Um Estudo Comparativo de Redes Convolucionais Profundas para Detecção de Insetos em Imagens

Jéssica Regina Di Domênico
Instituto Federal de Educação, Ciência e Tecnologia Sul-rio-grandense (IFSul)
Passo Fundo, RS, Brasil
Email: jessicadomenico.pf016@academico.ifsul.edu.br

Douglas Lau
Empresa Brasileira de Pesquisa Agropecuária (Embrapa Trigo)
Passo Fundo, RS, Brasil
Email: douglas.lau@embrapa.br

Daniel Delfini Ribeiro
Instituto Federal de Educação, Ciência e Tecnologia Sul-rio-grandense (IFSul)
Passo Fundo, RS, Brasil
Email: danielribeiro@ifsul.edu.br

Rafael Rieder
Universidade de Passo Fundo (UPF)
Passo Fundo, RS, Brasil
Email: rieder@upf.br

Telmo De Cesaro Júnior
Instituto Federal de Educação, Ciência e Tecnologia Sul-rio-grandense (IFSul)
Passo Fundo, RS, Brasil
Email: telmojunior@ifsul.edu.br

***Resumo*—Esse trabalho apresenta um estudo comparativo entre dois modelos de redes convolucionais profundas em tarefas de identificação e contagem de insetos em imagens digitais, considerando afídeos (Hemiptera: Aphididae) e parasitoides (Hymenoptera: Aphelinidae e Braconidae, Aphidiinae). Nesse estudo de caso, cada imagem pode conter centenas de espécimes, detritos, sobreposições e outros insetos com morfologia semelhante, o que dificulta o processo de detecção. Nesse sentido, comparou-se os resultados obtidos pelo sistema InsectCV, baseado em Mask R-CNN, em termos de tempo de treinamento, inferência e precisão, com um novo modelo, treinado com a rede DarkNet. Com a utilização de imagens em tons de cinza, com menor dimensão, processamento via GPU e uma rede convolucional de um estágio, é possível a redução do custo computacional e elevação da precisão na tarefa de detecção de objetos. Com base em 580 imagens utilizadas para a validação do modelo proposto foi possível obter a precisão média de 79,9%.**

***Abstract*—This work presents a comparative study between two deep convolutional network models in tasks of identification and counting of insects in digital images, considering aphids (Hemiptera: Aphididae) and parasitoids (Hymenoptera: Aphelinidae and Braconidae, Aphidiinae). In this case study, each image can contain hundreds of specimens, debris, overlaps, and other insects with similar morphology, making the detection process difficult. In this sense, we compared the results obtained by the InsectCV system, which was based on Mask R-CNN, in terms of training time, inference, and precision, with a new model, trained with the DarkNet network. Using grayscale images with smaller dimensions, processing via GPU, and a one-stage convolutional network, it is possible to reduce the computational cost and increase the precision in the object detection task. Based on the 580 images used to validate the proposed model, it was possible to obtain a mean Average Precision of 79.9%.**

I. Introdução

Nas regiões tritícolas do Brasil, dependendo das condições climáticas, é comum a presença de afídeos, também conhecidos como pulgões, que podem causar perdas significativas, em função de seus hábitos alimentares e por serem vetores de agentes patogênicos, como vírus [1]. Para monitorar a flutuação populacional desses insetos-praga utiliza-se, geralmente, armadilhas. Este recurso é amplamente utilizado e de fácil implementação, permitindo capturar insetos de uma determinada área para monitoramento [2] [3]. A partir da identificação e contagem dos insetos retidos na armadilha é possível correlacionar os níveis populacionais com danos à cultura, gerando informações para embasar programas de manejo [4].

Nesse sentido, a Embrapa Trigo formou uma Rede de Monitoramento de Pragas e Cereais de Inverno [5], através de parcerias com Cooperativas, Empresas e Institutos de Pesquisas para acompanhar populações de afídeos e parasitoides em culturas como trigo, cevada, triticale e aveia [6]. Nesse programa, são utilizadas armadilhas do tipo Moericke em ambientes externos (áreas de campo), constituídas por bandejas amarelas com água e detergente que atraem e capturam insetos. Semanalmente, é realizada a separação e contagem manual do material depositado nessas armadilhas. Após esse processo de triagem, quando realizado na Embrapa Trigo, as amostras são digitalizadas com a utilização de um scanner e armazenadas em um banco de dados.

