedings of the 5th International Conference on Intelligent Games and Simulation (GAME-ON 2004)*. [S.l.: s.n.], 2004. p. 33–37.

SREENIVAS, S. B. H. *Intelligent Tutoring Systems Using Reinforcement Learning*. Tese (Doutorado) — Indian Institute of Technology Madras, 2007.

STEVENS, R. et al. Artificial neural network-based performance assessments. *Computers in Human Behavior*, Elsevier, v. 15, n. 3, p. 295–313, 1999.

SUTTON, R. S.; BARTO, A. G. *Reinforcement Learning: An Introduction*. Cambridge: MIT Press, 1998.

THOMAZ, P. H. B.; MEGID, M. A. B. A. Recursos didáticos no ensino da matemática: o jogo como estratégia de ensino e o programa ler e escrever. *Currículo sem Fronteiras*, v. 17, n. 3, p. 833–847, 2017.

TIBSHIRANI, R.; WALTHER, G.; HASTIE, T. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, Wiley Online Library, v. 63, n. 2, p. 411–423, 2001.

VESANTO, J.; ALHONIEMI, E. Clustering of the self-organizing map. *IEEE TRANSACTIONS ON NEURAL NETWORKS*, v. 11, n. 3, 2000.

VOGEL, J. J. et al. Computer gaming and interactive simulations for learning: A meta-analysis. *J. Educational Computing Research*, v. 34, n. 3, p. 229–243, 2006.

WHITTON, N. *Learning with digital games: A practical guide to engaging students in higher education*. [S.l.]: Routledge, 2009. 112 p.

WHITTON, N. J. *An investigation into the potential of collaborative computer game-based learning in higher education*. Tese (Doutorado) — Edinburgh Napier University, 2007.

WILEY, D. A. et al. Connecting learning objects to instructional design theory: A definition, a metaphor, and a taxonomy. *The instructional use of learning objects*, v. 2830, n. 435, p. 1–35, 2000.

ZIMMERMAN, E. Narrative, interactivity, play, and games: Four naughty concepts in need of discipline. *First Person*, MIT Press, Cambridge, p. 154–163, 2004.

APÊNDICE

APÊNDICE A - Desafios no Protótipo

Número do Desafio	Valor Alvo	Operandos Disponíveis	Operações Disponíveis
1	3	1,2	Adição
2	4	2,2	Adição
3	5	2,3	Adição
4	5	1,4	Adição
5	9	3,6	Adição
6	6	2,3,4	Adição
7	5	1,2,3	Adição
8	8	2,4,6	Adição
9	7	1,3,6	Adição
10	9	3,3,4,5	Adição
11	7	2,3,5,6	Adição
12	8	1,3,4,5	Adição
13	9	2,2,3,5,6	Adição
14	1	2,3	Subtração
15	3	2,5	Subtração
16	3	5,8	Subtração
17	5	2,7	Subtração
18	4	1,2,5	Subtração
19	2	3,6,8	Subtração
20	3	4,5,6,7	Subtração
21	5	2,3,6	Adição, Subtração
22	4	3,5,7	Adição, Subtração
23	7	2,3,4	Adição, Subtração
24	6	1,2,3,4,5	Adição, Subtração
25	4	1,2,3,5,6	Adição, Subtração
26	5	3,4,6	Adição, Subtração
27	2	3,8,9	Adição, Subtração
28	4	2,5,7	Adição, Subtração
29	1	2,4,6,9	Adição, Subtração
30	6	3,5,7,8	Adição, Subtração
31	9	1,4,6,7	Adição, Subtração
32	10	3,5,8,9	Adição, Subtração
33	12	1,2,4,7	Adição, Subtração
34	11	1,2,2,3,7	Adição, Subtração
35	16	1,2,3,4,5,6	Adição, Subtração
36	17	6,7,8,8,9,9	Adição, Subtração
37	8	1,2,4	Adição, Subtração, Multiplicação
38	10	2,3,4	Adição, Subtração, Multiplicação

