Ministério da Educação
Secretaria de Educação Profissional e Tecnológica
Instituto Federal Catarinense
Campus Videira

MATHEUS ALVES BUENO MACHADO

APLICAÇÃO DE REDES NEURAIS CONVOLUCIONAIS PARA DETECÇÃO DE ANOMALIAS VISUAIS EM JOGOS DIGITAIS

Videira
2025

MATHEUS ALVES BUENO MACHADO

APLICAÇÃO DE REDES NEURAIS CONVOLUCIONAIS PARA DETECÇÃO DE ANOMALIAS VISUAIS EM JOGOS DIGITAIS

Projeto de Trabalho de Conclusão de Curso apresentado ao Curso de graduação em Ciência da Computação do Instituto Federal Catarinense – Campus Videira para obtenção do título de bacharel em Ciência da Computação.
Orientador: Rafael Antonio Zanin

Videira
2025

MATHEUS ALVES BUENO MACHADO

APLICAÇÃO DE REDES NEURAIS CONVOLUCIONAIS PARA DETECÇÃO DE ANOMALIAS VISUAIS EM JOGOS DIGITAIS

Este Trabalho de Curso foi julgado adequado para a obtenção do título de Bacharel em Ciência da Computação e aprovado em sua forma final pelo curso de graduação em Ciência da Computação do Instituto Federal Catarinense – Campus Videira.

Videira (SC), dia de mês de ano

Rafael Antonio Zanin
Instituto Federal Catarinense – Campus Videira

BANCA EXAMINADORA

FULANO
Instituto Federal Catarinense – Campus Videira

BELTRANO
Instituto Federal Catarinense – Campus Videira

Dedico este trabalho ...

AGRADECIMENTOS

agradecimentos

RESUMO

Neste trabalho Palavras-chave: palavra 1. palavra 2. palavra 3

ABSTRACT

This is the abstract of this work.

Key-words: Word 1. Word 2. Word 3

LISTA DE ILUSTRAÇÕES

LISTA DE ACRÔNIMOS

AF	Função de Ativação (do inglês *Activation Function*).
AM	Aprendizado de Máquina.
CNN	Rede Neural Convolucional (do inglês *Convolutional Neural Networks*).
ReLU	Rectified Linear Unit.
RNA	Rede Neural Artificial.

SUMÁRIO

1 INTRODUÇÃO

Os jogos de videogames e computadores conquistaram um espaço importante na vida de crianças, jovens e adultos e hoje é um dos setores que mais cresce na indústria de mídia e entretenimento (Savi e Ulbricht (2008)). Além do entretenimento, os jogos têm se expandido como ferramentas de educação, inclusão e até reabilitação, exigindo cada vez mais atenção a aspectos como qualidade visual, acessibilidade e integridade do gameplay.

Bernardo et al. (2020) destacam que diversas áreas da realidade social estão sendo gamificadas, ou seja, estão usando elementos de jogos e aplicando-os a outras áreas como educação, trabalho, terapia, negócios, guerra, desenvolvimento social, relacionamentos, entre outros.

Dessa forma, garantir que o ambiente do jogo esteja livre de falhas visuais, bugs ou comportamentos inesperados é um desafio contínuo para desenvolvedores. Tais anomalias podem não apenas comprometer a experiência do jogador, mas também dificultar a navegação e compreensão de elementos do jogo por pessoas com deficiências visuais ou cognitivas. Isto mostra a importância de soluções inteligentes que ajudem na detecção de problemas visuais, tanto para garantir a qualidade quanto a acessibilidade. Com o crescimento da indústria de jogos digitais, especialmente impulsionado pelos esportes eletrônicos, que atraem milhões de espectadores e movimentam grandes premiações em competições organizadas, a exigência por experiências visuais consistentes e inclusivas torna-se ainda mais essencial (Bernardo et al. (2020)).

Neste contexto, este trabalho tem como objetivo explorar o uso de Rede Neural Convolucional (do inglês *Convolutional Neural Networks*) (CNN) para a detecção automática de anomalias visuais em jogos digitais. A proposta envolve a análise de imagens extraídas de jogos para identificar padrões visuais que divergem do comportamento esperado, como objetos ausentes, sobreposição incorreta de elementos ou mudanças súbitas na interface. A partir disso, pretende-se propor um sistema que possa ser aplicado no suporte ao desenvolvimento de jogos digitais mais consistentes e confiáveis.

1.1 JUSTIFICATIVA

Durante o desenvolvimento de jogos digitais, é comum surgirem erros visuais como texturas corrompidas, objetos fora de lugar e falhas gráficas. Esses problemas afetam negativamente a jogabilidade, a imersão do jogador e a acessibilidade, especialmente para pessoas com

deficiências visuais ou cognitivas. Garantir um ambiente visual estável e livre de anomalias é essencial para oferecer uma experiência de jogo de qualidade.

Esses erros são comumente chamados de glitches. De acordo com Pinglestudio (2025), glitches são erros inesperados de software ou falhas dentro de jogos digitais que resultam em comportamentos não intencionais ou anomalias gráficas. Essas falhas podem variar desde pequenos problemas visuais até defeitos graves que impedem o progresso do jogador.

Para ilustrar esse tipo de problema, a imagem a seguir mostra o famoso glitch do *MissingNo* no jogo *Pokémon Red and Blue*, lançado para o *Game Boy*, um console portátil da Nintendo lançado em 1989, que marcou época por permitir jogos em cartucho em um dispositivo compacto e acessível (Nintendo (2025)).

Essa anomalia ocorre quando o jogador executa uma sequência específica de ações que leva o jogo a acessar dados não planejados na memória, resultando na aparição de um "Pokémon"corrompido (IGN Brasil (2020)). Visualmente, o *MissingNo* se manifesta como uma figura distorcida, composta por fragmentos de sprites de outros Pokémon e elementos gráficos do jogo, o que compromete completamente a coerência visual da interface (IGN Brasil (2020)).

Esse tipo de falha evidencia como erros na manipulação de dados e sprites podem comprometer a percepção do jogador e até gerar comportamentos inesperados na mecânica do jogo (IGN Brasil (2020)).

Figura 1 – Anomalia em Jogo

Fonte: GameBlast (2015)

Métodos tradicionais de detecção de erros, baseados em testes manuais, são lentos, caros e nem sempre eficazes. Por isso, há uma necessidade crescente por soluções automatizadas

que auxiliem nesse processo. O uso de CNNs surge como uma alternativa promissora, pois permite identificar falhas visuais de forma precisa e eficiente, tanto durante o desenvolvimento quanto em tempo real.

Segundo Stephens (2024), o processo tradicional de playtesting é logisticamente complexo e demorado, exigindo a participação simultânea de centenas de jogadores, além de apresentar dificuldades técnicas como pareamento, confiabilidade dos servidores e coleta de dados por meio de questionários.

A automação desses testes, incluindo a simulação de jogabilidade e a detecção de anomalias, representa uma solução viável para reduzir custos, aumentar a eficiência e melhorar a qualidade geral do produto final.

Este trabalho se justifica pela proposta de aplicar CNNs na detecção de anomalias visuais em jogos digitais, contribuindo para a criação de produtos mais consistentes, acessíveis e tecnicamente aprimorados. A pesquisa visa oferecer uma abordagem prática e inteligente para melhorar a qualidade visual dos jogos, otimizando o processo de desenvolvimento e beneficiando a experiência do usuário final.

1.2 PROBLEMA DE PESQUISA

Como as CNNs podem ser aplicadas na detecção de anomalias visuais em jogos digitais, contribuindo para a melhoria da qualidade gráfica, da acessibilidade e da experiência do jogador?

1.3 OBJETIVOS

1.3.1 Objetivo Geral

Investigar como as CNNs podem ser aplicadas na detecção de anomalias visuais em jogos digitais.

1.3.2 Objetivos Específicos

- Identificar os principais tipos de anomalias visuais em jogos digitais.
- Investigar abordagens existentes que utilizam CNNs para detecção de anomalias visuais.
- Desenvolver um modelo de CNN para identificar anomalias visuais em cenas de jogos.

- Avaliar a precisão e eficiência do modelo proposto.
- Analisar os benefícios e limitações do uso de CNN na detecção de anomalias em jogos digitais.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta os principais conceitos teóricos que embasam o desenvolvimento deste trabalho, cuja proposta está inserida no contexto de teste de software em jogos digitais. Inicialmente, na Seção 2.1, é abordado o cenário dos jogos digitais no Brasil, destacando seu crescimento expressivo, relevância cultural e impacto econômico. Na Seção 2.2, discute-se o conceito de anomalias visuais em jogos digitais, suas categorias mais comuns, causas e implicações na experiência do usuário.

A Seção 2.3 introduz os fundamentos do Aprendizado de Máquina (AM), com foco nas Redes Neurais Artificiais (RNAs), especialmente nas CNNs, explorando suas arquiteturas, funcionamento e aplicações na detecção de padrões visuais. Na Seção 2.4, são apresentadas as principais métricas utilizadas para avaliar o desempenho de modelos baseados em RNAs.

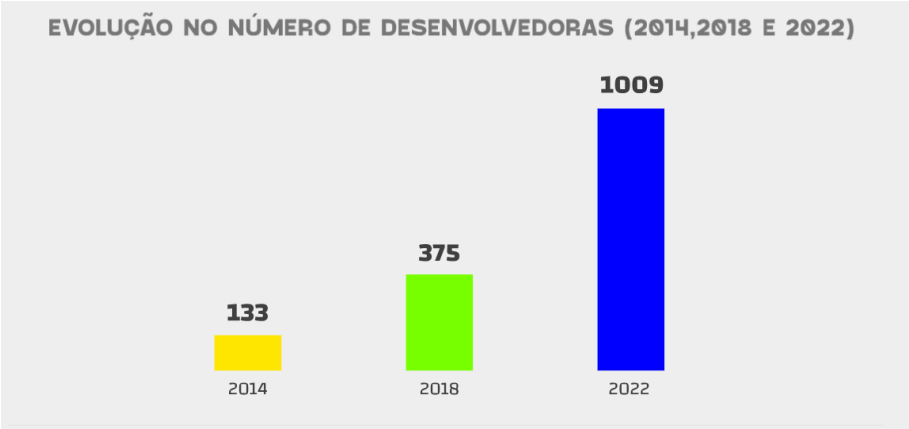
A Seção 2.5 trata dos tipos de classificação em redes neurais, com ênfase nas abordagens binária e multiclasse. Na Seção 2.6 discute o processo de teste de software em jogos digitais, destacando seus desafios específicos e o potencial da automação por meio de técnicas de visão computacional. Por fim, a Seção 2.7 apresenta os trabalhos correlatos, destacando pesquisas anteriores sobre o tema e situando a proposta no contexto científico atual.

