Leaf Detection Using YOLOv4 for Phytopathogenic Diagnosis

Alfredo Felipe Lopes Neto[1], Araken de Medeiros Santos[1], Silvio Fernandes[1]

[1]Programa de Pós-Graduação em Ciência da Computação (PPgCC), Departamento de Computação, Universidade Federal Rural do Semi-Árido (UFERSA)
Mossoró, RN – Brasil

{alfredo.lopes, araken, silvio}@ufersa.edu.br

Abstract. *Due to the impact that plant diseases cause on agricultural production, it is necessary to detect phytopathogens in an automated and efficient way to save time and resources. Thus, this article presents the first system module to aid in the detection of phytopathogens in plant leaves of six different species. This module aims to detect leaves and generate individual images of each leaf from the original image. The individual images generated will be used for individual classification of leaf diseases to then create a classification context in the next modules of the system. The detection module presented in this article was implemented using the YOLOv4 architecture. Results indicate accuracy of almost 90% and mAP of 91.27%, with equally important results in metrics, recall and IoU.*

Resumo. *Devido ao impacto que as doenças em plantas causam na produção agrícola, é necessário meios de detectar os fitopatógenos de forma automatizada e eficaz para economizar tempo e recursos. Assim, este artigo apresenta o primeiro módulo de sistema para auxiliar na detecção de fitopatógenos em folhas de plantas de seis espécies diferentes. Tal módulo objetiva detectar folhas e gerar imagens individuais de cada folha a partir da imagem original. As imagens individuais geradas serão usadas para classificação individual de doenças das folhas para em seguida criar um contexto de classificação nos próximos módulos do sistema. O módulo de detecção apresentado neste artigo foi implementado usando a arquitetura YOLOv4. Os resultados indicam precisão de quase 90% e mAP de 91,27%, com resultados igualmente importantes nas métricas, recall e IoU.*

1. Introdução

A agricultura 4.0 propõe a integração de tecnologias avançadas para desempenhar um papel importante no enfrentamento dos desafios agrícolas e o aperfeiçoamento na produção sustentável de alimentos. Essa integração inclui a agricultura de precisão, tendo como definição a tomada de decisões baseada em dados e o gerenciamento eficiente de recursos, visando o aumento da produtividade. De acordo com a Organização das Nações Unidas [UNEP 2021], os avanços no desenvolvimento da agricultura apresentam potencial para o aumento na produção de alimentos entre 75% e 85% até 2050, por meio da otimização na irrigação, no uso de fertilizantes e no controle de pragas. Desse modo, o monitoramento das lavouras se constitui parte crucial para o manejo de doenças que as

plantas podem vir a apresentar, garantindo assim a segurança alimentar em todo o processo.

Nesse contexto, considerando a realidade brasileira, a ocorrência de doenças em plantas representa desafios significativos para o aumento da produtividade agrícola. O Brasil, foi considerado em 2021, o 4°. maior produtor de grãos do mundo, tendo um percentual de aproximadamente 8,2% da produção mundial, perdendo apenas para Estados Unidos, China e Índia, segundo a Embrapa [Aragão and Contini 2022]. De acordo com [Zhang et al. 2022] doenças presentes na plantação podem afetar até 40% de toda produção, incidindo sobre grandes produtores, mas principalmente, os pequenos e médios produtores. Para combater esses desafios soluções baseadas em inteligência artificial são uma alternativa para a agricultura 4.0 uma vez que é produzido uma grande quantidade de informações em relação as plantações e as possíveis doenças que podem vir a acometê-las com o intuito de combater o avanço dessas doenças. Um dos mecanismos de inteligência artificial que pode processar um grande volume de informações para obter um resultado de qualidade são as redes neurais artificiais, possuindo arquiteturas e comportamentos inspirados no cérebro humano, baseando-se principalmente no funcionamento e nas conexões que os neurônios realizam, sendo capaz de, a partir de um conjunto de dados, aprender através de experiência, generalizando, assim, a informação e dando respostas coerentes para dados ainda não conhecidos [Kovács 2006].

Com as redes neurais artificiais é possível identificar padrões, dentre os dados coletados, e com esses padrões é possível utilizar técnicas para detecção de objetos em imagens. Essa técnica realiza um recorte do ambiente de interesse, busca identificar características especificas nessa imagem, aprender o padrão dessas características, para que ao final do processo sejam detectados esses mesmos padrões em outras imagens. Desse modo, sistemas para detecção de doenças em plantas surgem como um facilitador otimizando o gerenciamento no combate dessas pragas, permitindo o diagnóstico da doença [Ahmad et al. 2021].

