UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO

UNIVERSIDADE ESTADUAL DO RIO GRANDE DO NORTE

PROGRAMA DE POS-GRADUAÇÃO EM CIENCIAS DA COMPUTAÇÃO

ALFREDO FELIPE LOPES NETO

Aplicação de *Deep Learning* em dois Estágios para Classificação de Contexto de Patógenos em Plantas

MOSSORÓ

2023

ALFREDO FELIPE LOPES NETO

Aplicação de *Deep Learning* em dois Estágios para Classificação de Contexto de Patógenos em Plantas

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação - associação ampla entre a Universidade do Estado do Rio Grande do Norte e a Universidade Federal Rural do Semi-Árido, para a obtenção do título de Mestre em Ciência da Computação.

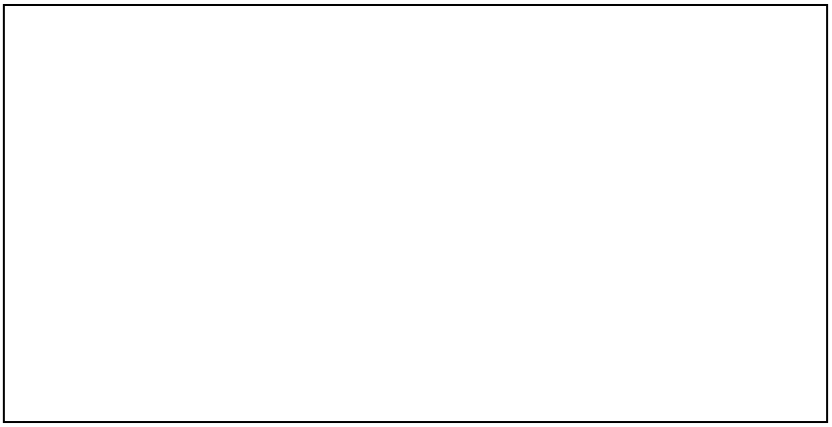
Orientador: Prof. Dr. Silvio Roberto Fernandes de Araújo - UFERSA

Coorientador: Prof. Dr. Araken de Medeiros Santos - UFERSA

MOSSORÓ

2023

Dados Internacionais de Catalogação na Publicação (CIP)
Biblioteca Central Orlando Teixeira (BCOT)
Setor de Informação e Referência (SIR)



Bibliotecário-Documentalista
Nome do profissional, Bib. Me. (CRB-15/10.000)

ALFREDO FELIPE LOPES NETO

Aplicação de *Deep Learning* em dois Estágios para Classificação de Contexto de Patógenos em Plantas

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação como requisito para obtenção do título de Mestre em Ciência da Computação.

Aprovada em:

BANCA EXAMINADORA

_____

Prof. Dr. Silvio Roberto Fernandes de Araújo - UFERSA

Presidente

_____

Prof. Dr. Araken de Medeiros Santos - UFERSA

Membro Examinador

_____

Prof. Dr. Daniel Sabino Amorim de Araújo - UFRN

Membro Examinador Externo

_____

Prof. Dr. Sebastião Emidio Alves Filho - UFERSA

Membro Examinador Interno

Dedico este trabalho a minha avó Lindalva Maria da Silva (*In Memoriam*). Quem me ensinou a enxergar beleza na simplicidade da vida.

AGRADECIMENTOS

Meus agradecimentos a Deus, a quem sempre peço sabedoria e inteligência para tomada de decisões.

Agradeço a minha família, mas principalmente a Jordana minha companheira, quem foi paciente, compreensiva e me deu apoio em todo o processo.

Muito obrigado ao meu orientador Silvio Araújo e ao meu coorientador Araken de Medeiros pelos ensinamentos e paciência. Quem para mim se tornaram uma referência profissional, pessoal e por terem me dado uma base acadêmica que me fez chegar aonde cheguei.

Agradeço ao Conselho Nacional de Pesquisa (CNPq) pelo apoio financeiro para realização deste trabalho por meio do projeto Implementação de Ferramentas Tecnológicas na Produção de Alimentos no Semiárido Potiguar, do edital nº18/2020 – CAPES. Projeto do qual conta com a participação dos Programas de Pós-Graduação em Manejo de Solo e Água (UFERSA), Produção Animal (UFERSA-UFRN) e Ciência da Computação (UFERSA-UERN), visando a colaboração mútua no desenvolvimento de estratégias inovadoras para resolução de problemas, com base na aproximação com o setor produtivo do semiárido para a produção de pesquisas transversais.

“O importante não é o que fazem de nós, mas o que nós fazemos daquilo que fazem de nós.”

Jean-Paul Sartre

RESUMO

Deep learning pertence ao domínio de aprendizado de máquina que tem alcançado notáveis êxitos em várias atividades, em comparação com outras abordagens desse domínio. Dentre essas atividades, a detecção e classificação de objetos em imagens destacam-se como exemplos notáveis. Comumente, aplicações desse tipo, segundo a literatura, utilizam uma única rede neural convolucional para realizar o processo de detecção, delimitando as regiões de interesse que contém o objeto, bem como determinando a classificação desse objeto em uma classe. Com o intuito de aprimorar os resultados na detecção e classificação de imagens referentes a plantas com folhas apresentando, ou não, patógenos, esse trabalho propõe explorar a construção de um sistema que utiliza redes neurais convolucionais, composto por dois estágios, um para efetuar a detecção de regiões de interesse na imagem e o outro para determinar a classificação dessas regiões, por intermédio de aprendizado supervisionado, e ao final, com base nos dados obtidos, determinar uma classificação por contexto, ou seja, classificar se a planta presente na imagem está doente ou não, com base no contexto das folhas detectadas. Para validar esse método, foram realizados experimentos utilizando uma base de dados composta por imagens de plantas de maçã, pimentão, milho. batata, tomate e uva. Os resultados indicam que a abordagem do sistema utilizando dois estágios apresentam um desfecho satisfatório, no primeiro estágio para detecção uma precisão de quase 90% e um *mAP* de 91,27%, com resultados igualmente importantes nas métricas, *recall* e *IoU*, e no segundo estágio para classificação apresentando uma média de precisão de 93%, com resultados igualmente importantes nas métricas *recall* e *F1-Score*. Por fim, o sistema como um todo, em decorrência dos resultados dos estágios, se mostra eficiente para realizar a classificação por contexto, de igual modo, esta abordagem permite uma maior flexibilidade na seleção das redes de detecção e classificação para adequar o modelo a específicos cenários.

Palavras-chave: *deep learning*, redes neurais convolucionais, aprendizado supervisionado, detecção de doenças em plantas, classificação por contexto

ABSTRACT

Deep learning belongs to the machine learning domain that has achieved remarkable successes in various activities compared to other approaches in this domain. Among these activities, the detection and classification of objects in images stand out as notable examples. Commonly, applications of this type, according to the literature, use a single convolutional neural network to carry out the detection process, delimiting the regions of interest that contain the object, as well as determining the classification of this object into a class. In order to improve the results in the detection and classification of images referring to plants with leaves showing, or not, pathogens, this work proposes to explore the construction of a system that uses convolutional neural networks, composed of two stages, one to carry out the detection of regions of interest in the image and the other to determine the classification of these regions, through supervised learning, and in the end, based on the data obtained, determine a classification by context, that is, classify whether the plant present in the image is sick or not, based on the context of the detected leaves. To validate this method, experiments were carried out using a database composed of images of apple, pepper and corn plants. potato, tomato and grape. The results indicate that the system approach using two stages presents a satisfactory outcome, in the first stage for detection an accuracy of almost 90% and an mAP of 91.27%, with equally important results in the recall and IoU metrics, and in the second stage for classification presenting an average accuracy of 93%, with equally important results in the recall and F1-Score metrics. Finally, the system as a whole, because of the results of the stages, proves to be efficient in carrying out classification by context. Likewise, this approach allows greater flexibility in the selection of detection and classification networks to adapt the model to specific scenarios.

Keywords: deep learning, convolutional neural networks, supervised learning, plant disease detection, context classification

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

SUMÁRIO

1 INTRODUÇÃO

Agricultura 4.0 tem como principal característica o uso de tecnologias digitais baseadas em conectividade para automação e troca de dados em processos utilizados no setor agrário, utilizando máquinas e implementos agrícolas conectados na internet. Tais tecnologias exercem um papel importante no enfrentamento de desafios 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.

Deste modo, os agricultores utilizam esses dados com o objetivo de se tornarem mais produtivos e sustentáveis. Destarte, a utilização de aplicativos para coletar dados nas plantações e controle de pragas na agricultura 4.0 para a tomada de decisão mais assertiva pode ser feita mesmo em grande escala de produção, desde grandes fazendas a pequenas.

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 e diminuição de gastos com a plantação.

Segundo o Instituto Mato-grossense de Economia Agropecuária (IMEA), o custo médio passou de R$ 779,84 por hectare na safra de 2019/2020 para R$ 937,48 na safra 2021/2022, e existe a estimativa de aumento para R$ 1437,80 por hectare na próxima safra 2022/2023 (IMEA, 2022). Igualmente, segundo o Sindicato Nacional da Indústria de Produtos Para Defesa Vegetal (SINDIVEG), os gastos do produtor rural brasileiro no primeiro semestre de 2021 cresceram 13% em relação ao mesmo período de 2020, passando de R$ 25,188 bilhões para R$ 28,462 bilhões (SINDIVEG, 2021). Assim, o

problema ambiental presentes com os fitopatógenos afeta diretamente a sustentabilidade e produção da agricultura e deve ser encarado como item central, confirmando a importância dos estudos nessa área, cuja criação de estratégias agroecológicas e tecnologia de proteção as plantas (LEFÈVRE et al., 2020).

Nesse contexto, 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; 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. Tendo em vista o combate desses 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.

Em outras palavras, é possível utilizar uma solução computacional como redes neurais artificias para explorar todas as informações coletadas. Para que, a partir de um conjunto de dados, e aprendendo através de treinamento, generalizar assim, a informação e dando respostas coerentes para dados ainda não conhecidos. (KOVÁCS, 2006)

1.1 Motivação

Doenças presentes em plantações são responsáveis pela perda de milhões de toneladas de alimentos anualmente, sendo um dos motivos de prejuízo econômico. Estima-se que às doenças as quais atacam especificamente espécies vegetais, são as grandes responsáveis pelo detrimento de boa parte das colheitas em plantações (ZHANG et al., 2022). Assim, a inspeção de plantações se faz fundamental para o combate de doenças, permitindo assim, um tratamento rápido e eficiente.

É sabido que a maioria das doenças presentes em plantas ocasiona alguma manifestação visível, principalmente nas suas folhagens (BARBEDO, 2018a). Sendo os métodos convencionais de identificação de patógenos em folhas, normalmente, envolvendo atuação de especialistas humanos para identificação ou análises laboratoriais para identificação, processos que consomem tempo e recursos (BOCK et al., 2010).

Assim sendo, em regiões onde a presença de profissionais capacitados ou de laboratórios estruturados, ou em situações que demandam rápida avaliação, a verificação de imagens utilizando métodos computacionais apresenta-se como uma possível solução viável.

Nos últimos anos, o crescente número de modelos de aprendizado de máquinas baseados em aprendizado profundo, em particular as redes neurais convolucionais, tendo sido utilizados com sucesso em tarefas de visão computacional, se mostrando, assim, ser uma eficiente técnica para extrair atributos baseado nos dados coletados (AHMAD et al., 2021). Com os avanços nessa área, devido principalmente ao desenvolvimento de novos algoritmos e *hardware* mais potente, permitindo a utilização de um volume maior de dados e treinamento de redes neurais artificiais mais complexas de forma mais eficazes.

Ademais, existem numerosas arquiteturas de redes neurais projetadas para superar desafios de visão computacional, especificamente relacionadas a tarefas como detecção de objetos e classificação de imagens, exibindo notável eficácia em relação com os últimos avanços documentados na literatura. Além disso, estas arquiteturas conferem benefícios em relação às metodologias anteriormente utilizadas para monitorizar as plantações, consequentemente, estas novas abordagens permitem um intervalo de tempo reduzido entre a coleta de dados, análise e tomada de decisões, mitigando potenciais repercussões para as plantações.

1.2 Problemática

Na agricultura brasileira, o combate às doenças de plantas que se manifestam nas

folhas apresenta grandes desafios. Uma das questões principais é a importância crítica da identificação e intervenção oportunas para prevenir a propagação de doenças e minimizar as perdas de colheitas.

Assim, as folhas costumam ser as primeiras partes da planta a exibir sintomas visíveis de doenças, tornando-as indicadores cruciais para a detecção precoce. A identificação imediata permite que os agricultores tomem as medidas apropriadas, como a implementação de tratamentos direcionados ou a remoção de plantas infectadas, para conter a doença antes que ela se espalhe para toda a safra ou campos vizinhos.

A identificação manual de doenças de plantas em folhas tem sido uma abordagem tradicional, mas apresenta limitações. Em primeiro lugar, o processo pode ser demorado e trabalhoso, particularmente para operações agrícolas de grande escala. Como o setor agrícola no Brasil envolve extensas áreas de produção, a inspeção manual de cada planta torna-se inviável e pode levar a atrasos na detecção de focos de doenças. Em segundo lugar, a identificação precisa de doenças requer experiência em fitopatologia, que pode não estar prontamente disponível para todos os agricultores, especialmente em regiões rurais remotas.

Embora, alguns trabalhos abordem essa questão de forma direta, categorizando uma condição a partir da imagem de uma única folha, classificar uma folha individual não indica com precisão a saúde geral da planta ou da folha específica. Consequentemente, o problema a ser abordado é categorizar a saúde das folhas da planta num determinado contexto, ou seja, abrangendo numerosas folhas situadas próximas do seu ambiente natural.

Apesar de encontrarmos vários desafios neste esforço, esperamos que a classificação baseada no contexto se revelará mais vantajosa e precisa no que diz respeito ao diagnóstico de doenças ou à avaliação da proliferação em comparação com a abordagem de classificação "convencional", que envolve a análise de folhas isoladas desprovidas de informação contextual.

1.3 Objetivos

Esse trabalho, busca alcançar os objetivos subdivididos em objetivo geral e objetivos específicos, conforme detalhados a seguir.

1.3.1 Objetivo Geral

O objetivo geral deste trabalho é: **desenvolvimento de um sistema computacional que realiza o reconhecimento visual e classificação do contexto de infecções de fitopatógenos.**

1.3.2 Objetivos Específicos

Para o desenvolvimento do sistema de classificação de contexto de doenças de plantas, será necessário a construção de dois subsistemas, os quais são listados a seguir nos objetivos específicos.

- Desenvolver um subsistema que realiza detecção de objetos (folhas de plantas) a partir de uma imagem digital e as separa em imagens individuais para o subsistema seguinte.
- Desenvolver um subsistema classificador de doenças em folhas de diversas espécies de plantas, a partir das imagens individuais advindas do subsistema detector.
- Desenvolver um sistema de classificação de contexto, o qual integra os dois subsistemas anteriores e é capaz de informar o percentual de infestação de uma fitopatologia no contexto da imagem fornecida na entrada do sistema.

1.4 Estrutura do Documento

Este trabalho está organizado conforme a seguinte estrutura:

- O capítulo 2, é apresentado o referencial teórico, contendo uma revisão sobre os principais assuntos abordados neste trabalho.
- O capítulo 3 apresenta trabalhos da literatura que estão relacionados ao problema de pesquisa deste trabalho.
- O capítulo 4 descreve o detalhamento da metodologia utilizada e implementação da solução proposta.
- Os resultados alcançados nos experimentos são apresentados no capítulo 5.
- Por fim, o capítulo 6, que contém as considerações finais, bem como as propostas para trabalhos futuros.

2 REFERENCIAL TEÓRICO

O presente capítulo discorre conceitos essenciais para o desenvolvimento deste trabalho. Inicialmente, retrata um retrospecto em relação ao surgimento e evolução das redes neurais artificiais, para em seguida apresentar o modelo de rede neural convolucional e caracterizando como uma técnica de aprendizado profundo. Enfim, realizando uma descrição de modelos e *framework* que utilizam redes neurais artificiais, possuindo como objetivo, a tarefa de detectar ou classificar objetos em imagens.

2.1 Redes Neurais

Tendo como base as caraterísticas presentes nos neurônios do sistema nervoso dos animais (MCCULLOCH; PITTS, 1943) recomendaram o primeiro modelo matemático que representava um neurônio, funcionando utilizando portas logicas e era capaz, por meio de combinações, de retratar funções booleanas. Contudo, se fazia necessário o ajuste manual dos pesos do modelo para o fornecimento dos resultados esperados como saída. Rosenblatt (1958) propôs uma evolução para esse modelo, criando o *Perceptron*, classificador binário capaz de aprender e ajustar os seus pesos com base nos exemplos dos objetos para cada classe fornecida como amostra do treinamento, fazendo assim, a utilização de algoritmo para a atualização de tais pesos.

Além disso, essa nova arquitetura apresentava uma incapacidade para classificar padrões não lineares separáveis, por exemplo, a saída de uma porta XOR, assim dentre outros motivos, fomentou um período de desinteresse e descrença em relação as redes neurais artificiais durante uma década.

Ainda que já existisse a ideia de organizar os neurônios em diversas camadas, o treinamento de tal proposta não poderia ser executado de maneira trivial. O interesse

nessa área somente renasceu com o trabalho (RUMELHART; HINTON; WILLIAMS, 1986), onde foi apresentado o algoritmo de *Backpropagation*, apontando um método para o treinamento de redes neurais artificiais formadas por várias camadas de neurônios, denominadas de redes *multi-layer Perceptron* (MLP). Tal trabalho permitiu que estas redes fossem capazes de solucionar problemas em conjuntos não linearmente separáveis para classificação.

Doravante, os modelos convolucionais começaram a evoluir, sendo a rede LeNet-5 (LECUN et al., 1998) uma das primeiras a ganhar notoriedade, sendo aplicada com sucesso para o reconhecimento de dígitos manuscritos em imagens, usando o *dataset* MNIST.