2.1 O CENÁRIO DOS JOGOS DIGITAIS NO BRASIL

A indústria de jogos digitais tem se consolidado como um dos segmentos mais relevantes do mercado global de entretenimento. Representando uma forma de expressão cultural, artística e tecnológica, os jogos digitais atingem públicos de todas as idades e estão presentes em diversas plataformas, refletindo o avanço das tecnologias digitais na sociedade contemporânea (Araújo e Leão (2024)).

No Brasil, esse fenômeno tem ganhado proporções significativas, impulsionado pela popularização de dispositivos móveis, pelo aumento do acesso à internet e pela ampliação do consumo de conteúdos digitais. Segundo a Abragames (2022), entre 2014 e 2022 o número de estúdios de desenvolvimento de jogos no Brasil aumentou de 133 para 1.009, evidenciando um crescimento de mais de sete vezes em menos de uma década. Esse avanço expressivo pode ser visualizado na Figura 2, que ilustra claramente a evolução do número de estúdios no período analisado.

Figura 2 – Evolução do número de estúdios de desenvolvimento de jogos no Brasil



Fonte: Abragames (2022)

Segundo os dados da Pesquisa Game Brasil, "três em cada quatro pessoas no Brasil usam celulares, videogames ou computadores para jogar"(CNN Brasil (2024)). Essa presença massiva reflete não apenas um crescimento no número de jogadores, mas também uma transformação no perfil do consumidor, que se mostra mais exigente em relação à qualidade gráfica, jogabilidade e performance dos títulos.

2.2 ANOMALIAS VISUAIS EM JOGOS DIGITAIS

O crescimento expressivo do público gamer no Brasil reforça a necessidade de experiências visuais cada vez mais refinadas e imersivas, o que evidencia a abrangência dos jogos eletrônicos como forma dominante de entretenimento. Conforme aponta Backus (2025), assim como qualquer outro software, os videogames estão sujeitos a erros que aparecem principalmente na forma de bugs e glitches. Embora distintos, esses termos são frequentemente confundidos pelos jogadores devido à complexidade envolvida.

Um bug é caracterizado como um erro interno do sistema que compromete o funcionamento adequado de certas mecânicas, como quando um jogador não consegue interagir com um objeto essencial. Já um glitch é uma falha que ocorre mesmo com os sistemas do jogo funcionando normalmente, afetando especialmente a aparência visual como por exemplo, quando objetos se sobrepõem ou atravessam cenários de maneira não natural, exatamente como descrito por Backus (2025).

2.3 MACHINE LEARNING

Machine Learning, ou Aprendizado de Máquina (AM), é um campo da inteligência artificial voltado ao desenvolvimento de algoritmos capazes de identificar padrões e tomar decisões com base em dados. Nessa perspectiva, segue a definição de Rossi (2015):

> O objetivo dos algoritmos de aprendizado de máquina é aprender, generalizar, ou ainda extrair padrões ou características das classes das coleções com base nos documentos textuais e rótulos (identificadores de classe) dos documentos informados por um usuário ou especialista de domínio (Rossi (2015, p. 2)).

Essa definição reforça a ideia de que, ao serem treinados com dados rotulados, os algoritmos conseguem construir modelos que reconhecem características relevantes e realizam previsões sobre novos dados.

Entre os diversos métodos utilizados no campo do Aprendizado de Máquina, um dos mais relevantes e amplamente adotados é o das Redes Neurais Artificiais (RNAs), que se destacam por sua capacidade de modelar relações complexas e não lineares entre os dados.

2.3.1 Redes neurais artificiais

As RNAs têm sua origem inspirada no funcionamento dos neurônios do cérebro humano. Elas são constituídas por um conjunto de neurônios artificiais que buscam simular a forma, o comportamento e as funções dos neurônios biológicos, o que permite às RNAs aprenderem e reconhecerem padrões de maneira semelhante ao sistema nervoso (Borges (2020)). Essa semelhança estrutural e funcional é fundamental para a aplicação das RNAs em diversas áreas, como reconhecimento de imagens, processamento de linguagem natural e previsão de séries temporais. Nesse contexto, destaca-se o seguinte trecho:

> As redes neurais artificiais (RNA) são definidas como processadores maciçamente paralelos que simulam o cérebro humano na intenção de coletar evidências empíricas e também à medida que preservam e permitem o uso do conhecimento experimental. Em seu funcionamento, existem sinapses (ou ligações inter-neurais) que são usadas para fins de aprendizagem e armazenamento do conhecimento. (Haykin, 1999, apud Santos et al., 2016)

Essa definição ressalta a capacidade das RNAs de aprender a partir de exemplos, armazenando e generalizando padrões complexos de dados. Para entender como isso acontece, é fundamental compreender os elementos básicos que compõem uma RNA.

Para Haykin (2001) uma RNA se assemelha ao cérebro humano em dois aspectos fundamentais: (a) o conhecimento é adquirido pela rede a partir de seu ambiente, por intermédio do processo de aprendizagem; e (b) as forças de conexão entre os neurônios, chamadas de

pesos sinápticos, são utilizadas para armazenar esse conhecimento adquirido. Esses pesos são ajustados ao longo do tempo durante o processo de treinamento da rede, permitindo que ela aprenda e se adapte a diferentes padrões de entrada.

Complementando essa estrutura, as redes neurais são compostas por uma determinada quantidade de entradas e unidades de processamento (ou neurônios), que são interligadas por meio de pesos sinápticos. As entradas são propagadas através da topologia da RNA, sendo transformadas tanto pelos pesos quanto pela Função de Ativação (do inglês *Activation Function*) (AF) dos neurônios (Haykin (2001)).

Essa organização permite que as RNAs tradicionais realizem tarefas de classificação, regressão e reconhecimento de padrões com eficiência. No entanto, ao lidar com dados de alta dimensionalidade e estruturas espaciais complexas, como imagens, vídeos e sinais visuais, essas redes podem apresentar limitações, como o alto número de parâmetros e a perda de informações espaciais.

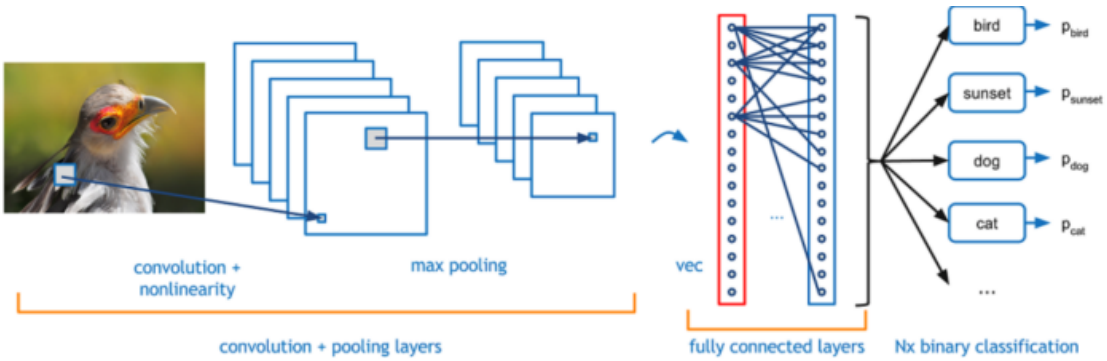
Para superar essas limitações, surgiram arquiteturas especializadas, como as CNNs. Diante da complexidade dos dados visuais presentes em ambientes gráficos de jogos digitais, o uso dessas redes torna-se essencial. Nesse contexto, as CNNs ganham destaque, sendo o foco principal deste trabalho por sua capacidade de identificar padrões visuais e detectar possíveis anomalias gráficas nesses ambientes.

2.3.2 Redes Neurais Convolucionais

Segundo Pereira (2025), as CNNs são amplamente utilizadas em tarefas de visão computacional devido à sua capacidade de extrair características hierárquicas espaciais por meio de operações de convolução. O autor destaca que essas redes se diferenciam das arquiteturas tradicionais por empregar filtros convolucionais que capturam padrões locais, permitindo o reconhecimento eficiente de objetos, formas e texturas em imagens.

A Figura 3 ilustra a estrutura de uma arquitetura típica de CNN, organizada em três estágios principais, conforme destacado visualmente: blocos convolucionais com não-linearidade, camadas de pooling, camadas totalmente conectadas e por fim uma camada de classificação binária. Cada componente é responsável por uma etapa distinta do processamento de imagens, conforme detalhado a seguir.

Figura 3 – Representação visual de uma arquitetura de CNN



Fonte: Adaptado de Cunha (2020)

1. **Blocos Convolucionais com Não-Linearidade**

   • Segundo Pereira (2025), cada camada convolucional aplica filtros que detectam padrões locais, seguidos por funções de ativação não lineares como Rectified Linear Unit (ReLU).

   • Como observado por Pereira (2025), camadas iniciais extraem características de baixo nível (bordas, cores), enquanto camadas mais profundas identificam padrões complexos.

2. **Camadas de Pooling**

   • Segundo Santos et al. (2017), a camada de pooling é responsável por reduzir a dimensionalidade dos mapas de características, diminuindo sua largura e altura. Nesse sentido, os autores afirmam que "A operação de pooling possibilita uma invariância espacial. O agrupamento de características, na maioria das arquiteturas de convolução, utiliza uma função de max pooling".