Atualmente disposto dos mais variados modelos para detecção de objetos, dentre eles o YOLO [Redmon et al. 2015] que possui diversas versões. Nesse trabalho optou-se pela utilização da versão YOLOv4 [Bochkovskiy et al. 2020], por ser bem documentada, consolidada e amplamente reconhecida por suas aplicações em projetos das mais diversas áreas como, por exemplo, na contagem de pessoas [Degadwala et al. 2021; Yu and Zhang 2021]; na agricultura, para a identificação de flores e frutos [Wu et al. 2020; Yang et al. 2021]; e, para a detecção de animais, identificação de doenças e estimativa de biomassa [Abinaya et al. 2022; Zhao et al. 2022].

Destarte, o presente trabalho se propõe a construir uma ferramenta para identificação de folhas de variadas espécies de plantas, as quais são recortadas para posterior classificação individual de doenças e em seguida relacionadas para classificação do contexto da imagem do ambiente da imagem original. O sistema de detecção das folhas foi construído por meio de uma rede neural com arquitetura YOLOv4, treinada a partir da junção do *dataset Plant Village* apresentado por [Geetharamani and Pandian 2019] e do *dataset* apresentado por [Moupojou et al. 2023].

O artigo está organizado com a seguinte estrutura: A Seção 2 apresenta estudos que abordam os tópicos relacionados a este trabalho; a Seção 3, detalha a metodologia adotada, explicando a base de dados utilizada, o pré-processamento realizado, o modelo pré-treinado do YOLOv4 e as métricas de treinamento e validação. Na Seção 4 são

demonstrados os experimentos e resultados obtidos, seguidos da conclusão e proposta de trabalhos futuros na Seção 5.

2. Trabalhos Relacionados

Chairma e colegas (2023) propuseram utilizar o modelo YOLOv4 com um foco no desempenho otimizado para tarefas em tempo real, capturando imagens via a câmera do celular. Foram utilizados 45 quadros por segundo em uma resolução de 750x1334 pixels, apresentando uma precisão entre 84 e 94% na sua validação para uma diversidade de quatro espécies de plantas (milho, batata, laranja e tomate), enquanto outros modelos apresentam a proposta para somente uma espécie. Esse modelo foi treinado utilizando a base de dados *Plant Village* [Geetharamani, G and J 2019] e imagens adquiridas do ambiente em tempo real.

Em [Susa et al. 2022] construiu uma versão para detecção em tempo real focado em detectar doenças nas folhas do algodão utilizando o modelo YOLOv4. Nesse modelo a quantidade de camadas foi reduzida para 106 visando otimizar a inferência. A precisão do modelo para detecção em vídeo, armazenado em disco, está entre 98 e 99%, já em testes do modelo em tempo real, utilizando uma webcam conectada ao computador capturando a 30 quadros por segundo e resolução de 480x480 pixels, a precisão fica entre 74 e 99%, apresentando assim um desempenho melhor que outros trabalhos com a mesma proposta de redução de camadas.

No estudo desenvolvido por Nanehkaran e colegas (2020), foi criado um sistema baseado em LAB, algoritmo de aprendizado de autômato bio inspirado, que utiliza uma abordagem evolutiva no seu treinamento proposto por Pappa e Freitas (2010), para segmentar imagens que trazem folhas com sintomas de doenças e em seguida essas imagens são inseridas em uma rede neural convolucional para classificação. Esse sistema foi treinado utilizando uma base de dados composta de 1000 imagens, sendo elas folhas de arroz, pepino e milho. Apresentando um *mAP*, sendo a taxa média de todas as precisões do sistema, de 75,59%, esse valor foi aproximadamente 15,51% maior em comparação a outros estudos que utilizaram o algoritmo LAB.

Em [Agarwal et al. 2020], temos como proposta um modelo de rede neural convolucional ToLeD, apresentando na sua arquitetura 3 camadas de convolução e *pooling* máximo com número variável de filtros em cada camada. Comparado a outros modelos pré-treinados apresentou uma redução de 67 vezes no espaço de armazenamento, utilizando somente 1,5 MB, trazendo uma precisão de 91,2%, para a avaliação do modelo foi utilizado a base de dados *Plant Village*, no qual o modelo apresenta resultado superiores em relação aos outros.

Com propósito aplicado a agricultura 4.0, Ponnusamy (2020) incorpora o modelo YOLOv3 em uma Raspberry PI Zero W, a qual tem um modulo de câmera *CSI*, acoplada em óculos. O modelo YOLO em questão foi construído com modificações em algumas camadas e treinado utilizando *transfer learning* visando uma maior facilidade no treinamento para modificar qual espécie de planta com doença ele pode detectar. Esse sistema apresentou um tempo de inferência menor que 3 segundos, sendo esse um bom resultado em comparação a outros modelos de detecção em tempo real.

O estudo de [Moupojou et al. 2023] apresentou uma nova base de dados. Diferentemente de outras bases de dados que foram construídas utilizando imagens capturadas em condições de laboratório e fundo uniforme, essa base é composta por um

total de 5.170 imagens de plantas coletadas diretamente das plantações e realizado a anotação manual para cada imagem, observar-se isso na Figura 1 a representação da diferença na composição das bases com imagens capturadas em laboratório e as imagens da base em questão. Com estudos a partir dessa base foi possível identificar que uma base de dados composta por imagem de folhas coletadas diretamente das plantações permite uma maior generalização após o treinamento da rede neural, dado isso, esta foi a base de dados utilizada na estrutura do nosso trabalho, visando garantir uma maior generalização em nossos resultados.

