A Comparative Study between Deep Learning Approaches for Aphid Classification

Brenda Slongo Taca, Douglas Lau, and Rafael Rieder

***Abstract*—This study presents a performance comparison between two convolutional neural networks in the task of detecting aphids in digital images: AphidCV, customized for counting, classifying, and measuring aphids, and YOLOv8, state-of-the-art in real-time object detection. Our work considered 48,000 images for training for six different insect species (8,000 images divided into four classes), in addition to data augmentation techniques. For comparative purposes, we considered evaluation metrics available to both architectures (Accuracy, Precision, Recall, and F1-Score) and additional metrics (ROC Curve and PR AUC for AphidCV; mAP@50 and mAP@50-95 for YOLOv8). The results revealed an average F1-Score = 0.891 for the AphidCV architecture, version 3.0, and an average F1-Score = 0.882 for the YOLOv8, medium version, demonstrating the effectiveness of both architectures for training aphid detection models. Overall, AphidCV performed slightly better for the majority of metrics and species in the study, serving its design purpose very well. YOLOv8 proved to be faster to converge the models, with the potential to apply in research considering many aphid species.**

Link to graphical and video abstracts, and to code: https://latamt.ieeer9.org/index.php/transactions/article/view/9209

***Index Terms*—aphids, AphidCV, classification, comparative study, object detection, YOLOv8.**

I. Introdução

Afídeos (pulgões) são insetos diminutos que se alimentam da seiva de plantas de diferentes espécies, incluindo culturas de inverno, como o trigo, a cevada e a aveia. Eles podem causar danos significativos, direta ou indiretamente, por meio da sucção da seiva, e pela transmissão de toxinas e do vírus causador do nanismo amarelo, *Barley Yellow Dwarf Virus* (BYDV) [1], sendo considerados pragas na agricultura.

O monitoramento e a flutuação populacional das espécies de afídeos é uma prática importante em programas de manejo integrado de pragas, pois possibilitam tomadas de decisões mais assertivas acerca do controle destes insetos [2]. Para isso, geralmente são empregadas armadilhas de campo do tipo Moericke [3] em larga escala, por conta de sua praticidade. Após a coleta dos insetos, os mesmos são separados dos detritos e dispostos em placas de teste, e em seguida, sua contagem e classificação é realizada manualmente por entomólogos especialistas. Esse processo demanda tempo e grande concentração, gerando cansaço, e implicando no aumento de suscetibilidade a erros por parte do profissional.

Quando ocorrem sobreposições envolvendo alto agrupamento de insetos, a contagem é apenas estimada, levando em conta as dimensões visualizadas, podendo prejudicar ainda mais a precisão das contagens [4]. Sob esta ótica, se torna evidente a necessidade de automatizar este processo, visando a diminuição em larga escala do tempo de contagem e classificação de afídeos, além de minimizar possíveis erros humanos ocasionados pela tarefa laboriosa.

A fim de resolver este problema, trabalhos anteriores exploraram o uso de visão computacional e inteligência artificial para tornar a tarefa de contagem destes insetos mais rápida e escalável, sem comprometer o desempenho dos resultados. Lins *et al.* [5], apresentaram um método e software denominado AphidCV, desenvolvido com uso da linguagem Java, que utiliza métodos de visão computacional com objetivo de automatizar a contagem, classificação e mensuração de afídeos da espécie *Rhopalosiphum padi (Rp)*.

Em 2020 o trabalho de Rodriguez e Rieder introduziu a segunda versão do software anterior, sendo este o AphidCV 2.0 [6], que por sua vez possuia suporte a uma espécie adicional, *Schizaphis graminum (Sg)*. O software foi reescrito na linguagem Python e um modelo próprio de rede neural foi criado, reduzindo em 10 vezes no tempo de inferência gasto para processar imagens de placas de Petri.

A versão seguinte do software, denominada AphidCV 3.0 foi desenvolvida em um ambientre web, utilizando a arquitetura REST e o framework Django para integração com a plataforma de monitoramento de insetos Trap System [7], desenvolvida pela Embrapa. A versão recebeu suporte para mais duas espécies, sendo estas *Metopolophium dirhodum (Md)* e *Sitobion avenae (Sa)*, além de funcionalidades como aplicação de filtros de brilho e contraste nas imagens.

Por fim, bucando ampliar o alcance deste recurso e torná-lo mais acessível, Kirinus *et al.* [8], desenvolveram uma versão mobile do software, compilada para as plataformas Android e iOS. Esta versão possibilitou maior praticidade pois as imagens para análise e processamento podem ser coletadas utilizando a própria câmera do dispositivo móvel, sem necessidade de um scanner próprio. Além de manter as anteriores, foi adicionado suporte às espécies *Myzus persicae (Mp)* e *Brevicoryne brassicae (Bb)*, totalizando seis espécies.

Com objetivo de melhorar o desempenho e a assertividade do AphidCV, pode-se considerar o uso de arquiteturas do estado da arte para detecção de objetos em tempo real como, por exemplo, YOLO [9]. Entretanto, na literatura não foram

The associate editor coordinating the review of this manuscript and approving it for publication was Carlos Thomaz (*Corresponding author: Brenda Slongo Taca*).

The Brazilian National Council for Scientific and Technological Development (CNPq) has supported this work.

Brenda Slongo Taca, and R. Rieder are with the University of Passo Fundo, Passo Fundo, Brazil (e-mails: 197402@upf.br, and rieder@upf.br).

D. Lau is with the Brazilian Agricultural Research Corporation, Embrapa, Brazil (e-mail: douglas.lau@embrapa.br).

encontrados estudos que realizassem uma comparação entre uma arquitetura desenvolvida especificamente para a tarefa de detecção de insetos diminutos — neste caso, afídeos — e uma arquitetura do estado da arte mais recente, YOLOv8.

