USO DA YOLOV8 NA IDENTIFICAÇÃO DAS DOENÇAS MANCHA-ALVO E MANCHA OLHO-DE-RÃ NA CULTURA DA SOJA.

Carlos Pinheiro Ferreira Junior

Mario Lúcio Gomes de Queiroz Pierre Junior

Morgana Mateus Santos

RESUMO

O Brasil é o principal produtor e exportador de soja do mundo, estando à frente de países como Estados Unidos e Argentina. Problemas com pragas e doenças são recorrentes, principalmente devido ao clima quente e úmido que prevalece na maior parte das áreas de produção da soja, o que leva a uma diminuição na produtividade agrícola. Entre essas doenças estão presentes a mancha-alvo e a mancha olho-de-rã causadas pelos fungos, *Corynespora cassiicola* e *Cercospora sojina,* respectivamente. Os sintomas conhecidos de ambas doenças aparecem principalmente nas folhas e ocasionam lesões necróticas, reduzindo a área foliar disponível para a fotossíntese, comprometendo a produção de energia necessária para o crescimento e desenvolvimento das plantas. A principal forma de combate é a aplicação de fungicidas, que deve ser feita o mais precoce possível assim que identificadas. Esta pesquisa teve como objetivo o uso do modelo *You Only Look Once* (YOLO), versão 8, na identificação dessas doenças através de imagens digitalizadas das folhas da soja. Para isso, foi necessário criar um banco de imagens com ambas enfermidades e analisar os resultados obtidos após o treinamento e validação. As validações realizadas durante o treinamento mostraram que mesmo na sua oitava versão, o YOLO ainda tem dificuldade em identificar objetos pequenos, apresentando métricas razoáveis como 54.6% para o mAP50:90, 85% para F1-score durante o treinamento. Resultou em 54.4% para mAP50:95, 84.6% para F1-score na validação cruzada, e, por fim, a avaliação final utilizando um conjunto de testes, que não havia sido utilizado no treinamento, resultou em 54.6% para mAP50:95 e 84.9% para F1-*score*. Apesar de suas limitações em identificar objetos pequenos, o modelo apresentado trouxe métricas consideráveis.

PALAVRAS-CHAVE: detecção de doenças; mancha-alvo; mancha olho-de-rã; YOLOv8.

ABSTRACT

Brazil is the main producer and exporter of soybeans in the world, ahead of countries such as the United States and Argentina. Problems with pests and diseases are recurrent, mainly due to the hot and humid climate that prevails in most soybean production areas, which leads to a decrease in agricultural productivity. Among these diseases are target spot and frogeye spot caused by the fungi, *Corynespora cassiicola* and *Cercospora sojina*, respectively. The known symptoms of both diseases appear mainly on the leaves and cause necrotic lesions, reducing the leaf area available for photosynthesis, compromising the production of energy necessary for plant growth and development. The main form of combat is the application of fungicides, which must be done as early as possible as soon as they are identified. This research aimed to use the You Only Look Once (YOLO) model, version 8, to identify these diseases through digitized images of soybean leaves. To do this, it was necessary to create an image bank with both diseases and analyze the results obtained after training and validation. Validations carried out during training showed that even in its eighth version, YOLO still has difficulty identifying small objects, presenting reasonable metrics such as 54.6% for mAP50:90, 85% for F1-score during training. It resulted in 54.4% for mAP50:95, 84.6% for F1-score for cross validation, and, the final evaluation using a set of tests, which had not been used in training, resulted in 54.6% for mAP50:95 and 84.9% for F1 -score. Despite its limitations in identifying small objects, the presented model brought considerable metrics.

KEYWORDS: disease detection; target spot; frogeye spot; YOLOv8.

1 INTRODUÇÃO

Os avanços tecnológicos e científicos trouxeram significativas contribuições em relação ao desenvolvimento agrícola, assim, aumentando em escala exponencial a produção de alimentos. O Brasil é um dos principais produtores e exportadores agrícolas, sendo que “as exportações brasileiras do agronegócio bateram recorde em 2023, atingindo US$ 166,55 bilhões. A cifra foi 4,8% superior em comparação a 2022, o que representa um aumento de US$ 7,68 bilhões.” (Brasil, 2024).

Mesmo produzindo variados grãos e frutos, o país é o principal *player* internacional quando se trata da soja, estando à frente de países como Estados Unidos e Argentina. A ascensão rápida do Brasil como um dos principais fornecedores globais de *commodities* é evidenciada pela destacada posição que o país ocupa na produção e exportação de soja, liderando esse mercado mundialmente. (Valdes; Gillespie; Dohlman, 2023).

No entanto, a produção de soja no país sofre com algumas perturbações. Doenças como a mancha-alvo, causada pelo fungo *Corynespora cassiicola*, emergem como fatores-chave na diminuição da produtividade da soja no Brasil, podendo resultar em uma queda de até 40% da produção. A mancha olho-de-rã, causada pelo fungo *Cercospora sojina,* é também um outro agravante do problema, que reduz a produtividade podendo causar perdas de 31% a 60%. (Barro et al., 2023)

A melhor forma de combater esses fungos é através da aplicação de fungicidas que constitui uma das principais estratégias de controle da mancha-alvo (Pivotto, 2023, p. 13). Para isso, o acompanhamento constante da cultura é o ideal, pois quanto mais cedo diagnosticados os problemas, mais eficaz será a aplicação do fungicida. Um dos aspectos que mais prejudicam a eficiência de fungicidas para o controle da mancha-alvo tem sido o atraso na primeira aplicação (Tormen; Blum; Balardin, 2017, p. 26).

Os progressos recentes no contexto computacional, principalmente no campo da inteligência artificial e visão computacional, propiciaram a viabilização de estudos que empregam essas tecnologias no reconhecimento pragas e doenças na agricultura. Essas inovações têm o potencial de facilitar o diagnóstico, aumentar a precisão e diminuir os gastos com pesticidas (Bevers; Sikora; Hardy, 2022).

Torna-se essencial o desenvolvimento de métodos que facilitem a identificação precoce dos sintomas de pragas e doenças nas culturas de soja. A eficácia dessas abordagens

reside na capacidade de permitir uma aplicação adequada de fungicidas, assegurando não apenas a eficiência no controle desses agentes patogênicos, mas também a otimização dos recursos agrícolas.

O presente trabalho está estruturado da seguinte maneira: os objetivos da pesquisa; o referencial teórico, onde serão apresentados os conhecimentos essenciais para a compreensão do estudo, incluindo as doenças abordadas, o funcionamento das redes neurais e do aprendizado profundo, o YOLO, e as métricas de avaliação utilizadas para medir o desempenho do modelo treinado; também serão discutidos trabalhos relacionados, destacando estudos que utilizaram o YOLO para a detecção de doenças e pragas em diversas culturas; em seguida, a metodologia detalha todas as etapas desenvolvidas durante a pesquisa, desde a construção do banco de imagens até o treinamento e validação do modelo. Na seção de resultados, são apresentadas as métricas obtidas após todas as etapas do estudo. Finalmente, a conclusão resume os achados principais, e são sugeridas direções para trabalhos futuros.

1.1 Justificativa

Atualmente existem vários modelos de visão computacional que podem ser utilizados para diferentes tarefas, entres eles, se tornou bastante popular o YOLO, que no presente momento está na sua oitava versão desenvolvida pela empresa *Ultralytics,* oferecendo uma abordagem inovadora que pode ser utilizada para a identificação de doenças causadas por fungos na folha de soja, como a mancha-alvo e a mancha olho-de-rã. Ao treinar o modelo com um conjunto diversificado de imagens de plantas infectadas, ele pode aprender a reconhecer padrões visuais específicos associados a essas doenças.

1.2 Problema de pesquisa

Nesse contexto, como o modelo da YOLOv8 pode ser empregado na identificação dos sintomas causados por fungos na folha de soja, incluindo a mancha-alvo e a mancha olho-de-rã, contribuindo assim para aprimorar as estratégias de diagnóstico e manejo fitossanitário nessa cultura agrícola?

2 OBJETIVOS

2.1 Objetivo geral

Identificação das doenças mancha-alvo e mancha olho-de-rã na cultura da soja utilizando o modelo da YOLOv8 para aprimorar as estratégias de diagnóstico e manejo fitossanitário.

2.2 Objetivos específicos

- Construção do banco de imagens (*dataset);*
- treinamento da YOLOv8;
- avaliação dos resultados obtidos.

3 REFERENCIAL TEÓRICO

Esta seção tem como objetivo apresentar os principais conceitos relacionados a este trabalho. Assim, serão apresentadas as duas doenças, mancha-alvo e mancha olho-de-rã, acompanhadas de seus sintomas causadores; técnicas e ferramentas computacionais que foram aplicados para a detecção, bem como trabalhos correlatos a esta pesquisa.

3.1 Mancha-alvo (*Corynespora Cassiicola*)

A mancha-alvo, doença causada pelo fungo *Corynespora Cassiicola*, é um mal que atinge mais de 400 espécies de plantas, incluindo culturas como, mamão, feijão, tomate, etc. Também é um problema que assola o cultivo da soja, sendo uma doença que se favorece com a boa distribuição de chuvas e aumento da umidade, presente em praticamente todas as regiões onde a soja é cultivada no Brasil. Parece ser endêmico e pode infectar uma ampla variedade de plantas, tanto nativas quanto cultivadas. Sua sobrevivência ocorre em restos de cultura e sementes infectadas (Godoy et al., 2023).

Na figura 1 são mostrados os sintomas predominantes da mancha-alvo incluem a formação inicial de lesões pardacentas, circundadas por um halo amarelado, que progressivamente formam manchas circulares nas folhas, a doença pode atingir a lavoura em qualquer momento e pode afetar as diferentes partes da planta, como as folhas, hastes, raízes e vagens. Dependendo da reação, as lesões podem expandir-se alcançando até 2 cm de diâmetro ou permanecer em tamanho reduzido, oscilando entre 1 mm e 3 mm, mas em maior

quantidade. Tipicamente, as manchas exibem uma pontuação central, acompanhada de anéis concêntricos de coloração mais intensa. (Godoy et al., 2023).