Figura 1. Exemplo de plantas que apresentam doenças. (a) imagens capturadas em laboratório; (b) imagens capturadas nas plantações. Adaptada de [Moupojou et al. 2023]

Em uma perspectiva semelhante aos estudos citados, o trabalho ora apresentado pretende utilizar a base de dados *Plant Village* e adicionar imagens de plantas coletadas diretamente das plantações, para assim garantir uma maior generalização no treinamento da rede neural convolucional YOLOv4. Portanto, o presente trabalho se diferencia dos apresentados anteriormente nesta seção, pois cria, a partir de uma imagem de entrada, um novo *dataset* em tempo de execução por meio das folhas detectadas na imagem de entrada. Assim, essa rede isola o objeto de interesse de eventuais informações irrelevantes que possam atrapalhar o processo de classificação posterior de cada folha individualmente. Logo, o detector de folhas proposto nesse trabalho é o primeiro módulo de um sistema de classificação de contexto de uma imagem, para tanto este detector separa cada folha para servir de entrada para um classificador tradicional. Em seguida, o sistema condensará cada classificação individual em uma classificação de contexto em relação a saúde da planta, considerando a imagem original que entrou no detector.

3. Metodologia

3.1 Base de Dados

Neste trabalho foi utilizado um recorte da base de dados *Plant Village* [Geetharamani and Pandian 2019], por essa base apresentar em sua totalidade imagens de folhas capturadas em laboratório, decidiu-se utilizar somente 65% dela, a parte que atendia o critério de ter uma classe de folhas saudáveis e pelo menos duas classes com algum fitopatógeno. Em seguida foi adicionada a base de dados de [Moupojou et al. 2023] tentando garantir uma maior diversidade nas imagens para o treinamento. Dessa forma, na junção foi obtido um total de 39 classes, como pode ser verificado na Tabela 1.

Tabela 1. Distribuição de imagens da base de dados

Classes de folhas com doenças		
Nome da classe	Número de imagens	
	Sem *data augmentation*	Com *data augmentation*
Maçã com sarna	630	1000
Maçã com podridão	621	1000
Maçã com ferrugem	275	1000
Maçã saudável	1645	1645
Mirtilo saudável	1502	1502
Cereja com oídio	1052	1052
Cereja saudável	854	1000
Milho com mancha cinza	513	1000
Milho com ferrugem	1192	1192
Milho com ferrugem da folha do norte	985	1000
Milho saudável	1162	1162
Uva com podridão	1180	1180
Uva com sarampo preto	1383	1383
Uva com ferrugem	1076	1076
Uva saudável	423	1000
Laranja com hanglongbing	5507	5507
Pêssego com bactérias	2297	2297
Pêssego saudável	360	1000
Pimenta com bactérias	997	1000
Pimenta saudável	1478	1478
Batata com ferrugem	1000	1000
Batata saudável	152	1000
Batata com requeima	1000	1000
Tomate com bactérias	2127	2127
Tomate com ferrugem precoce	1000	1000
Tomate saudável	1591	1591
Tomate com ferrugem	1909	1909
Tomate com molde de folha	952	1000
Tomate com folha de septoria	1771	1771
Tomate com ácaro	1676	1676
Tomate com ponto alvo	1404	1404
Tomate com vírus do mosaico	373	1000
Tomate com folha amarela	5357	5357
Fundo sem folhas	1143	1143

Objetivando diminuir o desbalanceamento das classes, foram utilizadas técnicas de *data augmentation* para garantir uma quantidade mínima de 1000 imagens para as classes com quantitativo menor que este. Nesse processo, foram realizadas variações que poderiam ser encontradas em um ambiente de plantação durante uma coleta de imagem, como ruído, variação da luminosidade e variações no eixo horizontal e vertical (Figura 2).

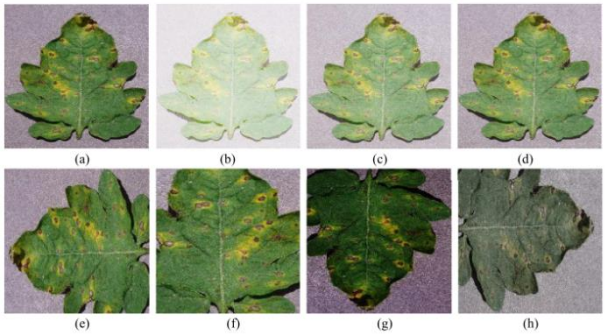


Figura 2. (a) Imagem original; (b) – (h) imagem com variação. Adaptada de (Geetharamani G, 2019)