Com isso em mente, o objetivo deste trabalho é realizar um estudo comparativo de desempenho entre a arquitetura própria do software AphidCV [10] e a arquitetura YOLOv8 [11], consolidada no estado da arte para detecções de objetos em tempo real. As próximas seções apresentam trabalhos relacionados, materiais e métodos para treino e comparação dos modelos, resultados e discussão, e conclusões.

II. Trabalhos Relacionados

Verma *et al.* [12] analisaram o desempenho de três versões da arquitetura de rede neural YOLO para detecção de pragas agrícolas, sendo elas YOLOv3 [13], YOLOv4 [14] e YOLOv5 [9]. O algoritmo de identificação proposto pelos autores visava distinguir cinco diferentes classes de insetos em culturas de soja que podem ter grande impacto na produtividade. Segundo o estudo, ficou evidente pelos resultados de simulação que o YOLO é um algoritmo eficaz para localização e detecção de insetos, pois todos os modelos utilizados apresentaram bom desempenho nestas tarefas. Na comparação, a YOLOv5 destacou-se pelo melhor desempenho, com mAP@.5=0,995 e F1-Score = 0,960.

Já Tian *et al.* [15] propuseram o uso de uma rede Multi-scale Dense YOLO (MD-YOLO) para a detecção de três espécies diferentes de pragas típicas de lepidópteros em 289 imagens, com o objetivo de implementar a rede em um software de alerta precoce de pragas. O modelo era composto por três componentes principais: a parte de extração de características de imagem, a rede de fusão de recursos e o módulo de previsão. Os resultados indicaram o sucesso do MD-YOLO em detectar as espécies de praga-alvo, obtendo mAP@50 = 0,862, F1-Score = 0,791, e IoU = 0,881. De acordo com os autores, a comparação do MD-YOLO com modelos recentes evidencia sua superioridade na grande maioria das métricas de avaliação, sendo inferior somente ao YOLOv8, que obteve IoU = 0,887.

Para comparação das versões YOLOv3, YOLOv5, YOLOv7 [16] e YOLOR [17] na capacidade de identificação de abelhas polinizadoras de alfafa, Zhang *et al.* [18] propuseram o treinamento destes modelos utilizando um banco de dados com 1.000 imagens. Cada imagem continha três das espécies mais comuns de abelhas polinizadoras de alfafa. Os resultados mostraram que, para as métricas de Precision, Recall, F1-Score e mAP@50, os modelos YOLOv3 e YOLOv5 foram superiores em relação ao YOLOv7 e ao YOLOR, alcançando taxa de precisão discriminatória de cerca de 100%, mesmo quando os insetos apresentavam diferentes posturas corporais.

Pereira *et al.* [19] apresentaram uma nova estratégia para detecção e classificação de moscas brancas e cincos estágios relacionados em imagens de folhas de soja. A estratégia se baseou no algoritmo de detecção YOLOv4, modificado para uso com técnicas de aumentação de dados. As 121 imagens contendo 973 objetos anotados foram coletadas de um experimento controlado infectado com ovos de mosca branca. A análise do desempenho do modelo proposto constatou um desempenho promissor, com o mesmo alcançando F1-Score = 0,87 em comparação com o algoritmo YOLOv4 original, que obteve F1-Score = 0,80.

Com objetivo de avaliar o desempenho de duas redes convolucionais profundas, Di Domênico *et al.* [20] compararam as redes Mask R-CNN e YOLOv4, em tarefas de detecção de insetos em armadilhas por meio do software InsectCV [4]. O conjunto de dados utilizado para o treinamento possuía 309 imagens, das quais 100 eram em escala de cinza. Neste *dataset* foram contabilizados no total 26.747 insetos, que posteriormente foram rotulados e organizados em duas classes: afídeos e parasitóides. Ao analisar os resultados, verificou-se que os coeficientes de determinação para análise das amostras com YOLOv4 obteve desempenho superior em comparação ao Mask R-CNN para a classe de parasitoides (R^2, 0,95 vs 0,92).

Já Lins *et al.* [5] apresentaram uma solução, denominada AphidCV, que automatiza a contagem e a classificação de afídeos, usando métodos de visão computacional e *deep learning*. Para isso, um conjunto de 30.000 recortes de imagens de 120×120 pixels foi utilizado, sendo 90% para treinamento e 10% para validação. Eles consideraram o uso da arquitetura Inception-v3 [21]. Como resultado dos treinamentos, após 30 épocas, o modelo obteve precisão de 98,10% no conjunto de dados de validação. Foi apresentada também uma comparação das contagens manuais realizadas por especialistas e os valores obtidos com o software, considerando 40 amostras. Os resultados foram promissores na contagem e classificação (rs = 0,92579) e mensuração (r = 0,9799).

Com base nos trabalhos relacionados, torna-se evidente o fato de que estudos recentes vem apostando na YOLO como arquitetura estado da arte para detecção de insetos em imagens, com resultados que demonstram sua capacidade em realizar tarefas desta categoria. No entanto, é notável que nenhum estudo prévio realizou uma comparação direta entre YOLO e outras arquiteturas de detecção de insetos.

Nesse sentido, a proposta do presente trabalho é abordar esta comparação, considerando a arquitetura YOLOv8 (versão mais recente do estado da arte) e a arquitetura proprietária do AphidCV 3.0 (versão mais recente do software). O objetivo é fornecer uma análise que avalia o desempenho da YOLO com um modelo de classificação de afídeos já consolidado, e que contribua significativamente para o avanço do campo da detecção de insetos em imagens.

III. Materiais e Métodos

Nesta seção são apresentados os materiais e métodos utilizados para a realização do estudo comparativo entre as redes AphidCV e YOLOv8, considerado o estado da arte. Para isso são apresentadas ambas as redes, assim como o *dataset* utilizado para os treinamentos, os recursos de hardware, modelos de arquitetura e parâmetros empregados no treinamento, validação e testes dos modelos gerados.

