Detecção de macroplásticos através da aplicação de ensemble de modelos de aprendizado profundo

Messyas Gois França
Campus Manaus Centro
Instituto Federal de Educação, Ciência e Tecnologia do Amazonas
IFAM - Manaus - AM, Brasil
messyasgoisfranca@gmail.com

Marcelo Chamy Machado
Campus Manaus Centro
Instituto Federal de Educação, Ciência e Tecnologia do Amazonas
IFAM - Manaus - AM, Brasil
marcelo.chamy@ifam.edu.br

***Resumo*—Visando aprimorar o monitoramento automatizado de resíduos plásticos em ambientes naturais, cujo descarte inadequado representa um desafio ambiental, especialmente pela dificuldade de detecção devido à aparência irregular dos objetos. Neste trabalho, propõe-se a aplicação de Redes Neurais Convolucionais (CNNs) para detecção de resíduos plásticos. Além disso, foi investigada a utilização da técnica de *ensemble Weighted Boxes Fusion* (WBF) para fundir caixas delimitadoras de modelos distintos. Utilizando uma base de dados pública anotada com classes de resíduos, com uso de transferência de aprendizado e aumento de dados, o modelo YOLOv11x alcançou 8.99% a mais na métrica mAP que o trabalho original. A abordagem de combinação de caixas se provou válida, melhorando em 0.43% na métrica mAP em relação ao trabalho original, tendo se destacado em um dos cenários para a identificação de múltiplos objetos.**

***Index Terms*—Deep Learning, Transfer-learning , Plastic detection, CNN, Ensemble**

I. INTRODUÇÃO

O descarte inadequado de resíduos derivados de materiais plásticos causa sérios problemas ao meio ambiente. No Brasil, esta questão se torna alarmante devido ao alto volume de material despejado em ecossistemas vulneráveis, como rios, igarapés e áreas costeiras. Estudos estimam que o rio Amazonas despeja anualmente cerca de 38 toneladas de resíduos plásticos [1], enquanto as maiores bacias hidrográficas contribuem com aproximadamente 3,4 milhões de toneladas desses materiais para o mar por ano [2].

Atualmente, não há mecanismos automatizados eficazes para o monitoramento e a identificação da presença desse tipo de resíduo, o que limita a eficiência na alocação de recursos destinados à mitigação do problema. Essa deficiência também dificulta estudos contínuos e precisos, fundamentais para o planejamento de ações preventivas e corretivas.

Uma abordagem promissora para suprir essa lacuna é a utilização de técnicas de visão computacional, especialmente com ênfase em aprendizado profundo, por meio de modelos derivados de Redes Neurais Convolucionais (CNNs).

Essa abordagem pode ser aplicada para detectar resíduos plásticos em ambientes que são propensos ao acúmulo desses materiais. Contudo, ainda há desafios significativos na identificação de objetos pequenos ou que foram deteriorados devido a impactos e outros fatores ambientais. Nesse contexto, o uso de modelos pré-treinados pode melhorar substancialmente a precisão detecção, facilitando sua adaptação a essa tarefa complexa.

Diante disso, este trabalho visa contribuir com o monitoramento automatizado de resíduos plásticos, por meio da avaliação comparativa de diferentes modelos de aprendizado profundo, a partir da aplicação de transferência de aprendizado. Com base em um conjunto de imagens publicamente disponível, foram implementados modelos baseados em CNNs utilizando as bibliotecas Detectron2 e Ultralytics, permitindo assim a comparação entre diferentes arquiteturas e abordagens.

O artigo está estruturado da seguinte forma: a Seção II apresenta os trabalhos relacionados; a Seção III descreve a metodologia adotada; a Seção IV detalha os resultados obtidos; por fim, a Seção V apresenta as conclusões e perspectivas para pesquisas futuras.

II. TRABALHOS RELACIONADOS

Lieshout et al. [3] propuseram um método automatizado para contagem automática de macroplásticos flutuantes em rios, utilizando o modelo Faster R-CNN combinado com uma Rede Inceptionv2 pré-treinada no conjunto de dados COCO. O conjunto de dados utilizado possui 1.272 imagens capturadas por uma câmera de vídeo em cinco pontes de Jacarta, Indonésia, com cerca de 14.968 anotações compostas por objetos da categoria plástico, que foram anotadas por voluntários. O modelo desenvolvido obteve 64,7% de precisão aplicando aumento de dados e ajuste dinâmico da taxa de aprendizado com otimizador ADAM. O estudo demonstrou que a adição de exemplos do local de destino melhora a generalização do modelo e que a contagem automatizada pode superar a contagem humana.

Tharani et al. [4] desenvolveram um método para detectar lixo flutuante em canais urbanos utilizando modelos com camadas de atenção. O conjunto de dados contém 13.500 imagens anotadas manualmente, coletadas em diferentes condições climáticas e horários, totalizando 48.898 objetos identificados. Foram avaliados os modelos SSD, YOLOv3, YOLOv3-Tiny e PeleeNet, e além disso, foi proposta uma camada de atenção baseada em escala logarítmica para melhorar a detecção de objetos pequenos. O modelo YOLOv3 com a aplicação da técnica proposta obteve 31.2% no conjunto de teste mais simples, que contém uma partição de imagens

com objetos maiores, superando os demais. A utilização de atenção em escala logarítmica resultou em ganhos na detecção de objetos pequenos de forma geral.