Para utilizar a base de dados no treinamento de detecção do modelo YOLO é necessário que, cada imagem tenha um respectivo arquivo de texto contendo as informações em relação a posição de cada objeto na imagem que se espera utilizar no treinamento do modelo. Nesse arquivo temos uma descrição no formato: classe, posição central em relação ao eixo X, posição central em relação ao eixo Y, altura e largura. Na Figura 3 é apresentado um exemplo do formato YOLO e as informações do arquivo de notação, onde temos a classe do objeto (0 – Cubo e 1 - Cilindro), eixo X, eixo Y, largura e altura [Jocher et al. 2020]. Em nosso trabalho, as anotações foram realizadas manualmente a partir das imagens originais utilizando *Bounding Boxes* através da ferramenta *Make Sense* [Skalski 2019] que calcula e gera os descritivos de todas as anotações visuais, salvando as informações em um arquivo de texto.

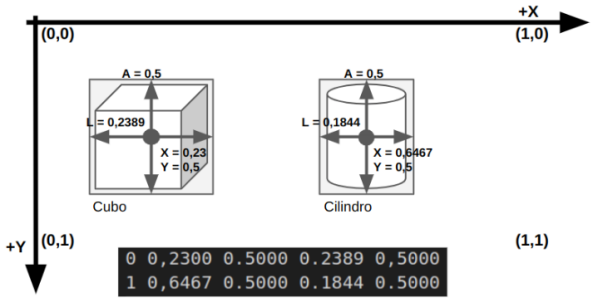


Figura 3. Descrição do mapeamento do objeto no formato YOLO.

A partir do *dataset* com as classes de folhas saudáveis e com alguma patogenia, o passo seguinte foi da criação de "supergrupos", os quais reúnem todas as imagens da mesma espécie de plantas, fundindo as classes de doença e saudáveis da mesma espécie de planta. Ao final deste processo, foi obtido um total de 6 novas classes a partir das 39 classes anteriores, conforme a Tabela 2. Em uma etapa futura deste trabalho, serão construídos classificadores de doença a partir das classes relacionadas a mesma espécie de planta, considerando as 39 classes originais.

Tabela 2. Descrição dos *supergrupos*.

Classe de folhas *super grupos*

Nome da classe	Número de imagens	Treino (70%)	Validação (30%)
Maça	8636	6045	2591
Pimentão	3357	2350	1007
Milho	6475	4532	1943
Batata	5402	3781	1621
Tomate	12877	9014	3863
Uva	3470	2429	1041

3.2 Modelo de Aprendizagem

O YOLOv4 pertence a uma família de arquiteturas, para detecção de objetos apresentado por Bochkovskiy e colegas (2020). Optou-se por essa versão por ser consolidada, amplamente reconhecida e documentada em diversos projetos, por exemplo o desenvolvido por Yu e Zhang (2021) e . Além disso, foi escolhido pela arquitetura de *backbone CSP-Darknet53*, uma rede neural desenvolvida em C juntamente com *Spatial Pyramid Pooling* (SPP) para otimizar o campo receptivo; *Path Aggregation Network*

(PAN) como método para a agregação de parâmetros; e o YOLOv3 com *head,* sendo responsável pela saída.

Com isso, para facilitar o treinamento da rede foi utilizado a técnica de *transfer learning* nas 137 primeiras camadas convolucionais iniciais. Incluindo 25 camadas treináveis específicas para a aplicação proposta.

A fim de obter um melhor desempenho no treinamento do modelo, foram utilizados os parâmetros listados na Tabela 3, baseados nos estudos de [Rahimzadeh et al. 2021].

Tabela 3. Parâmetros utilizados na configuração da rede

Parêmetro	Valor
Batch Size	64
Optimizer	adam
Loss Function	Categorical Crossentopy
Epochs	10000
Learning Rate	0.001

Durante o treinamento uma das métricas de avaliação, a IoU (do inglês *Intersection over Union*) realiza por meio da sobreposição a comparação que existe entre a marcação original esperada e a predita. Exemplificando na Equação (1), temos C_{PRED} e C_{REAL} como as características de cada marcação, sendo, os eixos X e Y, largura e altura (Zheng et al. 2020). Essas variações apresentam soluções específicas para diferentes problemas, melhorando a velocidade de convergência e a precisão da localização do objeto. Primeiro, GIoU (*Generalized* IoU) corrige o cálculo do gradiente quando a sobreposição entre as marcações não existe [Rezatofighi et al. 2019]. Em seguida, o DIoU (*Distance* IoU) atua para minimizar a distância entre os pontos centrais das marcações [Zheng et al. 2019]. As distinções entre os cálculos das funções podem ser observadas na Figura 4, onde o retângulo interno, em vermelho, representa a marcação ideal, definida na anotação da imagem, e o retângulo externo, em verde, representa a marcação real, obtida como retorno do modelo ao processar a imagem [Zheng et al. 2020].