A. AphidCV

A rede neural originalmente proposta por Rodriguez & Rieder [6], em sua versão 2.0, considera, inicialmente, a troca dos canais de cores da imagem de entrada para tons de cinza. Esta escolha foi feita pois em imagens da mesma espécie

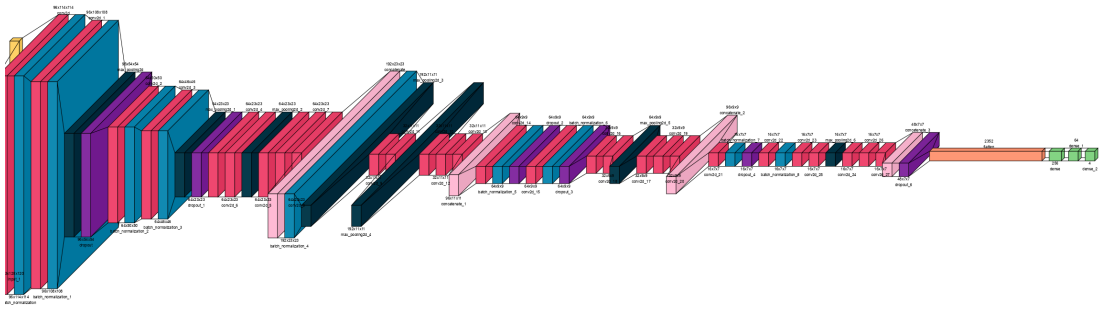

Fig. 1. Arquitetura da rede AphidCV 3.0. Representação visual gerada pela ferramenta Visualkeras [24].

quase não há diferenciação de cores, e fazer uso de três canais de cores somente para classificar diferentes formatos poderia diminuir a acurácia dos modelos.

Nas camadas internas da rede, foi utilizada a mesma técnica da InceptionV3 [21], com mais de uma convolução sobre a mesma camada, e com uma concatenação de vetores na sequência. Também suprimiu-se a conversão da imagem para tons de cinza, mantendo o suporte para imagens de entrada coloridas, culminado na versão AphidCV 3.0 [8]. A Fig. 1 apresenta em detalhes a arquitetura completa da rede.

A arquitetura AphidCV difere de algoritmos do estado da arte por ter um tamanho reduzido, o que diminui bruscamente o tempo de classificação, além de exigir um menor processamento. A taxa de aprendizado definida foi de 0,001, e utilizou-se para retropropagação o algoritmo de descida gradiente estocástica. Para implementação, utilizou-se a linguagem Python 3.10.12 e recursos da biblioteca TensorFlow 2.8.0.

A AphidCV 3.0 também aplica filtro de normalização no pré-processamento de todas as imagens de entrada. Por padrão, a AphidCV aplica técnicas de aumentação de dados, considerando transformações geométricas (rotação, espelhamento e redimensionamento) e filtros convolucionais (borramento, nitidez, relevo, dilatação, erosão, equalização de histograma e variação de brilho e contraste), com probabilidade de aplicação de 25% por época. Essas transformações usam recursos das bibliotecas albumentations 1.4.21 (*RandomRotate90, Flip, Affine (shear=[-45, 45], scale=[0.5, 1.5]), Blur, Sharpen, Emboss, CLAHE e RandomBrightnessContrast*) [22] e OpenCV 4.10 (*Opening e Closing*) [23].

B. YOLOv8

Baseando-se nas versões anteriores de algoritmos YOLO, a arquitetura da YOLOv8 é formada por um backbone de última geração, projetado para resultar em melhor desempenho da extração de características e da detecção de objectos. Sua arquitetura é livre de âncoras, o que contribui para melhorar a precisão e tornar o processo de detecção mais eficiente em comparação com abordagens anteriores [25]. A YOLOv8 foi preparada para ser veloz, precisa e fácil de usar, características que contribuem para que diferentes domínios de problema possam utilizá-la em tarefas que envolvem classificação de objetos de interesse e segmentação de imagens [26].

A YOLOv8 foi escolhida para o treinamento dos modelos deste comparativo pois é considerada estado da arte na detecção de objetos em tempo real, no momento em que este estudo foi realizado [27]. A Fig. 2 apresenta em detalhes a arquitetura completa da rede YOLOv8, versão “medium” (YOLOv8m), utilizada nesse estudo.

C. Dataset

Neste estudo, foram utilizadas 48.000 imagens para a criação de um *dataset* balanceado, selecionadas aleatoriamente de outro *dataset* mais robusto, fornecido pela Embrapa Trigo. Esse conjunto considerou imagens de seis espécies de afídeos (*Bb, Md, Mp, Rp, Sg, Sa*).

Cada conjunto de espécie é subdividido em quatro classes, sendo três estádios de desenvolvimento do afídeo (ninfas, ápteros e alados) e uma categoria relacionada a exosqueletos e detritos (falsos), como mostra a Fig. 3. Cada espécie possui 8.000 imagens, com cada classe contendo 2.000 imagens. As imagens com insetos foram automaticamente recortadas a partir de imagens digitalizadas de placas de Petri, por meio da ferramenta CropAphid [5], e selecionadas posteriormente por um especialista em entomologia.

Não foi necessário o uso de um software de rotulação do *dataset*, uma vez que cada imagem já possui somente o objeto de interesse (inseto) no centro dela, e organizada em pasta de classe correspondente, arranjadas em subpastas por espécie. As rotulações foram geradas por um script em Python que realiza a busca por todas as imagens de extensão .jpg no diretório do *dataset*, e em seguida, cria um arquivo de texto, de mesmo nome para cada uma delas, mudando somente a extensão. Em cada arquivo .txt, atribui-se um número para a classe do inseto presente na imagem (0 = “alado”, 1 = “aptero”, 2 = “falso” ou 3 = “ninfa”), seguido das coordenadas “0.5 0.5 1 1” que indicam sua posição centralizada.

