Detecção de Defeitos em Placas de Circuito Impresso com YOLOv10 Otimizado por Ghost Convolution

Gabriel Mitoso[1,2], **Tiago de Melo**[1]

[1] Programa de Pós-graduação em Engenharia Elétrica (PPGEEL)
Universidade do Estado do Amazonas (UEA) – Manaus, AM – Brasil

[2]Instituto de Desenvolvimento Tecnológico (INDT)

gabriel.mitoso@indt.org.br, tmelo@uea.edu.br

Abstract. *The automated inspection of Printed Circuit Boards (PCBs) plays a fundamental role in ensuring quality and reliability in the electronics industry. This work proposes the application of the YOLOv10 model for PCB defect detection, with optimizations in the backbone architecture using the Ghost Convolution technique, aiming to reduce computational complexity without compromising precision. The experiments were conducted using the YOLOv10 Nano, Small and Medium models, trained on a public dataset containing 10,668 images. The results demonstrated that modifying the backbone reduced the number of parameters and GFLOPs, increasing the FPS of the Small model from 128 to 150 while maintaining a mAP@50 of 99.22%, surpassing previous approaches in the literature. These findings indicate that the Ghost Convolution technique can be an efficient solution for real-time industrial applications, enabling its adoption in other models and computationally constrained contexts.*

Resumo. *A inspeção automatizada de Placas de Circuito Impresso (PCIs) desempenha um papel fundamental na garantia da qualidade e confiabilidade na indústria eletrônica. Neste trabalho, propõe-se a aplicação do modelo YOLOv10 para a detecção de defeitos em PCIs, com otimizações na arquitetura do backbone por meio da técnica de convolução fantasma (Ghost Convolution), visando reduzir a complexidade computacional sem comprometer a precisão. Os experimentos foram realizados utilizando os modelos YOLOv10 Nano, Small e Medium, treinados com um conjunto de dados públicos contendo 10.668 imagens. Os resultados demonstraram que a modificação do backbone permitiu reduzir o número de parâmetros e GFLOPs, aumentando o FPS do modelo Small de 128 para 150 e mantendo um mAP@50 de 99,22%, superando abordagens anteriores da literatura. Esses resultados indicam que a técnica de convolução fantasma pode ser uma solução eficiente para aplicações industriais em tempo real, possibilitando sua adoção em outros modelos e contextos computacionalmente restritos.*

1. Introdução

As Placas de Circuito Impresso (PCIs) são a base da indústria eletroeletrônica, sendo essenciais em uma ampla gama de dispositivos, desde smartphones até equipamentos militares. Dada sua relevância, qualquer falha mínima no processo de fabricação pode comprometer o funcionamento do equipamento, tornando a inspeção e o controle de qualidade

etapas fundamentais [Bhattacharya and Cloutier 2022]. A importância desse controle aumenta diante do crescimento acelerado do mercado global de eletrônicos, impulsionado pela demanda crescente por dispositivos de consumo, automação industrial e inovação tecnológica.

Em 2021, o setor de manufatura eletrônica e serviços de design foi estimado em USD 481,3 bilhões, com uma taxa de crescimento anual de 10,5%, projetando alcançar cerca de USD 1158,6 bilhões até 2030 [Fonseca et al. 2024]. Esse cenário reflete a crescente exigência por padrões elevados de qualidade e eficiência, à medida que a indústria enfrenta desafios para atender às demandas de um mercado cada vez mais competitivo e dinâmico.

Durante o ciclo de fabricação de uma PCI, uma série de processos distintos são executados, aumentando sua vulnerabilidade a uma variedade de defeitos. Estes podem variar desde falhas na inserção de componentes até erros decorrentes de processos manuais [Chen et al. 2023]. Tais defeitos precisam ser prontamente identificados e segregados, visando sua correção imediata. Esta abordagem não apenas previne potenciais custos associados a reparos futuros, mas também assegura a integridade e a qualidade do produto final.

