INSTITUTO FEDERAL DE SANTA CATARINA

WELLINGTON JOÃO GONÇALVES

ESTUDO DA UTILIZAÇÃO DE VISÃO COMPUTACIONAL NA IDENTIFICAÇÃO DE FRUTAS UTILIZANDO VARREDURA POR DRONE

Joinville - SC

Fevereiro/2025

ESTUDO DA UTILIZAÇÃO DE VISÃO COMPUTACIONAL NA IDENTIFICAÇÃO DE FRUTAS UTILIZANDO VARREDURA POR DRONE

Trabalho de conclusão de curso apresentado à Coordenadoria do Curso de Engenharia Elétrica Campus Joinville do Instituto Federal de Santa Catarina para a obtenção do diploma de Engenheiro Eletricista.

Orientador: Stefano Romeu Zeplin, Msc.

Joinville - SC

Fevereiro/2025

Gonçalves, Wellington João.

Estudo da utilização de visão computacional na identificação de frutas utilizando varredura por drone / Wellington João Gonçalves – Joinville, SC, 2025.
50 p.

Trabalho de Conclusão de Curso (Graduação) - Instituto Federal de Educação Ciência e Tecnologia de Santa Catarina, Curso de Bacharelado em Engenharia Elétrica, Joinville, 2025.
Orientador: Stefano Romeu Zeplin.

1. Visão Computacional. 2. Identificação de Frutas. 3. Drone Dj Tello. 4. Yolov8. I. Instituto Federal de Educação Ciência e Tecnologia de Santa Catarina. II. Título.

WELLINGTON JOÃO GONÇALVES

ESTUDO DA UTILIZAÇÃO DE VISÃO COMPUTACIONAL NA IDENTIFICAÇÃO DE FRUTAS UTILIZANDO VARREDURA POR DRONE

Este trabalho foi julgado adequado para obtenção do título de Engenheiro Eletricista, pelo Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina Campus Joinville, e aprovado na sua forma final pela comissão avaliadora abaixo indicada.

Joinville - SC, 06 de fevereiro de 2025:

Stefano Romeu Zeplin, Mr.Eng
Orientador
Instituto Federal de Santa Catarina

Michael Klug, Dr.Eng
Instituto Federal de Santa Catarina

Joice Luiz Jeronimo, Mr.Eng
Instituto Federal de Santa Catarina

Lucas Raniere Gonçalves

Lucas Raniere Gonçalves, Eng

Com muito carinho, dedico este trabalho à minha família,
que esteve ao meu lado em todos os momentos difíceis,
sempre me motivando e acreditando em mim.

AGRADECIMENTOS

Primeiramente, agradeço a Deus, cujo seu amor tornou tudo isso possível. Agradeço também à minha esposa pela compreensão e apoio durante as inúmeras noites de estudo. Aos meus filhos, que sempre me alegram e me motivam a seguir em frente. Agradeço aos meus pais, ao meu orientador, Stefano Romeu Zeplin, assim como aos demais professores, pela paciência e ensinamentos ao longo dessa jornada. E, finalmente, agradeço aos meus amigos pelas boas risadas e pelos bons momentos compartilhados.

RESUMO

Este trabalho de conclusão de curso investiga a aplicação de técnicas de visão computacional na identificação de frutas por meio de varredura com drones, especificamente utilizando o modelo Tello. O estudo destaca a crescente relevância da tecnologia de drones na agricultura, enfatizando sua capacidade de coletar dados visuais em larga escala e em tempo real. A pesquisa inclui uma revisão dos conceitos teóricos fundamentais, algoritmos e técnicas pertinentes, com foco na implementação do algoritmo YOLO (You Only Look Once) para a detecção de frutas.

Para a realização dos experimentos, foram desenvolvidos procedimentos para o controle do drone e aquisição de imagens e vídeos, que foram posteriormente processados em um ambiente de computação em nuvem. A coleta de dados foi realizada em ambientes controlados, permitindo uma análise comparativa da eficácia da identificação de frutas em diferentes condições. Os resultados esperados visam não apenas contribuir para o avanço do conhecimento na área de visão computacional, mas também demonstrar a viabilidade da tecnologia YOLO como uma ferramenta eficiente e versátil na agricultura, promovendo melhorias em termos de rapidez, precisão e segurança na identificação de frutas.

Palavras-chave: Visão Computacional, Identificação de Frutas, Drone Dj Tello, YOLOv8.

ABSTRACT

This thesis investigates the application of computer vision techniques for the identification of fruits through drone scanning, specifically using the Tello model. The study highlights the growing relevance of drone technology in agriculture, emphasizing its ability to collect visual data on a large scale and in real-time. The research includes a review of fundamental theoretical concepts, relevant algorithms, and techniques, focusing on the implementation of the YOLO (You Only Look Once) algorithm for fruit detection.

To conduct the experiments, procedures were developed for controlling the drone and acquiring images and videos, which were subsequently processed in a cloud computing environment. Data collection was performed in controlled environments, allowing for a comparative analysis of the effectiveness of fruit identification under different conditions. The expected results aim not only to contribute to the advancement of knowledge in the field of computer vision but also to demonstrate the viability of YOLO technology as an efficient and versatile tool in agriculture, promoting improvements in terms of speed, accuracy, and safety in fruit identification.

Keywords: Computer Vision, Fruit Identification, Drone DJ Tello, YOLOv8

LISTA DE ILUSTRAÇÕES

Figura 1 – Filtros em imagens 21
Figura 2 – Imagem equalizada 22
Figura 3 – Processos utilizando Pooling 24
Figura 4 – Exemplo de uma rede totalmente conectada 24
Figura 5 – Exemplo de uma rede neural convolucional e suas camadas 25
Figura 6 – Gráfico da eficiência do YOLO 25
Figura 7 – Processamento do algorítimo YOLO 26
Figura 8 – Exemplo de imagem com supressão não máxima 27
Figura 9 – Detecção de frutas e sua probabilidade da classe 27
Figura 10 – Arquitetura YOLO 29
Figura 11 – Drone tello dji 30
Figura 12 – Processamento de imagens - Câmeras 31
Figura 13 – Geometria da imagem aérea com drone 32
Figura 14 – Software para controle do drone 33
Figura 15 – Triangulo Retângulo 34
Figura 16 – Plano de teste 34
Figura 17 – Matriz de Confusão 36
Figura 18 – LabelImg - Anotação de imagens 38
Figura 19 – Analise modelo YOLOv8 - Identificação 39
Figura 20 – Analise modelo YOLOv8X - Rotação 40
Figura 21 – Analise modelo yoloV8X - Rotação imagem 2 40
Figura 22 – Analise modelo yoloV8X - Desfoque em 10% e 15% 41
Figura 23 – Analise modelo yoloV8X - Desfoque em 25% e 30% 41
Figura 24 – Analise modelo yoloV8X - oclusão 42
Figura 25 – Analise modelo yoloV8 42
Figura 26 – F1 vs confidence curve - comparativo entre modelos do YOLOv8 43
Figura 27 – Precision vs Confidence curve - comparativo entre modelos do YOLOv8 44
Figura 28 – Precision vs Recall Curve - comparativo entre modelos do YOLOv8 44
Figura 29 – Recall vs Confidence - comparativo entre modelos do YOLOv8 45
Figura 30 – Matriz confusão - comparativo entre modelos do YOLOv8 45
Figura 31 – Processametno YOLOv8 - GPU vs CPU 46

LISTA DE TABELAS

Tabela 1 – Plano de teste 35
Tabela 2 – Coordenadas de anotação vs. caixa delimitadora prevista 39
Tabela 3 – Métricas YOLOv8X vs YOLOv8N 46

LISTA DE ABREVIATURAS E SIGLAS

RNA Redes Neurals Artificials 22

CNN Redes Neurais Convolucionais 23

FAO Organização das Nações Unidas para a Alimentação e Agricultura 19

ONU Organização das Nações Unidas 19

YOLO You Only Look Once 19

Embrapa Empresa Brasileira de Pesquisa Agropecuária 19

ReLU Unidade de Retificação Linear 23

GSD Ground Sample Distance 32

SUMÁRIO

1 INTRODUÇÃO . . . 19
1.1 Objetivo geral . . . 20
1.2 Objetivo específico . . . 20

2 DESENVOLVIMENTO . . . 21
2.1 Revisão de literatura . . . 21
2.1.1 Visão computacional . . . 21
2.1.1.1 Baixo Nível . . . 21
2.1.1.2 Médio Nível . . . 22
2.1.1.3 Alto Nível . . . 22
2.1.2 Algoritmo de aprendizagem profunda em visão computadorizada . . . 22
2.1.3 Redes Neurais Convolucionais . . . 23
2.1.3.1 Camada Convolucional . . . 23
2.1.3.2 Camada Pooling . . . 23
2.1.3.3 Camada totalmente conectada . . . 24
2.1.4 YOLO – You Only Look Once . . . 25
2.1.4.1 Darknet . . . 28
2.1.4.2 COCO dataset . . . 28
2.1.4.3 Arquitetura YOLO v1 . . . 28
2.1.4.4 YOLO v8 . . . 29
2.1.5 Google Colaboratory . . . 29
2.1.6 Drone tello DJI . . . 30
2.1.7 Câmera . . . 30
2.1.8 Geometria de imagens aérea . . . 31

3 METODOLOGIA . . . 33
3.1 Coleta de dados . . . 33
3.1.1 Aquisição de imagens com o drone . . . 33
3.2 Processamento dos dados . . . 35
3.3 Indicadores de desempenho . . . 35
3.4 LabelImg . . . 37

4 ANÁLISE E DISCUSSÃO DOS RESULTADOS . . . 39
4.1 Analise das Métricas . . . 42
4.1.0.1 Matriz de confusão . . . 45
4.1.1 Processamento de dados utilizando GPU e CPU . . . 46

5 CONCLUSÃO . . . 47

REFERÊNCIAS . . . 49

1 INTRODUÇÃO

A produção global de frutas tem desempenhado um papel crucial na segurança alimentar da população mundial, que atingirá aproximadamente 8,2 bilhões de pessoas em 2024, de acordo com dados das Organização das Nações Unidas (ONU), (ONU, 2024). Segundo a Organização das Nações Unidas para a Alimentação e Agricultura (FAO), a produção mundial de frutas em 2020 alcançou aproximadamente 1.375 milhões de toneladas, representando um aumento de 17% em relação à década anterior.