Porém, somente em meados dos anos 2000 o tema *deep learning* ganhou relevância, devido a evolução dos *hardwares* e a criação das chamadas *deep belief networks* (HINTON; OSINDERO; TEH, 2006), redes neurais profundas que apresentavam a possibilidade de realizarem o treinamento camada por camada.

Já em 2012 o desafio ILSVRC (*ImageNet Large-Scale Visual Recognition Challenge*) foi vencido pela rede neural convolucional AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) que apresentou uma ampla vantagem em relação ao segundo colocado, consolidando a eficiência das redes convolucionais como arquitetura de aprendizado profundo, ao realizar classificação de imagens (PONTI et al., 2017). Ainda, a arquitetura da AlexNet apresentado trouxe ideias que atualmente são indispensáveis para a construção e treinamento de novas arquitetura para aprendizado profundo, por exemplo, a utilização de unidades de processamento gráfico (GPU) para o treinamento das redes.

Em seguida, os avanços tecnológicos na fabricação de GPUs, cada vez com mais capacidades de processamento, permitiu o desenvolvimento de redes convolucionais mais profundas, como por exemplo, a GoogLeNet (SZEGEDY et al., 2015) e a ResNet (HE et al., 2016). Na Tabela 1 encontra-se em resumo algumas redes convolucionais que se destacaram na tarefa de classificação.

Arquitetura	Parâmetros	Descrição

LeNet (LECUN et al., 1998)	60K	Originalmente utilizada para o reconhecimento de dígito, sendo uma das primeiras CNN a obter sucesso, com uma menor quantidade de parâmetros em relação a outros modelos.
AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012)	60M	Propagou a utilização das CNN em visão computacional, sendo considerada o primeiro modelo de CNN moderna, utilizando *dropout* evitando assim o sobreajuste e ReLU como função de ativação.
ZFNet (ZEILER; FERGUS, 2014)	42,6M	Semelhante ao modelo AlexNet, porém com menos parâmetros. Apresentou a utilização de rede deconvolucional para visualização. Vencedor do ILSVRC 2013.
VGGNet (SIMONYAN; ZISSERMAN, 2014)	133M a 144M	Apresenta múltiplos filtros 3 x 3 em sequência no lugar de filtros maiores (exemplo 5 x 5 ou 7 x 7). Variações do modelo como VGG16 e VGG19 mostram que profundidade da rede influencia no desempenho dela.
GoogLeNet (SZEGEDY et al., 2015)	7M	Modelo conhecido por possuir 100 camadas no total em sua arquitetura, também conhecido como “Inception”.
ResNet (HE et al., 2016)	25,6M a 60,2M	Variacões desse modelo podem atingir até 1200 camadas, se destaca na utilização do conceito de blocos residuais na construção de rede profundas.
MobileNet (HOWARD et al., 2017)	4,2M	Desenvolvida visando funcionamento em dispositivos móveis, apresenta considerável redução do número de parâmentros e de operações, ainda atingindo acurácia próxima aos demais modelos como a VGG.

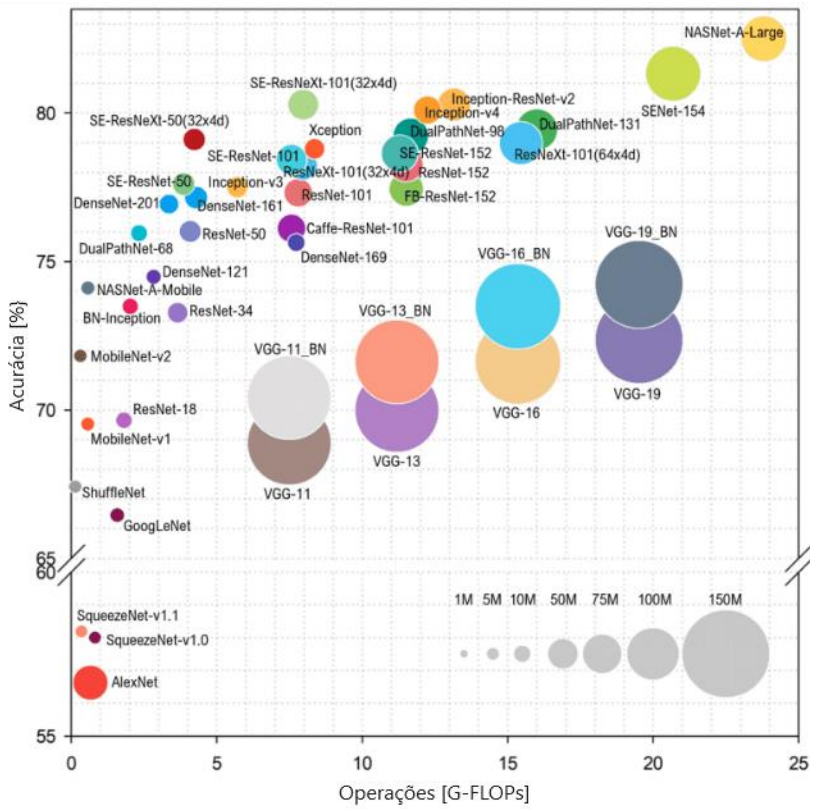
DenseNet (HUANG et al., 2017)	7,1M	Com um número reduzido de parâmetros, apresenta conexões densas entre uma camada e todas as anteriores, procurando evitar o problema de *vanishing gradiente*, assim facilitando que os atributos se propagem pela rede.
EfficientNet-B7 (TAN; LE, 2019)	66M	Arquitetura escalonável de uma rede convolucional, ajustando a largura da rede, profundidade, tipos e tamanhos de kernel de convolução, para trabalhar com modelos supergrandes.

Tabela 1 – Arquiteturas de Redes convolucionais profundas

Na Figura 1 podemos observar a relação complexidade computacional das arquiteturas e acurácia com base no conjunto de validação ImageNet-1k, o tamanho do círculo corresponde à complexidade da arquitetura.

Ademais, é perceptível que não há relação entre complexidade computacional e acurácia, por exemplo, o SENet-154 precisa de cerca de 3x o número de operações necessárias para o SE-ResNeXt-101, embora tenha quase a mesma acurácia. Além disso, não existe relação entre a complexidade do modelo e a precisão do reconhecimento: por exemplo, o VGG-13 tem um nível de complexidade do modelo (tamanho do círculo) muito mais elevado do que o ResNet-18, embora tenha quase a mesma acurácia. Assim, demonstrando que o desempenho de uma rede convolucional pode variar em relação ao número de parâmetros presentes na arquitetura.

Figura 1 – Arquiteturas de redes convolucionais em relação a acurácia e complexidade computacional



Fonte: adaptado de (BIANCO et al., 2018)

Dentre todas as arquiteturas citadas vale destacar duas, que serão utilizadas no decorrer desse trabalho, ResNet e EfficientNet.

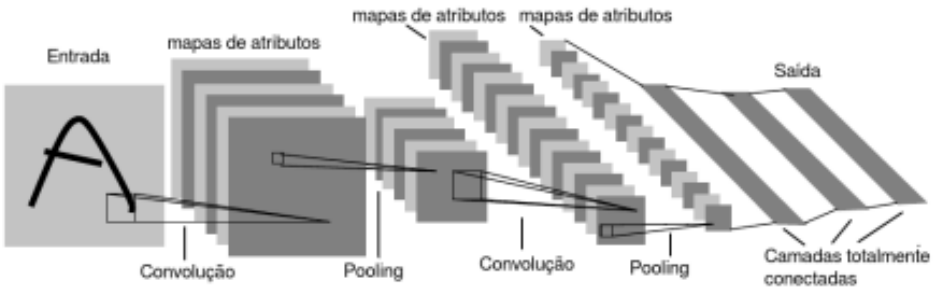
A ResNet-50 representa um avanço significativo no aprendizado profundo para tarefas de visão computacional. Uma de suas principais vantagens reside na introdução de conexões de salto ou conexões residuais, essas conexões permitem que a rede passe informações de forma eficaz através de múltiplas camadas, mitigando o problema de degradação encontrado em redes muito profundas. A incorporação de blocos residuais aumenta a capacidade da rede de aprender padrões e representações, levando a uma maior precisão e convergência durante o treinamento. Isto permite o treinamento de redes neurais profundas, superando limitações e facilitando a exploração de hierarquias de características mais complexas e expressivas.

Já a EfficientNet-B7 dimensiona como uma rede convolucional buscando equilibrar cuidadosamente a largura, a profundidade e a resolução da rede para alcançar melhor precisão e eficiência. Logo, o método de escalonamento é simples e altamente eficaz, que permite ampliar facilmente a estrutura de uma rede convolucional com linhas de base para quaisquer restrições de recursos, mantendo a eficiência da arquitetura.

Para elucidar a estrutura de uma rede convolucional, como ilustrada na Figura 2, temos, camadas de convolução, além de camadas de agrupamento (também conhecidas como *pooling*) e camadas conectadas totalmente (neurônios da camada atual conectados a todos os neurônios da camada anterior).

As camadas de convolução, conforme mencionado anteriormente, possuem filtros para a extração de características dos dados de entrada, colocando essas informações nos canais de saída, chamados mapas de atributos. Para garantir que a dimensionalidade de tais mapas não fique muito elevada, usualmente são aplicadas camadas de *Pooling* após cada camada de convolução. A função dessa camada de *Pooling* e agrupar os valores de entrada contidos e resumir em um único valor, normalmente sendo o maior (denominando assim o *maxpooling*). Por fim, para realizar a classificação, são adicionadas uma ou mais camadas, totalmente conectadas, realizando assim a distinção das diferentes classes empregadas.

Figura 2 – Modelo de uma rede convolucional.



Fonte: (LECUN et al., 1998)

Diversas camadas de convolução, seguidas, de camadas de *Pooling* são organizadas sequencialmente, de modo que as entradas das camadas subsequentes são o mapa de atributos fornecido pela camada anterior.

Assim, o aprendizado segue a seguinte lógica: quanto mais profunda a rede a camada adquiri filtros que extraem características mais complexas e abstratas que a anterior. De forma automática, esse processo hierárquico e incremental de extração de atributos vai adicionando cada vez mais informações abstratas imediatamente dos dados de entrada, excluindo o processo de *feature engineering*, isso, confere às redes neurais convolucionais sua principal característica que as defini com modelos *deep learning*

(PONTI et al., 2017).

O processo de aprendizado busca minimizar o resultado de uma função de custo diferençável, que está associada a rede. Tal função apresenta um valor de distância entre o valor fornecido pela rede ao final e o valor real fornecido pelos rótulos presentes nas amostras de treinamento, isso para os casos do aprendizado supervisionado.

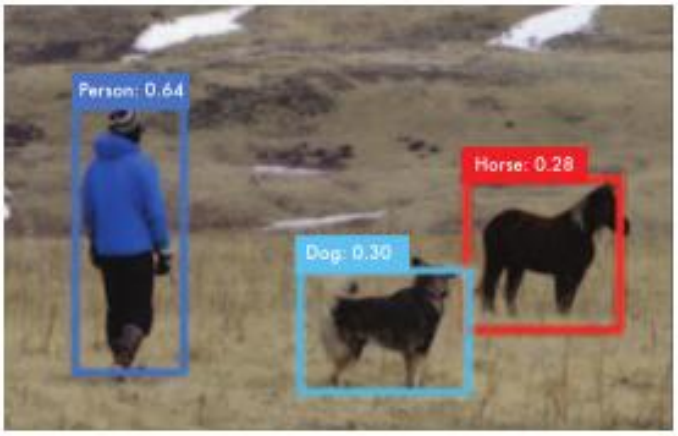
Empregando ferramentas de cálculo, como derivadas parciais e a regra da cadeia, torna-se possível a obtenção dos valores presentes nos gradientes da função de custo em relação aos pesos dos neurônios presentes da rede. Por meio do método de *Backpropagation* os valores dos pesos são atualizados a partir do valor retornado da função de custo da última camada da rede, passando das camadas finais para as camadas iniciais, realizando o cálculo da regra da cadeia visando obter a contribuição que cada peso teve no valor encontrado.

Em seguida as iterações de treinamento, considera que a função de custo tenha o seu valor diminuindo gradualmente e, de maneira esperada, que o erro apresentado pela rede neural também diminua. Dessa forma, de maneira automática, as CNNs aprendem quais os valores necessários para os filtros realizarem a extração das características presentes nas imagens.

2.2 Redes neurais para detecção de objetos

Detecção de objetos é um problema comum na visão computacional: temos uma imagem, deve-se identificar um ou mais sub-regiões, normalmente utilizando retângulos, onde estão presentes os objetos de interesse, tal como classificar tais objetos, exemplo ilustrado na Figura 3. Para solucionar esse tipo de problema a rede deve fornecer na sua saída as coordenadas espaciais da sub-região onde se encontra o objeto, assim como a probabilidade de classificação desse objeto. Logo, para solução desse problema, é necessário a realização de duas tarefas: i) regressão de valores das coordenadas das sub-regiões; ii) classificação das sub-regiões dentre as possíveis classes existentes (ZHAO et al., 2019).

Figura 3 – Exemplo de imagem onde está presente a detecção dos objetos e sua classificação por classes.



Fonte: (REDMON et al., 2015)

Assim, a literatura apresentar exemplos consolidados na detecção de objetos, nesse contexto, algumas dessas redes serão descritas nas seções a seguir.

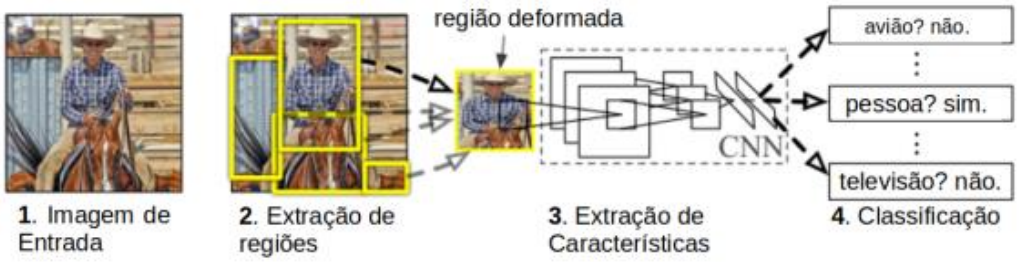
2.2.1 R-CNN, Fast R-CNN e Faster R-CNN

Uijlings (2013) apresenta as R-CNNs (*regions with CNN features*), tendo como uma de suas vantagens a capacidade de localizar e classificar objetos com alta precisão. As R-CNNs introduz uma proposta que um mecanismo que seleciona regiões potenciais onde os objetos podem se encontrar na imagem, submetendo posteriormente essas regiões a uma rede neural convolucional para extração e classificação de características. Este processo de dois estágios aumenta a precisão, concentrando a computação em regiões que provavelmente contêm objetos, na Figura 4 temos um exemplo da sua arquitetura.

Girshick (2014) utilizou esse modelo e propôs um processo de busca seletiva, utilizando um conjunto inicial com cerca de 2000 regiões de interesse, tais regiões foram redimensionadas e fornecidas com dados de entrada para uma rede convolucional, em

seguida, utilizando uma Máquina de Vetor de Suporte (SVM, do inglês *Support Vector Machine*) extrai um vetor de caraterísticas para realização da classificação de cada região. Tal procedimento era usualmente utilizado, posto que, a otimização das SVMs era um problema muito estudado e apresentando melhorias teóricas de convergência (MELLO; PONTI, [s.d.]).

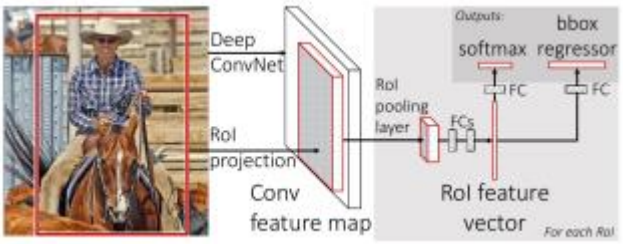
Figura 4 – Arquitetura de uma R-CNN



Fonte: Adaptada de (GIRSHICK et al., 2014)

As R-CNNs apresentam, como um dos principais problemas, o alto tempo de processamento, como também, o processo de extração das regiões de interesse, denominado de busca seletiva. Este procedimento é independente da rede neural, não havendo aprendizado em relação as previsões das regiões de interesse, já que somente é treinado o processo de classificação. Buscando contornar isso, Girshick (2015) apresentou uma atualização da R-CNN denominada Fast R-CNN, apresentada na Figura 5. Estão presentes nessa rede as regiões de interesse que são projetadas sobre o mapa de características extraído da imagem original, saindo das camadas de convolução. Além disso, cada projeção é enviada para uma camada de *Pooling* customizada (chamada de *Rol pooling*) para realizar a extração de atributos, assim, reduzindo a região para um vetor de tamanho fixo. Em seguida, segue sendo enviado para uma sequência de camadas totalmente conectadas em série, assim dividindo em dois ramos: i) para realizar a classificação do objeto esperado; ii) realizar regressão, para refinar as coordenadas da região de interesse.