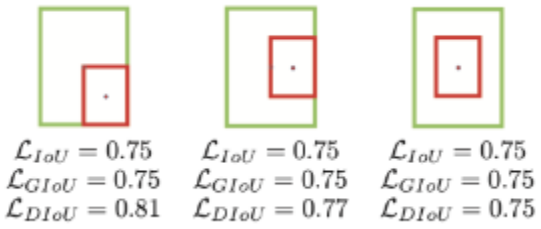


Figura 4. Comparação entre as métricas IoU, DIoU e GIoU.

Além da métrica DIoU, dada pela Equação (2), temos também *p* que representa a distância euclidiana entre os centros das formas A_{PRED} e A_{REAL}, e *c* corresponde ao comprimento da diagonal de menor forma que une as marcações. Para além disso, os possíveis fatores geométricos das imagens são tratados por dois novos parâmetros, sendo eles, *v* dada pela Equação (3), resultado da medida de consistência da proporção entre altura (*h*) e largura (*w*) e α, equivalente a Equação (4), usada como um parâmetro de balanceamento. Combinando esses parâmetros foi obtido o CIoU (*Complete* IoU), dado

pela Equação (5), o qual é utilizado como *loss function* padrão no YOLOv4 [Zheng et al. 2019].

$$IoU = \frac{|C_{PRED} \cap C_{REAL}|}{|C_{PRED} \cup C_{REAL}|} \quad (1)$$

$$DIoU = IoU + \frac{p^2(A_{PRED,}, A_{REAL})}{c^2} \quad (2)$$

$$v = \frac{4}{\pi^2} * (arctan\frac{w_{REAL}}{h_{REAL}} - arctan\frac{w_{PRED}}{h_{PRED}})^2 \quad (3)$$

$$\alpha = \frac{v}{\text{IoU}+v'} \quad (4)$$

$$CIoU = DIoU + \alpha v \quad (5)$$

Ademais, para avaliar os resultados em relação a detecção, temos as métricas como precisão, Equação (6), e o recall, Equação (7), que são calculadas com base nas detecções corretas de objetos presentes na imagem: TP (do inglês *True Positives*), detecções incorretas de objetos: FP (do inglês *False Positives*) e das detecções não identificadas de objetos: FN (do inglês *False Negatives*) (Zheng et al. 2020). Para o contexto de detecção de objetos quanto maior for a área sobreposta do objeto detectado melhor para a avaliação.

$$Precisão = \frac{TP}{TP+FP} \quad (6)$$

$$Recall = \frac{TP}{TP+FN} \quad (7)$$

4. Resultados

Para a realização deste trabalho foi decidido avaliar o modelo de duas formas, utilizando um detector geral e detectores individuais por classe. Ambos os modelos utilizam a mesma arquitetura apresentada na seção anterior.

Inicialmente, utilizando todo o *dataset* para as seis classes sendo denominado de "detector geral", tendo como objetivo tirar do usuário a responsabilidade de escolher uma classe, levando o modelo a realizar uma escolha de maneira automática, permitindo certa liberdade de análise desde que seja uma das seis classes previstas pelo modelo.

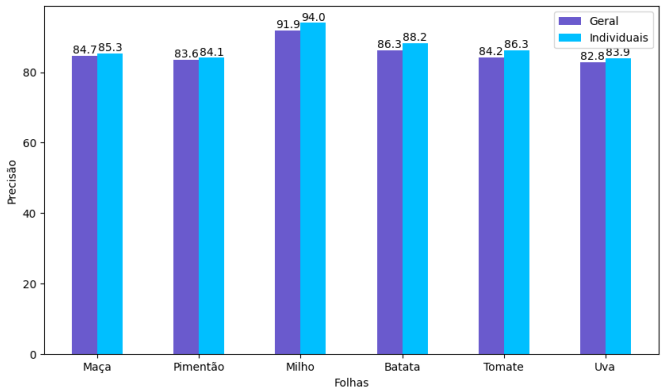
Em seguida, foi construído seis detectores individuais, um para cada classe de folhas, utilizando apenas a base de dados de cada classe correspondente, esses detectores têm como objetivo melhorar o desempenho, com a desvantagem de fazer com que o usuário escolha previamente qual a classe deseja detectar.

Esses dois grupos de detectores foram avaliados de maneira isonômica, visando obter a comparação entre eles. Realizando inferências nas imagens de 1024x1024 pixels

para cada grupo, focando em três métricas para comparação, sendo elas, precisão, *recall* e *IoU*.