Os dois métodos predominantes para detecção de defeitos em PCIs incluem a inspeção visual realizada por seres humanos e a inspeção óptica automatizada (*Automated Optical Inspection* - AOI). Com a evolução tecnológica, as PCIs estão se tornando progressivamente mais complexas, caracterizadas por componentes menores e em maior densidade [Rodrigues 2023]. Essa complexidade inviabiliza a eficácia da inspeção visual humana, destacando os sistemas de AOI como o método predominante para detecção de defeitos atualmente.

As máquinas de AOI capturam imagens de alta resolução das PCIs, utilizando um controle preciso de iluminação e posicionamento. Em seguida, aplicam algoritmos fundamentados na visão computacional clássica para identificar possíveis defeitos [Kim et al. 2021]. Embora ofereçam resultados satisfatórios, essas máquinas demandam configurações específicas para cada modelo de PCI, o que pode dificultar a agilidade e a adaptabilidade do processo de inspeção, características fundamentais na produção de PCIs [Zhou et al. 2023].

Como alternativa aos algoritmos clássicos de visão computacional, temos os métodos de Aprendizado Profundo (*Deep Learning*), os quais oferecem a capacidade de aprender representações complexas diretamente das imagens, permitindo uma detecção mais robusta, inclusive em contextos de grande variabilidade [Bochie et al. 2020]. Com uma base de dados diversificada, pode-se utilizar um modelo de Aprendizado Profundo para identificar defeitos em diferentes modelos de PCI, sem a necessidade de configurações específicas, trazendo agilidade e adaptabilidade ao processo de inspeção [Morais 2023].

Entre as técnicas de Aprendizado Profundo, destacam-se as redes neurais convolucionais (*Convolutional Neural Networks* - CNNs) por sua capacidade de aprender representações complexas de características visuais, sendo amplamente utilizadas para detecção e classificação de objetos [Ling and Isa 2023]. Um exemplo é a abordagem de detecção baseada em redes neurais convolucionais regionais (R-CNN) e suas variantes,

como Fast R-CNN e Faster R-CNN, que alcançaram resultados significativos em várias tarefas de detecção de objetos. Além disso, técnicas mais recentes, como o YOLO (*You Only Look Once*), têm demonstrado excelentes desempenhos em termos de precisão e velocidade de inferência [Jiao et al. 2019].

Este trabalho propõe a implementação do YOLO, em sua versão 10, para explorar um conjunto de imagens de defeitos em PCIs previamente utilizado em outros estudos. Além disso, o backbone original do YOLOv10 foi alterado com convoluções fantasmas (*Ghost Convolutions*), que utilizam operações computacionalmente leves para gerar parte das características da imagem, reduzindo o custo da rede. O objetivo é melhorar o desempenho e validar sua eficiência em diferentes aspectos, como precisão, tempo de inferência e otimização na quantidade de parâmetros da rede neural. Essa abordagem busca não apenas aprimorar o desempenho do modelo, mas também garantir sua aplicabilidade em cenários industriais.

Os resultados obtidos com os modelos modificados foram comparados com versões não otimizadas do YOLOv10 e com abordagens previamente publicadas na literatura, considerando métricas como mAP@50 e frames por segundo (FPS), possibilitando uma análise quantitativa das modificações propostas. Os experimentos realizados demonstraram que a modificação do backbone com convoluções fantasmas reduziu a complexidade computacional do modelo, resultando em uma diminuição no número de parâmetros e operações, sem comprometer a precisão. Esses resultados indicam que a técnica proposta pode ser uma alternativa viável para aplicações industriais, permitindo a implementação de modelos mais leves e eficientes para inspeção automatizada de PCIs em tempo real.

Este trabalho está estruturado da seguinte maneira: a Seção 2 revisa a literatura, a Seção 3 detalha a metodologia, a Seção 4 descreve os resultados alcançados e a Seção 5 conclui o artigo e sugere possíveis pesquisas futuras.