Figura 1- Mancha-alvo na folha da soja

Fonte: https://doi.org/10.5061/dryad.41ns1rnj3

3.2 Mancha olho-de-rã (*Cercospora Sojina*)

A segunda doença a ser identificada é a mancha olho-de-rã, que é causada pelo fungo *Cercospora Sojina,* é possível observar uma imagem da doença na figura 2. A enfermidade é favorecida por condições de alta umidade e temperatura, e pode surgir em qualquer fase do desenvolvimento da planta, mas, é mais frequente a partir do florescimento. Afeta folhas, hastes, vagens e sementes, manifestando-se inicialmente como pequenas áreas encharcadas (anasarca), que posteriormente se transformam em manchas com centros castanho-claros na página superior das folhas e cinzentos na página inferior, com bordas castanho avermelhadas em ambas as páginas. (Henning et al., 2014).

O fungo se dissemina através de sementes infectadas e esporos transportados pelo vento, sendo capaz de sobreviver em resíduos de plantas. As condições favoráveis para a doença incluem alta umidade e temperatura. Além disso, o patógeno tem a capacidade de gerar novas variantes (Henning et al., 2014).

De acordo com as recomendações, a adoção de cultivares resistentes em parceria com o tratamento de sementes utilizando fungicidas sistêmicos como os benzimidazois combinados com fungicidas de contato são medidas essenciais para o controle da doença e para prevenir a introdução do fungo ou o surgimento de novas variantes (Henning et al., 2014).

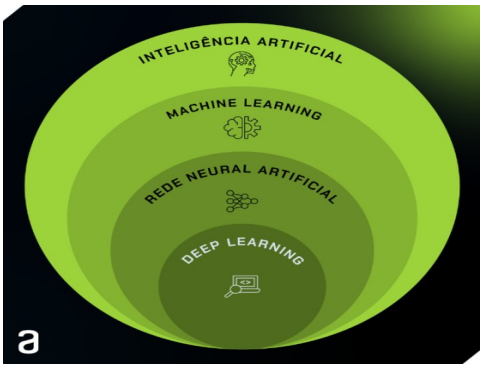
Figura 2 - Mancha olho-de-rã na folha da soja.

Fonte: https://doi.org/10.5061/dryad.41ns1rnj3

3.3 Redes neurais e aprendizado profundo

As redes neurais artificiais, ou redes neurais (RNs) como são popularmente conhecidas, é um subconjunto da área de inteligência artificial e aprendizado de máquina, que modela o funcionamento cerebral para executar tarefas específicas, utilizando componentes eletrônicos ou simulação por software em computadores (Haykin, 2009, p.2).

Figura 3 - Rede neural artificial e *deep learning* como subcampo da IA.

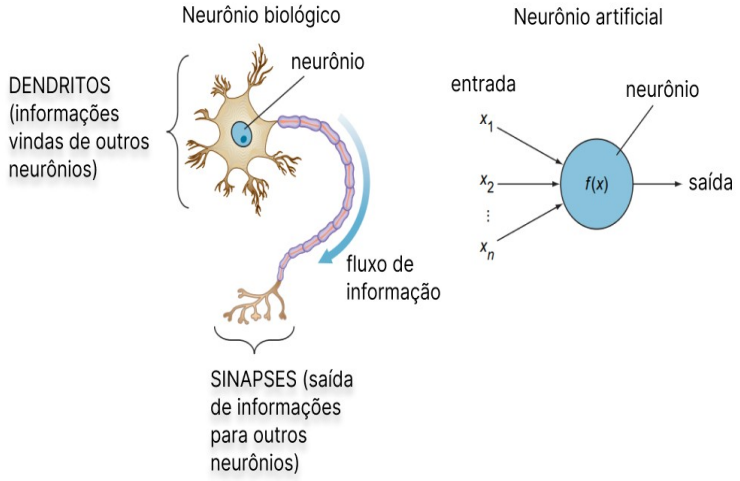


Fonte: https://www.alura.com.br/artigos/assets/machine-learning/grafico.png

Para que as RNs possam simular o funcionamento do cérebro, elas utilizam o *perceptron,* que se trata da unidade fundamental para o funcionamento de uma rede neural (Haykin, 2009). O *perceptron* opera de forma análoga a um neurônio biológico, que recebe sinais elétricos, os processa e emite um sinal de saída apenas quando a força total dos sinais de entrada ultrapassa um limite específico. Para replicar esse comportamento, o neurônio artificial calcula a soma ponderada das entradas, representando a intensidade total dos sinais,

e aplica uma função de ativação para determinar se a saída deve ser ativada (1) caso o sinal ultrapasse o limite ou desativada (0) caso contrário (Elgendy, 2020).

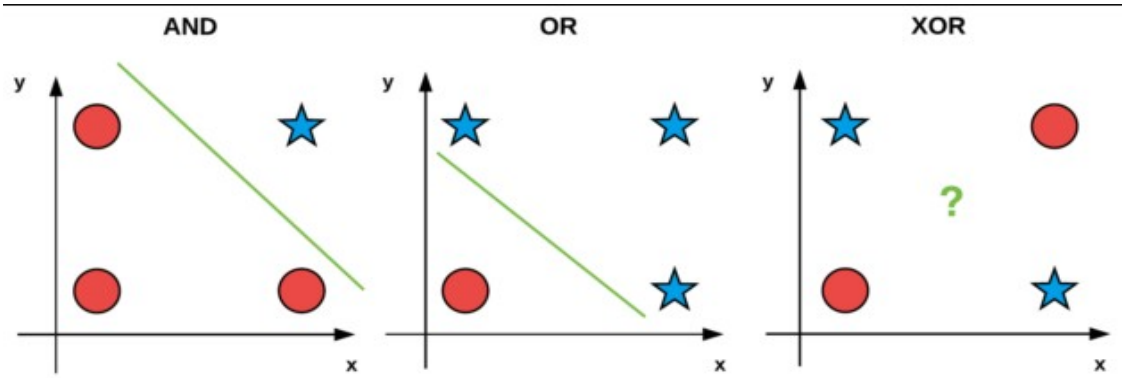
Figura 4 - Neurônios artificiais foram inspirados em neurônios biológicos.



Fonte: Elgendy, 2020, p. 9

Apesar dos *perceptrons* servirem como base para o funcionamento de uma rede neural, um neurônio sozinho não é capaz de realizar muita coisa, o *perceptron* pode ser entendido como uma representação linear, o que implica que, após o treinamento, o neurônio será capaz de traçar uma linha reta que divide os dados em duas categorias distintas (Elgendy, 2020, p. 43). Um único perceptron é capaz de resolver apenas problemas com separabilidade linear, como os que ocorrem nos casos *AND e OR* da figura 5, no qual há uma separação entre os círculos vermelhos e a estrela azul, indicando que apenas um neurônio é o suficiente para realizar essa separação. Sendo que o mesmo já não é possível com o problema XOR presente na mesma figura, reserve um segundo para se convencer de que é impossível desenhar uma única linha que separa as estrelas azuis dos círculos vermelhos (Rosebrock, 2017).

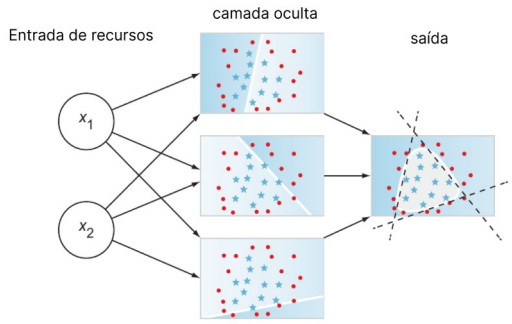
Figura 5 - Separação linear que ocorre utilizando apenas um perceptron (neurônio artificial).



Fonte: https://pyimagesearch.com/2021/05/06/implementing-the-perceptron-neural-network-with-python/

Para que as redes neurais possam ser utilizadas em problemas mais complexos, é necessário utilizar mais camadas de neurônios, surgindo assim o termo “aprendizado profundo”, que advém justamente da construção de múltiplas camadas neurais interconectadas, isso significa que precisamos criar uma arquitetura para usar dezenas e centenas de neurônios em nossa rede neural. (Elgendy, 2020, p. 46, tradução nossa).

Figura 6 - Utilizando 3 neurônios para separar os círculos vermelhos das estrelas azuis.

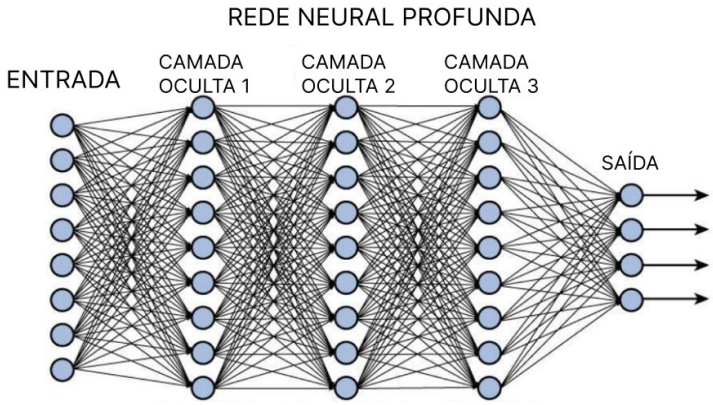


Fonte: Elgendy, 2020.

É nas camadas ocultas que a tudo acontece, é onde está o cerne do processo de aprendizado de características: as primeiras camadas detectam padrões simples, como linhas retas, enquanto as posteriores identificam padrões mais complexas dentro dessas especificações. Em redes neurais, adicionamos camadas ocultas para aprender características complexas através das camadas até ajustar nossos dados. Então, se sua rede neural não estiver se adaptando bem aos dados, talvez seja necessário adicionar mais camadas ocultas (Elgendy, 2020).

A camada oculta recebe esse nome porque não vemos ou controlamos a entrada que vai para essas camadas ou para a saída. Tudo o que fazemos é alimentar o vetor de recursos para a camada de entrada e ver a saída da camada final. (Elgendy, 2020, p. 46, tradução nossa). Essa ideia causa a sensação de que as redes neurais são implacáveis, e que basta sair adicionando camadas infinitamente e você terá a solução de todos os problemas, porém, quanto mais camadas, mais recursos computacionais são necessários, além de que, o excesso de profundidade pode levar ao *overfitting*, onde a rede se ajusta demais aos dados de treinamento e não generaliza bem com novos dados (Elgendy, 2020, p. 125).