Yang et al. [5] desenvolveram o modelo YOLOv5CBS para detecção de lixo flutuante em rios, aplicando melhorias em relação ao YOLOv5 original. Foram utilizados dois conjuntos de dados: um privado, com cerca de 2.400 imagens e 3.510 objetos anotados e o Flow-img, contendo 2.000 imagens e 5.271 objetos anotados. As adições incluíram a inclusão de um módulo de atenção coordenada (CA), substituição da *Path Aggregation Network* (PAN) por uma *Bidirectional Feature Pyramid Network* (BiFPN) e da função CIoU pela função SIoU. Os resultados foram superiores aos do modelo YOLOv5, Faster R-CNN, YOLOv3 e YOLOv4, alcançando em seu melhor resultado uma revocação de 0.86%, F1-score de 90.85% e 92.18% de AP no conjunto Flow-img.

Panwar et al. [6] utilizaram transferência de aprendizado com o modelo RetinaNet com uma rede ResNet-50. Além disso, foi introduzido o conjunto de dados AquaTrash, com 369 imagens anotadas manualmente. O modelo foi treinado por 20 épocas com *batch size* 8, alcançando mAP de 81,48%. A abordagem superou modelos de dois estágios, como Faster R-CNN, especialmente na detecção de pequenos objetos, atingindo confiança de até 88,1% para itens como sacolas plásticas e copos descartáveis.

Maharjan et al. [7] investigaram a detecção de resíduos plásticos em rios utilizando imagens aéreas capturadas por Drones. Foram coletadas imagens 4K em dois rios, divididas em quadrantes de 256×256 *pixels*, com 500 amostras por local. Foram avaliados modelos como YOLOv2, YOLOv3, YOLOv4 e YOLOv5 (assim como variações ”tiny”, ”spp”, ”m” e ”l”) com e sem transferência de aprendizado. O YOLOv4 utilizando transferência de aprendizado obteve os melhores resultados, com mAP de 0.83%.

Zhang et al. [8] propuseram o modelo YOLOv5-FF para detecção de objetos em ambientes de água doce, utilizando o conjunto FODS-00-01 com 6.240 imagens e 21.891 objetos anotados. As melhorias incluíram um mecanismo de atenção híbrida, convolução adaptativa, aprimoramento de características para multiescala e uma nova camada para objetos pequenos. O modelo também utilizou o algoritmo K-Medoids para ajuste das caixas delimitadoras. Os testes mostraram que o YOLOv5-FF superou modelos como YOLOv8n/s, YOLOv7, YOLOX, SSD e Faster R-CNN, atingindo mAP de 80,80%, recall de 79,60% e F1-score de 82,35%.

Tamin et al. [9] compararam o uso de imagens RGB e RGBNIR para detecção de resíduos plásticos em áreas costeiras, utilizando o modelo YOLOv5m. Foram coletadas 810 imagens, com cerca de 1.344 anotações manuais. Três configurações principais foram testadas: imagens isoladas, imagens combinadas com fundos e uma combinação de conjuntos de dados RGB + RGBNIR fundidos. A fusão dos conjuntos gerou os melhores resultados, com mAP@0.5 de 92,96% e mAP@0.5:0.95 de 69,47%. Constatou-se que adição de imagens de fundo reduziu falsos positivos e aumentou a robustez em diferentes cenários.

He et al. [10] propuseram o modelo EC-YOLOX para detecção de objetos flutuantes em ambientes aquáticos complexos, com dados anotados de garrafas, sacolas e galhos. O modelo incorpora os módulos *Coordinate Attention* (CA), *Efficient Channel Attention* (ECA) e perda varifocal. Os resultados mostraram superioridade frente a modelos como YOLOv3, YOLOv5 e YOLOX, com cerca 81,80% de mAP a mais, incluindo 95% de precisão para a classe bolas e 93% para garrafas.

Mandhati et al. [11] apresentaram um método para detecção e mapeamento de resíduos plásticos para ruas usando imagens georreferenciadas capturadas por câmeras acopladas a um veículo. O conjunto de dados resultante contém mais de 13.000 imagens e 79,101 anotações. Foram avaliados modelos como Faster R-CNN, RetinaNet, YOLOv3 e YOLOv5. O YOLOv5 obteve o melhor desempenho tendo mAP de 44,0% na variação do conjunto de dados com 4 classes. Os modelos treinados com pLitterStreet apresentaram maior generalização frente a base de dados como PlastOPol e TACO.

As Tabelas I e II resumem os principais resultados obtidos nos trabalhos relacionados. Como nem todos utilizaram as mesmas métricas, os resultados foram divididos em duas tabelas, cada uma com as métricas similares. A Tabela I mostra os resultados ordenados pela métrica AP e a Tabela II exibe os trabalhos que utilizaram a métrica mAP.

TABELA I
TRABALHOS RELACIONADOS (ORDENADOS POR AP).

Autores	Conjunto de Dados	Modelo	Resultado
Yang	FloW-Img	YOLOv5CBS	92.18 %
Lieshout	Jacarta River	Faster R-CNN + InceptionV2	68.7 %
Tharani	Floating Waste	YOLOv3 + Attn	31.2 %

TABELA II
TRABALHOS RELACIONADOS (ORDENADOS POR MAP).

