Detecção de Falhas do Tipo Spaghetti em Impressoras 3D Utilizando Visão Computacional com Redes Neurais Convolucionais

Igor H. Buzatto, Igor Yepes

[1]Instituto Federal Farroupilha – Campus Frederico Westphalen (IFFAR)
Frederico Westphalen – RS – Brazil

igor.2022002631@aluno.iffar.edu.br, igor.yepes@iffarroupilha.edu.br

Abstract. *This work presents the development of an intelligent system for the automatic detection of spaghetti-type failures in FDM 3D printers. The solution uses convolutional neural networks (CNNs) applied to the analysis of images captured in real time, integrated with OctoPrint to pause the printing process and send email notifications to the user. The model was trained using a proprietary dataset and achieved high accuracy in classifying normal and defective prints. The results demonstrate that the system reduces material waste and increases the safety of the 3D printing process.*

Resumo. *Este trabalho apresenta o desenvolvimento de um sistema inteligente para detecção automática de falhas do tipo spaghetti em impressoras 3D FDM. A solução utiliza redes neurais convolucionais (CNNs) aplicadas à análise de imagens capturadas em tempo real, integradas ao OctoPrint para pausar a impressão e enviar notificações por e-mail ao usuário. O modelo foi treinado com dataset próprio e alcançou alta acurácia na classificação entre impressões normais e defeituosas. Os resultados demonstram que o sistema reduz desperdícios e aumenta a segurança do processo de impressão 3D.*

1. Introdução

A manufatura aditiva, popularmente conhecida como impressão 3D, tem revolucionado a forma como objetos são projetados e fabricados em diferentes setores, desde a indústria até o uso doméstico. Com sua capacidade de produzir peças complexas e personalizadas a partir de modelos digitais, essa tecnologia tem se tornado cada vez mais acessível e difundida.

No entanto, apesar de seus avanços, a impressão 3D ainda está sujeita a falhas durante o processo, especialmente em impressoras do tipo FDM (Fused Deposition Modeling). Um dos problemas mais comuns nesse contexto é o fenômeno conhecido como spaghetti, que ocorre quando a peça se desprende da mesa de impressão e o filamento é extrudado no ar, formando um emaranhado desordenado de material. Essas falhas resultam em desperdício de tempo, material e podem até danificar o equipamento.

Diversas soluções têm sido propostas para mitigar esses problemas, mas muitas delas exigem intervenção constante do operador ou envolvem sensores físicos adicionais. Neste cenário, técnicas de visão computacional e inteligência artificial surgem como

alternativas promissoras para o monitoramento automático e em tempo real do processo de impressão.

Este trabalho tem como objetivo desenvolver um sistema inteligente capaz de detectar falhas do tipo spaghetti em impressoras 3D, utilizando redes neurais convolucionais (CNNs) aplicadas à análise de imagens. O sistema será integrado ao OctoPrint, plataforma amplamente utilizada para gerenciamento remoto de impressoras 3D, permitindo não apenas a detecção da falha, mas também a pausa automática do processo e o envio de notificações ao usuário.

2. Referencial Teórico

2.1 Impressão 3D: Fundamentos e Desafios

A impressão 3D, também conhecida como manufatura aditiva, é um conjunto de tecnologias capaz de transformar modelos digitais tridimensionais em objetos físicos por meio da deposição de camadas sucessivas de material. Diferente dos métodos tradicionais de fabricação, como usinagem ou moldagem por injeção, a manufatura aditiva permite maior liberdade geométrica e personalização de peças (GIBSON et al., 2010).

O processo de impressão 3D geralmente segue oito etapas: modelagem em software CAD, conversão para formato STL, preparação do arquivo, configuração da máquina, construção camada por camada, remoção da peça, pós-processamento e aplicação (GIBSON et al., 2010, cap. 1). Isso torna o processo altamente flexível e acessível, o que explica sua adoção crescente tanto em ambientes industriais quanto domésticos.

Entre os diversos métodos de impressão, o mais comum em impressoras de baixo custo é o Fused Deposition Modeling (FDM). Nesse processo, um filamento termoplástico (como PLA ou ABS) é aquecido e extrudado por um bico aquecido, sendo depositado camada por camada sobre uma base aquecida (GIBSON et al., 2010, cap. 6).

Segundo Chua e Leong (2014), o método FDM é especialmente propenso a falhas mecânicas e térmicas, devido à sensibilidade do processo ao ambiente físico e às configurações da impressora.

No entanto, esse tipo de impressão está sujeito a diversas falhas, especialmente quando não há monitoramento adequado. Alguns dos problemas mais recorrentes incluem:

- Warping: descolamento das bordas da peça da base de impressão;
- Under-extrusion e over-extrusion: falhas no controle da quantidade de filamento;
- Layer shifting: deslocamento de camadas;
- Spaghetti: quando a peça se solta da mesa ou falha parcialmente, e o bico começa a extrudar no ar, criando um emaranhado de filamento.

Essas falhas, além de comprometerem o resultado final da impressão, causam desperdício de tempo e material. Cunha (2024), ao relatar experiências com impressoras como a Ender 3 V2, destaca que erros como o spaghetti são frequentes e difíceis de prever apenas com o firmware ou sensores básicos do equipamento.

Diante disso, torna-se evidente o interesse em soluções automatizadas que utilizem visão computacional e inteligência artificial para detectar falhas em tempo real,

possibilitando ações corretivas ou notificações imediatas ao operador, com o objetivo de reduzir prejuízos e aumentar a confiabilidade do processo.

2.2 Visão Computacional Aplicada à Detecção de Falhas

A visão computacional é a área da inteligência artificial que capacita máquinas a interpretar e compreender imagens e vídeos, simulando o sistema visual humano. Conforme Szeliski (2011), a disciplina integra o processamento de imagens ao reconhecimento de padrões complexos, viabilizando soluções em diagnóstico médico, veículos autônomos e automação industrial.

No contexto de inspeção de qualidade, a visão computacional já se mostrou eficaz em sistemas automatizados para detectar falhas em linhas de produção, como rachaduras, deformações ou padrões visuais fora do esperado. Em aplicações médicas, algoritmos visuais auxiliam no diagnóstico precoce de doenças a partir de exames de imagem, como radiografias e ressonâncias (MA et al., 2023). Essas aplicações compartilham uma característica comum: a detecção de anomalias visuais que podem comprometer a integridade ou funcionalidade de algo — exatamente o caso das impressões 3D com falhas do tipo spaghetti.

A falha spaghetti ocorre quando o objeto impresso se descola da mesa de impressão ou sofre interrupções durante a extrusão, fazendo com que o filamento se acumule desordenadamente. Por se tratar de um padrão visual distinto, com emaranhados soltos e imprevisíveis, a visão computacional se torna uma solução apropriada para sua detecção automática.

