

Aplicação de Redes Neurais Convolucionais para Detecção, Classificação e Denoising de Imagens de Aeronaves Geradas por Satélite

Gabriel Alves Baltazar

Departamento de Telecomunicações, Universidade Estadual de Campinas,

Faculdade de Tecnologia, Limeira, São Paulo, Brasil

e-mail: g234628@dac.unicamp.br

Resumo - Este trabalho propõe o desenvolvimento de um workflow completo para detecção, classificação e remoção de ruído em imagens de aeronaves obtidas por satélite. Na etapa de detecção, é utilizada a arquitetura YOLO11 para identificar aeronaves em diferentes cenários e condições de imagem. As regiões detectadas são então submetidas a um modelo de classificação baseado em duas arquiteturas de redes neurais convolucionais distintas — EfficientNet e DenseNet121 — com o objetivo de reconhecer oito classes de aeronaves militares de diferentes categorias operacionais. Por fim, é implementado um modelo de remoção de ruído (denoising) baseado na arquitetura U-Net, voltado para atenuar ruído gaussiano e melhorar a qualidade das imagens processadas. O pipeline proposto busca integrar de forma unificada etapas fundamentais de análise de imagens aéreas, contribuindo para aplicações em vigilância, monitoramento e inteligência aeroespacial.

Palavras-chaves: Visão Computacional, Redes Neurais Convolucionais (CNNs), Detecção de Aeronaves, Denoising de Imagens, Imagens de Satélite.

1 INTRODUÇÃO

O avanço das tecnologias de sensoriamento remoto e o aumento da disponibilidade de imagens de satélite de alta resolução têm impulsionado uma ampla gama de aplicações no domínio aeroespacial. Entre essas aplicações, a análise automatizada de imagens para **detecção e reconhecimento de aeronaves** tem se mostrado de grande relevância em contextos de **segurança, defesa, vigilância e inteligência estratégica**. A identificação precisa de aeronaves em imagens orbitais pode subsidiar decisões táticas, apoiar o monitoramento de áreas críticas e contribuir para o planejamento de operações militares ou humanitárias.

Entretanto, essa tarefa apresenta desafios significativos. As imagens de satélite frequentemente sofrem com *ruídos, variações de iluminação e ângulo, diferenças de escala e sombras projetadas*, além de conterem aeronaves parcialmente ocultas ou sobrepostas a outras estruturas. Essas condições tornam o processo de detecção e classificação visual uma tarefa complexa, exigindo modelos de aprendizado profundo robustos e bem treinados.

Neste contexto, o presente trabalho propõe o desenvolvimento de um **workflow completo e integrado** para análise de imagens de satélite contendo aeronaves, dividido em três etapas principais:

1. **Detecção de aeronaves** utilizando o modelo **YOLO11**, uma rede neural de detecção em tempo real otimizada para alto desempenho e precisão;
2. **Classificação das aeronaves detectadas** em **oito classes específicas de aeronaves militares** — A10, B1, B2, B52, C17, E3, F22 e U2 — por meio de duas arquiteturas distintas de redes neurais convolucionais: **EfficientNet** e **DenseNet121**;

3. **Remoção de ruído (denoising)** em imagens corrompidas por ruído gaussiano, utilizando uma arquitetura **U-Net**, amplamente empregada em tarefas de restauração e segmentação de imagens.

A proposta busca não apenas avaliar o desempenho individual de cada etapa, mas também demonstrar a **viabilidade de integração** entre modelos de detecção, classificação e restauração dentro de um mesmo pipeline de visão computacional voltado ao domínio aeroespacial.

O artigo está organizado da seguinte forma: a **Seção 2** apresenta os trabalhos relacionados e a base teórica sobre as arquiteturas utilizadas; a **Seção 3** descreve a metodologia adotada, incluindo o preparo dos dados, treinamento e avaliação dos modelos; a **Seção 4** discute os resultados obtidos e as análises comparativas entre as abordagens propostas; e, por fim, a **Seção 5** traz as conclusões e perspectivas para trabalhos futuros.

2 TRABALHOS RELACIONADOS E FUNDAMENTAÇÃO TEÓRICA

Nesta seção são apresentados os principais conceitos e arquiteturas de redes neurais utilizadas neste trabalho, com ênfase nas abordagens de detecção, classificação e restauração de imagens aplicadas ao domínio aeroespacial. São descritas as arquiteturas YOLO11, EfficientNet, DenseNet121 e U-Net, bem como suas características e motivações para utilização neste projeto.

2.1 YOLO11 para Detecção de Aeronaves

Modelos da família *You Only Look Once* (YOLO) são amplamente reconhecidos pela sua eficiência em tarefas de detecção de objetos em tempo real. A abordagem proposta originalmente em [1] reformulou o problema de detecção como uma única tarefa de regressão, permitindo que a rede previsse simultaneamente as coordenadas dos *bounding boxes* e as classes dos objetos.

O YOLO11 é a iteração mais recente da família de detectores de objetos em tempo real desenvolvida pela Ultralytics, sucedendo o YOLOv8. Esta versão introduz avanços significativos na arquitetura, incluindo novos módulos de *backbone* e *neck*, camadas de atenção e otimizações obtidas por meio de *neural architecture search* (NAS). Essas melhorias ampliam a capacidade de extração de características, resultando em maior precisão na detecção — especialmente para objetos pequenos, como aeronaves em imagens de satélite.

Segundo a Ultralytics, o YOLO11 atinge uma *mean Average Precision* (mAP) superior ao YOLOv8m, utilizando cerca de 22% menos parâmetros, o que o torna mais eficiente computacionalmente sem comprometer o desempenho. Além disso, seu design refinado permite aplicações versáteis em diferentes ambientes — desde dispositivos de borda até servidores em nuvem com GPUs NVIDIA — e suporta múltiplas tarefas de visão computacional, como detecção, segmentação de instâncias, classificação e estimativa de pose.