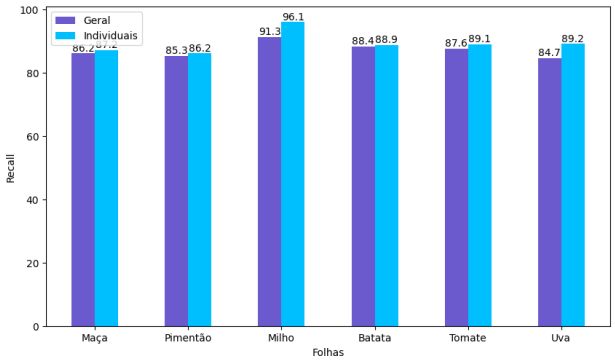
A Figura 5 apresenta a comparação da métrica de precisão, que consiste na quantidade de vezes que o modelo acerta em relação ao total de vezes que ele tenta acertar. Os detectores de únicas classes (detectores individuais) se sobressaem, como esperado, uma vez que sendo treinados com apenas uma única classe, tendem a identificar as características mais específicas da sua classe. Os detectores individuais são ligeiramente mais precisos, no entanto o detector geral manteve sua precisão acima de 82% para todas as classes, de modo que também se mostra eficiente para essa tarefa.

Figura 5. Comparação entre os detectores individuais e o detector geral em relação a precisão, tendo como base a média de 100 testes.



A Figura 6 apresenta a comparação da métrica de *recall*, que é a quantidade de vezes que o modelo acerta em relação ao total de vezes que ele deveria acertar. Verifica-se que os detectores individuais também apresentam melhores resultados em relação ao detector geral, assim, essa informação é relevante para entender a sensibilidade do modelo em detectar com sucesso resultados considerados verdadeiros positivos. Destaca-se que em ambos os detectores, o índice de acerto está acima de 84% para todas as classes.

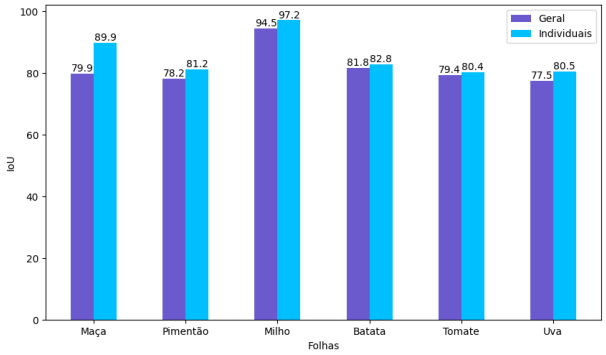
Figura 6. Comparação entre os detectores individuais e o detector geral em relação ao recall, tendo como base a média de 100 testes.



A Figura 7 apresenta a comparação da métrica *IoU* com os melhores resultados dos detectores individuais. Dessa forma, os detectores individuais, que são especializados

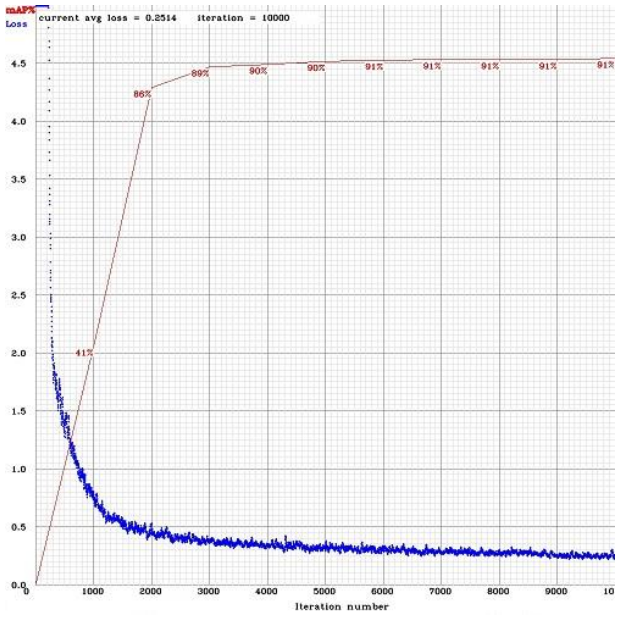
em um único tipo de planta, conseguem localizar melhor a região dos objetos detectados. Diante dos dados apresentados, pode-se concluir que o modelo YOLOv4 treinado (de forma) como detector individual, com resolução de entrada 1024x1024, apresentou melhores resultados em relação ao detector geral, à vista disso é a opção mais adequada para a apresentação junto ao usuário.

Figura 7. Comparação entre os detectores individuais e o detector geral em relação ao IoU, tendo como base a média de 100 testes.



O detector geral teve uma média de precisão de 85,6% com desvio padrão de 0,12 enquanto os detectores individuais uma média 87% precisão com desvio padrão de 0,14. A média do *recall* para os detectores geral e individuais foram, respectivamente, 87,2% como desvio padrão de 0,10 e 89,45% e desvio padrão de 0,13. Para a métrica de *IoU* as médias foram de 81,8% com desvio padrão 0,23 (geral) e 85% com desvio padrão de 0,25 (individuais). Além disso, para os detectores individuais foi obtido em média mAP de 91,27% utilizando a configuração de *threshold* com valor de 0.75, como podemos observar na Figura 8.