3. **Camadas Totalmente Conectadas**

   • Utsch (2018) descreve que uma camada totalmente conectada toma todos os neurônios da camada anterior e os conecta a cada neurônio da camada atual. Em seguida, o autor complementa: "As camadas totalmente conectadas que seguem as camadas convolucionais e de pooling têm por finalidade interpretar essas características de alto nível de abstração, efetuando funções do raciocínio complexo".

4. **Camada de Classificação Binária**

- Conforme Utsch (2018):

> Depois das camadas totalmente conectadas, vem a camada de saída. Para o problema de classificação, é comum usar tantos neurônios quanto existam classes a prever, e a saída de cada neurônio tem uma função de ativação de tipo softmax. A softmax é responsável por transformar as saídas dos neurônios em um formato de probabilidades. Isto é, todos os valores são não negativos, menores que um, e o somatório da saída de todos os softmax é igual a 1. Esse formato é necessário para a função de perda cross entropy (entropia cruzada), que é a mais utilizada para os casos de regressão logística, como o é a classificação de imagens (Utsch (2018, p. 23)).

- Embora o trecho citado de Utsch (2018) refira-se a problemas multiclasse (com softmax), em classificação binária, a abordagem padrão é utilizar um único neurônio com ativação sigmóide. Conforme explicado por Singh (2023), essa função é amplamente empregada para modelar saídas binárias (e.g., "sim"ou "não"), transformando a combinação linear das features em um valor de probabilidade entre 0 e 1. Essa saída probabilística é compatível com a função de perda binary cross-entropy, otimizando a distinção entre duas classes.

2.4 MÉTRICAS DE AVALIAÇÃO DE DESEMPENHO EM REDES NEURAIS

Após o treinamento de uma rede neural, torna-se fundamental avaliar seu desempenho por meio de métricas apropriadas, capazes de mensurar sua eficácia em relação aos objetivos da tarefa proposta. Diversas métricas são utilizadas para fornecer uma análise quantitativa e qualitativa dos resultados, conforme discutido por Rodrigues (2024), Haykin (2001) e Filho (2024).

2.4.1 Matriz de Confusão

De acordo com Filho (2024), para modelos de classificação, especialmente na abordagem binária, a matriz de confusão é uma das ferramentas mais utilizadas. Ela permite identificar quatro categorias principais que descrevem os acertos e erros do modelo durante a fase de teste:

- **Verdadeiro Positivo (VP)**: instâncias corretamente classificadas como positivas.
- **Falso Positivo (FP)**: instâncias negativas classificadas erroneamente como positivas (erro tipo I).
- **Verdadeiro Negativo (VN)**: instâncias corretamente classificadas como negativas.

• **Falso Negativo (FN)**: instâncias positivas classificadas erroneamente como negativas (erro tipo II).

Esses valores servem como base para o cálculo de métricas mais específicas, como acurácia, precisão, revogação, F1-score e AUC-ROC, que oferecem uma análise mais detalhada do desempenho do modelo.

2.4.2 Acurácia

A acurácia é uma métrica amplamente utilizada na avaliação de modelos de classificação, incluindo CNNs (HAYKIN, 2001). Ela expressa a proporção de predições corretas realizadas pelo modelo em relação ao total de amostras avaliadas.

$$\text{Acurácia} = \frac{VP + VN}{VP + FP + FN + VN} \tag{2.1}$$

Embora forneça uma visão geral do desempenho, a acurácia pode ser enganosa em conjuntos de dados desbalanceados, nos quais há uma disparidade significativa entre o número de amostras por classe (JURASZEK et al., 2014). Nesses casos, é recomendável utilizar métricas complementares.

2.4.3 Precisão

A precisão (ou *precision*) mede a proporção de verdadeiros positivos entre todas as predições positivas realizadas pelo modelo. É útil quando o custo de falsos positivos é alto.

$$\text{Precisão} = \frac{VP}{VP + FP} \tag{2.2}$$

2.4.4 Revogação (Recall)

A revogação, também chamada de sensibilidade, calcula a proporção de verdadeiros positivos em relação ao total de instâncias que realmente pertencem à classe positiva. Essa métrica é relevante quando o custo de falsos negativos é elevado.

$$\text{Recall} = \frac{VP}{VP + FN} \tag{2.3}$$

2.4.5 F1-Score

O F1-score é a média harmônica entre precisão e revogação. É especialmente útil em cenários onde há necessidade de equilibrar os impactos de falsos positivos e falsos negativos.

$$\text{F1-Score} = 2 \cdot \frac{\text{Precisão} \cdot \text{Recall}}{\text{Precisão} + \text{Recall}} \tag{2.4}$$

2.4.6 AUC-ROC

A métrica AUC-ROC (Área sob a Curva ROC) avalia a capacidade do modelo de distinguir entre classes positivas e negativas. Quanto maior a área sob a curva, melhor o desempenho do modelo em termos de discriminação (RODRIGUES, 2024).

2.5 TIPOS DE CLASSIFICAÇÃO EM REDES NEURAIS

2.5.1 Introdução

A classificação é uma das tarefas centrais em aprendizado supervisionado, sendo responsável por atribuir rótulos às entradas com base em padrões aprendidos durante o treinamento Castro e Braga (2011). Em redes neurais, essa tarefa é geralmente realizada na camada de saída, que define a estrutura final do modelo de acordo com o tipo de classificação desejado.

2.5.2 Classificação Binária

Neste tipo de problema, o modelo decide entre duas classes distintas, como "positivo"ou "negativo". A saída da rede geralmente consiste em um único neurônio com função de ativação sigmóide, que retorna um valor entre 0 e 1 representando a probabilidade de pertencimento a uma das classes. Esse tipo de estrutura é amplamente utilizado em tarefas como detecção de fraudes, diagnósticos médicos e classificação de sentimentos Rossi (2015).

A função sigmóide é especialmente adequada para esse tipo de tarefa por produzir uma saída contínua entre 0 e 1, o que facilita a interpretação probabilística do resultado Singh (2023).

2.5.3 Classificação Multiclasse

A classificação multiclasse envolve mais de duas categorias mutuamente exclusivas. A camada de saída da rede neural possui múltiplos neurônios, cada um representando uma classe, e utiliza a função de ativação softmax para normalizar as probabilidades. Esse tipo de

problema pode ser tratado como uma extensão da classificação binária, em que cada exemplo é atribuído exclusivamente a uma única classe Google Developers (2025). Como destacado pela Google Developers, "o processo pode ser repetido para cada uma das classes originais"Google Developers (2025).

Uma abordagem comum para problemas multiclasse é a decomposição em múltiplos classificadores binários. Por exemplo, em um cenário com três classes (A, B e C), pode-se construir um classificador binário que diferencie entre A+B e C, seguido por outro que distinga entre A e B. Essa técnica é conhecida como estratégia de um-contra-todos (one-vs-rest) ou um-contra-um (one-vs-one), dependendo da forma como os pares de classes são organizados Google Developers (2025).

Um exemplo clássico de aplicação é o reconhecimento de dígitos manuscritos, em que o modelo recebe uma imagem e deve identificar qual número de 0 a 9 está representado.

2.5.4 Classificação Multiclasse Um contra Um

Este tipo de classificação é utilizado principalmente para detecção de anomalias, sendo o modelo treinado exclusivamente com dados de uma classe considerada normal. O objetivo é identificar desvios ou padrões incomuns que não se encaixam na distribuição aprendida. Embora o foco desta classificação seja diferente da abordagem multiclasse, técnicas como o método um contra um também ilustram como modelos binários podem ser adaptados para cenários mais complexos. Segundo a documentação da Microsoft, "mesmo algoritmos de classificação binária podem ser adaptados para tarefas de classificação de várias classes por meio de diversas estratégias"Microsoft Learn (2024). Essa flexibilidade é essencial para lidar com problemas em que os dados de classes anômalas são escassos ou inexistentes.

2.5.5 Funções de Ativação e de Perda

A escolha adequada das funções de ativação e de perda é essencial para o desempenho e a capacidade de generalização dos modelos de aprendizado de máquina. As funções de ativação são responsáveis por introduzir não-linearidade nas redes neurais, permitindo que o modelo aprenda padrões complexos nos dados. Já as funções de perda orientam o processo de otimização, indicando o quão distante a predição está do valor esperado Ceccon (2020). Em tarefas de classificação binária, a função sigmoide é amplamente utilizada por sua capacidade de mapear valores reais para o intervalo $[0, 1]$, permitindo interpretar a saída como uma proba-

bilidade. Sua formulação matemática é dada por:

$$\sigma(x) = \frac{1}{1 + e^{-x}}, \tag{2.5}$$

sendo uma função não linear que possibilita ao modelo aprender relações complexas entre os dados. Conforme discutido por Gaio (2022), a função sigmoide apresenta comportamento assintótico, convergindo para 1 em entradas muito positivas e para 0 em entradas muito negativas, o que reforça sua aplicabilidade em cenários de decisão binária.

Por outro lado, em problemas de classificação multiclasse, a função *softmax* é mais apropriada, pois transforma o vetor de saídas em uma distribuição de probabilidade sobre as classes. Sua expressão é dada por:

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}, \tag{2.6}$$

onde K representa o número total de classes e z_i é o valor da saída para a classe i. A função *softmax* é geralmente combinada com a função de perda de entropia cruzada categórica, que penaliza fortemente previsões incorretas e acelera o processo de convergência durante o treinamento.

2.6 TESTE DE SOFTWARE EM JOGOS DIGITAIS

Os testes de software em jogos digitais apresentam desafios únicos em comparação com aplicações convencionais, devido à complexidade dos sistemas interativos, à imprevisibilidade do comportamento do jogador e à necessidade de garantir uma experiência fluida e imersiva. Segundo Costa (2024), a automação desses testes tem se mostrado uma alternativa promissora para lidar com essas dificuldades, especialmente em tarefas repetitivas como verificação de colisões, detecção de bugs gráficos e validação de regras de jogo. O estudo bibliográfico conduzido pelo autor destaca que, embora ainda haja limitações na cobertura de testes automatizados em ambientes altamente dinâmicos, ferramentas específicas para jogos têm evoluído significativamente, permitindo maior confiabilidade e eficiência no processo de desenvolvimento. Dessa forma, a adoção de estratégias automatizadas contribui não apenas para a melhoria da qualidade do produto final, mas também para a redução de custos e tempo de produção.