Figura 5 – Arquitetura de uma Fast R-CNN



Fonte: (GIRSHICK, 2015)

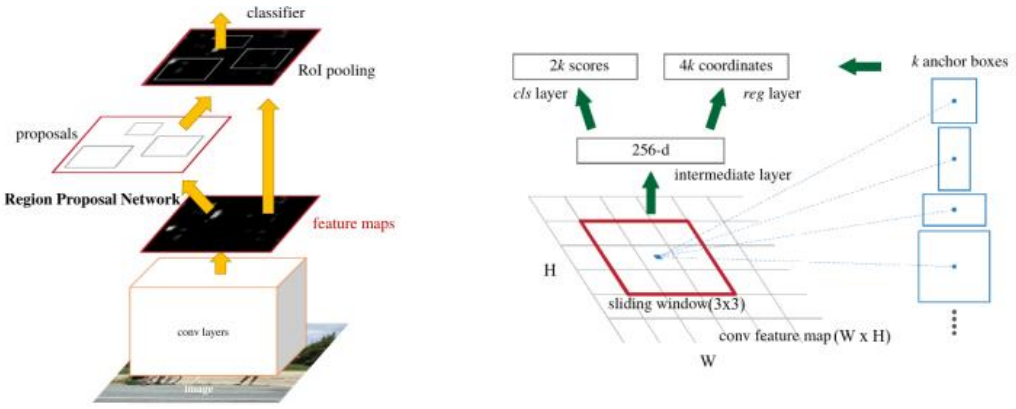
As Fast R-CNN, apesar de serem mais rápidas que as R-CNN, ainda apresentam um gargalo no processo de geração de regiões de candidatas, como a busca seletiva. Procurando solucionar esse problema, Ren (2015) introduz a rede Faster R-CNN, a qual traz a geração da região de interesse realizado pela própria rede, para só então realizar a sua classificação e regressão, executado da mesma forma que se dá na Fast R-CNN (Figura 6).

Assim, a composição da Faster R-CNN possui dois módulos: i) primeiro, apresentando uma rede completamente convolucional (FCN, do inglês *fully convolutional network*), sendo responsável por gerar as regiões de interesse, chamadas de *Region Proposal Network* (RPN); ii) segundo, desenvolvido essencialmente para classificar e aperfeiçoar as localizações das regiões de interesse fornecidas. Logo, o processo segue a seguinte ordem: i) a imagem de entrada passa inicialmente pela rede de base para a extração das características; ii) em seguida, o módulo RPN, utilizando um filtro 3 x 3, realiza a convolução sobre o mapa de atributos que apresenta uma dimensão de *W* x *H*, gerado a partir da imagem original, assim gerando então outro mapa de atributos de profundidade 256 (ou 512, a depender do modelo da implementação).

Analisando a arquitetura da Faster R-CNN da Figura 6, cada posição do mapa de entrada está associada a *k* regiões âncoras (*anchor boxes*), sendo elas responsáveis pelas proporções de tamanhos para detectar objetos diversos. Na primeira camada de convolução há dois ramos em paralelo: i) o primeiro, correspondendo a uma camada convolucional para realizar, como base na âncora, uma classificação binária, determinando se esta comtempla um objeto de interesse ou o fundo da imagem; ii) segundo, também corresponde a uma camada convolucional, mas realiza a regressão utilizando as quatro coordenadas espaciais da âncora (x_{centro}, y_{centro}, *largura* e *altura*) visando os melhores valores para igualar a posição real do objeto. Tanto o primeiro quanto o segundo ramo apresentam filtros de tamanho 1 x 1, fornecendo como saída

mapas de dimensões *W* x *H* x *2k* e *W* x *H* x *4k*, respectivamente. O mapa de características inicial, a ser utilizado no módulo RPN, pode ser obtido empregando alguma rede base que melhor se adequa a aplicação, exemplo redes mais profundas como a ResNet (HE et al., 2016), ou mais compactas, por exemplo a MobileNet (HOWARD et al., 2017).

Figura 6 – Arquitetura de uma Faster R-CNN



Fonte: Adaptada de (REN et al., 2015)

O modelo de rede Faster R-CNN possibilitou o treinamento de ponta-a-ponta de um modelo para detecção de objetos. Com isso, essas redes neurais diminuíram o tempo de processamento de maneira significativa, chegando a uma velocidade de operação para detectar de 7 *frames* por segundos (FPS) utilizando uma GPU, e apresentando resultados significativos em detecção de objetos utilizando os conjuntos PASCAL VOC 2007 e 2012 (REN et al., 2015).

As arquiteturas apresentadas até aqui para detecção de objetos são formadas por diferentes módulos que realizam tarefas diferentes, para localização e classificação. Isto significa que existe um módulo para a extração do mapa de características da imagem original e outro módulo para regressão das regiões de interesse e finalmente a classificação. No entanto, existem arquiteturas de redes neurais que desenvolvem todo o procedimento de forma completa, classificação e regressão, usando um único fluxo ao longo de toda a rede. Essa nova proposta foi chamada de YOLO (*You Only Look Once*)

por (REDMON et al., 2015).

2.2.2 Yolo

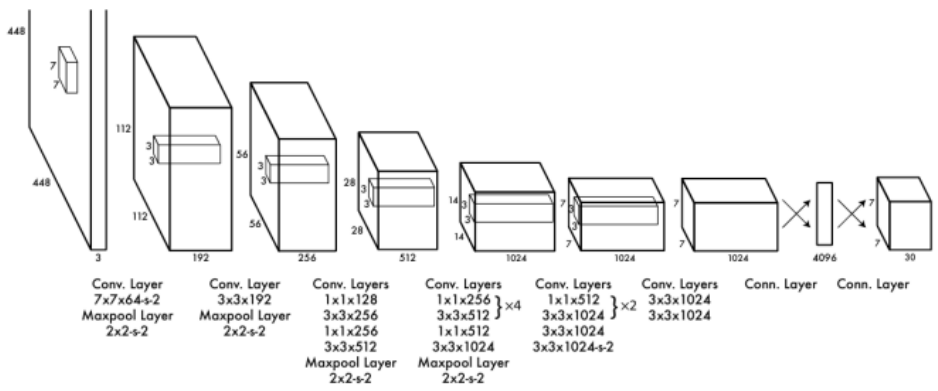
Redmon (2015) apresenta a rede Yolo e traz nessa arquitetura o conceito de dividir a imagem em uma grade de tamanho $S \times S$ e as células presentes na grade são associadas a um conjunto de *B* caixas âncoras, como na Faster R-CNN.

Assim, cada âncora e formada por um conjunto de cinco valores (x_{centro}, y_{centro}, *largura*, *altura*, *p*), este último equivale a probabilidade dessa âncora conter um objeto. Caso uma célula contenha as coordenadas do centro de um desses objetos, a mesma é responsável por realizar a detecção, usando as caixas de âncora para ajustar melhor os objetos que apresentam variadas proporções de tamanho. Ainda, a cada célula está atribuído um vetor de probabilidade referente às *P* classes de objetos presentes na imagem.

Logo, a rede apresenta uma saída correspondente a um tensor de dimensões $S \times S \times (B * 5 + P)$. A implementação do modelo usa 24 camadas convolucionais para mapeamento das características, seguidas de duas camadas totalmente conectadas.

Utilizando o conjunto de imagens PASCAL VOC, que traz P = 20 classes, usando S = 7 e B = 2 âncoras por célula, decorrendo em um tensor de saída com as dimensões $7 \times 7 \times 30$, conforme a Figura 7.

Figura 7 – Camadas da rede Yolo



Fonte: (REDMON et al., 2015)

A Yolov2, uma versão aprimorada da rede Yolo, foi proposta por Redmon (2017), trazendo melhores resultados para a detecção ao adicionar normalização de lote (*batch normalization*). Além de evoluir o mecanismo de geração das caixas âncoras e empregando uma nova rede de 19 camadas para realizar a extração de atributos.

Redmon (2018) introduz a Yolov3, que melhorou ainda mais a realização das detecções, por meio de três diferentes escalas (sendo capaz de detectar de maneira mais eficiente objetos pequenos. Ela também utiliza blocos residuais (como na rede ResNet) e um número maior de camadas de convolução, trazendo filtros de tamanhos 3×3 e 1×1, para a extração de características, totalizando 53 camadas, conforme a Figura 8.

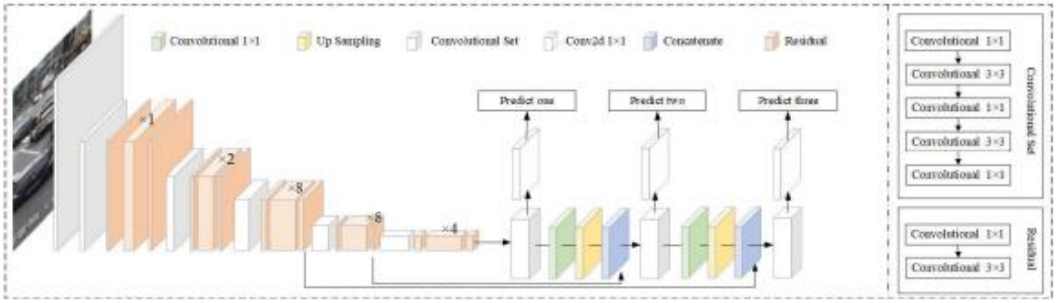
De maneira completa podemos verificar a arquitetura da rede Yolov3 na Figura 9, apresentando aprimoramentos como detecção em três escalas e a utilização de blocos residuais, para melhorar o desempenho de detecção em relação a suas versões anteriores.

Figura 8 – Camadas da rede Yolov3, utilizada como extrator base de características

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Fonte: (REDMON; FARHADI, 2018)

Figura 9 - Arquitetura da rede Yolov3.

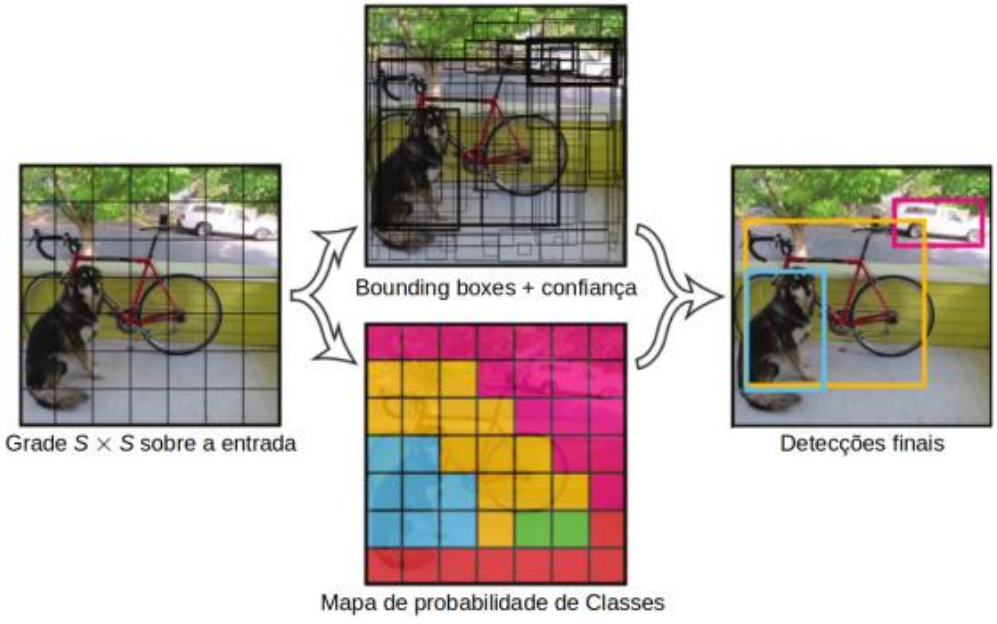


Fonte: (MAO et al., 2019)

Novas melhorias foram implementadas por Bochkovskiy (2020) quando desenvolveram a Yolov4, melhorando ainda mais a precisão da detecção em comparação com o Yolov3 e estabelecendo novos padrões de desempenho na detecção de objetos.

Assim, aumentado a eficiência na utilização das *bounding boxes*, formas geométricas que são sobrepostas na imagem de entrada considerando uma grade no tamanho $S \times S$. Essas células são preenchidas por *B bounding boxes*, as áreas cobertas recebem uma classificação permitindo a geração de novas *bounding boxes* com o nível de confiança *Pr* para a presença de um objeto em cada. Além disso, é determinado um centroide definido por (x, y) e a largura e altura (w, h), para cada área e cada classe *C*. Esse processo é ilustrado na Figura 10.

Figura 10 – Detecção de objetos por meio dos *bounding boxes*.



Fonte: Adaptada de (REDMON et al., 2015)

Com os *bounding boxes*, são utilizados os algoritmos de *Intersection Over Union* (IoU) e *NonMaximum Suppression* (NMS) que são aplicados para produzir as predições finais para cada classe. O algoritmo NMS determina apenas as predições que apresentarem confiança acima de um determinado *threshold*, ou seja um determinado ponto de corte, como mostrado na Figura 11.

Figura 11 – Processo de utilização do NMS com IoU.

Fonte: Adaptado de (BOCHKOVSKIY; WANG; LIAO, 2020)

Sendo uma métrica que indica o quanto uma área sobrepõe outra, a IoU trabalha em uma escala de 0 a 1 como uma porcentagem. A IoU pode ser utilizada de duas maneiras, em conjunto com NMS (como visto anteriormente), ou como métrica de acurácia para as *bounding boxes* preditas e esperadas. Quando utilizada com NMS, o valor elevado da IoU, levando em consideração um *threshold*, é usada para remover possíveis predições redundantes. Quando utilizada como métrica de acurácia, o valor elevado da IoU indica melhor predição. No capítulo 4 será retratado como é realizado o cálculo dessas métricas.

Durante o treinamento do primeiro estágio para identificar se a detecção da área de interesse será realizada de maneira correta podemos analisar a IoU por meio da sobreposição a comparação que existe entre a marcação original esperada e a predita.

Para isso utilizamos a Equação (1), onde temos C_{PRED} e C_{REAL} como as características de cada marcação, sendo, os eixos X e Y, largura e altura(ZHENG et al., 2019). Essas

variações apresentam soluções específicas para diferentes problemas, melhorando a velocidade de convergência e a precisão da localização do objeto.

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

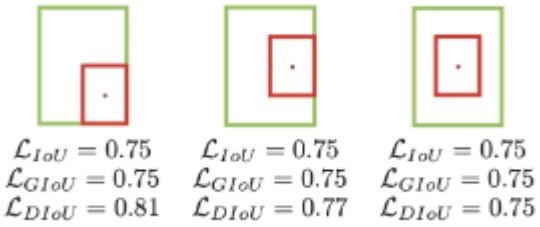
Primeiro, GIoU (*Generalized IoU*) corrige o cálculo do gradiente quando a sobreposição entre as marcações não existe (REZATOFIGHI et al., 2019), conforme a Equação (2). Em seguida, o DIoU (*Distance IoU*) atua para minimizar a distância entre os pontos centrais das marcações, conforme a Equação (3).

$$\text{GIoU} = \text{IoU} - \frac{|c \therefore (C_{PRED} \cup C_{REAL})|}{|c|} \quad (2)$$

$$DIoU = IoU + \frac{p^2(A_{PRED}, A_{REAL})}{c^2} \quad (3)$$

As distinções entre os cálculos das funções podem ser observadas na Figura 12, onde o retângulo interno, em vermelho, representa a marcação ideal, definida na anotação da imagem. O retângulo externo, em verde, representa a marcação real, obtida como retorno do modelo ao processar a imagem (ZHENG et al., 2019).

Figura 12 – Métricas IoU, DioU e GIoU



Fonte: (ZHENG et al., 2019)

Além da métrica DIoU, dada pela Equação (3), 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 (4), resultado da medida de consistência da proporção entre altura (*h*) e largura (*w*) e α, equivalente a Equação (5), usada como um parâmetro de balanceamento.

Combinando esses parâmetros obtemos a CIoU (*Complete* IoU), dado pela Equação (6), o qual é utilizado como *loss function* padrão no YOLOv4 (ZHENG et al., 2019).

Para o contexto de detecção de objetos quanto maior for a área sobreposta do objeto detectado melhor para a avaliação.

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

$$\alpha = \frac{v}{\text{IoU}+v'} \quad (5)$$

$$CIoU = DIoU + \alpha v \quad (6)$$

Ademais, para computarmos os resultados em relação a tanto ao primeiro quando ao segundo estágio, temos as métricas como precisão (*P*), Equação (7), e o *recall (R*), Equação (8), 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., 2019).

Com essas informações podemos calcular o *F1-score*, Equação (9), média que representa simetricamente tanto a precisão como o *recall*, como também, a acurácia, Equação (10), que é calculada dividindo o número de previsões corretas pelo número total de previsões.

$$P = \frac{TP}{TP+FP} \quad (7)$$

$$R = \frac{TP}{TP+FN} \quad (8)$$

$$F1 = \frac{2 \times P \times R}{P+R} \quad (9)$$

$$acurácia = \frac{TP+FP}{TP+FN+FP+TN} \quad (10)$$

Na seção a seguir serão descritos trabalhos relacionados a nossa proposta, destacando principalmente aqueles que utilizam a rede *Yolo*.

3 TRABALHOS RELACIONADOS

Segundo Sankaran (2010), doenças em plantas podem ser identificadas compreendendo alguns métodos diretos, como técnicas moleculares para análise e identificação dos agentes causadores da doença, ou métodos indiretos, exemplo espectroscopia e processamento de imagens. Os métodos diretos, por envolverem análise química, possuem sua principal limitação consumirem muito tempo e trabalho na obtenção dos resultados, requerendo procedimentos complexos para sua realização, além de serem invasivos. Métodos indiretos envolvendo espectroscopia e processamento de imagens multi ou hiperespectrais, apresentam avanços nos últimos anos, mas demandam a utilização de sensores volumosos e muitas vezes caros, tornando o processo pouco acessível (NGUGI; ABELWAHAB; ABO-ZAHHAD, 2021).

Entretanto, julgando que as doenças presentes nas plantas, em sua maioria, podem ser percebidas no espectro visível da luz (BARBEDO, 2013), assim possibilitando a obtenção de imagens por meio de câmeras digitais que se encontram presentes em dispositivos eletrônicos de maneira cada vez mais frequentes, o processamento dessas imagens mostra-se uma alternativa satisfatória.

Reconhecimento de padrões de imagens, utilizando técnicas de aprendizado de características baseadas em aprendizado profundo, mostram constantes evoluções (LECUN; BENGIO; HINTON, 2015). Especificamente, trazendo para o contexto da identificação de doenças utilizando imagens de plantas, essas técnicas se mostram promissoras, superando métodos clássicos para o processamento de imagens (KAMILARIS; PRENAFETA-BOLDÚ, 2018).