Segundo a Embrapa Trigo, a tarefa de identificação manual de insetos como parasitoides (Hymenoptera: Aphelinidae e Braconidae, Aphidiinae) e afídeos alados (Hemiptera: Aphididae) desenvolvida pela Rede de Monitoramento desde 2017 é demorada, fadigante e não escalável. Além disso, é comum amostras com sobreposição, oclusões, variações de pose e aglutinações de elementos, dificultando o reconhecimento de espécimes pelo especialista. Em casos de grandes concentrações de objetos nas lâminas de observação, a contagem manual dos espécimes de interesse é apenas estimada, com base em proporções espaciais e na experiência do especialista [7]. Nesse sentido, fica evidente a necessidade de automatizar essa tarefa, tornando-a escalável, reduzindo a alocação de recursos humanos e para incrementar a precisão.

Em termos de automatização, o trabalho de [8] propôs um sistema chamado InsectCV [9] para identificar e contabilizar

afídeos alados e parasitoides. Esse sistema utilizou uma rede neural convolucional profunda baseada em Mask R-CNN e alcançou precisão suficiente para identificar a flutuação dos níveis populacionais para esses insetos.

No InsectCV, em razão da dimensão das imagens utilizadas para o treinamento do modelo inteligente e a execução da inferência via CPU, o custo computacional foi relativamente alto segundo os autores. Sendo assim, realizou-se uma pesquisa para localizar implementações de redes convolucionais para detecção de objetos que necessitassem de menor poder computacional e com potencial para alcançar ou superar a precisão do InsectCV. Nessa busca, considerou-se os recursos de hardwares disponíveis: servidor gráfico com GPU RTX 3070 com 8 GB de VRAM para o treinamento do modelo inteligente e um segundo servidor com uma GPU RTX 2060 de 6 GB de VRAM para a inferência.

A partir da análise de resultados de trabalhos recentes e considerando os recursos de hardware citados, optou-se pelo algoritmo de código aberto YOLO (You Only Look Once) [10], que é baseado na rede convolucional DarkNet [11]. Nesse contexto, o objetivo desse trabalho é apresentar um comparativo em termos de tempo de treinamento e precisão entre dois modelos inteligentes para detecção de insetos em imagens digitais: o InsectCV [8] e o YOLO v4 [12].

Nas próximas seções são apresentados o referencial teórico, os materiais e métodos empregados no treinamento e validação do modelo inteligente, resultados, discussões e considerações finais sobre o comparativo.

A. Referencial Teórico

Recentes avanços nas áreas de Inteligência Artificial e Visão Computacional têm permitido o desenvolvimento de soluções computacionais para a agricultura, reduzindo tempo e custos em tarefas repetitivas, como a identificação e contagem de objetos em imagens digitais [13], [14]. A Rede Neural Convolucional Profunda (DCNN) [15] é uma das técnicas de aprendizado profundo que tem se destacado pelos resultados promissores nas tarefas de classificação de imagens, detecção e segmentação de objetos, superando outras técnicas, como a extração manual de caraterísticas de objetos em imagens. Kamilaris e Prenafeta-Boldú [13] apresentam uma seleção de 40 trabalhos, de 2010 a 2017, que aplicam técnicas de Inteligência Artificial na agricultura e, destes, 42% utilizam DCNNs. Na revisão apresentada por De Cesaro Júnior e Rieder [14], que selecionou 33 trabalhos de 2015 a 2019, também foi constatado a tendência pela utilização de DCNNS.

De acordo com Sun Y. *et al.* [16], DCNNs para a tarefa de detecção de objetos podem ser organizadas em dois grupos:

- Classificadores de dois estágios com *Region Proposal Network* (RPN):
  - Faster Region-Based Convolutional Neural Network (Faster R-CNN) [17];
  - Region-based Fully Convolutional Networks (R-FCN) [18];
  - Mask R-CNN [19].
- Classificadores de um estágio sem RPN:
  - Single Shot Multibox Detector (SSD) [20];
  - You Only Look Once (YOLO) [10];
  - RetinaNet [21].

Classificadores de dois estágios utilizam a RPN, assim, no primeiro estágio ocorre a geração de objetos candidatos via caixas delimitadoras *bounding boxes*. A RPN é uma rede totalmente convolucional que prediz as coordenadas do objeto na imagem, bem como, o nível de detecção [22]. O segundo estágio extrai as características destes usando *RoI-Pool*, e executa a classificação e regressão das *bounding boxes* [19]. No segundo estágio podem ocorrer variações dependendo da implementação. Mask R-CNN, por exemplo, possui uma ramificação adicional (*branch*) para segmentar objetos por meio de uma máscara em paralelo com a *branch* para reconhecer as *bounding boxes* [19].

Já os classificadores de um único estágio não possuem a RPN, realizando a proposição de regiões e classes em apenas uma etapa. Assim, comparado com a abordagem de dois estágios, nessa arquitetura o tempo necessário para a inferência da imagem é fixo, portanto, mais adequada para o desenvolvimento de aplicações que necessitem detectar objetos em tempo real [23].