Técnicas clássicas de visão computacional que podem ser aplicadas para esse tipo de problema incluem:

- Pré-processamento de imagem com filtros de suavização e realce (Cap. 3 – Szeliski, 2011);
- Detecção de bordas para identificar regiões com estrutura irregular (ex.: algoritmo de Canny);
- Segmentação de regiões por agrupamento de pixels com características similares (Cap. 5 – Szeliski, 2011);
- Detecção e extração de características (ex.: pontos SIFT, SURF ou Harris), úteis para comparar padrões normais e defeituosos (Cap. 4 – Szeliski, 2011).

Além das técnicas tradicionais, a integração com métodos modernos de aprendizado de máquina permite que sistemas de visão computacional sejam treinados com exemplos de impressões corretas e com falhas, aprendendo a diferenciá-las com precisão crescente (EBERMAM; KROHLING, 2018).

Diante disso, justifica-se o uso da visão computacional no desenvolvimento de sistemas inteligentes para monitoramento de impressoras 3D, especialmente com o foco em identificar visualmente falhas críticas como o spaghetti, que são perceptíveis por câmera, mas difíceis de antecipar apenas via sensores mecânicos.

2.3 Inteligência Artificial e Aprendizado de Máquina

A inteligência artificial (IA) é um campo da ciência da computação dedicado ao desenvolvimento de sistemas capazes de realizar tarefas que normalmente exigiriam inteligência humana, como percepção visual, tomada de decisão e aprendizagem.

Segundo Russell e Norvig (2013), a IA pode ser compreendida sob diferentes abordagens: sistemas que pensam como humanos, que agem como humanos, que pensam racionalmente ou que agem racionalmente.

Dentro da IA, destaca-se o aprendizado de máquina (machine learning), que tem como objetivo permitir que algoritmos melhorem seu desempenho em determinada tarefa a partir de experiências anteriores, geralmente representadas como dados. Em vez de programar regras explícitas, fornece-se ao sistema exemplos para que ele reconheça padrões por conta própria (GOODFELLOW et al., 2016).

Os dois principais paradigmas do aprendizado de máquina são:

- Supervisionado: o modelo é treinado com pares entrada-saída já conhecidos, aprendendo a prever saídas para novos dados com base nesse histórico. Esse é o método adotado neste trabalho;
- Não supervisionado: não há rótulos nos dados, e o sistema busca descobrir estruturas ou padrões ocultos por conta própria (RUSSELL; NORVIG, 2013, cap. 18).

Com o avanço das redes neurais profundas, tornou-se possível lidar com dados mais complexos, como imagens e vídeos. Em especial, a área de visão computacional se beneficiou imensamente, passando de sistemas com baixa precisão para soluções capazes de superar até mesmo o desempenho humano em algumas tarefas específicas (GOODFELLOW et al., 2016, cap. 9).

A combinação de IA com sensores, câmeras e softwares de controle possibilita a automação de sistemas de monitoramento. No caso da impressão 3D, algoritmos de aprendizado supervisionado podem ser treinados com imagens de impressões bem-sucedidas e com falhas — como o spaghetti — e, com isso, realizar a detecção automática em tempo real, sem a necessidade de intervenção humana.

Cunha (2024) mostra que mesmo com sensores físicos é possível aplicar técnicas supervisionadas com bons resultados. Já Ma et al. (2023) demonstram que modelos de IA são capazes de prever, detectar e até corrigir automaticamente problemas durante a impressão de modelos médicos, evidenciando a aplicabilidade da inteligência artificial em contextos práticos e de alta precisão.

2.4 Redes Neurais Convolucionais (CNNs)

As redes neurais convolucionais (CNNs) são modelos de aprendizado profundo projetados especialmente para o processamento de dados visuais, como imagens e vídeos. Inspiradas no funcionamento do córtex visual humano, essas redes se destacam por sua capacidade de identificar automaticamente padrões espaciais, sem necessidade de engenharia manual de atributos (GOODFELLOW et al., 2016).

Uma CNN é composta por uma sequência de camadas convolucionais, que aplicam filtros (ou kernels) sobre as imagens, extraindo características locais como bordas, texturas e formas. Essas camadas são seguidas por camadas de pooling, responsáveis por reduzir a dimensionalidade dos dados, tornando o modelo mais eficiente e menos sensível a variações. Após essas etapas, os dados são processados por camadas totalmente conectadas (fully connected), que realizam a decisão final. A função de ativação ReLU introduz não-linearidade, enquanto a softmax ou sigmoid é usada na saída para classificação (FSMA, 2018; FELTRIN, 2023).

Segundo Haykin (2005), redes neurais multicamadas com retropropagação são fundamentais no processo de aprendizagem supervisionada. Esse mecanismo também é empregado nas CNNs, onde os pesos dos filtros são ajustados automaticamente durante o treinamento, com base nos erros cometidos na saída. A aplicação prática das CNNs tem mostrado grande eficácia em tarefas como reconhecimento facial, análise médica de imagens, veículos autônomos e, mais recentemente, na inspeção de processos de manufatura. Ariel Nascimento (2022), por exemplo, utilizou a arquitetura VGG16 para classificar imagens de embarcações na região amazônica com base em risco operacional, alcançando resultados expressivos de acurácia. De forma semelhante, Ebermam e Krohling (2018) demonstraram o uso de CNNs para reconhecimento de caracteres com arquitetura LeNet, apresentando os fundamentos e etapas do treinamento em linguagem acessível.

Além disso, Ma et al. (2023) destacam o uso de CNNs no monitoramento e correção de falhas durante a impressão 3D de modelos médicos, o que reforça o potencial dessas redes em sistemas automatizados de alta precisão.

Esses casos demonstram que as CNNs são ideais para aplicações em classificação binária de imagens, como é o caso da detecção de falhas visuais do tipo spaghetti em impressoras 3D. Por meio do treinamento com imagens rotuladas (falha vs. normal), é possível construir um sistema de monitoramento automatizado com alta precisão, capaz de agir em tempo real para reduzir desperdícios e danos ao equipamento.

2.5 Automação de Processos e Notificações Inteligentes

Com o avanço da manufatura digital e da Internet das Coisas (IoT), tornou-se cada vez mais comum a integração de sistemas automatizados capazes de monitorar, agir e se comunicar em tempo real. No contexto da impressão 3D, essa automação pode reduzir desperdícios, aumentar a eficiência e evitar falhas que comprometam o processo ou danifiquem o equipamento (CUNHA, 2024; MA et al., 2023).

