INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA - CÂMPUS FLORIANÓPOLIS
DEPARTAMENTO ACADÊMICO DE ELETROTÉCNICA
CURSO DE GRADUAÇÃO EM ENGENHARIA ELÉTRICA

LUCAS SOUZA OLIVEIRA MENDONÇA

DETECÇÃO DE DEFEITOS EM COMPONENTES DE PLACAS DE CIRCUITO IMPRESSO COM APRENDIZADO PROFUNDO

FLORIANÓPOLIS, 2025.

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA - CÂMPUS FLORIANÓPOLIS
DEPARTAMENTO ACADÊMICO DE ELETROTÉCNICA
CURSO DE GRADUAÇÃO EM ENGENHARIA ELÉTRICA

LUCAS SOUZA OLIVEIRA MENDONÇA

DETECÇÃO DE DEFEITOS EM COMPONENTES DE PLACAS DE CIRCUITO IMPRESSO COM APRENDIZADO PROFUNDO

Trabalho de Conclusão de Curso submetido ao Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina como parte dos requisitos para obtenção do título de Engenheiro Eletricista.

Orientador:
Prof. Sérgio Luciano Ávila, Dr.

FLORIANÓPOLIS, 2025.

Ficha de identificação da obra elaborada pelo autor.

Mendonça, Lucas Souza Oliveira
DETECÇÃO DE DEFEITOS EM COMPONENTES DE PLACAS DE CIRCUITO IMPRESSO COM APRENDIZADO PROFUNDO / Lucas Souza Oliveira Mendonça; orientação de Sérgio Luciano Ávila - Florianópolis, SC, 2025.
62 p.

Trabalho de Conclusão de Curso (TCC) - Instituto Federal de Santa Catarina, Câmpus Florianópolis. Bacharelado em Engenharia Elétrica. Departamento Acadêmico de Eletrotécnica.
Inclui Referências.

1. Aprendizado Profundo. 2. Atenção Com Embaralhamento. 3. Detecção de Objetos. 4. Placas de Circuito Impresso. 5. YOLO. I. Ávila, Sérgio Luciano
. II. Instituto Federal de Santa Catarina. III. DETECÇÃO DE DEFEITOS EM COMPONENTES PLACAS DE CIRCUITO IMPRESSO COM APRENDIZADO PROFUNDO.

DETECÇÃO DE DEFEITOS EM COMPONENTES DE PLACAS DE CIRCUITO IMPRESSO COM APRENDIZADO PROFUNDO

LUCAS SOUZA OLIVEIRA MENDONÇA

Este trabalho foi julgado adequado para obtenção do título de Engenheiro(a) Eletricista e aprovado na sua forma final pela banca examinadora do Curso de Graduação em Engenharia Elétrica do Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina.

Florianópolis, 24 de fevereiro, 2025.

Banca Examinadora:

gov.br
Documento assinado digitalmente
SERGIO LUCIANO AVILA
Data: 07/03/2025 17:12:03-0300
Verifique em https://validar.iti.gov.br

_____

Sérgio Luciano Ávila, Dr. Eng.

gov.br
Documento assinado digitalmente
GABRIEL BEU NOGUEIRA DE MACEDO
Data: 08/03/2025 11:04:30-0300
Verifique em https://validar.iti.gov.br

_____

Gabriel Beu Nogueira de Macedo, M. Eng.

gov.br
Documento assinado digitalmente
MAURICIO EDGAR STIVANELLO
Data: 08/03/2025 11:22:54-0300
Verifique em https://validar.iti.gov.br

_____

Maurício Edgar Stivanello, Dr. Eng.

AGRADECIMENTOS

Aos meus familiares, pelo apoio, paciência e incentivo ao longo de toda a minha trajetória acadêmica. Ao Instituto Federal de Santa Catarina, pelo ensino de excelência e pelas oportunidades de aprendizado oferecidas. A todos que contribuíram para a realização deste trabalho, seja por meio de palavras de encorajamento, compartilhamento de conhecimento ou apoio técnico. Ao meu orientador, Sérgio Luciano Ávila, pela dedicação, orientação e incentivo ao meu crescimento profissional e acadêmico.

RESUMO

Montagens de Placas de Circuito Impresso – *Printed Circuit Board Assemblies* (PCBAs) são uma etapa crítica na fabricação de produtos eletrônicos, onde mesmo pequenos defeitos podem levar a falhas significativas no produto final. Embora diversas soluções tenham sido realizadas sobre inspeção de defeitos em nível de superfície para Placas de Circuito Impresso, há uma lacuna notável em estudos e metodologias voltadas especificamente para a detecção de defeitos em componentes durante a montagem das placas com técnicas de aprendizado profundo. A detecção precisa e a localização exata de defeitos são essenciais para garantir a qualidade do produto, pois as placas apresentam desafios mais complexos devido à sua estrutura tridimensional e à variedade de componentes. Este trabalho aborda essa lacuna, propondo uma nova metodologia que utiliza detecção de objetos com YOLOv11, aprimorada com uma arquitetura personalizada que integra o mecanismo de Atenção com Embaralhamento para melhorar a detecção de defeitos. O modelo proposto foi desenvolvido com um conjunto de dados reais de uma instalação de fabricação de eletrônicos em larga escala, sendo capaz de detectar e localizar defeitos de forma eficaz em diversos componentes. Os resultados demonstram a eficiência dessa abordagem na identificação e localização de defeitos em PCBAs, atingindo uma precisão de 96, 93% e um tempo de inferência de apenas 2,20 ms, contribuindo significativamente para avanços no controle de qualidade automatizado na fabricação de eletrônicos em grande escala.

Palavras-chave: Aprendizado Profundo. Atenção com Embaralhamento. Detecção de Objetos. Placas de Circuito Impresso. YOLO.

ABSTRACT

Printed Circuit Board Assemblies (PCBAs) are a critical stage in the manufacturing of electronic products, where even minor defects can lead to significant failures in the final product. While numerous solutions have been developed for surface-level defect inspection in Printed Circuit Boards, there is a notable gap in studies and methodologies specifically focused on detecting component defects in PCBA using deep learning techniques. Accurate detection and precise localization of defects are essential to ensuring product quality, as PCBAs pose more complex challenges due to their three-dimensional structure and the variety of components. This work addresses this gap by proposing a new methodology that leverages object detection with YOLOv11, enhanced with a custom architecture that integrates Shuffle Attention to improve defect detection. The proposed model was developed and validated in a real-world environment dataset within a large-scale electronics manufacturing facility, where it effectively detects and locates defects across various components. The results demonstrate the effectiveness of this approach in identifying and localizing defects in PCBAs, achieving an accuracy of 96, 9% and an inference time of just 2,2 ms, significantly contributing to advancements in automated quality control for large-scale electronics manufacturing.

Keywords: Deep Learning. Object Detection. Printed Circuit Board Assemblies. Shuffle Attention. YOLO.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AOI	*Automated Optical Inspection*
AP	*Average Precision*
ANN	*Artificial Neural Network*
CBAM	*Convolutional Block Attention Module*
CNN	*Convolutional Neural Network*
C2PSA	*Cross Stage Partial with Spatial Attention*
CSP	*Cross Stage Partial*
FPN	*Feature Pyramid Network*
GAP	*Global Average Pooling*
GMP	*Global Max Pooling*
GN	*Group Normalization*
HOG	*Histograms of Oriented Gradients*
mAP	*mean Average Precision*
PCB	*Printed Circuit Board*
PCBA	*Printed Circuit Board Assembly*
R-CNN	*Regions with Convolutional Neural Networks*
ReLU	*Rectified Linear Unit*
SGD	*Stochastic Gradient Descent*
SA	*Shuffle Attention*
SiLU	*Sigmoid Linear Unit*
SPPF	*Spatial Pyramid Pooling - Fast*
SSD	*Single Shot MultiBox Detector*
SURF	*Speeded-Up Robust Features*
SVM	*Support Vector Machines*
YOLO	*You Only Look Once*

SUMÁRIO

1 INTRODUÇÃO

Ao longo dos anos, o cenário industrial e tecnológico testemunhou uma transformação profunda, marcada pela influência de dispositivos e equipamentos eletrônicos em uma ampla gama de setores. Esses dispositivos eletrônicos, que vão desde aparelhos portáteis de comunicação até máquinas industriais complexas, agora constituem uma parte essencial do panorama industrial global.

No núcleo desses sistemas eletrônicos contemporâneos encontra-se um componente fundamental: a Placa de Circuito Impresso (PCBs, do inglês *Printed Circuit Boards*), que fornece a plataforma base para posicionar e interconectar estrategicamente diversos componentes eletrônicos (ZHANG; LIU, 2021). Assim que o processo de colocação dos componentes na PCB é concluído, ela se transforma em uma Montagem de Placa de Circuito Impresso (PCBA, do inglês *Printed Circuit Board Assembly*).

Além disso, à medida que as inovações eletrônicas continuam a se expandir em diferentes indústrias, a fabricação e a garantia de qualidade das unidades de placas de circuito impresso tornaram-se de importância crucial. Garantir a qualidade na fabricação de PCBAs é um desafio complexo devido à ampla variedade de defeitos que podem se manifestar durante o processo de produção (LING; ISA, 2023).

Na busca por assegurar a máxima qualidade e eficiência nos processos de produção, as indústrias têm investido significativamente em sistemas de Inspeção Óptica Automatizada (AOI, do inglês *Automated Optical Inspection*). Esses sistemas implementam técnicas e algoritmos de processamento de imagem para realizar inspeções detalhadas e detectar defeitos de fabricação com alta precisão (KUMAR; KUMAR, 2019). Com o passar do tempo, diversas metodologias e técnicas foram desenvolvidas para ampliar as capacidades dos sistemas de AOI. A abordagem mais amplamente utilizada na indústria é a metodologia de comparação de referência, que compara a PCB inspecionada com uma placa de referência considerada ideal, identificando discrepâncias e defeitos (AKHTAR, 2022).

Com base no avanço significativo em aprendizado profundo proporcionado pela criação da AlexNet em 2012 (KRIZHEVSKY; SUTSKEVER; HINTON, 2012), a integração de Redes Neurais Convolucionais (CNNs, do inglês *Convolutional Neural Networks*), uma categoria de redes neurais artificiais (ANNs, do inglês *Artificial Neural Networks*), na AOI tem apresentado um progresso notável em diversos aspectos. Os modelos de CNN são altamente eficazes na captura de características de defeitos, mesmo em condições desafiadoras, como sombras ou reflexos. Quando treinados em conjuntos de dados diversos que incluem esses artefatos visuais, eles reduzem a necessidade de informações suplementares adicionais, superando métodos tradicionais.

Como resultado, os algoritmos de detecção de objetos baseados em CNN consistentemente estabelecem novos recordes de desempenho em competições e se tornaram a abordagem dominante no campo de detecção de objetos (LING; ISA, 2023). Atualmente, as pesquisas continuam focadas no desafio de aprimorar ainda mais a precisão e exatidão dos sistemas de visão, garantindo sua robustez e confiabilidade em aplicações industriais cada vez mais complexas.

1.1 Justificativa

A constante evolução tecnológica, impulsionada por sua aplicação em diversos setores industriais, tem gerado uma crescente demanda por dispositivos eletrônicos de alta qualidade e confiabilidade. Atender às expectativas do mercado requer não apenas inovações no design de produtos, mas também avanços significativos nos processos de fabricação e inspeção, especialmente para as PCBAs, que desempenham um papel essencial nesses dispositivos.

Contudo, desenvolver metodologias robustas para a inspeção e detecção de defeitos em PCBAs enfrenta desafios consideráveis, devido aos altos custos e às limitações associadas aos métodos tradicionais como, por exemplo, algoritmos de comparação e subtração de imagem de referência para identificar defeitos (LING; ISA, 2023). Esses métodos dependem fortemente de ajustes manuais rígidos, tornando-se menos eficazes em cenários com *layouts* complexos, alta densidade de componentes ou variações sutis nos processos industriais modernos (CHEN; HWANG; HUANG, 2024). Essas limitações tornam o investimento em métodos mais sofisticados, como aqueles baseados em aprendizado profundo, um fator crítico para garantir maior eficiência e precisão na detecção de defeitos em PCBAs.

Adicionalmente, há uma lacuna significativa na literatura científica sobre técnicas de visão computacional para a detecção de defeitos em PCBAs (LING; ISA, 2023). Embora existam diversos estudos voltados para a detecção de defeitos em *layouts* e superfícies de PCBs, poucos trabalhos exploram a identificação de falhas específicas em componentes de PCBAs. Essa lacuna destaca a necessidade de novas abordagens capazes de lidar com a complexidade estrutural e tridimensional dessas montagens.

Diante desse contexto, torna-se evidente a importância do desenvolvimento de estudos e metodologias capazes de atender às crescentes exigências do mercado e da indústria. A implementação de soluções mais avançadas para a detecção de defeitos em PCBAs é fundamental não apenas para superar os desafios técnicos e econômicos apresentados pelos métodos tradicionais, mas também para garantir produtos mais confiáveis e competitivos, alinhados às necessidades de um mercado em constante evolução.

1.2 Definição do Problema

A fabricação de PCBs e sua montagem enfrentam desafios significativos no controle de qualidade devido à crescente complexidade estrutural das placas e à diversidade de componentes eletrônicos utilizados. Defeitos como componentes ausentes, mal posicionados ou com soldas defeituosas podem comprometer o desempenho dos dispositivos eletrônicos e gerar custos elevados de retrabalho ou perdas.

A crescente diversificação de aplicações industriais exigem que os métodos de inspeção de PCBAs sejam altamente adaptáveis. Isso inclui a capacidade de lidar com mudanças frequentes no design das placas, como novos *layouts*, componentes miniaturizados e variações na densidade de interconexões (REN *et al.*, 2022). Além disso, diferentes processos de fabricação e materiais utilizados podem introduzir variabilidades que afetam a detecção de defeitos. Um sistema eficaz deve ser capaz de se ajustar rapidamente a essas alterações sem a necessidade de reconfigurações manuais extensivas ou retrabalho significativo.

Desenvolver um sistema de inspeção que atenda a esses requisitos é uma tarefa desafiadora, especialmente devido à necessidade de garantir alta performance e operação em tempo real. A detecção de defeitos com baixa latência é crucial, pois atrasos no processo podem comprometer a eficiência da linha de produção e aumentar os custos operacionais. Além disso, para atingir altos níveis de precisão e confiabilidade, o sistema deve ser capaz de processar grandes volumes de dados de maneira eficiente, o que frequentemente implica em um custo computacional elevado. Modelos baseados em redes neurais convolucionais demandam *hardware* especializado e otimizações significativas para operar com desempenho ideal, especialmente em aplicações industriais com requisitos de alta densidade e complexidade de componentes (LI *et al.*, 2024).

Dessa forma, é essencial que os sistemas de inspeção de PCBAs sejam rápidos, precisos e adaptáveis a mudanças no *design* e na fabricação. Tecnologias como redes neurais convolucionais oferecem soluções promissoras, mas exigem otimizações para garantir desempenho e reduzir custos computacionais. A superação desses desafios será fundamental para a evolução da fabricação de dispositivos eletrônicos.

1.3 Objetivo

Nestes tópicos serão apresentados o objetivo geral e os objetivos específicos do trabalho.

1.3.1 Objetivo Geral

Desenvolver um modelo eficiente para a inspeção automatizada de defeitos em PCBAs, capaz de identificar falhas em tempo real com alta precisão e adaptabilidade.

1.3.2 Objetivos Específicos

Este trabalho tem como objetivos específicos:

a) elaborar uma fundamentação teórica acerca de aprendizado profundo;

b) analisar o estado da arte das técnicas e metodologias de visão computacional e aprendizado profundo para inspeção de PCBs e PCBAs;

c) projetar uma arquitetura de modelo que combine precisão com eficiência computacional;

d) levantar métricas de avaliação do modelo treinado e compará-las com as do modelo de base;

1.4 Estrutura do Trabalho

O trabalho segue uma estrutura organizada em 7 seções distintas. Na seção inicial, a "Introdução", são apresentados os objetivos e justificativas do estudo. A segunda seção, "Aprendizado Profundo e Redes Neurais", fornece uma fundamentação teórica sobre o assunto, abordando conceitos essenciais e técnicas relevantes. Na terceira seção, "Detecção de Objetos", são discutidos os conceitos fundamentais dessa área, bem como seu estado da arte. A quarta seção, intitulada "Análise da Arquitetura do Modelo Selecionado", oferece uma análise detalhada da arquitetura do modelo escolhido para o estudo. A quinta seção, "Desenvolvimento e Treinamento do Modelo", apresenta o processo de desenvolvimento do modelo proposto. A sexta seção, "Resultados e Discussão", explora os resultados obtidos e promove uma discussão crítica acerca dos mesmos. Finalmente, a sétima seção, "Conclusão", sintetiza as principais implicações do trabalho, além de sugerir possíveis direções para pesquisas futuras.