O YOLO é baseado na rede neural profunda Darknet [11], desenvolvido na linguagem de programação C, utiliza a API CUDA e possui suporte para CPU e GPU. Diversas versões foram disponibilizadas, por exemplo, YOLO v2 [24] processa imagens com FPS de 40-90, enquanto YOLO v3 [25] permite a escolha entre velocidade e precisão apenas alterando o tamanho do modelo, YOLO v4 [12] é baseado no conceito de *Bag of Freebies*, aumentando a precisão do detector sem modificar sua velocidade, e *Bag of Specials*, um tipo de *Data Augmentation* que aumenta a variedade de imagens de entrada. As últimas versões disponibilizadas foram YOLO v5 [26] e PP-YOLO [27].

YOLO v4 é considerada atualmente como o estado-da-arte na tarefa de detecção de objetos em tempo real. Essa versão foi projetada para a execução em um processador gráfico convencional, como a GTX 1080Ti ou RTX 2080Ti, visando menor custo computacional [12]. Nessa versão, a detecção ocorre como um simples problema de regressão. Inicialmente, a imagem de entrada é dividida em uma matriz NxN e cada célula da grade é responsável por prever um objeto que possa estar no seu centro. Com esses dados, são geradas *bounding boxes* e uma pontuação que representa o nível de confiança do detector de que existe um objeto nesse local. Além disso, é realizada a predição da classe com base no valor de confiança *threshold*. Através dessa técnica, YOLO v4 pode reconhecer vários objetos analisando a imagem inteira, de modo que suas previsões são geradas pelo contexto global da imagem [12].

Nesse sentido, ao analisar as implementações de DCNNs de um estágio, as versões YOLO v5 e PP-YOLO foram desconsideradas em função de ainda não estarem consolidadas em termos de documentação e testes de desempenho. Sendo assim, optou-se pela utilização da versão YOLO v4 para esse comparativo. Essa versão demanda recursos de hardware

compatíveis com os disponíveis, oferece níveis de precisão próximos a abordagens de dois estágios e tem documentação disponível para embasar a implementação das customizações necessárias para a detecção de pequenos objetos, como insetos.

B. Materiais e Métodos

Na formação do conjunto de imagens para treinamento do modelo inteligente foram reunidas as 209 imagens utilizadas no InsectCV e 100 novas imagens, em formato GrayScale, totalizando 309 imagens com dimensão de aproximadamente 6156 pixels de altura por 6156 pixels de largura. Nessas imagens, os insetos de interesse foram rotulados manualmente com o auxílio da ferramenta gráfica LabelImg [1], totalizando 26.747 insetos de interesse rotulados, organizados em duas classes: afídeos e parasitoides.

Com base nos testes iniciais de processamento, houve a necessidade de dividir cada imagem em partes menores, em função da quantidade de memória VRAM da GPU RTX 2060. Sendo assim, foi desenvolvido uma rotina na linguagem de programação Python, que, utilizando funções da biblioteca OpenCV, gerou 13.642 novas imagens de 608x608 pixels a partir das 309 imagens originais. Essa rotina também recalculou a posição de cada caixa delimitadora e criou dois grupos de imagens: treinamento e testes, na proporção de 80% e 20%, respectivamente. Em seguida, o conjunto de teste foi organizado manualmente de forma que houvesse a mesma proporção entre imagens com diferentes densidades de insetos e detritos.

No treinamento do modelo os pesos da rede foram inicializados com valores aleatórios. O valor dos parâmetros *batch* e *subdivisions* foram fixados em 1, em razão da memória disponível na GPU. O tamanho da rede de entrada foi definido com base nas dimensões das imagens: 608x608. Inicialmente foram estipuladas 20.000 interações, o que corresponde aproximadamente a 1/3 das interações executadas para o treinamento do InsectCV. O número de canais foi definido em 1 em função da utilização de imagens no formato GrayScale. A quantidade de classes (2) foi definida em cada uma das três camadas *yolo*, bem como, a quantidade de filtros (21) nas camadas convolucionais que precedem as camadas *yolo*. O recurso de aumento de dados não foi ativado em função do número de imagens (13.642) e memória disponível na GPU. Os demais parâmetros pertencentes a seção *net* não foram alterados, sendo aplicados os valores sugeridos na implementação de AlexeyAB [28].

Em termos de customização para o treinamento envolvendo pequenos objetos, como é caso dos insetos de interesse, que podem ser reduzidos para a dimensão de 16x16 pixels, após as operações de redimensionamento realizadas na camadas da DarkNet, foram alterados os parâmetros *layers* e *stride* conforme as recomendações contidas na implementação de AlexeyAB [2].