Em relação à distribuição do conjunto de dados, manteve-se a proporção 80/20, por espécie, para as tarefas de treinamento (1.600 por classe x 4 = 6.400 imagens) e de validação (400 por classe x 4 = 1.600 imagens).

D. Recursos Computacionais

O treinamento e a validação dos modelos foram realizados em uma máquina equipada com o processador Intel Core i7-6950X de 6ª geração, com 10 núcleos, e 20 threads, com a frequência máxima de clock podendo chegar a 4.0 GHz. A memória RAM era de 32G. A unidade de processamento gráfico utilizada foi uma GeForce GTX TITAN X com 12 GB de memória dedicada, capacidade computacional 5.2.

Esse equipamento estava configurado com o sistema operacional Ubuntu versão 22.04 LTS, com CUDA toolkit versão 11.8 e biblioteca cuDNN versão 8.7.

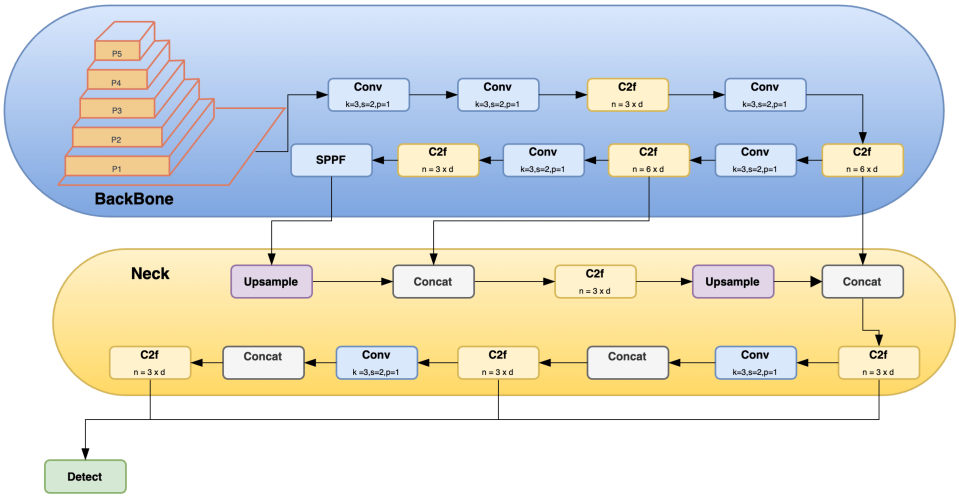

Fig. 2. Arquitetura da rede YOLOv8m. Representação visual retirada de [28].

Fig. 3. Exemplos de afídeos da espécie *Rp* para ilustrar as classes em cada conjunto, na ordem (esquerda para direita): ninfa, áptero, alado e falso.

E. Treinamento

Para treinar ambas as redes, parâmetros consistentes foram aplicados para garantir uma comparação padronizada. Para tanto, definiu-se um método que considera o uso de uma configuração em comum de parâmetros de treinamento, de aplicação de aumentação de dados, e de métricas de avaliação.

Primeiramente, os seguintes parâmetros em comum de treinamento foram assim configurados:

1) **Epochs -** Por padrão, o número de épocas foi configurado como 150;
2) **Batchsize -** Foram transferidas para os treinamentos 100 imagens a cada iteração;
3) **Patience -** Este parâmetro foi definido como 10, assim, se durante 10 épocas o modelo não apresentasse melhora na acurácia de validação dos resultados, o treinamento era interrompido para evitar *overfitting*.

Além disso, fez-se inclusão de aumentação de dados na YOLOv8m aplicando os mesmos filtros convolucionais e transformações utilizados pela AphidCV 3.0, citadas na Seção III-A.

O retorno dos treinamentos realizados com o YOLO informam os valores obtidos para as seguintes métricas: *Precision, Recall, mAP@50,* e *mAP@50-95*. A *Accuracy* foi calculada posteriormente com base nos dados de matriz de confusão, obtidos nas saídas. O *F1-Score* foi calculado com base nos valores de *Precision* e *Recall*.

Já o retorno dos treinamentos onde foi utilizado o AphidCV informavam os valores obtidos das métricas: *Accuracy, Precision, Recall, F1-Score, ROC AUC* e *PR AUC*.

Em razão disso, para comparação de desempenho, o estudo considera apenas as métricas comuns a ambas as arquiteturas: *Accuracy, Precision, Recall e F1-Score*. De acordo com Shiri *et al.* [29], essa padronização facilita uma análise precisa do desempenho de cada modelo, possibilitando a comparação entre abordagens. As demais métricas obtidas são utilizadas para análise individual complementar.

IV. Resultados e Discussão

Esta seção apresenta e analisa os resultados alcançados nos treinamentos realizados com ambas as redes, considerando desempenho, gasto computacional e vantagens de cada arquitetura na identificação de afídeos. Como abordado na Seção II, o objetivo é verificar se uma arquitetura estado da arte tem um desempenho ajustado à tarefa de detecção de insetos diminutos em imagens, sem alterações significativas em parâmetros ou camadas. Nesse âmbito, fez-se a comparação com uma arquitetura desenvolvida especificamente para esse fim. Além disso, contrastar diferentes modelos permite selecionar o modelo mais adequado para determinado estudo [30].

A geração dos modelos considerou a padronização apresentada na Seção III-E. Cada execução recebeu como entrada o mesmo conjunto de imagens por espécie do *dataset* disponível. Para organizar esse processo, arquivos de configuração YAML foram criados para cada espécie de afídeo, contendo: o caminho para o diretório do *dataset*, as pastas contendo as imagens de treinamento e de validação, o número e o nome das classes.