2. Trabalhos Relacionados

A detecção de defeitos em PCIs é um campo de pesquisa amplamente explorado, devido à sua importância para a qualidade e confiabilidade dos produtos eletrônicos. Nos últimos anos, diversos métodos e modelos de inteligência artificial, em especial as CNNs, têm sido desenvolvidos e aplicados para aprimorar a inspeção automatizada de PCIs. Esta seção revisa os principais avanços na área, destacando os datasets (conjunto de imagens) utilizados, os modelos de CNN implementados, suas métricas de desempenho e as limitações enfrentadas.

Um dos principais desafios na implementação de sistemas de inspeção baseados em IA é a disponibilidade de datasets robustos e diversificados, capazes de representar a ampla gama de defeitos que podem ocorrer em PCIs. O conjunto de dados (*dataset*) HRIPCB, apresentado no trabalho de [Huang et al. 2020], projetado para a detecção e classificação de defeitos em PCIs de alta resolução, é amplamente utilizado como referência em muitos estudos, servindo de base para a criação de variações e novos datasets em trabalhos subsequentes. Este dataset contém 1.386 imagens de alta resolução, capturando seis tipos de defeitos diferentes.

[Ding et al. 2019] introduziram o modelo TDD-Net, uma rede neural convolucional, projetada especificamente para a detecção de pequenos defeitos em PCIs. Este

modelo foi treinado em um dataset derivado do HRIPCB, modificado para incluir uma gama mais ampla de defeitos e permitir um treinamento mais eficiente. O TDD-Net alcançou um mAP (Mean Average Precision) de 98,9%, destacando-se pela sua precisão na detecção de defeitos em PCIs. No entanto, apesar da alta precisão, o TDD-Net requer considerável poder computacional, o que pode torná-lo menos adequado para aplicações em tempo real. Além disso, o *dataset* derivado deste trabalho também foi utilizado como referência em outros estudos [Niu et al. 2023, Park et al. 2023, Ancha et al. 2024], ampliando seu impacto na pesquisa de detecção de defeitos em PCIs.

[Ancha et al. 2024] apresentam uma abordagem baseada no YOLOv5 para detecção de defeitos em PCIs, utilizando o *dataset* MDD_PCB, que inclui defeitos intencionalmente induzidos para simular cenários realistas. O modelo alcançou uma mAP de 95%, destacando-se pela alta velocidade de inferência e capacidade de operar em tempo real em dispositivos como o Jetson Nano. Em outro experimento, utilizando o *dataset* de [Ding et al. 2019], o YOLOv5 obteve uma mAP de 92%. Apesar de sua eficiência e velocidade, o YOLOv5 apresenta limitações em cenários de produção devido à sua menor precisão em comparação a métodos como o TDD-Net.

O trabalho de [Park et al. 2023] oferece uma análise das técnicas de aprendizado profundo aplicadas à detecção de defeitos em PCIs, focando nos desafios associados à degradação de desempenho dos modelos em condições adversas. Neste trabalho, os autores utilizaram o *dataset* de [Ding et al. 2019] para treinar e avaliar diferentes modelos, incluindo o ResNet50 e o YOLOv7. Enquanto o ResNet50 alcançou uma acurácia de 98,3% em condições ideais, o desempenho caiu significativamente quando o modelo foi exposto a ruídos e outras formas de contaminação nas imagens. Por outro lado, o YOLOv7, apesar de sua robustez e rapidez, registrou um mAP de 97%, mas mostrou maior resiliência em ambientes com variabilidade e imperfeições.

Apesar dos avanços significativos proporcionados por modelos de IA como o TDD-Net e o YOLOv7, todos os métodos analisados apresentam limitações, seja na demanda por recursos computacionais, na velocidade de inferência ou na capacidade de generalizar para novos tipos de defeitos. Nesse contexto, o presente trabalho propõe a aplicação do YOLO na sua décima versão, apresentando uma nova abordagem que busca otimizar a precisão e a eficiência dos sistemas de inspeção de PCIs.