APÊNDICE B - Desafios na Versão Final

Categoria	Valor Alvo	Operandos Disponíveis	Operações Disponíveis	Valor de Complexidade Relativa
Adição Simples	6	2; 4	Adição	0.2
Adição Simples	8	3; 5	Adição	0.3
Adição Simples	3	1; 2	Adição	0.25
Adição Simples	14	5; 9	Adição	0.35
Adição Simples	13	3; 10	Adição	0.35
Adição Simples	6	5; 1	Adição	0.2
Adição Simples	17	8; 9	Adição	0.4
Adição Simples	11	4; 7	Adição	0.35
Adição Simples	12	4; 8	Adição	0.35
Adição Simples	12	4;6; 6;10	Adição	0.7
Adição Simples	8	2;3;4;5;7	Adição	0.85
Adição Simples	9	2;3;5;6;8	Adição	0.8
Adição Simples	6	1;2;3;4;7	Adição	0.75
Adição Simples	14	2;4;5;6;7;9	Adição	0.85
Adição Simples	12	2; 4; 5; 8	Adição	0.8f)
Subtração Simples	3	1; 4	Subtração	0.3
Subtração Simples	4	3; 7	Subtração	0.32
Subtração Simples	4	9; 5	Subtração	0.35
Subtração Simples	3	1; 4	Subtração	0.32
Subtração Simples	7	2; 9	Subtração	0.34
Subtração Simples	2	5; 7	Subtração	0.35
Subtração Simples	5	4; 9	Subtração	0.31
Subtração Simples	6	2;3;5;10;16	Subtração	0.65
Subtração Simples	7	2;3;5;10;16	Subtração	0.7
Subtração Simples	3	4;6;7;11;12	Subtração	0.75
Subtração Simples	5	3;4;10;15;19	Subtração	0.85
Subtração Simples	5	1;2;7; 10;16	Subtração	0.75f)
Adição Composta	8	2; 2; 4	Adição	0.3
Adição Composta	12	2; 3; 7	Adição	0.32
Adição Composta	15	2; 6; 7	Adição	0.33
Adição Composta	14	4; 4; 6	Adição	0.34
Adição Composta	7	1; 3; 3; 5	Adição	0.45
Adição Composta	12	2; 4; 6; 7; 9	Adição	0.50
Adição Composta	14	3; 4; 5; 7	Adição	0.55
Adição Composta	9	2; 3; 4; 4	Adição	0.55
Adição Composta	10	1; 2; 4; 4; 5	Adição	0.6
Adição Composta	15	3; 3; 3; 9	Adição	0.65
Adição Composta	6	2; 2; 2; 3; 5	Adição	0.55
Adição Composta	10	1; 2; 4; 5	Adição	0.5
Adição Composta	18	1; 3; 4; 6; 8	Adição	0.65f)

Subtração Composta	1	2; 4; 7	Subtração	0.3
Subtração Composta	3	2; 6; 11	Subtração	0.31
Subtração Composta	3	1; 10; 14	Subtração	0.32
Subtração Composta	6	4; 5; 15	Subtração	0.35
Subtração Composta	4	1; 2; 7; 8	Subtração	0.5
Subtração Composta	2	1; 5; 6; 9	Subtração	0.51
Subtração Composta	5	1; 2; 4; 8	Subtração	0.52
Subtração Composta	3	2; 2; 6; 7	Subtração	0.55
Subtração Composta	6	2; 5; 7; 8; 16	Subtração	0.65
Subtração Composta	5	1; 4; 7; 8; 14	Subtração	0.68
Subtração Composta	5	1; 2; 3; 9; 10	Subtração	0.7f)
Adição e Subtração	3	5;5;7	Adição; Subtração	0.5
Adição e Subtração	12	4;6;14	Adição; Subtração	0.55
Adição e Subtração	14	2;3;6;10	Adição; Subtração	0.6
Adição e Subtração	10	2;5;7	Adição; Subtração	0.6
Adição e Subtração	6	2;5;9;10;12	Adição; Subtração	0.7
Adição e Subtração	8	1;3;10;14	Adição; Subtração	0.72
Adição e Subtração	18	4;5;6;11;16	Adição; Subtração	0.75
Adição e Subtração	10	4;5;6;9	Adição; Subtração	0.76
Adição e Subtração	12	4;4;5;9;13	Adição; Subtração	0.9
Adição e Subtração	11	2;3;5;7	Adição; Subtração	0.8
Adição e Subtração	5	1;1;8;8;11;11	Adição; Subtração	0.85
Adição e Subtração	14	2;4;6;7;11	Adição; Subtração	0.87f)
Adição ou Subtração	6	2;4;9	Adição; Subtração	0.4
Adição ou Subtração	5	1;3;8	Adição; Subtração	0.4
Adição ou Subtração	8	4;4;7;9	Adição; Subtração	0.42
Adição ou Subtração	9	2;3;4;8;11	Adição; Subtração	0.45
Adição ou Subtração	5	1;3;3;7;8	Adição; Subtração	0.48
Adição ou Subtração	8	2;2;4;7;14	Adição; Subtração	0.65
Adição ou Subtração	7	1;3;3;5;13	Adição; Subtração	0.71
Adição ou Subtração	6	1;1;3;10	Adição; Subtração	0.75
Adição ou Subtração	8	1;4;6;15	Adição; Subtração	0.8f)
Multiplicação	4	2;2	Multiplicação	0.2
Multiplicação	8	1;2;4	Multiplicação	0.22
Multiplicação	9	3;3	Multiplicação	0.25
Multiplicação	10	2;4;5	Multiplicação	0.4
Multiplicação	8	2;3;4	Multiplicação	0.5
Multiplicação	16	2;4;8	Multiplicação	0.5
Multiplicação	8	2;2;2;3	Multiplicação	0.65
Multiplicação	16	1;2;2;3;4;5	Multiplicação	0.75
Multiplicação	18	1;2;2;3;3;4	Multiplicação	0.85f)