2.6.1 Classificação dos testes de software

Os testes de software podem ser classificados em diferentes categorias, conforme o objetivo e o escopo de cada abordagem. Essa organização é fundamental para garantir a qualidade do produto final, permitindo que diferentes aspectos do sistema sejam avaliados de forma sistemática. Segundo Costa (2024), os principais tipos de testes incluem:

- **Testes unitários**: verificam o comportamento de unidades individuais de código, como funções ou classes, assegurando que cada componente funcione corretamente de forma isolada.
- **Testes de integração**: avaliam a interação entre os componentes do sistema, garantindo que funcionem corretamente quando combinados.
- **Testes de sistema**: analisam o comportamento do sistema como um todo, verificando se ele atende aos requisitos definidos.
- **Testes de aceitação**: verificam se o sistema atende às expectativas e necessidades do usuário final.
- **Testes de desempenho**: medem aspectos como velocidade, escalabilidade e capacidade de resposta do sistema.
- **Testes de segurança**: identificam vulnerabilidades e ameaças, garantindo que o sistema esteja protegido contra acessos indevidos.
- **Testes de usabilidade**: avaliam a experiência do usuário, considerando fatores como facilidade de uso e interação com a interface.
- **Testes de compatibilidade**: verificam se o sistema funciona corretamente em diferentes plataformas, navegadores e dispositivos.

Essa classificação contribui para uma abordagem mais eficiente e estruturada no processo de teste, especialmente em contextos complexos como o desenvolvimento de jogos digitais.

2.6.2 Estratégias de teste

A definição de estratégias de teste é essencial para orientar como os testes serão conduzidos ao longo do desenvolvimento de um sistema. Elas estabelecem diretrizes que ajudam a selecionar as técnicas mais adequadas, os recursos necessários e o nível de profundidade das

análises. De acordo com Costa (2024), essas estratégias podem ser classificadas em diferentes abordagens, cada uma com objetivos específicos:

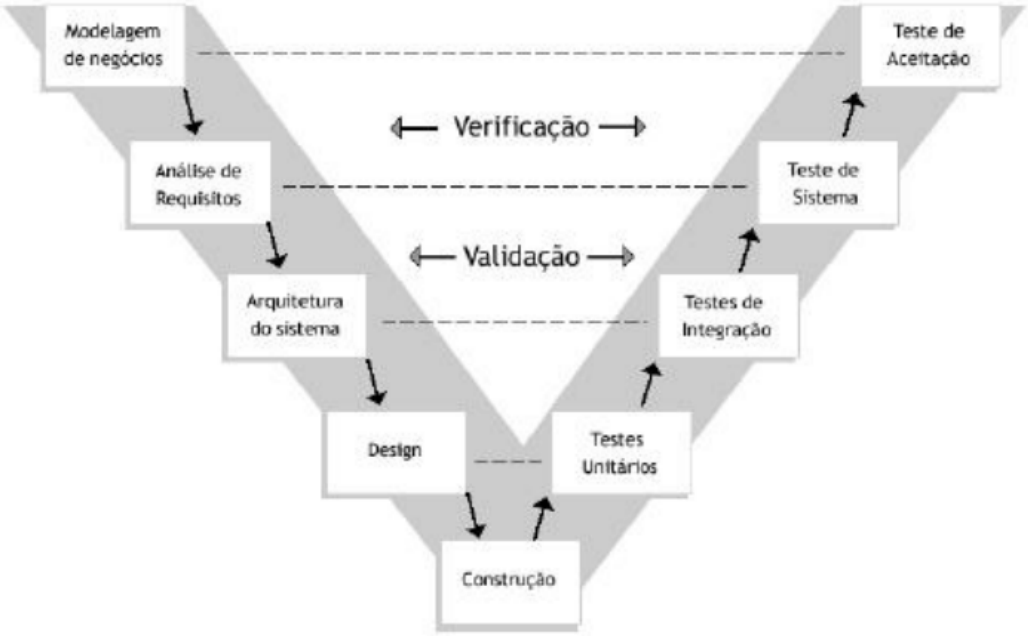
- **Testes caixa-preta**: analisam o comportamento do sistema sem considerar sua estrutura interna. Essa abordagem concentra-se nas entradas e saídas, sendo eficaz para verificar se os requisitos funcionais estão sendo corretamente atendidos.
- **Testes caixa-branca**: avaliam o funcionamento interno do sistema, incluindo a lógica de programação e a estrutura de dados. Essa estratégia permite examinar o fluxo de controle, a cobertura de código e a integridade dos componentes internos.
- **Testes caixa-cinza**: combinam elementos das abordagens caixa-preta e caixa-branca, possibilitando uma análise mais completa do sistema, tanto em relação ao seu comportamento externo quanto à sua estrutura interna.
- **Testes caixa-verde**: são voltados para requisitos específicos, como desempenho, segurança ou usabilidade. Essa estratégia é aplicada para validar características não funcionais que impactam diretamente na qualidade do produto final.

A escolha da estratégia mais adequada depende do contexto do projeto, dos objetivos dos testes e das particularidades do sistema em desenvolvimento. No caso de jogos digitais, por exemplo, estratégias como os testes caixa-verde e caixa-preta são especialmente relevantes para garantir uma experiência satisfatória ao usuário e o cumprimento dos requisitos de desempenho.

2.6.3 Fases do ciclo de vida de teste

O teste de software é composto por diversas fases que acompanham o ciclo de desenvolvimento, sendo planejadas desde os estágios iniciais do projeto. Segundo PEREIRA (2013), uma abordagem eficaz para estruturar essas fases é o modelo de desenvolvimento em "V", no qual as atividades de teste são integradas às etapas de desenvolvimento, permitindo maior controle de qualidade e identificação precoce de falhas. Esse modelo é ilustrado na Figura 4, que demonstra como cada fase de desenvolvimento possui uma etapa correspondente de teste, reforçando a importância da validação contínua ao longo do processo.

Figura 4 – Modelo de desenvolvimento em V.



Fonte: PEREIRA (2013)

A primeira fase é o **teste de unidade**, que tem como objetivo verificar se cada módulo ou componente do software atende às especificações definidas no projeto detalhado. Após essa etapa, os módulos são integrados conforme a arquitetura do sistema, iniciando o **teste de integração**, que busca identificar falhas na comunicação entre os componentes.

Com os subsistemas validados, realiza-se o **teste de sistema**, que avalia o comportamento do software como um todo, incluindo componentes de hardware e software, com o intuito de verificar se os requisitos funcionais estão sendo atendidos. Em seguida, são aplicados os **testes de aceitação**, geralmente conduzidos pelo cliente, com o objetivo de validar se o sistema satisfaz os requisitos definidos na fase de especificação.

Por fim, os **testes de regressão** são executados principalmente durante a fase de manutenção, com o propósito de garantir que modificações realizadas em partes do sistema não tenham introduzido erros em funcionalidades já existentes. Essas fases, quando bem estruturadas e aplicadas, contribuem significativamente para a confiabilidade e qualidade do produto final.

2.6.4 Testes no contexto dos jogos digitais

Embora o ciclo de vida de testes siga princípios semelhantes em diferentes tipos de software, os jogos digitais apresentam desafios específicos que exigem abordagens adaptadas. Em projetos desenvolvidos com motores como o Unity, o teste não deve ser encarado como uma etapa isolada, mas sim como um processo contínuo que permeia todas as fases do desenvolvimento Unity Technologies (2023).

Segundo Unity Technologies (2023), é essencial realizar testes desde os protótipos iniciais até as versões finais, incluindo cada atualização do jogo. Isso envolve não apenas testes funcionais, mas também avaliações de desempenho, experiência do usuário e compatibilidade entre plataformas. O objetivo é garantir que o jogo funcione de forma fluida, sem travamentos, artefatos visuais ou falhas de jogabilidade que possam comprometer a experiência do jogador.

Além dos testes tradicionais, os jogos digitais demandam:

- **Testes de jogabilidade**: Avaliam se as mecânicas do jogo são intuitivas, equilibradas e divertidas Unity Technologies (2023).
- **Testes de compatibilidade**: Verificam o funcionamento do jogo em diferentes dispositivos, sistemas operacionais e resoluções de tela Unity Technologies (2023).
- **Testes de desempenho**: Medem taxa de quadros (FPS), consumo de memória, aquecimento de dispositivos e tempo de carregamento Unity Technologies (2023).
- **Testes de localização**: Garantem que o conteúdo esteja corretamente traduzido e adaptado para diferentes regiões Unity Technologies (2023).
- **Testes com jogadores reais**: Conhecidos como *player testing*, são fundamentais para coletar feedback direto do público-alvo e ajustar o jogo conforme suas expectativas Unity Technologies (2023).

Essas práticas não apenas aumentam a qualidade técnica do produto, mas também fortalecem a reputação do estúdio e a fidelidade dos jogadores. Como destaca Unity Technologies (2023), “um único bug pode transformar empolgação em frustração”, tornando o teste uma etapa estratégica no desenvolvimento de jogos digitais.

3 TRABALHOS CORRELATOS

Nesta seção são apresentados estudos que se relacionam com o tema da detecção de anomalias e da aplicação de técnicas de inteligência artificial em jogos digitais. O objetivo é situar a presente pesquisa no contexto das investigações já existentes, destacando as principais contribuições identificadas na literatura e apontando lacunas que justificam a proposta deste trabalho.