Segundo (BARBEDO, 2018a), métodos tradicionais como análise discriminante, realce, análise de cores e *thresholding* geraram pequenos avanços apenas, nas últimas décadas, principalmente devido a alguns fatores como seu funcionamento se dando somente sob certas medidas e serem muitos específicos. Assim, recentes trabalhos estão utilizando técnicas de aprendizado profundo, nesse contexto, se apresentam duas possíveis abordagem para identificação de doenças em plantas, sendo elas, detecção e

classificação.

3.1 Métodos baseados em detecção

A maioria dos recentes trabalhos trazem o problema para identificação de doenças em plantas como classificação de imagens como um todo, usando modelos de redes que apresentam uma única saída para dada imagem de entrada. Um número mais reduzido de trabalhos traz uma abordagem dessa tarefa como detecção de objetos, fornecendo a localização das regiões de doenças na imagem.

Ramcharan (2019) utiliza uma rede neural para detecção de objetos, tendo como base a rede MobileNet, para a detecção de 7 diferentes tipos de doenças, divididos em 2 níveis de severidade cada, em folhas de mandioca, implementado em um sistema executado em *smartphones*. O sistema aponta que os resultados obtidos, embora de bom desempenho para a detecção dos sintomas relatados, encontra considerável dificuldade quando as doenças apresentam sintomas pouco evidentes, sugerindo uma maior variabilidade nas imagens de treinamento do modelo com essas características.

No trabalho de Selvara (2019) se desenvolve um método para detecção de folhas doentes em pepinos tendo como base um sistema de dois estágios: i) primeiro, realiza a detecção das folhas doentes, segmentando-as do fundo e de outros possíveis elementos presentes na imagem; ii) segundo, as regiões segmentadas são enviadas para uma rede neural convolucional para a classificação. Usando o conjunto de imagens obtidos de diferentes fazendas para a realização do treinamento e teste, os resultados apontam que tal método apresenta maior capacidade de generalização e menos sobre ajuste, se comparado, a detectores de um único estágio. O sistema, no entanto, não define qual a doença em questão, somente classifica as folhas em doentes ou saudáveis.

O trabalho descrito em (FUENTES et al., 2017) buscou superar algumas limitações existentes, como objetivo, para lidar com imagens em diferentes condições de cores, iluminação e tamanhos das regiões de doenças, além da existência de ruído de fundo nas imagens. Buscando criar um detector em tempo real para a identificação de nove doenças em folhas de tomate, foram coletadas um total de 5000 imagens. Após realizar a anotação dos *bounding boxes* das regiões de interesse, o número de amostra

para o treinamento aumentou para 43398. Foram utilizadas diversas redes profundas, como VGG-16, AlexNet, ResNet e GoogLeNet, para realizar a extração de características, em conjunto com o detector Faster R-CNN. A melhor combinação, Faster R-CNN e VGG-16, apresentou precisão média de 83,06%.

Em um trabalho seguinte Fuentes (2018) apresentaram a utilização de dois modelos de diagnóstico independentes: i) primeiro, correspondendo a utilização de uma rede Faster R-CNN sendo ela responsável por prever as prováveis regiões de interesse; ii) segundo um banco de filtros composto por várias CNNs, para cada possível classe, resultando em uma melhora relatada de 13% na precisão média, em relação ao trabalho anterior quando comparada.

3.2 Métodos baseados em classificação

Diferentemente dos trabalhos mencionados na seção anterior, os métodos apresentados nessa seção realizam a classificação da imagem como um tolo, categorizando-a, não informando a região de interesse na imagem.

Apresentam bons desempenhos quando a maior parte da imagem é ocupada pelo objeto de interesse, contudo, geralmente possui o desempenho afetado quando na imagem estão presentes outros objetos ou fundo ruidoso. Na literatura, esse método, é mais frequentemente utilizado para identificação de doenças em plantas via aprendizado profundo.

Sladojevic (2016) desenvolveram um trabalho utilizando rede neurais convolucionais para classificar 13 tipos de doenças em folhas, que variavam em quatro espécies de plantas (pera, maçã, videira e pêssego), com uma acurácia, em média, de 96,3%.

Mohanty (2016) utilizou as arquiteturas AlexNet e GoogLeNet para identificar 26 tipos de doenças usando 14 diferentes tipos de plantas, extraídos do conjunto de dados públicos *Plant Village* (HUGHES; SALATHÉ; OTHERS, 2015), que é composto por um total de 54309 imagens de folhas de plantas, variando entre doentes e saudáveis.

Dentre os modelos utilizados o melhor obtido foi a rede GoogLeNet usando o conjunto do ImageNet no pré-treinamento, alcançando uma acurácia de 99,35%. Porém, ao realizar testes com o mesmo modelo usando um conjunto de imagens de teste diferente, a acurácia reduziu significativamente para 31%.

Ainda utilizando o conjunto de dados do *Plant Village*, Ferentinos (2018) comparou o desempenho das redes AlexNetOWTBn, AlexNet, Oveerfeat, GoogLeNet e VGG16, obtendo com esta última 99,53% de acurácia na classificação de imagens. Porém, as arquiteturas novamente apresentaram redução em suas métricas quando treinados por imagens coletas em ambiente real e testadas com imagens coletadas em laboratório, sofrendo uma diminuição de 34,74%, confirmando a necessidade de utilização de um volume suficiente de imagens que representam condições reais e diversas nos dados de treinamento, assim possibilitando a geração de modelos mais robustos e generalistas. Barbedo (2018b) aponta que a falta de bases de imagens suficientemente abrangentes e diversificadas para o treinamento de modelos, acaba não permitindo uma maior generalização.

No trabalho de Kc (2019) é demonstrado uma análise das variações da rede MobileNet ao classificar 55 classes sendo essas compostas por pares doença-espécie presentes no conjunto *Plant Village*. A versão reduzida da rede apresentou 98,34%, sendo 29 vezes menor, em relação ao número de parâmetros, ao modelo VGG e 6 vezes menor em comparação ao modelo original da MobileNet.

Em (LIU et al., 2017) foi utilizado uma derivação da rede AlexNet na classificação de 4 diferentes tipos de doenças em folhas de maçã, atingindo a uma média na acurácia de 97,6%.

Wang (2017) utilizaram as redes ResNet, Inception-V3, VGG16 e VGG-19 na classificação de 4 diferentes estágios da mesma doença presente nas folhas de maçã. As redes foram analisadas recebendo tanto o treinamento completo, como também, pré-treinadas com base no conjunto ImageNet (*transfer learning*). Como melhor resultado o modelo VGG16, usando *transfer learning*, apresentou acurácia de 90,4%.

Em (AMARA; BOUAZIZ; ALGERGAWY, 2017) foi utilizado um subconjunto de imagens do *Plant Village*, onde foi treinado um modelo LeNet par classificar doenças em bananeiras. O modelo foi capaz de obter uma média de acurácia superior a 90% na

classificação.

DeChant (2017), em seu estudo, busca identificar uma doença presente nas folhas do milho, as imagens para o treinamento foram obtidas em condições de campo. O sistema está dividido em três etapas: i) primeiro, foi utilizado 5 CNNs independentes, que foram treinadas em um conjunto formado de sub-imagens a partir de recorte da imagem original; ii) segundo, com combinações entre as CNNs são gerados mapas de calor; iii) por fim, os mapas de calor são enviados como entrada para uma última CNN, que retorna à probabilidade de estar presente na imagem uma região com a doença. A média da acurácia foi de 96,7%.

Barbedo (2019) usa somente sub-regiões da imagem original, que apresentam doenças, para realizar o treinamento do modelo de classificação, o que proporciona um maior número de amostrar para o treinamento. Utilizando esse tipo de pré-processamento atingiu uma acurácia de 94%, em contraposição ao 82% quando utilizado a imagem inteira.

3.3 Considerações

Neste capítulo foram apresentados e detalhados alguns dos trabalhos encontrados na literatura em relação a identificação de doenças utilizando imagens de plantas por meio do aprendizado profundo.

Embora a grande parte dos trabalhos apresente como solução um único estágio para realizar a tarefa, soluções híbridas, trazendo mais de um estágio e envolvendo as duas abordagens, detectar e classificar, mostraram-se favoráveis para solucionar esse tipo de problema.

Podemos perceber ainda, que a maioria dos trabalhos listados realizam o treinamento e os testes de seus modelos em conjuntos de dados parecidos, ou seja, um subconjunto derivado de um conjunto maior. Nota-se, que a maior parte dos trabalhos usam o conjunto de imagens *Plant Village* para dividir entre o conjunto de treinamento e o de teste.

Tendo em vista os estudos analisados, este trabalho busca uma forma de melhorar a detecção e classificação de doenças em plantas por meio de imagens utilizando um modelo de dois estágios.

Ainda, buscar estruturar o *dataset* que será utilizado tentando garantir uma maior generalização do modelo resultante. E por fim, tentar contextualizar, em relação ao grau de severidade da doença, a planta que se encontra na imagem.

.

4 METODOLOGIA E IMPLEMENTAÇÃO

Buscamos nesse trabalho desenvolver um sistema baseado em dois estágios, um para detecção e outro para classificação de cada objeto detectado pelo primeiro estágio e assim criando uma classificação de contexto para identificação de doenças de plantas.

Considera-se que, ao isolar o processo de detecção de regiões do processo de classificação, pode-se obter um modelo mais robusto, capaz de alcançar melhor generalização, devido cada tarefa ser realizada por uma rede específica. Tal cenário será desenvolvido nesse trabalho e para realizar a implementação do sistema, treinamento e teste, foi utilizando um computador com as seguintes configurações:

- Sistema Operacional Windows 11 Pro - versão 22H2 64-bit;
- RAM de 16GB;
- Processador Intel Core I5-5575R;
- GPU GeForce GTX 750 Ti, memória 4GB, CUDA 10.1;
- SSD 500GB e HD 1TB.

Os algoritmos durante o treinamento as seguintes bibliotecas de código:

- CUDA e cuDNN: Desenvolvidas pela empresa Nvidia, tais bibliotecas, garante a utilização de GPUs para a execução de algoritmos de alto custo;
- Darknet: *Framework* desenvolvidos pelos autores do Yolo, contém a implementação da arquitetura das quatro primeiras versões do Yolo;
- Make Sense: *Software* de código aberto, utilizado para realizar as anotações das imagens da base de dados e exportar os respectivos arquivos, para os diferentes formatos de redes;
- Tensorflow 2: Biblioteca de código aberto desenvolvida pela Google,

sendo possível executar e configurar diferentes treinamentos utilizando variadas arquiteturas de redes neurais

No sistema proposta o primeiro estágio é responsável por encontrar regiões de interesse nas imagens de entrada. As regiões de interessem compreendem folhas de seis tipos de plantas diferentes. E o segundo estágio é responsável por classificar cada uma das classes de folhas como saudável ou alguma das doenças daquela classe de planta. Dessa forma, podemos contabilizar a quantidade de regiões de interesse que apresentam patógenos na imagem original, e com base nessa informação, podemos constatar uma proporcionalidade em relação a patógenos presentes na imagem original, classificando a mesma, a isso denominamos de classificação por contexto. Portanto, além da construção dos dois estágios, a comunicação entre eles e a classificação final de contexto a partir do trabalho dos dois estágios faz parte das principais contribuições deste trabalho. A seguir será detalhado a construção do *dataset*, seguido pela implementação de cada estágio separadamente e por fim o sistema como um todo para classificação de contexto.

4.1 Dataset

Para o processo de treinamento e validação de cada estágio do sistema proposto foi necessário primeiramente a construção de um conjunto de dados (*dataset*) de imagens de folhas de plantas saudáveis ou com algum patógeno. Para isso utilizamos dois *dataset* públicos: o primeiro *Plant Village* (GEETHARAMANI; J, 2019) e *FieldPlant* (MOUPOJOU et al., 2023). O primeiro é o mais utilizado nos trabalhos que propõem classificação de doenças em folhas. Originalmente este *dataset* possui 61486 imagens dividida em 42 classes, no entanto para alguns tipos de plantas há apenas folhas saudáveis, além de tais imagens serem capturadas em laboratório com um fundo padrão diferente do ambiente natural. Dessa forma, adotamos o seguinte critério de inclusão no nosso *dataset*: cada tipo de planta deve ter uma classe saudável e pelo menos um tipos de patógenos. Assim, utilizamos 65% das classes desse *dataset Plant Village*, o que correspondeu a 27 classes, sendo essas as mesmas classes encontradas no *dataset FieldPlant*, e 16259 imagens no total com os critérios de seleção, com a distribuição listada na quarta coluna (Qtd. Plant Village) da Tabela 3, e os número e nomes das

classes na segunda (No. da classe) e terceira (Classes) colunas respectivamente.

O segundo *dataset FieldPlant* foi usado para aumentar a variabiliade nas imagens e buscar ampliar a capacidade de generalização da rede, uma vez que é formado por imagens de plantas no ambiente natural. Tal *dataset* possui originalmente 8179 amostras distribuídas em 27 classes, as quais também foram selecionadas pelo mesmo critério de inclusão. Logo, para formação do nosso *dataset* utilizamos 100% desse segundo conjunto de dados, conforme a quarta coluna (Qtd. 2º dataset) da Tabela 3. A soma dos quantitativos selecionados para nosso *dataset* é apresentada na quinta coluna (Subtotal) desta tabela, resultando em 27 classes, os quais serão utilizados pelo segundo estágio do sistema para classificação.

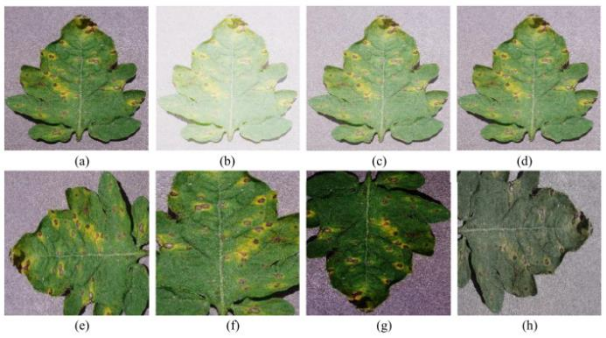
Superclasses	No. da classe	Classes	Qtd. Plant Village	Qtd. FieldPlant	Subtotal	Total com *data augmentation*	Total da superclasse
Maçã	00	Maçã com sarna	1136	612	1748	2408	8636
	01	Maçã com podridão	924	498	1422	2318	
	02	Maçã com ferrugem	706	381	1087	1910	
	03	Maçã saudável	1061	571	1632	2000	
Pimentão	04	Pimentão com mancha bacteriana	825	444	1269	1860	3357
	05	Pimentão saudável	720	388	1108	1497	
Milho	06	Milho com cercosporiose	618	334	952	1635	6475
	07	Milho com ferrugem	685	369	1054	1765	
	08	Milho com praga	544	293	837	1637	
	09	Milho saudável	588	316	904	1588	
Uva	10	Uva com podridão	338	182	520	838	3470
	11	Uva com sarampo negro	318	171	489	782	
	12	Uva com praga	275	148	423	952	
	13	Uva saudável	320	173	493	896	
Batata	14	Batata com queimadura	605	326	931	1529	5402
	15	Batata com fungo	489	264	753	1247	

	16	Batata com bacteria	481	259	740	1242	
	17	Batata saudável	555	301	856	1384	
Tomate	18	Tomate com queimadura	624	335	959	1553	12877
	19	Tomate saudável	688	371	1059	1624	
	20	Tomate com fungo	561	301	862	1369	
	21	Tomat com Mofo	514	277	791	1204	
	22	Tomate com mancha negra	560	303	863	1327	
	23	Tomate com ácaro	512	277	789	1295	
	24	Tomate com pinta preta	486	262	748	1364	
	25	Tomate com vírus	597	321	918	1607	
	26	Tomate com folha amarela	529	285	814	1534	

Tabela 2 – Informações do *Dataset*

Após a inclusão das classes dos dois *datasets*, obedecendo nossso critério de inclusão, verificou-se um desbalanceamento entre as classes. Dessa forma, foram utilizadas técnicas de data augmentation para garantir uma quantidade mínima imagens para as classes, buscando um quantitativo igualitario para cada classe dentro do grupo. 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. A Figura 13 ilustra essas variações, onde temos (a) sendo a imagem original e (b, c, d, e, f, g, h) imagens artificialmente geradas utilizando meios de transformação como ruído, variação da luminosidade e variações no eixo horizontal e vertical.

Figura 13 – Exemplo das técnicas de *data augmentation* aplicadas.



Fonte: adaptada de (GEETHARAMANI; J, 2019)

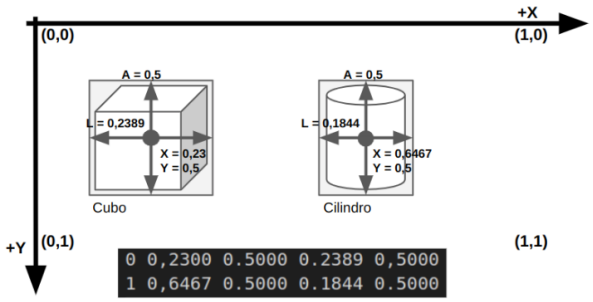
Para o primeiro estágio do sistema (detecção) estamos interessados apenas nas características gerais das folhas de uma determinada planta, independente se está saudável ou com algum patógeno. Dessa forma, agrupamos as diversas classes de patógenos de um mesmo tipo de planta em supergrupos, resultando em seis tipos de plantas, conforme a primeira coluna (Superclasses) da Tabela 3. A distribuição dos quantitativos das superclasses é listada na última coluna dessa tabela.