Dessa forma, o YOLO11 foi adotado neste trabalho pela sua **robustez, velocidade de inferência e alta acurácia** em cenários complexos e sujeitos a ruído visual. A Figura 1 apresenta a evolução da *COCO mAP(50–95)* entre as diferentes versões da família YOLO, evidenciando o ganho de desempenho obtido pelo YOLO11 em relação às iterações anteriores.

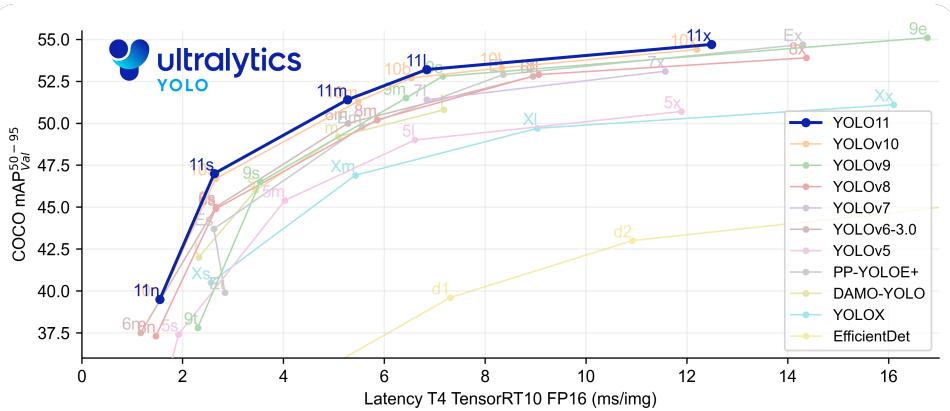


Figure 1: Comparação de desempenho da YOLO

2.2 EfficientNet e DenseNet121 para Classificação de Aeronaves

Após a detecção, as aeronaves extraídas das imagens de satélite são classificadas em diferentes categorias de modelos militares. Para essa etapa, foram empregadas duas arquiteturas de redes neurais convolucionais de destaque: **EfficientNet** e **DenseNet121**.

A **EfficientNet**, proposta por [2], introduz o conceito de *compound scaling*, uma abordagem sistemática que busca equilibrar simultaneamente a profundidade, a largura e a resolução da rede. Diferente das abordagens tradicionais, que ampliam apenas um desses fatores de forma isolada, o *compound scaling* aplica um coeficiente de escala uniforme, resultando em uma expansão mais eficiente e coerente da arquitetura.

Além disso, os autores empregaram *neural architecture search* (NAS) para projetar uma rede base otimizada, que posteriormente foi escalonada para originar uma família de modelos — *EfficientNet-B0* a *B7*. Essa combinação de busca automática de arquitetura e escalonamento composto levou a uma série de redes que atingem estado da arte em acurácia e eficiência.

A Figura 2 ilustra a relação entre a acurácia *Top-1* no ImageNet e o número de parâmetros (em milhões) para diferentes arquiteturas de redes neurais convolucionais, evidenciando a eficiência da família *EfficientNet*, que alcança alto desempenho com significativamente menos parâmetros em comparação a modelos como ResNet e Inception.

A **DenseNet121** [3], diferencia-se pelo uso de conexões densas entre as camadas, em que cada camada recebe como entrada os mapas de características de todas as camadas anteriores. Essa estrutura promove melhor propagação de gradientes, reutilização de características e maior eficiência paramétrica. No contexto deste trabalho, a DenseNet121 foi empregada como arquitetura complementar à EfficientNet, permitindo comparar desempenho entre modelos de natureza estrutural distinta.

2.3 U-Net para Denoising de Imagens de Satélite

A etapa final do pipeline proposto consiste na remoção de ruídos (denoising) presentes nas imagens de aeronaves, especialmente o ruído gaussiano adicionado artificialmente para fins de avaliação. Para essa tarefa, foi utilizada a arquitetura **U-Net** [4], proposta originalmente para segmentação biomédica, mas amplamente adaptada para restauração e filtragem de imagens.

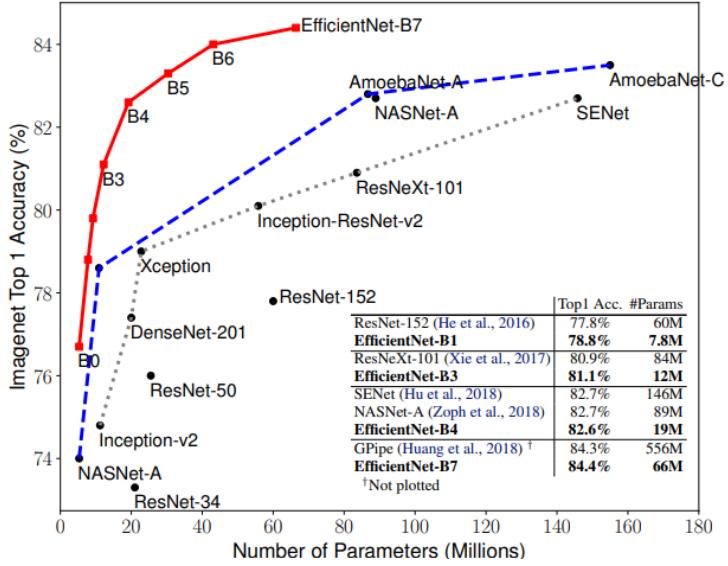


Figure 2: Acurácia Top-1 no ImageNet versus número de parâmetros para diferentes arquiteturas.