Figura 7 - Rede neural profunda com 3 camadas ocultas.



Fonte: https://jmvstream.com/wp-content/uploads/2023/07/Deep-Learning.jpeg

3.4 *You only look once* (YOLO)

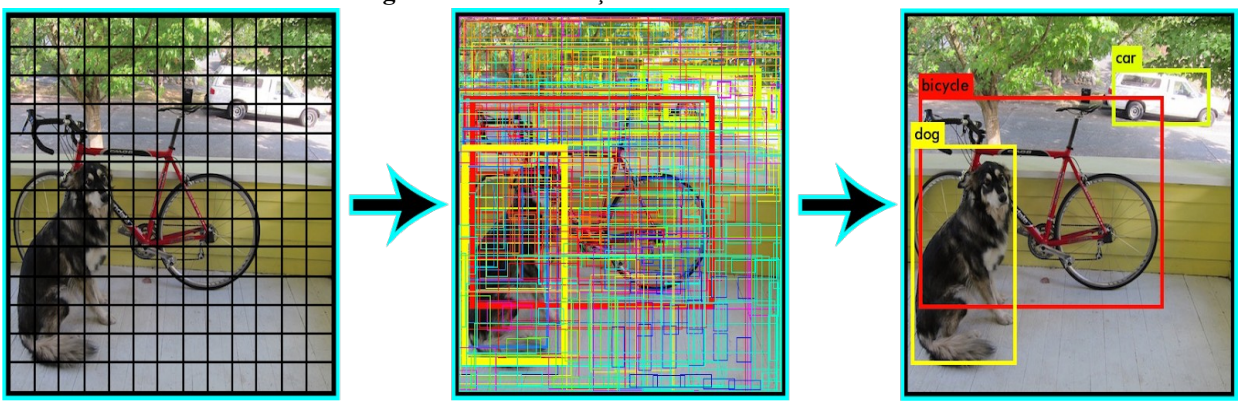
O YOLO é um modelo de detecção em tempo real, desenvolvido por Joseph Redmon e Ali Farhadi em 2015, que ficou muito popular devido sua eficiência e acurácia, obtendo uma precisão igual ou superior a outros modelos concorrentes (Alves, 2020).

O YOLO recebeu esse nome devido a sua característica de passada única (*single pass)*, no qual, diferente dos outros modelo de detecção em imagem, percorre a imagem uma única vez, dessa forma ele acaba se tornando muito mais ágil que os outros modelos (Redmon, 2020).

O funcionamento do YOLO é baseado na divisão da imagem em uma grade de células, geralmente com dimensões SxS. Na versão mais recente durante o desenvolvimento dessa pesquisa, o YOLOv8, por exemplo, essa grade tem 13 linhas por 13 colunas. Cada célula da grade é encarregada de prever 5 caixas delimitadoras, que são retângulos que definem a área onde um objeto pode estar presente na imagem. Essas caixas delimitadoras são acompanhadas por probabilidades de classe, indicando a chance de cada caixa conter um objeto de uma classe específica. Essa abordagem permite uma detecção eficiente e precisa de múltiplos objetos em tempo real (Aggarwal, 2020).

Além disso, o YOLO utiliza uma única arquitetura de rede neural para realizar a detecção e classificação dos objetos, sendo uma abordagem de ponta a ponta, o que o torna mais rápido e eficiente em comparação com métodos tradicionais que exigem múltiplas etapas de processamento.

Figura 8 - Demonstração do funcionamento do YOLO.



Fonte: https://blog.paperspace.com/yolov7/

No exemplo ilustrado na figura 8, são evidenciadas as etapas conduzidas pelo YOLO, que resultam na inclusão de 845 caixas delimitadoras, derivadas do cálculo de 13x13, totalizando 169 células, e posteriormente multiplicadas por 5, representando o número de caixas por célula. Contudo, a maioria dessas caixas delimitadoras apresenta baixa relevância, sendo consideradas apenas aquelas cuja pontuação excede 30% do limiar de confiança, o qual é ajustável. Simultaneamente à detecção da presença de objetos dentro das caixas delimitadoras, o modelo também realiza a classificação de qual é o item contido dentro da caixa em questão, cuja pontuação final resulta da combinação entre a predição da classe e a da caixa delimitadora. Essa abordagem ilustra a meticulosa seleção e avaliação das detecções efetuadas pelo modelo YOLO, refletindo sua capacidade de discernimento e precisão na identificação de objetos em imagens (Alves, 2020).

3.5 Métricas de avaliação

Para avaliarmos os resultados obtidos ao final do treinamento e da validação, empregamos métricas amplamente utilizadas em tarefas de visão computacional. Essas métricas incluem, precisão, *recall*, mAP50, mAP50:95 e F1-*score*. Para compreender cada métrica, é fundamental primeiro entender o conceito de matriz de confusão e a Interseção sobre União (IoU, do inglês *Intersection over Union*).

3.5.1 Matriz de confusão

A matriz de confusão serve para indicar os erros e acertos, comparando os resultados obtidos, com os resultados esperados (Rodrigues, 2019).

Figura 9 - Matriz de confusão.

		Valor Predito	
		Sim	**Não**
Real	**Sim**	Verdadeiro Positivo (TP)	Falso Negativo (FN)
	Não	Falso Positivo (FP)	Verdadeiro Negativo (TN)

Fonte: https://diegonogare.net/2020/04/performance-de-machine-learning-matriz-de-confusao/

Como é possível ver na imagem acima, existem 4 tipos de resultados possíveis, cada um está explicado abaixo:

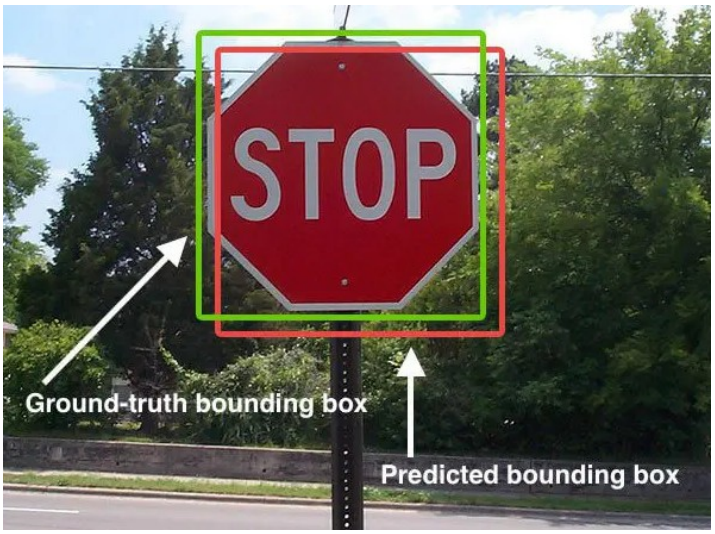
- **Verdadeiro positivo (TP – *true positive)*** - São os casos em que o modelo previu corretamente a classe positiva.
  - **Exemplo:** Em uma tarefa de detecção de doenças, se um exame identifica corretamente a presença de um sintoma (e a doença realmente está presente), isso é um verdadeiro positivo.
- **Verdadeiro negativo (TN – *true negative*)** - São os casos em que o modelo previu corretamente a classe negativa.
  - **Exemplo**: Na mesma tarefa de detecção de doença, se um exame identifica corretamente a ausência (e realmente não há doença), isso é um verdadeiro negativo.
- **Falso positivo (FP – *false positive*)** - São os casos em que o modelo previu incorretamente a classe positiva.
  - **Exemplo**: Se o exame indica a presença de uma doença, mas na verdade não há, isso é um falso positivo. É também chamado de "alarme falso".
- **Falso negativo (FN – *false negative*)** - São os casos em que o modelo previu incorretamente a classe negativa.
  - **Exemplo**: Se o exame não detecta a doença, mas a enfermidade está presente, isso é um falso negativo.

Todos esses valores são necessários para se calcular a métrica de avaliação do desempenho do modelo treinado.

3.5.2 Interseção sobre união (IoU)

Essa métrica é amplamente utilizada em modelos de identificação de objetos que empregam caixas delimitadoras (*bounding boxes*) em suas tarefas. A sua importância reside na capacidade de avaliar a precisão alcançada pelo modelo treinado, fornecendo uma medida crítica de desempenho para a eficácia do reconhecimento e delimitação de objetos (Rosebrock, 2022).

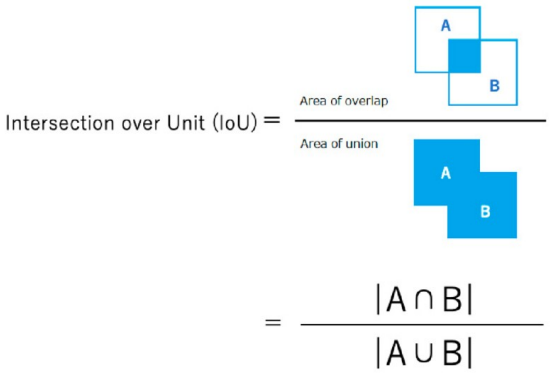
Figura 10 - A caixa com linha verde representa a caixa delimitadora real e a caixa vermelha delimitada pela rede neural.



Fonte: https://b2633864.smushcdn.com/2633864/wp-content/uploads/2016/09/iou_stop_sign.jpg?lossy=2&strip=1&webp=1

A IoU é usada para quantificar o quão bem uma caixa delimitadora prevista por um modelo de visão computacional se sobrepõe a caixa delimitadora real (*ground-truth*) como o exemplo da figura 10, onde a caixa verde representa a caixa delimitadora real e a vermelha representa a que foi identificada pelo modelo. O IoU é calculado como a área da sobreposição entre as duas caixas dividida pela área da união das caixas. Em termos práticos, se tivermos uma caixa delimitadora A que representa a previsão do modelo e uma caixa delimitadora B que representa a área real, o IoU seria a área de intersecção de A e B dividida pela área da união de A e B, a maneira que esse cálculo é feito pode ser visto com mais clareza na figura 11. Esta métrica é crucial para treinar modelos de visão computacional, que utilizam o IoU para melhorar a precisão das previsões de localização dos objetos. A maneira como esse cálculo é realizado sempre resultará em uma variação entre 0 e 1, sendo que 1 indica uma detecção precisa, enquanto 0 indica o contrário (Rosebrock, 2022).