Soluções modernas vêm utilizando visão computacional associada a redes neurais convolucionais (CNNs) para detectar falhas visuais durante a impressão. Ao identificar padrões que indiquem erro — como o spaghetti —, o sistema pode reagir de forma autônoma. Neste projeto, propõe-se a pausa automática da impressão via API do OctoPrint, imediatamente após a detecção da falha, evitando perda de material e esforço desnecessário.

Além da ação corretiva, o sistema enviará notificações por e-mail para informar o usuário sobre a falha ocorrida. Essa estratégia busca manter o operador informado mesmo à distância, garantindo maior segurança e confiabilidade no processo.

A integração com o OctoPrint é feita por meio de sua API REST, que permite enviar comandos programáticos para pausar, cancelar ou retomar a impressão. Isso possibilita uma automação simples e eficiente, sem necessidade de intervenção manual, e adaptável a impressoras já em uso (OCTOPRINT, 2024).

Trabalhos como o de Ma et al. (2023) demonstram o uso de IA para controle em tempo real em aplicações médicas, mas o foco do presente projeto é adaptar essas soluções a um ambiente doméstico ou maker, utilizando hardware acessível e interfaces conhecidas como o OctoPrint.

Essa combinação entre detecção automática via CNN, resposta imediata via API e notificação por e-mail torna o sistema proposto uma solução prática, inteligente e acessível para evitar falhas recorrentes na impressão 3D.

2.6 Trabalhos relacionados

As redes neurais convolucionais (CNNs) vêm sendo utilizadas em várias áreas, principalmente onde o reconhecimento visual é essencial. Elas se destacam porque conseguem aprender sozinhas, direto das imagens, o que é importante ou não. Isso elimina a etapa de precisar programar regras específicas. Nos últimos anos, começaram a surgir também soluções voltadas para a impressão 3D, com o objetivo de identificar falhas enquanto a peça ainda está sendo feita.

Um exemplo disso é o trabalho de Pedrota (2025), que desenvolveu um sistema para detecção de falhas do tipo *spaghetti* em impressões 3D, utilizando uma Raspberry Pi e uma câmera para a captura de dados. Através do treinamento de uma CNN com imagens reais, o modelo identificou erros com precisão e integrou-se ao OctoPrint para o envio de alertas automáticos. Contudo, o sistema operava de forma passiva, limitando-se a notificar o usuário sem interromper o processo de fabricação.

Outro trabalho interessante é o de Pessanha e Campos (2021). Eles não trabalharam com impressão 3D, mas com imagens de pele para identificar melanoma. Usaram uma CNN avançada chamada Inception-ResNet-v2, junto com transfer learning. O resultado foi uma acurácia alta, acima de 90%. Por mais que seja uma aplicação médica, esse estudo mostra como essas redes funcionam muito bem em tarefas de classificação binária — o que é parecido com separar imagens de impressão boa ou com falha.

No caso de Cunha (2024), a abordagem foi diferente. Em vez de imagens, ele utilizou sensores, como acelerômetros, para perceber vibrações fora do padrão durante a impressão. A ideia era detectar falhas que não são tão visíveis, mas que alteram o comportamento mecânico da impressora. O autor até comenta que usar visão computacional seria uma boa ideia para falhas como o spaghetti, mas aponta a falta de bases públicas como um obstáculo. Isso mostra que a área ainda está em construção.

Já Ma et al. (2023) apresentaram uma aplicação mais avançada, voltada à área médica. Eles usaram CNNs para monitorar impressões 3D de modelos de órgãos, detectando falhas como entupimentos ou falhas de extrusão. O diferencial aqui é que o sistema conseguia corrigir os problemas em tempo real, sem precisar de intervenção humana. Porém que isso foi feito num ambiente controlado e com infraestrutura robusta, o que nem sempre é o caso de quem usa impressora 3D em casa ou no laboratório.

Esses trabalhos, apesar de promissores, ainda não cobrem tudo. Muitos focam só na detecção, e poucos automatizam alguma reação. Alguns dependem de sensores extras, enquanto outros exigem processamento pesado. A proposta deste projeto tenta equilibrar esses pontos. Com uma CNN simples, mas eficaz, e integração direta ao OctoPrint, o sistema detecta falhas do tipo spaghetti, envia alertas por e-mail e ainda pausa automaticamente a impressão. Isso pode ser muito útil para quem imprime em casa, em laboratório ou em espaços compartilhados, onde nem sempre é possível ficar olhando a máquina o tempo todo.

3. Metodologia

3.1 Arquitetura do Sistema

O sistema de detecção de falhas foi concebido em uma arquitetura modular, projetada para operar de forma autônoma e em tempo real, integrando componentes de Visão Computacional, controle de hardware e interface de comunicação. A solução é centrada no servidor OctoPrint, que atua como o ponto de controle da impressora 3D.

O funcionamento lógico do sistema desenvolvido para o monitoramento das impressões 3D é detalhado no fluxograma a seguir. Como ilustrado na Figura 1, o ciclo se inicia com a captura e o pré-processamento de frames, seguidos pela inferência através da rede neural convolucional para a tomada de decisão autônoma sobre a presença de falhas.

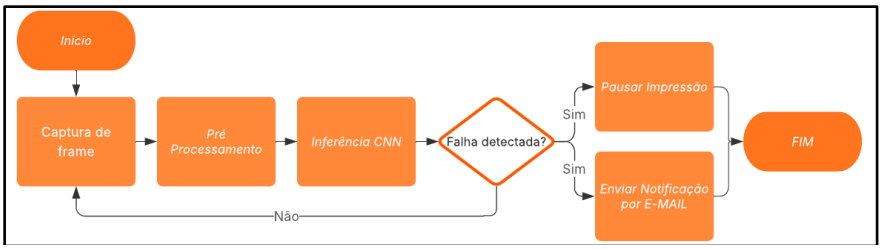

Figura 1. Fluxograma de funcionamento do sistema de detecção.
Fonte: Elaborado pelo autor (2024)

3.1.1. Componentes Funcionais

A arquitetura lógica do sistema é composta por três módulos principais que interagem para completar o ciclo de monitoramento e automação:

1. **Módulo de Captura e Processamento:** Responsável por acessar o *stream* de vídeo (MJPEG) da câmera da impressora e fornecer os *frames* para a análise.
2. **Módulo de Inferência de ML:** Carrega e executa o modelo EfficientNet-B0 (descrito em 3.3), realizando a classificação binária de cada *frame* capturado como "Normal" ou "Spaghetti".
3. **Módulo de Automação e Notificação:** Gerencia a comunicação com a API REST do OctoPrint para comandos de pausa e retoma, e dispara alertas por e-mail via protocolo SMTP-SSL.

3.1.2. Fluxo de Dados e Controle em Tempo Real

O sistema opera em um *pipeline* contínuo, orquestrado pela aplicação principal, garantindo a análise apenas durante a impressão.