A U-Net é caracterizada por sua estrutura simétrica em formato de “U”, composta por um caminho de contração (encoder) e um caminho de expansão (decoder), interligados por conexões de *skip*. Essas conexões preservam detalhes espaciais e permitem reconstruções de alta fidelidade, a Figura 3 mostra a arquitetura da rede. No contexto aeroespacial, o uso da U-Net visa aprimorar a qualidade das imagens pós-classificação, facilitando análises posteriores e reduzindo o impacto de ruídos de captura e compressão.

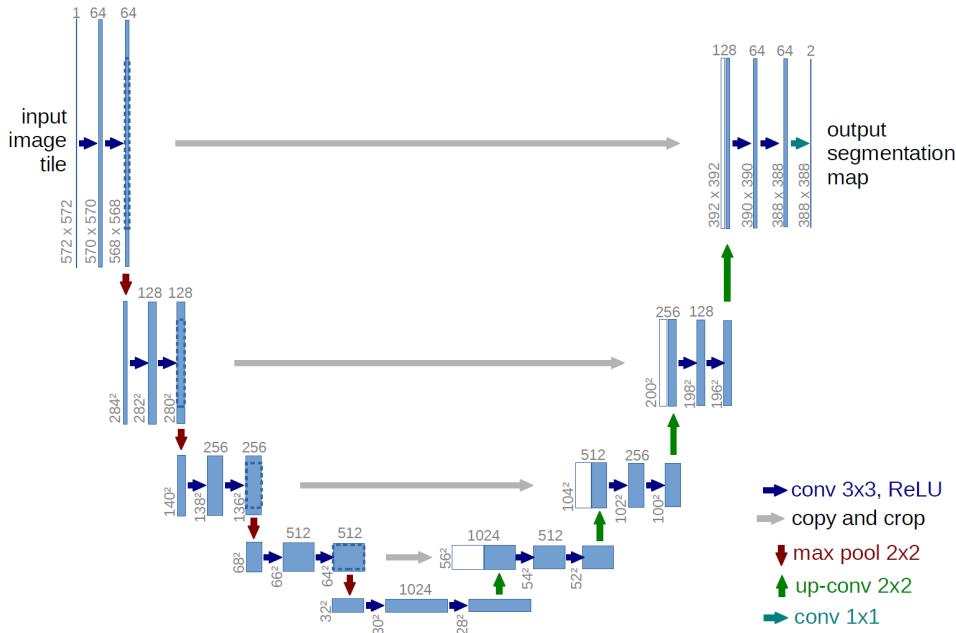


Figure 3: Arquitetura U-Net.

3 METODOLOGIA

Nesta seção, descrevemos o fluxo completo de desenvolvimento das tarefas abordadas neste trabalho. Para cada tarefa, detalhamos os datasets utilizados, o pré-processamento aplicado, a configuração dos arquivos e parâmetros dos modelos, bem como os procedimentos de treinamento e validação adotados.

Todas as etapas foram realizadas no ambiente do *Google Colab*, utilizando GPUs disponíveis de forma gratuita, o que permitiu acelerar o treinamento e a inferência sem a necessidade de hardware local dedicado. O uso dessa infraestrutura também possibilitou o manuseio de datasets de tamanho moderado e a experimentação com diferentes arquiteturas de redes neurais convolucionais, mantendo a reproduzibilidade e a eficiência computacional.

3.1 Detecção de Aeronaves

Para a tarefa de detecção de aeronaves, foi utilizada a arquitetura **YOLO11** com a implementação oficial disponibilizada pela *Ultralytics*. Foram empregados dois *datasets* públicos disponíveis no *Kaggle* [5, 6], ambos contendo imagens de aeronaves obtidas por sensoriamento remoto, acompanhadas de arquivos de anotação no formato `.txt` com as coordenadas dos *bounding boxes* para detecção. Inicialmente, os conjuntos foram unificados, garantindo a compatibilidade entre os rótulos de classe — todos normalizados para uma única categoria aeronave. Após a fusão, as imagens foram divididas em três subconjuntos: 4.321 imagens para treinamento, 249 para validação e 249 para teste. A Figura 4 apresenta exemplos representativos das imagens utilizadas no processo de detecção.

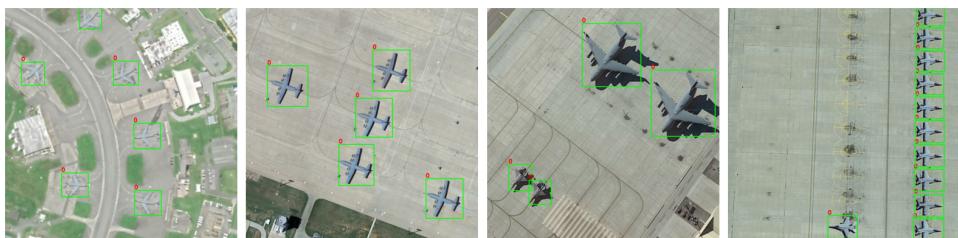


Figure 4: Exemplo de imagens do Dataset 1 com anotações de detecção.

O modelo **YOLO11x** foi treinado por **100 épocas**, com tamanho de imagem de **320×320 pixels** e **tamanho de lote (batch size) igual a 16**. O otimizador adotado foi o **NAdam**, com taxa de aprendizado inicial de **0.001** e taxa final de **0.01**, empregando estratégia de decaimento adaptativo. O treinamento utilizou **8 trabalhadores (workers)** para paralelismo no carregamento dos dados e incluiu **dropout de 0.3** para regularização. Além disso, o recurso de *mosaic augmentation* foi desativado nas 10 últimas épocas, e um critério de **paciente de 50 épocas** foi configurado para parada antecipada (*early stopping*).