3.1 TESTES DE SOFTWARE AUTOMATIZADOS EM JOGOS DIGITAIS

O estudo de Costa (2024) aborda de forma abrangente o tema da automação de testes em jogos digitais, enfatizando a importância de mecanismos capazes de avaliar a qualidade, estabilidade e desempenho dos softwares de entretenimento. O autor ressalta que, com o aumento da complexidade dos ambientes virtuais e a multiplicidade de elementos gráficos e interativos, métodos tradicionais de verificação manual tornam-se inviáveis ou ineficientes. Dessa forma, técnicas automatizadas emergem como uma alternativa promissora para reduzir custos e tempo de desenvolvimento, além de minimizar a ocorrência de falhas não detectadas.

O trabalho apresenta um panorama sobre ferramentas e metodologias de automação, discutindo desde abordagens baseadas em scripts até o uso de algoritmos heurísticos para teste de comportamento. No entanto, observa-se que o estudo concentra-se principalmente em aspectos conceituais e técnicos da automação de processos, sem avançar para o campo do aprendizado de máquina. Em particular, o uso de redes neurais profundas para a detecção de anomalias visuais, como artefatos gráficos, erros de textura ou problemas de renderização, não é explorado. Essa lacuna evidencia a necessidade de investigações que apliquem técnicas de visão computacional no contexto de testes automatizados em jogos, o que motiva diretamente a proposta desta pesquisa.

3.2 REDES NEURAIS ARTIFICIAIS EM JOGOS DIGITAIS

O trabalho de Nascimento e Filho (2021) propõe o uso de redes neurais artificiais (RNAs) em jogos digitais com foco na adaptação dinâmica da dificuldade, especialmente em jogos de estratégia. Os autores demonstram que, ao coletar dados em tempo real sobre o desempenho e o comportamento do jogador, é possível ajustar parâmetros do jogo de forma automática, promovendo uma experiência personalizada e equilibrada. Essa aplicação evidencia

o potencial das RNAs como ferramentas capazes de aprender e reagir a padrões complexos de interação.

Contudo, apesar de sua relevância para o campo da inteligência artificial aplicada a jogos, o estudo de Nascimento e Filho (2021) restringe-se à esfera da jogabilidade e do balanceamento dinâmico, sem abordar o uso de redes neurais voltadas à análise de aspectos visuais. Em particular, a pesquisa não contempla o emprego de redes neurais convolucionais (CNNs), que são amplamente reconhecidas por sua eficácia em tarefas de visão computacional, como classificação de imagens e detecção de anomalias visuais. Assim, o presente trabalho diferencia-se por explorar especificamente o potencial das CNNs na identificação automática de falhas gráficas em ambientes de jogos, contribuindo para a melhoria do controle de qualidade no processo de desenvolvimento.

3.3 SÍNTESE

Os trabalhos analisados demonstram que tanto a automação de testes quanto a aplicação de técnicas de inteligência artificial representam tendências consolidadas na área de desenvolvimento de jogos digitais. Entretanto, observa-se que ainda existe uma lacuna significativa no que diz respeito à aplicação direta de modelos de aprendizado profundo para a detecção de anomalias visuais.

Dessa forma, o presente TCC busca ampliar o escopo das pesquisas existentes ao propor uma abordagem baseada em redes neurais convolucionais (CNNs) voltada à análise automática de imagens de jogos. Ao integrar conceitos de visão computacional e aprendizado profundo ao contexto de testes de qualidade, este estudo pretende oferecer uma contribuição prática e inovadora, capaz de apoiar a indústria de jogos na identificação precoce de falhas gráficas e na otimização de seus processos de verificação e validação.

4 DESENVOLVIMENTO

Neste capítulo, detalhou-se o processo de desenvolvimento do sistema de detecção automática de anomalias visuais em jogos digitais utilizando Redes Neurais Convolucionais (CNNs). O desenvolvimento foi estruturado em várias etapas, desde a preparação dos dados até a implementação e avaliação do modelo.

4.1 METODOLOGIA APLICADA

O trabalho adotou uma abordagem quantitativa, aplicada e experimental, com o objetivo de investigar o uso de CNNs para a detecção automática de anomalias visuais em jogos digitais. A pesquisa fundamentou-se na análise sistemática de dados visuais para mensurar o desempenho do modelo na identificação de falhas gráficas, tais como objetos flutuantes, texturas corrompidas e glitches visuais.

A metodologia aplicada foi estruturada em cinco etapas principais, descritas a seguir.

4.1.1 Base de Dados

Foram utilizadas imagens de jogos digitais provenientes do dataset *Gameplay Images*, disponível no repositório Kaggle (Magotra (2023)). O conjunto continha aproximadamente **10 000 imagens** no formato `.png`, extraídas de *frames* de vídeos do YouTube e organizadas em **10 classes**, correspondentes a jogos populares: *Among Us*, *Apex Legends*, *Fortnite*, *Forza Horizon*, *Free Fire*, *Genshin Impact*, *God of War*, *Minecraft*, *Roblox* e *Terraria*. Cada classe continha cerca de 1 000 imagens com resolução padronizada de 640×360 pixels.

Para a criação de classes com anomalias, foram aplicadas artificialmente distorções visuais às imagens originais por meio de bibliotecas de processamento de imagem, como `imgaug` e `albumentations`, simulando falhas gráficas comuns. Entre as distorções aplicadas estavam: *blur* (desfoque), *noise* (ruído), *pixel dropout*, distorções geométricas e alterações abruptas de cor.

Dessa forma, o conjunto de dados foi composto por imagens “normais” e imagens “com anomalias”, garantindo controle experimental sobre os tipos de falhas presentes e permitindo a avaliação precisa do modelo.

4.1.2 Pré-processamento

Todas as imagens foram redimensionadas e normalizadas para garantir uniformidade no processamento, o que foi fundamental para a eficiência do treinamento da rede neural.

4.1.3 Anotação das Anomalias

As imagens tiveram seus rótulos atribuídos automaticamente, conforme o processo de aplicação das anomalias (original = normal; distorcida = com anomalias), configurando assim um conjunto rotulado para aprendizado supervisionado.

4.1.4 Aumento da Base de Dados (Data Augmentation)

Foram aplicadas técnicas como rotações, espelhamentos e variações de brilho para ampliar a variabilidade dos dados de entrada, fortalecendo a robustez do modelo e evitando *overfitting*.

4.1.5 Implementação da CNN

O modelo foi desenvolvido utilizando bibliotecas como TensorFlow e PyTorch, com foco na classificação binária. O treinamento foi realizado com os dados rotulados e o desempenho avaliado por meio de um conjunto de teste, utilizando métricas quantitativas como acurácia, precisão, recall e matriz de confusão. O tempo de inferência foi considerado como métrica adicional de eficiência.

4.2 ARQUITETURA DO MODELO

A arquitetura proposta para o sistema de detecção automática de anomalias visuais em jogos digitais foi organizada em duas etapas principais: (i) geração e preparação das imagens com anomalias e (ii) treinamento da Rede Neural Convolucional (CNN) responsável pela classificação binária (normal/anômala).

4.2.1 Geração de Anomalias e Pré-processamento

Antes do treinamento da rede, foi desenvolvido um módulo em `Python` para a criação automática de anomalias visuais sintéticas a partir de imagens originais de gameplay. Esse módulo foi implementado utilizando as bibliotecas `OpenCV`, `NumPy`, `Matplotlib` e `Albumentations`, além de transformações customizadas definidas manualmente.

Foram criadas seis funções de distorção visual com o objetivo de simular falhas gráficas reais em jogos digitais, incluindo texturas ausentes, ruídos visuais e artefatos de renderização. As transformações desenvolvidas foram:

- **Missing Texture**: substitui regiões da imagem por blocos coloridos ou padrões quadriculados, simulando falhas no carregamento de texturas;
- **Screen Tearing**: desloca faixas horizontais da imagem, reproduzindo o efeito de rasgo na tela comum em renderizações instáveis;
- **UI Occlusion**: insere retângulos coloridos aleatórios, representando elementos de interface sobrepostos de forma incorreta;
- **Pixelation**: reduz a resolução local de determinadas áreas, criando efeito de pixelização forçada;
- **Polygon Clipping**: aplica máscaras poligonais aleatórias, gerando cortes e áreas com preenchimento incorreto;
- **Shader Artifacts**: altera a luminosidade por faixas verticais, simulando falhas em shaders e iluminação.

Essas transformações foram agrupadas por meio da classe `A.Compose()` da biblioteca `Albumentations`, permitindo a aplicação sequencial de múltiplos efeitos sobre cada imagem de entrada. O pipeline foi configurado para gerar três variações anômalas por imagem original, ampliando o conjunto de treinamento e garantindo maior diversidade visual.

Todas as imagens foram redimensionadas para 640×360 pixels e normalizadas, garantindo consistência durante o processamento. O sistema foi projetado para exibir visualmente amostras de imagens originais e distorcidas, possibilitando a verificação qualitativa dos artefatos gerados.

4.2.2 Arquitetura da CNN

A rede neural convolucional foi implementada utilizando as bibliotecas `TensorFlow` e `PyTorch`, com foco na classificação binária entre imagens normais e anômalas. A arquitetura foi composta por múltiplas camadas convolucionais intercaladas com funções de ativação `ReLU`, camadas de `MaxPooling` e blocos de normalização por lotes (`Batch Normalization`). Ao final, foram adicionadas camadas densas (`Fully Connected`) e uma camada de saída com ativação `Sigmoid`, adequada para problemas binários.

O treinamento utilizou a função de perda `Binary Cross-Entropy` e o otimizador `Adam`, com taxa de aprendizado ajustada empiricamente. O processo foi monitorado por meio das métricas de acurácia, precisão, *recall* e análise da matriz de confusão, além da medição do tempo médio de inferência, que serviu como indicador de eficiência computacional.