Nesse cenário, o Brasil desempenha um papel fundamental tanto no abastecimento interno quanto nas exportações, ocupando o terceiro lugar no ranking mundial, ficando atrás da China e da Índia. Juntos, esses três países respondem por cerca de 45,9% da produção mundial de frutas.

De acordo com a Empresa Brasileira de Pesquisa Agropecuária (Embrapa), o Brasil produziu aproximadamente 59 milhões de toneladas de frutas em 2021, representando cerca de 5,4% da produção global. Essa produção abrange uma ampla variedade de frutas, incluindo banana, laranja, maçã, uva, manga e diversas outras (EMBRAPA, 2021).

Sendo a agricultura fundamental na alimentação da população global, e uma parte essencial desse setor é a produção de frutas, a identificação precisa e eficiente das frutas em pomares e plantações junto com a detecção de doenças e pragas, é um desafio enfrentado pelos agricultores, uma vez que a inspeção manual requer tempo, recursos humanos e pode ser suscetível a erros.

Com o avanço da tecnologia em visão computacional, tornou-se possível executar uma avaliação não invasiva de atributos como aparência, qualidade e volume. Nesse contexto, a utilização de visão computacional surge como uma solução promissora para automatizar e aprimorar esse processo.

A visão computacional é uma área da inteligência artificial que se dedica ao processamento e análise de imagens e vídeos (FORSYTH; PONCE, 2003). Ela oferece a capacidade de extrair informações valiosas a partir desses dados visuais, permitindo a detecção, segmentação e classificação de objetos de interesse.

Os algoritmos empregados em visão computacional são de aprendizagem profunda, baseados em redes neurais artificiais. Essas redes são compostas por camadas de neurônios artificiais que processam as informações de entrada e produzem saídas que podem ser interpretadas como uma classificação ou detecção de objetos (FUJIYOSHI; HIRAKAWA; YAMASHITA, 2019).

Com o uso de algoritmos e técnicas avançadas, a visão computacional tem sido aplicada em uma ampla gama de domínios, incluindo medicina, automação industrial e agricultura. Uma das técnicas avançadas utilizadas neste projeto será o You Only Look Once (YOLO).

O algoritmo YOLO é uma técnica de visão computacional que pode ser aplicada para identificação de objetos em tempo real. Esse algoritmo é baseado em redes neurais convolucionais. Ele foi projetado para realizar a detecção e classificação de objetos com alta precisão e rapidez (REDMON et al., 2016).

Com o avanço da tecnologia, uma solução eficaz para auxiliar a agricultura é a utilização de drone. Esses dispositivos são capazes de capturar imagens e vídeos de alta resolução, enquanto algoritmos avançados de visão computacional como o YOLO podem processar esses dados identificando e classificando objetos de interesse.

Com esse objetivo, este trabalho tem o intuito de investigar a aplicação da visão computacional na identificação de frutas utilizando drone Tello como plataforma de varredura. Serão explorados conceitos

teóricos, algoritmos e técnicas relevantes para essa abordagem, bem como sua viabilidade e eficácia em diferentes cenários.

Para atingir esses objetivos, serão realizados experimentos práticos utilizando drones equipados com câmeras e sistemas de visão computacional. Serão coletados dados de objetos em ambientes controlados, a fim de comparar os resultados obtidos na identificação de objetos.

Espera-se que este estudo contribua para o aumento do conhecimento no campo da identificação de objetos, demonstrando os benefícios da utilização de visão computacional com o auxílio de drones. Além disso, espera-se que os resultados obtidos possam impulsionar a adoção dessa tecnologia como uma ferramenta eficiente e versátil para a identificação de objetos em diferentes setores, proporcionando maior rapidez, precisão e segurança.

1.1 Objetivo geral

Estudar a aplicação de visão computacional na identificação de frutas, utilizando imagens captadas pelo drone DJI Tello e processadas pelo algoritmo YOLOv8.

1.2 Objetivo específico

Para a elaboração do trabalho, alguns conceitos serão vistos posteriormente, sendo necessário contemplar alguns pontos como:

- Estudo dos conceitos de visão computacional;

- Estudo dos conceitos de redes neurais convolucionais;

- Estudo do algoritmo YOLO;

- Aplicação do algoritmo YOLO e identificar frutas pré definidas;

- Controlar e obter imagens a partir de um drone tello;

- Análise dos resultados quantitativos e qualitativos obtidos na experimentação do algoritmo;

2 DESENVOLVIMENTO

2.1 Revisão de literatura

2.1.1 Visão computacional

Para os seres humanos, a visão é o sentido mais avançado, porém somos limitados à banda visual do espectro eletromagnético. Já as máquinas conseguem cobrir essa limitação. Nesse contexto, é de suma importância o estudo e desenvolvimento de algoritmos e técnicas para permitir que as máquinas adquirem e interpretem imagens e vídeos de forma semelhante aos seres humanos. Em outras palavras, podemos definir a visão computacional sendo o processo que as máquinas analisam e compreendem o conteúdo visual de imagens e vídeos, incluindo a identificação de objetos, pessoas, rostos, movimentos, gestos e outros elementos visuais (GONZALEZ; WOODS, 2010).

Conforme elucidado por (MARENGONI; STRINGHINI, 2019), podemos dividir o espectro que vai do processamento de imagens até a visão computacional em três níveis: baixo-nível, nível-médio e alto-nível.

2.1.1.1 Baixo Nível

O baixo nível engloba o pré-processamento de imagens. Nessa etapa, o foco é melhorar a qualidade das imagens aplicando técnicas com o objetivo de remover ruídos indesejados e destacar as regiões de interesse (MARENGONI; STRINGHINI, 2010). Entre elas podemos destacar a aplicação de filtros.

Os métodos de filtragem de imagens são normalmente classificados em duas categorias, as técnicas de filtragem espacial e as técnicas de filtragem no domínio da frequência.

Os processos que atuam no domínio espacial realizam cálculos diretamente sobre a matriz de pixel, normalmente utilizando operações de convolução com máscaras. Por outro lado, podemos descrever os métodos que atuam no domínio da frequência como sendo baseados na modificação da transformada de Fourier da imagem. Essas técnicas podem ser utilizadas em conjunto, para garantir uma análise robusta dos dados (FILHO; NETO, 1999).

Esses filtros são eficazes para realçar detalhes e melhorar a nitidez, redução de ruídos e podem ser aplicados para suavizar áreas indesejadas ou remover informações de alta frequência, como demonstrado na Figura 1.

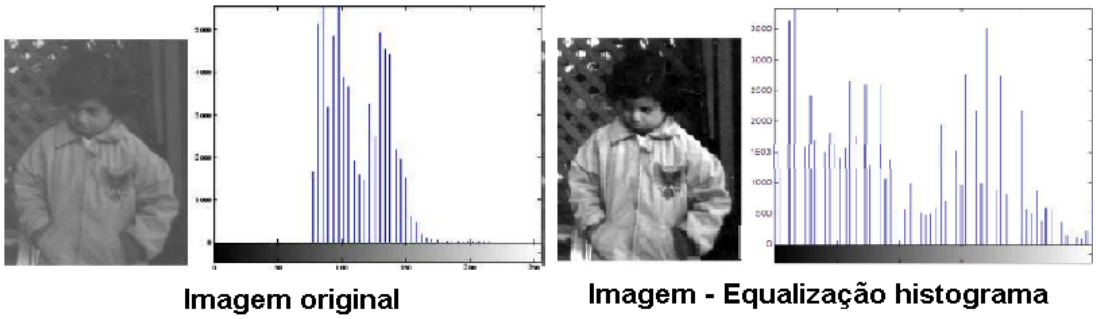
Figura 1 – Filtros em imagens

Fonte:(FELGUEIRAS, 2005). Modificado pelo Autor

Outra técnica comum no pré-processamento de imagens é a normalização do histograma. Ela

visa equalizar a distribuição de intensidade na imagem, ajustando os níveis de contraste e melhorando a visualização das informações presentes (MARENGONI; STRINGHINI, 2019). Podemos visualizar essa técnica na imagem Figura 2. Esse processo é particularmente útil quando a imagem possui um intervalo limitado de intensidades e é necessário ampliar sua faixa dinâmica.

Figura 2 – Imagem equalizada



Fonte: (BEZERRA, 2005). Modificado pelo Autor

2.1.1.2 Médio Nível

Nessa etapa, são realizadas operações do tipo segmentação e reconhecimento. O processo de segmentação consiste em dividir uma imagens por regiões. Esse processo leva em conta características da região, por exemplo a cor ou tamanho do objeto (MARENGONI; STRINGHINI, 2010). Um dos processos mais simples de segmentação consiste em separar os objetos, mudando a cor destes para preta e o fundo para branca, tem-se então uma imagem binária, que é muito usual nas diversas aplicações de visão computacional.

2.1.1.3 Alto Nível

Os Processos de alto-nível, estão relacionados com as tarefas de cognição associadas com a visão humana. Esses processos incorporam a validação da satisfação dos dados obtidos, estimativa de parâmetros sobre a imagem e classificação dos objetos obtidos em diferentes categorias (GONZALEZ; WOODS, 2010).

Existem duas áreas principais de metodologias de reconhecimento. Uma área utiliza descritores quantitativos, como comprimento, área e textura, para realizar a comparação. A outra área utiliza descritores qualitativos para realizar a comparação (GONZALEZ; WOODS, 2010).

Outros importantes avanços recentes na área incluem as redes neurais convolucionais, que têm permitido uma melhoria significativa no desempenho de tarefas de visão computacional, e o uso de técnicas de aprendizado profundo para reconhecimento de objetos em imagens e vídeos.

2.1.2 Algoritmo de aprendizagem profunda em visão computadorizada

Os algoritmos de aprendizagem profunda são uma classe de algoritmos de inteligência artificial que permitem que as máquinas aprendam a partir de dados sem serem explicitamente programados. Geralmente são baseados em Redes Neurais Artificiais (RNA) inspiradas no funcionamento do cérebro humano, que possui milhões de neurônios interconectados colaborando na aprendizagem e processamento de informações. Essas redes são compostas por camadas de neurônios artificiais que processam as informações de entrada e produzem saídas que podem ser interpretadas como uma classificação ou detecção de objetos (FUJIYOSHI; HIRAKAWA; YAMASHITA, 2019).