As principais métricas utilizadas para avaliar o desempenho do modelo foram a mAP@0.5, a mAP@0.5:0.95, além das métricas médias de precisão e revocação, conforme implementadas na biblioteca Ultralytics. A métrica mAP@0.5 representa a média da precisão média considerando um limiar de Intersection over Union (IoU) de 0.5, enquanto a mAP@0.5:0.95 calcula a média sobre múltiplos limiares de IoU, fornecendo uma avaliação mais abrangente da capacidade de detecção do modelo.

3.2 Classificação de Aeronaves Militares

Para a tarefa de **classificação**, foi utilizado um *dataset* público disponível na plataforma **Kaggle**, contendo **4.369 imagens de aeronaves militares** divididas em **oito classes distintas**, correspondentes a diferentes modelos de aeronaves (*A10, B1, B2, B52, C17, E3, F22 e U2*). O conjunto de dados foi particionado em **treinamento (3.072 imagens)**, **validação (640 imagens)** e **teste (672 imagens)**, mantendo a proporcionalidade entre as classes. A Figura 5 apresenta um exemplo de imagem representativa de cada classe do dataset.



Figure 5: Exemplo representativo de cada classe do dataset de aeronaves militares utilizado para classificação. Cada imagem ilustra um modelo distinto de aeronave, evidenciando variações de perspectiva e escala.

Duas arquiteturas de redes neurais convolucionais (CNNs) foram adotadas para fins comparativos: **EfficientNetV2S** e **DenseNet121**, ambas com pesos pré-treinados no conjunto *ImageNet*. O uso de redes pré-treinadas (estratégia de *transfer learning*) visa acelerar a convergência e melhorar a generalização, dado o tamanho moderado do conjunto de dados.

As implementações das arquiteturas foram realizadas utilizando o framework **TensorFlow**, que oferece suporte nativo para construção, treinamento e avaliação de modelos baseados em aprendizado profundo, além de permitir o uso eficiente de aceleração por GPU. As camadas superiores das redes de classificação foram substituídas por uma nova *head*, composta por uma operação de **Global Average Pooling**, seguida de uma camada **densa com 256 neurônios e ativação ReLU**, e uma camada de saída **Softmax** com **oito unidades**, correspondentes às classes de aeronaves. O modelo foi compilado utilizando o **otimizador Adam** e a função de perda **sparse categorical cross-entropy**, adequada para problemas de classificação multiclasse com rótulos inteiros.

Durante o treinamento, que ocorreu por **100 épocas**, foram empregadas estratégias de otimização e regularização por meio de *callbacks*. O **ModelCheckpoint** salvou os melhores pesos com base na acurácia de validação, o **ReduceLROnPlateau** reduziu a taxa de aprendizado em 30% após cinco épocas sem melhoria e o **EarlyStopping** interrompeu o treinamento após 20 épocas de estagnação, restaurando os melhores pesos obtidos.

A avaliação dos modelos foi conduzida no **conjunto de teste**, utilizando métricas clássicas de classificação: **acurácia, precisão, revocação, F1-score e matriz de confusão**. Além disso, foram gerados gráficos de evolução da *loss* e da acurácia ao longo das épocas.

3.3 Denoising de Imagens de Aeronaves Militares

Nesta etapa do pipeline proposto, foi desenvolvido um modelo de **remoção de ruído (denoising)** em imagens de aeronaves utilizando a arquitetura **U-Net**. O objetivo é reduzir o ruído gaussiano adicionado artificialmente às imagens, preservando as estruturas e contornos característicos das aeronaves.

O *dataset* empregado nesta tarefa é o mesmo utilizado no módulo de classificação, garantindo consistência entre as diferentes etapas do pipeline (detecção → classificação → denoising). Para otimizar o processamento e reduzir a complexidade computacional, todas as imagens foram convertidas para **escala de cinza**, com dimensão final de 256×256 pixels, mantendo os elementos estruturais essenciais para a reconstrução.

O processo de preparação dos dados compreendeu as seguintes etapas: (1) carregamento das imagens originais, utilizadas como base limpa (*clean dataset*); (2) conversão para escala de cinza, reduzindo a dimensionalidade de três para um canal; (3) adição de ruído gaussiano controlado, simulando degradações realistas (*noisy dataset*); e (4) divisão dos dados em **treinamento (80%)** e **validação (20%)**, preservando a distribuição original das classes.

A Figura 6 ilustra exemplos de imagens do conjunto de dados antes e após a adição do ruído gaussiano, evidenciando o impacto visual do processo de degradação e a necessidade da etapa de *denoising*.

A arquitetura implementada com o **framework TensorFlow** foi configurada com profundidade de **4 níveis, 32 filtros** na primeira camada convolucional (dobrando a cada etapa da codificação), função de ativação **ReLU**, normalização em lote (*Batch Normalization*) e camadas opcionais de *Dropout* para regularização. A entrada do modelo é uma imagem de dimensão $256 \times 256 \times 1$, e a saída corresponde à versão reconstruída da mesma imagem, com o ruído atenuado.

O treinamento foi conduzido utilizando a função de perda baseada no **Mean Squared Error (MSE)**, adequada para medir discrepâncias de intensidade pixel a pixel. O otimizador **Adam** foi empregado com taxa de aprendizado inicial de 10^{-3} , e as seguintes estratégias de controle foram aplicadas: **ReduceLROnPlateau** (redução adaptativa da taxa de aprendizado em fator de 0.5 após 5 épocas sem melhoria na validação), **EarlyStopping** (interrupção antecipada após 15 épocas sem melhoria significativa, restaurando os melhores pesos), e **ModelCheckpoint** (salvamento automático do melhor modelo). O treinamento foi executado por até **200 épocas**, com *batch size* igual a 8.