Para a utilização da base de dados gerada no treinamento de detecção do modelo Yolov4 se faz necessário que, cada imagem tenha um respectivo arquivo de texto contendo as informações da classe presente e 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 14 é apresentado um exemplo do formato Yolov4 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 das imagens utilizadas do *dataset* FieldPlant foram realizadas manualmente utilizando *bounding boxes* através da ferramenta *Make Sense* (SKALSKI, 2019), o software calcula e gera os descritivos de todas as anotações visuais, salvando as informações em um arquivo de texto. As imagens do *dataset* Plant VIllage já possuíam o arquivo de notação para cada imagem. Nesta etapa consideramos o *dataset* com as seis superclasses e padronizamos os arquivos de texto com o nome de cada classe, conforme a primeira coluna (Superclasses) da Tabela 3

Figura 14 - Descrição do mapeamento do objeto no formato Yolov4



Fonte: (BOCHKOVSKIY; WANG; LIAO, 2020)

Essas imagens, juntamente com esses novos arquivos de texto, foram utilizadas como entradas para o treinamento e validação do primeiro estágio do sistema. Assim, a Tabela 4 demonstra essa distribuição quantitativa.

Classe de folhas *super grupos*

Nome da classe	Número de imagens	Treino (70%)	Teste (20%)	Validação (10%)
Maça	8636	6045	1727	864
Pimentão	3357	2350	671	336
Milho	6475	4532	1295	648
Batata	5402	3781	1080	541
Tomate	12877	9014	2575	1288
Uva	3470	2429	694	347

Tabela 3 – Dados do treinamento no primeiro estágio

Na Tabela 5 está demonstrado a distribuição quantitativa em relação a cada classe, essas informações serão utilizadas para a realização o treinamento do segundo estágio, que vamos abordar mais a frente.

Superclasses	No. da classe	Classes	Qtd. Imagens	Treino (70%)	Teste (20%)	Validação (10%)
Maçã	00	Maçã com sarna	2408	1685	481	242
	01	Maçã com podridão	2318	1622	463	233
	02	Maçã com ferrugem	1910	1337	382	191
	03	Maçã saudável	2000	1400	400	200
Pimentão	04	Pimentão com mancha bacteriana	1860	1302	372	186
	05	Pimentão saudável	1497	1048	299	150

Milho	06	Milho com cercosporiose	1635	1144	327	164
	07	Milho com ferrugem	1765	1235	353	177
	08	Milho com praga	1637	1146	327	164
	09	Milho saudável	1588	1112	317	159
Uva	10	Uva com podridão	838	586	167	85
	11	Uva com sarampo negro	782	547	156	79
	12	Uva com praga	952	665	190	97
	13	Uva saudável	896	627	179	90
Batata	14	Batata com queimadura	1529	1070	305	154
	15	Batata com fungo	1247	873	249	125
	16	Batata com bacteria	1242	869	248	125
	17	Batata saudável	1384	968	276	140
Tomate	18	Tomate com queimadura	1553	1087	310	156
	19	Tomate saudável	1624	1136	324	164
	20	Tomate com fungo	1369	958	273	138
	21	Tomat com Mofo	1204	842	240	122
	22	Tomate com mancha negra	1327	929	265	133
	23	Tomate com ácaro	1295	906	259	130
	24	Tomate com pinta preta	1364	954	272	138
	25	Tomate com vírus	1607	1124	321	162
	26	Tomate com folha amarela	1534	1073	306	155

Tabela 4 - Dados do treinamento no segundo estágio

A vantagem do *dataset* utilizado ser construído com base em conjuntos de dados diferentes, onde imagens foram selecionadas e realizado o processo de anotação, permitindo realizar uma validação externa do detector, buscando garantir uma maior generalização.

De fato, alguns trabalhos expõem que conjuntos de dados de imagens de plantas atuais não são suficientemente extensos para permitir uma maior generalização, por exemplo, (FERENTINOS, 2018) e (BARBEDO, 2019) apontando que para esse segmento existem poucos *datasets* com grande volume de amostras, em comparação temos o *dataset* ImageNet o qual ele apresenta mais de 14 milhões de amostras, logo, a construção de um conjunto de dados especifico é valido para uma maior generalização para os estágios do sistema, que vamos retratar a seguir.

4.2 Primeiro Estágio

Na composição do sistema, o primeiro estágio consiste na utilização de uma rede neural convolucional para deteção de objetos, Yolov4 (BOCHKOVSKIY; WANG; LIAO, 2020) tendo como entrada uma imagem digital, a qual chamaremos de imagem original, e fornecendo como saída as regiões de interesse (folhas). O principal objetivo desse primeiro estágio é eliminar partes irrelevantes para a classificação da região de interesse. Dessa forma, o sistema utiliza as coordenadas de cada folha detectada para recordar a imagem original e assim gerar uma nova imagem para cada região de interesse, conforme mostra a Figura 15.

Figura 15 – Representação do primeiro estágio do sistema.

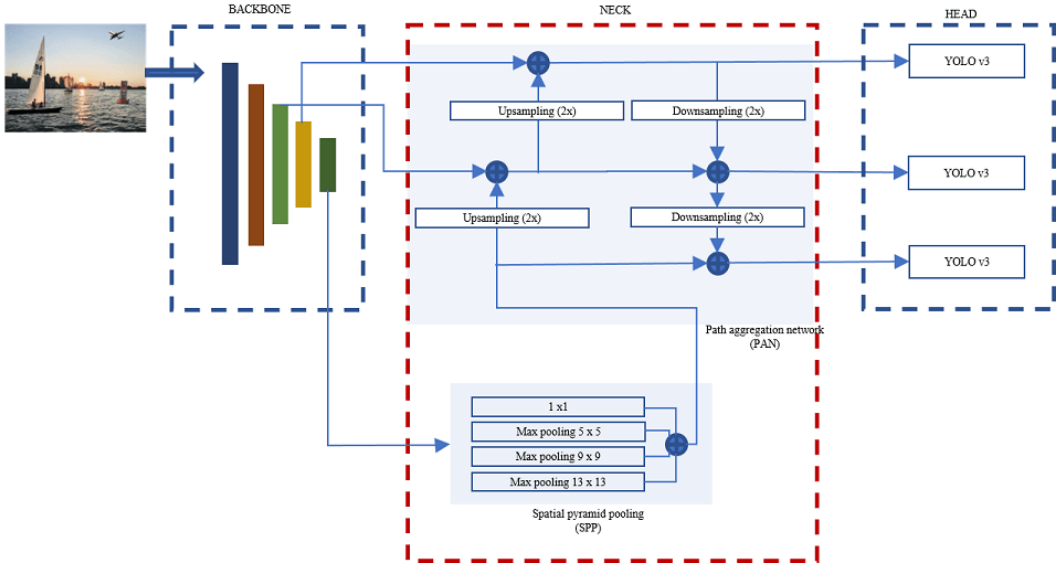
Fonte: autoria própria

O primeiro estágio do sistema é formado principalmente por uma rede Yolov4~~,~~ configurada para buscar o equilíbrio entre a resolução da rede de entrada juntamente com o

número de camadas convolucionais, para manter a precisão e a velocidade, por isso, optamos pela arquitetura detalhada na Figura 16:

- *Backbone*: *Cross-Stage-Partial-connections* com Darknet53 (CSPDarknet53);
- *Neck*: *Spatial Piramid Pooling* (SPP), para otimizar o campo receptivo;
- *Neck*: *Path Aggregation Network* (PAN), método para a agregação de parâmetros;
- *Head*: Yolov3, sendo responsável pela saída.

Figura 16 – Arquitetura Yolov4



Fonte: (BOCHKOVSKIY; WANG; LIAO, 2020)

Com isso, para facilitar o treinamento da rede foi utilizado a técnica de *transfer learning* nas 137 primeiras camadas convolucionais. 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 6, baseados nos estudos de (RAHIMZADEH; ATTAR; SAKHAEI, 2021). A seguir vamos apresentar o segundo estágio, que vai receber os dados de saída desse primeiro estágio.

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

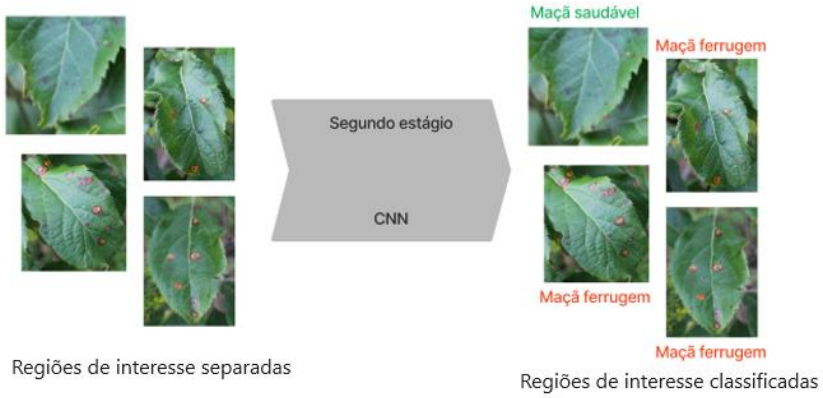
Tabela 5 - Parâmetros utilizados na configuração do primeiro estágio

4.3 Segundo Estágio

Nesse estágio a entrada consiste em receber as regiões de interesse, analisar e classificar essas regiões, por meio de uma rede neural convolucional EfficienNet-B7, atribuindo uma classe a cada região de interesse, assim, determinando a presença ou não de um patógeno.

Tendo como principal objetivo desse estágio a correta classificação das áreas de interesse, para realizar a verificação dessas classificações utilizamos os arquivos de texto das anotações da imagem original para comparação. Exemplificado o funcionamento do segundo estágio na Figura 17.

Figura 17 - Representação do segundo estágio do sistema



Fonte: autoria própria

Para realizar o treinamento desse estágio foi utilizado a biblioteca Tensorflow/Keras,

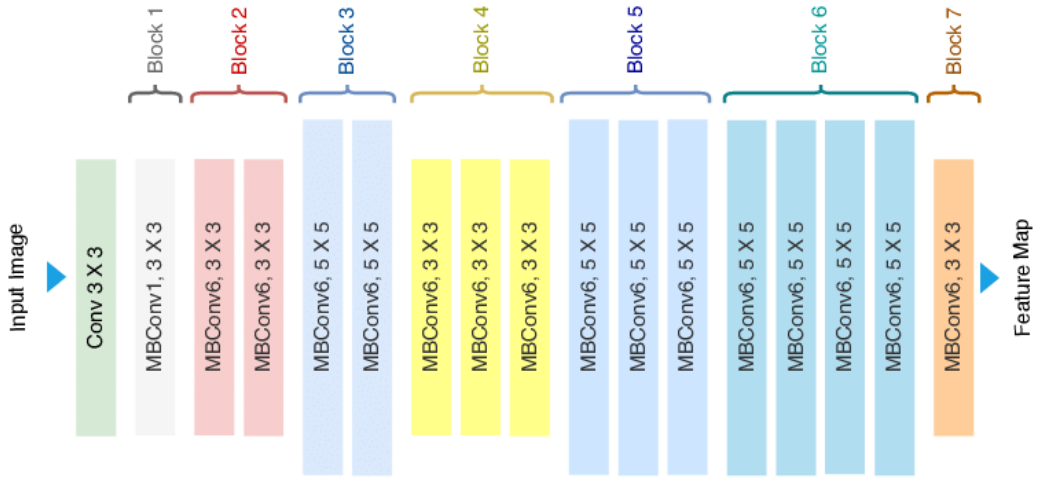
para o processamento das informações, a biblioteca Numpy para a manipulação dos dados referentes as imagens, a biblioteca OpenCV para visualização e manipulação gráfica das imagens. Além disso, para obter um melhor desempenho no treinamento do modelo, foram utilizados os parâmetros listados na Tabela 7.

Parêmetro	Valor
Batch Size	32
Optimizer	adam
Loss Function	Categorical Crossentopy
Epochs	10
Learning Rate	0.001

Tabela 6 - Parâmetros utilizados na configuração do segundo estágio

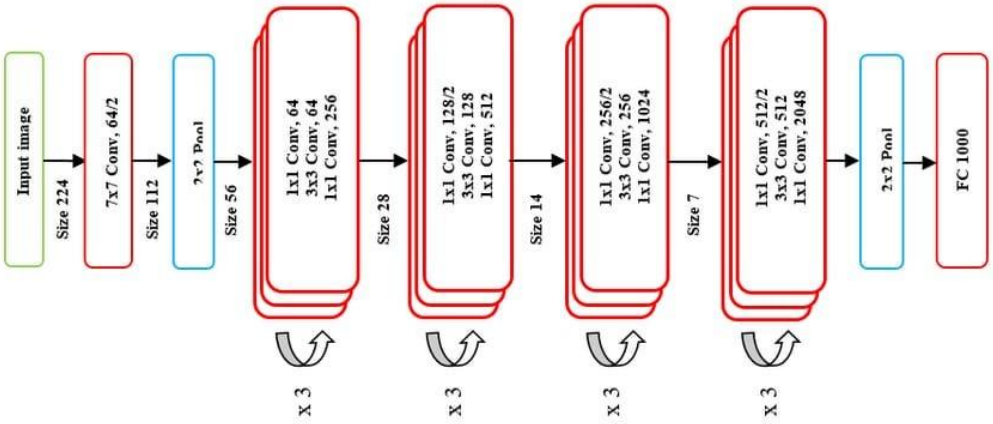
Assim, para a escolha dessa rede neural convolucional buscamos na literatura opções e chegamos a duas opções EfficientNet-B7 (TAN; LE, 2019), podemos observar o detalhamento de sua arquitetura na Figura 18, e ResNet-50 (HE et al., 2016), a Figura 19 exibe em detalhes sua arquitetura.

Figura 18 – Arquitetura EfficienNet-B7



Fonte: (TAN; LE, 2019)

Figura 19 – Arquitetura ResNet-50



Fonte: (HE et al., 2016)

Tentando identificar qual a melhor rede para esse estágio, foi realizado testes na EfficientNetB7 e ResNet-50, utilizando *transfer learning* buscando garantir um melhor desempenho e menor tempo de treinamento. Inicialmente, as redes foram submetidas ao treinamento com base no mesmo conjunto de dados utilizado no primeiro estágio, podemos verificar os resultados na Tabela 8.

Métricas das Arquiteturas de Classificação

Rede	Precisão Média (%)	Acurácia (%)	Recall (%)	F1-Score (%)
ResNet-50	79,0	81,0	80,0	83,0
EfficientNet-B7	84,0	86,0	84,0	87,0

Tabela 7 – Métricas das Arquiteturas de Classificação

Os primeiros resultados encontrados, indicavam valores satisfatórios, porém, aventavam uma possibilidade de melhora na acurácia da arquitetura do classificador, assim, utilizamos o primeiro estágio do sistema (detector) para recortar as folhas individualmente de todo o *dataset* apresentado na Tabela 5 e geramos um novo *dataset* para o classificador. Esse novo *dataset*, que chamaremos de *Single Leaves*, utilizam os mesmos rótulos do *dataset* anterior, mas agora contendo 89.874 amostras, distribuídas conforme a Tabela 9.

Utilizamos o *Single Leaves* para realizar um novo treinamento no classificador tendo como proposito trazer até o classificador somente a área de interesse que será classificada, assim, garantindo para o sistema esse filtro onde no primeiro estágio temos a realização do treinamento para garantir a máxima generalização possível para detecção de folhas e no

segundo estágio um treinamento minucioso para a classificação das folhas.

Superclasses	No. da classe	Classes	Qtd. Imagens	Treino (70%)	Teste (20%)	Validação (10%)
Maçã	00	Maçã com sarna	5380	3766	1076	538
	01	Maçã com podridão	5177	3625	1035	517
	02	Maçã com ferrugem	4266	2987	853	426
	03	Maçã saudável	4467	3128	893	446
Pimentão	04	Pimentão com mancha bacteriana	4156	2910	831	415
	05	Pimentão saudável	3345	2342	669	334
Milho	06	Milho com cercosporiose	3652	2557	730	365
	07	Milho com ferrugem	3942	2760	788	394
	08	Milho com praga	3656	2560	731	365
	09	Milho saudável	3545	2482	709	354
Uva	10	Uva com podridão	1872	1311	374	187
	11	Uva com sarampo negro	1746	1223	349	174
	12	Uva com praga	2125	1488	425	212
	13	Uva saudável	2002	1401	400	201
Batata	14	Batata com queimadura	3415	2391	683	341
	15	Batata com fungo	2785	1950	557	278
	16	Batata com bacteria	2774	1942	555	277
	17	Batata saudável	3091	2164	618	309
Tomate	18	Tomate com queimadura	3470	2429	694	347
	19	Tomate saudável	3627	2540	725	362
	20	Tomate com fungo	3057	2141	611	305
	21	Tomat com Mofo	2690	1883	538	269
	22	Tomate com	2962	2074	592	296

		mancha negra				
	23	Tomate com ácaro	2894	2026	579	289
	24	Tomate com pinta preta	3045	2132	609	304
	25	Tomate com vírus	3590	2513	718	359
	26	Tomate com folha amarela	3425	2398	685	342

Tabela 8 - Dados do treinamento no segundo estágio utilizando o *dataset Single Leaves*

As arquiteturas, utilizando o *dataset Single Leaves*, passaram por um novo treinamento apresentando melhores resultados, os quais vamos detalhar na seção 5.2, sendo que a rede EfficientNet-B7 se destacou entre essas duas opções, assim, tornando-se a escolha de arquitetura para o classificador.