A partir de 2010, o aprendizado profundo ganhou proeminência, especialmente no domínio dos algoritmos de visão, resultando no desenvolvimento das Redes Neurais Convolucionais (CNN). Essa abordagem inovadora possibilitou o uso das redes para extrair características de imagens por meio de aprendizado e, subsequentemente, para realizar detecções. Esse avanço tornou o processo mais robusto, uma vez que as características cruciais para a detecção são determinadas pela própria rede (FUJIYOSHI; HIRAKAWA; YAMASHITA, 2019).

2.1.3 Redes Neurais Convolucionais

As redes neurais convolucionais, (*Convolutional Neural Network*) CNN, foram criadas para resolverem uma tarefa específica, o reconhecimento de imagens (TECH, 2020). Elas são semelhantes às redes neurais tradicionais, contendo neurônios que possuem pesos e *bias* que necessitam ser treinados. Para as camadas de convolução das CNN são necessários apenas definir a arquitetura dos filtros, quantidade, tamanhos e stride (VARGAS; PAES; VASCONCELOS, 2016).

O processo de aprendizado das CNN altera os pesos ao longo do treinamento, até encontrar os melhores valores dos filtros para o conjunto de dados utilizado. Uma distinção significativa nas CNN é a habilidade de desenvolver filtros n-dimensionais (VARGAS; PAES; VASCONCELOS, 2016).

Uma CNN é formada por uma série de camadas. Além da camada de entrada, que geralmente é uma imagem com largura, altura e profundidade, há três principais: a camada convolucional, a camada de pooling e a camada totalmente conectada (PACHECO, 2019).

2.1.3.1 Camada Convolucional

A parte mais densa do processamento computacional está na camada convolucional. Nesse processo a rede consegue apreender utilizando conjuntos de filtros (ou *kernels*) que são treinados utilizando várias técnicas de processamento de imagens .

Podemos definir os *kernels* como matrizes que possuem valores reais interpretados como pesos na rede. Esses filtros se alteram durante o treinamento, ajudando a rede a identificar as melhores características da imagem. Após esse processo, essas matrizes passam por uma técnica de convolução com os dados de entrada gerando um mapa de características (*features*) (PACHECO, 2019).

O processo de convolução é definido através do produto escalar entre uma região da imagem do tamanho do filtro e o próprio filtro. Esse filtro se desloca sobre as imagens respeitando o valor definido no *stride*, movendo tanto para o lado como para baixo até percorrer toda a imagem. Para cada movimento do filtro o processo de cálculo é realizado novamente, gerando uma saída tridimensional com os dados mais relevantes para cada filtro (PACHECO, 2019).

Após o processo de convolução e pooling, é comumente utilizada uma função denominada Unidade de Retificação Linear (ReLU), que consiste em aplicar a função Max(0, x) a cada elemento do resultado da convolução. Essa característica reduz significativamente a quantidade de pesos da CNN em comparação com uma RNA. Essas camadas, quando colocadas em sequência (ou empilhadas), forma uma arquitetura de uma CNN.(PACHECO, 2019).

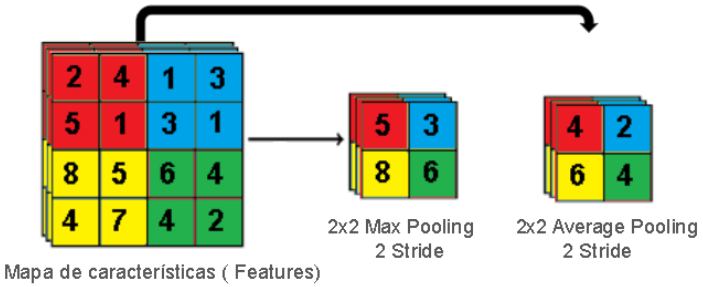
2.1.3.2 Camada Pooling

O pooling é uma técnica crucial para preservar as características mais relevantes e otimizar a eficiência computacional do modelo.

Existem diversos tipos de pooling, sendo os mais comuns o Max Pooling, onde é selecionado o valor máximo de um conjunto de valores em uma região específica do mapa de características preservando as

características dominantes, como bordas ou texturas. Já o Average Pooling calcula a média dos valores em uma determinada região do mapa de características. O Average Pooling é útil para suavizar a representação, reduzindo o impacto de ruídos ou detalhes irrelevantes e contribuindo para uma representação mais generalizada. É importante mencionar que a camada de pooling não reduz a profundidade da entrada, ela apenas reduz a altura e largura de um mapa (VARGAS; PAES; VASCONCELOS, 2016). Na Figura 3, demonstra o processo utilizando o Max pooling e Average Pooling.

Figura 3 – Processos utilizando Pooling



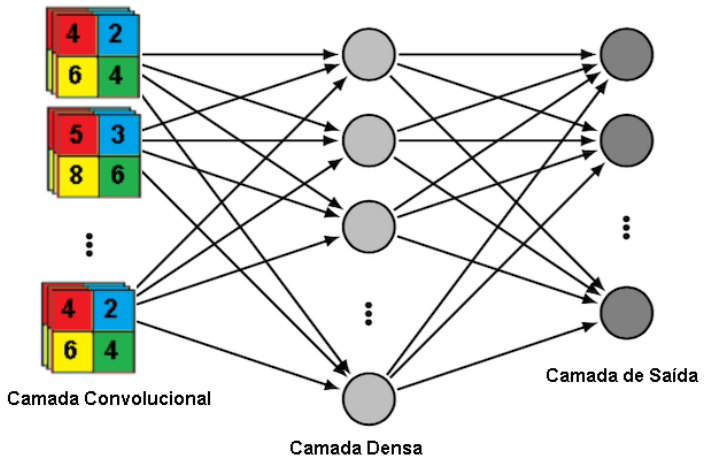
Fonte: (KALMBACH, 2021). Modificado pelo Autor.

2.1.3.3 Camada totalmente conectada

Uma rede neural totalmente conectada , também conhecida como rede neural densa , é uma arquitetura na qual cada neurônio em uma camada está conectado a todos os neurônios na camada seguinte. Essa estrutura é frequentemente empregada nas últimas camadas das CNN para realizar tarefas como classificação, regressão ou qualquer outra que demande a combinação de características extraídas.

As redes neurais densas são ferramentas poderosas em aprendizado profundo, amplamente utilizadas em diversas aplicações, desde reconhecimento de imagens e processamento de linguagem natural até jogos e robótica (VARGAS; PAES; VASCONCELOS, 2016). Um exemplo de uma rede totalmente conectada pode ser observado na Figura 4.

Figura 4 – Exemplo de uma rede totalmente conectada

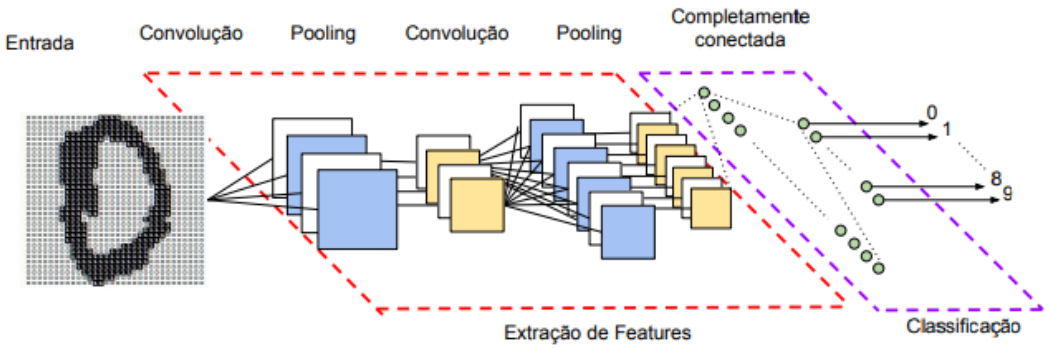


Fonte: (NETO; BONINI, 2014). Modificado pelo Autor.

Embora as redes neurais sejam capazes de aprender características abstratas a partir de dados brutos, o treinamento desses algoritmos pode ser complexo e exigir grandes quantidades de dados e recursos computacionais. É necessário um grande conjunto de dados rotulados para o treinamento do

modelo, além de ajustes finos e validação rigorosa para garantir a eficácia e a confiabilidade dos resultados. Na Figura 5, é possível visualizar as etapas de uma CNN.

Figura 5 – Exemplo de uma rede neural convolucional e suas camadas



Fonte:(VARGAS; PAES; VASCONCELOS, 2016).

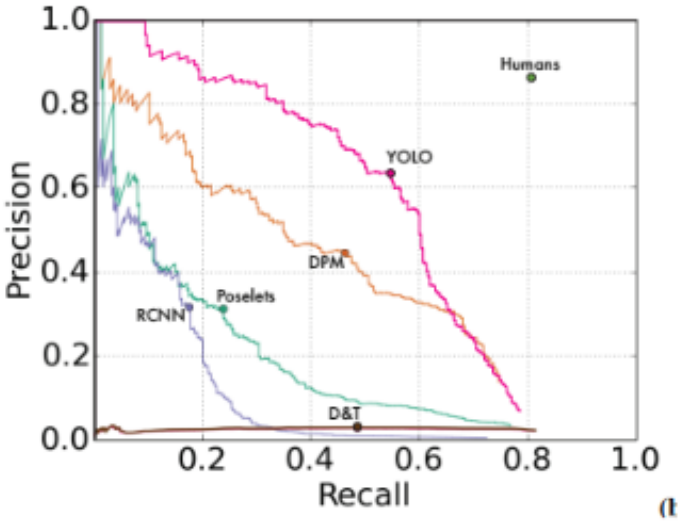
2.1.4 YOLO – You Only Look Once

O YOLO é uma ferramenta de visão computacional que foi desenvolvida por Joseph Redmon e Ali Farhadi em 2015 durante o seu doutorado. Essa ferramenta, proporciona a detecção de objetos analisando imagens e vídeos, com rapidez, eficiência e precisão. (REDMON et al., 2016).

A velocidade de detecção de objetos do YOLO se dá pela maneira que ele processa as imagens. O algoritmo processa a imagem em uma única etapa, diferentemente das demais ferramentas (REDMON et al., 2016).

Podemos verificar a eficiência na Figura 6 logo abaixo:

Figura 6 – Gráfico da eficiência do YOLO



Fonte:(REDMON et al., 2016).