3. Materiais e Métodos

3.1. Dataset

O *dataset* utilizado neste trabalho é uma variação do *dataset* HRIPCB, apresentado por [Ding et al. 2019]. O *dataset* HRIPCB é composto por imagens de alta resolução (2.777 x 2.138 pixels) e abrange seis tipos de defeitos: buracos ausentes (*missing hole*), circuitos abertos (*open circuit*), curtos-circuitos (*short*) e falhas na trilha de cobre que podem ser especificadas como *mousebite, spur, spurious copper*, exemplificados na Figura 1. Para aprimorar a diversidade e a robustez do conjunto de dados para treinamento, [Ding et al. 2019] aplicaram técnicas de aumento de dados, incluindo alterações de iluminação, rotações, espelhamentos e recortes aleatórios. As imagens originais também foram segmentadas em subimagens de 600 x 600 pixels, resultando em 10.668 imagens, com as classes de defeitos distribuídas de forma balanceada, conforme descrito na Tabela 1.

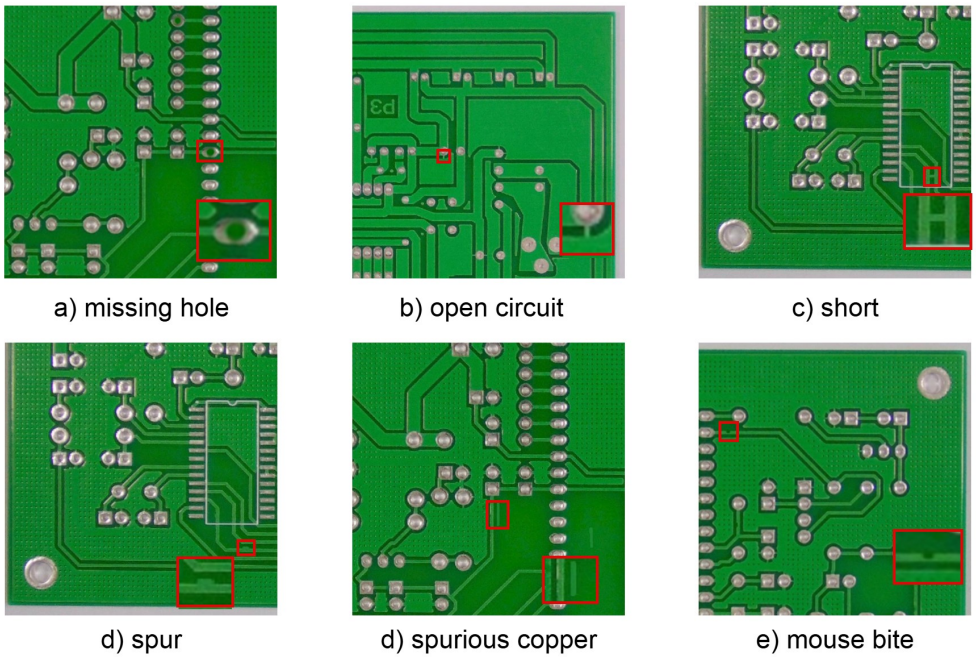

Figura 1. Exemplos de defeitos.

Tipos de Defeito	Quantidade de Imagens	Número de Defeitos
missing hole	1.832	3.612
mouse bite	1.852	3.684
open circuit	1.740	3.548
short	1.732	3.508
spur	1.752	3.636
spurious copper	1.760	3.676
Total	10.668	21.664

Tabela 1. Distribuição dos defeitos no dataset de [Ding et al. 2019].

3.2. Métricas de Avaliação

A avaliação do desempenho do modelo YOLOv10 é realizada por meio das métricas de precisão (*P*), revocação (*R*), mAP@50, mAP@50-90 e tempo de inferência. Além disso, detalhes do modelo, como quantidade de parâmetros, gradientes, camadas e *Giga Floating Point Operations per Second* (GFLOPs) também foram considerados.