Figura 11 - Fórmula do IoU, sendo que o valor pode variar entre 0 e 1.



Fonte: Iwasa et al,. 2022.

A IoU também é um fator determinante para a classificação de uma detecção como verdadeira ou falsa. Comumente, um limiar (*threshold*) é definido, como por exemplo, 0,5 (50%). Se a IoU entre a caixa predita e a caixa real for maior ou igual ao limiar, considera-se que a detecção é verdadeira positiva (*True Positive*). Caso contrário, é considerada uma falsa positiva (*False Positive*) (Shah, 2023).

3.5.3 Precisão

O cálculo da precisão é realizado dividindo o número de verdadeiros positivos pelo total de exemplos classificados como positivos, que é a soma dos verdadeiros positivos e dos falsos positivos (Rodrigues, 2019). A fórmula matemática pode ser expressa como:

$$precisão = \frac{verdadeiros\ positivos}{verdadeiros\ positivos + falsos\ positivos}$$

Este cálculo é fundamental para avaliar a exatidão de um modelo de classificação, assim, “a precisão pode ser usada em uma situação em que os Falsos Positivos são considerados mais prejudiciais que os Falsos Negativos” (Rodrigues, 2019). Refletindo a proporção de predições positivas corretas em relação ao total de predições positivas feitas pelo modelo.

3.5.4 Recall

O *Recall*, também conhecido como sensibilidade, é uma métrica de desempenho utilizada para avaliar a eficácia de um modelo de classificação. O cálculo de recall é definido

como a razão entre o número de verdadeiros positivos e a soma dos verdadeiros positivos e dos falsos negativos (Rodrigues, 2019). A fórmula matemática é:

$$recall = \frac{verdadeiros\ positivos}{verdadeiros\ positivos + falsos\ negativos}$$

Essa métrica mede a capacidade do modelo de identificar corretamente todas as instâncias relevantes (verdadeiros positivos) no conjunto de dados. "O *recall* pode ser usado em uma situação em que os Falsos Negativos são considerados mais prejudiciais que os Falsos Positivos" (Rodrigues, 2019). Por exemplo, o modelo deve identificar todos os pacientes doentes, mesmo que classifique alguns saudáveis como doentes (falsos positivos). Isso significa que o modelo deve ter um alto *recall*, pois classificar pacientes doentes como saudáveis pode ter consequências graves (Rodrigues, 2019).

3.5.5 Mean aavarage precision (mAP)

O *mean average precision* (mAP) é uma métrica utilizada para avaliar a performance de modelos de detecção de objetos e classificação multiclasse. O cálculo leva em consideração todas as métricas discutidas anteriormente.

Primeiro precisamos calcular a precisão média de cada classe, sendo calculada como a média ponderada das precisões em cada limiar definido, no caso do mAP50, "a precisão média calculada com um limiar de intersecção sobre união (IoU) de 0,50. É uma medida da precisão do modelo considerando apenas as detecções 'fáceis'" (Ultralytics, 2023, tradução própria).

No caso do mAP50:95, "a precisão média é calculada em diferentes limiares de IoU, variando de 0,50 a 0,95, a fim de obter uma visão abrangente do desempenho do modelo em diferentes níveis de dificuldade de detecção." (Ultralytics, 2023, tradução nossa). O cálculo pode ser observado abaixo:

$$mAP = \frac{1}{N} \cdot \sum_{i=1}^{N} AP_i$$

AP = precisão média ; N = número de classes; i = iteração

É possível observar que o mAP é uma média entres todas as médias obtidas em cada classe, ou seja, é a somatória de todos os *avarage precision (*AP) dividido pela quantidade de categorias.

3.5.6 F1-score

O F1-*score* “determina o limiar de confiança ideal, onde a precisão e o recall resultam na maior pontuação F1. A pontuação F1 mede o equilíbrio entre precisão e recall.” (Shah, 2023, tradução própria). O cálculo F1 pode ser analisado abaixo:

$$F1-score=\frac{2\cdot(precisão\cdot recall)}{precisão+recall}$$

Apesar de ser uma operação matemática simples, essa métrica muito importante para uma avaliação geral do desempenho de uma rede neural.

3.6 Trabalhos relacionados

O estudo *Efficient Tobacco Pest Detection in Complex Environments Using an Enhanced YOLOv8 Model* por Daozong Sun et al. (2024) destaca um modelo YOLOv8 aprimorado para detectar pragas do tabaco em ambientes complexos. A equipe coletou manualmente 460 imagens de quatro tipos de pragas do tabaco: 143 de lagartas, 102 de aranhas vermelhas, 113 de afídeos e 102 de lagarta-enroladeira. As imagens foram anotadas usando o software *Labelme* e aumentadas artificialmente com um estilo mosaico, resultando em um total de 4000 imagens, das quais 80% foram usadas para treinamento e 20% para validação. O modelo aprimorado incorpora o mecanismo de atenção SimAM e o módulo VoV-GSCSP para melhorar a extração de características e a localização do alvo. Testes comparativos indicam que o modelo reduz a quantidade de parâmetros e os requisitos computacionais, enquanto melhora métricas de desempenho como médias gerais, recall e precisão. O YOLOv8 aprimorado demonstra superior precisão de detecção e eficiência em comparação com outras estruturas de detecção de objetos. Em comparação com a versão padrão YOLOv8L, a versão aprimorada usa 23,4 milhões de parâmetros a menos e apresenta uma melhora de 2% nas médias gerais e na precisão, atingindo 88,9% e 92,7%, respectivamente. Segundo os autores o YOLOv8 aprimorado equilibra o consumo de recursos com a precisão da detecção. Alcançando uma redução significativa nos parâmetros e nos requisitos computacionais, ao mesmo tempo que aprimora as capacidades de detecção do modelo. (Daozong et al, 2024, p. 19, tradução nossa).

O trabalho *Alpha-EIOU-YOLOv8: An Improved Algorithm for Rice Leaf Disease Detection*, apresenta um estudo de pesquisa de Trinh et al. O trabalho traz um algoritmo aprimorado, Alpha-EIOU-YOLOv8, projetado para detectar doenças nas folhas do arroz com

rapidez e precisão. O sistema utiliza uma configuração de hardware que consiste em um microcontrolador, câmera USB e módulo GSM para capturar imagens em tempo real de plantas de arroz no campo, identificar doenças de plantas usando uma versão modificada da YOLOV8 para detecção, e alertar os agricultores por e-mail e mensagem de texto. Para a etapa de treinamento, os autores coletaram manualmente 1643 imagens, sendo que o *dataset* contava com 3 pragas comuns à cultura do arroz: lagarta desfolhadora, lagarta-enroladeira e a mancha-marrom. As imagens foram adquiridas em diferentes plantações de arroz e em diferentes condições climáticas. Também foi realizado um aumento artificial nesse conjunto de imagens, sendo aplicadas modificações verticais, horizontais e translações. Além disso, também foram feitas imagens mosaicas. Ao final o *dataset* continha 3175 imagens, dividido em 3 partes, sendo 2608 para treinamento, 326 para validação e 241 para testes. A seção de metodologia descreve a visão geral do sistema, design de hardware, preparação de dados, arquitetura YOLOv8, métricas de avaliação e cálculo de perdas. O estudo emprega métricas como precisão, recall, mAP e pontuação F1 para fins de avaliação. A pesquisa demonstra melhorias no desempenho de detecção de doenças em comparação com outros métodos de última geração. No geral, a pesquisa de Trinh et al. contribui para os avanços na engenharia agrícola, introduzindo um algoritmo mais eficiente para detectar doenças nas folhas do arroz, que pode ajudar os agricultores na gestão oportuna de doenças e na proteção das culturas.

O terceiro trabalho é o V*egetable disease detection using an improved YOLOv8 algorithm in the greenhouse plant environment* desenvolvido pelos pesquisadores Wang e Liu (2024). Este estudo apresenta o modelo YOLOv8n-vegetal, um algoritmo leve e eficiente para detecção de doenças em vegetais. O modelo incorpora vários aprimoramentos, incluindo o módulo GhostConv, módulo de atenção de percepção de oclusão, camada de detecção de objetos de pequeno porte e função de perda de HioU. Os autores treinaram essa rede modificada para detectar 20 doenças nas culturas do pepino, berinjela e tomate. Ao total foram coletadas cerca de 28000 imagens, divididas em 80% para treinamento, 10% validação e 10% para testes. Os resultados demonstram o desempenho superior do modelo na detecção precisa das doenças, atingindo médias gerais de 92.91% e com uma velocidade superior quando comparadas com a versão padrão do YOLOV8n. O destaque ficou principalmente para as doenças de pequena escala e ocluídas, sendo essas uma dificuldade que o YOLO encontra devido a maneira como funciona. O modelo proposto mostra-se promissor para aplicação no mundo real na detecção e manejo de doenças vegetais.

4 METODOLOGIA

Para o desenvolvimento deste trabalho foram necessárias as seguintes etapas: criação do banco de imagens (*dataset*), rotulagem das amostras, divisão do *dataset*, aumento artificial (*augmentation*), treinamento e validação cruzada. A seguir serão descritas cada uma das etapas.

4.1 Criação do banco de imagens

Para funcionar eficazmente, as redes neurais utilizam neurônios interconectados e um algoritmo de aprendizagem para ajustar os pesos sinápticos e alcançar objetivos de design desejados (Haykin, 2009, p. 2, tradução nossa).

Ter um conjunto de dados com quantidade e variedade de amostras é fundamental para obter bons resultados no treinamento de redes neurais. Atualmente já existem vários sites que são focados no compartilhamento de conjuntos de dados prontos para utilização, como o *Kaggle* e o *Roboflow universe*.

Porém, nem sempre é possível encontrar um banco de imagens com o que se procura, sendo necessário, por vezes, a criação manual de construção do *dataset*, passando pelas etapas de aquisição de imagens, e pré-processamento das amostras.