[1]Disponível em: https://github.com/tzutalin/labelImg

[2]Valores disponíveis em: https://github.com/AlexeyAB/darknet#how-to-train-to-detect-your-custom-objects

Para a validação do modelo foram utilizadas as mesmas 580 imagens aplicadas para a validação do InsectCV, contendo insetos coletados em armadilhas tipo Moericke instaladas em campos da Embrapa Trigo em Passo Fundo e Coxilha, RS, durante as safras de 2019 e 2020. Essas imagens foram organizadas em 14 séries: nove séries de imagens para analisar a identificação dos afídeos e cinco séries para validar o modelo na tarefa de identificação de parasitoides.

Para realizar a inferência das imagens na etapa de validação foi necessário aplicar uma técnica de recorte nas originais que possuem a dimensão de 6156x6156 pixels, pois a GPU RTX 2060 de 6 GB de VRAM não tem capacidade para processar imagens nesse formato. Sendo assim, utilizou-se a biblioteca DarkHelp [29], codificada na linguagem de programação C e de código aberto. Através do recurso de *tiling* disponível nessa biblioteca a imagem original é dividida em partes de 608x608 pixels, correspondente ao tamanho da rede, em seguida cada recorte de imagem é inferido pelo modelo e no final do processo os objetos reconhecidos são reposicionados na imagem original.

Em relação as métricas utilizadas para a validação do modelo, foram consideradas a *precision* para indicar o nível de acerto do modelo, a *mean Average Precision* (mAP), a *Intersection Over Union IoU*, o *recall* e o *F1 score*. Além desses, os coeficientes angulares, a distorção no intercepto e os coeficientes de determinação R^2, gerados pela comparação com a identificação manual realizada pelo especialista da Embrapa. Na próxima seção são apresentados os resultados em termos de precisão para o modelo baseado no YOLO v4 e Mask R-CNN.

C. Resultados

Com a execução das 20.000 interações na etapa de treinamento o valor de perda alcançado foi de 3,39. O tempo necessário para a execução de cada mil interações foi de aproximadamente 2 horas, totalizando 40 horas de processamento com a GPU RTX 3070. O tempo para a inferência de cada imagem foi de aproximadamente 30 segundos com a utilização da GPU RTX 2060.

Com base no melhor modelo gerado pelo treinamento e limiar de aceitação de 30% foi possível alcançar os seguintes percentuais na inferência do conjunto de imagens para teste: 79,9% de *mAP*. A precisão média (AP) para a classe parasitoide foi de 82,8%, enquanto que para afídeos obteve-se AP de 77,1%. Considerando as duas classes, a *precision* foi de 0,98, *recall* de 0,61 e *F1-score* de 0,75. A métrica *average IoU* foi de 71,1%. O total de verdadeiros positivos (TP) foi de 4902, falsos positivos (FP) 118 e falsos negativos (FN) 3143.

Com relação as 580 imagens reservadas para a validação do modelo, na Tabela I e na Tabela II são listados os valores para os coeficientes obtidos na comparação com a identificação manual realizada pelo especialista da Embrapa. Portanto, para essa validação, foram organizadas nove séries referentes as safras de 2019 e 2020. Na Tabela III, são listadas outras cinco séries, referente a safra de 2020 (01/07/2020 a 30/10/2020).

Tabela I
COEFICIENTES PARA AFÍDEOS EM 2019

Modelo	Armadilha	Coeficiente Angular	Intercepto	R^2
Mask R-CNN	0	0,67 * X	+ 3,00	0,99
YOLO v4	0	0,80 * X	- 0,51	0,99
Mask R-CNN	1	0,40 * X	+ 10,77	0,90
YOLOv4	1	0,72 * X	+ 0,45	0,99
Mask R-CNN	2	0,61 * X	+ 2,01	0,98
YOLO v4	2	0,82 * X	- 0,27	0,98
Mask R-CNN	4	0,67 * X	+ 1,22	0,99
YOLO v4	4	0,73 * X	- 1,25	0,98
Mask R-CNN	0,1,2,4	**0,54 * X**	**+ 17,11**	**0,97**
YOLO v4	0,1,2,4	**0,76 * X**	**- 1,33**	**0,99**