A precisão mede a proporção de detecções corretas em relação ao total de predições, enquanto a revocação mede a capacidade do modelo de identificar todos os defeitos presentes na imagem. A métrica mAP@50 (*Mean Average Precision*) calcula a média das precisões para cada classe considerando um limiar de 50% de Interseção sobre União (IoU), enquanto o mAP@50-90 avalia a precisão em múltiplos limiares de IoU (50% a 90%), oferecendo uma análise mais detalhada do desempenho do modelo.

O tempo de inferência será avaliado em frames por segundo (FPS), indicando quantas imagens o modelo consegue processar em um segundo. Essa métrica indica se o modelo é capaz de processar em tempo real as imagens, onde um alto FPS significa maior

eficiência no fluxo de inspeção. O cálculo do FPS é realizado pela equação:

$$FPS = \frac{N}{T} \tag{1}$$

Onde N representa o número total de imagens processadas e T é o tempo total de inferência em segundos. Modelos com menor número de parâmetros tendem a apresentar um FPS mais alto, enquanto modelos maiores, com mais camadas e complexidade, resultam em um FPS reduzido. Logo, o número de parâmetros está diretamente relacionado ao seu custo computacional, que pode ser expresso em GFLOPs. O GFLOPs mede a quantidade de operações de ponto flutuante realizadas pelo modelo para processar uma única imagem.

Para este trabalho, serão enfatizadas as métricas mAP@50 e FPS. O mAP@50 será utilizado como referência por sua adoção em estudos anteriores [Ancha et al. 2024, Niu et al. 2023], permitindo comparações diretas com outros modelos de detecção de defeitos em PCIs. Já a métrica de FPS será para avaliar o impacto da alteração no *backbone*, indicando se as modificações implementadas resultaram em uma melhoria no desempenho do modelo. A combinação dessas métricas possibilita uma análise equilibrada entre desempenho e eficiência computacional, garantindo que a solução proposta seja precisa e rápida o suficiente para a inspeção automatizada em ambientes produtivos.

3.3. Modelo Utilizado

O YOLO é uma das arquiteturas mais populares para detecção de objetos em tempo real, conhecida por sua eficiência e velocidade. Desde sua primeira versão, o YOLO tem evoluído, incorporando avanços que melhoram tanto a precisão quanto a capacidade de generalização do modelo. Sua abordagem baseada em detecção única permite a localização e classificação de múltiplos objetos em uma única passagem pela rede neural, tornando-o ideal para aplicações que exigem inferência rápida [Jiang et al. 2022].

O YOLOv10 representa um avanço significativo na detecção de objetos em tempo real, introduzindo melhorias notáveis em eficiência e precisão. Uma das principais inovações é a eliminação da necessidade de Supressão de Não-Máximos (NMS) durante a inferência, alcançada por meio de uma estratégia de atribuições duplas consistentes durante o treinamento. Essa abordagem não apenas simplifica o processo de inferência, mas também reduz a latência, tornando o modelo mais adequado para aplicações que exigem respostas rápidas [Hussain 2024, Wang et al. 2024]. Além disso, o YOLOv10 apresenta diferentes variantes de modelo para atender a requisitos específicos de desempenho e custo computacional. Entre essas variantes, destacam-se os modelos *Nano* (n), *Small* (s), *Medium* (m) e *Large* (l), cada um ajustado para um equilíbrio entre precisão, velocidade e consumo de recursos computacionais. O modelo *Nano*, por exemplo, é projetado para dispositivos embarcados, oferecendo inferência rápida com menor uso de memória, enquanto as versões *Medium* e *Large* são voltadas para cenários que exigem maior precisão na detecção.

No que diz respeito à arquitetura, o YOLOv10 utiliza uma versão aprimorada da CSPNet (*Cross Stage Partial Network*) como *backbone* para extração de características. Essa modificação melhora o fluxo de gradientes e reduz a redundância computacional,

resultando em uma extração de características mais eficaz [Hussain 2024]. Neste trabalho, a camada de backbone foi modificada para utilizar a convolução fantasma (*Ghost Convolution*), uma técnica que permite a geração de mais características de entrada com um custo computacional reduzido. Essa abordagem visa melhorar o tempo de inferência, mantendo a precisão da rede.