Autores	Conjunto de Dados	Modelo	Resultado
Maharjan	HMH e TT	YOLOv5s	83.0 %
He	River scenes	EC-YOLOX	81.80 %
Panwar	AquaTrash	RetinaNet + ResNet50-fpn	81.4 %
Zhang	FODS-00-01	YOLOv5-FF	80.80 %
Tamin	RGB e RGBNIR	YOLOv5m	69.47 %
Mandhati	pLitterStreet	YOLOv5	44.0 %

É fundamental ressaltar que as tabelas não consideram a complexidade dos conjuntos de dados utilizados pelos autores. As características intrínsecas dos objetos de uma base de dados podem impactar significativamente no desempenho dos modelos testados, sendo esse um elemento que não foi abordado nas comparações anteriores.

III. METODOLOGIA

A base de dados publica *pLitterStreet* foi selecionada devido às suas características representativas da realidade. Ela possui cerca de 13.064 imagens capturadas por câmeras embarcadas em veículos e dispositivos móveis, totalizando 79.101 anotações de objetos [11]. Originalmente, os objetos

são distribuídos em dez rótulos de classes, além de possuir uma variação com 4 classes. Para este estudo, optou-se pela variação com 4 classes (“plástico”, “pilha de lixo”, “máscara facial” e “lixeira”). Essa escolha fundamenta-se na observação dos autores da base de dados original de que tal configuração resulta em melhor desempenho quando comparada à variação com dez classes, principalmente devido à dificuldade intrínseca na distinção de objetos deteriorados entre as categorias mais granulares.

O conjunto de dados pode ser dividido em 3 tamanhos de objetos, os critérios de tamanho de área em *pixels* e a quantidade podem ser observados na Tabela III.

TABELA III
DISTRIBUIÇÃO DOS OBJETOS POR ÁREA OCUPADA.

Categoria	**Área (pixels²)**	**Quantidade**
Pequeno	32^2	46.001
Médio	$32^2 \leq$ Área $< 96^2$	28.000
Grande	$\geq 96^2$	5.100

Conforme ilustrado na Figura 1, a base de dados apresenta imagens de alta resolução (4k) capturadas em uma diversidade de ambientes urbanos, incluindo margens de rios e vias públicas. Um aspecto relevante do conjunto de dados é a predominância de objetos pequenos. Embora essa característica represente fidedignamente à realidade dos resíduos em cenários urbanos, ela impõe desafios significativos ao treinamento de modelos baseados em CNNs, dado o volume reduzido de informações visuais que esses objetos oferecem. Contudo, a alta qualidade das imagens, aliada à variedade de contextos geográficos e condições de iluminação, confere ao conjunto uma considerável capacidade de generalização para modelos nele treinados.

Além da base de treino, este trabalho desenvolveu uma base de dados não anotada, a partir de imagens extraídas de vídeos do YouTube. Estes vídeos são predominantemente filmagens de reportagens sobre a poluição de igarapés em Manaus/AM. Como ela não é anotada, optou-se por utilizar este conjunto apenas para testes, a fim de investigar o desempenho dos modelos em situações reais e distintas da base em que foram treinados.

A. Ambiente de desenvolvimento

A fase de implementação e avaliação experimental dos modelos foi conduzida utilizando a linguagem de programação Python, utilizando as bibliotecas Ultralytics e Detectron2 [18] [15], ambas amplamente empregadas em aplicações de visão computacional. O desenvolvimento e a execução dos algoritmos ocorreram em um ambiente Conda, utilizando Jupyter Lab como interface de desenvolvimento. Os recursos computacionais mobilizados para os experimentos incluíram uma unidade de processamento gráfico (GPU) NVIDIA RTX 3090 com cerca de 24GB de VRAM, e uma NVIDIA RTX 3070 com 8GB de VRAM em uma estação de trabalho local utilizada para modelos e testes leves.

Fig. 1. Exemplo de imagens do conjunto de dados pLitterStreet.

B. Pré-processamento e Aumento de dados

No pré-processamento, as imagens foram inicialmente lidas no formato BGR (Blue, Green, Red). Em seguida, o menor lado das imagens foi redimensionado para valores entre 600 *pixels* e 1024 *pixels*, com o lado maior mantendo a proporção do lado menor. Se, após esse ajuste, o lado maior exceder 1000 *pixels*, seria aplicado à imagem um novo redimensionamento, para que o lado maior ficasse com no máximo 1000 *pixels* (o lado menor pode ficar abaixo de 600 nesse caso). Essa abordagem foi necessária, pois a base de dados possui imagens retiradas em diferentes orientações.

As transformações utilizadas para aumento de dados, que podem afetar a posição das caixas delimitadoras, como rotação de eixo foram igualmente aplicadas às anotações, o que permitiu manter a correspondência para as coordenadas das caixas delimitadoras. Por fim, as imagens em fomato de *Array* foram convertidas para um tensor *PyTorch*, onde a ordem dos eixos foi alterada para CHW (Canais, Altura, Largura).

A aplicação de técnicas de aumento de dados em CNNs é uma prática amplamente difundida na literatura, devido ao seu significativo potencial para aprimorar o desempenho de modelos, mitigar o sobreajuste e possibilitar o treinamento de modelos robustos, mesmo diante de conjuntos de dados limitados ou desbalanceados, e essa estratégia foi utilizada em todos os modelos presentes neste trabalho.

O sobreajuste manifesta-se quando um modelo assimila a função de treinamento com variância excessivamente alta, resultando em uma modelagem precisa dos dados de treino, porém com falhas na generalização para dados não observados previamente. O aumento de dados constitui uma estratégia

eficaz, atuando no espaço dos dados para contornar essa problemática. Ao expandir artificialmente a diversidade do conjunto de treinamento, essa abordagem busca minimizar a discrepância entre os domínios de treinamento e validação, promovendo, assim, uma maior capacidade de generalização do modelo.