Tabela II
COEFICIENTES PARA AFÍDEOS EM 2020

Modelo	Armadilha	Coeficiente Angular	Intercepto	R^2
Mask R-CNN	0	0,54 * X	+ 2,99	0,91
YOLO v4	0	0,52 * X	+ 1,87	0,95
Mask R-CNN	1	0,25 * X	+ 11,27	0,65
YOLO v4	1	0,24 * X	+ 11,32	0,55
Mask R-CNN	2	0,30 * X	+ 11,47	0,87
YOLO v4	2	0,39 * X	+ 6,80	0,91
Mask R-CNN	3	0,26 * X	+ 9,27	0,77
YOLO v4	3	0,35* X	+ 6,71	0,68
Mask R-CNN	4	0,28 * X	+ 8,95	0,77
YOLO v4	4	0,37 * X	+ 6,05	0,66
Mask R-CNN	0,1,2,3,4	**0,30 * X**	**+ 37,96**	**0,85**
YOLO v4	0,1,2,3,4	**0,38 * X**	**+ 26,69**	**0,78**

A coluna Modelo indica os valores obtidos pelo InsectCV baseado em Mask R-CNN ou pelo YOLO v4.

Na Figura 1 e na Figura 2 são ilustrados exemplos de inferência na detecção de afídeos e parasitoides. As caixas delimitadoras em vermelho indicam a detecção e a localização de afídeos e as verdes parasitoides. Os insetos de interesse estão numerados. Na Figura 1 existem onze afídeos, Mask R-CNN (a) identificou corretamente apenas dois afídeos. YOLO v4 (a) identificou corretamente seis 6 afídeos. Na Figura 2, que possui três parasitoides, Mask R-CNN (a) não os identificou.

Tabela III
COEFICIENTES PARA PARASITOIDES EM 2020

Modelo	Armadilha	Coeficiente Angular	Intercepto	R^2
Mask R-CNN	0	0,53 * X	+ 0,12	0,89
YOLO v4	0	0,67 * X	- 0,09	0,95
Mask R-CNN	1	0,49 * X	+ 0,64	0,80
YOLO v4	1	0,55 * X	+ 1,17	0,77
Mask R-CNN	2	0,40 * X	+ 0,13	0,75
YOLO v4	2	0,53 * X	- 0,39	0,93
Mask R-CNN	3	0,41 * X	+ 0,21	0,92
YOLO v4	3	0,57* X	+ 0,15	0,82
Mask R-CNN	4	0,32 * X	+ 0,67	0,64
YOLO v4	4	0,39 * X	+ 0,29	0,77
Mask R-CNN	0,1,2,3,4	**0,45 * X**	**+ 1,03**	**0,92**
YOLO v4	0,1,2,3,4	**0,56 * X**	**+ 0,60**	**0,95**

YOLO v4 (b) identificou corretamente todos os parasitoides, sendo que dois (1 e 3) estão com a cauda próxima ao seu corpo.

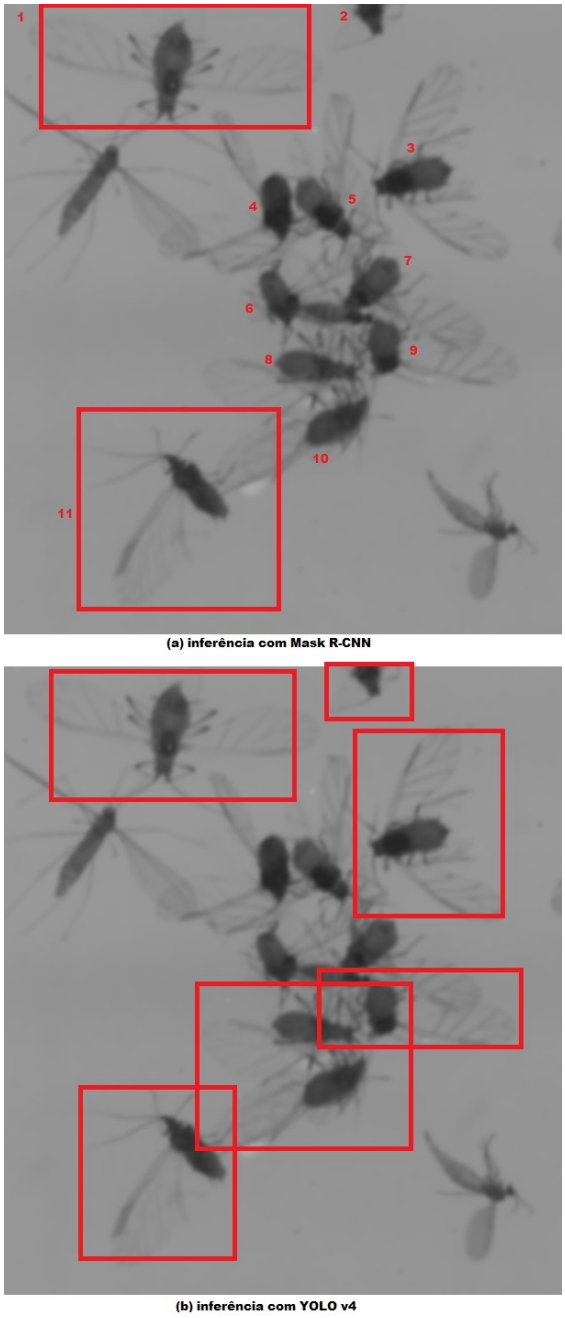

Figura 1. Caixas vermelhas: afídeos identificados pelos modelos.