2 APRENDIZADO PROFUNDO E REDES NEURAIS

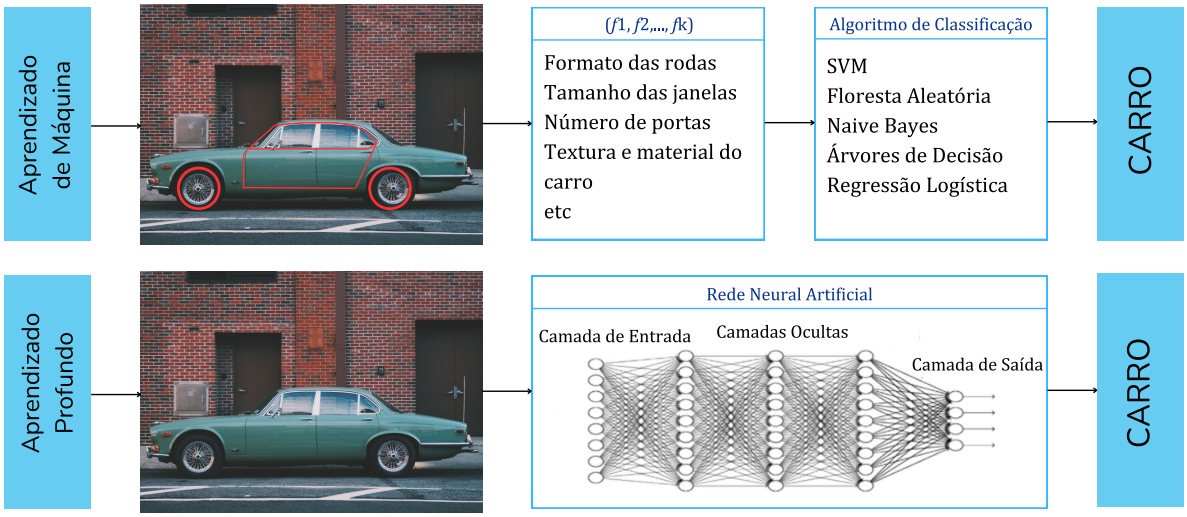
Neste capítulo, serão apresentados os conceitos fundamentais que embasam a compreensão do tema central deste trabalho. Primeiramente, será explorado o conceito de aprendizado profundo, destacando sua relevância no campo de aprendizado de máquina. Em seguida, será introduzido o tema das redes neurais, com ênfase em sua estrutura básica, funcionamento e papel essencial como fundamento do aprendizado profundo. Por fim, será dado um enfoque detalhado às redes neurais convolucionais, ressaltando sua importância para o processamento de imagens e a visão computacional.

2.1 Aprendizado Profundo

O aprendizado profundo (do inglês *Deep Learning*) é uma subárea de aprendizado de máquina que se baseia no uso de redes neurais artificiais para resolver problemas complexos (SARKER, 2021). Inspirada no funcionamento do cérebro humano, essa abordagem busca modelar padrões em dados através de múltiplas camadas de processamento hierárquico, onde cada camada aprende representações progressivamente mais abstratas e relevantes das informações de entrada (GOODFELLOW; BENGIO; COURVILLE, 2016).

Ao contrário dos algoritmos clássicos de aprendizado de máquina, que dependem da definição manual das características relevantes dos dados de entrada, o aprendizado profundo se destaca por sua capacidade de aprender automaticamente essas características durante o processo de treinamento (ROBINS, 2020). Em modelos tradicionais de aprendizado de máquina, o processo de extração de características é um dos principais desafios, pois exige que especialistas forneçam uma definição explícita das características que devem ser usadas para treinar o modelo. No entanto, com o aprendizado profundo, esse processo é automatizado. À medida que o modelo é alimentado com grandes volumes de dados, ele ajusta-se de forma autônoma para identificar padrões, hierarquias de informações e abstrações diretamente a partir dos dados brutos, sem a necessidade de intervenção humana para selecionar manualmente quais atributos são importantes para a tarefa (SHARMA; SHARMA; JINDAL, 2021). Isso permite que os modelos de aprendizado profundo lidem com dados mais complexos, como imagens, áudios e textos, onde as relações entre as características são muitas vezes altamente não lineares e difíceis de capturar por métodos tradicionais (SARKER, 2021). A Figura 1 ilustra de forma clara a diferença entre os algoritmos de aprendizado de máquina convencionais e o aprendizado profundo, destacando como cada abordagem trata os dados e as etapas de extração de características.

Figura 1 – Comparação entre Aprendizado de Máquina e Aprendizado Profundo.



Fonte: Adaptado de ROBINS (2020).

O estudo de técnicas relacionadas ao aprendizado profundo remonta à década de 1940, com os trabalhos pioneiros de McCulloch e Pitts (1943) sobre redes neurais artificiais. Entretanto, durante décadas, a área enfrentou grandes desafios, principalmente devido à limitação do poder computacional, que restringia a capacidade de treinar redes mais complexas (GOODFELLOW; BENGIO; COURVILLE, 2016).

A revolução no aprendizado profundo começou a ganhar força com o avanço da tecnologia computacional, em particular com o uso de unidades de processamento gráfico para acelerar o treinamento de redes neurais, uma ideia popularizada por Raina, Madhavan e Ng (2009). Esse avanço possibilitou a implementação prática de redes neurais artificiais em problemas reais (KUMAR; KUMAR, 2019).

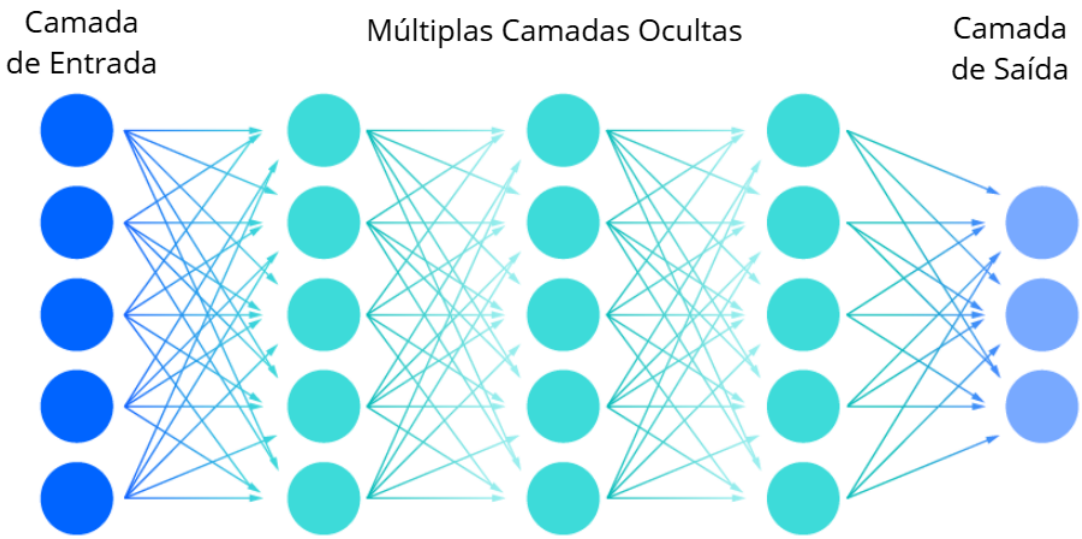
2.2 Redes Neurais Artificiais

As redes neurais artificiais são sistemas computacionais projetados para processar informações de forma semelhante à maneira como os neurônios biológicos operam. Elas consistem em uma coleção de unidades interconectadas, chamadas de neurônios artificiais, organizadas em camadas. Cada neurônio recebe entradas, realiza cálculos com base em pesos e funções de ativação, e transmite os resultados para os neurônios da próxima camada (ZAKARIA; AL-SHEBANY; SARHAN, 2014). Essa estrutura permite que as redes neurais aprendam padrões complexos a partir de dados, tornando-as ferramentas poderosas para resolver problemas de classificação, regressão e reconhecimento de padrões em diversas áreas (TACCHINO *et al.*, 2019).

A estrutura básica de uma rede neural artificial consiste em três tipos de camadas: a camada de entrada, que recebe os dados iniciais; as camadas ocultas, responsáveis por processar as informações e aprender padrões; e a camada de saída,

que gera o resultado final. Cada camada é composta por neurônios artificiais conectados uns aos outros, onde cada conexão possui um peso que é ajustado durante o treinamento. Redes neurais profundas, por sua vez, são uma extensão das redes neurais artificiais e se caracterizam pela presença de múltiplas camadas ocultas, o que lhes permite modelar padrões mais complexos e abstratos (ZAKARIA; AL-SHEBANY; SARHAN, 2014). A Figura 2 apresenta a estrutura de uma rede neural artificial profunda, ilustrando o fluxo de informações entre suas camadas.

Figura 2 – Rede neural artificial profunda.



Fonte: Adaptado de IBM (2023).

2.2.1 Neurônio Artificial

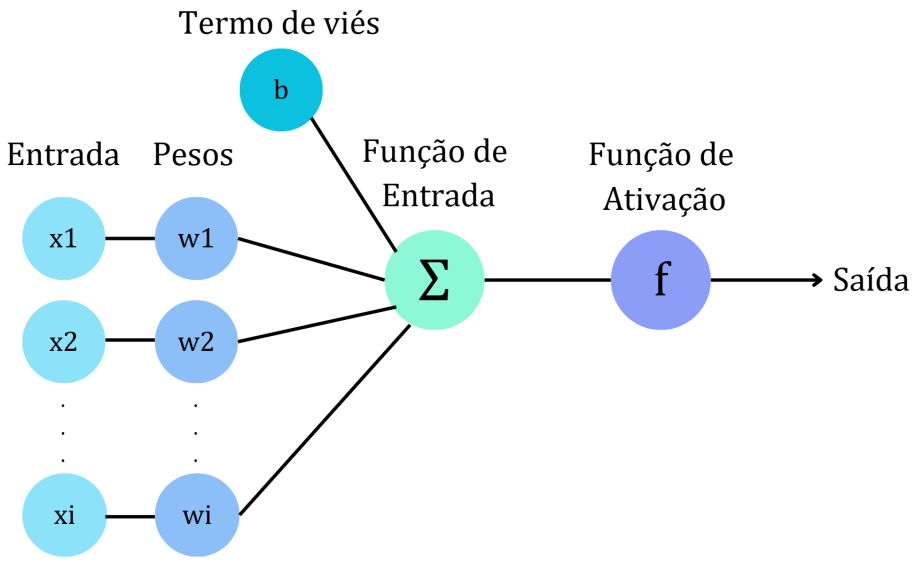
O neurônio artificial, também chamado de nó, mostrado na Figura 3, é a unidade fundamental de uma rede neural artificial. Ele realiza cálculos matemáticos simples para processar as informações recebidas das conexões com outros neurônios. Cada entrada x_i é associada a um peso w_i, que determina sua influência no cálculo. O neurônio calcula uma soma ponderada das entradas, adiciona um termo de viés b, e aplica uma função de ativação f para produzir uma saída y (GHORBANI *et al.*, 2021). Essa relação é descrita pela equação (1).

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{1}$$

Onde:
w_i = peso associado à entrada x_i
x_i = valor da entrada i

b = termo de viés
f = função de ativação aplicada à soma ponderada

Figura 3 – Diagrama de um neurônio artificial.



Fonte: Elaboração própria (2024).

2.2.2 Funções de ativação

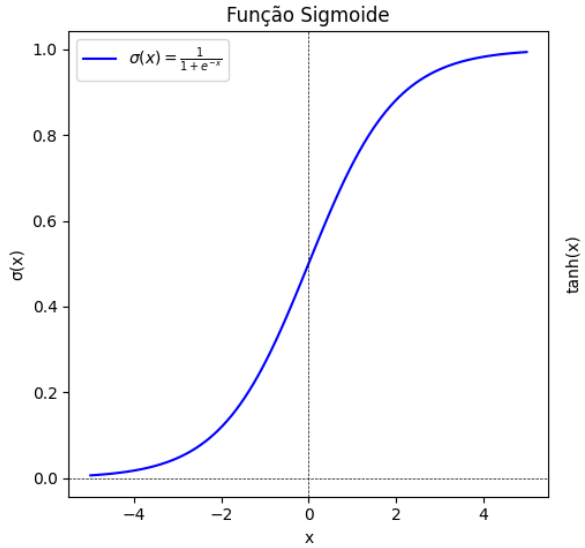
A função de ativação em redes neurais é um componente crucial no processo de aprendizado, responsável por introduzir não-linearidades no modelo (MISRA, 2020). Ela determina a saída de um neurônio, com base em sua entrada ponderada, e permite que a rede neural aprenda e se adapte a dados complexos. Sem uma função de ativação, a rede seria limitada a representar apenas funções lineares, o que reduziria significativamente sua capacidade de resolver problemas mais difíceis. As funções de ativação mais comuns incluem ReLU (do inglês, *Rectified Linear Unit*), sigmoide e tangente hiperbólica (LEDERER, 2021).

A função de ativação sigmoide é uma função não-linear que transforma a entrada em um valor entre 0 e 1, seguindo uma curva em forma de "S", conforme ilustra a Figura 4. Isso a torna útil em tarefas como classificação binária, onde a saída pode ser interpretada como uma probabilidade. Apesar de sua simplicidade e utilidade em camadas finais de redes, a sigmoide pode sofrer com o problema da dissipação do gradiente, especialmente em redes profundas, o que dificulta o aprendizado eficiente (LEDERER, 2021). A função sigmoide pode ser definida conforme a equação (2).

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2}$$

Onde:
x = entrada do neurônio

Figura 4 – Função de ativação Sigmoide.



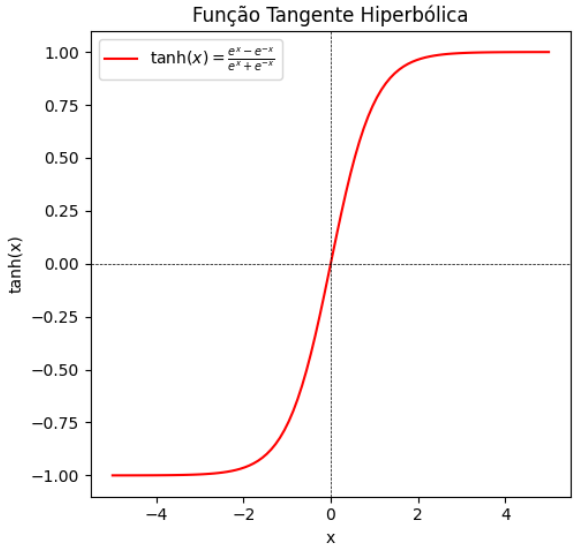
Fonte: Elaboração própria (2024).

Assim como a função sigmoide, a tangente hiperbólica (tanh) é uma função de ativação amplamente utilizada em redes neurais, mas com uma diferença significativa: ela mapeia valores reais para o intervalo (-1, 1), como pode ser verificado na Figura 5. Uma de suas principais vantagens em relação à sigmoide é que suas saídas são centradas em torno de zero, o que reduz o desbalanceamento na propagação de gradientes e acelera a convergência durante o treinamento (DUBEY; SINGH; CHAUDHURI, 2022). Definida pela equação (3), a tanh é uma versão escalada e centrada da sigmoide, o que a torna mais adequada para modelos onde valores negativos têm significado. Essa simetria em torno de zero ajuda a evitar o problema de deslocamento de valores em uma direção específica, facilitando o aprendizado eficiente da rede. No entanto, assim como a sigmoide, a tanh também enfrenta o problema de dissipação do gradiente em valores extremos, o que pode dificultar o aprendizado em redes muito profundas.

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{3}$$

Onde:
x = entrada do neurônio

Figura 5 – Função de ativação Tangente Hiperbólica.



Fonte: Elaboração própria (2024).