Seguiu-se a seguinte ordem, em relação às espécies, para compilação dos modelos: Bb, Md, Mp, Rp, Sg e Sa. Inicialmente, foram gerados os modelos da arquitetura AphidCV 3.0. Em seguida, gerou-se os modelos da YOLOv8m.

A Tabela I e a Tabela II apresentam as métricas de desempenho obtidas para ambas as redes, para cada espécie de afídeo. Os valores apresentados consideram somente métricas obtidas para o conjunto de validação, pois estas imagens não foram usadas durante o treinamento. Desta forma, é possível testar a capacidade de generalização de cada modelo.

Embora não tenha havido diferença significativa na acurácia dos modelos treinados, o modelo AphidCV 3.0 apresentou

TABELA I
RESULTADOS DE CLASSIFICAÇÃO COM YOLOV8M, CONSIDERANDO O CONJUNTO DE VALIDAÇÃO

	YOLOv8m							
Espécie	Tempo de treino	Parada antecipada	Acurácia	Precisão	Revocação	F1-Score	mAP@50	mAP@50-95
Brevicoryne brassicae (Bb)	29 min 38 s	50 épocas	0,845	0,891	0,887	**0,889**	0,950	0,950
Metopolophium dirhodum (Md)	28 min 8 s	48 épocas	0,817	0,873	0,887	**0,880**	0,941	0,940
Myzus persicae (Mp)	27 min 0s	47 épocas	0,800	0,857	0,886	**0,871**	0,918	0,918
Rhopalosiphum padi (Rp)	14 min 13 s	25 épocas	0,847	0,890	0,893	**0,891**	0,944	0,943
Schizaphis graminum (Sg)	17 min 38 s	30 épocas	0,827	0,866	0,893	**0,879**	0,942	0,939
Sitobion avenae (Sa)	18 min 54 s	32 épocas	0,847	0,864	0,896	**0,880**	0,931	0,930

TABELA II
RESULTADOS DE CLASSIFICAÇÃO COM APHIDCV 3.0, CONSIDERANDO O CONJUNTO DE VALIDAÇÃO

	AphidCV 3.0							
Espécie	Tempo de treino	Parada antecipada	Acurácia	Precisão	Revocação	F1-Score	ROC AUC	PR AUC
Brevicoryne brassicae (Bb)	45 min 45 s	77 épocas	0,872	0,883	0,864	**0,873**	0,973	0,935
Metopolophium dirhodum (Md)	58 min 22 s	104 épocas	0,903	0,908	0,896	**0,902**	0,981	0,953
Myzus persicae (Mp)	61 min 10 s	109 épocas	0,868	0,877	0,857	**0,867**	0,973	0,933
Rhopalosiphum padi (Rp)	34 min 09 s	59 épocas	0,903	0,908	0,896	**0,902**	0,981	0,953
Schizaphis graminum (Sg)	33 min 57 s	56 épocas	0,914	0,920	0,913	**0,916**	0,990	0,975
Sitobion avenae (Sa)	48 min 07 s	81 épocas	0,887	0,894	0,882	**0,888**	0,982	0,952

desempenho superior em todos os casos, bem como na maioria dos resultados da métrica de precisão. Em contrapartida, o modelo YOLOv8m obteve os resultados mais favoráveis para a métrica de revocação.

Quanto ao F1-Score, média harmônica entre *Precision* e *Recall*, os modelos comparados obtiveram resultados próximos. Porém, novamente o AphidCV 3.0 retornou resultados levemente superiores para a maioria dos modelos analisados, exceto nas espécies *Bb* (F1-Score = 0,873) e *Mp* (F1-Score = 0,867), onde obteve resultados inferiores em comparação aos modelos das mesmas espécies treinados pela YOLOv8m (respectivamente (F1-Score = 0,889 e 0,871).

Estes valores próximos das métricas de avaliação entre as abordagens destacam o potencial de ambas as redes. No caso da AphidCV 3.0, pela sua especificidade em ser projetada para a identificação e classificação de afídeos, destinado ao uso de pesquisadores da área agrícola, entomólogos e biólogos. E no caso da YOLOv8m, pela sua generalidade e extensibilidade, uma vez que sua arquitetura é flexível o suficiente para o aprendizado de padrões independente do domínio de problema.

As espécies que obtiveram os valores de F1-Score mais elevados (acima de 0,9) foram: *Md* e *Rp*, ambas com 0,902, e *Sg*, com 0,916. Observa-se que os três melhores modelos foram treinados com AphidCV 3.0. Porém, pode-se também visualizar na Tabela I e na Tabela II que não existe diferença significativa entre as arquiteturas. No geral, a maior vantagem métrica do AphidCV 3.0 é de 0,037 (*Sg*), e da YOLOv8m é de 0,016 (*Bb*). E, calculando a média geral de ambas, a AphidCV apresenta uma leve vantagem de 0,01 em relação a YOLO (F1-Score, 0,891 vs 0,881).

Além do melhor desempenho geral, AphidCV 3.0 apresenta vantagens individuais, sendo estas representadas pelas métricas ROC AUC e PR AUC. Elas indicam, respectivamente, a capacidade do modelo distinguir entre classes positivas e negativas independente do limiar de decisão escolhido, e o seu desempenho geral em encontrar e prever corretamente todas as instâncias positivas. Modelos com valores de AUC próximos de 1 possuem excelente capacidade de distinguir entre as classes, enquanto que em modelos com valores próximos de 0 ocorre o contrário [31]. Dentre todos os modelos, a menor porcentagem obtida, considerando tanto ROC AUC quanto PR AUC foi de 0,933, o que deixa claro a elevada capacidade de distinção dos modelos treinados.