A avaliação do modelo foi realizada por meio das métricas **PSNR (Peak Signal-to-Noise Ratio)** e **SSIM (Structural Similarity Index)**, que mensuram, respectivamente, a fidelidade e a similaridade

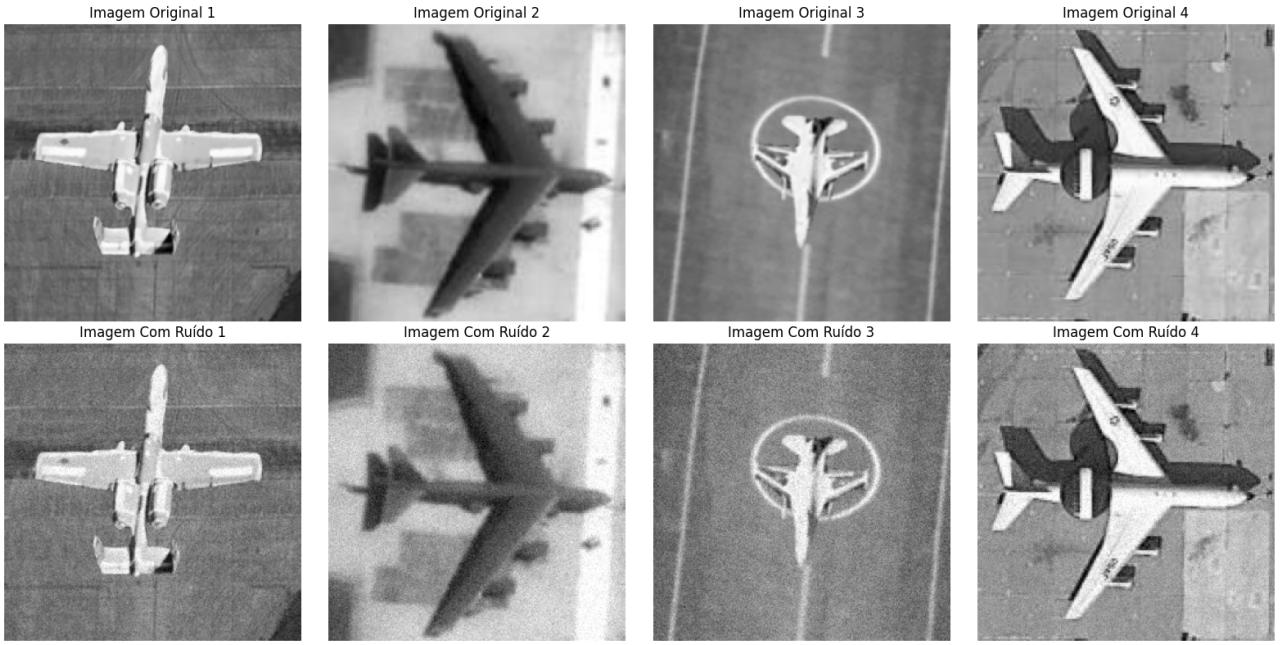


Figure 6: Exemplos de amostras do conjunto de dados utilizado na tarefa de *denoising*. Primeira linha: imagens originais (sem ruído). Segunda linha: imagens com ruído gaussiano adicionado.

estrutural entre a imagem reconstruída e a imagem original.

4 RESULTADOS E DISCUSSÕES

Esta seção apresenta e analisa os resultados obtidos nas diferentes etapas do pipeline proposto. O objetivo é avaliar o desempenho dos modelos implementados tanto de forma quantitativa, por meio de métricas objetivas (por exemplo, mAP, acurácia, PSNR e SSIM), quanto qualitativa, através da inspeção visual das predições e reconstruções. As análises buscam evidenciar as contribuições de cada componente do sistema — desde a utilização do YOLO11 para localizar aeronaves em imagens de satélite, passando pela comparação entre EfficientNet e DenseNet121 na tarefa de classificação, até a aplicação da U-Net na restauração de imagens degradadas.

4.1 Tarefa de Detecção

A etapa de detecção teve como objetivo localizar aeronaves em imagens aéreas. O modelo **YOLO11**, configurado e treinado com o conjunto de dados anotado manualmente, apresentou desempenho consistente na identificação das aeronaves, demonstrando boa capacidade de generalização entre diferentes condições de iluminação, ângulos e escalas.

A avaliação quantitativa foi realizada com base nas métricas **mAP@0.5** e **mAP@0.5:0.95**, amplamente utilizadas em tarefas de detecção de objetos. As Figuras 7 e 8 apresentam a evolução das métricas de desempenho durante o treinamento, incluindo tanto o mAP quanto a precisão e revocação, destacando a melhoria do modelo ao longo das épocas. A Figura 9 apresenta exemplos qualitativos das predições geradas pelo modelo, evidenciando a correta identificação e o enquadramento das aeronaves dentro das regiões delimitadas.

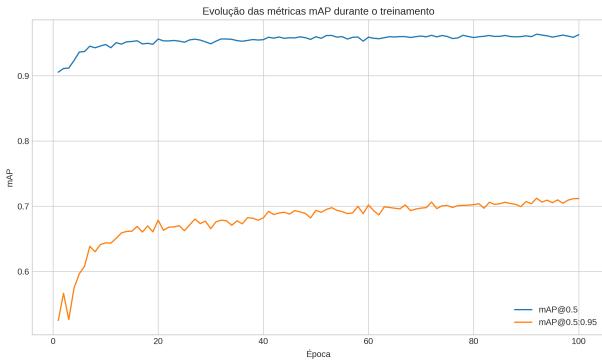


Figure 7: Evolução das métricas de desempenho durante o treinamento do modelo de detecção. Observa-se o aumento progressivo do **mAP@0.5** e do **mAP@0.5:0.95**, indicando melhor capacidade do modelo em localizar e classificar corretamente as aeronaves.