APÊNDICE C - Algoritmo em C# para Dificuldade Adaptativa

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using System.Linq;

public class MathAgent {

 public float[][] value_table; // The array containing the
 //estimated competency (0.0 - 1.0) of the player for each
 //concept
 // Simple Add, Simple Sub, 2+ Add, 2+ Sub, Add Sub, Add
 //extras, Sub extras

 public float confidence = 0.08f; //softmax temperature;
 //lower = more likely to pick best valued

 float learning_rate = 0.35f; // The rate at which to
 //update the value estimates given a reward.

 public float[][] correlation_table;
 public float[] c_add =
     new float[]{1.0f,0,0.4f,0,0.3f,0.3f,0.2f}; //Simple Add
 //correlates to 2+ Add and Add Extras
 public float[] c_sub =
     new float[]{0,1.0f,0,0.4f,0.3f,0.3f,0};
 public float[] c_2add =
     new float[]{1.0f,0,1.0f,0,0.4f,0.4f,0.35f};
 public float[] c_2sub =
     new float[]{0,1.0f,0,1.0f,0.4f,0.4f,0};
 public float[] c_addNsub =
```

```
    new float[]{1.0f,1.0f,0.6f,0.6f,1.0f,0.8f,0.3f};
public float[] c_addORsub =
    new float[]{0.6f,0.6f,0.3f,0.3f,0.45f,1.0f,0.15f};
public float[] c_mult =
    new float[]{0.8f,0,0.6f,0,0,0,1.0f};

public MathAgent(int stateSize, int actionSize) {
 value_table = new float[stateSize][];
 for (int i = 0; i < stateSize; i++) {
  value_table[i] =
  new float[]{0.70f,0.60f,0.40f,
  0.30f,0.10f,0.20f,0.10f};
 }
 correlation_table =
     new float[][]{c_add,c_sub,c_2add,c_2sub,
     c_addNsub,c_addORsub,c_mult};
}

//Should strive not to pick either a lowly-rated option nor
//a very highly-rated one: not too hard, not too easy
//Pick question within group according to player skill
//questions can be easier or harder in the same category


//0.83f is the threshold for "too easy" - such
//actions will be avoided
public int PickAction(int state) {
 int selectedElement = 0;
 List<int> validActions = new List<int>();
 List<float> validWeights = new List<float>();
 for (int i = 0; i < value_table[state].Count(); i++) {
  if (value_table[state][i] < 0.83f) {
```

```
  validActions.Add (i);
  validWeights.Add(value_table[state][i]);
 }
}

if (validActions.Count () < 1) {
 return
 Random.Range (0, value_table [state].Count ());
}

if (validActions.Count > 3) {
 int indexMin = validWeights.IndexOf (validWeights.Min ());
 validActions.RemoveAt (indexMin);
 validWeights.RemoveAt (indexMin);

 indexMin =
     validWeights.IndexOf (validWeights.Min ());
 validActions.RemoveAt (indexMin);
 validWeights.RemoveAt (indexMin);
}

float[] relativeWeights =
    new float[validWeights.Count()];
for (int i = 0; i < validWeights.Count (); i++) {
 relativeWeights [i] =
     1 - (Mathf.Abs (0.65f - validWeights [i]));
}

float[] probs =
    softmax (relativeWeights.ToArray(), confidence);

float cumulative = 0.0f;
```

```
 float total = 0.0f;
 for (int i = 0; i < probs.Length; i++) {
  total += probs [i];
 }

 float diceRoll = Random.Range (0f, total);
 for (int i = 0; i < probs.Length; i++)
 {
  cumulative += probs[i];
  if (diceRoll < cumulative)
  {
   selectedElement = validActions[i];
   break;
  }
 }
 return selectedElement;
}

//Should update not only values referring to current state,
//but also related states, according to the correlation table

public void UpdatePolicy(int state, int action, float reward) {
 float value_increase =
   learning_rate * (reward - value_table[state] [action]);
 float correlation_mod = value_increase > 0 ? 1.0f : 0.65f;
 for (int i = 0; i < value_table [state].Count (); i++) {
  value_table [state] [i] +=
   correlation_mod * value_increase * correlation_table [action] [i];
  value_table [state][i] = Mathf.Min (value_table [state][i], 1.0f);
 }
}
```

```
 float[] softmax(float[] values, float temp) {
  float[] softmax_values = new float[values.Length];
  float[] exp_values = new float[values.Length];
  for (int i = 0; i < values.Length; i++) {
   exp_values [i] = Mathf.Exp (values [i] / temp);
  }
  for (int i = 0; i < values.Length; i++) {
   softmax_values[i] = exp_values[i] / exp_values.Sum();
  }
  return softmax_values;
 }

 int GetRandomGivenWeights(float[] weights){
  int selectedElement = 0;
  float cumulative = 0.0f;
  float total = 0.0f;
  for (int i = 0; i < weights.Length; i++) {
   total += weights[i];
  }
  float diceRoll = Random.Range (0f, total);
  for (int i = 0; i < weights.Length; i++){
   cumulative += weights[i];
   if (diceRoll < cumulative){
    selectedElement = i;
    break;
   }
  }
  return selectedElement;
 }
}
```