D. Discussão

Em função da diferença na dimensão e quantidades de imagens utilizadas no treinamento dos modelos e recursos de hardware aplicados, não é possível afirmar que o YOLO v4 teve menor custo computacional. No entanto, a geração de imagens menores via rotina desenvolvida em Python possibilitou reduzir o tempo de treinamento e eliminação do pré-processamento para recorte aleatório. Porém, nesse processo

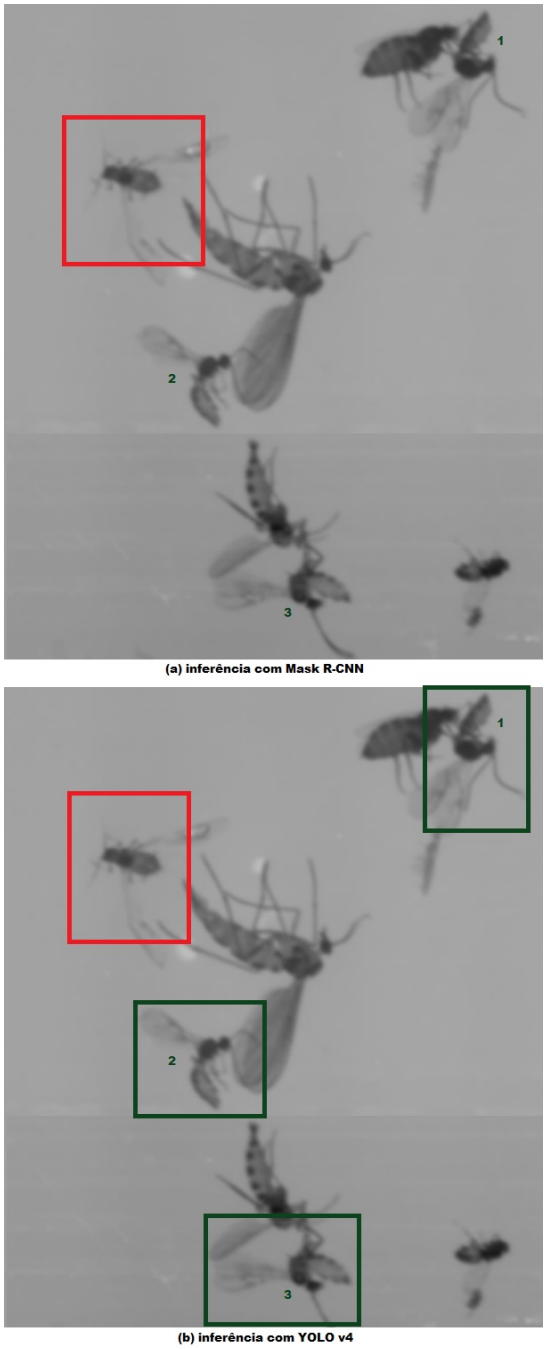

Figura 2. Caixas verdes: parasitoides identificados pelos modelos.

de divisão das imagens originais, houve perda de informação nos casos onde o inseto de interesse estava localizado nas áreas de divisão.

Para o treinamento do modelo baseado no YOLO v4, foram necessárias menos interações em relação ao modelo treinado com Mask R-CNN. Para o YOLO v4 foram executadas 20.000 interações, obtendo o valor de perda de 3,39, totalizando 40 horas de processamento. A opção de executar essa quantidade de interações foi baseada na avaliação do valor de perda, que se manteve estável nas últimas 4.000 interações. No modelo baseado em Mask R-CNN foram necessárias 62.300 para alcançar o valor de perda 1,4, totalizando aproximadamente 35 dias de processamento. Considerando apenas os valores de perda, não é possível concluir que esse dado foi determinante para a seleção do melhor modelo. Ao avaliar os resultados obtidos na validação dos modelos, verificou-se que o YOLO v4, com valor de perda superior e menor quantidade de interações por imagem, obteve melhor desempenho.