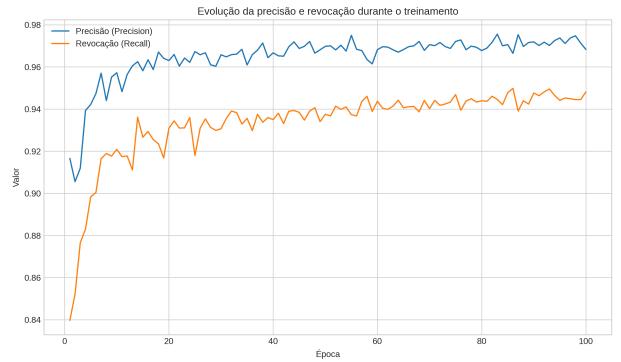


Figure 8: Evolução da **precisão** e **revocação** ao longo das épocas de treinamento. A estabilidade e o crescimento conjunto dessas métricas demonstram boa generalização e equilíbrio entre falsos positivos e falsos negativos.

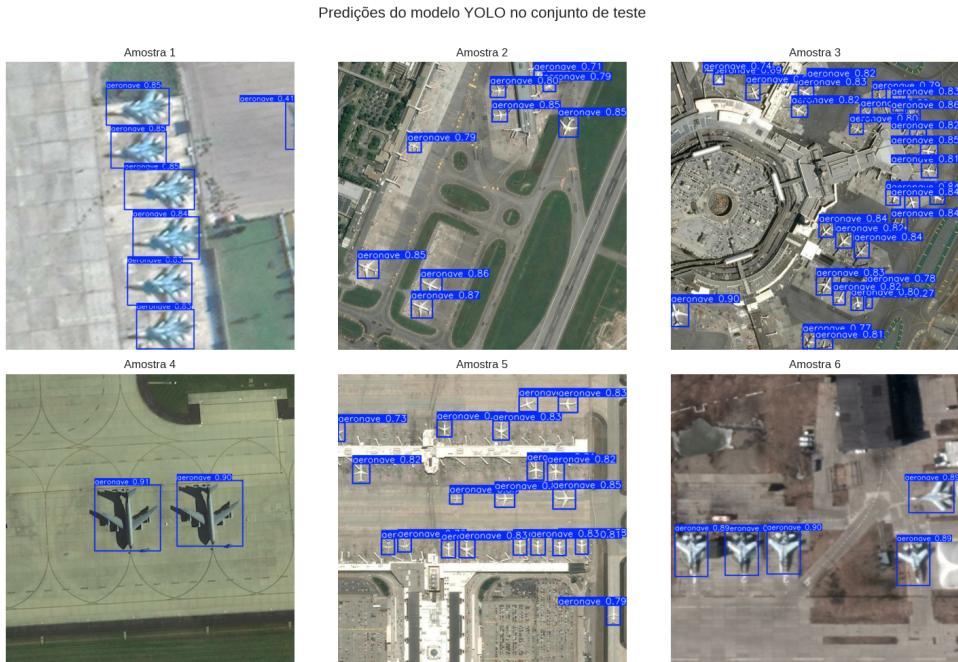


Figure 9: Resultados qualitativos da tarefa de detecção utilizando o modelo YOLO11. As caixas delimitadoras indicam as regiões detectadas correspondentes às aeronaves.

4.2 Tarefa de Classificação

Nesta etapa, foram comparados o desempenho das arquiteturas **EfficientNetV2S** e **DenseNet121**, ambas implementadas com o framework *TensorFlow* e inicializadas com pesos pré-treinados no conjunto *ImageNet*. O objetivo principal foi avaliar a capacidade de generalização de cada modelo na tarefa de reconhecimento de aeronaves militares, considerando o mesmo conjunto de treinamento, validação e teste.

A Figura 10 apresenta a evolução da **função de perda** (*loss*) durante o treinamento e validação das arquiteturas EfficientNetV2S e DenseNet121. Observa-se que ambas as redes convergiram de forma

estável, sem diferenças significativas entre elas, indicando desempenho comparável em termos de generalização. O treinamento foi interrompido automaticamente pelo *early stopping*, ocorrendo na época 34 para a EfficientNetV2S e na época 35 para a DenseNet121, demonstrando que ambos os modelos alcançaram um ponto de convergência satisfatório sem sinais de sobreajuste.

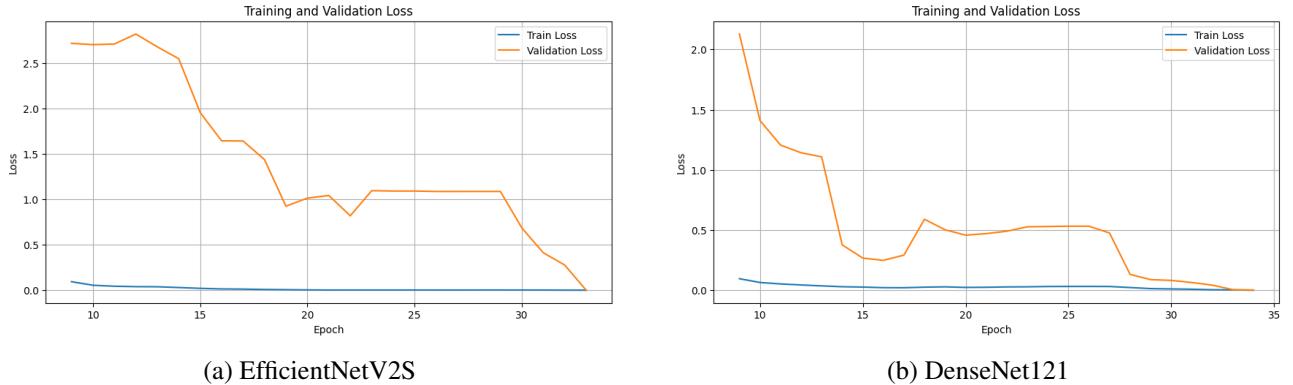


Figure 10: Evolução da função de perda (treinamento e validação) para as arquiteturas EfficientNetV2S e DenseNet121.