Para implementar o aumento de dados, foram utilizados filtros que aplicam transformações de imagem, como inversões nos eixos e rotações em ângulos variados. Adicionalmente, aplicaram-se ajustes como o aumento de brilho, saturação e contraste das imagens. Essas modificações foram introduzidas de forma aleatória durante a leitura dos dados.

C. Treinamento dos modelos

Treinar modelos derivados de CNNs envolve diversos desafios, pois a má escolha dos parâmetros pode afetar a precisão e a generalização resultantes. Inicialmente foram utilizados valores de hiperparâmetros indicados pelas bibliotecas para treinar os modelos, e a partir da observação dos *logs* de treinamentos, diversas adequações foram sendo aplicadas. Além disso, o limite de épocas em que os modelos poderiam ser treinados foi limitado a 100, o que não ocorreu para nenhum modelo, já que a maioria convergiu em poucas épocas. O valor de *momentum* foi de 0.9 para todos os modelos. O parâmetro Precisão Mista Automática (AMP) foi ativado, o que permite diminuir o tempo de treino simplificando cálculos menos importantes. Por último, os modelos foram compilados com o otimizador SGD.

Para melhorar os resultados foi utilizado o ajuste fino (*fine-tuning*), que é um tipo de técnica de *transfer learning*, que permite transferir a aprendizagem de modelos previamente treinados, o que aumenta a precisão final de forma significativa em relação ao treinamento feito completamente do zero.

A paciência para o parâmetro *early stopping*, que permite parar o treinamento quando a evolução estagna, foi definida entre 5 e 20 épocas, dependendo do modelo. Esse valor final foi ajustado a partir da observação da curva de evolução de métricas como *mean average precision* (mAP) e *Loss*.

O tamanho de lote (*batch size*) foi ajustado para valores entre 8 e 16 de acordo com a quantidade de memória disponível. Em modelos mais pesados, como Cascade R-CNN por exemplo, o valor foi menor devido à sua maior necessidade de recursos. Observou-se que modelos como RetinaNet e YOLO utilizam menos recursos, e por isso o tamanho de lote foi maior.

O valor para decaimento de pesos foi de 0.0001 para modelos como Faster R-CNN, Cascade R-CNN e RetinaNet. A taxa de aprendizado teve o valor inicial comum de 0.00001 para os modelos implementados com a biblioteca Detectron2. Após variações com valores menores e maiores, a taxa de aprendizado final para o Cascade-RCNN e Faster-Rcnn baseado em DenseNet101 foi de 0.004.

Para os modelos YOLO, o treinamento seguiu as configurações disponíveis na documentação da biblioteca Ultralytics. Isso incluiu o uso de pesos pré-treinados no conjunto de dados COCO, a ativação de aumento de dados, e o tamanho de entrada padrão de 640. Outros hiperparâmetros, como a taxa de aprendizado inicial, foi de 0.01 utilizando o otimizador SGD, tendo sido gerenciada dinamicamente pelo *framework* durante o treinamento.

D. Combinação de modelos

Visando investigar a possibilidade de realizar a combinação de modelos distintos, abordagem essa que ainda não havia sido explorada por outros autores, empregou-se uma metodologia de combinação (*ensemble*) que usa as predições de múltiplos modelos. Essa técnica permite fundir as caixas delimitadoras (*bounding boxes*) geradas por modelos distintos, o que pode melhorar o resultado final da detecção e refina o posicionamento das caixas delimitadoras.

Para realizar essa fusão, utiliza-se o algoritmo *Weighted Boxes Fusion* (WBF). O WBF consolida caixas delimitadoras de diferentes modelos para um mesmo objeto. Ele as agrupa com base na Intersecção sobre União (IoU) e, crucialmente, calcula uma nova caixa fundida utilizando a média ponderada das coordenadas e pontuações de todas as caixas no agrupamento [13]. Diferentemente do *Non-Maximum Suppression* (NMS) e suas variantes, que tendem a descartar caixas, o WBF aproveita a informação de múltiplas predições, mesmo que imprecisas. A Figura 2 ilustra como o NMS/soft-NMS selecionaria apenas uma das caixas imprecisas, e como o WBF as utiliza para gerar uma caixa fundida potencialmente mais próxima da verdade fundamental, lidando melhor com a interpolação de caixas.

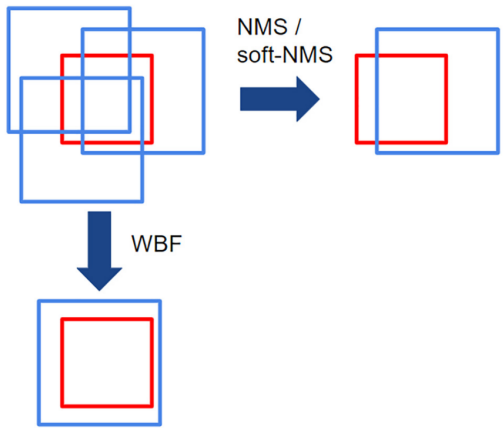

Fig. 2. Ilustração esquemática dos resultados do NMS/soft-NMS e do WBF para um conjunto de previsões imprecisas. Azul – previsões de diferentes modelos, vermelho – verdade fundamental.