Através da implementação do recurso de *Tiling* disponível na biblioteca DarkHelp, o tempo para a inferência das imagens foi de aproximadamente 30 segundos. Em comparação com a técnica de inferência desenvolvida no InsectCV, na qual a imagem original era reduzida para a dimensão de 5568x5568 e inferida pelo modelo via processamento em CPU, houve significativa redução no tempo de processamento. Nesse sentido, a estratégia de divisão da imagem proposta nesse trabalho, possibilitou a execução via GPU, uma vez que 6GB de VRAM não é suficiente para processar uma imagem com dimensão de 6156x6156 pixels. Além disso, essa estratégia elimina a necessidade de pré-processamento para reduzir o tamanho da imagem, gerando perdas de detalhes em objetos pequenos. No entanto, existe a probabilidade da geração de falsos positivos para os insetos que estão localizados nas áreas de divisão.

Ao comparar os coeficientes apresentados na Tabela I para os dois modelos na tarefa de detecção de afídeos nas 580 imagens utilizadas para a validação, o YOLO v4 teve desempenho superior em comparação ao Mask R-CNN. A distorção no intercepto para as populações mais baixas foi de apenas -1,33 enquanto que no Mask R-CNN esse valor foi de +17,11.

Para as quatro séries de 2020, cujas imagens contém mais detritos em relação ao ano anterior, a precisão do YOLO v4 na identificação de afídeos também foi superior ao Mask R-CNN (Tabela II). No entanto, o desempenho em relação ao ano anterior foi menor em virtude da dificuldade do modelo em detectar insetos conectados e com a presença de muitos detritos. Essa constatação também foi destacada no InsectCV. Nessas séries, o $R^2 = 0,85$ alcançado pelo Mask R-CNN foi maior que o do YOLO v4 ($R^2 = 0,78$).

Em relação ao desempenho na identificação de parasitoides, conforme os dados apresentados na Tabela III, o modelo YOLO v4 alcançou coeficiente mais próximo de 1, menor valor para a distorção no intercepto e $R^2 = 0,95$, superando os níveis obtidos pelo Mask R-CNN. No entanto, essas diferença não foram significativas.

Em termos de capacidade do modelo YOLO v4 na detecção de afídeos em diferentes poses e parasitoides em diferentes ângulos da cauda, foi constatado que houve incremento em relação ao modelo treinado com Mask R-CNN. Na Figura 1 e na Figura 2 é possível visualizar casos que demonstram essa evolução.

II. Considerações Finais

Nesse trabalho foi apresentada uma comparação entre dois modelos inteligentes para a detecção e contagem de insetos em imagens digitais. Pode-se concluir que as técnicas aplicadas para o treinamento do modelo inteligente baseado em YOLO

v4 possibilitaram alcançar maior precisão em comparação com o modelo utilizado no InsectCV, o qual foi baseado no Mask R-CNN. Além disso, as estratégias de divisão de imagens e a utilização de GPU para treinamento e inferência diminuíram significativamente o tempo de processamento, sem necessariamente reduzir a precisão na identificação de afídeos e parasitoides. Houve também incremento na capacidade de detecção de parasitoides em diferentes posições de cauda e afídeos em poses variadas.

Em relação ao aprimoramento do modelo é válido aplicar a técnica de aumento de dados para incrementar o conjunto de imagens para treinamento e o incremento da quantidade de interações, visando a redução do valor de perda. Outro aprimoramento que pode ser aplicado para esse estudo de caso é a utilização de imagens no formato RGB, referentes a amostras de insetos recentes, que ainda mantém a sua coloração original. A avaliação de versões mais recentes do YOLO, bem como, o estudo de outras abordagens convolucionais de um estágio é relevante para a continuidade dessa pesquisa. Em relação as imagens utilizadas para a validação dos modelos é relevante agrupá-las de acordo com a proporção de objetos de interesse, detritos e parasitoides com variação do ângulo da cauda e pose, para que se possa estimar o desempenho dos modelos em cada grupo de forma mais precisa. Em termos de triagem das armadilhas e geração das imagens, é necessário revisar os procedimentos de campo para reduzir a coleta de imagens muito densas, que geram um ambiente altamente confuso para análise do modelo.

REFERÊNCIAS

[1] M. Savaris, S. Lampert, J. Salvadori, D. Lau, P. d. S. Pereira, and M. Smaniotto, "Population growth and damage caused by rhopalosiphum padi (l.)(hemiptera, aphididae) on different cultivars and phenological stages of wheat," *Neotropical entomology*, vol. 42, no. 5, pp. 539–543, 2013.

[2] G. D. HEATHCOTE, "The comparison of yellow cylindrical, flat and water traps, and of johnson suction traps, for sampling aphids," *Annals of Applied Biology*, vol. 45, no. 1, pp. 133–139, 1957. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1744-7348.1957.tb00449.x

[3] R. Morris, "First experiences with water traps," *Methodology*, 2018.