A convolução fantasma foi introduzida por [Han et al. 2020] como uma técnica para reduzir a redundância de cálculos em redes neurais profundas, mantendo a representatividade dos dados. Em vez de calcular diretamente todas as características com convoluções padrão, a convolução fantasma gera uma parte das características utilizando operações computacionalmente leves, como transformações lineares e convoluções em profundidade. Isso resulta em um modelo mais leve, reduzindo o número de parâmetros e operações necessárias para a inferência.

No presente trabalho, essa técnica foi aplicada ao YOLOv10 para otimizar o backbone dos modelos *Nano*, *Small* e *Medium*, reduzindo o custo computacional. A Tabela 2 apresenta uma comparação entre os pesos originais dos modelos YOLOv10 e as versões otimizadas com convolução fantasma.

Modelo	Parâmetros Originais	Parâmetros Otimizados
YOLOv10n	2.7M - 8.2 GFLOPs	2.31M - 6.6 GFLOPs
YOLOv10s	8.04M - 24.5 GFLOPs	7.37M - 18.3 GFLOPs
YOLOv10m	16.5M - 64 GFLOPs	12.8M - 38.6 GFLOPs

Tabela 2. Pesos dos YOLOv10 originais e otimizados com *Ghost Convolution*.

4. Resultados

Para o treinamento dos modelos, o conjunto de dados foi dividido nas proporções de 80% para treinamento, 10% para teste e 10% para validação. Essa divisão resultou em um total de 8.532 imagens destinadas ao treinamento, 1.070 imagens para teste e 1.066 imagens para validação. Durante o treinamento, os seguintes hiperparâmetros foram definidos: 500 épocas, tamanho de imagem de 640 *pixels*, *batch size* de 16. Além disso, foram aplicadas técnicas de aumento de dados, aplicadas pela própria biblioteca do YOLO antes do treinamento, incluindo ajustes na saturação e valor do espaço de cores HSV (*hsv_h = 0.015, hsv_s = 0.7, hsv_v = 0.4*), rotações de até 15 graus, deslocamentos de 10% e variações de escala de até 50%.

O treinamento foi realizado em um ambiente configurado com a biblioteca Ultralytics YOLOv8.2.84, executando em Python 3.12.7 e utilizando PyTorch 2.4.0 com suporte à CUDA 12.1. A GPU utilizada foi uma NVIDIA GeForce RTX 3060 (12 GB). Com essa configuração, o tempo necessário para o treinamento completo variou conforme o tamanho do modelo: o YOLOv10 *Nano* e sua versão otimizada levaram aproximadamente 14 horas, já o YOLOv10 *Small* exigiu cerca de 21 horas, com sua versão otimizada levando cerca de 20 horas, enquanto o YOLOv10 *Medium* levou cerca de 39 horas, com sua versão otimizada levando 32 horas.

Os resultados obtidos para os modelos avaliados são apresentados na Tabela 3. Nota-se que tanto o YOLOv10 *Nano*, *Small* e *Medium* otimizados com convolução fantasma mantiveram os valores elevados para precisão, revocação e mAP@50, destacando

sua robustez na detecção dos defeitos. A inserção da convolução fantasma no backbone proporcionou especialmente ao modelo *Small* uma melhoria significativa no FPS, passando de 128 para 150, indicando um aumento na velocidade de inferência sem comprometer o mAP@50, que subiu de 99,11% para 99,22%.