4.2.3 Experimentos Realizados

Após a definição da arquitetura e do processo de treinamento, foram conduzidos três experimentos principais, cada um com objetivos específicos. Esses experimentos tiveram como propósito avaliar diferentes aspectos do modelo, desde a escolha da arquitetura até estratégias de otimização e generalização. A seguir, descrevem-se os experimentos realizados:

- **Experimento 1 – Comparação de Arquiteturas CNN**: comparação do desempenho de diferentes arquiteturas de redes convolucionais (VGG16, VGG19, ResNet50, DenseNet121, EfficientNetV2-M e AlexNet) aplicadas ao problema de detecção de anomalias visuais.
- **Experimento 2 – Ajuste de Hiperparâmetros**: investigação do impacto de diferentes combinações de *learning rate* e *batch size* nos modelos mais promissores identificados no Experimento 1.
- **Experimento 3 – Estratégias de Generalização e Fine-Tuning**: aplicação de técnicas de aumento de dados, regularização e *fine-tuning* para otimizar o modelo final e reduzir problemas de sobreajuste.

Cada experimento é detalhado nas subseções seguintes, incluindo seus objetivos, configurações, procedimentos adotados e resultados esperados.

4.3 EXPERIMENTO 1 – COMPARAÇÃO DE ARQUITETURAS CNN

O primeiro experimento teve como objetivo comparar o desempenho de diferentes arquiteturas de Redes Neurais Convolucionais (CNNs) na tarefa de detecção automática de anomalias visuais em jogos digitais. Foram avaliados seis modelos amplamente utilizados em visão computacional: **VGG16**, **VGG19**, **ResNet50**, **DenseNet121**, **EfficientNetV2M** e **AlexNet**.

4.3.1 Configuração Experimental

Cada modelo foi ajustado para classificação binária (*normal* e *anômala*) com base em imagens do conjunto de dados preparado na Seção 4. O treinamento foi realizado com as seguintes configurações:

- **Otimizador**: Adam;
- **Função de perda**: Binary Cross-Entropy;
- **Taxa de aprendizado**: 0,001;
- **Batch size**: 32;
- **Número de épocas**: 30;
- **Proporção de divisão**: 80% treino e 20% validação;
- **Data augmentation**: rotação de ±15°, brilho entre 0,8 e 1,2, e espelhamento horizontal.

O processo foi implementado utilizando a biblioteca `TensorFlow/Keras` e executado em ambiente GPU. Para cada arquitetura testada, foram gerados automaticamente os seguintes gráficos: evolução da acurácia e perda, matriz de confusão, métricas por classe (precisão, revogação e F1-score) e curva ROC.

4.3.2 Resultados Obtidos

A seguir, são apresentados os gráficos que ilustram o desempenho de cada modelo durante o treinamento e validação.

VGG16

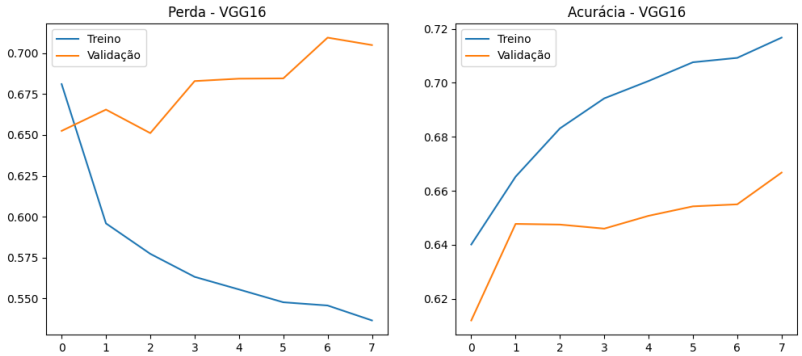


Figura 5 – Evolução da acurácia e perda da VGG16 durante o treinamento.

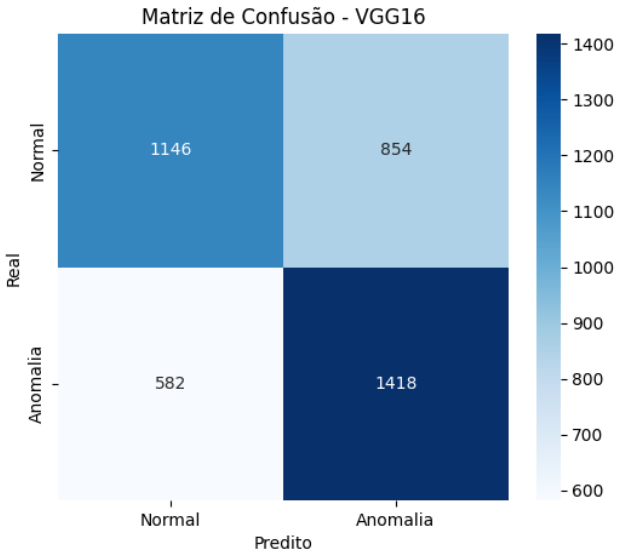


Figura 6 – Matriz de confusão da VGG16.

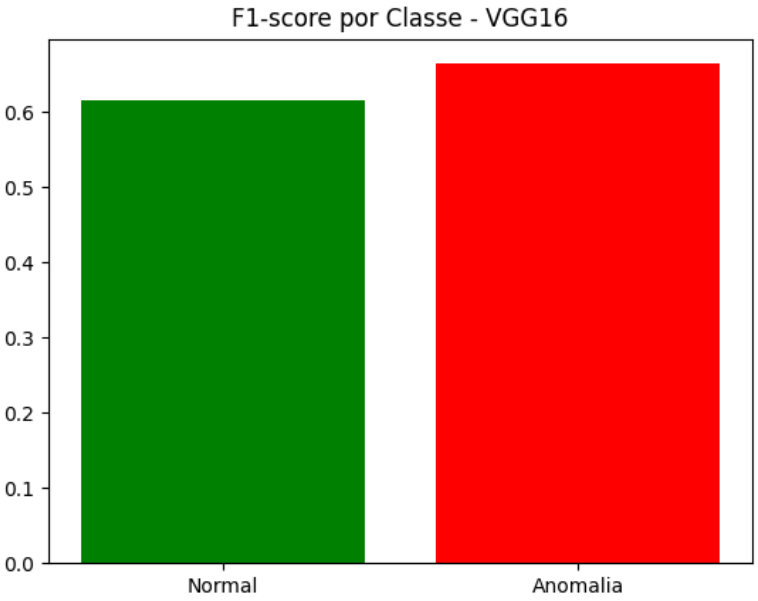


Figura 7 – F1-score por classe da VGG16.

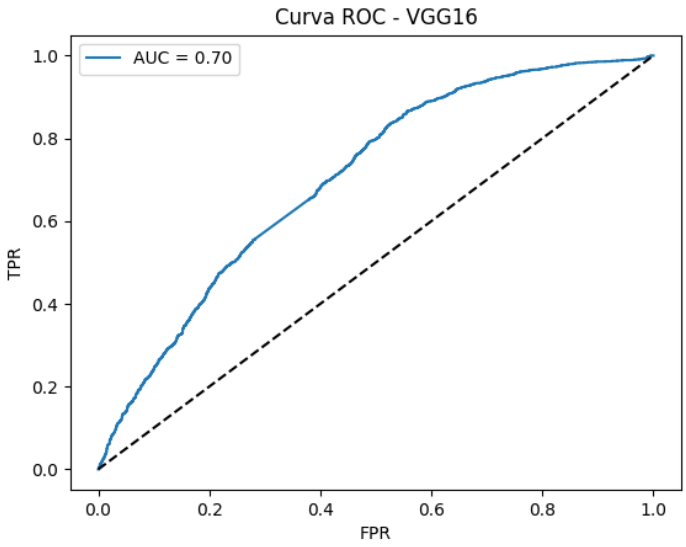


Figura 8 – Curva ROC da VGG16.

VGG19

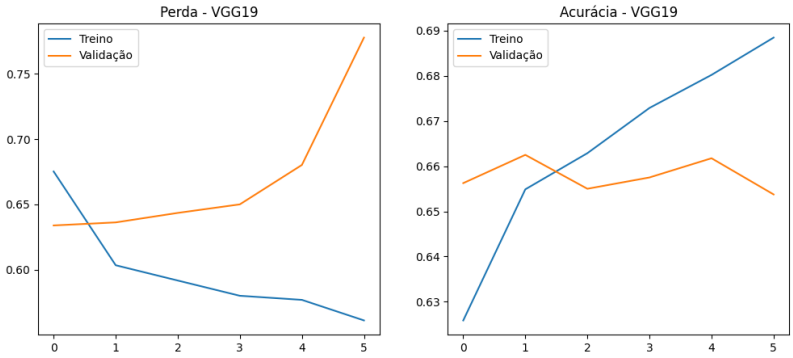


Figura 9 – Evolução da acurácia e perda da VGG19 durante o treinamento.

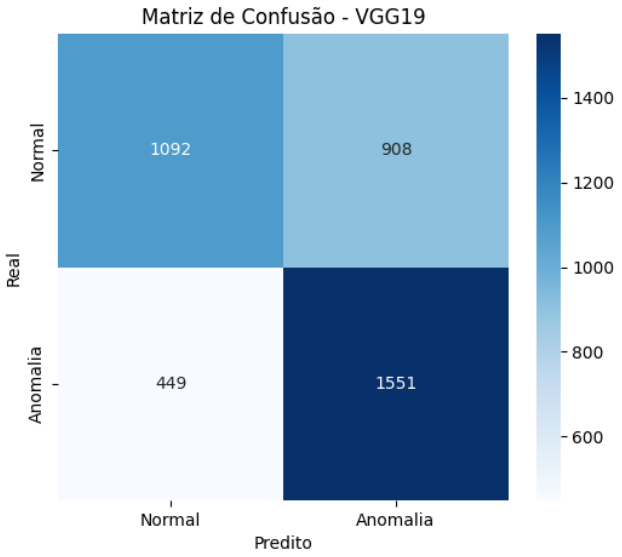


Figura 10 – Matriz de confusão da VGG19.