Por sua vez, a rede YOLOv8m oferece métricas de análise de sumarização em valor da área que fica abaixo da curva precision x recall, como mAP@50 e mAP@50-95. Elas consideram a média das precisões para cada classe considerando os limites de interseção sobre união (*IoU, Intersection over Union*), variando de 0 a 1 - quanto mais próximos de 1 forem os valores obtidos nestas métricas, maior é a precisão do modelo [32]. Tanto os valores obtidos em mAP@50 quanto em mAP@50-95 com a utilização desta rede foram muito próximos, além de superiores a 0,918, o que indica que os modelos possuem alta precisão, com resultados fortemente confiáveis na ampla maioria das detecções.

Um panorama interessante é a discrepância no número de épocas que as arquiteturas levam para convergir cada modelo por espécie - o que também está associado ao tempo de treinamento. Observa-se que, na média, a YOLOv8m leva, aproximadamente, metade do tempo para gerar seus modelos ($\overline{x}_{(YOLOv8m)} = 39$, $\overline{x}_{(AphidCV3.0)} = 81$), mesmo tendo um número maior de camadas e, consequentemente, de parâmetros. Essa diferença significativa fica evidente no número de épocas necessários para AphidCV 3.0 gerar determinados modelos, como *Md* ($+56$), *Mp* ($+62$), e *Sa* ($+49$).

Entende-se que isso pode estar relacionado às diferentes técnicas aplicadas à arquitetura YOLOv8 para reduzir sua complexidade computacional de processamento e, consequentemente, melhorar seu desempenho, como ajustes nas primeiras camadas e no bloco principal para a extração de características (*stem tweak*, *backbone bottleneck tweak* e *backbone building block swap*), e no pescoço (*path aggregation network*) e cabeça da rede (*anchor-free head*) [33]. De fato, a combinação dessas abordagens tende a otimizar o tempo

de inferência da rede, sem afetar o desempenho [34]. Como a complexidade computacional não foi alvo desse trabalho, entende-se que é recomendável um estudo futuro para avaliar métodos e técnicas empregados em cada arquitetura, como forma de otimizar o modelo AphidCV 3.0.

De todo modo, o tempo de treinamento é um fator relevante na criação e manutenção de modelos para ferramentas de monitoramento de flutuação populacional de afídeos. Enquanto o suporte é destinado a um baixo número de espécies para as quais são gerados modelos de detecção, o AphidCV 3.0 se torna a melhor opção por conta dos seus melhores resultados. Porém, a partir do momento em que se pretende gerar modelos para dar suporte a um número maior de espécies, e trabalhar com *datasets* maiores, a utilização da YOLOv8m acaba sendo mais viável, por conta de sua rapidez em obter precisão muito similar ao AphidCV 3.0.

Por fim, observou-se também que determinados conjuntos de espécies, como *Bb* e *Mp*, possuem imagens com nitidez inferior quando comparadas com as demais, além de suas classes possuírem imagens consideravelmente semelhantes entre si. Nota-se também que elas tenderam a levar mais tempo de treinamento para convergir seus modelos, o que pode ter comprometido o desempenho dos mesmos. Nesses casos, a solução é gerar novos modelos que considerem um número maior de imagens no *dataset*, por espécie.

V. Conclusão

Este trabalho apresentou uma comparação de desempenho entre duas arquiteturas de redes neurais convolucionais na tarefa de detecção de afídeos em imagens digitais. Com base nos resultados, pode-se afirmar que a AphidCV 3.0 apresentou, no geral, métricas de avaliação levemente superiores à YOLOv8m, atendendo bem ao seu propósito específico de detecção de afídeos, útil para ferramentas com foco no monitoramento da flutuação populacional deste tipo de inseto.

A comparação direta entre as arquiteturas possibilitou concluir que ambas possuem desempenho infimamente próximo, porém apresentam peculiaridades distintas. Enquanto a AphidCV se destaca por sua alta precisão, a YOLOv8 demonstra maior flexibilidade, eficiência e capacidade de generalização.

O estudo também contribuiu mostrando que a YOLOv8m é mais rápida na convergência de modelos, levando, aproximadamente, metade do tempo para obter métricas de avaliação similares às da AphidCV 3.0. Isso sugere que, em treinamentos envolvendo mais classes ou espécies, seu uso tende a ser vantajoso para minimizar gastos computacionais.

Nesse âmbito, a comparação elucida que a YOLOv8m tem potencial de aplicação para tarefas de identificação e classificação de insetos minúsculos. A tendência é que seja possível gerar novos modelos de forma eficaz e precisa, sem necessidade de alterações significativas em suas configurações, com desempenho igual ou superior à AphidCV 3.0. Ademais, pode-se aproveitar o fato da evolução contínua da estrutura unificada da YOLO na comunidade científica, considerando recursos de versões mais recentes.

O resultado desse estudo também indica possíveis otimizações a serem incorporadas pela AphidCV, em caso de dar seguimento à evolução de sua arquitetura, como a inclusão de *path aggregation network* ou outros métodos baseados em *gradient path planning* [35].

Como trabalhos futuros, sugere-se trabalhar na evolução computacional da arquitetura AphidCV, como forma de aperfeiçoar o processo de detecção em tarefas de classificação de afídeos; incorporar os modelos YOLOv8 na plataforma de monitoramento BIS [36], e avaliar seu desempenho em tarefas cotidianas realizadas por pesquisadores da área agrícola; ampliar o *dataset* com um montante maior de imagens por espécie, visando melhorar a assertividade dos modelos e dar suporte a novas espécies, contribundo significativamente para o avanço da detecção de insetos em imagens.

Agradecimentos

Este trabalho faz parte do escopo dos projetos "Desenvolvimento e validação de ferramentas para monitoramento e tomada de decisão de manejo de epidemias causadas por vírus transmitidos por insetos", Chamada Universal CNPq/MCTI/FNDCT No. 18/2021 - Faixa A Grupos Emergentes; e "Soluções de Visão Computacional para manejo de epidemias causadas por vírus transmitidos por insetos", Chamada CNPq No 08/2022. Agradecimentos ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo auxílio financeiro e de cota de iniciação científica, e pela concessão de Bolsa de Produtividade em Desenvolvimento Tecnológico e Extensão Inovadora - DT ao pesquisador Rafael Rieder, processo No. 302773/2022-3.