A função de ativação Softmax, diferentemente das funções de ativação ponto a ponto, opera sobre um vetor de entradas, transformando seus valores em uma distribuição de probabilidade que garante que a soma das saídas seja igual a 1. Isso permite que cada saída possa ser interpretada como a probabilidade da entrada pertencer a uma determinada classe, como ilustrado na Figura 6. Essa propriedade torna a Softmax a escolha natural para a camada de saída de redes neurais aplicadas a tarefas de classificação, pois possibilita uma decisão baseada nas probabilidades associadas a cada classe (DUBEY; SINGH; CHAUDHURI, 2022).

Definida pela equação (4), a função Softmax realça diferenças entre os valores de entrada ao amplificar as probabilidades da classe com maior ativação e minimizar as demais. No entanto, assim como a sigmoide, ela pode sofrer com o problema da saturação para entradas muito grandes ou pequenas, resultando em gradientes pequenos e dificultando o treinamento em redes profundas. Para mitigar esse problema, técnicas como normalização dos dados de entrada e inicializações adequadas dos pesos são frequentemente utilizadas.

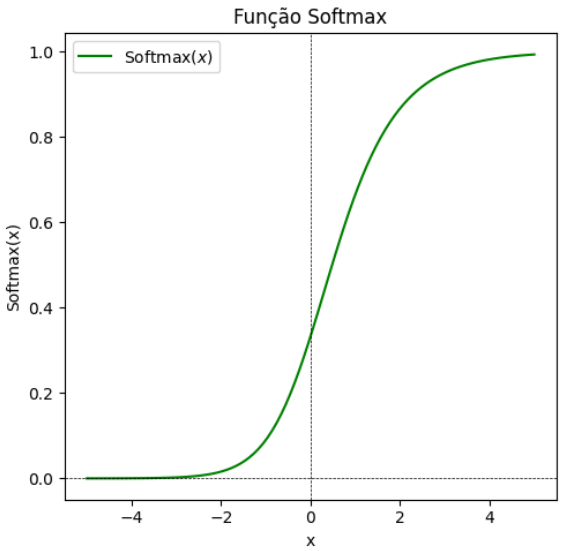
$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \tag{4}$$

Onde:

x_i = entrada do neurônio i

n = número total de neurônios na camada

Figura 6 – Função de ativação Softmax.



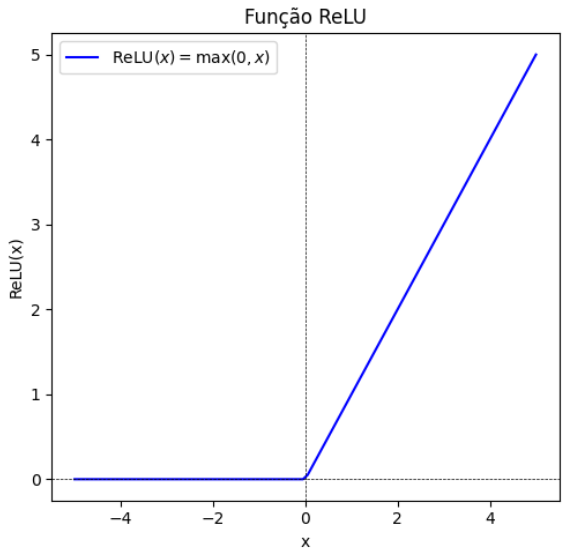
Fonte: Elaboração própria (2024).

A função de ativação ReLU mapeia valores negativos para zero e mantém os positivos, como mostrado na equação (5) e na Figura 7. Isso ajuda a mitigar o problema de dissipação do gradiente, favorecendo o treinamento em redes profundas (DUBEY; SINGH; CHAUDHURI, 2022). No entanto, a ReLU pode causar o problema do "neurônio morto", onde unidades desativadas impedem o aprendizado.

$$\text{ReLU}(x) = \max(0, x) \tag{5}$$

Onde:
x = entrada do neurônio

Figura 7 – Função de ativação ReLU.



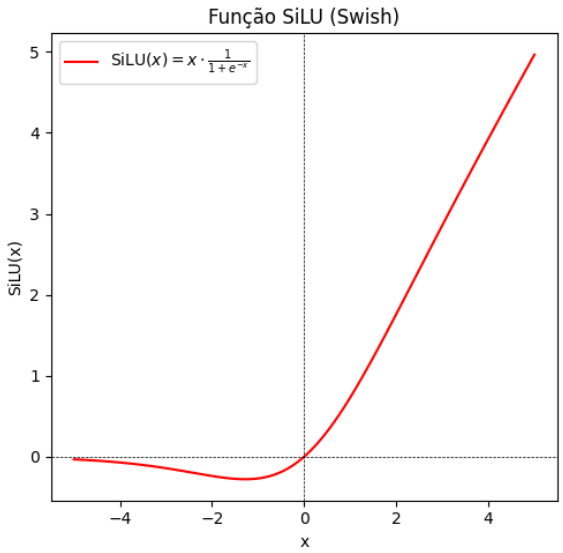
Fonte: Elaboração própria (2024).

Para mitigar o problema da dissipação do gradiente, a função SiLU (do inglês, *Sigmoid Linear Unit*) foi proposta como uma alternativa mais suave à ReLU (HENDRYCKS; GIMPEL, 2016). Diferente da ReLU, a SiLU não zera valores negativos abruptamente, mas os atenua de forma contínua, conforme definido na equação (6) e ilustrado na Figura 8. Essa suavidade permite que pequenos valores negativos ainda contribuam para o aprendizado, reduzindo o problema dos neurônios mortos e melhorando a propagação do gradiente (ELFWING; UCHIBE; DOYA, 2018).

$$\text{SiLU}(x) = x \cdot \sigma(x) = x \cdot \frac{1}{1 + e^{-x}} \tag{6}$$

Onde:
x = entrada do neurônio

Figura 8 – Função de ativação SiLU.



Fonte: Elaboração própria (2024).

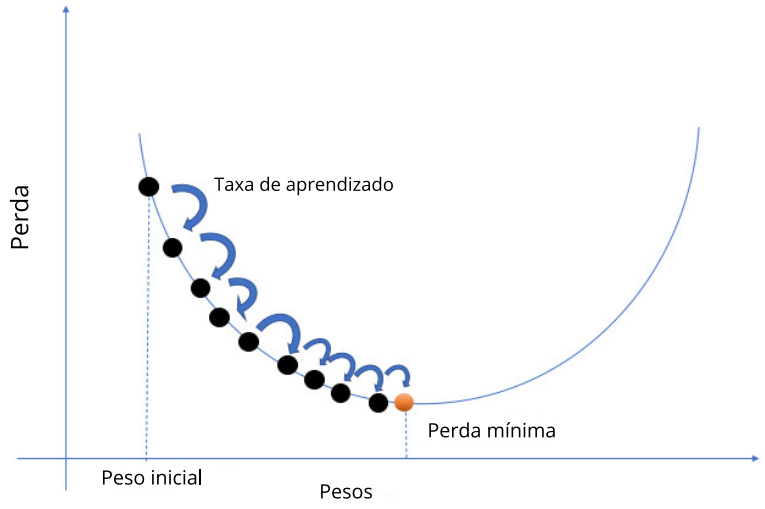
2.3 Treinamento de Redes Neurais

O treinamento de redes neurais é um processo fundamental para que o modelo aprenda a extrair padrões e realizar previsões com precisão. Esse processo consiste na otimização dos pesos dos neurônios e vieses por meio de algoritmos de aprendizado, sendo o mais comum o *backpropagation* combinado com métodos de gradiente estocástico (SGD, do inglês *stochastic gradient descent*) (JAISWAL; GUPTA; AMBIKAPATHY, 2018). Durante o treinamento, os parâmetros da rede são ajustados iterativamente para minimizar uma função de perda, que mede o erro entre as previsões do modelo e os valores esperados (GOODFELLOW; BENGIO; COURVILLE, 2016).

As funções de perda quantificam o erro entre as previsões do modelo e os valores reais esperados (GOODFELLOW; BENGIO; COURVILLE, 2016). A escolha da função de perda depende do tipo de problema a ser resolvido. Em tarefas de classificação, é comum o uso da entropia cruzada, que mede a discrepância entre as probabilidades previstas e as verdadeiras (MURPHY, 2012). Já em problemas de regressão, a função de erro quadrático médio é amplamente utilizada, pois penaliza desvios maiores entre a previsão e o valor real (BISHOP, 2006). O cálculo da função de perda orienta o processo de ajuste dos pesos da rede, permitindo que o algoritmo de otimização atualize os parâmetros de forma a reduzir gradativamente o erro ao longo das iterações do treinamento.

O processo de retropropagação é fundamental para garantir que o modelo aprenda de maneira eficiente (RUMELHART; HINTON; WILLIAMS, 1986). Esse algoritmo calcula o gradiente da função de perda em relação aos parâmetros da rede por meio da regra da cadeia, propagando os erros da camada de saída até as camadas iniciais (GOODFELLOW; BENGIO; COURVILLE, 2016). A atualização dos pesos é realizada por um otimizador, como o método SGD ou variantes mais avançadas, como Adam, que ajustam dinamicamente a taxa de aprendizado para melhorar a convergência (GHOSH *et al.*, 2020), conforme ilustra a Figura 9. Dessa forma, a retropropagação permite que a rede ajuste seus pesos de maneira iterativa, minimizando a função de perda e aprimorando a capacidade do modelo de realizar previsões precisas.

Figura 9 – Método do gradiente.



Fonte: Adaptado de GHOSH *et al.* (2020).

2.4 Redes Neurais Convolucionais

As Redes Neurais Convolucionais revolucionaram o campo da visão computacional ao introduzir camadas especializadas capazes de extrair automaticamente

padrões espaciais de imagens (LECUN *et al.*, 1989). Inspiradas no funcionamento do córtex visual humano, as CNNs utilizam operações de convolução para detectar características como bordas, texturas e formas em diferentes níveis de abstração. Essa capacidade de aprendizado hierárquico tornou as CNNs a base de diversos avanços em tarefas como classificação, segmentação e detecção de objetos (KRIZHEVSKY; SUTSKEVER; HINTON, 2012). Diferentemente das redes neurais tradicionais, que tratam os dados como um vetor plano, as CNNs preservam a estrutura espacial das imagens, permitindo um processamento mais robusto (ZHAO *et al.*, 2024).

Dessa forma, as CNNs são compostas por camadas de convolução, responsáveis por aplicar filtros para extração de características, e camadas de pooling, que reduzem a dimensionalidade dos mapas de ativação, tornando o modelo mais eficiente (ZHAO *et al.*, 2024).

2.4.1 Camada de convolução

A convolução é uma operação matemática fundamental em redes neurais convolucionais que pode ser definida como a integral do produto entre duas funções, que no contexto das CNNs, são a função de entrada (geralmente uma imagem ou uma matriz) e o filtro (BHATT *et al.*, 2021). A equação da convolução em uma única dimensão é dada pela equação (7).

$$y(t) = \int_{-\infty}^{\infty} x(\tau) \cdot h(t - \tau) d\tau \tag{7}$$

Onde:
$y(t)$ = resultado da convolução no instante t
$x(\tau)$ = função de entrada
$h(t - \tau)$ = função filtro que define a transformação aplicada à entrada
τ = variável de integração

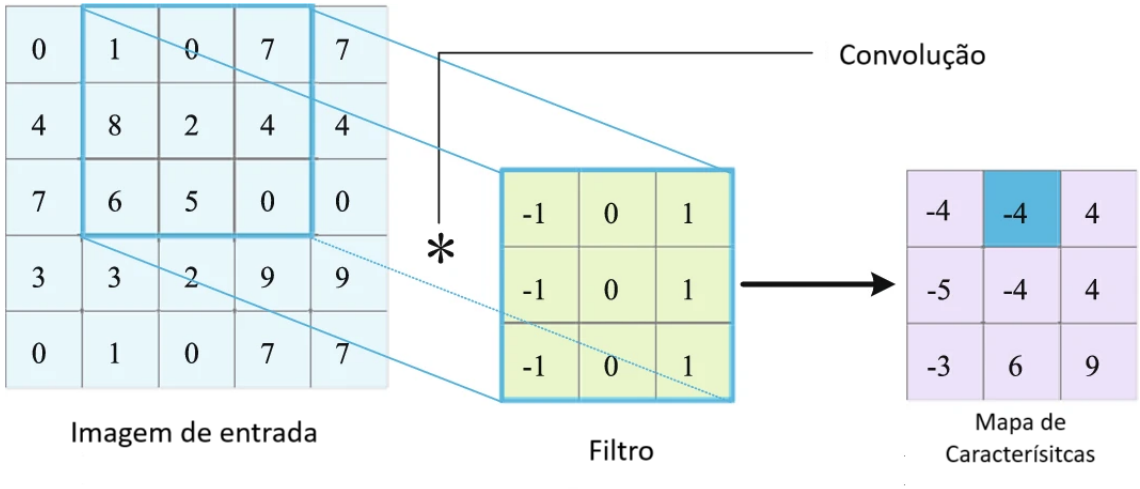
No contexto das CNNs, a operação de convolução é aplicada de forma discreta e bidimensional, diferindo da definição contínua utilizada no processamento de sinais, conforme apresentado na equação (8) (GOODFELLOW; BENGIO; COURVILLE, 2016). Essa operação consiste em deslocar um filtro h(m,n) sobre a entrada x(i,j) e calcular a soma ponderada dos valores sobrepostos, gerando um mapa de características, conforme ilustra a Figura 10.

$$y(i, j) = \sum_{m} \sum_{n} x(i - m, j - n) \cdot h(m, n) \tag{8}$$

Onde:
$y(i, j)$ = resultado da convolução na posição (i, j)

$x(i-m, j-n)$ = valor da entrada deslocado pelas coordenadas (m, n)
$h(m, n)$ = valor do filtro (ou *kernel*) na posição (m, n)
m, n = índices que percorrem o filtro ao longo da entrada

Figura 10 – Operação de convolução.



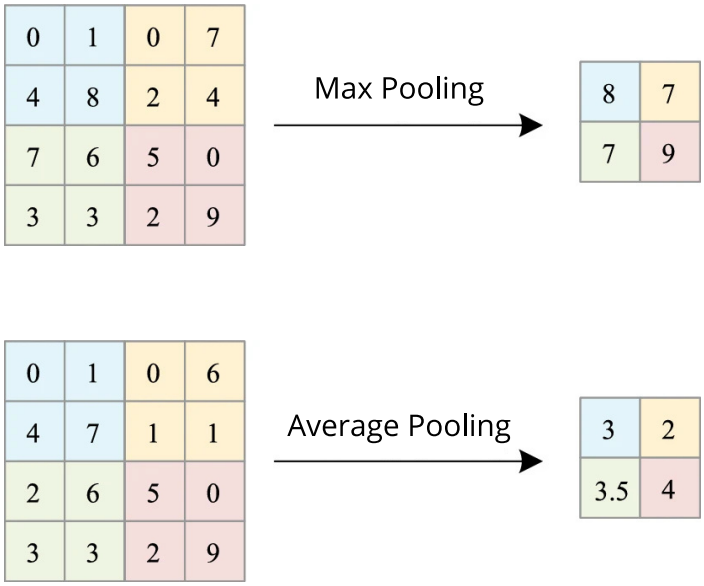
Fonte: Adaptado de ZHAO *et al.* (2024).

2.4.2 Camada de *pooling*

As camadas de *pooling* são projetadas para reduzir a dimensionalidade dos mapas de ativação enquanto preservam as informações mais relevantes (BHATT *et al.*, 2021). Essas camadas operam aplicando uma função de agregação sobre pequenas regiões da entrada, diminuindo a sensibilidade a pequenas variações espaciais e melhorando a eficiência computacional do modelo (GOODFELLOW; BENGIO; COURVILLE, 2016).

Os dois tipos mais comuns de *pooling* são o *max pooling*, que seleciona o maior valor dentro da região considerada, e o *average pooling*, que calcula a média dos valores, como ilustrado na Figura 11. O *max pooling* é frequentemente utilizado para extrair as características mais importantes e invariantes, proporcionando uma redução de dimensionalidade mais eficiente. Já o *average pooling* tende a suavizar as informações, o que pode ser útil em algumas situações onde a generalização é mais importante que a precisão de características específicas (ZHAO *et al.*, 2024).

Figura 11 – Operação de *pooling*.



Fonte: Adaptado de ZHAO *et al.* (2024).