Para esta pesquisa, as imagens foram obtidas através da internet com os trabalhos "*Pictures of diseased soybean leaves by category captured in field and with controlled backgrounds: Auburn soybean disease image dataset* (ASDID)" (Bevers; Sikora; Hardy, 2022) e "*SoyNet: Indian Soybean Image dataset with quality images captured from the agriculture field (healthy and disease Images)*" (Rajput; Shukla; Thakur, 2023).

No trabalho de Bevers, Sikora e Hardy (2022), os autores disponibilizaram um banco de imagens contendo oito categorias diferentes de folhas de soja, totalizando mais de 40 GB. As categorias incluem, mancha bacteriana marrom, mancha púrpura da semente, míldio, mancha olho-de-rã, mancha-alvo, ferrugem-asiática, deficiência de potássio e folha saudável. As imagens foram capturadas com uma câmera Canon EOS 7D Mark II, com resolução de 5472 x 3648 *pixels* e tamanho médio de arquivo de 4MB.

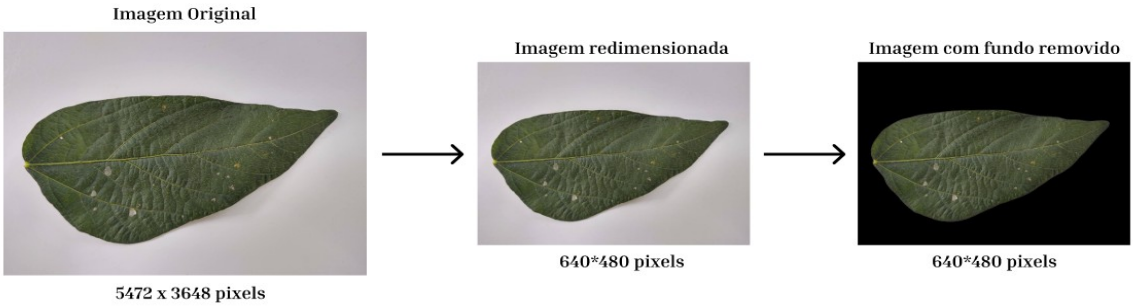
O segundo banco de imagens também conta com várias categorias diferentes, porém a intenção foi conseguir mais imagens da folha saudável, ignorando as outras categorias. Ao total foram adicionadas 509 capturas da folha saudável, sendo que as imagens foram

capturadas utilizando uma câmera Nikon L810 e um smartphone Motorola G4, com resolução de 5472 x 3648 *pixels* e tamanho médio de 5,7 MB.

Com os *datasets* definidos, iniciou-se o pré-processamento das imagens. Inicialmente, as imagens foram redimensionadas utilizando a biblioteca PIL (Python Imaging Library) da *Python* para uma resolução de 640 x 480 *pixels*, reduzindo para uma média de 24,6 KBs por imagem, pois além de facilitar o manuseio das imagens, é fundamental remover informações desnecessárias e manter apenas os aspectos relevantes para impactar positivamente na velocidade de treinamento (Elgendy, 2020).

Em seguida, foi utilizada outra biblioteca *Python* chamada *rembg* para remover o fundo das imagens. Essa ferramenta foi especialmente projetada para essa finalidade, ajudando a remover objetos que não são de interesse, trazendo maior eficiência na hora do treinamento da rede neural. Esse processo de tratamento foi essencial para preparar as imagens para o treinamento de modelos de aprendizado profundo.

Figura 12 - tratamento realizado nas imagens.



Fonte: elaborado pelo autor

4.1.1 Rotulagem das amostras

Após concluir as alterações nas imagens, iniciou-se a etapa de rotulagem. Este procedimento é conduzido manualmente, onde são inseridas etiquetas de coordenadas nas imagens, conhecidas como *bounding boxes* (caixas delimitadoras). Tais etiquetas fornecem à rede neural as coordenadas da localização sobre os objetos de interesse na imagem, permitindo que ela aprenda a identificar os padrões associados a esses objetos. Diversas ferramentas estão disponíveis para facilitar essa etapa, sendo o *Roboflow* escolhido para este trabalho devido a sua abrangência e facilidade de uso, possibilitando uma anotação eficiente e organização do banco de imagens.

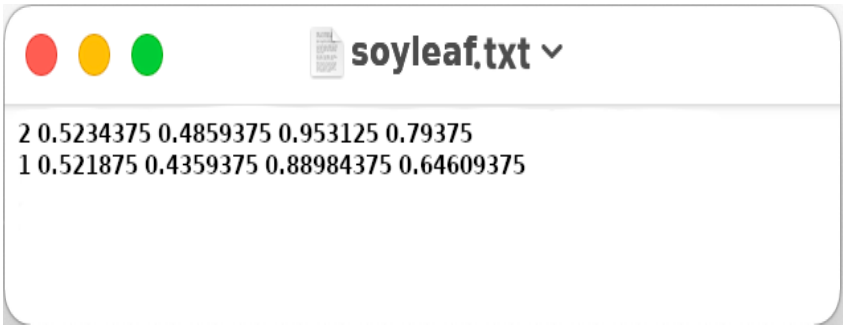
Cada modelo de rede neural utiliza um formato de anotação específico para as *bounding boxes*. O formato de anotação utilizado para este trabalho, consiste em um arquivo

de texto para cada imagem rotulada, onde cada linha representa uma etiqueta e contém as coordenadas (x, y, largura, altura) da caixa delimitadora, seguido pela classe do objeto e possíveis atributos adicionais, dependendo da aplicação específica (Ultralytics, 2023).

Figura 13 - Exemplo de anotação; caixa verde indica “folha da soja” e a vermelha indica “saudável”.

Fonte: elaborado pelo autor

Figura 14 - Arquivo de texto com as coordenadas das anotações.



Fonte: elaborado pelo autor

No total, foram anotadas 1586 imagens, distribuídas da seguinte forma: 200 para manchas olho-de-rã, 200 para manchas-alvo e 1186 para folhas saudáveis. Além dessas três categorias, foi criada uma categoria adicional para identificar se a imagem representa uma folha de soja, sendo esta aplicada a todas as imagens. Embora possa parecer incorreto o uso de um número muito maior de imagens de folhas saudáveis, é importante considerar o número de caixas delimitadoras por imagem. Nas imagens de folhas saudáveis, geralmente são necessárias apenas uma ou duas caixas delimitadoras, enquanto as imagens de manchas olho-de-rã e manchas-alvo requerem mais de 20 anotações em algumas situações.

Tabela 1 – Quantidade de imagens anotadas x quantidade de bounding boxes

Categoria	Quantidade de imagens	Quant. bounding boxes
Folha saudável	1186	1283
Mancha-alvo	200	2898
Mancha olho-de-rã	200	7649
Folha da soja	586	1782
Total	**1586**	**13612**

Fonte: Elaborado pelo próprio autor

4.1.2 Divisão do *dataset*

Após todos esses passos, para o treinamento de um modelo de aprendizado de máquina, os dados são tipicamente divididos em conjuntos de treinamento e teste. O conjunto de treinamento é utilizado para ajustar os pesos do modelo, enquanto o conjunto de teste é reservado para avaliar a capacidade de generalização do modelo em dados não vistos anteriormente. É uma prática fundamental nunca utilizar os dados de teste durante o treinamento, a fim de evitar qualquer tipo de viés ou sobreajuste por parte do modelo. A avaliação do desempenho do modelo é realizada após cada época (iteração) de treinamento, utilizando métricas como precisão e erro. Para evitar a quebra dessa regra, é comum também utilizar um conjunto de validação para ajustar os parâmetros do modelo durante o treinamento. Após a conclusão do treinamento, o desempenho final do modelo é testado no conjunto de dados de teste para uma avaliação imparcial de sua eficácia (Elgendy, 2020).

É comum dividir as imagens obtidas em conjuntos distintos para treinamento e validação. As proporções mais frequentemente utilizadas são 66% para treinamento e 33% para validação, ou 75% para treinamento e 25% para validação, respectivamente (Rosebrock, 2017, tradução nossa). Baseado nisso, o *dataset* para este trabalho foi dividido em 70% treinamento, 20% validação e 10% para teste, resultando respectivamente em 1155, 305, 151 amostras de imagens para cada conjunto.

4.1.3 Aumento artificial *(Augmentation)*

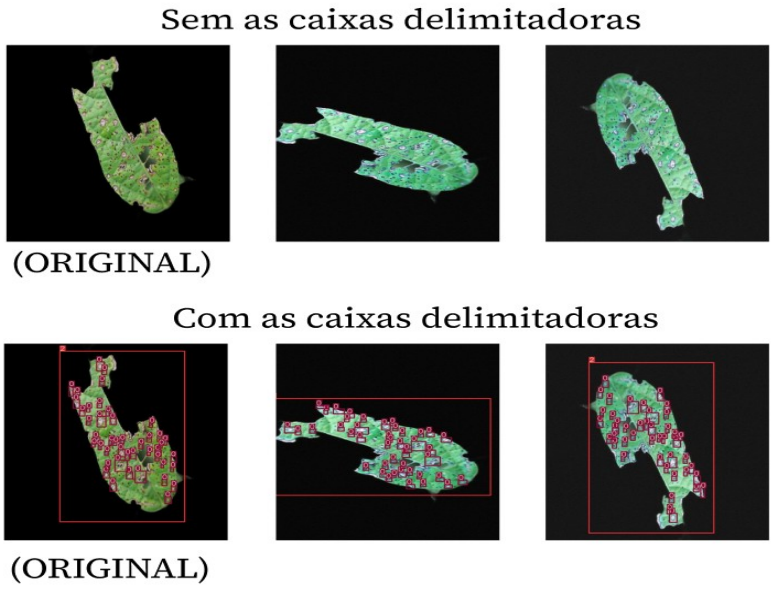
Conforme mencionado anteriormente, a quantidade de dados disponíveis é diretamente proporcional ao sucesso do treinamento de uma rede neural. No entanto, a aquisição da quantidade desejada de dados nem sempre é viável. Para mitigar esse problema, costuma-se empregar técnicas de aumento artificial de dados, conhecidas como data *augmentation*. O data *augmentation* é responsável por gerar uma variedade de novas instâncias a partir dos dados

existentes, aumentando assim a quantidade e diversidade dos dados de treinamento. A utilização dessas técnicas desempenha um papel crucial na prevenção do overfitting, permitindo que o modelo generalize melhor para dados novos e não vistos, ao invés de se ajustar excessivamente aos dados de treinamento disponíveis. Além disso, o aumento de dados pode aprimorar a precisão e a robustez do modelo, ao expô-lo a uma variedade mais ampla de dados (Faizan, 2023).