1. **Monitoramento da Impressão:** Uma *thread* dedicada (monitorar_octoprint) verifica o estado do OctoPrint em intervalos regulares. A análise de ML só é iniciada **automaticamente** quando a impressora está no estado **"printing"**.
2. **Ciclo de Análise:** Ao iniciar, o sistema começa a capturar *frames* do *stream* MJPEG. Cada *frame* é imediatamente processado pelo **Módulo de Inferência**.

3. **Decisão Autônoma:** Se a probabilidade da classe "Spaghetti" **exceder o limiar predefinido** (neste caso, 50%), o sistema aciona o Módulo de Automação para intervenção imediata.
4. **Ação de Saída:** O Módulo de Automação envia o comando pause para a API do OctoPrint e dispara uma **notificação de alerta por e-mail** (SMTP-SSL), anexando o *frame* da falha para a rápida intervenção do usuário.
5. **Interface de Usuário (Controle):** O servidor web Flask (app.py) fornece uma interface (index.html) para visualização em tempo real do status, permitindo controle manual (start/stop/pause) e acesso ao histórico de detecções.

3.1.3. Tecnologias e Infraestrutura

As principais tecnologias e bibliotecas utilizadas para a implementação dos módulos do sistema estão detalhadas na Tabela 1.

Tabela 1. Tecnologias e bibliotecas utilizadas no desenvolvimento.

Componente	Tecnologia	Função no Sistema
Núcleo de ML	PyTorch, EfficientNet-B0	Carregamento do modelo e execução da inferência.
Processamento de Imagem	OpenCV (cv2), PIL	Captura, manipulação e pré-processamento dos *frames*.
Interface Web	Flask (Python), HTML/Tailwind CSS e JavaScript	Servidor web para API de *status* e interface de controle.
Comunicação/Automação	requests, OctoPrint API	Requisições HTTP para controle da impressora e obtenção de dados de *status*.
Concorrência	threading	Execução não-bloqueante das rotinas de análise e monitoramento.

Fonte: Elaborado pelo autor (2025).

3.2 Detalhamento do Modelo de Machine Learning

A solução de detecção de falhas de filamento, ou "Spaghetti", foi desenvolvida utilizando uma arquitetura de Redes Neurais Convolucionais (CNNs) baseada em Transfer Learning, visando a alta precisão com eficiência computacional.

3.2.1. Arquitetura CNN e Transfer Learning

O modelo de classificação de imagens adotado foi o EfficientNet-B0 . Ele pertence à família EfficientNet, que se destaca por balancear de forma eficiente a profundidade, a largura e a resolução da rede, otimizando a relação entre desempenho e complexidade computacional.

Para acelerar e refinar o treinamento, o modelo foi inicializado com pesos pré-treinados do ImageNet (EfficientNet_B0_Weights.IMAGENET1K_V1), utilizando a técnica de Transfer Learning. A etapa de adaptação envolveu a substituição da camada de classificação final do EfficientNet-B0 por uma nova camada densa linear com duas saídas, correspondentes às classes binárias do problema: Normal e Spaghetti.

A estratégia de treinamento utilizada foi o Fine-Tuning completo. Isso significa que todos os parâmetros da rede (param.requires_grad = True) tiveram seus gradientes habilitados e foram ajustados durante o processo. Essa abordagem foi fundamental para que o modelo pudesse adaptar os recursos gerais (extraídos do ImageNet) ao domínio específico das texturas e características visuais das falhas de impressão 3D, em vez de apenas ajustar a camada final.

3.2.2. Dataset e Pré-processamento Otimizado

O pré-processamento das imagens de treinamento foi conduzido no *script* (treino_melhor_qualidade.py) com o objetivo de maximizar a captação de detalhes finos, essenciais para a identificação de falhas incipientes (como a falha *Spaghetti*).

O processo de redimensionamento utilizou uma resolução elevada de 352 X 352 pixels. Em seguida, foi aplicado um recorte aleatório para o tamanho de 320 X 320. Essa resolução final é superior ao padrão de 224 X 224 tipicamente usado para o EfficientNet-B0, o que garante a preservação de detalhes cruciais do filamento impresso.

Para aumentar a robustez e generalização do modelo, foi aplicada uma série de técnicas de Aumento de Dados (*Augmentation*) . Isso permitiu simular variações de captura da câmera. As transformações incluídas foram: RandomResizedCrop, RandomHorizontalFlip, RandomRotation e ColorJitter, expondo o modelo a variações de iluminação, escala e posição da peça.

O dataset utilizado neste trabalho foi construído pelo próprio autor a partir de imagens reais capturadas durante impressões 3D em ambiente operacional. Após a aplicação de técnicas de aumento de dados (data augmentation), o conjunto final totalizou 1.452 imagens, sendo 792 imagens da classe “Normal” e 660 imagens da classe “Spaghetti”.

As técnicas de augmentation foram aplicadas de forma que cada imagem original gerasse múltiplas variações, aumentando a diversidade visual do conjunto e contribuindo

para a robustez do modelo. Ressalta-se que os valores apresentados já correspondem ao dataset final após a aplicação dessas técnicas.

O dataset foi dividido em conjuntos de treinamento e validação, mantendo a proporção entre as classes para garantir uma avaliação justa do desempenho do modelo durante o processo de aprendizado. Este conjunto é composto por imagens reais e variações geradas por técnicas de aumento de dados (Figura 2 e Figura 3), que totalizaram 1.452 imagens. A Figura 2 apresenta a classe 'Normal', destacando a imagem original (A) e suas derivações (B, C, D), enquanto a Figura 3 exibe a classe 'Spaghetti', evidenciando como o *augmentation* simula diferentes condições de captura para o emaranhado de filamento.

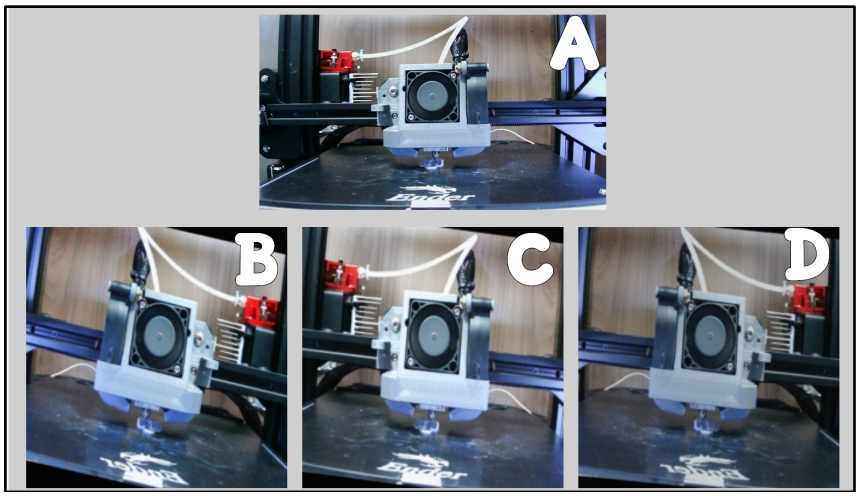

Figura 2. Amostras da classe Normal: (A) imagem original; (B, C, D) variações via *data augmentation*. Fonte: Elaborado pelo autor (2025).