2.4.3 Camadas totalmente conectadas

As camadas totalmente conectadas, ou *fully connected layers*, são operações globais dentro das CNNs, em contraste com as camadas de convolução e *pooling*, que operam localmente (ZHAO *et al.*, 2024). Essas camadas são geralmente utilizadas na parte final da rede, onde os mapas de características extraídos ao longo das camadas anteriores são transformados em um vetor unidimensional e passados para a camada totalmente conectada. Nessa estrutura, cada neurônio está conectado a todos os neurônios da camada anterior. Esse mecanismo permite que a rede combine as informações extraídas das diferentes regiões da imagem e tome decisões de classificação.

2.5 Mecanismos de Atenção

Nos últimos anos, os mecanismos de atenção têm desempenhado um papel fundamental no avanço do aprendizado profundo, permitindo que redes neurais processem informações de forma mais eficiente e seletiva. Originalmente introduzidos no contexto de tradução automática, esses mecanismos foram amplamente adotados em diversas áreas, incluindo visão computacional, processamento de linguagem natural e bioinformática (VASWANI *et al.*, 2017).

A atenção funciona atribuindo pesos diferentes às partes mais relevantes de uma entrada, permitindo que o modelo foque em informações essenciais enquanto ignora detalhes irrelevantes. Essa abordagem melhora significativamente a capacidade das redes neurais de capturar dependências de longo alcance e modelar rela-

ções complexas nos dados. Na visão computacional, os mecanismos de atenção são utilizados para aprimorar a extração de características em tarefas como segmentação, detecção de objetos e reconhecimento de padrões (HU *et al.*, 2020).

O mecanismo de atenção pode ser descrito matematicamente como uma função que destaca regiões discriminativas em uma entrada e processa essas regiões de forma mais eficiente. De acordo com Guo *et al.* (2022), essa ideia pode ser formalmente expressa pela equação (9).

$$\text{Atenção} = f(g(x), x) \tag{9}$$

Onde:
x = entrada do modelo, representando a informação bruta
$g(x)$ = função que gera os pesos de atenção, destacando regiões discriminativas
$f(g(x), x)$ = função que processa a entrada x baseada nos pesos de atenção $g(x)$

Nesta formulação, a função g(x) é o mecanismo que aprende a importância relativa das diferentes partes da entrada x, enquanto f(g(x),x) representa a aplicação desses pesos ao processamento da entrada, enfatizando as regiões mais relevantes. Essa definição é bastante abrangente e pode ser usada para descrever diversos tipos de mecanismos de atenção existentes na literatura, desde os baseados em pesos estáticos até os modelos dinâmicos empregados em redes neurais profundas.

2.5.1 Atenção por Canal

Os canais representam diferentes conjuntos de características extraídas de uma imagem ao longo das camadas da rede. Em uma imagem de entrada em escala de cinza, por exemplo, há apenas um canal, enquanto imagens coloridas possuem três canais (vermelho, verde e azul) (LECUN *et al.*, 1989). À medida que a imagem passa pelas camadas convolucionais, novas representações são aprendidas e armazenadas em múltiplos canais, cada um capturando aspectos distintos, como bordas, texturas ou padrões mais abstratos. Esses canais formam um volume tridimensional de dados, onde cada canal pode ser interpretado como uma matriz que contém informações específicas sobre diferentes propriedades da imagem (HU *et al.*, 2020).

Dessa forma, a atenção por canal, ou *channel attention*, é um mecanismo de atenção que atua nos canais de um mapa de características em redes neurais profundas, atribuindo pesos diferentes a cada canal para destacar informações mais relevantes (HU *et al.*, 2020). Em vez de considerar apenas a localização espacial das características, a atenção por canal foca na importância relativa de cada canal, permitindo que a rede aprenda quais aspectos das características extraídas são mais informativos para a tarefa em questão. Esse mecanismo pode ser interpretado como

um processo de seleção de objetos, onde a rede decide automaticamente o que deve receber mais atenção, recalibrando os pesos de cada canal de forma adaptativa (GUO *et al.*, 2022).

Esse mecanismo é especialmente útil em CNNs, onde diferentes canais podem capturar aspectos distintos das imagens de entrada, como texturas, bordas ou padrões específicos. A atenção por canal funciona aplicando operações de agregação global, como *Global Average Pooling* (GAP) ou *Global Max Pooling* (GMP), seguidas por camadas totalmente conectadas ou convoluções para calcular coeficientes de atenção que realçam ou suprimem determinados canais (WOO *et al.*, 2018).

2.5.2 Atenção Espacial

A atenção espacial, ou *spatial attention*, é um mecanismo de atenção que opera no domínio espacial das características extraídas por uma rede neural convolucional, enfatizando regiões da imagem que contêm informações mais relevantes para a tarefa em questão (WOO *et al.*, 2018). Em vez de ajustar a importância relativa dos canais, como ocorre na atenção por canal, a atenção espacial aprende a identificar quais partes da imagem devem receber maior peso durante o processamento, permitindo que a rede foque em áreas discriminativas (GUO *et al.*, 2022). Esse mecanismo melhora a capacidade da rede de capturar relações espaciais entre os elementos da cena, o que é particularmente útil em tarefas como detecção de objetos e segmentação semântica (HU *et al.*, 2020).

Além de destacar regiões importantes da imagem, a atenção espacial pode ser implementado de diversas formas, como por meio de mapas de atenção gerados a partir de operações convolucionais ou funções de agregação de características (WOO *et al.*, 2018). Uma abordagem comum envolve a combinação das informações de diferentes canais para calcular um mapa de atenção espacial, que é então aplicado à entrada original para reforçar as regiões mais relevantes (GUO *et al.*, 2022). Esse mecanismo é especialmente útil em redes neurais convolucionais, pois complementa a atenção por canal, permitindo que a rede aprenda tanto quais características são importantes (no nível dos canais) quanto onde essas características estão localizadas na imagem (GUO *et al.*, 2022). Ao combinar ambos os mecanismos, os modelos podem melhorar significativamente o desempenho em tarefas de visão computacional, como segmentação e detecção de objetos (HU *et al.*, 2020).

2.5.3 Módulo de Atenção em Bloco Convolucional

O Módulo de Atenção em Bloco Convolucional – *Convolutional Block Attention Module* (CBAM) é um mecanismo de atenção que combina atenção por canal e atenção espacial para aprimorar a extração de características em redes neurais con-

volucionais (WOO *et al.*, 2018). O CBAM opera em duas etapas sequenciais: primeiro, aplica atenção por canal para identificar quais canais contêm informações mais relevantes para a tarefa, e em seguida, utiliza atenção espacial para localizar as regiões espaciais mais importantes na imagem (GUO *et al.*, 2022). Essa arquitetura modular permite que o CBAM seja integrado de maneira simples a arquiteturas convolucionais existentes, melhorando a representatividade dos mapas de características.

O CBAM pode ser descrito como um módulo composto por duas operações sequenciais: a operação de atenção por canal e a espacial, como apresenta a Figura 12. Conforme definido por Woo *et al.* (2018), dado um mapa de características intermediário $\mathbf{F} \in \mathbb{R}^{C \times H \times W}$, onde **C**, **H** e **W** representam, respectivamente, o número de canais, a altura e a largura da entrada, o CBAM aprimora esse mapa aplicando sucessivamente os módulos de atenção.

Primeiramente, a atenção por canal gera um mapa de atenção a partir de agregações globais dos valores espaciais utilizando operações de média e máximo, conforme a equação (10).

$$\mathbf{M}_c(\mathbf{F}) = \sigma(\text{MLP}(\text{AvgPool}(\mathbf{F})) + \text{MLP}(\text{MaxPool}(\mathbf{F}))) \tag{10}$$

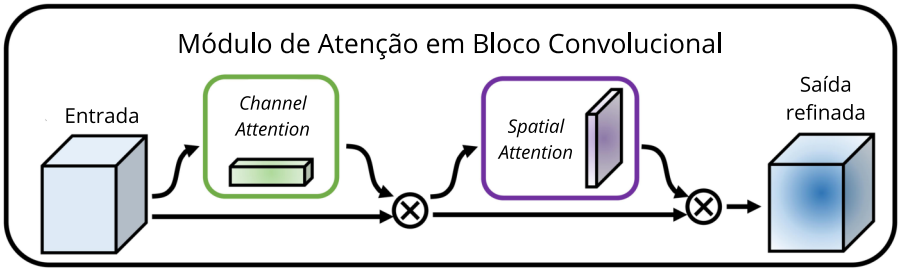
Onde σ representa a função sigmoide e MLP é uma rede neural de múltiplas camadas aplicada sobre os vetores gerados pela média global (*AvgPool*) e máximo global (*MaxPool*) dos canais. O mapa resultante $\mathbf{M_c}$ é multiplicado elemento a elemento com **F** para realçar os canais mais relevantes.

Em seguida, o mecanismo de atenção espacial é aplicado ao mapa de características modificado $\mathbf{F}' = \mathbf{M_c} \odot \mathbf{F}$, sendo $\odot$ o produto Hadamard (multiplicação elemento a elemento). Esse módulo é definido conforme a equação (11).

$$\mathbf{M_s}(\mathbf{F}') = \sigma(\text{Conv}_{7\times7}([\text{AvgPool}(\mathbf{F}'); \text{MaxPool}(\mathbf{F}')])) \tag{11}$$

Onde $\text{Conv}_{7\times7}$ representa uma convolução com um filtro de 7×7 aplicada a uma concatenação ($[\cdot;\cdot]$) dos mapas gerados por média e máximo ao longo do eixo dos canais. O mapa resultante $\mathbf{M_s}$ é então multiplicado por $\mathbf{F}'$, produzindo a saída final $\mathbf{F}'' = \mathbf{M_s} \odot \mathbf{F}'$, que contém características refinadas espacial e semanticamente.

Figura 12 – Diagrama do Módulo de Atenção em Bloco Convolucional.



Fonte: Adaptado de WOO *et al.* (2018).

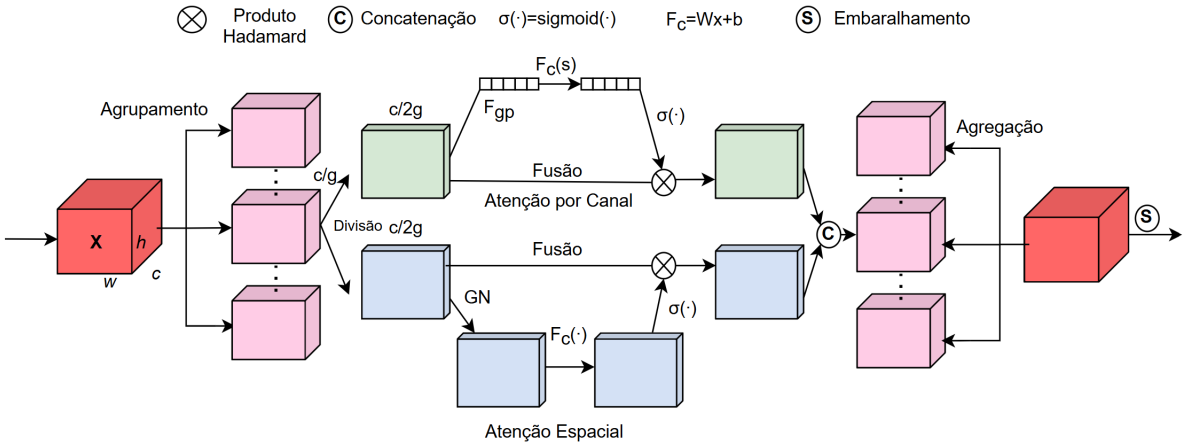
Segundo Zhang e Yang (2021), embora o CBAM integre de maneira eficiente e simplificada os mecanismos de atenção espacial e por canal, aprimorando a extração de características relevantes, ele ainda apresenta certas limitações. Em particular, o CBAM pode sofrer com dificuldades de convergência durante o treinamento, o que pode impactar a estabilidade do modelo e exigir ajustes cuidadosos nos hiperparâmetros. Além disso, a introdução de operações adicionais, como múltiplas convoluções e transformações não lineares, pode aumentar significativamente a complexidade computacional, tornando o método menos eficiente para aplicações que requerem baixo custo computacional.

2.5.4 Atenção com Embaralhamento

A atenção com embaralhamento, popularmente conhecida como *Shuffle Attention* (SA), é um mecanismo de atenção projetado para melhorar a eficiência e a expressividade da modelagem de dependências em redes neurais profundas (ZHANG; YANG, 2021). Diferente de abordagens convencionais, que aplicam a atenção de forma independente nos domínios espacial e de canal, o SA introduz um processo de embaralhamento para capturar interdependências entre canais de maneira mais eficaz. Essa estratégia permite que a rede distribua informações relevantes entre diferentes canais sem aumentar significativamente a complexidade computacional, tornando-se uma solução atrativa para aplicações em visão computacional que exigem um equilíbrio entre desempenho e eficiência.

O SA pode ser dividido em quatro etapas principais: Propriedade de Agrupamento, Atenção por Canal, Atenção Espacial e Agregação (ZHANG; YANG, 2021). Essas etapas permitem que a atenção seja aplicada de forma eficiente, distribuindo informações entre os canais e garantindo um processamento mais dinâmico e adaptativo das características extraídas. A Figura 13 ilustra o fluxo do mecanismo de SA.

Figura 13 – Diagrama do Módulo de Atenção com Embaralhamento.



Fonte: Adaptado de ZHANG; YANG (2021).

A abordagem de agrupamento de características no SA segue uma estrutura hierárquica de dois níveis. Considerando que o tensor de entrada do módulo de atenção seja $X \in \mathbb{R}^{C \times H \times W}$, onde C representa o número de canais e $H \times W$ as dimensões espaciais do mapa de características. O primeiro estágio do SA separa X em G grupos ao longo da dimensão dos canais, gerando subconjuntos de características representados por $\tilde{X} \in \mathbb{R}^{\frac{C}{G} \times H \times W}$ (ZHANG; YANG, 2021).

Esses grupos seguem para os módulos de atenção, onde são novamente divididos em dois subconjuntos ao longo da dimensão dos canais. Um desses subconjuntos é destinado a atenção espacial, enquanto o outro processa a atenção por canal. Dessa forma, cada subconjunto pode ser descrito como $\hat{X} \in \mathbb{R}^{\frac{C}{2G} \times H \times W}$.

Para a etapa de anteção por canal, o SA inicialmente aplica uma operação de GAP sobre $\hat{X}$, reduzindo-o a um tensor com dimensões $\frac{C}{2G} \times 1 \times 1$. Em seguida,a saída do GAP, denotada como s, é processada por uma camada totalmente conectada $F_c(s)$, responsável por aprender uma transformação linear dos coeficientes de atenção, conforme descrito na equação (12).

$$X'_{k1} = \sigma(F_c(s)) \cdot \hat{X} = \sigma(W_1 s + b_1) \cdot \hat{X} \tag{12}$$

Onde:
W_1 = matriz de pesos da rede neural
b_1 = o vetor de viés da rede neural

Na etapa de atenção espacial, a entrada $\hat{X}$ é reduzida por meio da Normalização em Grupo – *Group Normalization* (GN) para obter estatísticas espacialmente relevantes. Em seguida, $F_c(\cdot)$ é utilizado para aprimorar a representação do tensor reduzido (ZHANG; YANG, 2021). Esse processo é descrito pela equação (13).

$$X'_{k2} = \sigma(W_2 \cdot GN(\hat{X}) + b_2) \cdot \hat{X} \tag{13}$$

Onde:
W_2 = matriz de pesos da rede neural
b_2 = vetor de viés da rede neural

Por fim, as saídas das ramificações de atenção espacial e atenção ao canal são concatenadas. Após essa concatenação, uma estratégia de embaralhamento de canais é adotada para permitir o fluxo de informações entre os diferentes grupos de canais. Isso assegura que a saída final mantenha a mesma dimensão do tensor de entrada ao módulo SA.

Com essa abordagem, o módulo de anteção com embaralhamento é capaz de integrar de maneira eficiente as atenções espaciais e de canal, proporcionando uma modelagem mais rica das relações entre características sem adicionar um grande custo computacional (ZHANG; YANG, 2021).

2.6 Métricas de Avaliação

As métricas de avaliação são essenciais para quantificar o desempenho de modelos e algoritmos. Elas fornecem uma base objetiva para comparar diferentes abordagens e entender a eficácia das soluções propostas. Nesta seção, abordaremos as principais métricas utilizadas e suas aplicações específicas no contexto do estudo.