References

[1] D. Lau, T. B. Mar, C. D. R. dos Santos, E. Engel, P. R. d. V. da Silva *et al.*, "Advances in understanding the biology and epidemiology of barley yellow dwarf virus (bydv)," in *Achieving durable disease resistance in cereals*. Burleigh Dodds Science Publishing, 2021. [Online]. Available: https://doi.org/10.1201/9781003180715

[2] P. Batz, T. Will, S. Thiel, T. M. Ziesche, and C. Joachim, "From identification to forecasting: the potential of image recognition and artificial intelligence for aphid pest monitoring," *Frontiers in Plant Science*, vol. 14, p. 1150748, 2023. [Online]. Available: https://doi.org/10.3389/fpls.2023.1150748

[3] T. F. Döring, "How aphids find their host plants, and how they don't," *Annals of Applied Biology*, vol. 165, no. 1, pp. 3–26, 2014. [Online]. Available: https://doi.org/10.1111/aab.12142

[4] T. D. C. Júnior, R. Rieder, J. R. Di Domênico, and D. Lau, "InsectCV: A system for insect detection in the lab from trap images," *Ecological Informatics*, vol. 67, p. 101516, 2022. [Online]. Available: https://doi.org/10.1016/j.ecoinf.2021.101516

[5] E. A. Lins, J. P. M. Rodriguez, S. I. Scoloski, J. Pivato, M. B. Lima, J. M. C. Fernandes, P. R. V. da Silva Pereira, D. Lau, and R. Rieder, "A method for counting and classifying aphids using computer vision," *Computers and Electronics in Agriculture*, vol. 169, p. 105200, 2020. [Online]. Available: https://doi.org/10.1016/j.compag.2019.105200

[6] J. P. Rodriguez and R. Rieder, "AphidCV 2.0: uma nova abordagem de classificação, contagem e mensuração de afídeos," in *Anais Estendidos da XXXIII Conference on Graphics, Patterns and Images*. Porto Alegre, RS, Brasil: SBC, 2020, pp. 159–162. [Online]. Available: https://doi.org/10.5753/sibgrapi.est.2020.13001

[7] IFSUL Passo Fundo. (2016) Trap System. Accessed on Ago. 12, 2024. [Online]. Available: http://gpca.passofundo.ifsul.edu.br/traps/

[8] N. Kirinus, B. Taca, M. Iora, R. Rieder, T. C. Junior, and D. Lau, "AphidCV mobile: ferramenta para classificação e contagem de afídeos em dispositivos móveis," in *Anais Estendidos da XXXVI Conference on Graphics, Patterns and Images*. Porto Alegre, RS, Brasil: SBC, 2023, pp. 144–147. [Online]. Available: https://doi.org/10.5753/sibgrapi.est.2023.27468

[9] G. Jocher. (2020) Ultralytics YOLOv5. Https://github.com/ultralytics/yolov5. [Online]. Available: https://doi.org/10.5281/zenodo.3908559

[10] N. W. Kirinus, E. G. Pessolano, D. Lau, T. De Cesaro Jr., and R. Rieder. (2024) AphidCV 3.0: Abordagem web integrada a uma plataforma de monitoramento de insetos. Accessed on Ago. 12, 2024. [Online]. Available: https://periodicos.ifsul.edu.br/index.php/ctipf/article/view/3382/2269

[11] G. Jocher, A. Chaurasia, and J. Qiu. (2023, Jan.) Ultralytics YOLO. Accessed on Ago. 12, 2024. [Online]. Available: https://github.com/ultralytics/ultralytics

[12] S. Verma, S. Tripathi, A. Singh, M. Ojha, and R. R. Saxena, "Insect detection and identification using YOLO algorithms on soybean crop," in *TENCON 2021 - 2021 IEEE Region 10 Conference (TENCON)*, 2021, pp. 272–277. [Online]. Available: https://doi.org/10.1109/TENCON54134.2021.9707354

[13] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," *arXiv*, no. 1804.02767, 2018. [Online]. Available: https://doi.org/10.48550/arXiv.1804.02767

[14] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," *arXiv*, no. 2004.10934, 2020. [Online]. Available: https://doi.org/10.48550/arXiv.2004.10934

[15] Y. Tian, S. Wang, E. Li, G. Yang, Z. Liang, and M. Tan, "MD-YOLO: Multi-scale dense YOLO for small target pest detection," *Computers and Electronics in Agriculture*, vol. 213, p. 108233, 2023. [Online]. Available: https://doi.org/10.1016/j.compag.2023.108233

[16] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," *arXiv*, no. 2207.02696, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2207.02696

[17] H.-S. Chang, C.-Y. Wang, R. R. Wang, G. Chou, and H.-Y. M. Liao, "YOLOR-based multi-task learning," *arXiv*, no. 2309.16921, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2309.16921

[18] C.-J. Zhang, T. Liu, J. Wang, D. Zhai, Y. Zhang, Y. Gao, H.-Z. Wu, J. Yu, and M. Chen, "Evaluation of the YOLO models for discrimination of the alfalfa pollinating bee species," *Journal of Asia-Pacific Entomology*, vol. 27, no. 1, p. 102195, 2024. [Online]. Available: https://doi.org/10.1016/j.aspen.2023.102195

[19] R. de Castro Pereira, E. Hirose, O. L. Ferreira de Carvalho, R. M. da Costa, and D. L. Borges, "Detection and classification of whiteflies and development stages on soybean leaves images using an improved deep learning strategy," *Computers and Electronics in Agriculture*, vol. 199, p. 107132, 2022. [Online]. Available: https://doi.org/10.1016/j.compag.2022.107132