Para o aumento artificial dos dados, foi utilizado o *Albumentations*, uma biblioteca *Python* desenvolvida especificamente para essa finalidade. Além de realizar as modificações nas imagens, o *Albumentations* ajusta automaticamente a posição das caixas delimitadoras, garantindo a consistência dos dados. É importante destacar que esse processo é aplicado apenas ao conjunto de treinamento, preservando a integridade dos conjuntos de validação e teste.

Ao todo, foram realizadas 9 iterações de aumento de dados, resultando em um total de 11.550 imagens, incluindo as originais e as modificadas. Durante esse processo, foram aplicadas diversas transformações, tais como inversão horizontal, variações de brilho e contraste, adição de ruído, rotação aleatória de 90º e rotação vertical. Essas iterações contribuíram significativamente para a diversidade e robustez do conjunto de dados, beneficiando o treinamento do modelo de detecção de doenças nas folhas de soja.

Figura 15 - Aumento artificial utilizando o *Albumentations.*



fonte: elaborado pelo autor.

Acima é possível ver que a partir de uma imagem original é possível gerar várias outras utilizando modificações. O *Albumentation* além de gerar as novas imagens, também gera as anotações das caixas delimitadoras, sem a necessidade de realizar novas anotações.

4.2 Treinamento do YOLO

Inicialmente, optou-se pela utilização da plataforma Google Colab devido à sua conveniência, oferecendo um ambiente de desenvolvimento pré-configurado e de fácil acesso. No entanto, a versão gratuita da plataforma apresenta restrições significativas em termos de tempo de uso contínuo e capacidade de memória disponível. Estas limitações impuseram a necessidade de adotar uma estratégia alternativa para prosseguir com a pesquisa de forma eficaz. Por conseguinte, optou-se por utilizar um computador pessoal equipado com um processador Intel © Core(™) i7-12700 CPU @ 2.10GHz x 12; memória de 8GB, DDR4, 3200MHz; SSD M.2. de 1TB; Placa gráfica NVIDIA R © GeForce © RTX 3070 TI de 8GB, GDDR5; Sistema operacional Windows 11 Pro.

Para viabilizar o treinamento, procedeu-se à configuração do ambiente, empregando-se a plataforma Anaconda e o *Jupyter Notebook* como ambiente de desenvolvimento e editor de código. Adicionalmente, foi necessário utilizar o CUDA *Toolkit*, uma suíte de ferramentas essencial para aproveitar ao máximo o poder de processamento das placas gráficas NVIDIA durante o treinamento de modelos de aprendizado profundo.

Para a compreensão do próximo paragrafo, serão apresentados alguns novos termos que são importantes no contexto do treinamento de modelos como o YOLO. Época (epoch) refere-se a uma passagem completa por todo o conjunto de dados de treinamento. Paciência (p*atience)* é um parâmetro usado em técnicas de parada antecipada (*early stopping*), definindo o número de *epochs* consecutivas sem melhora na métrica de validação antes de interromper o treinamento para evitar sobreajuste (*overfitting*). Tamanho de lote (*batch size*) é o número de amostras de dados processadas antes da atualização dos parâmetros do modelo, influenciando a estabilidade e a velocidade do treinamento.

O treinamento foi feito utilizando a YOLOv8s, uma versão mais leve, que conta com 225 camadas e 11.137.148 parâmetros. Ao total foram utilizadas 200 *epochs* com uma "paciência" de 100. Foi utilizado 16 *batch size,* correspondente ao grupo de imagens selecionadas para o treinamento da rede. Os resultados evidenciam que lotes pequenos

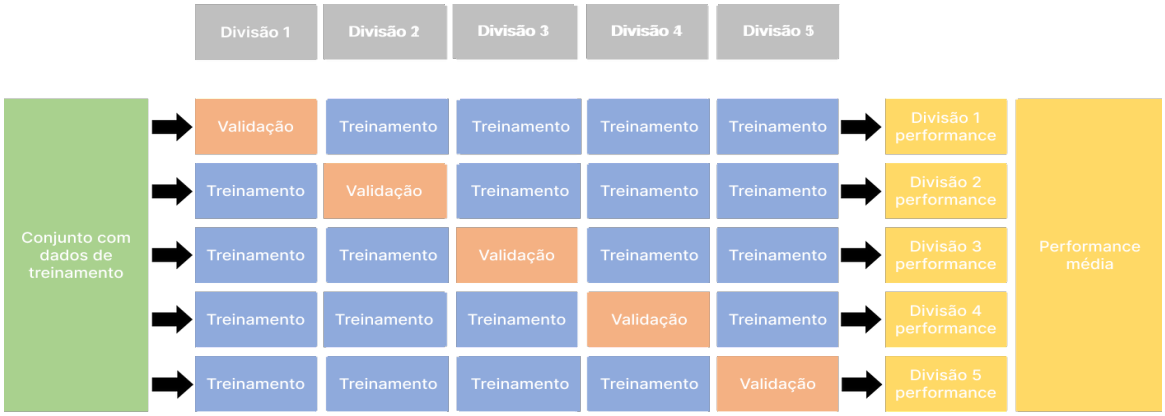
asseguram estabilidade e generalização superiores durante o treinamento, com os tamanhos de lote mais eficazes geralmente variando entre 2 e 32 (Masters e Luschi, 2018).

O otimizador foi selecionado automaticamente pela rede neural sendo definido o método do Gradiente Descendente Estocástico (SGD) com uma taxa de aprendizado de 0.1. Um otimizador de rede neural é um algoritmo usado durante o treinamento para ajustar os pesos das conexões entre neurônios, visando minimizar a função de perda do modelo. O SGD, um dos algoritmos mais amplamente empregados na atualidade, destaca-se por sua eficácia na atualização iterativa dos pesos da rede neural com base em gradientes calculados em amostras de dados de treinamento (Carvalho, 2021).

4.2.1 Validação cruzada

A validação cruzada é uma técnica amplamente utilizada para avaliar o desempenho de modelos em conjuntos de dados limitados. Ela consiste na divisão dos dados em subconjuntos mutuamente exclusivos, onde um subconjunto é utilizado para treinamento do modelo e os demais são reservados para validação. Esse processo é repetido várias vezes, com diferentes combinações de dados de treinamento e validação, permitindo uma avaliação mais abrangente do desempenho do modelo. A validação cruzada é particularmente útil para evitar o sobreajuste (*overfitting*) e fornecer estimativas mais confiáveis da capacidade de generalização do modelo para novos dados.

Figura 16: Validação cruzada com 5-*fold*.



Fonte: https://docs.ultralytics.com/pt/guides/kfold-cross-validation/#introduction

Ela consiste em dividir o conjunto de dados em k subconjuntos (ou *folds*) de tamanho aproximadamente igual. O modelo é treinado k vezes, utilizando k-1 subconjuntos como dados de treinamento e o subconjunto restante como dados de validação. Em cada uma das k iterações, um *fold* diferente é utilizado como conjunto de validação, enquanto os demais são

combinados para treinamento. Ao final das k iterações, são obtidos k resultados de desempenho (como acurácia ou erro) que são então combinados (geralmente calculando a média) para obter uma estimativa final da performance do modelo. Como no exemplo da figura 16, a validação cruzada com 5 divisões (5-*fold cross-validation*), o conjunto de dados seria dividido em 5 partes, e o modelo seria treinado e validado 5 vezes, cada vez utilizando uma parte diferente como conjunto de validação e as outras partes como treinamento. (Ultralytics, 2023).

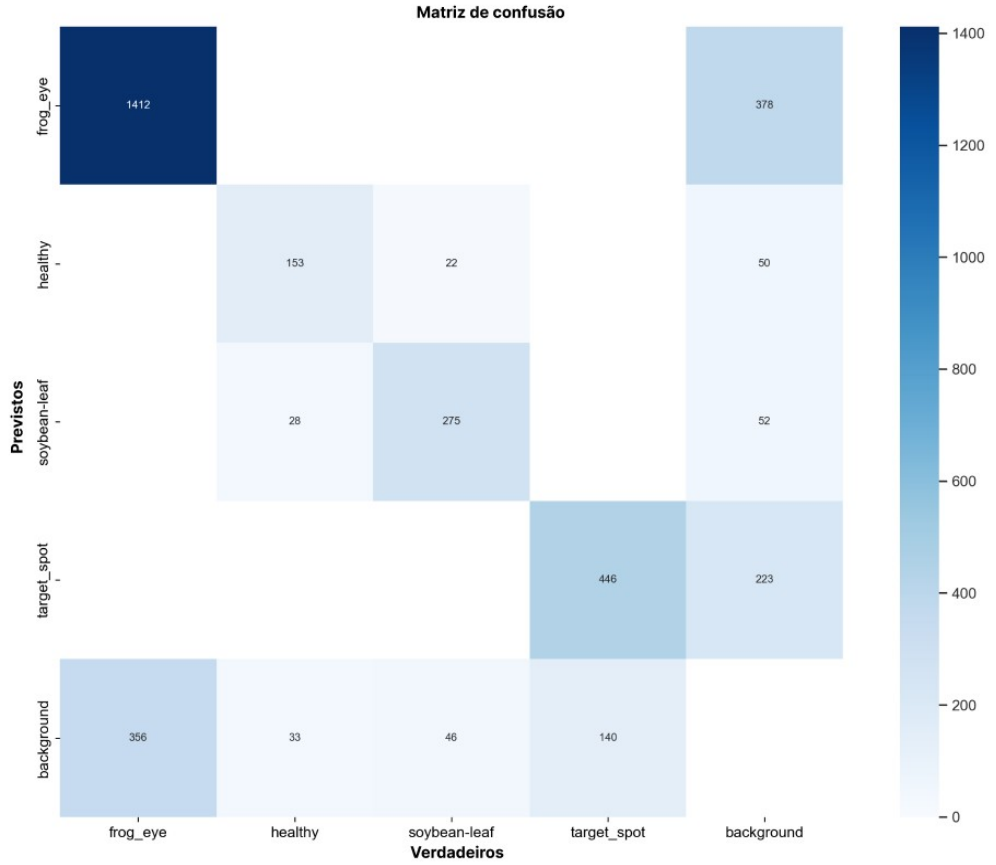
5 RESULTADOS

Nesta seção, apresentaremos os resultados obtidos, incluindo medidas de desempenho e o custo computacional mensurado pelo tempo de cada treinamento. Focaremos na melhor *epoch* obtida durante o treinamento, pois essa é a fase em que o modelo alcançou seu ápice de desempenho durante o processo de aprendizado. A escolha da melhor *epoch* é crucial para avaliar a capacidade máxima do modelo treinado, representando o momento em que ele demonstra o melhor equilíbrio entre precisão e generalização.