2.6.1 Precisão e Sensibilidade

A precisão e a sensibilidade são métricas essenciais para avaliar o desempenho do modelo de detecção de objetos. A precisão indica a proporção de predições corretas entre todas as predições positivas feitas pelo modelo, sendo definida conforme equação (14) (GOUTTE; GAUSSIER, 2005).

$$\text{Precisão} = \frac{VP}{VP + FP} \tag{14}$$

Onde VP representa os verdadeiros positivos e FP os falsos positivos. Uma alta precisão indica que poucos falsos positivos foram identificados pelo modelo.

A sensibilidade, também conhecida como *recall*, por outro lado, mede a capacidade do modelo de identificar corretamente todas as instâncias positivas, sendo expressa conforme equação (15) (GOUTTE; GAUSSIER, 2005).

$$\text{Sensibilidade} = \frac{VP}{VP + FN} \tag{15}$$

Onde *FN* são os falsos negativos. Um alto valor de sensibilidade significa que a maioria dos casos positivos foram corretamente detectados pelo modelo.

2.6.2 F-*Score*

O balanceamento entre essas duas métricas é crucial, pois um aumento na precisão pode resultar na redução da sensibilidade e vice-versa. Para avaliar esse compromisso, utiliza-se a métrica F-*Score*, que é a média harmônica entre a precisão e a sensibilidade (GOUTTE; GAUSSIER, 2005). A métrica é definida conforme a equação (16).

$$F\text{-score} = 2 \times \frac{\text{Precisão} \times \text{Sensibilidade}}{\text{Precisão} + \text{Sensibilidade}} \tag{16}$$

Essa métrica fornece uma avaliação equilibrada do desempenho do modelo, especialmente em cenários onde existe um desequilíbrio entre as classes.

2.6.3 Precisão Média

A precisão média – *Average Precision* (AP) é uma métrica utilizada na avaliação de modelos de detecção de objetos e classificação de imagens. Ela é calculada a partir da curva de Precisão-Sensibilidade, integrando a precisão em diferentes níveis de sensibilidade. Essencialmente, a AP mede a área sob essa curva, proporcionando uma única pontuação que reflete a capacidade do modelo de equilibrar precisão e revocação ao longo de diferentes limiares de confiança (ZAIDI *et al.*, 2022).

Dessa forma, a média desses valores, denominada *mean Average Precision* (mAP), é calculada para todas as classes em um conjunto de dados. Ele fornece uma visão global do desempenho do modelo, considerando sua capacidade de detectar corretamente múltiplas classes de objetos (KOIRALA *et al.*, 2019).

O mAP pode ser calculado de diferentes formas, dependendo do critério utilizado para considerar uma detecção correta. O mAP@50 avalia o modelo considerando um único limite fixo de sobreposição entre a predição e o objeto real na imagem. Se a predição atingir ou superar esse limite, ela é considerada correta. Já o mAP@50-95 adota uma abordagem mais rigorosa, calculando a precisão média em diversos níveis de exigência, desde um limite mínimo até um limite muito mais restritivo. Isso significa que o mAP@50-95 oferece uma visão mais completa do desempenho do modelo, penalizando detecções imprecisas e favorecendo aqueles que conseguem localizar os objetos com maior exatidão em diferentes situações.

3 DETECÇÃO DE OBJETOS

A detecção de objetos é uma tarefa fundamental em visão computacional que consiste na identificação e localização de instâncias de objetos de interesse dentro de uma imagem ou vídeo. Diferente da classificação de imagens, que apenas atribui um rótulo à imagem como um todo, a detecção envolve a previsão das coordenadas dos objetos detectados juntamente com suas respectivas classes (ZHAO *et al.*, 2024). Essa tarefa é amplamente utilizada em diversas aplicações, como veículos autônomos, videomonitoramento, diagnósticos médicos e sistemas de inspeção industrial (ZOU *et al.*, 2023).

O processo de detecção de objetos geralmente envolve duas etapas principais: a localização dos objetos e a sua categorização (ZHAO *et al.*, 2024). Inicialmente, o sistema identifica regiões de interesse na imagem onde os objetos possivelmente estão posicionados. Esse passo pode ser realizado por meio de técnicas como a extração de propostas de regiões ou pelo uso de redes neurais que realizam previsões diretas de caixas delimitadoras, ou *bounding boxes*. Em seguida, os objetos detectados nessas regiões são classificados em diferentes categorias com base em características extraídas, permitindo que o sistema não apenas localize, mas também reconheça os diferentes elementos presentes na cena.

Antes do avanço das redes neurais profundas, a detecção de objetos na visão computacional era baseada em abordagens clássicas que utilizavam técnicas de extração manual de características combinadas com classificadores tradicionais. Métodos como o Histograma de Gradientes Orientados – *Histograms of Oriented Gradients* (HOG) (DALAL; TRIGGS, 2005) e Características Baseadas em *Haar Wavelets* (VIOLA; JONES, 2001) eram amplamente utilizados para representar padrões visuais presentes nas imagens. Após a extração dessas características, algoritmos como *Support Vector Machines* (SVM) ou *Viola-Jones* eram empregados para classificar e detectar objetos. Além disso, técnicas baseadas em descritivos de imagens, como *Speeded-Up Robust Features* (SURF) (BAY *et al.*, 2008), permitiam identificar e rastrear objetos mesmo sob variações de escala e rotação. Apesar de serem eficazes em cenários controlados, esses métodos apresentavam limitações em relação à robustez e generalização para diferentes condições de iluminação, perspectiva e oclusão, o que motivou a transição para abordagens baseadas em aprendizado profundo.

Com o avanço das redes neurais profundas, a detecção de objetos passou a se basear em arquiteturas mais sofisticadas que extraem automaticamente características em múltiplos níveis de abstração por meio de camadas convolucionais, capturando desde padrões de bordas até representações semânticas complexas (REDMON *et al.*, 2016). Atualmente, os modelos de detecção de objetos são amplamente cate-

gorizados em duas classes principais: modelos de estágio único e modelos de dois estágios.

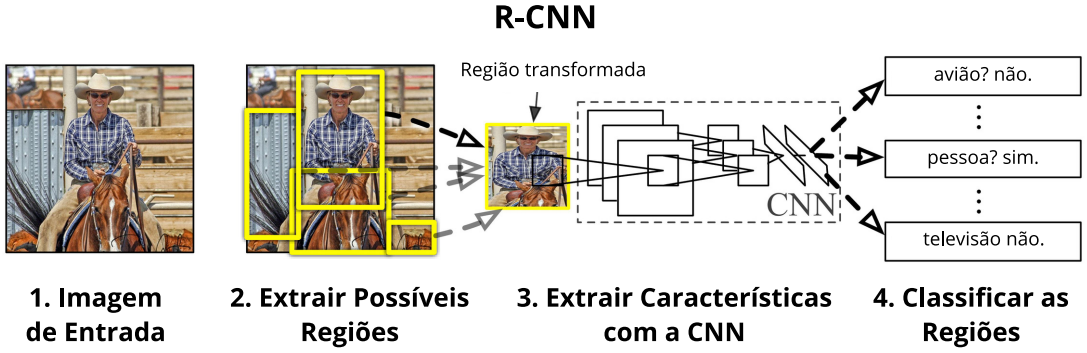
3.1 Abordagens em Dois e Um Estágio

Após a introdução das duas categorias principais na seção anterior, focamos aqui nos aspectos distintivos e nas inovações que cada abordagem aporta à detecção de objetos.

3.1.1 Detecção de Dois Estágios

As abordagens de dois estágios dividem o processo de detecção em duas fases distintas. Primeiramente, são geradas propostas de regiões que podem conter objetos, e em seguida, essas propostas são refinadas e classificadas. Um exemplo clássico desse tipo de abordagem é o R-CNN (do inglês, *Regions with Convolutional Neural Networks*) (GIRSHICK *et al.*, 2014), que inicialmente gera diversas propostas de regiões usando métodos como *Selective Search* e, em seguida, aplica uma rede neural convolucional a cada região para classificação e ajuste fino das caixas delimitadoras, conforme apresenta a Figura 14.

Figura 14 – Mecanismo de detecção do modelo R-CNN.



Fonte: Adaptado de GIRSHICK *et al.* (2014).

Modelos subsequentes, como *Fast* R-CNN (GIRSHICK, 2015) e *Faster* R-CNN (REN *et al.*, 2015), melhoraram a eficiência ao introduzir a computação compartilhada de características e a Rede de Propostas Regionais, respectivamente.

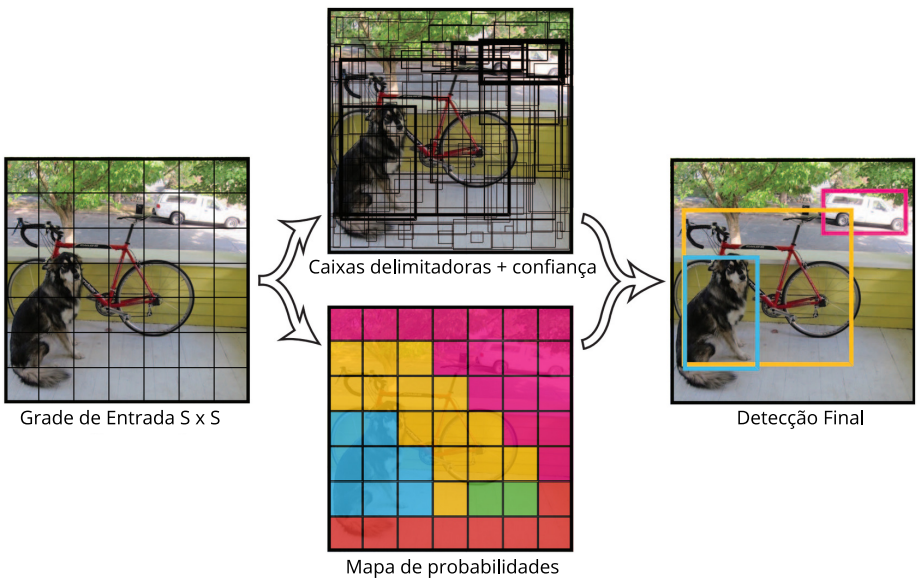
Apesar das melhorias na precisão, as abordagens de dois estágios, apresentam algumas desvantagens significativas. Um dos principais problemas está no alto custo computacional, pois essas arquiteturas exigem múltiplas etapas para a detecção, incluindo a geração de propostas de regiões e a posterior classificação e refinamento das previsões (LING; ISA, 2023). Esse processo torna o modelo consideravelmente mais lento, dificultando sua aplicação em cenários de tempo real.

3.1.2 Detecção de Um Estágio

Por outro lado, as abordagens de detecção de um estágio integram as tarefas de extração de características, proposta de região e classificação em um único fluxo de trabalho, resultando em um modelo que prediz diretamente os objetos nas imagens de entrada. Uma das arquiteturas mais conhecidas desta categoria é a *Single Shot MultiBox Detector* (SSD) (LIU *et al.*, 2016), que divide a imagem em uma grade e atribui caixas delimitadoras e classes de objetos diretamente a cada célula da grade, permitindo uma detecção rápida e eficiente.

Outra abordagem influente é a *You Only Look Once* (YOLO) (REDMON *et al.*, 2016), que trata a detecção de objetos como um problema de regressão única, reduzindo significativamente o tempo de processamento. A YOLO divide a imagem em uma grade uniforme e, para cada célula dessa grade, gera múltiplas caixas delimitadoras associadas a pontuações de confiança e probabilidades de classe (REDMON *et al.*, 2016). O modelo utiliza uma única CNN para processar toda a imagem de entrada e prever simultaneamente a localização e a categoria dos objetos, o que a torna extremamente eficiente. Cada célula da grade é responsável por prever objetos cujo centro esteja contido dentro de sua região, atribuindo pesos para indicar a relevância de cada predição. Com isso, a YOLO alcança um balanço entre velocidade e precisão, tornando-se uma escolha popular para aplicações em tempo real. Esse mecanismo pode ser verificado na Figura 15.

Figura 15 – Mecanismo de detecção do modelo YOLO.



Fonte: Adaptado de REDMON *et al.* (2016).

Além disso, a YOLO prevê, para cada caixa delimitadora, um conjunto de variáveis de saída. Essas variáveis incluem as coordenadas (x, y), que representam a

posição do centro da caixa em relação à célula da grade. Também incluem a largura (*w*) e a altura (*h*), que determinam suas dimensões, além da pontuação de confiança, que expressa a probabilidade de a caixa conter um objeto e a precisão da sua localização. Por fim, há o identificador da classe, que especifica a categoria do objeto detectado (REDMON *et al.*, 2016).

Destarte, devido à sua abordagem inovadora e alto desempenho, a YOLO rapidamente se tornou uma referência na detecção de objetos em tempo real (LING; ISA, 2023). Além disso, versões subsequentes da YOLO introduziram aprimoramentos, como arquiteturas mais profundas e eficientes, técnicas de ancoragem aprimoradas e estratégias de pré-processamento otimizadas.

No entanto, apesar de suas vantagens, a YOLO apresenta algumas limitações que impactam seu desempenho em determinados cenários. Um dos principais desafios é a dificuldade na detecção de objetos pequenos, pois a segmentação da imagem em uma grade fixa pode não capturar detalhes sutis em escalas reduzidas. Além disso, a YOLO pode ter problemas ao lidar com objetos parcialmente ocluídos ou muito próximos entre si, dificultando a diferenciação adequada.

3.2 Estado da Arte em Detecção de Objetos para Placas de Circuito Impresso

A detecção de falhas em PCBs e PCBA é um campo bem estabelecido no contexto de AOIs. Historicamente, as abordagens têm sido divididas em duas categorias principais: técnicas clássicas de visão computacional e técnicas baseadas em aprendizado profundo. Apesar de a maioria dos estudos concentrarem-se apenas em defeitos de PCBs, poucas metodologias foram desenvolvidas especificamente para PCBA e há uma dificuldade na obtenção de conjuntos de dados reais para o treinamento e validação dos modelos, visto que a maioria dos estudos existentes utilizam dados sintéticos devido à dificuldade de acesso a amostras reais (LING; ISA, 2023).

3.2.1 Técnicas de Visão Computacional Clássica

Técnicas de visão computacional clássica desempenham um papel significativo nas soluções AOI, comumente empregando métodos referenciais como subtração de imagem, correspondência de características ou correspondência de modelos. Onshaunjit e Srinonchat (2021) apresentaram um algoritmo que utiliza conversão em escala de cinza, filtragem mediana, transformações afins e agrupamento c-means difuso para segmentação de imagem, visando detectar e classificar 14 tipos de defeitos de superfície de PCB. A identificação de defeitos é realizada através da subtração de imagens. Hassanin *et al*. (2019) desenvolveram um sistema utilizando tecnologia SURF e visão computacional clássica para detectar defeitos, empregando operações

XOR e avaliação de distância Euclidiana. No entanto, métodos que dependem de referências podem sofrer redução de desempenho devido a problemas como desalinhamento de imagens e variações nas condições ambientais durante a captura.

Métodos não referenciais oferecem uma solução para os desafios de desalinhamento, baseando-se na verificação da regra de design geral, mas exigem um entendimento abrangente do design de PCBA (AKHTAR, 2022). Zhu *et al.* (2020) desenvolveram uma metodologia para detectar e localizar furos circulares em PCBs, usando um detector de Canny para gerar mapas de bordas e transformações de distância para criar mapas espaciais. Annaby *et al.* (2023) apresentaram uma versão modificada do método de correlação cruzada normalizada para reduzir o custo computacional, convertendo sub-imagens 2D em descritores de recursos 1D.

Geetha e Bharathi (2021) introduziram uma metodologia para inspecionar PCBs utilizando operações NOT e XOR para diferenciar e subtrair tipos de defeitos de uma imagem modelo. Contudo, a aplicação do algoritmo de limiar de Kittler-Illingworth tem sua eficácia reduzida devido a impurezas ou irregularidades que são erroneamente identificadas como defeitos (DENG; LUO; DAI, 2018). Uma limitação chave das técnicas clássicas reside na sua inflexibilidade em configurações do sistema, abrindo espaço para a adoção crescente de métodos baseados em aprendizado profundo.