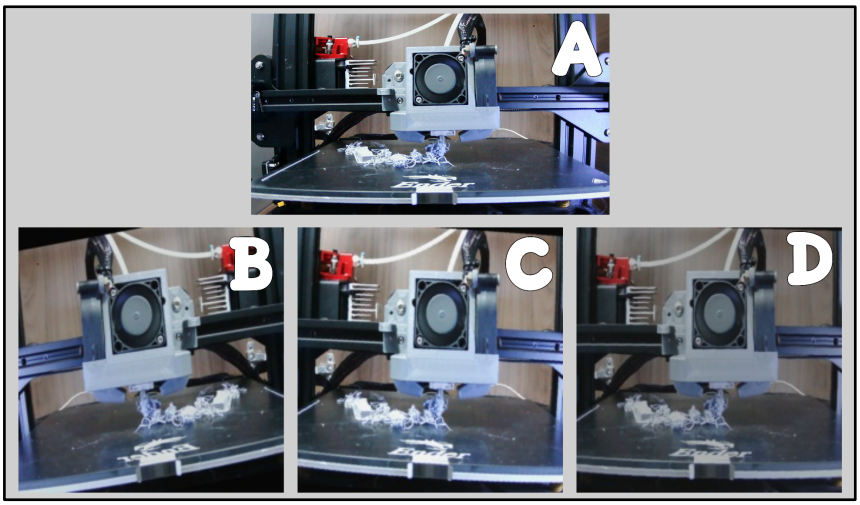

Figura 3. Amostras da classe Spaghetti: (A) imagem original; (B, C, D) variações via *data augmentation*. Fonte: Elaborado pelo autor (2025).

3.2.3. Treinamento e Critérios de Otimização

O treinamento foi conduzido sob as seguintes configurações:

- **Dispositivo e Reprodução:** Utilizou-se GPU (CUDA) quando disponível, com semente de *Random Number Generator* (RNG) fixada em 42 para garantir a reprodutibilidade dos resultados.
- **Balanceamento de Classes:** Para mitigar o desequilíbrio entre o número de amostras "Normal" (majoritárias) e "Spaghetti" (minoritárias), foi implementada a função de custo *Cross-Entropy Loss* com pesos balanceados. Os pesos são calculados inversamente proporcionais à frequência de cada classe no *dataset*.
- **Otimizador e Scheduler:** O otimizador AdamW foi utilizado, e a taxa de aprendizado (LR) foi ajustada dinamicamente utilizando o *scheduler* ReduceLROnPlateau com base na acurácia de validação. Isso garante que a taxa de aprendizado seja reduzida quando o modelo para de melhorar, promovendo uma convergência mais estável.

3.3 Integração e Automação

O projeto transpõe a detecção da CNN para a esfera da automação, utilizando o OctoPrint como plataforma central de controle e notificação.

3.3.1. Comunicação e Controle do OctoPrint

A interação com a impressora é realizada por meio da API REST do OctoPrint. A chave de acesso (API Key) e o endereço do servidor são carregados a partir de variáveis de ambiente.

- **Pausa Autônoma:** Em caso de detecção positiva de falha ("spaghetti") e se a falha ainda não tiver sido registrada (if not falha_detectada), o sistema executa uma requisição HTTP POST para o endpoint de controle de *job* do OctoPrint, enviando o comando pause. Esta ação é imediata e visa interromper o desperdício de material.
- **Monitoramento de Estado:** Uma *thread* dedicada (monitorar_octoprint) mantém a escuta ativa do status do OctoPrint. Quando o estado da impressora muda para "printing" (imprimindo), o sistema automaticamente inicia a análise (start_analysis). Similarmente, ele interrompe a análise quando o estado muda para "cancelled", "complete" ou "operational".

A integração física e lógica entre os módulos segue uma estrutura em camadas, facilitando a comunicação entre o script de detecção e o controle da impressora. Essa organização pode ser observada na Figura 4.

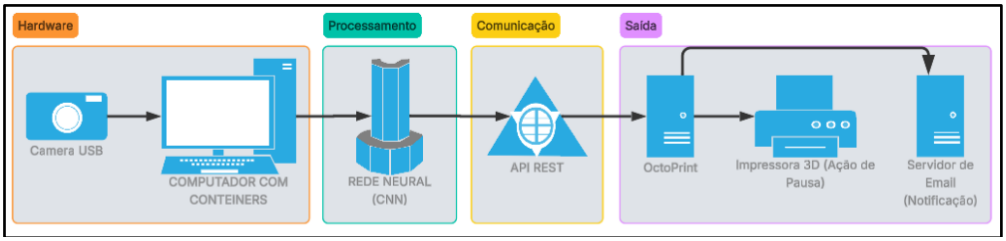

Figura 4. Diagrama de arquitetura e integração entre camadas de hardware e software.
Fonte: Elaborado pelo autor (2025).

3.3.2. Sistema de Alerta e Registro de Histórico

O sistema não apenas intervém na falha, mas também garante que o usuário seja prontamente informado sobre a ação e mantém um registro persistente de todas as análises realizadas.

Para o alerta imediato, o sistema utiliza o protocolo SMTP-SSL para o envio de notificações críticas. A detecção de uma falha "spaghetti" dispara automaticamente um e-mail para o endereço configurado, contendo a mensagem de que a impressão foi pausada. O ponto mais importante dessa notificação é o anexo: o frame da imagem capturado no exato momento da detecção é enviado, permitindo ao usuário verificar a falha de forma instantânea e tomar a decisão de retomar ou cancelar o job.

Todas as predições de inferência, sejam elas acionadas automaticamente ou capturadas manualmente, são registradas no arquivo historico.json. Cada registro é um log detalhado que inclui o timestamp da análise, o nome do arquivo de impressão ativo (job), informação obtida via API do OctoPrint, e o resultado completo da predição, englobando a classe detectada e as porcentagens de probabilidade calculadas pela CNN. Esse histórico é crucial para a auditoria do desempenho do sistema e para futuras análises de treinamento.

4. Resultados e Análise

Este capítulo apresenta os resultados obtidos após o treinamento do modelo de Rede Neural Convolucional (CNN) e a validação do sistema em ambiente operacional, confirmando a eficácia da solução proposta em atingir os objetivos estabelecidos.