[4] L. C. Wright and W. W. Cone, "Population Dynamics of Brachycorynella asparagi (Homoptera: Aphididae) on Undisturbed Asparagus in Washington State," *Environmental Entomology*, vol. 17, no. 5, pp. 878–886, 10 1988.

[5] D. Lau, "Plataforma integrada para monitoramento, simulação e tomada de decisão no manejo de epidemias causadas por vírus transmitidos por insetos." in *X Simpósio Sobre Atualidades em Fitopatologia*, 2020, pp. 83–91. [Online]. Available: http://ainfo.cnptia.embrapa.br/digital/bitstream/item/221717/1/DOuglasLau-2020-Anais.pdf

[6] E. Trigo, "Rede de monitoramento de pragas em cereais de inverno," 2015. [Online]. Available: https://www.embrapa.br/busca-de-noticias/-/noticia/43111123/treinamento-para-aprimorar-tecnicas-de-identificacao-de-insetos-em-cereais-de-inverno

[7] 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: http://www.sciencedirect.com/science/article/pii/S0168169919306039

[8] T. De Cesaro Jr., "InsectCV: um sistema para detecção de insetos em imagens digitais," Master's thesis, Programa de Pós-Graduação em Computação Aplicada, 2020, instituto de Ciências Exatas e Geociências – ICEG. [Online]. Available: http://tede.upf.br:8080/jspui/handle/tede/1956

[9] T. De Cesaro Jr., R. Rieder, D. Lau, and J. R. D. Domenico, "InsectCV," Programa de Computador. Número do registro: BR512021000542-2, data de registro: 19/03/2021, Instituição de registro: INPI - Instituto Nacional da Propriedade Industrial.

[10] 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.

[11] J. Redmon, "Darknet: Open source neural networks in c," http://pjreddie.com/darknet/, 2013–2016.

[12] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," 2020.

[13] A. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey," *Computers and Electronics in Agriculture*, vol. 147, pp. 70–90, 2018. [Online]. Available: https://doi.org/10.1016/j.compag.2018.02.016

[14] T. De Cesaro Jr. and R. Rieder, "Automatic identification of insects from digital images: A survey," *Computers and Electronics in Agriculture*, vol. 178, p. 105784, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0168169920311224

[15] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015. [Online]. Available: https://doi.org/10.1038/nature14539

[16] Y. Sun, X. Liu, M. Yuan, L. Ren, J. Wang, and Z. Chen, "Automatic in-trap pest detection using learning for pheromone-based Dendroctonus valens monitoring," *Biosystems Engineering*, vol. 176, pp. 140–150, dec 2018. [Online]. Available: https://doi.org/10.1016/j.biosystemseng.2018.10.012

[17] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016. [Online]. Available: https://doi.org/10.1109/TPAMI.2016.2577031

[18] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, ser. NIPS'16. Red Hook, NY, USA: Curran Associates Inc., 2016, p. 379–387. [Online]. Available: https://dl.acm.org/doi/10.5555/3157096.3157139

[19] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2980–2988. [Online]. Available: https://doi.org/10.1109/ICCV.2017.322

[20] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 21–37. [Online]. Available: https://doi.org/10.1007/978-3-319-46448-0_2

[21] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 318–327, 2018. [Online]. Available: https://doi.org/10.1109/TPAMI.2018.2858826

[22] M. Saeidi and A. Ahmadi, "Deep Learning Based on Parallel CNNs for Pedestrian Detection," *International Journal of Information & Communication Technology Research*, vol. 10, no. 4, 2018. [Online]. Available: http://ijict.itrc.ac.ir/article-1-410-en.html

[23] C.-Y. Wang, A. Bochkovskiy, and H. Liao, "Scaled-yolov4: Scaling cross stage partial network," *ArXiv*, vol. abs/2011.08036, 2020.

[24] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," *arXiv preprint arXiv:1612.08242*, 2016.

[25] ——, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[26] G. Jocher, "yolov5: v5.0 - yolov5-p6 1280 models, aws, supervise.ly and youtube integrations," 2020. [Online]. Available: https://zenodo.org/record/4679653#.YOSVzXDPzIV

[27] X. Long, K. Deng, G. Wang, Y. Zhang, Q. Dang, Y. Gao, H. Shen, J. Ren, S. Han, E. Ding, and S. Wen, "Pp-yolo: An effective and efficient implementation of object detector," 2020.

[28] A. Bochkovskiy, "Darknet. yolov4 / scaled-yolov4 / yolo - neural networks for object detection (windows and linux version of darknet)." [Online]. Available: https://github.com/AlexeyAB/darknet

[29] S. Charette, "Darkhelp. c++ wrapper library for darknet." [Online]. Available: https://www.ccoderun.ca/darkhelp/api/index.html