Com base nas etapas delineadas na metodologia, foi identificado que a vigésima quarta época apresentou o melhor desempenho, seguido de uma diminuição gradual até que o modelo atingiu sua estabilidade e alcançou o limite de tolerância, encerrando-se na época 179, levando um total de 5 horas e 5 minutos de treinamento.

No gráfico 1 é apresentado a matriz de confusão obtida após o treinamento com uma descrição dos resultados. O desempenho do modelo ao longo das épocas é detalhado no gráfico 2, que ilustra a trajetória do desempenho do modelo durante o seu treinamento. Estes gráficos fornecem uma visão abrangente do comportamento do modelo durante o treinamento, permitindo uma avaliação detalhada dos pontos de desempenho máximo, as tendências e o alcance da estabilidade final.

Gráfico 1 - Matriz de confusão obtido após treinamento.

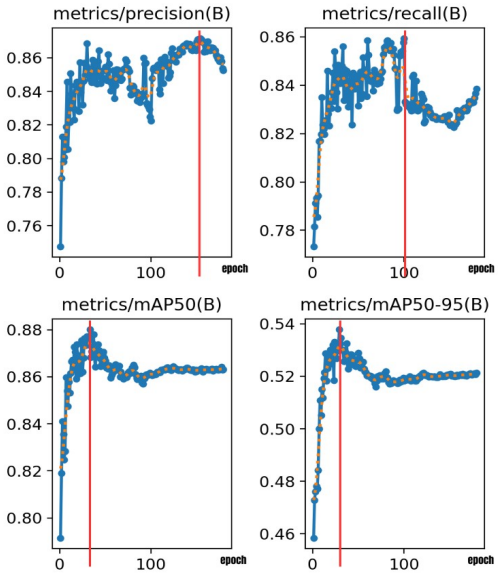


Fonte: elaborado pelo próprio autor.

A matriz de confusão apresentada foi obtida a partir do processo de validação do modelo treinado. Cada célula da matriz representa a quantidade de acertos (verdadeiros positivos) e erros (falsos negativos) para cada classe considerada. Especificamente, para a classe "*frog_eye*" (mancha olho-de-rã), o modelo identificou corretamente 1412 instâncias (verdadeiros positivos) e falhou em detectar 356 (*background*) instâncias, classificando-as incorretamente (falsos negativos). Na classe "*healthy*" (saudável), o modelo obteve 153 verdadeiros positivos, indicando a correta identificação de 153 instâncias, enquanto houve 61 (33 *background* + 28 *soybean-leaf)* falsos negativos, representando instâncias não detectadas corretamente. Para a classe "*soybean-leaf*" (folha da soja), foram registrados 275 verdadeiros positivos, evidenciando o número de detecções corretas, e 68 (22 *healthy* + 46 *background*) falsos negativos. Na classe "*target_spot*" (mancha-alvo), o modelo conseguiu identificar corretamente 446 instâncias (verdadeiros positivos), mas falhou em 140 (*background*) casos. É importante destacar que a categoria "*background*" (fundo da imagem) não representa nenhuma classe específica e, portanto, não é considerada na avaliação.

A análise da matriz de confusão é crucial para entender o desempenho detalhado do modelo em cada categoria. Os verdadeiros positivos indicam a eficácia do modelo em identificar corretamente as instâncias de cada classe, enquanto os falsos negativos apontam as limitações do modelo em reconhecer todas as instâncias positivas.

Gráfico 2- Gráfico com mAP50, mAP50:95, precisão e recall obtidos durante o treinamento.



Fonte: elaborado pelo próprio autor.

O gráfico acima representa a variação entre cada época durante o treinamento. Inicialmente o mesmo mostra uma rápida ascensão e após isso uma estabilização, não ocorrendo grande variação entre as *epochs*. As maiores variações foram encontradas no *recall* e na precisão, sendo que a primeira entrou em uma leve queda após a época 100, porém mostra recuperação ao final. A precisão também apresentou uma variação e atingiu um pico de aproximadamente 89% por volta da época 150.

As métricas mAP50 e mAP50:95 mostraram pouquíssimas variações, após atingirem seu ápice na época 24, posteriormente isso houve uma leve queda, e então uma estabilidade. Os dados obtidos da melhor *epoch* também podem ser analisados na tabela abaixo:

Tabela 2: métricas obtidas a partir da validação da melhor epoch.

Categoria	Precisão	*Recall*	mAP50	MAP50:95	F1-*score*
Folha saudável	0.924	0.972	0.981	0.753	0.947
Mancha-alvo	0.717	0.682	0.714	0.283	0.699
Mancha olho-de-rã	0.821	739	0.819	0.369	0.778
Folha da soja	0.96	0.983	0.99	0.747	0.971
Total	**0.856**	**0.844**	**0.876**	**0.538**	**0.85**

Fonte: Elaborado pelo próprio autor.

Por hora será levado em consideração principalmente as métricas F1-*score* e mAP, sendo que com ambas podemos ter uma noção geral do desempenho encontrado. Como dito anteriormente, F1-*score* é uma métrica harmônica entre precisão e *recall*, indicando que o modelo atingiu uma média de 85% no total.

É possível observar que as categorias mancha-alvo e mancha olho-de-rã atingiram métricas inferiores quando comparadas com as outras categorias, em especial a mAP50:95, isso se deve a principal característica de passada única do YOLO, "embora possa identificar rapidamente objetos em imagens, ele tem dificuldade em localizar com precisão alguns objetos, especialmente os pequenos." (Redmon et al., 2016, tradução nossa), e uma vez que as caixas delimitadoras dessas doenças são pequenas, esse fator acaba ficando evidente. Apesar dessa limitação, o modelo treinado ainda identifica as pragas investigadas corretamente, mesmo apresentando um baixo limiar de confiança. Na imagem abaixo é possível analisar uma demonstração do modelo treinado identificando uma folha com a mancha olho-de-rã.

Figura 17 - Mostra do modelo detectando a mancha olho-de-rã (categoria que obteve as piores métricas).

Fonte: elaborado pelo autor

Cada caixa delimitadora representa uma detecção realizada pelo modelo. No exemplo da figura 19, foi detectada uma caixa delimitadora para folha de soja (*soybean-leaf*), destacada em laranja, com uma confiança de 75%. Além disso, foram detectadas 30 caixas delimitadoras para manchas olho-de-rã (*frog-eye*), destacadas em vermelho, com níveis de confiança variando entre 53% e 74%. É importante ressaltar que o modelo classificou corretamente todas as instâncias das classes presentes na imagem.

Mesmo apresentando métricas relativamente baixas para algumas categorias, o modelo ainda conseguiu identificar a maior parte das manchas presentes na folha, gastando de 14.4ms

para concluir a tarefa. Com essa velocidade, o modelo poderia ser utilizado sem problemas para detecção em tempo real, ou em vídeos.

Para garantir que não houve um sobreajuste dos pesos durante o treinamento, também foi realizada a validação cruzada, que após aplicadas variações, acabou gerando médias não muito distantes das exibidas anteriormente, pode-se observar as médias obtidas após 5-subconjuntos (*folds)* na tabela abaixo:

	Precisão	**Recall**	**mAP50**	**mAP50:95**	**F1-*score***
Total	0.856	0.839	0.872	0.544	0.846

Tabela 3: médias das métricas obtidas na validação cruzada.

É importante ressaltar que foram analisadas apenas as métricas gerais de cada treinamento realizado durante a validação cruzada, no total levou-se um tempo estimado de 26 horas para realizar todas as 5-*folds*. A análise das métricas de desempenho do modelo revela um desempenho equilibrado. A precisão de 0.856 indica alta eficácia em evitar falsos positivos, enquanto o *recall* de 0.839 demonstra a capacidade de detectar a maioria das instâncias positivas. A mAP50 de 0.872 destaca uma boa precisão média com um limiar de IoU de 50%, e a mAP50:95 de 0.544 reflete um desempenho razoável sob critérios mais rigorosos. O F1-score de 0.846 sugere um bom equilíbrio entre precisão e *recall*, indicando que o modelo está balanceado.

Por fim, foi realizada uma última validação utilizando o conjunto de dados de teste. Essa última etapa é importante para analisarmos como o modelo treinado lida com imagens não utilizadas anteriormente. O conjunto teste foi separado e não passou por *augmentation*, justamente para simular a aplicação do modelo em um ambiente real.

Esse *dataset* de teste contou com 151 imagens, sendo 26 imagens da mancha olho-de-rã, 27 manchas-alvo e 98 para a folha saudável. Os números obtidos podem ser analisados na tabela abaixo:

Categoria	**Precisão**	**Recall**	**mAP50**	**MAP50:95**	**F1-*score***
Folha saudável	0.927	0.988	0.965	0.757	0.956
Mancha-alvo	0.749	0.619	0.69	0.293	0.677
Mancha olho-de-rã	0.777	0.8	0.833	0.372	0.788
Folha da soja	0.957	0.982	0.991	0.763	0.969
Total	**0.853**	**0.847**	**0.87**	**0.546**	**0.849**

Tabela 4: métricas obtidos após validação final com banco de imagens de teste.

As métricas apresentam resultados não muito diferentes das obtidas anteriormente, com apenas algumas flutuações. No total, o modelo apresenta uma precisão média de 0.853 e um *recall* de 0.847, com um mAP50 de 0.87 e um mAP50:95 de 0.546. O F1-score geral de 0.849 sugere que o modelo possui um desempenho robusto e equilibrado na maioria das

categorias. No entanto, a detecção de "Mancha-alvo" e "Mancha olho-de-rã" são áreas que ainda apresentaram menor precisão e *recall*, afetando a eficácia global do modelo. Esses resultados indicam que, enquanto o modelo é geralmente eficaz, algumas categorias ainda são desafio para o modelo. Dessa maneira analisamos que apesar de suas limitações, o modelo treinado apresenta métricas aceitáveis e que foram validadas de maneiras diferentes.