4.1. Performance do Modelo Durante o Treinamento

O treinamento do modelo EfficientNet-B0 foi realizado por 40 épocas, utilizando a estratégia de Fine-Tuning completo e pesos de classe balanceados para maximizar a precisão da detecção.

4.1.1. Métricas de Desempenho

Os resultados finais demonstraram a alta capacidade de aprendizado do modelo em diferenciar imagens "Normal" de "Spaghetti".

Acurácia de Validação Máxima: O modelo atingiu uma acurácia máxima de 99.66% no dataset de validação.

Nome do Modelo Final: O arquivo de pesos que alcançou esse pico de desempenho foi salvo como modelo_final_detalhado.pt.

Perda de Treinamento e Validação: Durante as épocas finais, a Perda de Treinamento (train_loss) se manteve extremamente baixa, aproximando-se de 0.001, enquanto a Perda de Validação (val_loss) permaneceu estável em torno de 0.05, indicando que o modelo generalizou o aprendizado de forma eficaz, sem evidências significativas de overfitting.

O resultado de 99.66% na acurácia de validação (val_acc) comprova que o modelo treinado com imagens de alta resolução e data augmentation é robusto e está apto para a implementação em tempo real.

4.2. Análise da Detecção em Tempo Real e Eficácia da Automação

Os testes práticos de detecção em tempo real foram realizados em um servidor externo virtualizado, operando de forma independente do hardware da impressora 3D. Tanto o servidor OctoPrint quanto o sistema de inferência foram executados em um ambiente dedicado, configurado com 4 núcleos de CPU e 2 GB de memória RAM.

Durante a fase de inferência, o sistema operou inteiramente nesse ambiente, demonstrando desempenho suficiente para analisar os frames da câmera sem comprometer o processo de impressão. Ressalta-se que, no estágio atual, o sistema não foi executado em dispositivos embarcados limitados (como Raspberry Pi), sendo essa otimização sugerida para trabalhos futuros.

Após o treinamento do modelo de Rede Neural Convolucional, a solução foi integrada ao ambiente operacional por meio do OctoPrint, permitindo a avaliação do comportamento do sistema em tempo real durante impressões reais. Essa etapa teve como objetivo verificar não apenas a capacidade de detecção do modelo, mas também a efetividade da automação proposta, incluindo a pausa automática da impressão e o envio de notificações ao usuário.

Os testes foram realizados em condições reais de uso, com variações naturais de iluminação, ângulo da câmera e modelos impressos. A análise foi ativada apenas quando a impressora se encontrava no estado de impressão (“printing”), conforme o monitoramento contínuo do OctoPrint, garantindo que o sistema operasse de forma contextualizada e eficiente.

O modelo de 99.66% de acurácia foi integrado ao sistema OctoPrint para testes em ambiente real, conforme a metodologia. A análise se concentrou em medir a eficácia da detecção e a resposta do sistema de automação.

Além da acurácia, observou-se estabilidade entre as curvas de treinamento e validação, indicando que o modelo não apresentou comportamento de overfitting significativo. O uso de pesos balanceados na função de custo contribuiu para uma melhor aprendizagem da classe minoritária (“Spaghetti”), reduzindo o viés de classificação.

4.2.1. Eficácia na Detecção e Mitigação de Falhas

Durante os testes realizados, o sistema demonstrou alta eficácia na identificação de falhas do tipo spaghetti. Todas as ocorrências reais desse tipo de falha registradas nos testes foram corretamente detectadas pelo modelo, acionando automaticamente a pausa da impressão por meio da API do OctoPrint.

A atuação precoce do sistema permitiu interromper o processo ainda nos estágios iniciais da falha, evitando a extrusão contínua de filamento no ar e reduzindo significativamente o desperdício de material. Além disso, a automação reduziu drasticamente a dependência de monitoramento constante por parte do operador, cumprindo o objetivo de aumentar a segurança e a confiabilidade do processo de impressão 3D.

Esses resultados indicam que a integração entre a CNN e o sistema de controle da impressora é viável e eficaz, demonstrando que técnicas de visão computacional podem ser aplicadas com sucesso em ambientes reais e não controlados, como laboratórios e uso doméstico.

Em termos de desempenho temporal, o sistema apresentou latência compatível com operação em tempo real. O tempo médio entre a captura do frame da câmera, o pré-processamento da imagem, a inferência pelo modelo EfficientNet-B0 e a tomada de decisão foi inferior a 1 segundo, permitindo a pausa da impressão ainda nos estágios iniciais da falha.

Esse resultado evidencia a eficiência da arquitetura escolhida, que consegue equilibrar precisão e baixo custo computacional, sendo adequada para aplicações de monitoramento contínuo em impressoras 3D.

4.2.2. Robustez e Tratamento de Falsos Positivos

Embora o sistema tenha apresentado eficácia satisfatória, foram observadas algumas ocorrências de falsos positivos durante a operação. A análise dos registros indicou que, do total de 16 alertas de falha gerados, 6 corresponderam a falsos positivos, representando aproximadamente 37,5% dos alertas emitidos.

Essas detecções incorretas estiveram, em sua maioria, associadas a fatores externos, como peças nas quais a sua estrutura se parece com a falha spaghetti, variações de iluminação, sombras projetadas sobre a peça ou ruídos visuais no ambiente de impressão. Para reduzir o impacto dessas ocorrências, foi adotado um limiar mínimo de probabilidade de 50% para a confirmação da falha, evitando pausas excessivas ou injustificadas.

Mesmo nos casos de falso positivo, o impacto operacional foi considerado baixo, uma vez que a ação automática do sistema se limita à pausa da impressão, permitindo rápida verificação visual e retomada do processo pelo operador. Nenhuma falha real deixou de ser detectada durante os testes, indicando que o sistema prioriza a sensibilidade à falha em detrimento de pequenas interrupções, característica desejável em um cenário de prevenção de desperdício e proteção do equipamento.

4.3. Análise da Interface Web e Usabilidade

Além da detecção automática de falhas, o sistema proposto inclui uma interface web desenvolvida para facilitar o acompanhamento do estado da análise e o controle do processo por parte do usuário. Essa interface teve papel fundamental na validação prática da solução e na interação homem-máquina.

4.3.1. Interface e controle do sistema

A interface web foi implementada utilizando o framework Flask, em conjunto com tecnologias web padrão, permitindo o acesso via navegador a partir de qualquer dispositivo conectado à rede local. Por meio dessa interface, o usuário pode visualizar o status atual da análise, acompanhar o resultado da última inferência realizada e controlar manualmente a ativação ou desativação do sistema de detecção.

A simplicidade da interface contribuiu para uma boa usabilidade, tornando o sistema acessível mesmo para usuários sem conhecimento técnico aprofundado em inteligência artificial ou visão computacional. Dessa forma, a solução se apresenta como prática e aplicável ao contexto real de uso de impressoras 3D.