A primeira combinação utilizada foi composta pelo modelo Faster R-CNN baseado na rede DenseNet101 combinado ao Cascade R-CNN baseado em ResNet50. A segunda combinando o Faster R-CNN (DenseNet101-FPN) com o YOLOv8x, para analisar a integração entre modelos de estágios distintos (*two-stage e one-stage*). Por ultimo, os modelos Cascade R-CNN e YOLOv8x também foram combinados.

Para executar a fusão de caixas delimitadoras, é necessário aplicar parâmetros para: Definir um limiar de IoU, que é

responsável por definir o grau de sobreposição para que duas caixas sejam consideradas pastes do mesmo *cluster*, e assim, fundidas. Definir o limiar que limita a confiança para pular caixas, o que permite garantir que caixas com um limiar muito baixo, por exemplo menor que 0.01 (1%) sejam consideradas. E por último, é necessário definir os pesos que serão atribuídos a cada modelo, sendo possível atribuir um peso maior para um modelo com métricas melhores e peso menor para um inferior.

Na implementação, foi definido um limiar de IoU de 0.5 para a fusão e um limiar de confiança de 0.3 para as caixas individuais. Para os experimentos de combinação entre YOLOv8x e Faster R-CNN, foi atribuído peso maior ao modelo YOLOv8x (2, 1), com o objetivo de investigar se essa abordagem compensaria a diferença de desempenho observadas nas métricas de avaliação. Já para a combinação entre Faster R-CNN e Cascade-RCNN, apesar da pequena discrepância em algumas métricas, foram utilizados pesos equilibrados (1, 1), dada a maior proximidade entre arquiteturas. A combinação de YOLOv8x e Cascade-RCNN também utilizou (1, 1). Por último, no caso do modelo YOLOv8x combinado ao YOLOv11x, assim como no artigo de referência de aplicação do ensemble com WBF, foram testadas variações de pesos, como (2, 1) e (3, 2), tendo sido a final (1, 1), assim como nos outros casos [13].

E. Métricas de Avaliação

Para avaliar o desempenho dos modelos de detecção de forma objetiva, são utilizadas algumas métricas de avaliação padronizadas. Como os modelos produzem predições com um grau de confiança associado, as métricas permitem traduzir essas saídas em um indicador de performance consolidado [17]. A avaliação se baseia em componentes fundamentais que classificam cada predição feita pelo modelo em comparação com as anotações reais, como pode ser visto a seguir:

- **Verdadeiro Positivo:** O modelo detecta corretamente um objeto que existe.
- **Falso Positivo:** O modelo detecta um objeto que não existe ou classifica incorretamente um objeto existente.
- **Falso Negativo:** O modelo não detecta um objeto que existe.

1) Revocação: A Revocação mede a capacidade do modelo de encontrar todos os objetos relevantes presentes nas imagens. É definida como a proporção de verdadeiros positivos em relação a todos os objetos que realmente existem.

$$\text{Revocação} = \frac{TP}{TP + FN}$$

2) F1-Score: É uma métrica que busca um equilíbrio entre a Precisão e a Revocação. Ele é a média harmônica de ambas as métricas, resultando em um único valor que penaliza modelos que são bons em uma métrica, mas ruins na outra. Um F1-Score alto indica que o modelo possui tanto alta precisão quanto alta revocação.

$$\text{F1-Score} = 2 \times \frac{\text{Precisão} \times \text{Revocação}}{\text{Precisão} + \text{Revocação}}$$

3) Média da precisão média (mAP): É a principal métrica de avaliação para tarefas de detecção de objetos. Seu cálculo é feito em duas etapas:

- **Precisão Média (*Average Precision*):** Primeiro, calcula-se o AP para cada classe de objeto individualmente. O AP resume a curva de Precisão-Revocação, representando a área sob essa curva. De forma prática, ele fornece um valor único que quantifica o desempenho do modelo para uma classe específica. A fórmula para o cálculo interpolado do AP é:

$$\text{AP} = \sum_{k=1}^{n} (R_k - R_{k-1}) P_k$$

  Onde P_k e R_k são a precisão e a revocação no k-ésimo limiar de confiança.
- **Cálculo do mAP:** É a média dos valores de AP de todas as N classes do conjunto de dados. Representando o desempenho geral do modelo em todas as categorias.

$$\text{mAP} = \frac{1}{N} \sum_{i=1}^{N} \text{AP}_i$$

Uma de suas variações é o mAP@50, que calcula o mAP com um limiar de Interseção sobre União (IoU) de 0.5. O que implica em uma detecção só ser considerada verdadeira se a área de sobreposição entre a caixa prevista e a caixa real for de pelo menos 50%. Sendo ela importante para avaliar a capacidade geral do modelo em localizar objetos, mesmo que a delimitação da caixa não seja perfeitamente precisa, sendo um padrão de avaliação amplamente adotado na literatura.

F. RESULTADOS E DISCUSSÕES

Nesta seção, serão descritos os resultados obtidos com a execução dos procedimentos metodológicos adotados. Para validar o desempenho de cada modelo de forma justa, foram executados testes de inferência utilizando a partição do conjunto de dados original que contém imagens em que o modelo não foi exposto durante o treinamento. Este tipo de avaliação permite verificar a capacidade de generalização dos modelos para novos contextos.