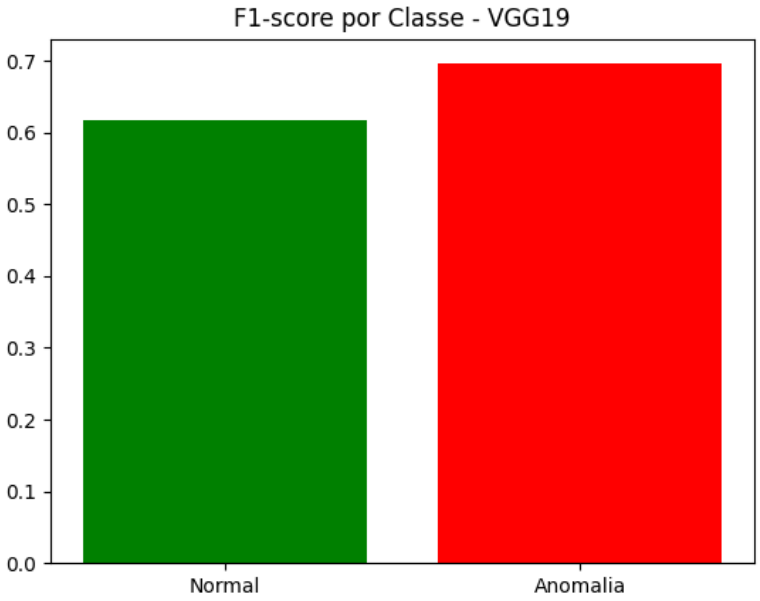


Figura 11 – F1-score por classe da VGG19.

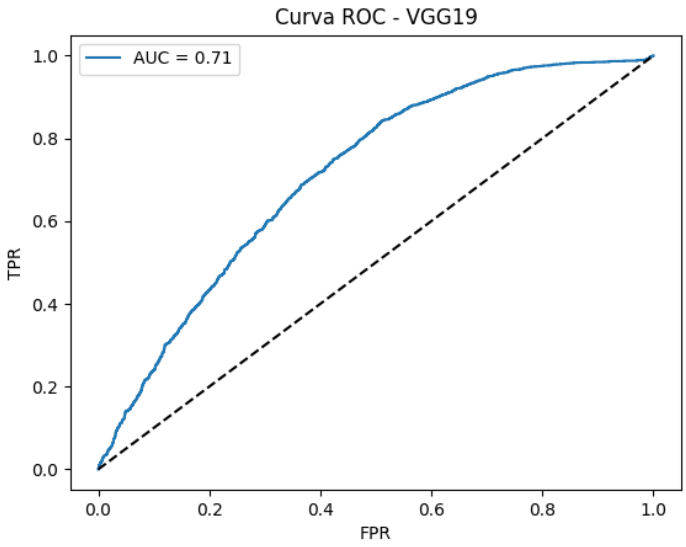


Figura 12 – Curva ROC da VGG19.

DenseNet121

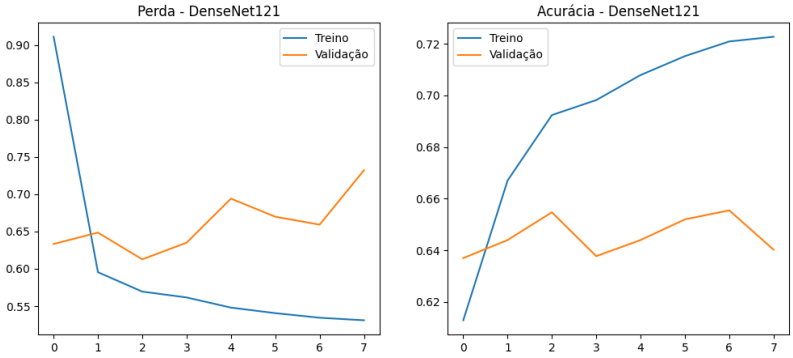


Figura 13 – Evolução da acurácia e perda da DenseNet121 durante o treinamento.

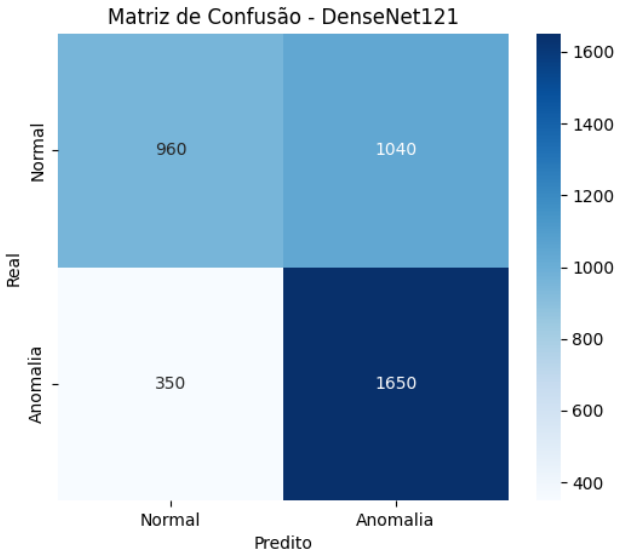


Figura 14 – Matriz de confusão da DenseNet121.

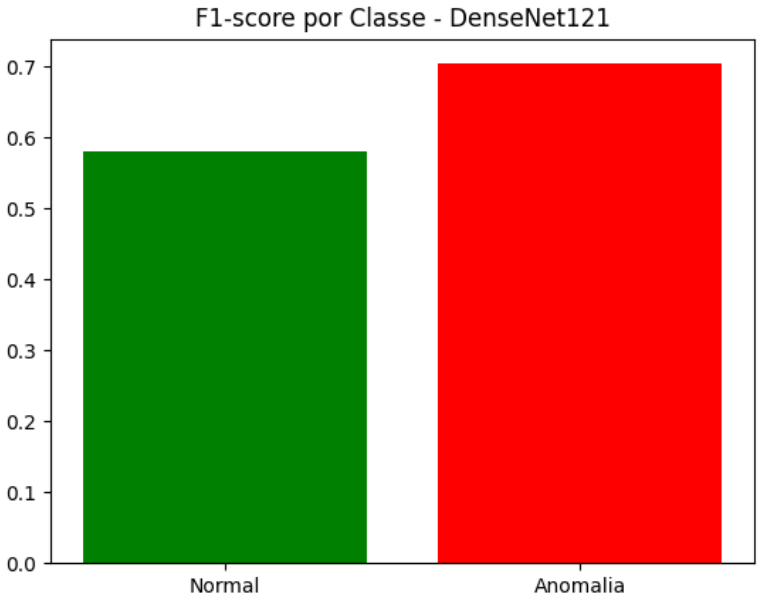


Figura 15 – F1-score por classe da DenseNet121.

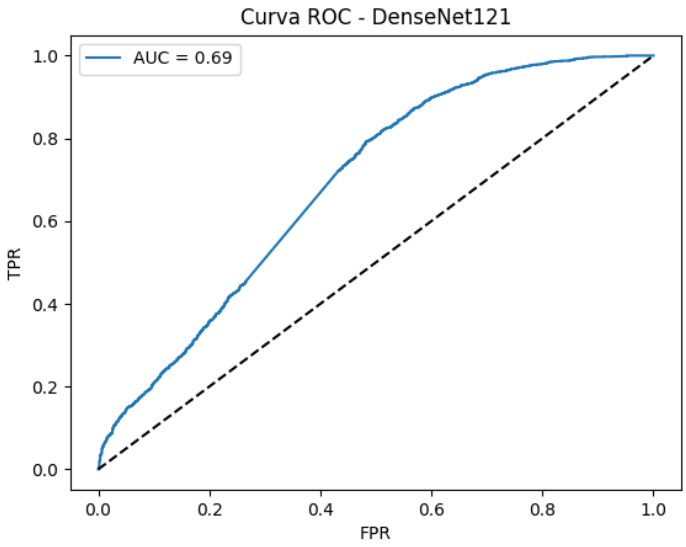


Figura 16 – Curva ROC da DenseNet121.

AlexNet, ResNet50 e EfficientNetV2M

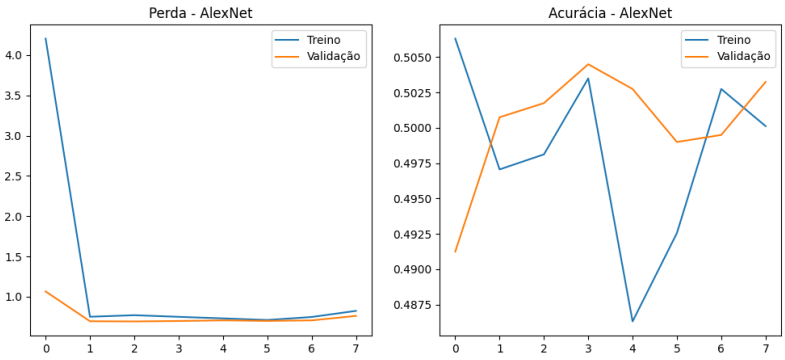


Figura 17 – Evolução da acurácia e perda da AlexNet.

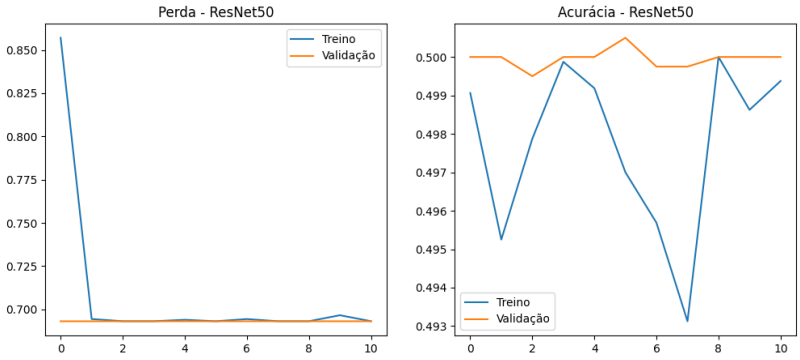


Figura 18 – Evolução da acurácia e perda da ResNet50.