A Figura 11 mostra a evolução da **acurácia** nos conjuntos de treino e validação. Novamente, ambos os modelos apresentaram desempenho muito próximo, com acurácia de validação estabilizando em níveis similares. A análise das curvas indica que nenhuma das arquiteturas se sobressaiu de maneira consistente sobre a outra, confirmando que, para este conjunto de dados de classificação de aeronaves, tanto a EfficientNetV2S quanto a DenseNet121 fornecem resultados equivalentes em termos de acurácia e estabilidade de treinamento.

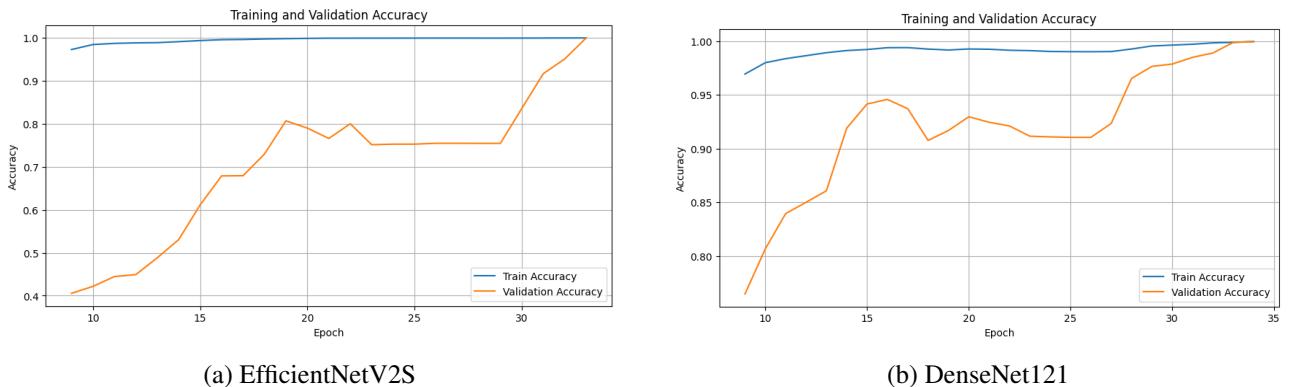


Figure 11: Evolução da acurácia (treinamento e validação) para as arquiteturas EfficientNetV2S e DenseNet121.

4.3 Tarefa de Denoising

Para a tarefa de remoção de ruído (*denoising*), foi empregada a arquitetura **U-Net**, amplamente utilizada em aplicações de restauração de imagem devido à sua capacidade de preservar detalhes espaciais enquanto reconstrói regiões degradadas. O modelo foi treinado com imagens corrompidas por ruído aditivo gaussiano e avaliou-se seu desempenho nos conjuntos de treinamento e validação.

A Figura 12 apresenta a evolução das principais métricas durante o treinamento: **função de perda (loss)**, **PSNR (Peak Signal-to-Noise Ratio)** e **SSIM (Structural Similarity Index)**. As três primeiras

curvas (linha superior) correspondem ao conjunto de treinamento, enquanto as três inferiores representam o conjunto de validação. O treinamento foi interrompido automaticamente pelo *early stopping*, ocorrendo na época 50, quando o modelo atingiu o ponto de convergência sem melhora significativa na perda de validação.