A interface desenvolvida para o monitoramento em tempo real permite ao usuário visualizar a transmissão da câmera e o status da classificação. Como ilustrado na Figura 5, o sistema exibe a probabilidade de falha e o estado atual da impressão, oferecendo botões para intervenção manual caso necessário.

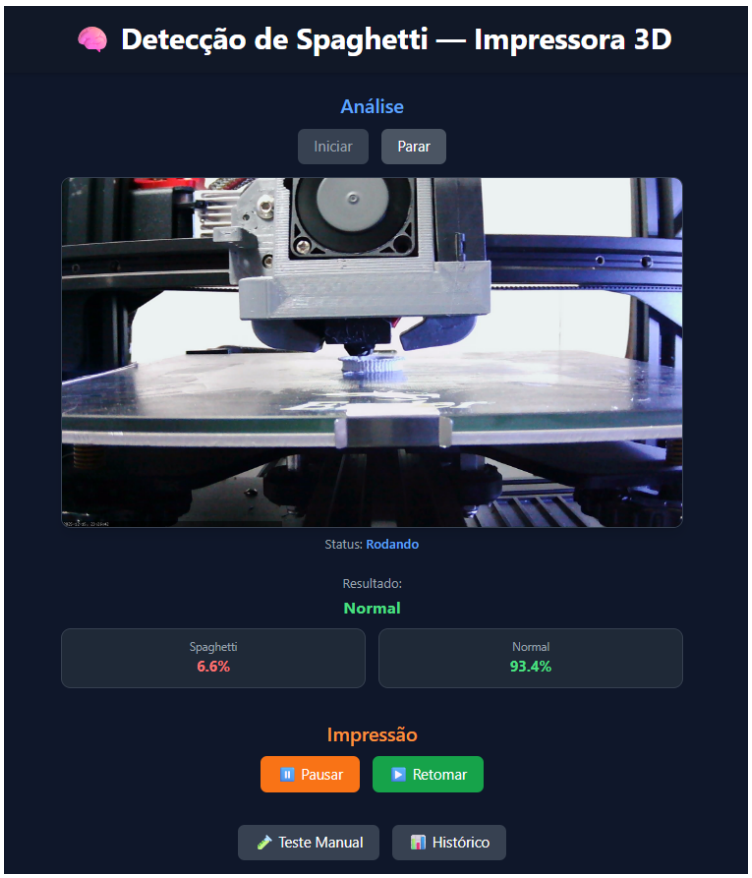

Figura 5. Interface web de monitoramento integrada ao OctoPrint.
Fonte: Elaborado pelo autor (2025)

4.3.2. Ferramentas de Teste e Validação

Para validar o funcionamento completo do sistema, foram realizados testes em diferentes cenários operacionais, incluindo impressões bem-sucedidas, impressões com falhas reais e situações que resultaram em falsos positivos. Também foram avaliadas transições de estado da impressora, como início, pausa, cancelamento e finalização da impressão.

Esses testes confirmaram que o sistema responde corretamente às mudanças de estado do OctoPrint, iniciando e interrompendo a análise automaticamente conforme necessário. Além disso, foi possível validar a comunicação correta entre os módulos de inferência, automação e notificação, garantindo a consistência da solução como um todo.

Assim que a falha é confirmada pelo modelo, o sistema executa o protocolo de segurança via OctoPrint. Além da interrupção física da impressão, uma notificação é disparada automaticamente para o e-mail do usuário, como demonstrado na Figura 6. O alerta contém informações sobre a detecção, permitindo que o usuário tome providências mesmo estando distante do equipamento.

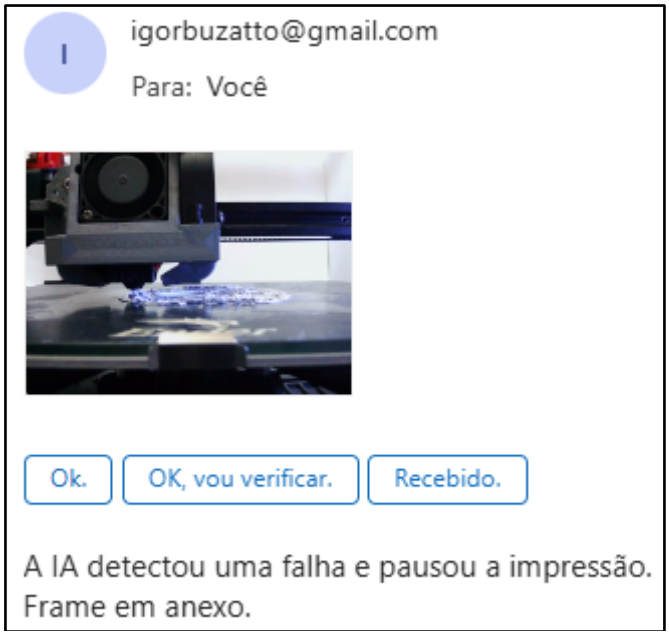

Figura 6. Exemplo de notificação de falha enviada automaticamente por e-mail.
Fonte: Elaborado pelo autor (2025).

4.3.3. Gestão e Análise do Histórico

Todas as inferências realizadas pelo sistema foram registradas em arquivos de histórico, organizados em categorias de impressões normais, falhas do tipo spaghetti e falsos positivos. Ao todo, foram analisados 33 registros, sendo 17 classificados como impressões normais, 10 como falhas reais e 6 como falsos positivos.

Esses registros permitiram uma análise detalhada do comportamento do modelo em ambiente real, possibilitando a identificação de padrões associados a erros de classificação e a avaliação da robustez do sistema. O histórico também se mostra relevante para futuras melhorias, como ajustes no limiar de decisão, refinamento do modelo e re-treinamento com novos dados coletados durante a operação.

A manutenção desse histórico contribui para a transparência do sistema e para a validação contínua de seu desempenho, reforçando a confiabilidade da solução proposta.

5. Considerações Finais

O desenvolvimento deste trabalho permitiu a criação e validação de um sistema inteligente para a detecção automática de falhas do tipo spaghetti em impressoras 3D FDM. A integração entre técnicas de Visão Computacional, por meio de Redes Neurais Convolucionais (CNN), e a automação via API do OctoPrint demonstrou ser uma solução eficaz para mitigar desperdícios de material e aumentar a segurança do processo de manufatura aditiva.

O modelo escolhido, baseado na arquitetura EfficientNet-B0 e aprimorado com técnicas de Transfer Learning e Fine-Tuning, apresentou um desempenho robusto. Durante a fase de treinamento, a acurácia de validação de 99,66% indicou uma alta capacidade de generalização na distinção entre impressões normais e falhas. Nos testes práticos em ambiente real, essa capacidade se confirmou: o sistema foi capaz de identificar corretamente todas as ocorrências de falhas reais, acionando a pausa automática e notificando o usuário conforme planejado.