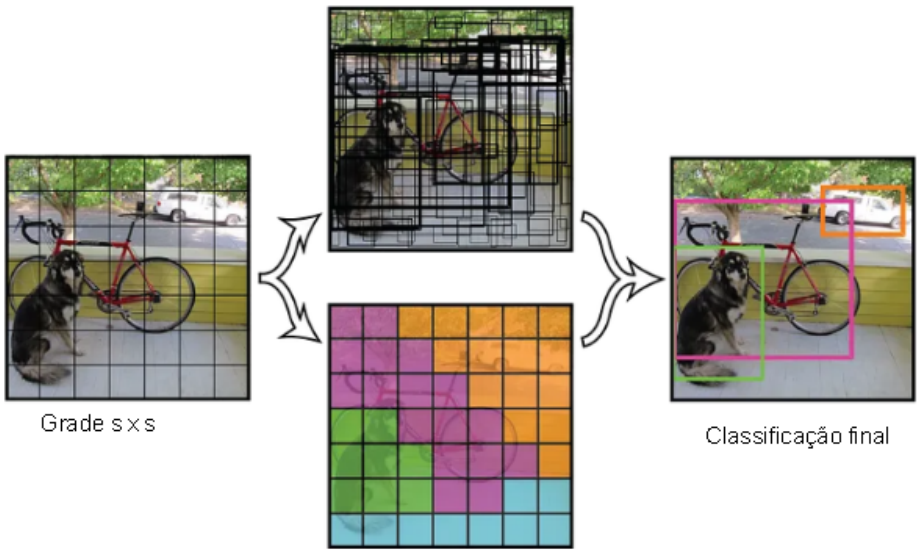
O YOLO utiliza uma única CNN para processar a imagem inteira. Nesse processo a imagem de entrada é dividida em uma grade de dimensões S×S. Cada célula dessa grade é responsável por prever

um conjunto fixo de caixas delimitadoras (*bounding boxes*) e as respectivas probabilidades de classe dos objetos presentes naquela região específica.

Em seguida, é aplicado um filtro para selecionar as caixas delimitadoras mais prováveis e eliminar as detecções redundantes(REDMON et al., 2016).

O processo para detecção de objetos em imagem é observado na imagem Figura 7.

Figura 7 – Processamento do algorítimo YOLO



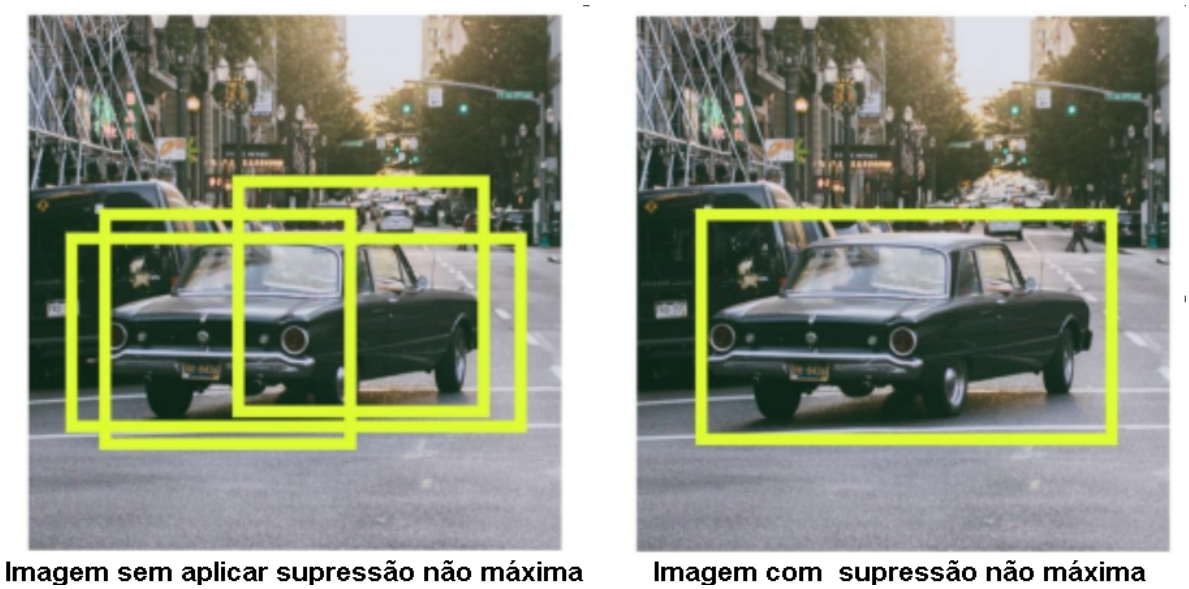
Fonte: (REDMON et al., 2016). Modificado pelo Autor.

Para cada célula da grade, são geradas caixas delimitadoras que contém as seguintes variáveis:

- Confiança indica o grau de certeza do modelo de que a caixa delimitadora contém um objeto.
- x e y representam as coordenadas do centro da caixa detectada em relação às bordas da célula da grade onde a previsão foi feita.
- w e h correspondem à altura e à largura da caixa em relação à imagem completa.
- Probabilidades das classes para cada caixa. Esse string sempre vem em conjunto com o valor da confiança.

Nem todas as caixas contém um objeto em seu interior. Por isso, aplica-se uma técnica chamada Supressão Não Máxima (Non Max-Suppression). Essa técnica é executada fora da CNN, eliminando as caixas delimitadoras com as menores probabilidades de conter um objeto e combinar aquelas que sobrepõem a mesma região, como ilustrado na Figura 8.

Figura 8 – Exemplo de imagem com supressão não máxima

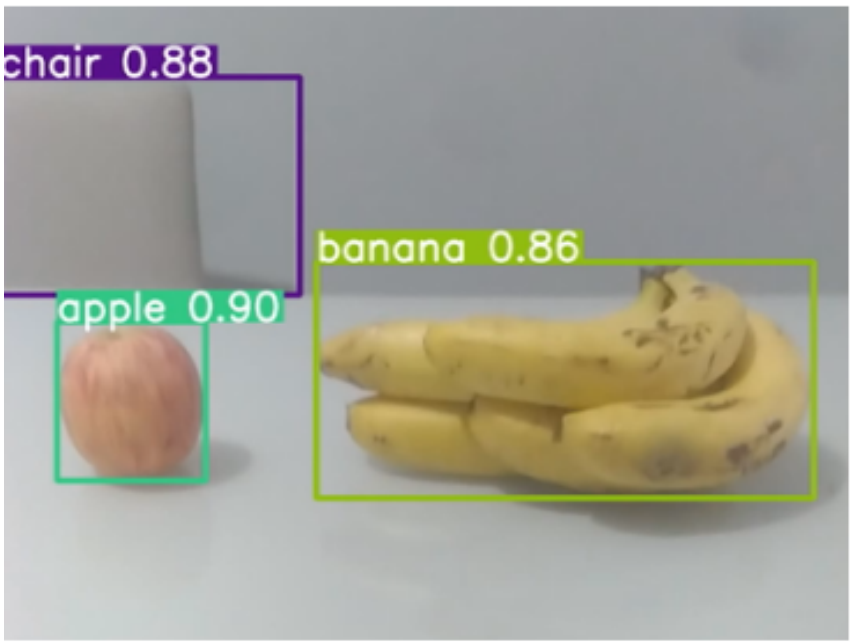


Fonte:(JAIN; NANDY, 2019). Modificado pelo Autor

Em seguida, o valor de confiança, que pode variar de 0 até 1, é atribuído à caixa delimitadora e combinado com a probabilidade da classe. Esse resultado mostra a pontuação final da detecção.

Uma maneira de controlarmos a confiança das detecções é com o parâmetro chamado *threshold*. O *threshold* representa o limite mínimo para considerar uma detecção válida. Podemos ver na Figura 9 as caixas delimitadoras junto com o valor de confiança e a classe dos objetos. Nota-se que se o *threshold* estivesse configurado para um valor de 0,9 o objeto chair não teria sido detectado.

Figura 9 – Detecção de frutas e sua probabilidade da classe



Fonte: O Autor

2.1.4.1 Darknet

A Darknet é uma estrutura de rede neural profunda escrita em C e CUDA. É rápida, fácil de instalar e suporta computação de CPU e GPU. Ela foi criada com foco em simplicidade e eficiência, especialmente para tarefas de visão computacional como a detecção de objetos. Sua arquitetura foi projetada para ser altamente otimizada para treinamento e inferência rápida de redes neurais convolucionais CNN, que são a base do YOLO.A Darknet serve como o back-end que define e treina a arquitetura de redes neurais convolucionais usadas no YOLO.(REDMON et al., 2016).

2.1.4.2 COCO dataset

O COCO (Common Objects in Context) é um dos principais datasets empregados na área de visão computacional. Ele foi desenvolvido e mantido pelo time da Microsoft, servindo como uma base sólida para os treinamentos dos modelos de aprendizado profundo. Esse dataset possui uma ampla gama de imagens.

2.1.4.3 Arquitetura YOLO v1

O formato da arquitetura utilizada nas redes YOLO consiste em três partes principais: o backbone (espinha dorsal), que obtém características e captura informações relevantes das imagens de entrada através de camadas de convolução; o neck (pescoço), parte intermediária da arquitetura que recebe as características extraídas pelo backbone e aplica transformações para melhorar a representação dos dados e refinar as informações; e a head (cabeça), que recebe as características aperfeiçoadas pelo neck e toma as decisões para gerar as saídas finais da rede, como as caixas delimitadoras e probabilidades de classe dos objetos detectados.

A primeira versão do YOLO era formada pelas seguintes estruturas:

- Entrada (Input), a imagem começa com uma entrada de 448x448, que é a dimensão da imagem original.

- Camadas Convolucionais (Conv. Layers), cada camada convolucional tem dimensões específicas para os filtros, representadas como "7x7x64", "3x3x192", entre outros, onde "7x7"é o tamanho do filtro e o último dígito representa o número de filtros aplicados.

- Camadas de Pooling (Maxpool Layers), para as camadas de pooling, descrita como "2x2-2"indica um filtro de pooling 2x2 com stride (passo) de 2, o que significa que a dimensão da imagem é reduzida pela metade em cada direção.