3.2.2 Técnicas de Aprendizado Profundo

As técnicas de aprendizado profundo, particularmente as Redes Neurais Convolucionais, superam as limitações dos métodos clássicos, processando a imagem inteira e aprendendo características essenciais para a classificação e detecção. No entanto, essas técnicas exigem uma quantidade substancial de dados diversificados para treinamento eficiente.

Lim *et al.* (2023) introduziram um algoritmo para detecção de defeitos em PCBs usando uma Rede de Pirâmide de Características – *Feature Pyramid Network* (FPN) aprimorado para melhorar o desempenho do modelo de detecção de objetos YOLOv5. Yang e Kang (2023) desenvolveram um método que aprimora a rede YOLOv7 com uma estrutura Swin Transformer V2 melhorada, introduzindo o mecanismo de *Shuffle Attention* de Fator de Magnificação.

Li *et al.* (2021) propuseram um sistema híbrido combinando YOLOv2 e *Faster RCNN* com um método de auto-adaptação para coleta de dados excepcionais dos quais o modelo cometeu erros, permitindo o treinamento e adaptação contínua. Yuan *et al.* 2024 apresentaram a rede YOLO-HorNet-MCBAM-CARAFE como uma melhoria do YOLOv5, visando a identificação precisa de pequenos defeitos em PCBs.

Zhang *et al.* (2024) projetaram a LDD-Net, uma rede leve para a detecção de defeitos, equilibrando precisão e custo computacional ao introduzir uma rede de

extração de características leve e um módulo de atenção eficiente. Wang *et al.* (2024) propuseram melhorias no algoritmo YOLOv8 para aumentar a precisão de detecção, incorporando GhostNet e HGNetV2 como parte da arquitetura do modelo.

3.3 Considerações Finais do Capítulo

Este capítulo abordou os conceitos fundamentais da detecção de objetos e analisou o estado da arte na área. A partir dessa análise, foi definido que o modelo a ser adotado para a detecção de defeitos neste trabalho será a arquitetura YOLO. A escolha pela YOLO se justifica, sobretudo, pela sua notável capacidade de processamento em tempo real, característica essencial para cenários industriais. A capacidade do YOLO de realizar detecção de objetos em uma única passagem pela imagem, ao invés de múltiplas iterações, proporciona uma alta eficiência computacional, garantindo a viabilidade de sua aplicação em sistemas que requerem respostas rápidas e constantes, como em linhas de produção ou inspeção automatizada.

A partir desse contexto e dos conceitos introduzidos nesta seção, será possível detalhar a arquitetura da YOLO, analisar seu funcionamento e, por fim, configurar os parâmetros necessários para o treinamento do modelo.

4 ANÁLISE DA ARQUITETURA DO MODELO SELECIONADO

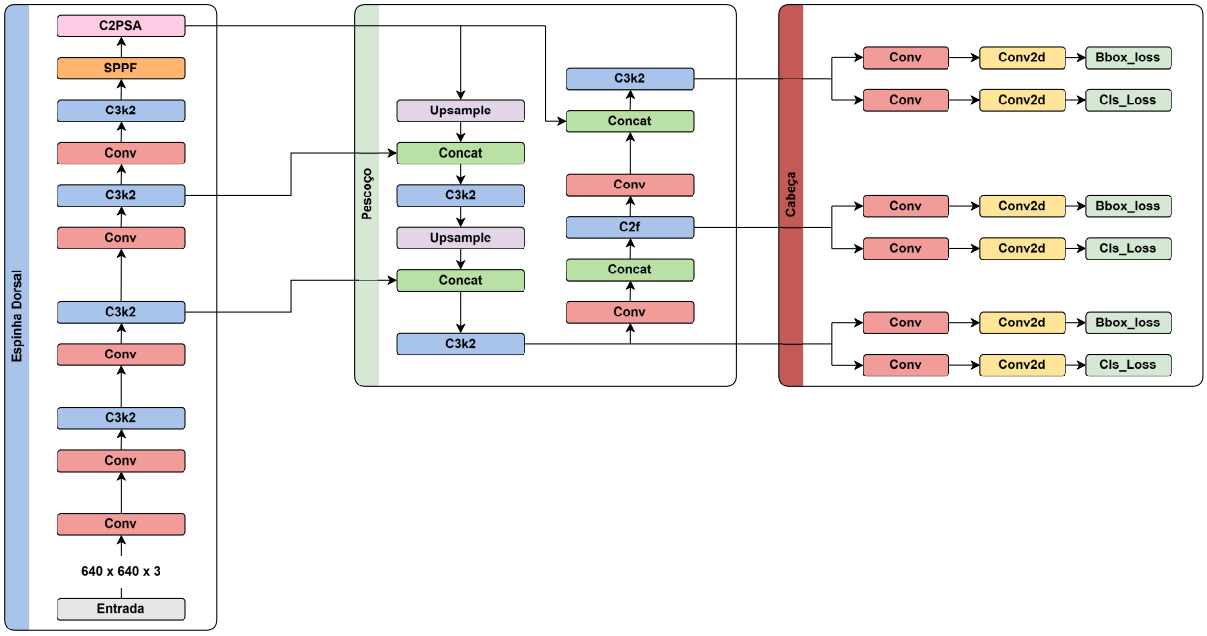
Neste capítulo, será realizada uma análise detalhada da arquitetura do modelo selecionado para a detecção de objetos. Dada a evolução dos modelos de detecção ao longo dos anos, optou-se por utilizar a versão mais recente da família YOLO, a YOLOv11.

A versão 11 do modelo incorpora diversas melhorias em relação às suas predecessoras, incluindo otimizações na extração de características, aprimoramentos na detecção de objetos pequenos e refinamentos no balanceamento entre velocidade e precisão (JOCHER; QIU, 2024). Dessa forma, serão abordados os principais componentes da arquitetura da YOLOv11 e suas características inovadoras.

4.1 Componentes e Arquitetura do Modelo

A arquitetura da YOLO é composta por três elementos fundamentais, cada um desempenhando um papel essencial no processo de detecção de objetos. O primeiro componente, denominado espinha dorsal, é responsável pela extração de características, utilizando redes neurais convolucionais para transformar os dados brutos da imagem em mapas de características multi-escala. Em seguida, o pescoço atua como uma camada intermediária, empregando mecanismos especializados para agregar e refinar as representações das características extraídas, garantindo melhor aproveitamento das informações em diferentes escalas. Por fim, a cabeça desempenha a função de predição, gerando as saídas finais para a localização dos objetos e suas respectivas classificações a partir dos mapas de características refinados (KHANAM; HUSSAIN, 2024). A Figura 16 apresenta uma visão geral dessa arquitetura.

Figura 16 – Arquitetura base da YOLOv11.



Fonte: Elaboração própria (2024).

4.2 Espinha Dorsal

A espinha dorsal, ou *backbone*, é um componente essencial na arquitetura da YOLO, sendo responsável pela extração de características da imagem de entrada em múltiplas escalas. Esse processo é realizado por meio do empilhamento de camadas convolucionais e blocos especializados, que geram mapas de características em diferentes resoluções, garantindo que informações detalhadas e contextuais sejam capturadas em diversos níveis da imagem (ZHAO *et al.*, 2024).

4.2.1 Camadas Convolucionais

A YOLOv11 mantém uma estrutura semelhante às suas predecessoras, utilizando camadas convolucionais iniciais para realizar a redução de resolução da imagem. Essas camadas formam a base do processo de extração de características, reduzindo gradualmente as dimensões espaciais enquanto aumentam o número de canais. Uma das principais melhorias introduzidas na YOLOv11 é a adoção do bloco C3k2, em substituição ao C2f, utilizado em versões anteriores (KHANAM; HUSSAIN, 2024). O C3k2 é uma versão mais eficiente do *Cross Stage Partial* (CSP), projetado para reduzir o custo computacional sem comprometer o desempenho (WANG *et al.*, 2020). Esse bloco adota duas convoluções menores em vez de uma única convolução maior, como na YOLOv8. O sufixo k2 no nome do bloco indica o uso de um tamanho de filtro reduzido, contribuindo para um processamento mais rápido enquanto preserva a capacidade de aprendizado do modelo (JOCHER; QIU, 2024).

4.2.2 Módulos de aprimoramento de características

Além das melhorias na estrutura da espinha dorsal, a YOLOv11 mantém o bloco *Spatial Pyramid Pooling - Fast* (SPPF), já presente em versões anteriores, mas introduz um novo módulo denominado *Cross Stage Partial with Spatial Attention* (C2PSA), posicionado logo após o SPPF (KHANAM; HUSSAIN, 2024). O C2PSA é um aprimoramento significativo que fortalece a atenção espacial nos mapas de características, permitindo que o modelo se concentre de maneira mais eficaz nas regiões mais importantes da imagem. Esse mecanismo de atenção espacial possibilita um refinamento adicional na detecção de objetos, tornando a YOLOv11 mais eficiente na identificação de alvos de diferentes tamanhos e posições dentro da cena (KHANAM; HUSSAIN, 2024).

4.3 Pescoço

O pescoço, ou *neck*, desempenha um papel fundamental na arquitetura da YOLOv11, combinando características extraídas em diferentes escalas da imagem e transmitindo-as para a cabeça do modelo, onde as detecções finais são realizadas. Esse processo envolve operações de aumento de resolução de imagem e concatenação de mapas de características provenientes de diferentes níveis da rede, permitindo que o modelo capture informações de múltiplas escalas de forma eficiente. A estrutura do pescoço na YOLOv11 pode ser visualizada na Figura 16, onde são destacadas suas principais camadas e conexões.

Uma das principais mudanças introduzidas na YOLOv11 é a substituição do bloco C2f, utilizado em versões anteriores, pelo novo C3k2 no pescoço, assim como na espinha dorsal. Essa alteração aprimora a eficiência e o desempenho no processo de agregação de características, resultando em uma melhor performance geral do modelo (KHANAM; HUSSAIN, 2024).

4.4 Cabeça

A estrutura da cabeça, ou *head*, do YOLOv11 é responsável por gerar as previsões finais de detecção e classificação de objetos. Ela recebe os mapas de características processados pelo pescoço e realiza um refinamento dessas informações, produzindo as caixas delimitadoras e os rótulos de classe dos objetos presentes na imagem. A cabeça do YOLOv11 foi aprimorada com melhorias significativas, que aumentam tanto a eficiência computacional quanto a precisão na detecção (JOCHER; QIU, 2024).

Dentro dessa estrutura, estão presentes as camadas de convolução CBS (do inglês, *Convolution-BatchNorm-SiLU*), que desempenham um papel crucial no re-

finamento das características extraídas. Esses blocos estabilizam o fluxo de dados através da rede e utilizam a função de ativação SiLU, o que contribui para uma detecção mais precisa e eficiente (KHANAM; HUSSAIN, 2024).

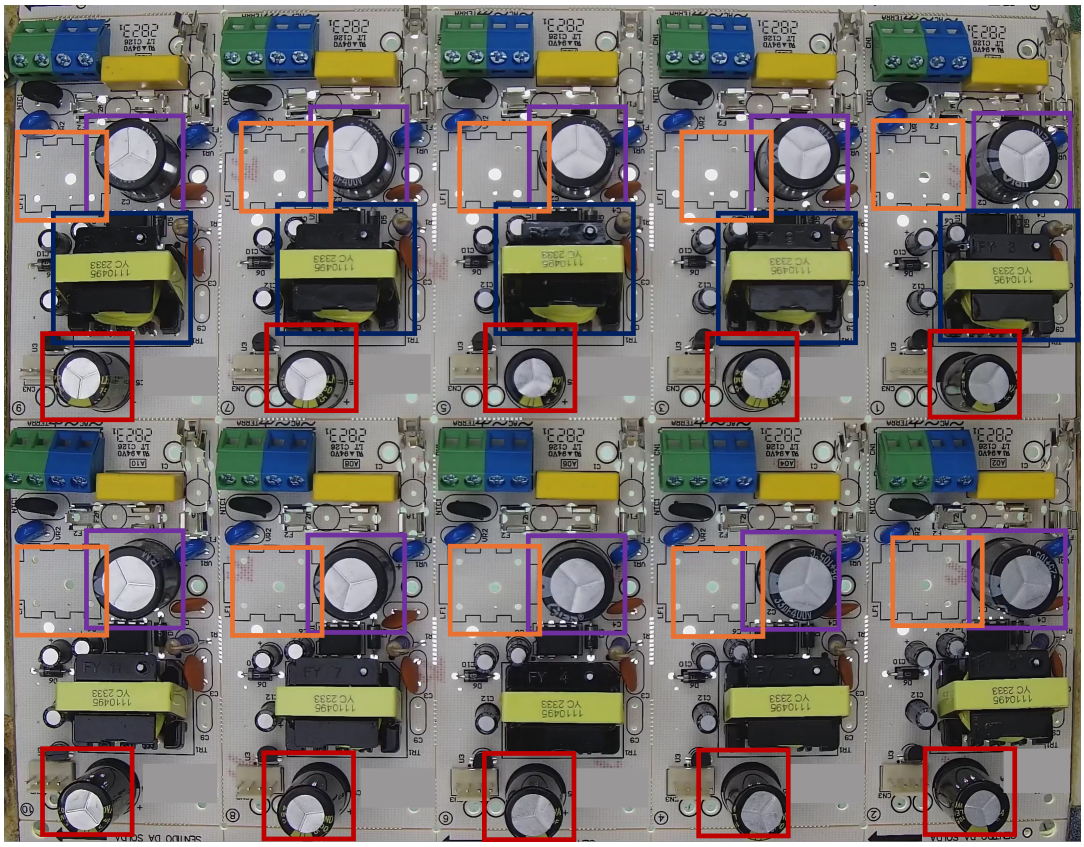
5 DESENVOLVIMENTO E TREINAMENTO DO MODELO

Nesse capítulo será discutida a preparação do conjunto de dados, com a coleta e anotação das imagens de PCBAs. Em seguida, será apresentada a implementação do mecanismo de atenção com embaralhamento para melhorar o desempenho do modelo. Também serão exploradas as configurações dos hiperparâmetros e o processo de treinamento do modelo.

5.1 Conjunto de Dados

O conjunto de dados foi coletado manualmente usando o sistema de câmeras da AOI de uma linha de produção industrial, consistindo em 1600 imagens 2D RGB de PCBAs. A AOI possui um sistema composto por cinco câmeras, posicionadas de forma circular ao redor do ponto onde a peça é posicionada pela esteira da linha de produção. Cada peça que passa pela AOI contém 10 PCBAs que são capturadas em diferentes ângulos pelas câmeras, garantindo uma cobertura abrangente para inspeção. A Figura 17 mostra uma peça com 10 PCBAs, na qual defeitos foram identificados e marcados na imagem.

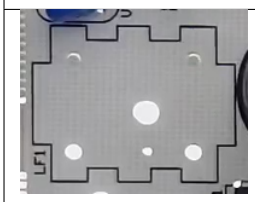
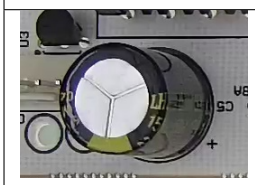
Figura 17 – Peça defeituosa capturada pelo sistema de câmeras da linha de produção.



Fonte: Elaboração própria (2024).

Do conjunto total de dados, 350 imagens retratam componentes ausentes, 400 imagens apresentam transformadores deslocados, 450 imagens correspondem a capacitores com polaridade invertida e 400 imagens são de PCBAs sem defeitos. Os defeitos podem ser verificados no Quadro 1. Além disso, as imagens do conjunto de dados possuem dimensões de 3840 x 2160. As imagens de entrada para o modelo de detecção de objetos são configuradas com dimensões de 640 x 640.

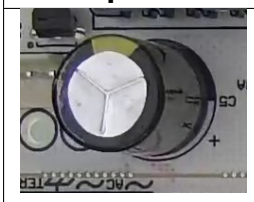
Quadro 1 – Tipos de defeitos do conjunto de dados.

Defeito	Descrição
	Componente Ausente
	Capacitor Reversamente Polarizado 1
	Capacitor Reversamente Polarizado 2
	Transformador Desalinhado

Fonte: Elaboração própria (2024).