Embora tenham sido observados falsos positivos, decorrentes principalmente de variações de iluminação e sombras no ambiente de impressão, o comportamento conservador do sistema, priorizando a pausa em situações de dúvida, mostrou-se adequado para a proposta de preservação do equipamento e economia de filamento. A implementação da interface web e do sistema de notificações garantiu que o operador mantivesse o controle e a ciência do status da impressão, mesmo remotamente.

Conclui-se, portanto, que os objetivos propostos foram alcançados. O sistema desenvolvido oferece uma alternativa viável, de baixo custo e alta eficiência para o monitoramento de impressoras 3D, superando as limitações da observação humana contínua e contribuindo para a automação inteligente no contexto da Indústria 4.0 e do movimento Maker.

6. Trabalhos Futuros

Apesar dos bons resultados obtidos, o sistema desenvolvido ainda pode ser aprimorado e expandido em diferentes aspectos, abrindo caminho para evoluções que tornem a solução mais precisa e adaptável a diferentes cenários de impressão 3D.

Uma possibilidade de continuidade do trabalho é o uso mais aprofundado dos registros gerados pelo próprio sistema ao longo do tempo. A análise desses históricos pode auxiliar na identificação de situações recorrentes que levam a falsos positivos, permitindo ajustes progressivos no comportamento do sistema conforme o padrão de uso da impressora.

Outra evolução natural consiste na utilização de informações já disponíveis no OctoPrint, como altura da camada, velocidade de impressão e tipo de material configurado. A combinação desses dados com a análise das imagens pode contribuir para decisões mais confiáveis, reduzindo ambiguidades na classificação apenas visual.

Também é possível investigar formas de tornar o sistema mais eficiente do ponto de vista computacional, possibilitando sua execução direta em dispositivos com menor capacidade de processamento, como a própria Raspberry Pi. Isso tornaria a solução mais simples de instalar e mais acessível para uso doméstico e educacional.

Além disso, o sistema pode evoluir para diferenciar estágios iniciais e avançados de falhas, permitindo ações graduais, como alertas preventivos antes da pausa automática da impressão. Essa abordagem aumentaria a flexibilidade do sistema e reduziria interrupções desnecessárias.

Como forma de reduzir a ocorrência de falsos positivos observada durante os testes, uma possível melhoria futura consiste na adoção de uma janela de confirmação

temporal, na qual a impressão só seria pausada caso a falha fosse detectada em múltiplos frames consecutivos. Essa estratégia pode reduzir interrupções desnecessárias sem comprometer a sensibilidade do sistema à detecção de falhas reais.

Por fim, a ampliação da detecção para outros tipos de falhas comuns na impressão 3D, bem como a visualização das regiões da imagem que influenciam a decisão do modelo, pode contribuir para maior compreensão do comportamento do sistema e facilitar sua validação e aceitação por usuários e pesquisadores.

Referências

CHUA, C. K.; **LEONG**, K. F. *3D Printing and Additive Manufacturing: Principles and Applications*. 4. ed. Singapore: World Scientific, 2014.

CUNHA, Leonardo Kenny Treichel da. *Sistema de aquisição de dados e detecção de falhas para impressoras 3D utilizando modelos inteligentes*. 2024. Projeto de Diplomação II – Universidade Federal do Rio Grande do Sul, Escola de Engenharia, Departamento de Engenharia Elétrica, Porto Alegre, 2024.

EBERMAM, Elivelto; **KROHLING**, Renato A. Uma Introdução Compreensiva às Redes Neurais Convolucionais: Um Estudo de Caso para Reconhecimento de Caracteres Alfabéticos. **Revista de Sistemas de Informação da FSMA**, n. 22, p. 49-59, 2018.

FELTRIN, Fernando. *Redes neurais artificiais convolucionais*. São Paulo: Amazon Serviços de Varejo do Brasil, 2023. eBook Kindle. Disponível em: https://www.amazon.com.br/dp/B0C5S4K239. Acesso em: 29 jun. 2025.

GIBSON, Ian; **ROSEN**, David W.; **STUCKER**, Brent. *Additive Manufacturing Technologies: Rapid Prototyping to Direct Digital Manufacturing*. New York: Springer, 2010.

GIBSON, Ian; **ROSEN**, David; **STUCKER**, Brent. *Additive Manufacturing Technologies: 3D Printing, Rapid Prototyping, and Direct Digital Manufacturing*. 2. ed. New York: Springer Science+Business Media, 2015.

GOODFELLOW, Ian; **BENGIO**, Yoshua; **COURVILLE**, Aaron. *Deep Learning*. Cambridge, MA: MIT Press, 2016.

HAYKIN, Simon. *Redes neurais: Princípios e prática*. 2. ed. Tradução Paulo Martins Engel. Porto Alegre: Bookman, 2005. (Baseado na edição de 1999).

MA, Liang et al. Application of artificial intelligence in 3D printing physical organ models. **Materials Today Bio**, v. 23, p. 100792, 2023.

NASCIMENTO, Ariel Victor do. *Rede Neural Convolucional (CNN) Aplicada na Análise de Risco de Acidentes das Embarcações que Navegam nos Rios da Amazônia*. 2022. Dissertação (Mestrado em Engenharia Naval) – Universidade Federal do Pará, Belém, 2022.

OCTOPRINT. OctoPrint Documentation. 2024. Disponível em: https://docs.octoprint.org/en/master/index.html. Acesso em: 29 jun. 2025.

PEDROTA, Victor Hugo Godoi. *Detecção de Falhas em Impressões 3D: Uma Abordagem de Aprendizado de Máquina para Identificação de Problemas de Impressão*. 2025. 83 f. Trabalho de Conclusão de Curso (Engenharia de Computação) – Universidade Federal de São Paulo, São José dos Campos, 2025.

PESSANHA, Gabriel R. G.; **CAMPOS**, Eleanderson. Um Modelo de Inteligência Artificial Para Detecção de Melanoma via Redes Neurais Convolucionais. In: ESCOLA REGIONAL DE COMPUTAÇÃO APLICADA À SAÚDE (ERCAS), 8., 2021, São Paulo. **Anais** [...]. Porto Alegre: Sociedade Brasileira de Computação, 2021. p. 46-49.

RUSSELL, Stuart; **NORVIG**, Peter. *Inteligência Artificial*. Tradução da Terceira Edição. Rio de Janeiro: Elsevier, 2013.

SZELISKI, Richard. *Computer Vision: Algorithms and Applications*. London: Springer, 2011.