6 CONCLUSÃO

Este trabalho teve como objetivo utilizar a YOLOv8 para o reconhecimento das pragas mancha-alvo e mancha olho-de-rã que afetam as culturas de soja, com o qual foi construído um banco de imagens (*dataset*) de ambas doenças e da folha saudável com o objetivo de obter um modelo que pudesse identificar com confiança tais doenças.

Para a criação deste *dataset* foram utilizadas imagens obtidas na internet, tendo ao final 1586 amostras, divididas em 1186 da folha saudável, 200 da mancha-alvo, 200 da mancha olho-de-rã e 1586 da folha da soja. Após o tratamento dessas imagens, foram feitas as anotações das caixas delimitadoras, indicando para a rede neural o objeto de interesse que ela deveria aprender o padrão. Ao total foram feitas 13612 anotações manualmente, representando 1283 para folha saudável, 2898 para mancha-alvo, 7649 para mancha olho-de-rã e 1782 para folha da soja.

Para garantir uma maior variedade de imagens foi utilizado o recurso de *augmentation*, no qual é realizado um aumento artificial da quantidade de imagens que, aumentou o *dataset* inicial para 11550 imagens, sendo divididas em 70% para treinamento, 20% para validação e 10% para testes.

Os testes realizado após o treinamento mostraram que a rede neural obteve 53% na métrica mAP50:95 e 85% na F1-*score*, ressaltando um desempenho razoável, porém quando analisadas individualmente, as classes das doenças mostraram resultados baixos, sendo 28% e 70% para as métricas mAP50:95 e F1-*score* da mancha-alvo respectivamente. As métricas para a mancha olho-de-rã foram 37% e 77% para mAP50:95 e F1-*score*. Uma das razões para as baixas métricas encontradas pode ser a dificuldade que a versão utilizada do YOLO enfrenta ao reconhecer objetos pequenos. Isso indica que o modelo padrão não é ideal para essa tarefa específica.

Apesar dessas limitações, quando utilizado em um ambiente de execução em tempo real, o modelo treinado ainda poderá reconhecer com velocidade a maior parte das doenças de maneira satisfatória, apesar do baixo limiar de confiança.

Os resultados alcançados com essa pesquisa mostram que a versão padrão da YOLOv8s não se mostra a ferramenta mais apropriada para a tarefa de detecção das doenças mancha-alvo e mancha olho-de-rã na cultura da soja, ressalvado casos em que se deseja utilizar um ambiente de execução em tempo real.

Caso seja de interesse do leitor, o modelo treinado e o conjunto de dados podem ser acessados através dos links disponíveis nos apêndices.

7 TRABALHOS FUTUROS

A fim de aprimorar a pesquisa seria interessante a utilização das versões futuras da YOLO, no momento em que essa pesquisa foi escrita estavam perto do seu lançamento a YOLO 9 e YOLO 10. Também seria interessante testes com diferentes tipo de aumentos artificiais, utilizando o *augmentation mosaic* que é técnica no qual uma única imagem é dividida em quadrantes que são aleatoriamente substituídos por partes de outras imagens do conjunto de dados. Isso cria novas imagens sintéticas que ajudam a aumentar a diversidade e a complexidade dos dados de treinamento, melhorando assim a capacidade do modelo de generalizar para diferentes condições e variações encontradas na aplicação real.

REFERÊNCIAS BIBLIOGRÁFICAS

AGGARWAL, A. **YOLO Explained**. 2020, Disponível em: <https://medium.com/analytics-vidhya/yolo-explained-5b6f4564f31>. Acesso em: 22 abr. 2024.

Albumentations Documentation – Why Albumentations. Disponível em: <https://albumentations.ai/docs/introduction/why_albumentations/>.

ALVES, G. **Detecção De Objetos Com YOLO – Uma Abordagem Moderna**. Disponível em: <https://iaexpert.academy/2020/10/13/deteccao-de-objetos-com-yolo-uma-abordagem-moderna/?doing_wp_cron=1716882331.4762709140777587890625>. Acesso em: 28 maio. 2024.

BALARDIN, R. S; BLUM, L. E. B; TORMEN, N. R. Alvo na mira. **Cultivar**, Pelotas-RS, v. XVIII, n. 223, p. 24-26, dez. 2017/jan. 2018. Disponível em: http://www.aprender.posse.ueg.br:8081/jspui/bitstream/123456789/164/1/Cultivar %20223.pdf

BEVERS, NOAH; SIKORA, EDWARD J.; HARDY, NATE B. (2022). **Pictures of diseased soybean leaves by category captured in field and with controlled backgrounds**: Auburn soybean disease image dataset (ASDID) [Dataset]. Dryad. https://doi.org/10.5061/dryad.41ns1rnj3

BEVERS, N.; SIKORA, E. J.; HARDY, N. B. Soybean disease identification using original field images and transfer learning with convolutional neural networks. **Computers and Electronics in Agriculture**, v. 203, p. 107449, 1 dez. 2022.

BRASIL. **Exportações do agronegócio fecham 2023 com US$ 166,55 bilhões em vendas**. Disponível em: <https://agenciagov.ebc.com.br/noticias/202401/exportacoes-do-agronegocio-fecham-2023-com-us-166-55-bilhoes-em-vendas#:~:text=Dezembro%2F2023>. Acesso em: 27 maio. 2024.

DONG CONG TRINH et al. Alpha-EIOU-YOLOv8: an Improved Algorithm for Rice Leaf Disease Detection. **AgriEngineering**, v. 6, n. 1, p. 302–317, 4 fev. 2024.

ELGENDY, M. **Deep learning for vision systems**. [s.l.] Shelter Island, Ny Manning Publications Co, 2020.

GODOY, C. V. et al. **Eficiência De Fungicidas Para O Controle Da mancha-alvo, Corynespora cassiicola, Na Cultura Da soja, Na Safra 2022/2023: Resultados Sumarizados Dos Ensaios Cooperativos**. [s.l: s.n.]. Disponível em: <https://ainfo.cnptia.embrapa.br/digital/bitstream/doc/1154756/1/Circ-Tec-194.pdf>. Acesso em: 27 maio. 2024.

HAYKIN, S. S. **Neural networks and learning machines**. New York: Prentice Hall/Pearson, 2009.

HENNING, A. A. et al. **5ª edição**. [s.l: s.n.]. Disponível em: <https://ainfo.cnptia.embrapa.br/digital/bitstream/item/105942/1/Doc256-OL.pdf>.

IWASA, Y. et al. Automatic Segmentation of Pancreatic Tumors Using Deep Learning on a Video Image of Contrast-Enhanced Endoscopic Ultrasound. **Journal of Clinical Medicine**, v. 10, n. 16, p. 3589, 1 jan. 2021.

JHONATAN PAULO BARRO et al. Frogeye Leaf Spot Caused by Cercospora sojina: a Review. **Tropical Plant Pathology**, v. 48, n. 4, p. 363–374, 15 jun. 2023.

JOCHER, G. **YOLOv8 Documentation**. Disponível em: <https://docs.ultralytics.com/>.

MASTERS, D.; LUSCHI, C. Revisiting Small Batch Training for Deep Neural Networks. **arXiv:1804.07612 [cs, stat]**, 20 abr. 2018.

RAJPUT, A. S.; SHUKLA, S.; THAKUR, S. S. SoyNet: Indian Soybean Image dataset with quality images captured from the agriculture field (healthy and disease Images). **data.mendeley.com**, v. 2, 2 jun. 2023.

REDMON, J. et al. **You Only Look Once: Unified, Real-Time Object Detection**. [s.l: s.n.], 2016. Disponível em: <https://arxiv.org/pdf/1506.02640>. Acesso em: 21 abr. 2024.

RODRIGUES, V. **Métricas de Avaliação: acurácia, precisão, recall… quais as diferenças?,** 2019**.** Disponível em: <https://vitorborbarodrigues.medium.com/m%C3%A9tricas-de-avalia%C3%A7%C3%A3o-acur%C3%A1cia-precis%C3%A3o-recall-quais-as-diferen%C3%A7as-c8f05e0a513c>. Acesso em: 22 maio. 2024.

ROSEBROCK, A. **Intersection over Union (IoU) for object detection**. Disponível em: <https://pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/>. Acesso em: 22 maio. 2024.

ROSEBROCK, A. Deep Learning for Computer Vision with Python. 1. ed. [s.l.]Philadelphia, Pa: Adrian Rosebrock, Pyimagesearch.com, 2017.

SHAH, D. **Mean Average Precision (mAP) Explained: Everything You Need to Know**. Disponível em: <https://www.v7labs.com/blog/mean-average-precision>. Acesso em: 24 maio. 2024.

SHAH, D. **Intersection over Union (IoU): Definition, Calculation, Code**. Disponível em: <https://www.v7labs.com/blog/intersection-over-union-guide>. Acesso em: 23 maio. 2024.

SUN, D. et al. Efficient Tobacco Pest Detection in Complex Environments Using an Enhanced YOLOv8 Model. **Agriculture**, v. 14, n. 3, p. 353–353, 22 fev. 2024.

ULTRALYTICS. **YOLO Métricas de desempenho**. Disponível em: <https://docs.ultralytics.com/pt/guides/yolo-performance-metrics/#class-wise-metrics>. Acesso em: 25 maio. 2024.

VALDES, C.; GILLESPIE, J.; DOHLMAN, E. N. Soybean production, marketing costs, and export competitiveness in Brazil and the United States. **Economic Research Service**, v. 262, 1 dez. 2023.

WANG, X.; LIU, J. Vegetable disease detection using an improved YOLOv8 algorithm in the greenhouse plant environment. **Scientific Reports**, v. 14, n. 1, 21 fev. 2024.

APÊNDICE

LINK PARA ACESSAR O *DATASET*

(https://app.roboflow.com/tcc-ei06d/soy-leaf-disease/overview)

LINK PARA ACESSAR OS RESULTADOS

(https://github.com/jnaps12/TCCII)