Os defeitos presentes nas PCBAs podem ser identificados com base em características visuais específicas. No caso do "Componente ausente", a falha é evidenciada pela ausência do componente na placa, sendo sua posição indicada por um desenho do *layout*, facilitando a identificação do defeito. Para os capacitores, o "Capacitor reversamente polarizado 1" apresenta o defeito quando a marca dourada do capacitor está virada para baixo, indicando uma inversão na sua polarização. Já no caso do "Capacitor reversamente polarizado 2", a falha ocorre quando a marca branca do capacitor fica virada para a direita, também sinalizando uma polarização incorreta. Por fim, o "Transformador desalinhado" é identificado quando a parte amarela inferior do componente fica exposta, indicando que o transformador não está corretamente posicionado na placa. Os componentes sem defeitos podem ser vistos no Quadro 2, servindo como referência para a correta identificação dos componentes.

Quadro 2 – Componentes sem defeitos do conjunto de dados.

Componente	Descrição
	Capacitor 1
	Capacitor 2
	Transformador

Fonte: Elaboração própria (2024).

Para tornar o modelo mais robusto e melhorar sua capacidade de diferenciar entre componentes defeituosos e normais, também foram anotadas as peças sem defeito. Essa abordagem permite que o modelo aprenda não apenas a identificar falhas, mas também a reconhecer corretamente os componentes que estão em conformidade. Dessa forma, reduz-se a taxa de falsos positivos, evitando que peças normais sejam erroneamente classificadas como defeituosas.

Neste estudo, foi alocado 70% do conjunto de dados para treinamento, 20% para validação e 10% para teste. Essa distribuição garante uma abordagem equilibrada, permitindo que o modelo aprenda de forma eficaz a partir dos dados de treinamento, enquanto mantém uma porção substancial para validação, a fim de ajustar hiperparâmetros e evitar sobreajustes. O conjunto de teste fornece uma avaliação imparcial do desempenho do modelo, garantindo que os resultados sejam generalizáveis e aplicáveis a novos dados.

5.2 Tecnologias e ferramentas

Nesta seção, define-se as principais tecnologias e ferramentas utilizadas ao longo do desenvolvimento do projeto. O foco será dado às soluções que desempenharam um papel fundamental na implementação e otimização dos processos de detecção de objetos e inspeção de defeitos em PCBAs.

5.2.1 Linguagem de programação Python

Python foi escolhido para este projeto devido à sua ampla adoção na comunidade de aprendizado de máquina, sua vasta biblioteca de ferramentas e sua facilidade de uso para manipulação de dados e desenvolvimento de modelos de aprendizado profundo.

5.2.2 PyTorch

PyTorch é uma biblioteca de aprendizado de máquina em Python que facilita a criação e o treinamento de redes neurais profundas. Sua interface dinâmica e flexível permite experimentação eficiente e desenvolvimento rápido de modelos de visão computacional. Além disso, as versões atuais da YOLO utilizam o PyTorch como ferramenta de desenvolvimento (KHANAM; HUSSAIN, 2024).

5.2.3 Ultralytics

A Ultralytics é uma biblioteca em Python que fornece as implementações otimizadas do YOLO, permitindo fácil configuração e treinamento de modelos de detecção de objetos (JOCHER; QIU, 2024). Sua biblioteca é amplamente utilizada para tarefas que exigem alta precisão e desempenho em tempo real.

5.2.4 Roboflow

O Roboflow é uma plataforma *online* projetada para otimizar o processamento, organização e anotação de conjuntos de dados utilizados em visão computacional. Ela simplifica as etapas de preparação e pré-processamento de imagens, desempenhando um papel crucial na criação eficiente de modelos de detecção de objetos. Através do Roboflow, é possível realizar a anotação de imagens no contexto do projeto de forma prática e eficaz, agilizando significativamente o fluxo de trabalho.

5.3 Ambiente de Execução

Os experimentos foram conduzidos em um ambiente de alto desempenho, o que assegura a eficiência no processamento dos modelos de visão computacional. O *hardware* utilizado é composto pelos itens abaixo.

a) processador: Intel(R) Core(TM) i9-14900K;

b) placa de vídeo: NVIDIA(R) GeForce RTX 4090 com 24 GB de memória VRAM;

c) memória RAM: 128 GB.

Essa configuração permite treinar e inferir modelos complexos de forma eficiente, o que resulta em tempos de execução reduzidos e suporte para operações em larga escala.

5.4 Customização da Arquitetura

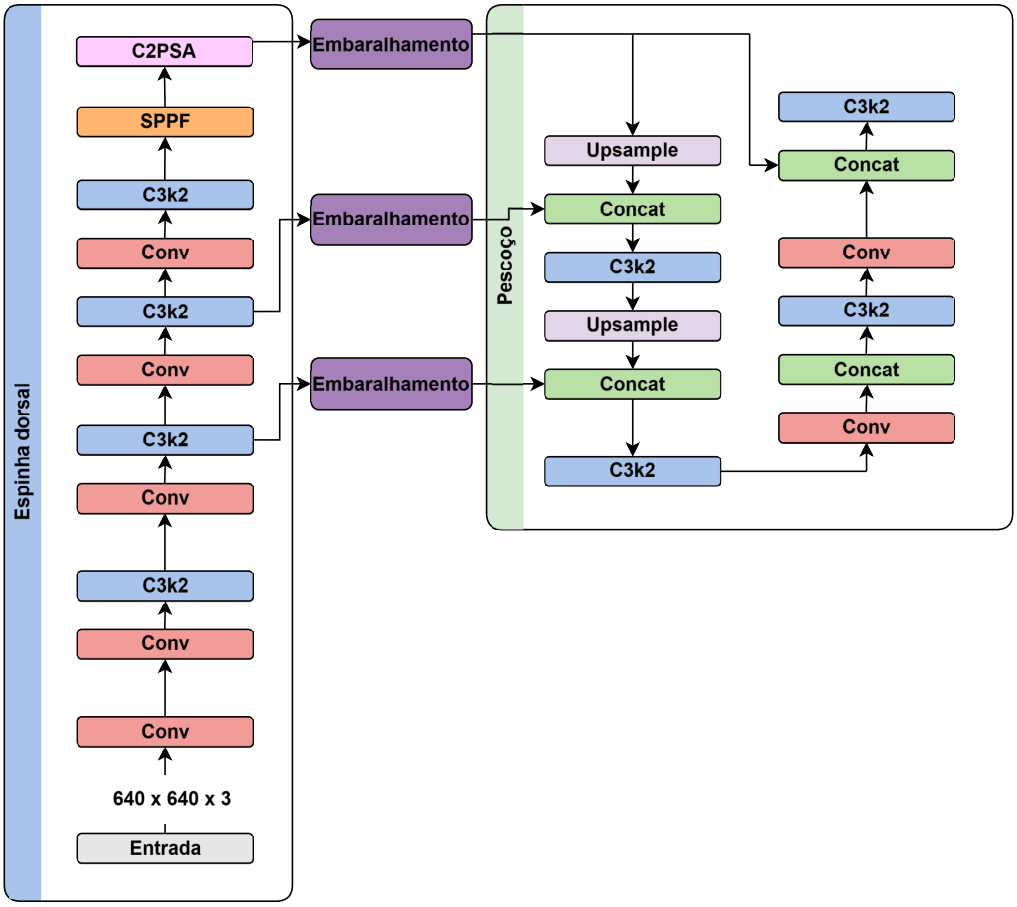
A inspeção de defeitos em PCBAs enfrenta desafios notáveis, especialmente devido à variação de escala dos objetos presentes na placa. Desde componentes minúsculos até problemas de desalinhamento, essas variações podem impactar negativamente a performance dos algoritmos de detecção. Modelos como o YOLO, embora eficientes em muitos cenários, têm dificuldades ao lidar com tais desafios (LING; ISA, 2023).

Dessa forma, a fim de aprimorar o desempenho do YOLO e mitigar esses problemas, é comum realizar customizações na arquitetura do modelo e empregar módulos adicionais. Tais adaptações podem incluir o uso de módulos de atenção, que ajudam a destacar características relevantes em diferentes escalas, e técnicas de fusão de características, que integram informações de níveis variados de resolução (WANG; SHEN; LI, 2024).

No entanto, muitas dessas técnicas de customização podem resultar em um aumento no uso de poder computacional, impactando negativamente a eficiência computacional do modelo. Tais modificações podem ocasionalmente levar a uma diminuição no tempo de inferência, tornando o modelo menos viável para aplicações em tempo real, onde a performance é essencial (ZHANG; YANG, 2021).

Portanto, para superar esses desafios sem sacrificar a eficiência computacional, foi implementado um módulo de Atenção com Embaralhamento entre a espinha dorsal e o pescoço da YOLO, como ilustra a Figura 18.

Figura 18 – Arquitetura aprimorada com Anteção com Embaralhamento.



Fonte: Elaboração própria (2024).

Conforme discutido na Seção 2 deste trabalho, essa abordagem proporciona uma melhoria no desempenho do modelo enquanto mantém o custo computacional otimizado, o que é fundamental para aplicações em tempo real na inspeção de PCBAs. Assim, ao posicionar o módulo de atenção nesta etapa, a arquitetura permite que o pescoço combine e aprimore essas características extraídas, resultando em uma melhoria na precisão de detecção sem introduzir uma sobrecarga computacional significativa.

5.5 Configuração de Hiperparâmetros

A configuração adequada dos hiperparâmetros é um passo crucial no treinamento de modelos de aprendizado profundo, pois influencia diretamente tanto a convergência quanto o desempenho do modelo. Para o treinamento do modelo, optou-se por uma configuração que proporciona um bom equilíbrio entre a velocidade de treinamento e a precisão de detecção.

Com base nos resultados experimentais obtidos por Ju e Cai (2023), que indicaram que um número de épocas entre 60 e 70 resulta em um bom desempenho,

os hiperparâmetros foram configurados para um máximo de 150 épocas. Uma época representa uma passagem completa por todo o conjunto de dados de treinamento. Essa margem permite que o modelo refine seu aprendizado a partir dos dados sem incorrer em sobreajuste excessivo. A taxa de aprendizagem inicial foi estipulada em 0,001, uma decisão estratégica para assegurar uma convergência estável do modelo, minimizando o risco de flutuações durante o processo de treinamento.

Além disso, o tamanho de entrada das imagens foi definido como 640x640 *pixels*. Esse dimensionamento é um balanço entre preservação de detalhes importantes nas imagens e viabilidade computacional, dado que tamanhos menores de entrada podem não capturar adequadamente detalhes críticos de defeitos, enquanto tamanhos muito grandes exigiriam mais recursos computacionais (GOODFELLOW; BENGIO; COURVILLE, 2016).

Para inicializar o processo de treinamento, foram utilizados os pesos da rede pré-treinada padrão do modelo pequeno YOLOv11.

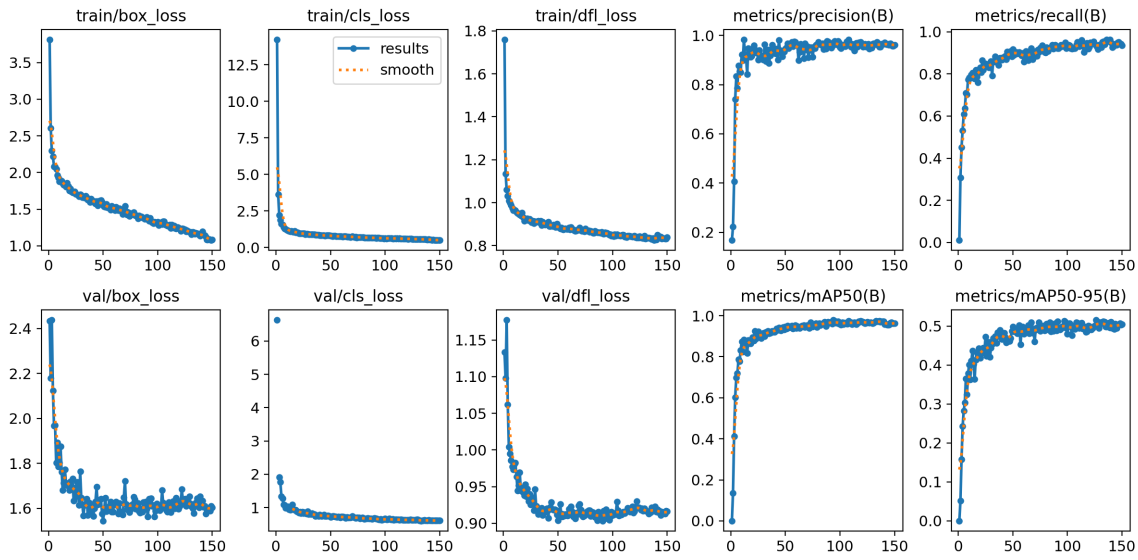
6 RESULTADOS E DISCUSSÃO

Neste capítulo, serão apresentados e analisados os resultados obtidos com a implementação do modelo adaptado para a detecção de defeitos em PCBAs. Os experimentos realizados visaram avaliar tanto a eficácia quanto a eficiência do modelo, considerando métricas de desempenho chave, como precisão, *F-Score* e tempo de inferência.

Os gráficos apresentados na Figura 19 mostram que as perdas durante o treinamento — perda de caixa, perda de classificação e perda focal de distribuição — exibem uma clara tendência de queda, indicando um aprendizado eficaz à medida que o modelo converge. As perdas de validação também diminuem significativamente, demonstrando uma boa generalização dos dados de treinamento para os dados de validação, embora algumas flutuações iniciais sejam observadas, especialmente na perda de classificação, que se estabiliza com o tempo.

Em termos de métricas, a precisão, a sensibilidade e a mAP mostram melhorias consistentes ao longo do processo de treinamento. A precisão estabiliza-se próximo a 1,0, refletindo a capacidade do modelo de fazer previsões precisas. A sensibilidade melhora de forma constante, alcançando um nível elevado, o que sugere que o modelo está identificando corretamente a maioria dos defeitos. As métricas mAP@50 e mAP@50-95, que medem a precisão geral do modelo em diferentes limiares, também mostram um aumento significativo, com o mAP@50 alcançando valores próximos a 0,97 e o mAP@50-95 em torno de 0,5, indicando ainda mais um forte desempenho.

Figura 19 – Resultados do treinamento do modelo.



Fonte: Elaboração própria (2024).

Os resultados detalhados por classe, conforme Tabela 1, demonstram que o modelo apresenta excelente desempenho na identificação de defeitos específicos. A classe "Componente ausente" obteve a maior pontuação, com um mAP@50 de 100%, indicando que o modelo consegue detectar essa falha com extrema precisão. As classes "Capacitor reversamente polarizado 1" e "Capacitor sem defeito 1" também apresentam altos valores de mAP@50, superiores a 97%, refletindo a capacidade do modelo de diferenciar corretamente componentes com e sem defeito. Para as classes "Capacitor reversamente polarizado 2" e "Capacitor sem defeito 2", os resultados são semelhantes, mantendo um desempenho consistente com valores próximos a 97%. No caso do "Transformador desalinhado", observa-se um pequeno decréscimo na mAP@50, atingindo 93,54%, o que pode indicar maior variabilidade na aparência desse defeito. A classe "Transformador sem defeito" também apresenta um desempenho levemente inferior às demais, com um F-Score de 91,85%, possivelmente devido a semelhanças visuais com defeitos sutis. No geral, o modelo exibe alta precisão e sensibilidade na detecção das classes avaliadas, garantindo um desempenho robusto na inspeção dos componentes analisados.

Tabela 1 – Resultados experimentais para cada classe.

Classe	mAP@50	mAP@50-95	F-Score
Componente ausente	100,00%	54,47%	100%
Capacitor reversamente polarizado 1	98.82%	52,71%	98,42%
Capacitor sem defeito 1	97,73%	52,86%	97,61%
Capacitor reversamente polarizado 2	96,68%	51,39%	96,28%
Capacitor sem defeito 2	96,89%	51,66%	96,44%
Transformador desalinhado	93,54%	49,28%	93,92%
Transformador sem defeito	94,85%	50,23%	91,85%

Fonte: Elaboração própria (2024).

Na Figura 20, é possível visualizar um exemplo de detecção correta para todos os tipos de defeitos presentes no conjunto de dados. O modelo identificou corretamente componentes ausentes, transformadores desalinhados e capacitores com polaridade invertida, demonstrando sua capacidade de generalização e precisão na identificação das falhas. Esse resultado reforça a importância do processo de anotação detalhada, incluindo tanto os defeitos quanto os componentes em conformidade, garantindo um melhor desempenho do modelo.

Figura 20 – Resultado de predição do modelo.

Fonte: Elaboração própria (2024).