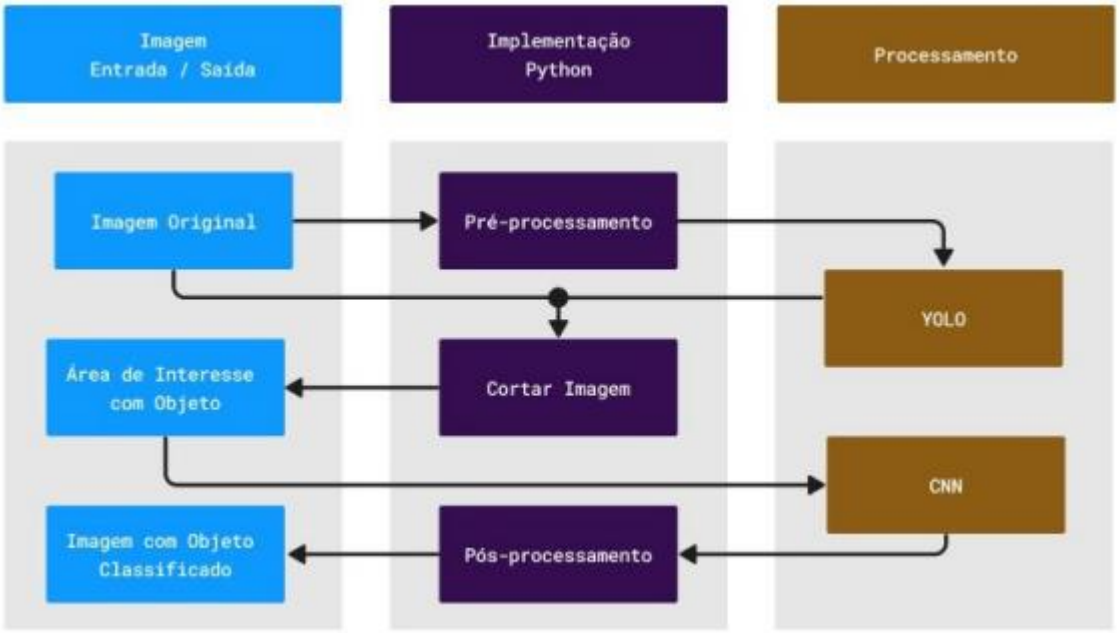
4.4 Sistema Completo de Classificação de Contexto

A princípio, nossa abordagem propõe um sistema, que utilizando o primeiro e segundo estágios, se compromete a apresentar como resultado uma classificação baseada no contexto em que a planta está inserida. Para exemplificar o funcionamento do sistema temos a Figura 20.

Figura 20 - Representação do Sistema

Portanto, o sistema segue um esquema, que se encontra na Figura 21, para receber a imagem, passando pelos estágios e realizando todos os processos necessários de tratamento para chegar ao resultado.

Figura 21 – Sistema de Classificação de Contexto



Devido a isso, temos o sistema dividido em 8 módulos para o seu correto funcionamento, os quais vamos detalhar a seguir.

O primeiro módulo, imagem original, são as imagens presentes no *dataset*, buscamos utilizar imagens com a resolução 1024x1024, já que elas serão recortadas em processos futuros, e assim garantir uma boa definição nas imagens resultantes.

No segundo módulo, pré-processamento, temos a verificação se as imagens possuem o arquivo de texto com as anotações das áreas de interesse, as imagens que não possuírem o arquivo de anotação receberão as anotações de maneira manual e a criação do arquivo de texto. Nesse módulo, todas as imagens recebem um segundo arquivo de texto, o primeiro arquivo além de possuir as coordenadas da área de interesse tem a determinação de qual super grupo essas áreas pertencem. Já no segundo arquivo, o que o diferencia do primeiro é que no lugar da definição do super grupo, tem a definição de qual classe, dentre as 27 presentes no *dataset*, essa área de interesse pertence. O primeiro arquivo de texto será enviado junto a imagem para o primeiro estágio e o segundo arquivo de texto será utilizado no pós-processamento.

O terceiro módulo, *yolo*, representa o primeiro estágio, onde na saída temos as áreas de interesse detectadas, que no nosso caso são as folhas, juntamente com as coordenadas de sua localização na imagem. Nesse módulo a imagem original recebe graficamente, utilizando a biblioteca OpenCV, marcações dos *bounding box* com a presença dos valores de confiança

determinados com base no *threshold* e o nome do super grupo pertencente.

No quarto módulo, cortar imagem, recebemos a imagem original e os dados resultantes do módulo anterior, assim, utilizando a linguagem Python e a biblioteca OpenCV recortamos a imagem original nas coordenadas provenientes do processo de detecção, a imagem resultante é armazenada em um array numpy.

No quinto módulo, área de interesse com o objeto, a imagem da folha passa pelo processo de normalização para se adequar a entrada do próximo módulo, já que dos mais variados formatos e tamanhos de folhas são detectadas no primeiro estágio.

O sexto módulo, CNN, representa o segundo estágio, tem como entrada as imagens do módulo anterior. Assim, realiza o processo de classificação, gerando a informação de qual classe a imagem da folha em questão pertence. As imagens classificadas são armazenadas em um array numpy.

No sétimo módulo, pós-processamento, o array onde as imagens classificadas estão será analisado, para saber se classificações foram corretas, comparando com base no arquivo de texto produzido no segundo módulo.

O último módulo recebe as informações do módulo anterior de quantas e quais imagem foram corretamente classificadas, assim, analisando quantas folhas da imagem original são saudáveis ou apresentam algum tipo de patógeno, dessa forma, classificando o contexto da imagem original.

No próximo capítulo apresentaremos os resultados dos estágios, como também do sistema como um todo.

5 RESULTADOS E EXPERIMENTOS

Neste capítulo expomos e discutimos os resultados obtidos neste trabalho. Nas Seções 5.1 e 5.2 são apresentados o desempenho do primeiro e segundo estágio, respectivamente.

Por fim, na Seção 5.3 apresentamos os resultados do sistema como um classificador de contexto para doenças de plantas.

5.1 Primeiro estágio - Detecção

Para analisarmos esse estágio escolhemos dividir em dois grupos de detectores: o geral e os individuais. O primeiro deles, "detector geral", utiliza todo o *dataset* construído contendo as seis classes de folhas.

Seu objetivo é 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. Na Tabela 10 podemos observar que os valores das métricas são próximos em quase todas as classes, destacando a classe milho, que todos os valores estão acima de 91%, a hipótese para essa discrepância é no formato único da folha do milho, quando comparado com as folhas das outras classes.

Na Tabela 10 podemos observar os resultados de validação para o detector geral.

Métricas do Detector Geral

Classe	Precisão (%)	Recall (%)	IoU (%)	F1-Score (%)
Maça	84,70	86,20	79,90	85,44
Pimentão	83,60	85,30	78,20	84,44
Milho	91,90	91,30	94,50	91,60
Batata	86,30	88,40	81,80	87,34
Tomate	84,20	87,60	79,40	85,87
Uva	82,80	84,70	77,50	83,74

Tabela 9 – Métricas do detector geral

O segundo cenário, é um conjunto de seis detectores individuais, um para cada classe

de folhas, utilizando apenas a base de dados de cada classe correspondente. Buscamos com esses detectores avaliar se há melhoria no desempenho, mesmo com a desvantagem de fazer com que o usuário precise escolher previamente qual a classe deseja detectar**.**

Na Tabela 11 podemos observar os resultados de validação para o conjunto dos detectores individuais, que apresentam melhores resultados do que os presentes na Tabela 10, reforçando a ideia que detectores especialistas para cada classe, possuem melhores resultados**.**

Métricas dos Detectores Individuais

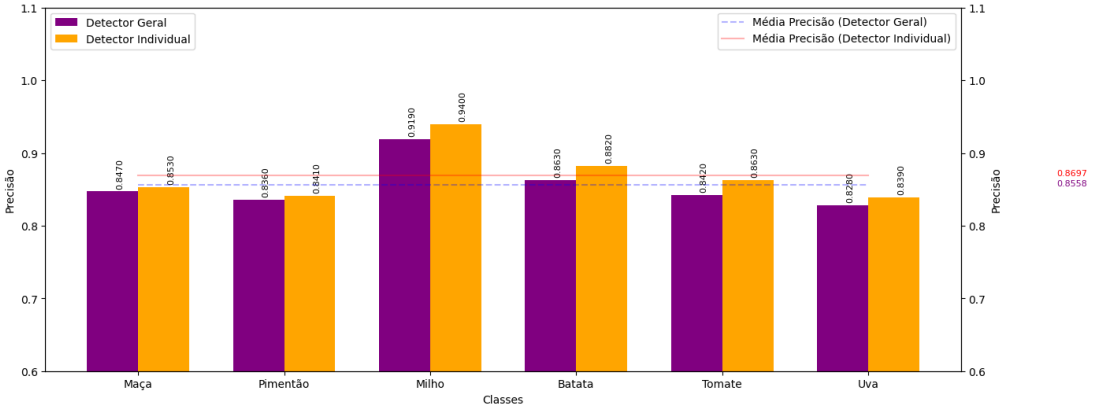
Classe	Precisão (%)	Recall (%)	IoU (%)	F1-Score (%)
Maça	85,30	87,20	89,90	86,24
Pimentão	84,10	86,20	81,20	85,14
Milho	94,00	96,10	97,20	95,04
Batata	88,20	88,90	82,80	88,55
Tomate	86,30	89,10	80,40	87,68
Uva	83,90	89,20	80,50	86,47

Tabela 10 - Métricas dos Detectores Individuais

Os dois cenários (detector geral e grupo de detectores individuais) foram avaliados de maneira isonômica, visando obter a comparação entre eles de forma mais semelhante possível. Assim, realizamos inferências de imagens de 1024x1024 pixels para cada grupo, para realizar comparação das métricas dos detectores.

A Figura 22 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. Nessa comparação é possível perceber que os detectores individuais se sobressaem em todas as classes. 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 relativamente eficiente para essa tarefa. Na mesma figura, as linhas pontilhadas representam as médias de ambos os modelos, confirmando o desempenho do modelo individual 1,6 vezes maior que o geral.

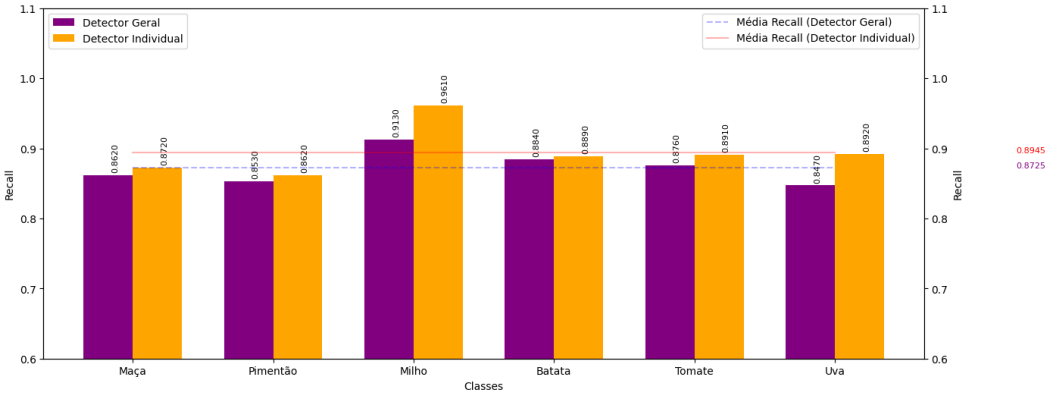
Figura 22 – Comparação entre os detectores individuais e o detector geral em relação a precisão.



Fonte: autoria própria

Na Figura 23 temos 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. Podemos analisar 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 e média geral de 89% e 87% respectivamente para os modelos individuais e geral.

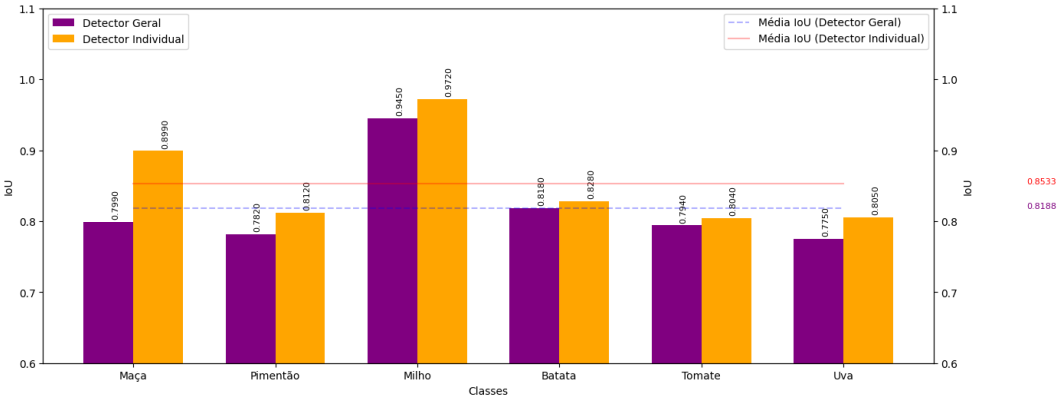
Figura 23 - Comparação entre os detectores individuais e o detector geral em relação ao *recall*.



Fonte: autoria própria

Na Figura 24 temos a comparação da métrica *IoU* com os melhores resultados também dos detectores individuais. Isso significa que os detectores individuais, especializados em um único tipo de planta, conseguem localizar melhor a região dos objetos detectados. Salientando, o detector individual da classe milho, atingindo acima de 97%. Tendo uma média de 81% para o detector geral e 85% para os detectores individuais.

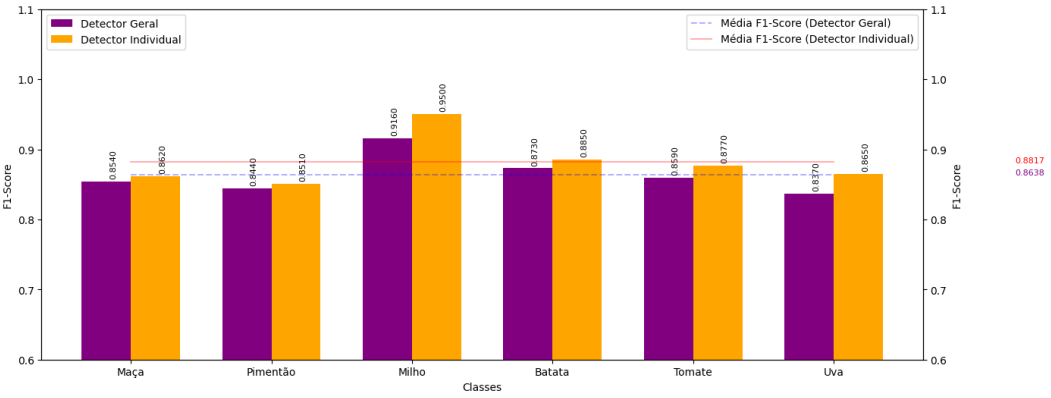
Figura 24 - Comparação entre os detectores individuais e o detector geral em relação ao IoU



Fonte: autoria própria

Na Figura 25 observamos a comparação da métrica *F1-Score*, tratando de uma média harmônica entre a precisão e o *recall* validando os resultados obtidos. A média dos detectores individuais ficou em 88% contra 86% do detector geral.

Figura 25 - Comparação entre os detectores individuais e o detector geral em relação ao F1-Score

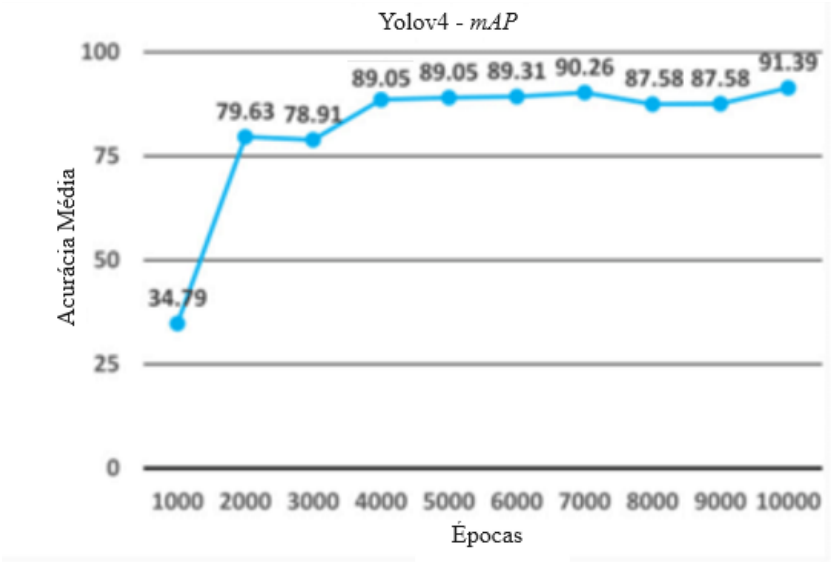


Fonte: autoria própria

O detector geral teve uma média de precisão de 85,6% com desvio padrão de 0,12 enquanto os detectores individuais uma média 86,97% de precisão com desvio padrão de 0,14. A média do *recall* para os detectores geral e individuais foram, respectivamente, 87,25% como desvio padrão de 0,10 e 89,45% e desvio padrão de 0,13. Para a métrica de *IoU* as médias foram de 81,88% com desvio padrão 0,23 (geral) e 85,33% com desvio padrão de 0,25 (individuais). Em relação ao *F1-Score*, para o detector geral uma média de 86,38% e desvio padrão de 0,26 e para os detectores individuais uma média de 88,17% e desvio padrão de 0,8.

Na Figura 26 temos a métrica *mAP*, do inglês Mean Average Precision, atingindo 91,39% utilizando a configuração de *threshold* com valor de 0.75.