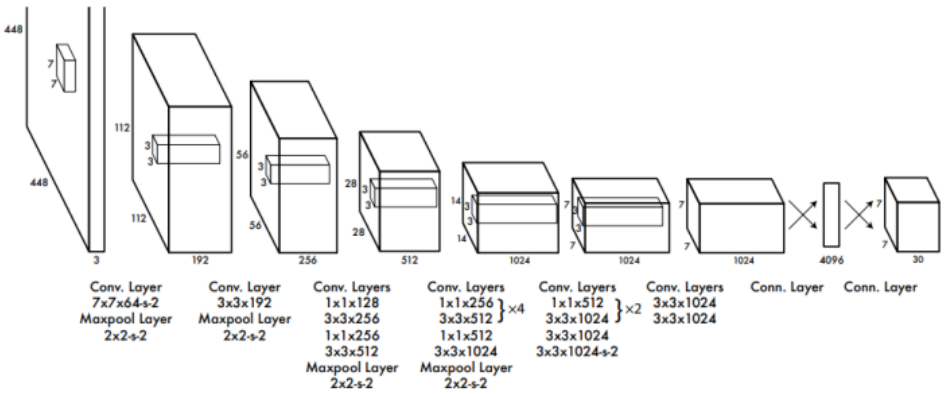
- Redução Progressiva de Dimensões, podemos observar que através das camadas as dimensões das imagens de entrada são progressivamente reduzidas (por exemplo, de 448x448 para 112x112, depois para 56x56, e assim por diante), enquanto o número de filtros aumentar.

- Camadas Totalmente Conectadas (Fully Connected Layers), no final, a rede possui camadas totalmente conectadas, onde cada neurônio está conectado a todos os neurônios da camada anterior. Isso permite a combinação das características extraídas pelas camadas convolucionais para tomar decisões finais (por exemplo, classificar a imagem).

- Camada de Saída, finalmente, a camada de saída geralmente corresponde ao número de classes que a rede está tentando prever.

Na Figura 10, é possível verificar a estrutura da arquitetura do YOLO.

Figura 10 – Arquitetura YOLO



Fonte:(REDMON et al., 2016).

2.1.4.4 YOLO v8

A versão v8 do YOLO, desenvolvida pela Ultralytics, demonstrou alguns aprimoramentos significativos em comparação com suas antecessoras.

A arquitetura YOLO v8 oferece uma ampla variedade de modelos pré-treinados com diferentes características. Dependendo da aplicação desejada, essas características podem ser ajustadas, variando a velocidade ou a precisão. Os modelos disponíveis incluem YOLOv8n (Nano), YOLOv8s (Small), YOLOv8m (Medium), YOLOv8l (Large) e YOLOv8x (Extra Large) (ULTRALYTICS, 2024).

Analisando os extremos dos modelos pré-treinados do YOLOv8 temos o modelo YOLOv8x (Extra Large). Esse modelo é o maior do pacote desenvolvido para o YOLOv8. Ele é utilizado quando se deseja uma alta precisão, porém o tempo de processamento computacional é maior (ULTRALYTICS, 2024).

Na outra ponta, encontra-se o modelo YOLOv8n (Nano). Esse modelo é focado no tempo de processamento dos dados, reduzindo a capacidade da precisão. O modelo é o mais leve dos pacotes para o YOLOv8 (ULTRALYTICS, 2024).

Comparando o yoloV8 com as versões anteriores podemos destacar as melhorias:

- Arquiteturas avançadas de backbone e de neck: Para o YOLOv8, a empresa Ultralytics aprimoraram a arquitetura da espinha dorsal e de pescoço, o que levou a um aumento no desempenho da extração de características, aumentando a detecção de objetos (ULTRALYTICS, 2024).

- Cabeça dividida Ultralytics sem âncoras: Esta técnica consiste em melhorar a precisão e a eficiência do processo de detecção de objetos.

Comparando com versões anteriores, que dependiam de âncoras, ou seja, caixas delimitadoras pré-definidas espalhadas pela imagem com a finalidade de predizer a localização e a classe dos objetos, o YOLOv8, com a melhoria do desenvolvimento do algoritmo, eliminou a dependência de âncoras estáticas (ULTRALYTICS, 2024).

- Compensação optimizada entre precisão e velocidade: Como os vários modelos disponíveis, é possível manter um equilíbrio ótimo entre precisão e velocidade.

2.1.5 Google Colaboratory

O Google Colaboratory (Google Colab) é uma plataforma com acesso gratuito a recursos computacionais, incluindo GPUs e TPUs. Essa plataforma oferece uma ampla variedade de ferramentas para o

desenvolvimento de pesquisa na área de aprendizado de máquina (COLAB, 2024).

2.1.6 Drone tello DJI

O drone Tello DJI é um drone estudantil de baixo custo que foi desenvolvido pela empresa chinesa Ryze Tech. Ele conta com um processador Intel Movidius Myriad e é conhecido por possibilitar o controle através de uma linguagem de programação. Ele possui uma câmera frontal responsável pela aquisição de imagens e vídeos (RYZE, 2024). O modelo tello conta com as seguintes características:

- Câmera: 5 MP (2.592x1.936);

- Qualidade do vídeo: HD (720p);

- Formato: JPG (foto); MP4 (vídeo)

- FOV: 82,6°

- Autonomia da bateria: 13 min;

- Alcance: 100 m;

- Altura máxima de voo: 30 m

- Velocidade máxima: 28 km/h.

Podemos visualizar o drone Tello na Figura 11.

Figura 11 – Drone tello dji

Fonte:(RYZE, 2024).

2.1.7 Câmera

Para aquisitar as imagens utilizamos uma câmera, onde o seu princípio de funcionamento envolve os elementos descritos a seguir:

- O primeiro elemento que podemos destacar em uma câmera é a lente. A lente é formada por várias peças contendo o elemento vidro que auxiliam a focalizar e ajustar a luz que entra na câmera permitindo que a imagem seja projetada em um sensor de imagem (SIMãO, 2020).

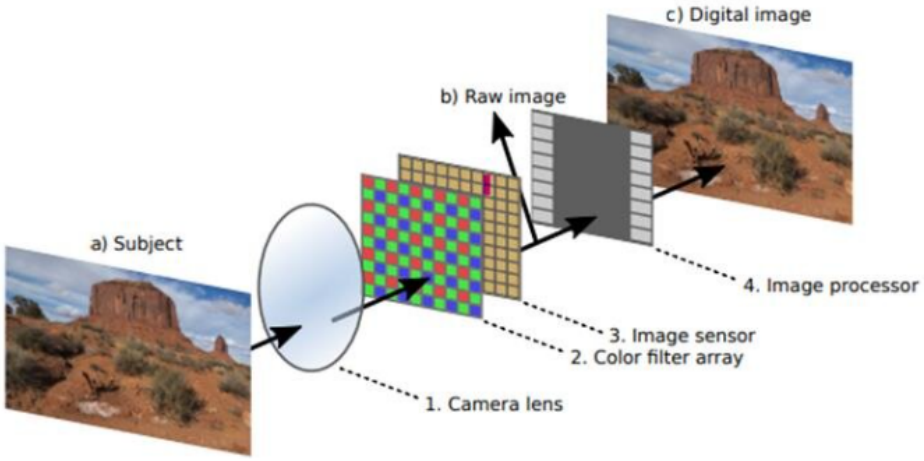
- Após a passagem da luz pela lente, a luz passa pelo array do filtro de cores. Junto com o array de cores está o sensor de imagem. Esse sensor converte a luz que passou pelo filtro de cores em uma imagem digital. Esse circuito eletrônico é composto por vários fotodetectores que tem a função de

mensurar a quantidade de luz e converter em sinal eletrônico. Os sinais eletrônicos posteriormente são processados pelo processador de imagem.

- O processador de imagem é responsável por todas as informações do sensor de imagem. Nesse processo é ajustado todas as variáveis da imagem como brilho, nitidez e saturação das cores, gerando uma imagem de alta qualidade.

  Podemos observar o esquema de processamento de imagens na Figura 12.

Figura 12 – Processamento de imagens - Câmeras



Fonte:(SALMAN; KALAKECH, 2024). Modificado pelo Autor

2.1.8 Geometria de imagens aérea

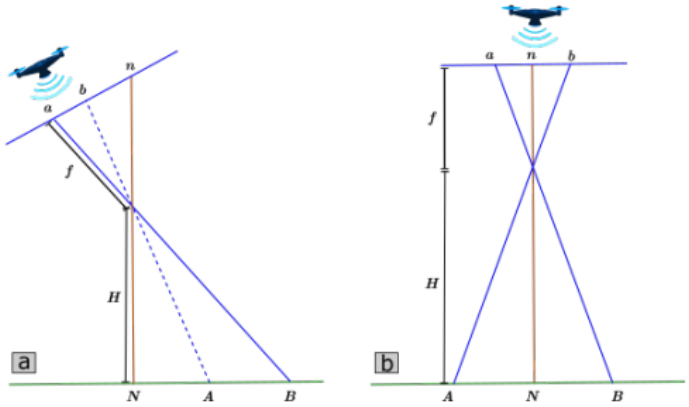
Compreender a geometria de uma imagem aérea é crucial, pois os componentes envolvidos na aquisição de imagens determinam a escala ou a resolução espacial da imagem. Podemos descrever o processo de aquisição de uma imagem aérea utilizando princípios trigonométricos. Nessa análise verificamos a reflexão da luz solar ao entrar em contato com um objeto no solo, parte dessa luz acaba sendo refletida e é capturada pela lente da câmera (VERAMENDI, 2022).

Podemos classificar as imagens aéreas de três formas: verdadeira, verticalmente inclinada e imagem aérea oblíqua. Essa classificação leva em conta o ângulo óptico da câmera em relação ao ângulo perpendicular ao objeto em estudo, esse ângulo é chamado de nadir (VERAMENDI, 2022). A classificação das imagens segue a seguinte variação:

- Imagens aéreas verdadeiras, é formada quando o ângulo óptico da câmera está $\pm\ 0^{o}$ em relação ao nadir.
- Imagens aéreas verticalmente inclinadas, é denominada quando o ângulo óptico da câmera está 0^{o} e $\pm\ 3^{o}$ em relação ao nadir.
- Imagens aéreas oblíqua, é composta por qualquer imagem que apresenta um ângulo óptico da câmera maior que $\pm\ 3^{o}$.

A Figura 13 descreve a classificação das imagens aéreas comparando o ângulo nadir . Na Figura 13-A temos a aquisição de uma imagem oblíqua e na Figura 13-B uma imagem verdadeira.

Figura 13 – Geometria da imagem aérea com drone



Fonte:(VERAMENDI, 2022).

Uma forma de termos uma relação da escala das imagens aéreas e a distância do objeto de estudo comparando ao solo é o Ground Sample Distance (GSD). O GSD calcula a resolução espacial das imagens aéreas adquiridas pelo drone, ou seja, a resolução do objeto visível no solo. Simplificadamente, o GSD representa o tamanho de cada pixel no solo no momento em que a imagem é registrada.