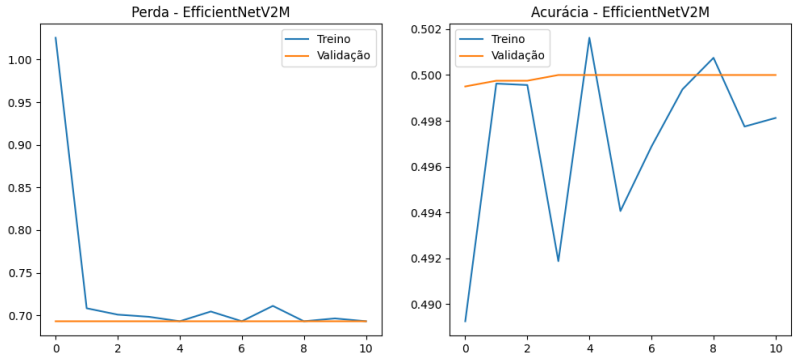


Figura 19 – Evolução da acurácia e perda da EfficientNetV2M.

4.3.3 Desempenho Quantitativo

A Tabela 2 resume os principais resultados de validação obtidos por cada arquitetura.

Tabela 2 – Resultados de validação dos modelos testados no Experimento 1.

Modelo	**Acurácia**	**Precisão**	**Recall**	**F1-score**
VGG19	0,6607	0,6697	0,6607	**0,6562**
DenseNet121	0,6525	**0,6731**	0,6525	0,6418
VGG16	0,6410	0,6437	0,6410	0,6393
AlexNet	0,5020	0,5914	0,5020	0,3408
ResNet50	0,5000	0,2500	0,5000	0,3333
EfficientNetV2M	0,5000	0,2500	0,5000	0,3333

4.3.4 Resumo Técnico

Os modelos **VGG19**, **DenseNet121** e **VGG16** apresentaram os melhores desempenhos gerais, com acurácia entre 64% e 66% e F1-score acima de 0,63. O modelo **VGG19** destacou-se por apresentar maior equilíbrio entre precisão e revogação. Por outro lado, **ResNet50** e **EfficientNetV2M** não demonstraram aprendizado efetivo, mantendo acurácia próxima a 50%, o que indica possível falha na adaptação ao problema binário. O modelo **AlexNet**, embora simples, teve desempenho intermediário.

A análise detalhada desses resultados será discutida no Capítulo 5.

5 RESULTADOS E DISCUSSÃO

Resultados...A

6 CONCLUSÃO

Este trabalho apresentou uma abordagem para a detecção automática de anomalias visuais em jogos digitais utilizando Redes Neurais Convolucionais (CNNs). A metodologia adotada envolveu a criação de um conjunto de dados composto por imagens de jogos populares, com e sem distorções artificiais, permitindo o treinamento e avaliação do modelo proposto.

REFERÊNCIAS

Abragames. **Pesquisa da Indústria Brasileira de Games**. 2022. Acesso em: 02 jun. 2025. Disponível em: <https://www.abragames.org/pesquisa-da-industria-brasileira-de-games.html>.

ARAÚJO, Markondes Lacerda; LEÃO, Marcelo Franco. Produção científica nacional sobre jogos digitais no ensino de ciências (2004-2021). **Educação**, p. e50–1, 2024.

BACKUS, Jessica. Investigating the prominence and severity of bugs and glitches within games and their effects on player experience. **arXiv preprint arXiv :2504.19010**, 2025.

BERNARDO, Claudio Gonçalves et al. Classificação de jogos eletrônicos como tecnologia assistiva para pessoas com deficiência visual. **Informação & Informação**, v. 25, n. 1, p. 141–170, 2020.

BORGES, Pedro Pinguelli. **Aplicação de redes neurais para a determinação do coeficiente de atrito de filmes plásticos flexíveis**. Dissertação (B.S. thesis) — Universidade Tecnológica Federal do Paraná, 2020.

CASTRO, Cristiano Leite de; BRAGA, Antônio Pádua. Aprendizado supervisionado com conjuntos de dados desbalanceados. **Sba: Controle & Automação Sociedade Brasileira de Automatica**, SciELO Brasil, v. 22, p. 441–466, 2011.

CECCON, Denny. **Funções de ativação: definição, características, e quando usar cada uma**. 2020. <https://iaexpert.academy/2020/05/25/funcoes-de-ativacao-definicao-caracteristicas-e-quando-usar-cada-uma/>. Acesso em: 22 de julho 2025.

CNN Brasil. **Público gamer cresce e 3 em cada 4 brasileiros consomem jogos eletrônicos**. 2024. Acesso em: 25 jun. 2025. Disponível em: <https://www.cnnbrasil.com.br/tecnologia/publico-gamer-cresce-e-3-em-cada-4-brasileiros-consomem-jogos-eletronicos/>.

COSTA, Moisés Oliveira. Testes de software automatizados em jogos digitais: um estudo bibliográfico. 2024.

CUNHA, Leonardo Cardoso da. **Redes neurais convolucionais e segmentação de imagens: uma revisão bibliográfica**. Dissertação (Mestrado) — Universidade Federal de Ouro Preto, 2020. Monografia (Graduação em Engenharia de Controle e Automação). Disponível em: <http://www.monografias.ufop.br/handle/35400000/2872>.

FILHO, Paulo Cleber Farias da Silva. Light curve imaging for exoplanet detection with deep learning: a conceptual trial. 2024.

GAIO, Daniel Eliel. Análise comparativa das técnicas de implementação de arquiteturas da função sigmoide. Universidade Federal do Pampa, 2022.

GameBlast. **Problemas em games: bugs, glitches e falhas**. 2015. Acesso em: 25 jun. 2025. Disponível em: <https://www.gameblast.com.br/2015/10/problemas-games-bugs-glitches-falhas.html>.

Google Developers. **Classificação: classificação multiclasse | Machine Learning**. 2025. Acesso em: 25 jun. 2025. Disponível em: <https://developers.google.com/machine-learning/crash-course/classification/multiclass?hl=pt-br>.

HAYKIN, Simon. **Redes neurais: princípios e prática**. [S.l.]: Bookman Editora, 2001.

IGN Brasil. **MissingNo: o pokémon que se tornou o glitch mais famoso da franquia**. 2020. Acesso em: 23 jun. 2025. Disponível em: <https://br.ign.com/pokemon-red-version/109919/news/missingno-o-pokemon-que-se-tornou-o-glitch-mais-famoso-da-franquia>.

JURASZEK, Guilherme Defreitas et al. Reconhecimento de produtos por imagem utilizando palavras visuais e redes neurais convolucionais. Universidade do Estado de Santa Catarina, 2014.

MAGOTRA, Adit. **Gameplay Images Dataset**. 2023. Acesso em: 22 jun. 2025. Disponível em: <https://www.kaggle.com/datasets/aditmagotra/gameplay-images>.

Microsoft Learn. **Multiclasse Um contra Um - Azure Machine Learning**. 2024. Acesso em: 14 ago. 2025. Disponível em: <https://learn.microsoft.com/pt-br/azure/machine-learning/component-reference/one-vs-one-multiclass?view=azureml-api-2>.

NASCIMENTO, Jônatas C.; FILHO, Joaquim Pessoa. Redes neurais artificiais em jogos digitais. **Revista de Pesquisa em Jogos Digitais**, Universidade Presbiteriana Mackenzie, 2021. Disponível em: Universidade Presbiteriana Mackenzie.

Nintendo. **Game Boy | Hardware | Nintendo History**. 2025. Acesso em: 23 jun. 2025. Disponível em: <https://www.nintendo.com/pt-pt/Consolas-e-acessorios/Historia-da-Nintendo/Game-Boy/Game-Boy-627031.html>.

PEREIRA, Adriano Diniz. **Redes neurais convolucionais para análise de mamografias**. Tese (Doutorado) — Universidade Tecnológica Federal do Paraná, Ponta Grossa, PR, 2025. Dissertação (Mestrado em Engenharia de Produção). Disponível em: <http://repositorio.utfpr.edu.br/jspui/handle/1/36848>.

PEREIRA, Sílvia Arantes. Benefícios do teste de software. 004, 2013.

Pinglestudio. **What are glitches in gaming?** 2025. Acesso em: 25 jun. 2025. Disponível em: <https://pinglestudio.com/knowledge-base/what-are-glitches-in-gaming>.

RODRIGUES, Miguel Elias Silva. Avaliação de desempenho de redes neurais convolucionais no diagnóstico de amiloidose cardíaca. Universidade Federal da Paraíba, 2024.

ROSSI, Rafael Geraldeli. **Classificação automática de textos por meio de aprendizado de máquina baseado em redes**. Tese (Tese de Doutorado) — Universidade de São Paulo (USP), São Carlos, SP, 2015. Acesso aberto. Instituição de defesa: Biblioteca Digital de Teses e Dissertações da USP. Disponível em: <http://www.teses.usp.br/teses/disponiveis/55/55134/tde-05042016-105648/>.

SANTOS, Alan et al. Uma abordagem de classificação de imagens dermatoscópicas utilizando aprendizado profundo com redes neurais convolucionais. In: SBC. **Simpósio Brasileiro de Computação Aplicada à Saúde (SBCAS)**. [S.l.], 2017.

SANTOS, Murilo Alves et al. Aplicação de redes neurais no brasil: um estudo bibliométrico. **Biblionline**, v. 12, n. 2, p. 101–116, 2016.

SAVI, Rafael; ULBRICHT, Vania Ribas. Jogos digitais educacionais: benefícios e desafios. **Revista Novas Tecnologias na Educação**, v. 6, n. 1, 2008.

SINGH, Vikash. **Função Sigmóide: O que é e como implementá-la?** 2023. Acesso em: 22 jun. 2025. Disponível em: <https://www.datacamp.com/pt/tutorial/sigmoid-function>.

STEPHENS, Conor David. **Balancing multiplayer games design using deep learning techniques**. Tese (Doutorado) — University of Limerick, 2024.

Unity Technologies. **Dicas de teste e garantia de qualidade para projetos Unity**. 2023. <https://unity.com/pt/how-to/testing-and-quality-assurance-tips-unity-projects>. Acesso em: 21 jul. 2025.

UTSCH, Kaio Giurizatto. Uso de redes neurais convolucionais para classificação de imagens digitais de lesões de pele. **Trabalho de Conclusão de Curso (Graduação em Engenharia Elétrica)-Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo. Espírito Santo**, 2018.