Figura 26 – Representação *mAP* ao longo de 10000 interações



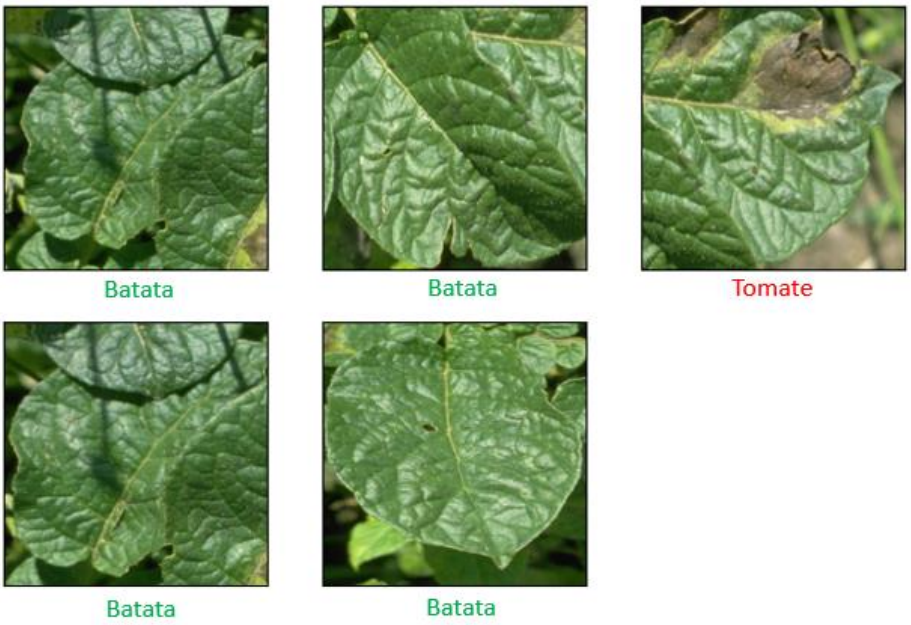
Fonte: autoria própria

Diante dos dados apresentados, pode-se concluir que os detectores individuais apresentaram melhores resultados em relação ao detector geral, à vista disso é a opção mais adequada para integrar o sistema, no caso de optar por melhor performance na detecção.

Esses dados demonstram, ainda, que para detecções mais precisas no primeiro estágio podem ser utilizadas os modelos individuais. No entanto, detector geral também se torna viável em situações especificas onde diferentes espécies de plantas estão presentes na mesma imagem, com isso, pode-se utilizar o detector geral, o qual possui somente uma pequena diminuição no desempenho em relação aos individuais.

Para ilustrar o funcionamento de um dos detectores individuais temos na Figura 27 um exemplo de resultado, utilizando uma imagem 1024x1024 como entrada, que após a rede devolver as coordenadas das folhas detectadas o sistema as recorta como imagens individuais.

Figura 27 – Saída do detector



Fonte: autoria própria

Nossos resultados são muito próximos a outros trabalhos da literatura que tratam de detecção de folhas utilizando Yolo, como pode ser comparado na Tabela 12.

Trabalho	Precisão	Recall	IoU	F1-Score
(SLADOJEVIC et al., 2016)	91% – 98%	90,53%	-	93%
(MOHANTY; HUGHES; SALATHÉ, 2016)	99,35%	97,32%	-	98,20%
(AMARA; BOUAZIZ; ALGERGAWY, 2017)	92% – 99%	92,88%	-	92,94%
(BARBEDO, 2019)	82% – 94%	85,49%	-	87,22%
(LU et al., 2017)	95,48%	93,38%	-	96,12%
(FERENTINOS, 2018)	95,48%	92,84%	-	96,78%
(LIU et al., 2017)	97,62%	-	-	-
(DECHANT et al., 2017)	96,7%	94%	-	97,23%
(WANG; SUN; WANG, 2017)	90,4%	-	-	-
(KC et al., 2019)	98,34%	96,38%	-	98,42%
(RAMCHARAN et al., 2019)	75% – 94% (mAP)	-	-	-
(SELVARAJ et	88,1%	87,5%	92,6%	86%

al., 2019)				
(FUENTES et al., 2018)	83% / 96% (mAP)	87%	91,30%	89%
Detectores individuais do nosso trabalho	86,97%	89,45%	85,33%	88%

Tabela 11 - Resultados dos Trabalhos Relacionados

Vale salientar que nos trabalhos relacionados listados, todos utilizam imagens capturadas em laboratório, o que pode afetar negativamente inferências no ambiente real, diferentemente da nossa proposta que mesclou dois *datasets*, um com ambiente de laboratório e outro no ambiente natural. Percebe-se que em relação a precisão, que nosso trabalho se encontro com a média abaixo da maioria dos outros estudos analisados que utilizam somente um estágio, mas em comparação ao outros dois que também utilizam dois estágios, (FUENTES et al., 2018) e (SELVARAJ et al., 2019), o nosso trabalho se encontra equiparado a eles. Em relação ao recall, nossos resultados se encontram na média em relação aos outros estudos para essa métrica. E em relação ao IoU, percebemos que somente os trabalhos que utilizam dois estágios apresentam essa métrica, já que ela é importante para detecção de objetos, o nosso trabalho se encontra com valor abaixo dos outros estudos, mas com resultado satisfatório.

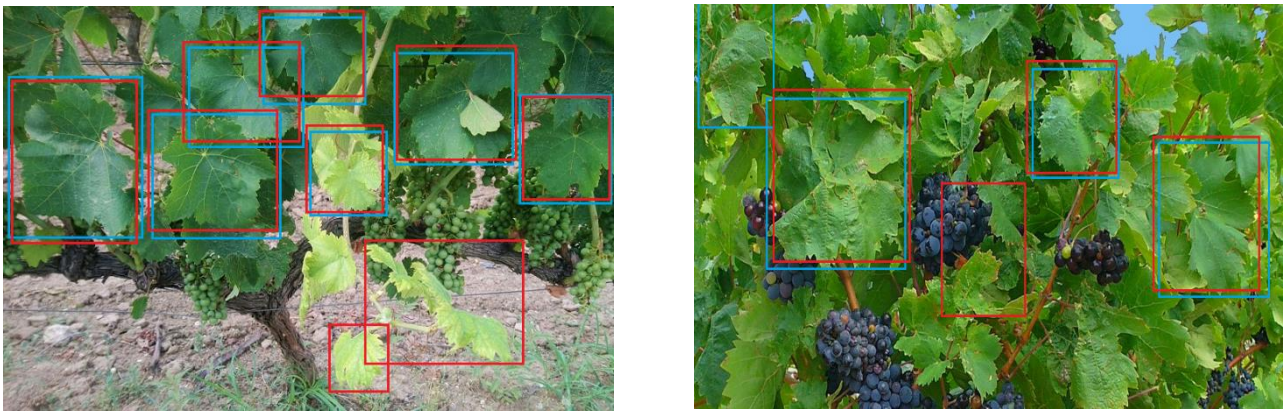
Outra informação muito relevante para nosso sistema, principalmente para a classificação de contexto que será apresentada na seção 5.3, é a quantidade de regiões detectadas. Essa quantidade é comparada com a quantidade de anotações de cada imagem do nosso conjunto de testes. Na Tabela 13 a seguir, é apresentado os resultados da quantidade de folhas detectadas em comparação com as das anotações. Percebe-se que a diferença em algumas classes é negativa, o que significa que nossa rede detectou mais elementos que os das anotações do *dataset*, seja por detecção equivocada ou por anotações incompletas. Além disso, temos um desvio padrão elevado o que significa que as distribuições das amostras estão muito dispersas.

	Maçã	**Pimentão**	**Milho**	**Batata**	**Tomate**	**Uva**
Qtd anotações	2936	1140	2201	1836	3348	1180
Qtd detectadas	2745	1113	2293	1746	3123	1227
Diferença	191	27	-92	90	225	-47
Diferença Total	672					
Média	112					
Desvio padrão	13,32					

Tabela 12 - Resultados da Quantidade de folhas detectadas em comparação com as das anotações

Para investigar tais diferenças, durante o teste do detector criamos uma lista com o identificador das imagens que apresentam quantidades de detecções acima das anotações para em seguida fazer inspeção visual. Para tanto, cada uma das imagens dessa lista passou por um processo de inclusão de *bounding boxings* em azul que representa a anotação do *dataset* e outros *bounding boxings* em vermelho que representam as detecções. Das 139 imagens que se encontram nessa situação, consideramos que 78 foram detecções equivocadas e 61 com anotações incompletas. A Figura 28(a) apresenta um exemplo de uma imagem do conjunto de teste que possui 7 anotações para folhas de uva e que nossa rede detectou 9 folhas, portanto com anotações incompletas. E a Figura 28(b) com detecções equivocadas.

Figura 28 – Exemplo de imagem com diferenças entre anotações e detecções: (a) anotações incompletas; (b) detecção equivocada



(a) (b)

A partir dessa análise, consideramos que 25% das detecções foram equivocadas e 28% por anotações incompletas, logo esse percentual superior de anotações incompletas compreende que o erro deve diminuir com a inclusão de novas anotações, porém, não podemos ignorar a porcentagem das detecções equivocadas que está com valores próximo, o que representa falhas no detector.

5.2 Segundo estágio - Classificação

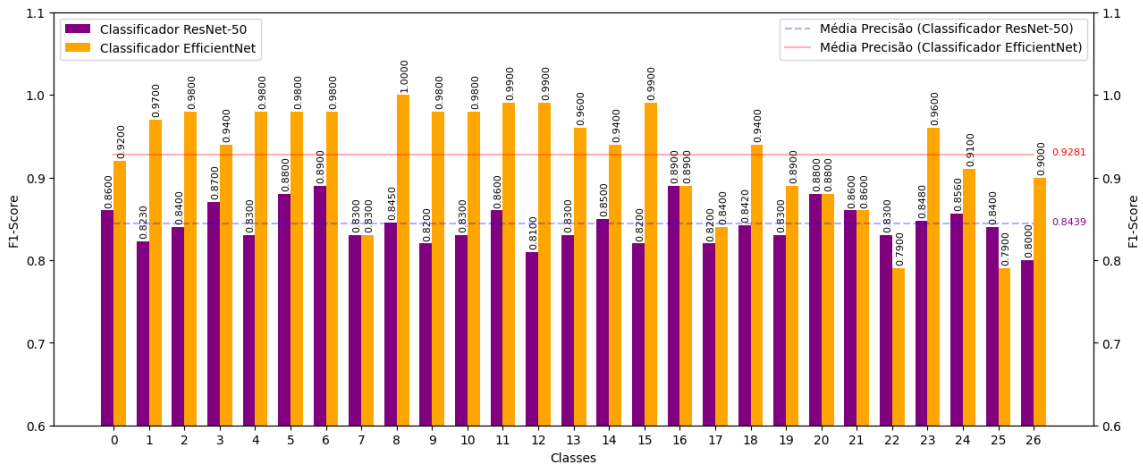
Para o treinamento desse segundo estágio foi realizado um pós-processamento com a

saída do primeiro estágio a fim de fornecer informações para realizar a classificação. Às imagens geradas no estágio de detecção juntamente com o arquivo de texto, presente nas imagens originais, que informa qual o tipo(s) de doença(s) se encontrava em algumas folhas detectadas ou se a folha está saudável, são as entradas desse estágio.

Logo, às imagens passam pelo redimensionamento para o tamanho de 224 x 224 pixels (o que representa a entrada da rede neural para a classificação).

Na Figura 29 podemos analisar a precisão das duas redes em relação a cada classes utilizando a base de dados *Single Leaves*, a qual foi gerada por meio do detector do primeiro estágio.

Figura 29 – Comparação entre os classificadores em relação a precisão



Fonte: autoria própria

Na Tabela 14 são apresentados os resultados da Precisão, Recall e F1-Score das redes ResNet-50 e EfficientNetB7 utilizando o *dataset Single Leaves* para cada classe.

Com esse novo *dataset* a rede EfficientNetB7 apresentou uma eficiência em média de 92,81% de precisão, uma melhoria de 6% em relação a sua versão utilizando o *dataset* original. Para este *dataset*, a rede ResNet-50 teve desempenho médio de 84,39% de precisão o que corresponde a uma melhora de 5% em relação a sua versão utilizando o *dataset* original. E comparando as duas redes, a EfficienteNetB7 ficou quase 10% superior a ResNet-50 em relação a precisão.

Classe	Precisão		Recall		F1-Score	
	ResNet-50	EfficientNetB7	ResNet-50	EfficientNetB7	ResNet-50	EfficientNetB7
Maçã com sarna	0.86	0.92	0.88	0.96	0.87	0.94
Maçã com podridão	0.82	0.97	0.84	0.98	0.83	0.98
Maçã com ferrugem	0.84	0.98	0.84	0.94	0.85	0.96
Maçã saudável	0.87	0.94	0.89	0.98	0.88	0.96
Pimentão com mancha	0.83	0.98	0.86	0.96	0.85	0.97
Pimentão saudável	0.88	0.98	0.91	0.98	0.90	0.95
Milho com cercosporiose	0.89	0.98	0.90	0.78	0.89	0.87
Milho com ferrugem	0.83	0.83	0.87	0.99	0.86	0.98
Milho com praga	0.84	1.00	0.89	0.98	0.87	0.90
Milho saudável	0.82	0.98	0.90	0.98	0.89	0.99
Uva com podridão	0.83	0.98	0.88	0.98	0.86	0.98
Uva com sarampo negro	0.86	0.99	0.89	0.97	0.87	0.98
Uva com praga	0.81	0.99	0.85	0.99	0.83	0.99
Uva saudável	0.83	0.96	0.89	0.99	0.85	0.99
Batata com queimadura	0.85	0.94	0.87	0.96	0.87	0.96
Batata com fungo	0.82	0.99	0.88	0.94	0.86	0.94
Batata com bacteria	0.89	0.89	0.92	0.98	0.84	0.98
Batata saudável	0.82	0.84	0.87	0.89	0.84	0.89
Tomate com queimadura	0.84	0.94	0.86	0.80	0.86	0.82
Tomate saudável	0.83	0.89	0.89	0.88	0.88	0.91
Tomate com fungo	0.88	0.88	0.91	0.89	0.90	0.89
Tomat com Mofo	0.86	0.86	0.89	0.72	0.87	0.79
Tomate com mancha negra	0.83	0.79	0.87	0.90	0.84	0.88
Tomate com ácaro	0.84	0.96	0.85	0.73	0.86	0.76
Tomate com pinta preta	0.86	0.91	0.89	0.94	0.88	0.95
Tomate com vírus	0.84	0.79	0.88	0.96	0.86	0.93
Tomate com folha amarela	0.80	0.90	0.85	0.96	0.86	0.87

Tabela 13 – Métricas de validação das redes ResNet-50 e EfficientNetB7

5.3 Classificação de Contexto

Com o primeiro e segundo estágio completos podemos interpretar as informações que existem em uma imagem utilizando o que as redes aprenderam e distinguir a área de interesse que é buscado, no caso as folhas e para realizar isso foi escolhido o melhor apresentado de cada estágio, para servir como estrutura para o sistema e, assim, avaliarmos.

Na Figura 30 podemos observar uma das informações de saída do sistema, onde é apresentado se a folha classificada apresenta alguma doença, o nome dessa doença e se a classificação foi efetuada de maneira correta. As informações abaixo das imagens na cor verde indicam que o sistema classificou corretamente, enquanto o texto em vermelho representa uma classificação equivocada, baseado nos rótulos do *dataset*.

Figura 30 – Saída do Sistema

Fonte: autoria própria

Na Figura 30 todas são folhas de batata e o sistema retorna um erro de classificação, em relação a classe, determinado na linha superior, e classifica corretamente todas as folhas em relação a presença de patógenos, determinado na linha inferior. Assim, o sistema analisa todas as informações, preditas e esperadas, para classificar o contexto da imagem original.

Para avaliar a classificação de contexto do sistema, ou seja, a partir dos dois estágios como um pipeline, utilizamos o mesmo conjunto de teste do nosso *dataset* original. Isso quer

dizer que entram as imagens de 1024 x 1024 que podem conter mais de uma folha no primeiro estágio, o qual detecta cada folha e devolve as coordenadas delas. O sistema recorta a imagem original e transforma em um vetor de “i” imagens, onde “i” indica a quantidade de folhas detectadas. Em seguida, cada imagem do vetor de folhas individuais é redimensionada para o formato 224 x 224 e enviada em sequência para o classificador. O resultado de classificação de cada folha é então contabilizado pelo sistema para calcular o percentual de cada classe indicada, indicado assim a classificação de contexto da imagem de entrada.

Dessa forma, analisamos o classificador de contexto em relação ao tempo de inferência e percentual de acertos das classificações. Os resultados dessa análise foram obtidos com a combinação das melhores versões de cada estágio**.**

Na Tabela 15 são apresentados os tempos médios para inferência de cada classe em segundos (s), na última coluna. Além disso, também apresentamos os tempos médios de cada etapa do sistema, na Tabela 16, de modo que possamos verificar os gargalos do sistema, os quais poderão ser alvo de melhorias em trabalhos futuros. A partir dos dados coletados, verificamos que com a soma desses tempos temos o tempo de inferência para 1 imagem de determinada classe, além disso, detectamos que a classe “Milho” apresenta tempo de resposta, utilizando os detectores individuais, 35% menor do que a média geral, em comparação a classe “Tomate” apresenta tempo de resposta, utilizando os mesmos detectores, 18% maior que a média geral.

Todas as classes apresentam tempos aceitáveis para a inferência de uma única imagem, contudo, quando a quantidade de imagens aumenta esse tempo passa ser um problema.