Vale ressaltar que, conforme a definição padrão na óptica de lentes, a distância focal (f) é medida em milímetros (mm), enquanto a altura do voo (H) é medida em metros (m). O GSD é definido pela seguintes equações:

$$GSD = \frac{Alturadevoo * Tamanhodopixelnosensor}{Distânciafocaldacâmera} \tag{2.1}$$

Como o tamanho do pixel não é diretamente fornecido para a câmera do tello DJI, podemos fazer uma aproximação com base na resolução da imagem e no campo de visão.

$$W = 2 \times \text{Altura de voo} \times \tan\left(\frac{\text{FOV horizontal}}{2}\right) \tag{2.2}$$

Conforme descrito pelo fabricante, podemos considerar 82,6° para o FOV horizontal do drone tello DJI. Com esses dados podemos calcular o GSD das imagens adquiridas pelo drone.

$$\text{GSD} = \frac{\text{Largura da área coberta (W)}}{\text{Resolução da imagem em pixels}} \tag{2.3}$$

Como todas as imagens adquiridas pelo drone serão oblíqua, o GSD pode ser usado para calcular o GSD oblíquo correspondente em um determinado ângulo de inclinação, multiplicando este valor pela Taxa GSD conforme descrito na Equação 2.4.

$$\text{GSD rate} = \frac{1}{\cos(\text{theta} + \text{phi})} \tag{2.4}$$

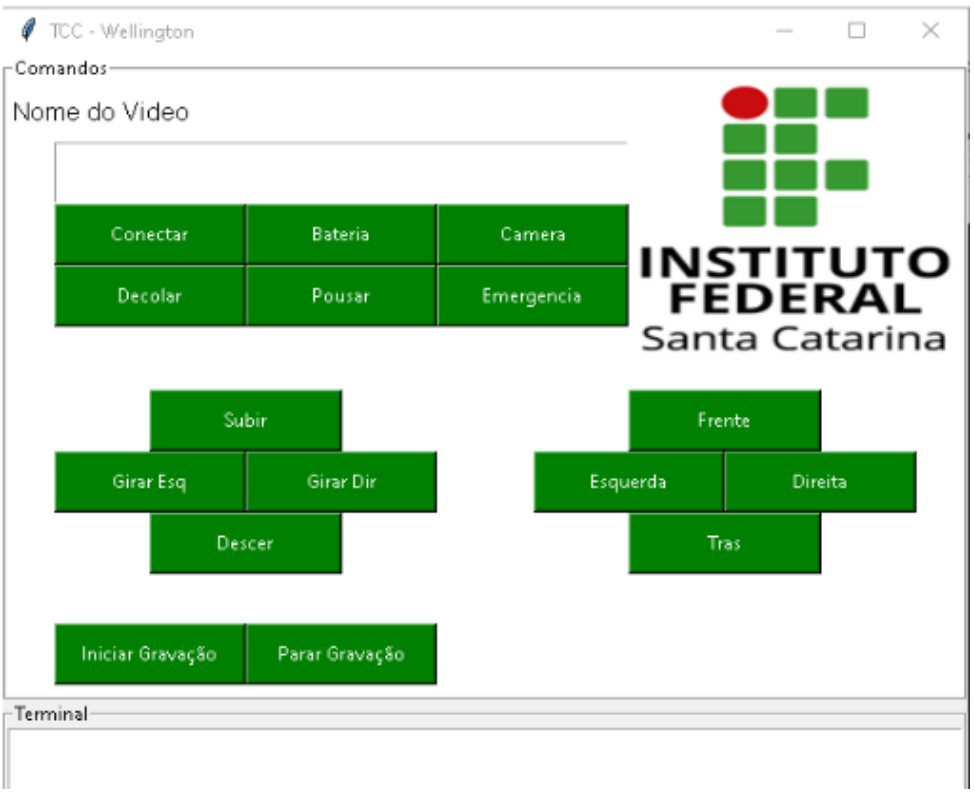
Imagens com GSD baixo geram arquivos maiores e mais detalhados, o que requer mais capacidade de armazenamento e processamento de dados, mas melhora a precisão da análise. Entretanto imagens com GSD alto são mais leves, mas perdem detalhes importantes para a detecção de objetos menores.Tendo isso em mente, vale salientar que a altura de voo do drone influencia diretamente o GSD. Quanto mais alto o drone voa, maior será o GSD e, portanto, menor a resolução da imagem.

3 METODOLOGIA

Esta seção descreve as etapas utilizadas para o estudo da identificação de frutas utilizando varredura com drones, especificamente o modelo Tello.

Para isso, foi desenvolvido um software em Python utilizando o manual criado pela Ryze e o canal do João Reis (DJI Tello, Controlando o drone usando Python) disponível no youtube. A função inicial do software incluía tanto o controle do drone quanto o processamento simultâneo das imagens capturadas. Contudo, devido a limitação de hardware, o computador utilizado não possui uma GPU, o escopo do software foi ajustado, sendo utilizado exclusivamente para controlar e adquirir imagens e vídeos do drone. Todo o código-fonte e os programas desenvolvidos estão disponíveis no seguinte repositório do GitHub: https://github.com/Wellington-wjg/Software-Drone-Dji-tello. Na Figura 14 podemos visualizar o front-end do software utilizado no projeto.

Figura 14 – Software para controle do drone



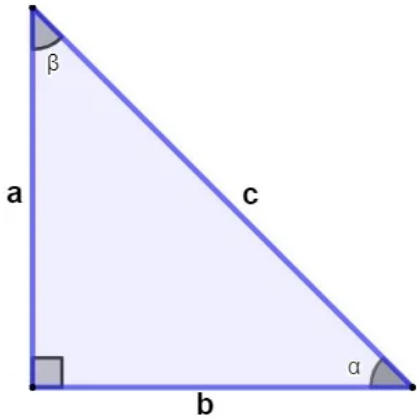
Fonte: O Autor

3.1 Coleta de dados

3.1.1 Aquisição de imagens com o drone

Devido a localização da câmera do drone, o ângulo alfa máximo para captura de imagens ou vídeos é 35°, porém para o trabalho foi escolhido como ângulo alfa máximo de 30°. Esse dado foi escolhido utilizando medições realizadas em campo e para o cálculo foi empregado a trigonometria do ângulo retângulo demostrado na Figura 15 e descrito nas equações a seguir.

Figura 15 – Triangulo Retângulo



Fonte: O Autor

Os dados adquiridos em campo foram de a=70cm e b=100cm.

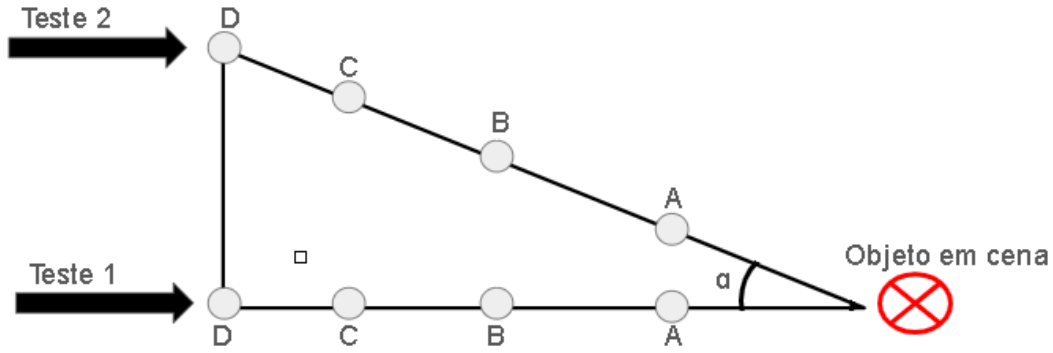
$$alfa = \tan^{-1}\left(\frac{\mathrm{a}}{\mathrm{b}}\right) \tag{3.1}$$

$$alfa = \tan^{-1}\left(\frac{70}{100}\right) \tag{3.2}$$

$$alfa \approx 35° \tag{3.3}$$

Seguindo essa limitação de hardware, foi elaborado um plano de teste como descrito na Figura 16 com o objetivo de verificar o desempenho da utilização das imagens geradas pelo drone.

Figura 16 – Plano de teste



Fonte: O Autor

Podemos verificar as distâncias dos testes na Tabela 1 a seguir.

Tabela 1 – Plano de teste

Teste	Ponto	Distância[cm]	Altura[cm]
Teste 1	A	50	5
Teste 1	B	100	5
Teste 1	C	150	5
Teste 1	D	200	5
Teste 2	A	50	29
Teste 2	B	100	58
Teste 2	C	150	87
Teste 2	D	200	116

Fonte: O Autor

3.2 Processamento dos dados

Após a coleta de dados, partimos para o processamento no colab, onde limitamos o algoritmo a identificar apenas três classes. Laranja, maçã e banana. No diretório do colab se encontram todas as funções para a análise dos dados adquiridos. Após a análise dos dados com o modelo Yolo.V8x, os dados foram submetidos à análise do pacote YoloV8n para verificar suas diferenças.

3.3 Indicadores de desempenho

As tarefas de localização, segregação e classificação podem ser abordadas com eficiência utilizando o YOLO v8. A importância de avaliar o desempenho desses modelos é destacada pela vasta quantidade de pesquisas publicadas sobre o tema. Esses trabalhos frequentemente exploram alternativas, como o desenvolvimento de novas métricas de avaliação ou a comparação de um conjunto de métricas existentes, para avaliar a eficácia dos algoritmos.(DEMŠAR, 2006).

Quando o objetivo é classificar mais de duas classes, como no caso de uma matriz de confusão com três classes (Apple, Banana e Orange), a interpretação se torna mais complexa, pois cada classe é analisada individualmente em relação às outras. Isso é chamado de análise classe por classe.

Em uma matriz de confusão para múltiplos grupos, os valores na diagonal principal (por exemplo, Apple/Apple, Banana/Banana, Orange/Orange) representam os Verdadeiros Positivos (VP), ou seja, quando o modelo classificou corretamente os itens para cada classe específicas, no caso, a fruta que foi identificada manualmente. Já os valores fora da diagonal representam erros de classificação, que podem ser:

- Verdadeiro Positivo (VP): quando o modelo prevê corretamente um resultado positivo.
- Verdadeiro Negativo (VN): são calculados indiretamente. Para uma classe específica, os VN incluem todos os itens que não pertencem a essa classe e foram corretamente identificados como algo diferente. Por exemplo, para a classe Apple, os VNs seriam todas as previsões corretas de Banana e Orange, mas que não envolvem a Apple.
- Falso Positivo (FP): quando o modelo prevê um resultado positivo incorretamente (o resultado real é negativo).
- Falso Negativo (FN): quando o modelo prevê um resultado negativo incorretamente (o resultado real é positivo).