Figura 8. Representação *mAP* ao longo de 10000 interações



Esses dados demonstram que para detecções mais precisas podem ser utilizados os modelos individuais, o que significa que o utilizador deverá indicar a priori qual o tipo de planta para selecionar o detector adequado. No entanto, para detecções mais automatizadas, inclusive com diferentes espécies de plantas na mesma imagem pode-se utilizar o detector geral, o qual possui uma pequena diminuição na qualidade em comparação aos individuais.

5. Conclusão e Trabalhos Futuros

O presente trabalho visou demonstrar uma metodologia para a criação de um detector de folhas de plantas utilizando uma arquitetura YOLOv4 e uma base de dados customizada fazendo uso de técnicas de *data augmentation* e anotação manual dos objetos de interesse, apresentando bons resultados nas métricas de avaliação.

Isso posto, conclui-se que o modelo proposto atingiu o propósito esperado de detectar objetos (folhas) em uma imagem, armazenando as coordenadas dos objetos detectados pelo sistema. Tais coordenadas são usadas pelo sistema proposto para gerar imagens individuais de cada objeto detectado, "recordando-os" da imagem original. Tal funcionalidade separa os objetos de interesse de informações desnecessárias para classificação de tais objetos em uma etapa futura do sistema. Foi implementado um detector geral para seis classes de plantas e seis detectores individuais, uma para cada classe. A avaliação dos detectores foi realizada por meio das métricas de precisão, *recall*, IoU e *mAP*. Nesta avaliação foi constatado uma boa eficiência para o modelo implementado sobre o respectivo *dataset*, sobretudo para os detectores individuais de cada classe, que ficaram em geral 2 pontos percentuais acima do detector geral, sendo que, uma imagem apresentando um IoU maior ou igual a 80% se apresenta com uma boa amostra para o treinamento do classificador e um *mAP* de 91,27%. Como o propósito dos detectores é ser parte de uma classificação de contexto, não há grande prejuízo em não se detectar alguns objetos, visto que a maioria dos casos em que isso ocorre é quando o objeto está por trás de outro objeto, e em alguns casos são outras folhas detectadas. Com uma relação de custo-benefício na qual pode-se optar por maior qualidade da detecção para usuário com conhecimento das espécies de plantas ou ligeira diminuição na qualidade para detecção automática pelo sistema, podendo haver múltiplas espécies na mesma imagem.

Contudo, verifica-se algumas limitações do trabalho, que nos conduzem a novas perspectivas para os trabalhos futuros. Dentre elas, as especificidades do *dataset* utilizados para o treinamento e a avaliação dos modelos, devido a insuficiência referente a diversidade de folhas e balanceamento delas. Assim, é valido realizar a construção de um *dataset* próprio para não necessitar de *data augmentation*, preferencialmente, utilizando folhas da região.

Em suma, esse trabalho ao apresentar uma precisão de quase 90% demonstra de forma satisfatória alcançar o seu objetivo, a construção de um detector, sendo a primeira parte de um sistema maior, o qual utilizará as coordenadas das folhas detectadas recortá-las da imagem original. Cada folha separada será usada por um modelo de classificação individual. Esse classificador irá definir se a folha em questão apresenta ou não algum fitopatógeno. Ao final, o sistema utilizará a classificação individual das imagens separadas para realizar classificação de contexto, ou seja, determinando se a planta que

originou as imagens inicialmente apresenta mais folhas adjacentes doentes ou saudáveis determinando o grau de doença presente no contexto da imagem.

6. Agradecimentos

Os autores agradecem a UFERSA, ao PPgCC (Programa de Pós-Graduação em Ciência da Computação) e ao projeto Implementação de Ferramentas Tecnológicas na Produção de Alimentos no Semiárido Potiguar, do edital nº 18/2020 – CAPES.

7. Referências

Abinaya, N. S., Susan, D. and Sidharthan, R. K. (jun 2022). Deep learning-based segmental analysis of fish for biomass estimation in an occulted environment. *Computers and Electronics in Agriculture*, v. 197, p. 106985.

Agarwal, M., Singh, A., Arjaria, S., Sinha, A. and Gupta, S. (2020). ToLeD: Tomato Leaf Disease Detection using Convolution Neural Network. *Procedia Computer Science*, v. 167, p. 293–301.

Ahmad, M., Abdullah, M., Moon, H. and Han, D. (2021). Plant Disease Detection in Imbalanced Datasets Using Efficient Convolutional Neural Networks With Stepwise Transfer Learning. *IEEE Access*, v. 9, p. 140565–140580.

Aragão, A. and Contini, E. (2022). O agro no Brasil e no Mundo: uma síntese do período de 2000 a 2021. https://www.embrapa.br/documents/10180/26187851/O+agro+no+Brasil+e+no+mundo/098fc6c1-a4b4-7150-fad7-aaa026c94a40, [accessed on Jun 19].