A arquitetura customizada YOLOv11 supera significativamente o modelo base da YOLOv11 em termos de precisão e eficiência para a detecção de defeitos em PCBA, conforme demonstrado na Tabela 2. Especificamente, a YOLOv11 customizada alcança um mAP@50 mais alto (96, 93% em comparação com 89, 64%) e uma melhoria notável no mAP@50-95, atingindo 51, 80% em relação aos 41, 23% do modelo base. Além disso, o F-Score aumenta de 89% para 96, 36%, indicando um melhor equilíbrio entre precisão e sensibilidade. Além dessas melhorias, o tempo de inferência do modelo customizado é ligeiramente mais rápido, em 2,20 ms comparado a 2,50 ms do modelo base, sugerindo que a arquitetura personalizada não apenas melhora o desempenho de detecção, mas também mantém a eficiência.

Tabela 2 – Resultados experimentais para os modelos.

Tipo de Modelo	mAP@50	mAP@50-95	F-Score	Inferência (ms)
YOLOv11 (base)	89,64%	41,23%	89,00%	2,50
YOLOv11 (proposto)	96,93%	51,80%	96,36%	2,20

Fonte: Elaboração própria (2024).

7 CONCLUSÃO

As PCBAs são essenciais na fabricação de dispositivos eletrônicos, com a qualidade do produto dependendo fortemente de sua integridade. Garantir PCBAs livres de defeitos é, portanto, fundamental. Neste estudo, apresentamos uma abordagem para detecção de defeitos utilizando o YOLOv11, aprimorado com uma arquitetura personalizada que integra um módulo de Atenção com Embaralhamento, aumentando significativamente a precisão da detecção e a exatidão da localização dos defeitos.

Diferentemente de metodologias tradicionais que dependem de coordenadas pré-definidas ou imagens de referência, a abordagem proposta processa imagens completas de PCBAs, detectando e localizando automaticamente defeitos como componentes ausentes, transformadores desalinhados e capacitores com polaridade invertida. Com um mAP elevado de 96, 93%, um *F-score* de 96, 36% e um tempo de inferência rápido de 2,20 ms, o sistema proposto não apenas identifica defeitos com precisão, mas também atende aos requisitos industriais para aplicações em tempo real.

Com base nos resultados obtidos, futuras melhorias poderiam focar no aumento da robustez do modelo para diferentes condições industriais. Para isso, seria necessário expandir o treinamento e a validação para considerar variações como mudanças nos ângulos de captura e ruídos nas imagens, assegurando um desempenho confiável em cenários reais de produção. Outra direção relevante seria a otimização computacional, empregando técnicas como *pruning* de parâmetros e quantização, permitindo que o modelo opere de forma mais eficiente, inclusive em dispositivos embarcados com restrições de *hardware*. Além disso, a integração do sistema a plataformas de controle de qualidade em tempo real viabilizaria uma automação mais fluida no processo produtivo. Também seria interessante adaptar o modelo para identificar novos tipos de falhas e componentes, garantindo que ele acompanhe a evolução dos designs de PCBAs. Por fim, um estudo aprofundado sobre o impacto do sistema na produtividade e na economia de custos, comparando-o a métodos tradicionais de inspeção, ajudaria a demonstrar seus benefícios e viabilidade para aplicação industrial.

REFERÊNCIAS

AKHTAR, M. B. **The Use of a Convolutional Neural Network in Detecting Soldering Faults from a Printed Circuit Board Assembly**. *High Tech Journal*, EUHERA, v. 9, p. 837–841, 2022. 13, 40

ANNABY, M. H.; FOUDA, Y. M.; RUSHDI, M. A. **Improved normalized cross-correlation for defect detection in printed-circuit boards**. *IEEE Trans. Semicond. Manuf.*, p. 199–211, 2023. 40

BAY, H. *et al.* **Speeded-Up Robust Features (SURF)**. *Computer Vision and Image Understanding*, v. 110, n. 3, p. 346–359, 2008. 36

BHATT, D. *et al.* **CNN variants for computer vision: history, architecture, application, challenges and future scope.** *Electronics*, 2021. 26, 27

BISHOP, C. M. ***Pattern Recognition and Machine Learning.*** 1. ed. [S.l.]: Springer, 2006. 25

CHEN, I.-C.; HWANG, R.-C.; HUANG, H.-C. **PCB Defect Detection Based on Deep Learning Algorithm**. *Processes*, v. 11, p. 775, 2024. 14

DALAL, N.; TRIGGS, B. **Histograms of oriented gradients for human detection.** *In*: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. San Diego, Califórnia: IEEE, 2005. 36

DENG, Y.-S.; LUO, A.-C.; DAI, M.-J. **Building an Automatic Defect Verification System Using Deep Neural Network for PCB Defect Classification**. *In*: *2018 4th International Conference on Frontiers of Signal Processing (ICFSP)*. Poitiers, Vienne: [s.n.], 2018. p. 145–149. 40

DUBEY, S. R.; SINGH, S. K.; CHAUDHURI, B. B. **Activation functions in deep learning: A comprehensive survey and benchmark**. *Neurocomputing*, p. 92–108, 2022. 21, 22, 23

ELFWING, S.; UCHIBE, E.; DOYA, K. **Sigmoid-Weighted Linear Units for Neural Network Function Approximation in Reinforcement Learning**. *Neural Networks*, v. 107, p. 3–11, 2018. 24

GEETHA, R.; BHARATHI, M. **Printed circuit board defect detection using mathematical expression in image processing**. *In*: *Proc. of the First International Conference on Advanced Scientific Innovation in Science, Engineering and Technology*. Chennai, Tâmil Nadu: ICASISET, 2021. p. 16–17. 40

GHORBANI, F. *et al.* **A deep learning approach for inverse design of the metasurface for dual-polarized waves**. *Applied Physics A*, v. 127, p. 869, 2021. 19

GHOSH, B. *et al.* **An Empirical Analysis of Generative Adversarial Network Training Times with Varying Batch Sizes.** *In*: *The 11th IEEE Annual Ubiquitous Computing, Electronics and Mobile Communcation Conference*. Cidade de Nova Iorque, Nova Iorque: IEEE, 2020. 25

GIRSHICK, R. **Fast r-cnn**. *In*: *Proceedings of the IEEE international conference on computer vision*. Santiago: IEEE, 2015. p. 1440–1448. 37

GIRSHICK, R. *et al.* **Rich feature hierarchies for accurate object detection and semantic segmentation**. *In*: *Proceedings of the IEEE conference on computer vision and pattern recognition*. Columbus, Ohio: IEEE, 2014. p. 580–587. 37

GOODFELLOW, I. J.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. MIT Press, 2016. Disponível em: https://www.deeplearningbook.org/. Acesso em: 15 nov. 2024. 17, 18, 24, 25, 26, 27, 52

GOUTTE, C.; GAUSSIER, E. **A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation**. *In*: *Advances in Information Retrieval*. [S.l.]: Springer Berlin Heidelberg, 2005. p. 345–359. 34, 35

GUO, M.-H. *et al.* **Attention mechanisms in computer vision: A survey**. *Computational Visual Media*, v. 8, 2022. 29, 30, 31

HASSANIN, A.-A.; EL-SAMIE, F. A.; BANBY, G. E. **A real-time approach for automatic defect detection from PCBs based on surf features and morphological operations**. *Multimedia Tools Appl.*, p. 1–19, 2019. 39

HENDRYCKS, D.; GIMPEL, K. **Gaussian Error Linear Units (GELUs)**. *arXiv*, 2016. Disponível em: https://arxiv.org/abs/1606.08415. Acesso em: 13 nov. 2024. 24

HU, J. *et al.* **Squeeze-and-excitation networks.** *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 45, p. 2011 – 2023, 2020. 29, 30

IBM. **What is a neural network?** 2023. Disponível em: https://www.ibm.com/think/topics/neural-networks. Acesso em: 18 nov. 2024. 19

JAISWAL, P.; GUPTA, N. K.; AMBIKAPATHY, A. **Comparative study of various training algorithms of artificial neural network**. *In*: *2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)*. Greater Noida, Gautam Buddha Nagar: IEEE, 2018. p. 1097–1101. 24

JOCHER, G.; QIU, J. ***Ultralytics YOLO11***. 2024. Disponível em: https://docs.ultralytics.com/models/yolo11/. Acesso em: 20 jan. 2025. 42, 43, 44, 49

JU, R.-Y.; CAI, W. **Fracture detection in pediatric wrist trauma x-ray images using yolov8 algorithm**. *Scientific Reports*, p. 20077, 2023. 51

KHANAM, R.; HUSSAIN, M. ***YOLOv11: An Overview of the Key Architectural Enhancements***. arXiv, 2024. Disponível em: https://doi.org/10.48550/arXiv.2410.17725. Acesso em: 20 dez. 2024. 42, 43, 44, 45, 49

KOIRALA, A. *et al.* **Deep learning – Method overview and review of use for fruit detection and yield estimation**. *Computers and Electronics in Agriculture*, v. 162, p. 219–234, 2019. 35

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. **ImageNet classification with deep convolutional neural networks**. *Proc. NIPS*, NeurIPS, v. 1, p. 1097–1105, 2012. 13, 26

KUMAR, M.; KUMAR, M. **A Survey on Various Approaches of Automatic Optical Inspection for PCB Defect Detection**. *International Journal of Computer Sciences and Engineering*, IJCSE, v. 7, p. 837–841, 2019. 13, 18

LECUN, Y. *et al.* **Backpropagation Applied to Handwritten Zip Code Recognition**. *Neural Computation*, v. 1, p. 541–551, 1989. 26, 29

LEDERER, J. **Activation Functions in Artificial Neural Networks: A Systematic Overview**. *arXiv*, 2021. Disponível em: https://arxiv.org/abs/2101.09957. Acesso em: 24 nov. 2024. 20

LI, Y.-T.; KUO, P.; GUO, J.-I. **Automatic Industry PCB Board DIP Process Defect Detection System Based on Deep Ensemble Self-Adaption Method**. *IEEE Transactions on Components, Packaging and Manufacturing Technology*, p. 312–323, 2021. 40

LI, Z. *et al.* **Lightweight PCB defect detection algorithm and deployment based on ASF-YOLO**. *In*: *2024 7th International Conference on Computer Information Science and Application Technology (CISAT)*. Hangzhou: IEEE, 2024. p. 36–40. 15

LIM, J. *et al.* **A deep context learning based PCB defect detection model with anomalous trend alarming system**. *Results Eng.*, 2023. 40

LING, Q.; ISA, N. A. M. **Printed Circuit Board Defect Detection Methods Based on Image Processing, Machine Learning and Deep Learning: A Survey**. *IEEE Access*, IEEE, v. 11, p. 15921–15922, 2023. 13, 14, 37, 39, 50

LIU, W. *et al.* **SSD: Single shot multibox detector**. *Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science*, p. 21–37, 2016. 38

MCCULLOCH, W.; PITTS, W. **A logical calculus of the ideas immanent in nervous activity**. *Bulletin of Mathematical Biophysics*, v. 5, p. 115–133, 1943. 18

MISRA, D. **Mish: A Self Regularized Non-Monotonic Activation Function**. *In*: *Proc. 31st British Machine Vision Virtual Conference (BMVC)*. [S.l.: s.n.], 2020. 20

MURPHY, K. P. ***Machine Learning: A Probabilistic Perspective.*** 1. ed. [S.l.]: MIT Press, 2012. 25

ONSHAUNJIT, J.; SRINONCHAT, J. **Algorithmic Scheme for Concurrent Detection and Classification of Printed Circuit Board Defects**. *Computers, Materials and Continua*, p. 355–367, 2021. 39

RAINA, R.; MADHAVAN, A.; NG, A. Y. **Large-scale deep unsupervised learning using graphics processors**. *In*: *Proceedings of the 26th Annual International Conference on Machine Learning*. Cidade de Nova Iorque, Nova Iorque: Association for Computing Machinery, 2009. p. 873–880. 18

REDMON, J. *et al.* **You Only Look Once: Unified, Real-Time Object Detection.** *In*: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, Nevada: [s.n.], 2016. p. 779–788. 36, 38, 39

REN, J. *et al.* **Defect Detection for Printed Circuit Board Assembly Using Deep Learning**. *In*: *2022 8th International Conference on Control Science and Systems Engineering (ICCSSE)*. Guangzhou: IEEE, 2022. p. 85–89. 15

REN, S. *et al.* **Faster r-cnn: Towards real-time object detection with region proposal networks**. *Advances in neural information processing systems*, p. 91–99, 2015. 37

ROBINS, M. **The Difference Between Artificial Intelligence, Machine Learning and Deep Learning**. Intel Artificial Intelligence, 2020. Disponível em: https://community.intel.com/t5/Blogs/Tech-Innovation/Artificial-Intelligence-AI/The-Difference-Between-Artificial-Intelligence-Machine-Learning/post/1335666. Acesso em: 18 nov. 2024. 17, 18

RUMELHART, D.; HINTON, G.; WILLIAMS, R. **Learning representations by back-propagating errors.** *Nature*, p. 533–536, 1986. 25

SARKER, I. H. **Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions**. *Advances in Computational Approaches for Artifcial Intelligence, Image Processing, IoT and Cloud Applications*, SN Computer Science, p. 420–440, 2021. 17

SHARMA, N.; SHARMA, R.; JINDAL, N. **Machine Learning and Deep Learning Applications-A Vision**. *Global Transitions Proceedings*, v. 2, p. 24–28, 2021. 17

TACCHINO, F. *et al.* **. An artificial neuron implemented on an actual quantum processor**. *npj Quantum Information*, Springer Science and Business Media LLC, 2019. 18

VASWANI, A. *et al.* **Attention Is All You Need.** *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2017. 28

VIOLA, P.; JONES, M. **Rapid object detection using a boosted cascade of simple features**. *In*: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*. Kauai, Havaí: IEEE, 2001. v. 1. 36

WANG, C.-Y. *et al.* **Cspnet: A new backbone that can enhance learning capability of cnn**. *In*: *Proc. of the IEEE/CVF conference on computer vision and pattern recognition workshops*. Seattle, Washington: IEEE, 2020. p. 390—391. 43

WANG, H.; SHEN, S.; LI, M. **PCB Defect Detection Algorithm Based on Improved YOLOv8**. *In*: *2024 5th International Conference on Electronic Communication and Artificial Intelligence (ICECAI)*. Shenzhen, Guangdong: IEEE, 2024. p. 323–327. 41, 50

WOO, S. *et al.* **CBAM: Convolutional Block Attention Module**. *Computer Vision – ECCV 2018*, Springer International Publishing, p. 3–19, 2018. 30, 31, 32

YANG, Y.; KANG, H. **An Enhanced Detection Method of PCB Defect Based on Improved YOLOv7**. *Electronics*, 2023. 40

YUAN, M. *et al.* **YOLO-HMC: An Improved Method for PCB Surface Defect Detection**. *IEEE Transactions on Instrumentation and Measurement*, IEEE, 2024. 40

ZAIDI, S. S. A. *et al.* **A survey of modern deep learning based object detection models**. *Digital Signal Processing*, v. 126, p. 103514, 2022. ISSN 1051-2004. 35

ZAKARIA, M.; AL-SHEBANY, M.; SARHAN, S. **Artificial Neural Network : A Brief Overview**. *Int. Journal of Engineering Research and Applications*, IJERA, p. 07–12, 2014. 18, 19

ZHANG, J. C. L. *et al.* **LDD-Net: Lightweight printed circuit board defect detection network fusing multi-scale features**. *Engineering Applications of Artificial Intelligence*, 2024. 40

ZHANG, Q.; LIU, H. **Multi-scale defect detection of printed circuit board based on feature pyramid network**. *In*: *2021 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*. Dalian, Liaoning: IEEE, 2021. p. 911–914. 13

ZHANG, Q.-L.; YANG, Y.-B. **SA-Net: Shuffle Attention for Deep Convolutional Neural Networks**. *In*: *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Toronto, Ontário: IEEE, 2021. p. 2235–2239. 32, 33, 34, 50

ZHAO, X. *et al.* **A review of convolutional neural networks in computer vision.** *Artificial Intelligence Review*, v. 57, 2024. 26, 27, 28, 36, 43

ZHU, W.; GU, H.; SU, W. **A fast PCB hole detection method based on geometric features**. *Measurement Science and Technology.*, 2020. 40

ZOU, Z. *et al.* **Object Detection in 20 Years: A Survey.** *Proceedings of the IEEE*, p. 257–276, 2023. 36