Com base nesses valores, diversos indicadores de desempenho podem ser calculados, como precisão, revocação, F1-score, entre outros, fornecendo uma análise mais aprofundada da eficácia do modelo de classificação (FAWCETT, 2006). Podemos verificar na Figura 17 a estrutura de uma matriz de confusão.

Figura 17 – Matriz de Confusão

Fonte: O Autor

.

Podemos analisar os indicadores de desempenho do modelo de classificação com a acurácia, recall, f-score e precisão.

- A acurácia fornece o percentual de acertos obtidos pela rede.

$$Acurácia = \frac{VP + VN}{VP + VN + FP + FN} \tag{3.4}$$

- A Revocação (recall),também conhecida como sensibilidade, avalia a capacidade da rede em detectar com sucesso os resultados classificados como positivos. Ela reflete a capacidade do modelo de capturar todos os casos positivos (AGGARWAL et al., 2015).

$$recall = \frac{VP}{VP + FN} \tag{3.5}$$

- Precisão é a proporção de verdadeiros positivos (detecções corretas) entre todas as detecções feitas. Ela reflete a capacidade do modelo de evitar falsos positivos.

$$Precisão = \frac{VP}{VP + FP} \tag{3.6}$$

- O f-score é uma harmônica calculada com base na precisão e recall. Ele é útil para equilibrar os dois aspectos, especialmente em cenários de classes desbalanceadas (POWERS, 2020).

$$f - score = \frac{2 * Precisão * recall}{Precisão * recall} \tag{3.7}$$

- Mean Average Precision (mAP), é uma métrica padrão em detecção de objetos que mede a precisão média ponderada por classe em diferentes limiares de confiança. É comumente usada para avaliar modelos como YOLO. É calculado como a média da precisão em todos os níveis de recall para cada classe (LIU et al., 2016).

- Intersection over Union (IoU), IoU é uma métrica espacial que mede a sobreposição entre a caixa delimitadora prevista e a caixa delimitadora real. É amplamente utilizada na detecção de objetos (REZATOFIGHI et al., 2019)

$$IoU = \frac{\text{Área de Interseção}}{\text{Área de União}} \tag{3.8}$$

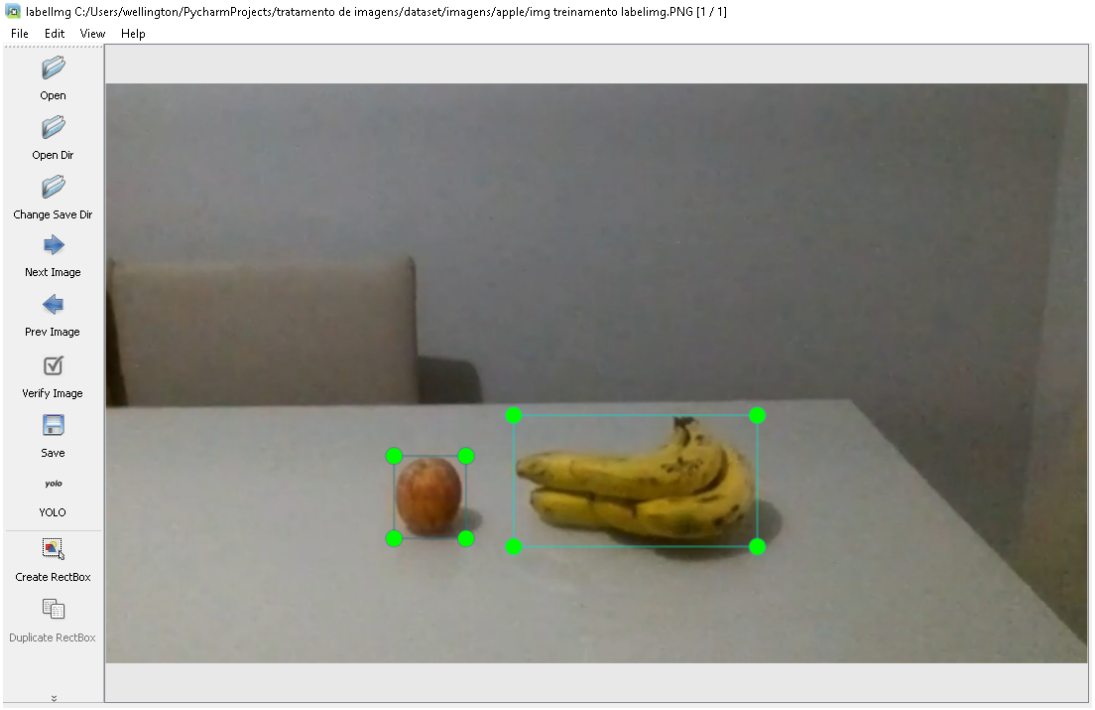
Essas métricas proporcionam uma compreensão abrangente do desempenho do modelo, abordando diferentes aspectos da qualidade das predições.

3.4 LabelImg

Para avaliar o desempenho do modelo previamente treinado, foi utilizado um conjunto de imagens anotadas, que servem como referência para validar a precisão das detecções em comparação com as anotações feitas. Essas imagens de teste foram cuidadosamente selecionadas para serem totalmente inéditas para o modelo, ou seja, não fizeram parte do conjunto de dados de treinamento, o que garante uma avaliação imparcial da capacidade de generalização do modelo. Todas as imagens foram capturadas exclusivamente para esta etapa pelo autor.

As anotações das imagens foram realizadas manualmente com o software open-source LabelImg como demonstra na Figura 18. Essa ferramenta é amplamente usada para esse tipo de tarefa. Com o LabelImg, o usuário desenha caixas delimitadoras ao redor dos objetos presentes nas imagens, gerando um arquivo no formato .txt que contém as classes e as coordenadas dos objetos. Esse arquivo de anotação serve como parâmetro de referência, permitindo comparar os resultados do modelo com as anotações reais e medir objetivamente a precisão das detecções.

Figura 18 – LabelImg - Anotação de imagens



Fonte: O Autor

.

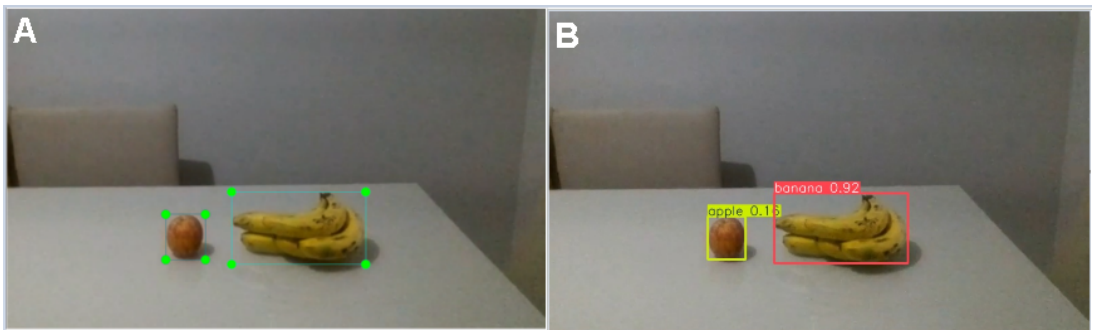
Essa metodologia assegura uma análise robusta da performance do modelo, ao mesmo tempo que reforça a importância de usar dados de teste distintos dos dados de treinamento para avaliar a capacidade do modelo de generalizar para novos dados.

4 ANÁLISE E DISCUSSÃO DOS RESULTADOS

Para avaliar o desempenho do algoritmo em diferentes condições, foram utilizados dados capturados pelo drone tello no plano de teste e fora dele. Esse procedimento resultou em um conjunto diversificado de imagens, representando situações variadas e permitindo uma análise qualitativa detalhada da eficácia do modelo. É importante destacar que as imagens utilizadas nessa etapa não fizeram parte do conjunto de treinamento do algoritmo, garantindo uma avaliação imparcial.

A análise foi conduzida com a versão YOLOv8, utilizando imagens previamente anotadas com o software LabelImg. Abaixo, na Figura 19, é apresentado uma imagem anotada (A) e o respectivo resultado de seu processamento pelo modelo YOLOv8x (B).

Figura 19 – Analise modelo YOLOv8 - Identificação



Fonte: O Autor

.

Como mostrado na Tabela 2, os valores anotados para as caixas delimitadoras apresentam grande semelhança com aqueles gerados pelo algoritmo YOLOv8, evidenciando a eficácia do modelo na tarefa de detecção. Essa proximidade entre os valores anotados e os previstos sugere que o modelo captura com precisão as características visuais e espaciais dos objetos.

Tabela 2 – Coordenadas de anotação vs. caixa delimitadora prevista

Processo	Objeto	X inicial	Y inicial	Altura	Largura
Manual	Maçã	0,33	0,71	0,07	0,14
Manual	Banana	0,54	0,68	0,25	0,23
Yolov8	Maçã	0,33	0,71	0,07	0,13
Yolov8	Banana	0,54	0,68	0,25	0,22

Fonte: O Autor

Pensando em cenários diferentes, foi proposta uma imagem com uma orientação diferente (invertida) ilustrada na Figura 20 e Figura 21. Embora essa configuração seja pouco provável de ocorrer na prática utilizando o drone tello, optou-se por avaliar possíveis variações na detecção. Na Figura 20, proveniente do Teste 2 - Ponto B, observa-se que a única classe identificada foi a da banana. Podemos notar que o nível de confiança associado a essa detecção apresentou uma redução significativa. Essa diminuição pode indicar uma limitação do algoritmo ou uma vulnerabilidade à orientação da imagem.

Figura 20 – Analise modelo YOLOv8X - Rotação

Fonte: O Autor

.

Já na Figura 21, resultante do Teste 2 - Ponto A, observa-se que a única classe identificada incorretamente foi a da maçã1. Além disso, é relevante destacar que houve uma diminuição no nível de confiança.

Figura 21 – Analise modelo yoloV8X - Rotação imagem 2

Fonte: O Autor

.