		Pré-processamento detector	Detecção	Pós-processamento detector	Média Classificação/folha	Média Classificação/imagem	Pós-processamento classificador	Subtotal
Maçã	**Detector Geral**	0,78	1,02	1,23	0,75	1,68	1,85	7,31
	Detectores Individuais	0,74	0,89	1,03	0,61	1,27	1,36	5,90
Pimentão	**Detector Geral**	0,79	1,06	1,17	0,81	1,62	1,79	7,24
	Detectores Individuais	0,73	0,94	1,09	0,74	1,13	1,30	5,93
Milho	**Detector Geral**	0,70	0,86	1,01	0,59	1,09	1,12	5,37
	Detectores Individuais	0,62	0,71	0,94	0,43	0,98	1,01	4,69
Batata	**Detector Geral**	0,76	0,84	1,35	0,81	1,39	1,56	6,71
	Detectores Individuais	0,72	0,81	1,19	0,72	1,26	1,38	6,08
Tomate	**Detector Geral**	0,92	1,24	1,39	0,91	1,83	1,95	8,24
	Detectores Individuais	0,87	1,19	1,14	0,82	1,67	1,81	7,50
Uva	**Detector Geral**	0,58	0,76	1,09	0,74	1,45	1,43	6,05
	Detectores Individuais	0,49	0,68	1,02	0,60	1,33	1,29	5,41

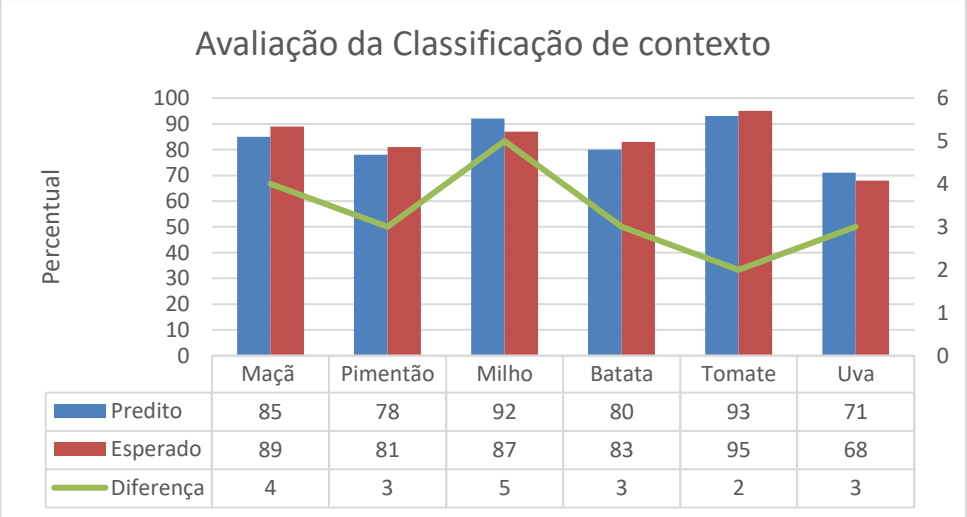
Tabela 14 – Tempo de Inferência das etapas do Sistema

Classes	Média por Classe
Maçã	6,60
Pimentão	6,58
Milho	5,03
Batata	6,39
Tomate	7,87
Uva	5,73
Média Geral	6,37

Tabela 15 – Média do tempo de inferência

Em relação ao percentual de acerto da classificação de contexto, o sistema leva em consideração a quantidade de folhas detectadas, mesmo que haja diferenças em relação as anotações, conforme discutido na seção 5.1. Dessa forma, o sistema de posse da classificação individual de cada folha e da quantidade de folhas detectadas indica o percentual de um determinado patógeno predito (uma das 27 classes). Esse percentual predito é comparado ao percentual esperado do conjunto de testes do *dataset* (considerando a mesma quantidade detectada). Os resultados de comparação são apresentados na Figura 31, com os valores preditos, reais e a diferença absoluta entre eles para cada classe.

Figura 31 – Diferença entre percentual predito e o percentual esperado do conjunto de teste



	Maçã	Pimentão	Milho	Batata	Tomate	Uva
Predito	85	78	92	80	93	71
Esperado	89	81	87	83	95	68
Diferença	4	3	5	3	2	3

Assim, para o conjunto de testes utilizados, a diferença total, considerando todas as classes, é de 20%, isso significa que o percentual de acerto da classificação de contexto será de 80%, com base na comparação do percentual predito e do percentual esperado, para todas as 27 classes.

6 CONSIDERAÇÕES FINAIS

O presente trabalho apresentou a criação de um sistema composto por um detector de folhas de plantas utilizando uma arquitetura YOLOv4 e um classificador utilizando uma rede neural, uma base de dados customizada fazendo uso de técnicas de data augmentation e anotação manual dos objetos de interesse.

Isso posto, conclui-se que o sistema proposto atingiu o propósito esperado de detectar objetos (folhas) em uma imagem, armazenando as coordenadas dos objetos detectados pelo sistema, classificar esses objetos e determinar se a imagem da planta original apresentava alguma doença ou não.

O primeiro estágio do sistema trás uma boa eficiência para o modelo implementado sobre o respectivo *dataset* construido, 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.

O segundo estágio evidencia uma média satisfatoria de classificação, o que torna o sistema como um todo eficiente em comparação aos trabalhos apresentados por (SELVARAJ et al., 2019) e (FUENTES et al., 2018), que trazem também a proposta de um sistema com dois estágio . Na Tabela 17 podemos observar a comparação do nosso trabalho com outros anteriormente mencionados.

Trabalho	Abordagem	Rede	Dataset	Descrição	Acurácia	Qtd de estágios
(SLADOJEVIC et al., 2016)	Classificação	CaffeNet	Próprio (4483 imagens)	13 classes de doenças em 5 espécies	91% – 98%	1
(MOHANTY; HUGHES; SALATHÉ, 2016)	Classificação	AlexNet e GoogLeNet	Plant Village	26 doenças em 14 espécies	99,35%	1
(AMARA; BOUAZIZ; ALGERGAWY, 2017)	Classificação	LeNet	Plant Village (3700 imagens)	2 doenças em banana	92% – 99%	1
(BARBEDO, 2019)	Classificação	GoogLeNet	Próprio	79 doenças em 14 espécies	82% – 94%	1
(LU et al., 2017)	Classificação	Baseado na AlexNet	Próprio (500 imagens)	10 doenças em arroz	95,48%	1
(FERENTINOS, 2018)	Classificação	Diversas CNNs	Plant Village	26 doenças em 14 espécies	95,48%	1
(LIU et al., 2017)	Classificação	Baseado na AlexNet	Próprio (13689 imagens)	4 doenças em maçãs	97,62%	1

(DECHANT et al., 2017)	Classificação	Diversas CNNs	Próprio (1796 imagens)	1 doença em milho	96,7%	1
(WANG; SUN; WANG, 2017)	Classificação	Diversas CNNs	Plant Village (2086 imagens)	1 doença em maçã (4 graus de severidade)	90,4%	1
(KC et al., 2019)	Classificação	MobileNet e VGG	Plant Village	55 classes (pares de espécie de doença)	98,34%	1
(RAMCHARAN et al., 2019)	Detecção	MobileNet	Próprio (2415 imagens)	7 doenças em mandioca (2 graus de severidade)	75% – 94% (mAP)	1
(SELVARAJ et al., 2019)	Detecção e Classificação	Faster R-CNN e VGG	Próprio	1 doença em pepino	86%	2
(FUENTES et al., 2018)	Detecção e Classificação	Faster R-CNN + (AlexNet, VGG-16, GoogLeNet, ResNet)	Próprio (5000 imagens)	9 doenças em tomate	83% / 96% (mAP)	2
Esse trabalho	Detecção e Classificação	Yolov4 e EfficientNetB7	Próprio	27 classes	93%	2

Tabela 16 - Resumo dos trabalhos baseados em aprendizado profundo para identificação de doenças em imagens de plantas

Podemos observar que em comparação com outros trabalhos utilizamos um *dataset* próprio, como a grande maioria, objetivando melhores resultados por meio da generalização. No que se refere a quantidade de estágios empregados no desenvolvimento dos trabalhos, os únicos que igualmente apresentam dois estágios desenvolvem o estudo utilizando uma quantidade menor no número de classes, ainda assim, o nosso trabalho apresenta melhores resultados quando comparando a acurácia.

Contudo, verifica-se algumas limitações no nosso trabalho, que nos conduzem a novas perspectivas para os trabalhos futuros. Dentre elas, as especificidades do dataset utilizado para o treinamento e a avaliação dos modelos, devido a insuficiência referente a diversidade de folhas e balanceamento delas. Desse modo, é valido realizar a construção de um dataset próprio melhor balanceado, preferencialmente, utilizando folhas da região.

Em um trabalho futuro, propomos substituir o estágio do classificador, que foi desenvolvido unicamente em *software*, por uma solução em *hardware* com objetivo de melhorar a eficiência do sistema, Tal solução já foi testada por outro aluno que participa do mesmo grupo de pesquisa, Lucas Gomes, que utilizando a mesma base de dados obteve resultados significativos.

REFERÊNCIAS

AHMAD, M. et al. Plant Disease Detection in Imbalanced Datasets Using Efficient Convolutional Neural Networks With Stepwise Transfer Learning. **IEEE Access**, v. 9, p. 140565–140580, 2021.

AMARA, J.; BOUAZIZ, B.; ALGERGAWY, A. **A Deep Learning-based Approach for Banana Leaf Diseases Classification**. Datenbanksysteme für Business, Technologie und Web. **Anais**...2017. Disponível em: <https://api.semanticscholar.org/CorpusID:38788670>

ARAGÃO, A.; CONTINI, E. **O agro no Brasil e no Mundo: uma síntese do período de 2000 a 2021**. Disponível em: <https://www.embrapa.br/documents/10180/26187851/O+agro+no+Brasil+e+no+mundo/098fc6c1-a4b4-7150-fad7-aaa026c94a40>. Acesso em: 19 jun. 2023.

BARBEDO, J. G. Digital image processing techniques for detecting, quantifying and classifying plant diseases. **SpringerPlus**, v. 2, n. 1, p. 660, 7 dez. 2013.

BARBEDO, J. G. Plant disease identification from individual lesions and spots using deep learning. **Biosystems Engineering**, v. 180, p. 96–107, abr. 2019.

BARBEDO, J. G. A. Digital image processing techniques for detecting, quantifying and classifying plant diseases. **SpringerPlus**, p. 19–33, 2018a.

BARBEDO, J. G. A. Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification. **Computers and Electronics in Agriculture**, v. 153, p. 46–53, out. 2018b.

BIANCO, S. et al. Benchmark Analysis of Representative Deep Neural Network Architectures. **IEEE Access**, v. 6, p. 64270–64277, 2018.

BOCHKOVSKIY, A.; WANG, C.-Y.; LIAO, H.-Y. M. YOLOv4: Optimal Speed and Accuracy of Object Detection. 22 abr. 2020.

BOCK, C. H. et al. Plant Disease Severity Estimated Visually, by Digital Photography and Image Analysis, and by Hyperspectral Imaging. **Critical Reviews in Plant Sciences**, v. 29, n. 2, p. 59–107, 10 mar. 2010.

DECHANT, C. et al. Automated Identification of Northern Leaf Blight-Infected Maize Plants from Field Imagery Using Deep Learning. **Phytopathology®**, v. 107, n. 11, p. 1426–1432, nov. 2017.

FERENTINOS, K. P. Deep learning models for plant disease detection and diagnosis. **Computers and Electronics in Agriculture**, v. 145, p. 311–318, fev. 2018.

FUENTES, A. et al. A Robust Deep-Learning-Based Detector for Real-Time Tomato Plant Diseases and Pests Recognition. **Sensors**, v. 17, n. 9, p. 2022, 4 set. 2017.

FUENTES, A. F. et al. High-Performance Deep Neural Network-Based Tomato Plant Diseases and Pests Diagnosis System With Refinement Filter Bank. **Frontiers in Plant Science**, v. 9, 29 ago. 2018.

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

GIRSHICK, R. et al. **Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation**. 2014 IEEE Conference on Computer Vision and Pattern Recognition. **Anais**...IEEE, jun. 2014.

GIRSHICK, R. **Fast R-CNN**. 2015 IEEE International Conference on Computer Vision (ICCV). **Anais**...IEEE, dez. 2015.

HE, K. et al. **Deep Residual Learning for Image Recognition**. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). **Anais**...IEEE, jun. 2016.

HINTON, G. E.; OSINDERO, S.; TEH, Y.-W. A Fast Learning Algorithm for Deep Belief Nets. **Neural Computation**, v. 18, n. 7, p. 1527–1554, jul. 2006.

HOWARD, A. G. et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications. **arXiv preprint arXiv:1704.04861**, 2017.

HUANG, G. et al. **Densely Connected Convolutional Networks**. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). **Anais**...IEEE, jul. 2017.

HUGHES, D.; SALATHÉ, M.; OTHERS. An open access repository of images on plant health to enable the development of mobile disease diagnostics. **arXiv preprint arXiv:1511.08060**, 2015.

IMEA. **Relatório de Mercado**. Disponível em: <: https://www.imea.com.br/imea-site/relatoriosmercado.>. Acesso em: 18 jun. 2023.

JOCHER, G. et al. **Yolov5**.

KAMILARIS, A.; PRENAFETA-BOLDÚ, F. X. Deep learning in agriculture: A survey. **Computers and Electronics in Agriculture**, v. 147, p. 70–90, abr. 2018.

KC, K. et al. Depthwise separable convolution architectures for plant disease classification. **Computers and Electronics in Agriculture**, v. 165, p. 104948, out. 2019.

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

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. **ImageNet Classification with Deep Convolutional Neural Networks**. Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1. **Anais**...: NIPS'12.Red Hook, NY, USA: Curran Associates Inc., 2012.

LECUN, Y. et al. Gradient-based learning applied to document recognition. **Proceedings of the IEEE**, v. 86, n. 11, p. 2278–2324, 1998.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **Nature**, v. 521, n. 7553, p. 436–444, 28 maio 2015.

LEFÈVRE, A. et al. Challenges of complying with both food value chain specifications and agroecology principles in vegetable crop protection. **Agricultural Systems**, v. 185, p. 102953, nov. 2020.

LIU, B. et al. Identification of Apple Leaf Diseases Based on Deep Convolutional Neural Networks. **Symmetry**, v. 10, n. 1, p. 11, 29 dez. 2017.

LU, Y. et al. Identification of rice diseases using deep convolutional neural networks. **Neurocomputing**, v. 267, p. 378–384, dez. 2017.

MAO, Q.-C. et al. Mini-YOLOv3: Real-Time Object Detector for Embedded Applications. **IEEE Access**, v. 7, p. 133529–133538, 2019.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. **The Bulletin of Mathematical Biophysics**, v. 5, n. 4, p. 115–133, dez. 1943.

MELLO, R.; PONTI, M. Machine_Learning. [s.d.].

MOHANTY, S. P.; HUGHES, D. P.; SALATHÉ, M. Using Deep Learning for Image-Based Plant Disease Detection. **Frontiers in Plant Science**, v. 7, 22 set. 2016.

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

NGUGI, L. C.; ABELWAHAB, M.; ABO-ZAHHAD, M. Recent advances in image processing techniques for automated leaf pest and disease recognition – A review. **Information Processing in Agriculture**, v. 8, n. 1, p. 27–51, mar. 2021.

PONTI, M. A. et al. **Everything You Wanted to Know about Deep Learning for Computer Vision but Were Afraid to Ask**. 2017 30th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T). **Anais**...IEEE, out. 2017.

RAHIMZADEH, M.; ATTAR, A.; SAKHAEI, S. M. 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, jul. 2021.

RAMCHARAN, A. et al. A Mobile-Based Deep Learning Model for Cassava Disease Diagnosis. **Frontiers in Plant Science**, v. 10, 20 mar. 2019.

REDMON, J. et al. You Only Look Once: Unified, Real-Time Object Detection. 8 jun. 2015.

REDMON, J.; FARHADI, A. **YOLO9000: better, faster, stronger**. Proceedings of the IEEE conference on computer vision and pattern recognition. **Anais**...2017.

REDMON, J.; FARHADI, A. Yolov3: An incremental improvement. **arXiv preprint arXiv:1804.02767**, 2018.

REN, S. et al. Faster r-cnn: Towards real-time object detection with region proposal networks. **Advances in neural information processing systems**, v. 28, 2015.

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

ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. **Psychological Review**, v. 65, n. 6, p. 386–408, 1958.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. **Nature**, v. 323, n. 6088, p. 533–536, out. 1986.

SANKARAN, S. et al. A review of advanced techniques for detecting plant diseases. **Computers and Electronics in Agriculture**, v. 72, n. 1, p. 1–13, jun. 2010.

SELVARAJ, M. G. et al. AI-powered banana diseases and pest detection. **Plant Methods**, v. 15, n. 1, p. 92, 12 dez. 2019.

SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. **arXiv preprint arXiv:1409.1556**, 2014.

SINDIVEG. **Levantamento dos principais números do setor de agrotóxicos no Brasil**. Disponível em:

<https://sindiveg.org.br/mercado-total>. Acesso em: 18 jun. 2023.

SKALSKI, P. **Make Sense**.

SLADOJEVIC, S. et al. Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification. **Computational Intelligence and Neuroscience**, v. 2016, p. 1–11, 2016.

SZEGEDY, C. et al. **Going deeper with convolutions**. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). **Anais**...IEEE, jun. 2015.

TAN, M.; LE, Q. V. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. 28 maio 2019.

UIJLINGS, J. R. R. et al. Selective Search for Object Recognition. **International Journal of Computer Vision**, v. 104, n. 2, p. 154–171, 2 set. 2013.

UNEP. **How to Feed the World in 2050**. Disponível em: <http://www.fao.org/fileadmin/templates/wsfs/docs/expert_paper/How_to_Feed_the_World_in_2050.pdf.>. Acesso em: 19 jun. 2023.

WANG, G.; SUN, Y.; WANG, J. Automatic Image-Based Plant Disease Severity Estimation Using Deep Learning. **Computational Intelligence and Neuroscience**, v. 2017, p. 1–8, 2017.

ZEILER, M. D.; FERGUS, R. **Visualizing and understanding convolutional networks**. Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13. **Anais**...2014.

ZHANG, Q. et al. Wheat yield losses from pests and pathogens in China. **Agriculture, Ecosystems & Environment**, v. 326, p. 107821, mar. 2022.

ZHAO, Z.-Q. et al. Object Detection With Deep Learning: A Review. **IEEE Transactions on Neural Networks and Learning Systems**, v. 30, n. 11, p. 3212–3232, nov. 2019.

ZHENG, Z. et al. Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression. 19 nov. 2019.