Modelo	P(%)	R(%)	mAP@50(%)	mAP@50-90(%)	FPS
YOLOv10n	97,95	97,44	99,03	65,19	283
YOLOv10n+*Ghost*	98,33	98,16	99,09	63,74	280
YOLOv10s	98,48	98,19	99,11	71,88	128
YOLOv10s+*Ghost*	98,53	98,49	99,22	69,83	150
YOLOv10m	98,70	98,40	99,00	72,70	64,5
YOLOv10m+*Ghost*	98,50	98,30	99,10	72,50	82

Tabela 3. Resultado dos experimentos por modelo.

Ao analisar o desempenho por tipo de defeito, detalhado na Tabela 4, observa-se que os modelos tiveram desempenho consistente e elevado em praticamente todos os defeitos. Esses resultados reforçam que a otimização realizada na arquitetura do modelo manteve a performance dos modelos em diferentes classes de defeitos.

Defeitos	mAP@50(%)					
	YOLOv10n	YOLOv10n+	YOLOv10s	YOLOv10s+	YOLOv10m	YOLOv10m+
missing hole	99,4	99,4	99,4	99,4	99,4	99,3
mouse bite	98,5	99,0	99,1	99,2	99,1	98,7
open circuit	99,3	99,4	99,4	99,4	99,4	99,4
short	98,4	97,8	97,9	98,4	97,7	98,1
spur	99,1	99,4	99,4	99,4	99,0	99,5
spurious copper	99,5	99,5	99,5	99,5	99,5	99,5

Tabela 4. Resultado dos experimentos por modelo e tipos de defeito.

Na comparação com trabalhos anteriores, resumidos na Tabela 5, os modelos propostos alcançaram desempenho superior em relação aos modelos já consolidados na literatura recente. Em termos de precisão e velocidade, o YOLOv10 Small otimizado mostrou-se mais eficiente que o YOLOv5 de [Niu et al. 2023] (mAP@50 de 99,1% com 86 FPS) e YOLOv7 modificado por [Chen and Dang 2023] (mAP@50 de 97,5% com 83,3 FPS). Essa combinação de alta precisão e velocidade reforça o potencial da abordagem proposta para aplicação prática em linhas de produção industriais.

Modelo	mAP@50 (%)	FPS
YOLOv5 [Niu et al. 2023]	99,10	86
YOLOv7 [Chen and Dang 2023]	97,50	83,3
YOLOv10n+Ghost (proposto)	99,09	280
YOLOv10s+Ghost (proposto)	**99,22**	**150**
YOLOv10m+Ghost (proposto)	99,10	82

Tabela 5. Comparação de resultados com modelos anteriores.

5. Conclusão

A implementação do YOLOv10 com o backbone modificado utilizando a técnica de convolução fantasma apresentou resultados expressivos, demonstrando alta eficiência e excelente desempenho em termos de mAP@50 e FPS. Essa adaptação comprovou-se eficaz, reduzindo a complexidade dos modelos *Nano*, *Small* e *Medium* sem comprometer a precisão e revocação, possibilitando sua aplicação prática em cenários industriais onde a velocidade de inferência é fundamental. Além disso, a utilização da convolução fantasma revelou-se uma importante contribuição deste trabalho, destacando seu potencial para reduzir o peso das redes neurais mantendo a precisão, com potencial aplicação em outras arquiteturas e em diferentes contextos de inspeção visual.

Para trabalhos futuros, recomenda-se estender a aplicação da técnica proposta neste estudo para modelos mais robustos do YOLO, como a versão *Large*. Também é sugerido realizar testes em dispositivos embarcados, como a linha Jetson da Nvidia, simulando ambientes reais da indústria, a fim de validar a viabilidade operacional e o desempenho dos modelos otimizados em situações que demandam processamento rápido e preciso.

6. Agradecimentos

Este trabalho foi apoiado pelo Instituto de Desenvolvimento Tecnológico (INDT), no âmbito do Projeto VISIONAI4ALL e sob os termos da Lei Federal nº 8.387/91. Os autores também agradecem ao Programa de Pós-Graduação em Engenharia Elétrica (PPGEEL) da Universidade do Estado do Amazonas (UEA) pelo apoio institucional e aos professores pelas contribuições nas disciplinas ministradas.