Bochkovskiy, A., Wang, C.-Y. and Liao, H.-Y. M. (22 apr 2020). YOLOv4: Optimal Speed and Accuracy of Object Detection.

Chairma, L., B, P., G, S., et al. (feb 2023). Yolo for Detecting Plant Diseases. In *2023 Third International Conference on Artificial Intelligence and Smart Energy (ICAIS)*. . IEEE. https://ieeexplore.ieee.org/document/10073875/.

Degadwala, S., Vyas, D., Chakraborty, U., Dider, A. R. and Biswas, H. (mar 2021). Yolo-v4 Deep Learning Model for Medical Face Mask Detection. In *2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS)*. . IEEE.

Geetharamani, G. and J, A. P. (jun 2019). Identification of plant leaf diseases using a nine-layer deep convolutional neural network. *Computers & Electrical Engineering*, v. 76, p. 323–338.

Geetharamani, G. and Pandian, A. (2019). Identification of plant leaf diseases using a nine-layer deep convolutional neural network. *Computers & Electrical Engineering*, v. 76, p. 323–338.

Jocher, G., Nishimura, K., Mineeva, T. and Vilariño, R. (2020). yolov5. *Code repository*,

Kovács, Z. L. (2006). *Redes neurais artificiais*. 4. ed. São Paulo: Editora Livraria da Física.

Moupojou, E., Tagne, A., Retraint, F., et al. (2023). FieldPlant: A Dataset of Field Plant Images for Plant Disease Detection and Classification With Deep Learning. *IEEE Access*, v. 11, p. 35398–35410.

Nanehkaran, Y. A., Zhang, D., Chen, J., Tian, Y. and Al-Nabhan, N. (4 sep 2020). Recognition of plant leaf diseases based on computer vision. *Journal of Ambient Intelligence and Humanized Computing*,

Pappa, G. L. and Freitas, A. (2010). *Automating the Design of Data Mining Algorithms*. Berlin, Heidelberg: Springer Berlin Heidelberg.

Ponnusamy, V., Coumaran, A., Shunmugam, A. S., Rajaram, K. and Senthilvelavan, S. (jul 2020). Smart Glass: Real-Time Leaf Disease Detection using YOLO Transfer Learning. In *2020 International Conference on Communication and Signal Processing (ICCSP)*. . IEEE.

Rahimzadeh, M., Attar, A. and Sakhaei, S. M. (jul 2021). A fully automated deep learning-based network for detecting COVID-19 from a new and large lung CT scan dataset. *Biomedical Signal Processing and Control*, v. 68, p. 102588.

Redmon, J., Divvala, S., Girshick, R. and Farhadi, A. (8 jun 2015). You Only Look Once: Unified, Real-Time Object Detection.

Rezatofighi, H., Tsoi, N., Gwak, J., et al. (jun 2019). Generalized Intersection Over Union: A Metric and a Loss for Bounding Box Regression. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. . IEEE.

Skalski, P. (2019). Make Sense.

Susa, J. A. B., Nombrefia, W. C., Abustan, A. S., Macalisang, J. and Maaliw, R. R. (28 apr 2022). Deep Learning Technique Detection for Cotton and Leaf Classification Using the YOLO Algorithm. In *2022 International Conference on Smart Information Systems and Technologies (SIST)*. . IEEE.

UNEP (2021). How to Feed the World in 2050. http://www.fao.org/fileadmin/templates/wsfs/docs/expert_paper/How_to_Feed_the_World_in_2050.pdf., [accessed on Jun 19].

Wu, D., Lv, S., Jiang, M. and Song, H. (nov 2020). Using channel pruning-based YOLO v4 deep learning algorithm for the real-time and accurate detection of apple flowers in natural environments. *Computers and Electronics in Agriculture*, v. 178, p. 105742.

Yang, B., Gao, Z., Gao, Y. and Zhu, Y. (12 jun 2021). Rapid Detection and Counting of Wheat Ears in the Field Using YOLOv4 with Attention Module. *Agronomy*, v. 11, n. 6, p. 1202.

Yu, J. and Zhang, W. (8 may 2021). Face Mask Wearing Detection Algorithm Based on Improved YOLO-v4. *Sensors*, v. 21, n. 9, p. 3263.

Zhang, Q., Men, X., Hui, C., Ge, F. and Ouyang, F. (mar 2022). Wheat yield losses from pests and pathogens in China. *Agriculture, Ecosystems & Environment*, v. 326, p. 107821.

Zhao, S., Zhang, S., Lu, J., et al. (jul 2022). A lightweight dead fish detection method based on deformable convolution and YOLOV4. *Computers and Electronics in Agriculture*, v. 198, p. 107098.

Zheng, Z., Wang, P., Liu, W., et al. (19 nov 2019). Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression.

Zheng, Z., Wang, P., Liu, W., et al. (3 apr 2020). Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression. *Proceedings of the AAAI Conference on Artificial Intelligence*, v. 34, n. 07, p. 12993–13000.