A Tabela IV apresenta um comparativo do desempenho entre os modelos avaliados no conjunto de testes da base de dados. As métricas incluem mAP, mAP50, Revocação e F1-Score. O modelo YOLOv11x foi o que mais se destacou em métricas relevantes como mAP, tendo atingido 72.21%, seguido pelo YOLOv8x com de 70.52% mAP, que teve resultados próximos. O modelo Cascade-RCNN ficou um pouco atrás, atingindo 64.06% mAP, os modelos Faster-RCNN (ResNet50 e DenseNet101) tiveram resultados próximos, com cerca de 39.97% e 36.65% respectivamente. Contudo, em modelos como o Cascade R-CNN, foi observada uma capacidade de identificação de objetos presentes nas imagens por vezes significativamente maior, o que talvez possa ser representado pelo valor de revocação mais alto, com cerca de 74.37%. O modelo RetinaNet teve um desempenho abaixo dos demais, como pode ser observado na tabela.

TABELA IV
COMPARATIVO DE MÉTRICAS DE DETECÇÃO.

Modelo	mAP	mAP@50	Revocação	F1-Score
YOLOv11x	52.99 %	72.21 %	61.84 %	70.76 %
YOLOv8x	51.46 %	70.52 %	62.91 %	69.70 %
Cascade R-CNN	41.21 %	64.06 %	74.37 %	55.99 %
Faster R-CNN (ResNet50)	39.97 %	63.85 %	53.94 %	–
Faster R-CNN (DenseNet101)	36.65 %	62.17 %	68.76 %	65.14 %
RetinaNet	22.86 %	34.11 %	37.15 %	46.62 %

Em alguns dos testes, ocorreu a ativação precoce do *early stopping*. Nesses casos, com o objetivo de melhorar os resultados para uma comparação justa entre os modelos, os hiperparâmetros como a taxa de aprendizado e paciência foram ajustados, o que resultou na melhoria do desempenho do modelo YOLOv11x que teve o parâmetro de paciência modificado para 20.

Na Figura 3 é possível observar a evolução da precisão para a classe plástico do modelo Cascade-RCNN, onde a métrica evoluiu e continuou melhorando seus resultados em termos de AP até estagnar próximo do fim do treinamento, esse gráfico é importante pois mostra que o modelo se ajustou à tarefa até o fim do treinamento.

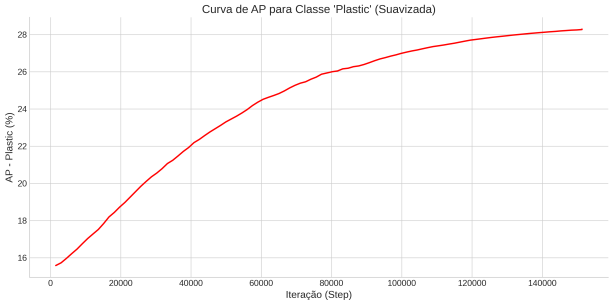

Fig. 3. Gráfico da evolução da métrica AP para a classe plástico.

A Figura 4 mostra a evolução do ajuste do modelo Cascade-RCNN aos dados durante o treinamento, sendo possível observar que o modelo teve maior convergência nas etapas iniciais.

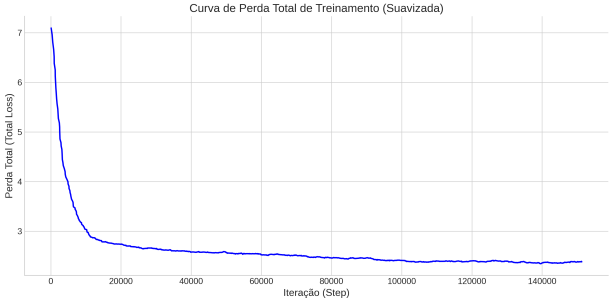

Fig. 4. Gráfico de *Loss*.

G. Tempo de treinamento

Outra forma relevante de comparar os modelos é o tempo de treinamento, que em geral está ligado à profundidade da rede utilizada. Modelos que realizam etapas adicionais, como validações duplas durante a inferência, também tendem a demandar mais tempo para serem treinados. Como observado na Tabela V, redes mais profundas, como DenseNet101 e ResNet50, apresentaram os maiores tempos de treinamento. Em contrapartida, modelos de estágio único foram, em geral, mais rápidos. Adicionalmente, observa-se que os modelos da família YOLO treinaram mais rápido e, ainda assim, obtiveram os melhores resultados nas métricas avaliadas.

TABELA V
TEMPO DE TREINAMENTO DOS MODELOS.

Modelo	Tempo de Treino (Horas)
Faster R-CNN (DenseNet101)	27.05
Faster R-CNN (ResNet50)	23.54
Cascade R-CNN	18.13
YOLOv11x	13.45
YOLOv8x	11.00
RetinaNet	2.24

H. Resultados da Combinações de modelos

Foram executados testes com modelos distintos utilizando *ensemble* de caixas delimitadoras, com resultados indicados na Tabela VI. A tabela exibe métricas como mAP, AP@0.50 e Revocação.

TABELA VI
COMPARATIVO DOS TESTES DE COMBINAÇÃO DE MODELOS.

Combinação Ensemble	mAP (%)	mAP@0.50 (%)	Revocação (%)
Cascade R-CNN + Faster-RCNN	44.34 %	70.97 %	58.28 %
YOLOv8x + Faster-RCNN	38.94 %	59.31 %	49.93 %
Cascade-RCNN + YOLOv8x	36.10 %	54.80 %	59.80 %
YOLOv11x + YOLOv8x	20.46 %	20.16 %	22.73 %

Observa-se que a combinação de modelos com arquiteturas e desempenhos semelhantes (Cascade R-CNN + Faster R-CNN) resultou em um ganho de performance em relação ao original, alcançando um mAP de 44.34%. Este valor é superior ao mAP obtido por ambos os modelos isoladamente (41.21% e 39.97%, respectivamente), indicando que o WBF foi eficaz em agregar valor ao refinar as predições de ambos.