Referências

Ancha, V. K., Sibai, F. N., Gonuguntla, V., and Vaddi, R. (2024). Utilizing yolo models for real-world scenarios: Assessing novel mixed defect detection dataset in pcbs. *IEEE Access*.

Bhattacharya, A. and Cloutier, S. G. (2022). End-to-end deep learning framework for printed circuit board manufacturing defect classification. *Scientific reports*, 12(1):12559.

Bochie, K., da Silva Gilbert, M., Gantert, L., Barbosa, M. d. S. M., de Medeiros, D. S. V., and Campista, M. E. M. (2020). Aprendizado profundo em redes desafiadoras: Conceitos e aplicações. *Sociedade Brasileira de Computação*.

Chen, B. and Dang, Z. (2023). Fast pcb defect detection method based on fasternet backbone network and cbam attention mechanism integrated with feature fusion module in improved yolov7. *Ieee Access*, 11:95092–95103.

Chen, X., Wu, Y., He, X., and Ming, W. (2023). A comprehensive review of deep learning-based pcb defect detection. *IEEE Access*.

Ding, R., Dai, L., Li, G., and Liu, H. (2019). Tdd-net: a tiny defect detection network for printed circuit boards. *CAAI Transactions on Intelligence Technology*, 4(2):110–116.

Fonseca, L. A. L. O., Iano, Y., Oliveira, G. G. d., Vaz, G. C., Carnielli, G. P., Pereira, J. C., and Arthur, R. (2024). Automatic printed circuit board inspection: a comprehensible survey. *Discover Artificial Intelligence*, 4(1):10.

Han, K., Wang, Y., Tian, Q., Guo, J., Xu, C., and Xu, C. (2020). Ghostnet: More features from cheap operations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Huang, W., Wei, P., Zhang, M., and Liu, H. (2020). Hripcb: a challenging dataset for pcb defects detection and classification. *The Journal of Engineering*, 2020(13):303–309.

Hussain, M. (2024). Yolov5, yolov8 and yolov10: The go-to detectors for real-time vision. *arXiv preprint arXiv:2407.02988*.

Jiang, P., Ergu, D., Liu, F., Cai, Y., and Ma, B. (2022). A review of yolo algorithm developments. *Procedia computer science*, 199:1066–1073.

Jiao, L., Zhang, F., Liu, F., Yang, S., Li, L., Feng, Z., and Qu, R. (2019). A survey of deep learning-based object detection. *IEEE access*, 7:128837–128868.

Kim, J., Ko, J., Choi, H., and Kim, H. (2021). Printed circuit board defect detection using deep learning via a skip-connected convolutional autoencoder. *Sensors*, 21(15):4968.

Ling, Q. and Isa, N. A. M. (2023). Printed circuit board defect detection methods based on image processing, machine learning and deep learning: A survey. *IEEE Access*, 11:15921–15944.

Morais, L. E. C. (2023). Detecção de defeitos de fabricação em placas de circuito impresso através de visão computacional e aprendizado profundo. *[sn]*.

Niu, J., Li, H., Chen, X., and Qian, K. (2023). An improved yolov5 network for detection of printed circuit board defects. *Journal of Sensors*, 2023(1):7270093.

Park, J.-H., Kim, Y.-S., Seo, H., and Cho, Y.-J. (2023). Analysis of training deep learning models for pcb defect detection. *Sensors*, 23(5):2766.

Rodrigues, R. B. (2023). *O uso de redes neurais artificiais na análise óptica automática e detecção de defeitos em circuitos impressos*. PhD thesis, [sn].

Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z., Han, J., and Ding, G. (2024). Yolov10: Real-time end-to-end object detection. *arXiv preprint arXiv:2405.14458*.

Zhou, Y., Yuan, M., Zhang, J., Ding, G., and Qin, S. (2023). Review of vision-based defect detection research and its perspectives for printed circuit board. *Journal of Manufacturing Systems*, 70:557–578.