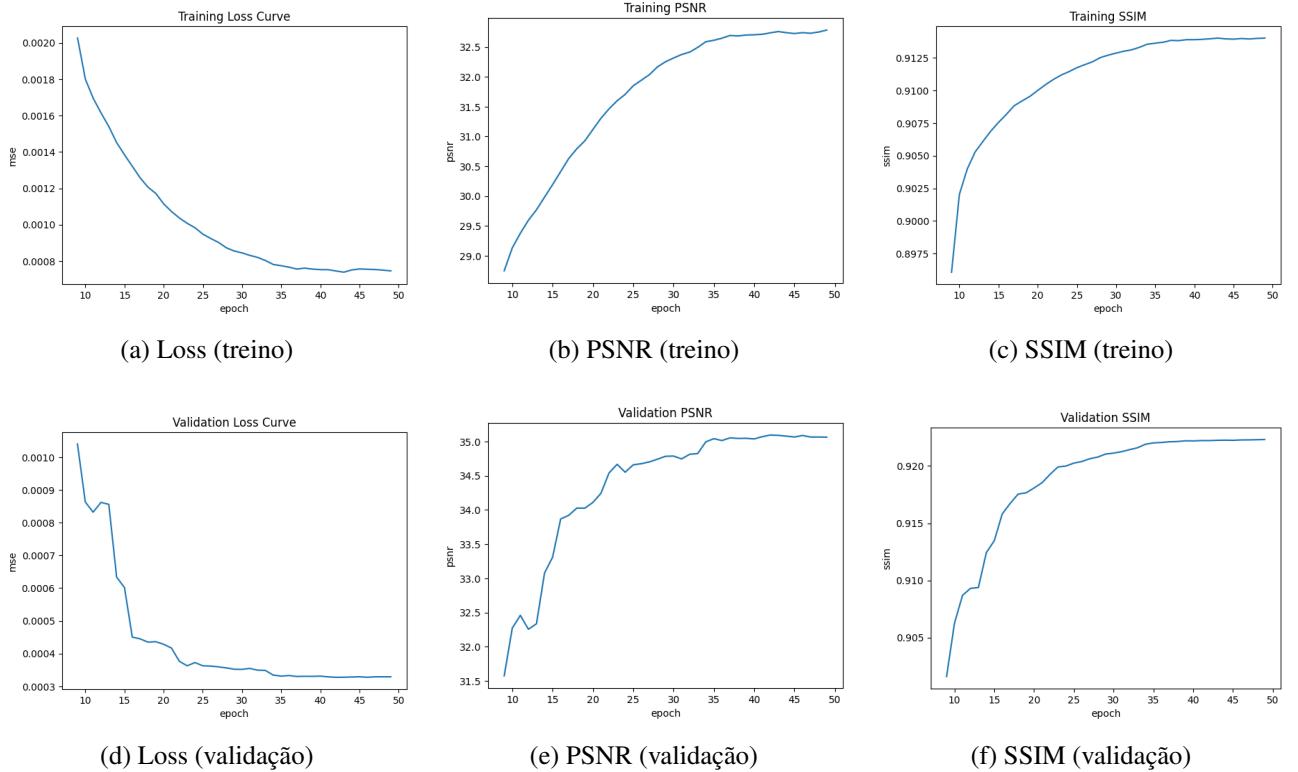


Figure 12: Evolução das métricas de desempenho da U-Net durante o treinamento e validação na tarefa de denoising.

A Figura 13 ilustra resultados qualitativos obtidos pelo modelo, exibindo amostras do conjunto de teste. Cada exemplo apresenta, da esquerda para a direita, a imagem de referência limpa, a imagem original ruidosa e a imagem restaurada pela U-Net. Observa-se que a U-Net foi capaz de reduzir significativamente o ruído preservando bordas e detalhes estruturais, conforme refletido pelos altos valores de PSNR e SSIM obtidos.

5 CONCLUSÃO

Este trabalho apresentou um pipeline completo para o processamento e análise de imagens de aeronaves militares, composto por três etapas principais: **detecção, classificação e remoção de ruído**. Na tarefa de detecção, o modelo **YOLO11** demonstrou alta precisão na identificação das aeronaves, apresentando bom equilíbrio entre desempenho e custo computacional. Para a classificação, as arquiteturas **EfficientNetV2S** e **DenseNet121**, ambas implementadas com o framework *TensorFlow* e pré-treinadas no conjunto *ImageNet*, mostraram resultados consistentes, validando a eficácia da estratégia de *transfer learning*. Já na etapa de **denoising**, o modelo **U-Net** foi capaz de reconstruir imagens degradadas preservando detalhes estruturais relevantes, evidenciando sua adequação para tarefas de restauração visual.

De forma geral, os resultados obtidos confirmam o potencial das redes neurais convolucionais em

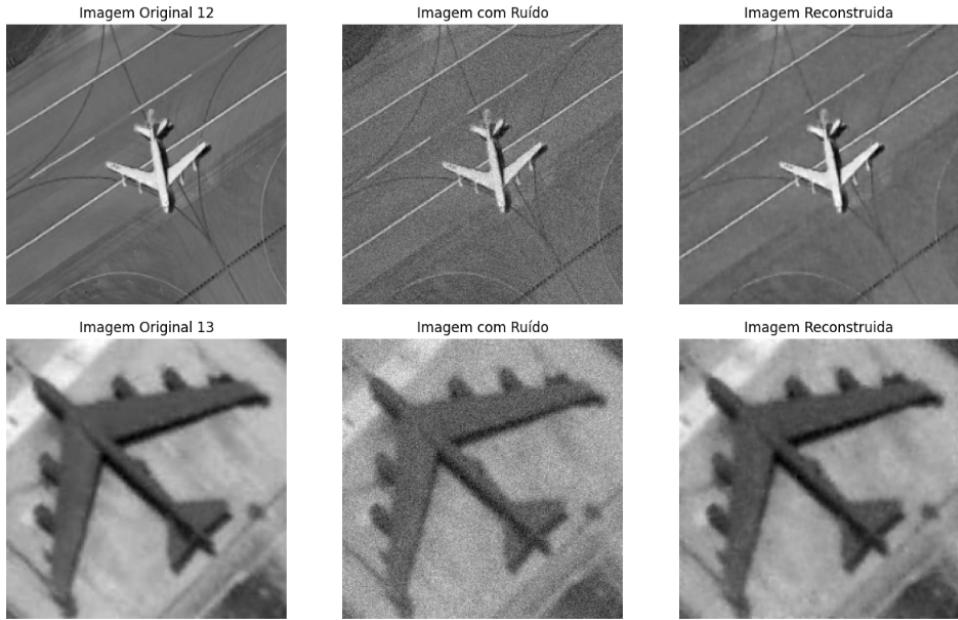


Figure 13: Exemplos qualitativos de denoising utilizando a U-Net: imagem de referência, imagem ruidosa e imagem restaurada.

aplicações de visão computacional voltadas à identificação e aprimoramento de imagens de aeronaves. As métricas e análises qualitativas indicam que a integração das três etapas no pipeline proposto contribui para um fluxo de processamento mais robusto e eficiente. Como perspectivas futuras, pretende-se ampliar o conjunto de dados, explorar técnicas avançadas de *data augmentation* e investigar o uso de arquiteturas mais recentes baseadas em atenção e visão transformadora (*Vision Transformers*), visando melhorar ainda mais a capacidade de generalização dos modelos.

REFERENCES

- [1] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.
- [2] M. Tan and Q. Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” in *International conference on machine learning*, PMLR, 2019, pp. 6105–6114.
- [3] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.
- [4] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical image computing and computer-assisted intervention*, Springer, 2015, pp. 234–241.
- [5] Kaggle, *Planes dataset*, <https://www.kaggle.com/datasets/tokarooo/aircraft-detection-with-yolov8>, Accessed: August 2025, 2024.
- [6] Kaggle, *Planes satellite imagery*, <https://www.kaggle.com/datasets/zidane10aa/airplanes-satellite-imagery?select=train>, Accessed: August 2025, 2024.