[20] J. D. Domênico, D. Lau, D. Ribeiro, R. Rieder, and T. C. Júnior, "Um estudo comparativo de redes convolucionais profundas para detecção de insetos em imagens," in *Anais Estendidos da XXXIV Conference on Graphics, Patterns and Images*. Porto Alegre, RS, Brasil: SBC, 2021, pp. 183–188. [Online]. Available: https://doi.org/10.5753/sibgrapi.est.2021.20036

[21] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2818–2826. [Online]. Available: https://doi.org/10.1109/CVPR.2016.308

[22] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: Fast and flexible image augmentations," *Information*, vol. 11, no. 2, 2020. [Online]. Available: https://doi.org/10.3390/info11020125

[23] OpenCV Team. (2024) Opencv - open computer vision library. Accessed on Ago. 12, 2024. [Online]. Available: https://opencv.org

[24] P. Gavrikov. (2020) Visualkeras. Accessed on Ago. 12, 2024. [Online]. Available: https://github.com/paulgavrikov/visualkeras

[25] J. Terven, D.-M. Córdova-Esparza, and J.-A. Romero-González, "A comprehensive review of YOLO architectures in computer vision: From YOLOv1 to YOLOv8 and YOLO-NAS," *Machine Learning and Knowledge Extraction*, vol. 5, no. 4, pp. 1680–1716, 2023. [Online]. Available: https://doi.org/10.3390/make5040083

[26] M. Sohan, T. Sai Ram, and C. V. Rami Reddy, "A review on YOLOv8 and its advancements," in *Data Intelligence and Cognitive Informatics*, I. J. Jacob, S. Piramuthu, and P. Falkowski-Gilski, Eds. Singapore: Springer Nature Singapore, 2024, pp. 529–545. [Online]. Available: https://doi.org/10.1007/978-981-99-7962-2_39

[27] D. Reis, J. Kupec, J. Hong, and A. Daoudi, "Real-time flying object detection with YOLOv8," *arXiv*, no. 2305.09972, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2305.09972

[28] N. Sharma, S. Baral, M. P. Paing, and R. Chawuthai, "Parking time violation tracking using yolov8 and tracking algorithms," *Sensors*, vol. 23, no. 13, p. 5843, 2023. [Online]. Available: https://doi.org/10.3390/s23135843

[29] F. M. Shiri, T. Perumal, N. Mustapha, and R. Mohamed, "A comprehensive overview and comparative analysis on deep learning models: Cnn, rnn, lstm, gru," *arXiv preprint arXiv:2305.17473*, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2305.17473

[30] M. Miric, N. Jia, and K. G. Huang, "Using supervised machine learning for large-scale classification in management research: The case for identifying artificial intelligence patents," *Strategic Management Journal*, vol. 44, no. 2, pp. 491–519, 2023. [Online]. Available: https://doi.org/10.1002/smj.3441

[31] S. A. Khan and Z. Ali Rana, "Evaluating performance of software defect prediction models using area under precision-recall curve (auc-pr)," in *2019 2nd International Conference on Advancements in Computational Sciences (ICACS)*, 2019, pp. 1–6. [Online]. Available: https://doi.org/10.23919/ICACS.2019.8689135

[32] R. C. Domingues, G. V. Fruet, H. F. Abud, and D. G. Gomes, "Imagens de raios-x e YOLOv8 para avaliação automatizada, precisa e não destrutiva da qualidade de sementes braquiária (urochloa brizantha)," in *Anais do XIV Congresso Brasileiro de Agroinformática*. SBC, 2023, pp. 167–174. [Online]. Available: https://doi.org/10.5753/sbiagro.2023.26555

[33] P. Vasanthi and L. Mohan, "Efficient YOLOv8 algorithm for extreme small-scale object detection," *Digital Signal Processing*, vol. 154, p. 104682, 2024. [Online]. Available: https://doi.org/10.1016/j.dsp.2024.104682

[34] E. Casas, L. Ramos, E. Bendek, and F. Rivas-Echeverria, "YOLOv5 vs. YOLOv8: Performance benchmarking in wildfire and smoke detection scenarios," *Journal of Image and Graphics*, vol. 12, no. 2, 2024. [Online]. Available: https://doi.org/10.18178/joig.12.2.127-136

[35] C.-Y. Wang, H.-Y. M. Liao, and I.-H. Yeh, "Designing network design strategies through gradient path analysis," *arXiv*, no. 2211.04800, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2211.04800

[36] Embrapa Trigo. (2024) BIS - Brazilian Insect Survey. Accessed on Nov. 12, 2024. [Online]. Available: https://www.bis.cnptia.embrapa.br

Brenda Slongo Taca Undergraduate student of Computer Engineering at the Universidade de Passo Fundo (UPF) since 2023. Fellow of a PIBIC/UPF undergraduate scholarship, working on Computing Applied to Agriculture research projects that explore Deep Learning, Image Processing, and Computer Vision methods.

Douglas Lau Ph.D. in Agronomy-Phytopathology from Universidade Federal de Viçosa in 2004. Researcher at the Brazilian Agricultural Research Corporation (EMBRAPA) since 2006. Research areas: phytopathology, virology of plants and insect vectors of pathogens, biological and molecular characterization of phytoviruses, genetic resistance of plants to pathogens, monitoring of insect and mite vectors of pathogens, epidemiology, disease management, and introduced pests monitoring.

Rafael Rieder Ph.D. in Computer Science from Pontifícia Universidade Católica do Rio Grande do Sul in 2011. Full professor and researcher at Universidade de Passo Fundo (UPF) since 2011. UPF Faculty member of the Graduate Programs in Applied Computing and Agronomy. Fellow of a CNPq Productivity in Technological Development and Innovative Extension scholarship - Level 2 since 2023. Research areas: Virtual and Augmented Reality, Deep Learning, Image Processing, and Computer Vision.