Para o caso da combinação de modelos YOLO, notou-se um desempenho inferior em relação às outras abordagens. Foram realizadas diversas variações nos parâmetros de confiança de IOU, e mesmo com a tentativa de ajustar parâmetros como o de *SKIP* para limiares mais permissivos como 0.001, não foram alcançados ganhos relevantes.

A combinação de modelos com arquiteturas e desempenhos distintos (YOLOv8x + Faster R-CNN e YOLOv8x + Cascade) apresentou um resultado mediano em comparação com as outras, ficando abaixo de ambos os modelos testados de forma isolada. Mesmo com a aplicação de pesos diferenciados, a grande disparidade de performance dificultou uma fusão benéfica, sugerindo que o WBF performa melhor com modelos próximos em desempenho e tende a ser mais compatível com modelos como Faster-RCNN e Cascade-RCNN.

A Figura 5 ilustra o resultado de inferência executada em uma imagem do conjunto de testes da base pLitter para avaliar

a combinação do modelo Faster-RCNN e Cascade-RCNN. Apenas predições com acima do limiar de 70% de precisão estão presentes.

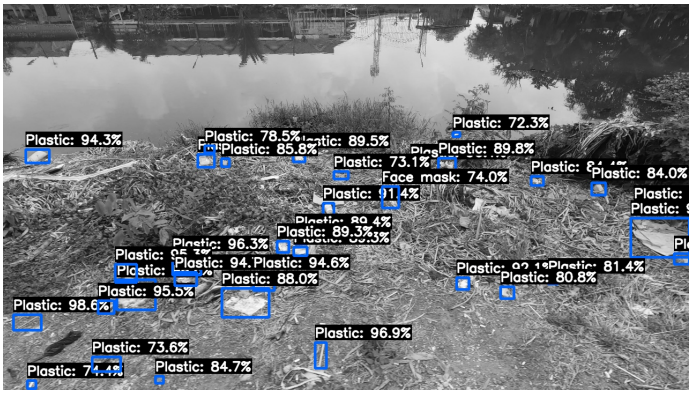

Fig. 5. Exemplo de inferência utilizando o ensemble Cascade-RCNN + Faster R-CNN, ilustrando um cenário onde fusão pode resultar em predições boas, muito provavelmente devido à proximidade de desempenho dos modelos. Contudo, pode-se observar casos de falsos positivos.

Os resultados dos modelos YOLOv11x e YOLOv8x demonstram uma superioridade notável, alcançando 52.99% e 51.46% na métrica mAP respectivamente, o que representa um aumento de 8.99% e 7.46% de mAP sobre o melhor resultado do trabalho original que obteve 44.0% na métrica mAP [11]. Além disso, a combinação de modelos Cascade R-CNN com Faster R-CNN atingiu 44.34% de mAP, superando o trabalho de referencia em 0.43% de mAP. Embora mais modesto, este resultado é significativo, pois indica que a abordagem de *ensemble* de caixas delimitadoras, quando aplicada a modelos compatíveis, pode aumentar os resultados finais das predições.

A falha da combinação dos modelos YOLO e Faster-RCNN, assim como nos outros casos negativos ressalta que a seleção dos modelos a serem combinados deve ser investigada de forma cautelosa. O *ensemble* dos modelos Cascade e Faster-RCNN, representa uma contribuição positiva, superando os resultados estabelecidos no trabalho de base em termos de mAP.

I. Testes com cenários reais

Para demostrar a proposta em situações reais, foram realizados testes de validações em imagens retiradas da base de igarapés menciona no início da Seção III. A proposta dessa comparação foi avaliar o desempenho de um dos modelos e da melhor combinação de modelos em cenários reais no aspecto da capacidade de generalização.

No exemplo apresentado na Figura 6, o modelo YOLOv8x teve um desempenho ruim para identificar os objetos presentes na imagem, mostrando uma baixa capacidade de generalizar para outros cenários diferentes da base de dados em que foi treinado.

A combinação dos modelos Cascade-RCNN e Faster R-CNN obteve um resultado superior em generalização, o que pode ser observado na Figura 7, na qual a combinação

Fig. 6. Teste de inferência utilizando o modelo YOLOv8x aplicado a uma imagem registrada em um igarapé transbordado em manaus.

Fig. 7. Teste de inferência utilizando *ensemble* com Cascade-RCNN + Faster R-CNN aplicado a uma imagem registrada em um igarapé transbordado em manaus.

identificou mais objetos e obteve um nível maior de precisão em suas predições.

IV. CONCLUSÕES

Neste trabalho foram investigadas abordagens para a detecção de resíduos plásticos em ambientes urbanos, utilizando e comparando diversas arquiteturas de CNN com a aplicação de ajuste-fino com transferência de aprendizado. Os experimentos foram realizados utilizando uma base de dados disponível publicamente, que é composta por imagens de ambiente aberto, característica que é vital para o monitoramento de plásticos.

Os experimentos demonstraram que o modelo YOLOv11x apresentou o melhor desempenho geral, alcançando um mAP 8.99% maior que o trabalho de referência, que introduziu a base de dados utilizada. Contudo, observou-se que modelos da família perderam mais objetos que deviam ser identificados do que modelos como Cascade-RCNN.

A abordagem de *ensemble* via WBF foi 0.43% superior na métrica mAP em relação ao trabalho original, demonstrando-se válida como uma estratégia para consolidar predições e potencialmente aumentar a robustez geral. No entanto, percebeu-se que o sucesso dessa abordagem depende fortemente do equilíbrio de desempenho de ambos os modelos combinados.

De forma geral, os resultados foram promissores e indicam o potencial da visão computacional para detecção de macroplásticos. As limitações dos modelos observados envolvem a capacidade de generalização, principalmente para modelos da família YOLO que em média foram mais precisos, mas tenderam a perder mais objetos que deviam ser identificados. A combinação de modelos Cascade-RCNN e Faster-RCNN teve um desempenho pratico promissor, tendo sido superior a modelos como YOLOv8 em alguns cenários.

Como sugestão para trabalhos futuros, encoraja-se o uso de conjuntos de dados mais amplos, com anotações mais consistentes e balanceadas. Além disso, explorar abordagens mais recentes como a incorporação de mecanismos de atenção a um modelo base, ou mesmo modelos baseados em arquiteturas *Transformers*, podem trazer ganhos para a generalização e a correta classificação de objetos plásticos em cenários mais complexos.

REFERÊNCIAS

[1] L. C. M. Lebreton, J. Van Der Zwet, J.-W. Damsteeg, B. Slat, A. Andrady, and J. Reisser, "River plastic emissions to the world's oceans," *Nature Commun.*, vol. 8, no. 1, pp. 15611, 2017.

[2] BLUEKEEPS, "Diagnóstico das fontes de escape de resíduos plásticos para o oceano," Pacto Global da ONU – Rede Brasil, 2022. [Online]. Disponível em: https://go.pactoglobal.org.br/SumarioExecutivoBlueKeepersDocumentoPDF.

[3] C. van Lieshout et al., "Automated river plastic monitoring using deep learning and cameras," *Earth and Space Science*, vol. 7, no. 8, p. e2019EA000960, 2020. [Online]. Disponível em: https://doi.org/10.1029/2019EA000960

[4] M. Tharani et al., "Attention neural network for trash detection on water channels," *arXiv preprint*, arXiv:2007.04639, 2020. [Online]. Disponível em: https://doi.org/10.48550/arXiv.2007.04639

[5] X. Yang et al., "Detection of river floating garbage based on improved YOLOv5," *Mathematics*, vol. 10, no. 22, p. 4366, 2022. [Online]. Disponível em: https://doi.org/10.3390/math10224366

[6] H. Panwar et al., "AquaVision: Automating the detection of waste in water bodies using deep transfer learning," *Case Stud. Chem. Environ. Eng.*, Elsevier, 2020. [Online]. Disponível em: https://doi.org/10.1016/j.cscee.2020.100026

[7] N. Maharjan et al., "Detection of river plastic using UAV sensor data and deep learning," *Remote Sensing*, vol. 14, no. 13, p. 3049, 2022. [Online]. Disponível em: https://doi.org/10.3390/rs14133049

[8] X. Zhang et al., "YOLOv5-FF: Detecting floating objects on the surface of fresh water environments," *Appl. Sci.*, vol. 13, no. 13, 2023. [Online]. Disponível em: https://doi.org/10.3390/app13137367

[9] O. Tamin et al., "On-shore plastic waste detection with YOLOv5 and RGB-near-infrared fusion," *Big Data Cogn. Comput.*, vol. 7, no. 2, p. 103, 2023. [Online]. Disponível em: https://doi.org/10.3390/bdcc7020103

[10] J. He et al., "EC-YOLOX: A deep learning algorithm for floating objects detection in ground images of complex water environments," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, 2024. [Online]. Disponível em: https://ieeexplore.ieee.org/document/10449338

[11] S. R. Mandhati et al., "pLitterStreet: Street level plastic litter detection and mapping," *arXiv preprint*, arXiv:2401.14719, 2024. [Online]. Disponível em: https://arxiv.org/abs/2401.14719

[12] T. Jia et al., "Advancing deep learning-based detection of floating litter using a novel open dataset," *Front. Water*, vol. 5, p. 1298465, 2023. [Online]. Disponível em: https://doi.org/10.3389/frwa.2023.1298465

[13] R. Solovyev et al., "Weighted boxes fusion: Ensembling boxes from different object detection models," *Image and Vision Computing*, vol. 107, p. 104117, 2021. [Online]. Disponível em: https://doi.org/10.1016/j.imavis.2020.104117

[14] C. Shorten and T. M. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning," *Journal of Big Data*, vol. 6, art. no. 60, pp. 1–48, 2019. [Online]. Available: https://doi.org/10.1186/s40537-019-0197-0

[15] WU, Y.; KIRILLOV, A.; MASSA, F.; LO, W.-Y.; GIRSHICK, R., "Detectron2," Facebook AI Research, 2019. [Online]. Disponível em: https://github.com/facebookresearch/detectron2.

[16] V. Pham, C. Pham, and T. Dang, "Road damage detection and classification with detectron2 and faster r-cnn," in *2020 IEEE International Conference on Big Data (Big Data)*, pp. 5592–5601, 2020. [Online]. Disponível em: https://doi.org/10.1109/BigData50022.2020.9378027

[17] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes (VOC) Challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010. [Online]. Disponível em: https://doi.org/10.1007/s11263-009-0275-4

[18] Ultralytics, "Ultralytics YOLO: Real-Time Object Detection," *GitHub Repository*, 2024. [Online]. Disponível em: https://github.com/ultralytics/ultralytics