Além das condições apresentadas anteriormente foi considerado que em alguns casos podem ocorrer distorções nas imagens capturadas, como o desfoque do frame, pois o modelo de drone utilizado

não possui estabilizador de imagem. Com o intuito de avaliar esta condição, as Figura 22 e Figura 23 foram alteradas inserindo diferentes níveis de desfoque. Para a Figura 22 A o desfoque foi de 10% e para a Figura 22 B foi de 15%.

Figura 22 – Analise modelo yoloV8X - Desfoque em 10% e 15%

Fonte: O Autor

.

Para a última etapa deste quesito foi verificada a resposta do algoritmo quando a intensidade do desfoque é aumentada para 25% e 30%. Podemos observar que na Figura 23 C, a maça 1 não obteve uma classificação e as demais amostras apresentaram uma redução na confiança. Já na Figura 23 D, onde a imagem sofreu um desfoque de 30%, apenas a laranja foi detectada. Com essa análise podemos verificar que com um desfoque acima de 15% o algoritmo perde a capacidade de detecção de algumas frutas. Desse modo certos níveis de desfoque abaixo de 15% são aceitáveis e promovem um baixo impacto da detecção.

Figura 23 – Analise modelo yoloV8X - Desfoque em 25% e 30%

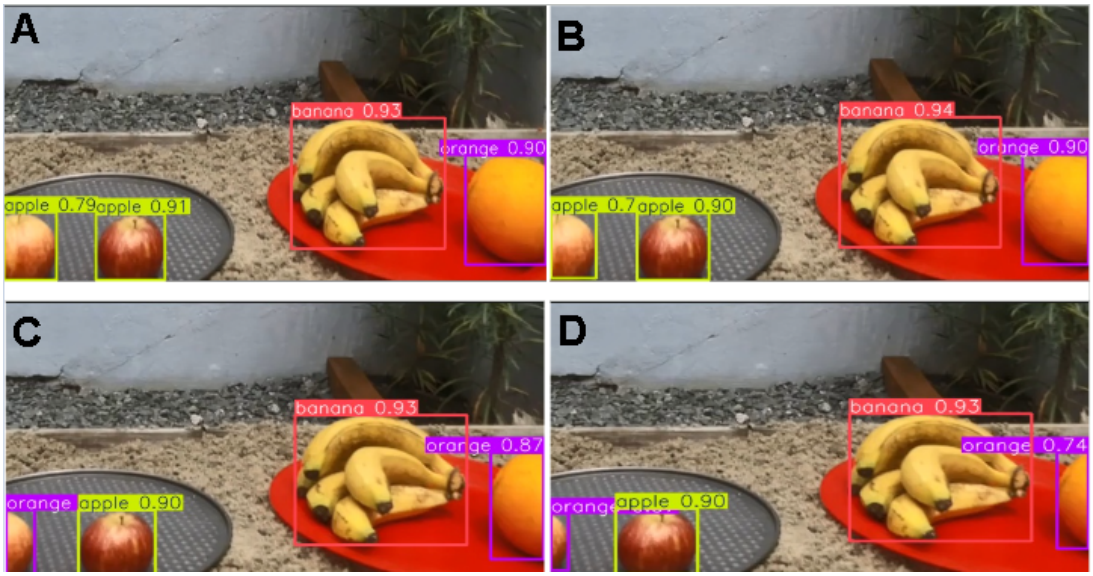
Fonte: O Autor

.

Outra análise realizada envolveu o recorte de uma imagem gerada no plano de teste, simulando a oclusão parcial das frutas. No teste, foram utilizadas a maçã1 e a laranja, posicionadas nos extremos da

imagem. Mesmo com parte da imagem recortada, o algoritmo conseguiu identificar a laranja corretamente. No entanto, ao reduzir a visibilidade da maçã em 1/3, o algoritmo passou a classificá-la incorretamente, como demonstrado na Figura 24 C e D.

Figura 24 – Analise modelo yoloV8X - oclusão

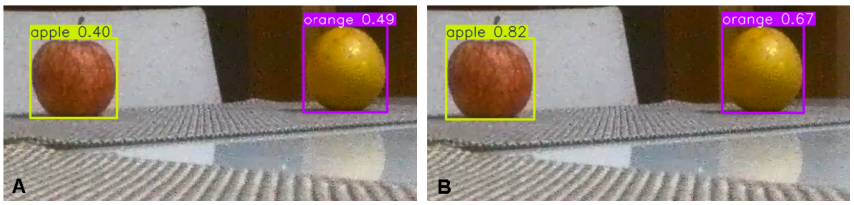


Fonte: O Autor

.

Analisando os dados gerados, foi possível identificar as diferenças entre os pacotes YOLOv8n e YOLOv8x. Observa-se que o YOLOv8x apresenta um valor de confiança significativamente superior em relação ao YOLOv8n, resultando em uma maior precisão nas detecções. No entanto, essa melhoria na confiança veio com o custo de tempo de processamento mais elevado. Na Figura 25, é possível comparar os níveis de confiança entre os dois modelos, destacando essa diferença de desempenho. A Figura 25 A foi processada com modelo YOLOv8n e a Na Figura 25 B com o modelo YOLOv8x.

Figura 25 – Analise modelo yoloV8



Fonte: O Autor

.

4.1 Analise das Métricas

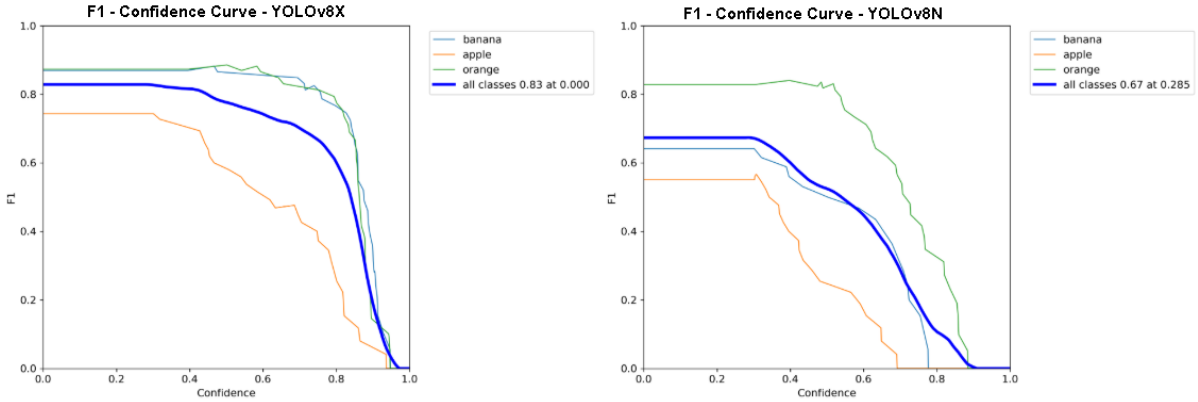
Para avaliar a eficiência do algoritmo, foi realizada a captura de cinquenta imagens utilizando o drone, executando o plano de teste em quatro cenários distintos. O objetivo dessa variação foi garantir que o modelo fosse exposto a cenários diversos. Essas imagens foram processadas utilizando os dois pacotes do YOLOv8. O pacote YOLOv8nano e o YOLOv8X. Essa análise permite uma comparação direta entre suas performances. Essa abordagem busca validar a robustez do algoritmo em identificar corretamente as classes de interesse, independentemente das alterações no ambiente de teste.

No gráfico à esquerda, referente ao YOLOv8x, observa-se que o modelo mantém um desempenho

elevado para todas as classes ao longo da maioria dos níveis de confiança. A classe banana(banana) apresenta o melhor resultado, com valores de F1 superiores a 0,85 em quase toda a faixa de confiança. As classes laranja(orange) e maçã(apple) têm desempenhos ligeiramente inferiores, mas ainda mostram boa consistência, com F1 acima de 0,7 na maior parte dos níveis de confiança. A linha azul escura, que representa o desempenho médio de todas as classes, indica um F1 de 0,83 em sua melhor performance.

Já no gráfico à direita, referente ao YOLOv8n, o desempenho médio é inferior, com a linha azul escura atingindo um F1 máximo de 0,67. A classe banana(banana) novamente apresenta o melhor resultado, mas as classes laranja(orange) e maçã(apple) mostram quedas mais acentuadas no F1 em níveis de confiança mais altos. Isso evidencia uma maior dificuldade do YOLOv8n em manter alta precisão e revocação em condições mais rigorosas. Podemos visualizar os dois gráficos na Figura 26.

Figura 26 – F1 vs confidence curve - comparativo entre modelos do YOLOv8



Fonte: O Autor

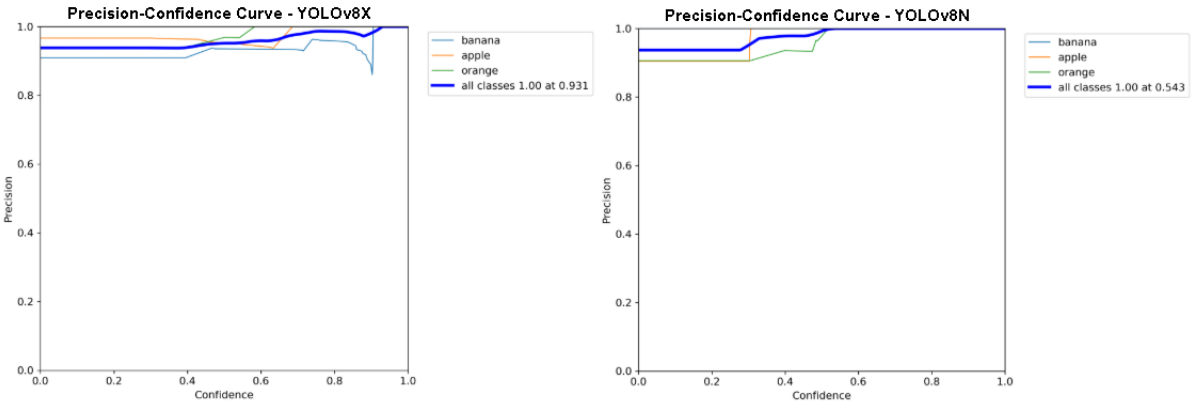
.

Na Figura 27, podemos observar o gráfico à esquerda, que se refere ao YOLOv8x, constatamos um desempenho excepcional, com a linha azul escura, que representa a precisão média de todas as classes, atingindo o valor máximo de 1,00 em uma confiança de 0,931. As curvas individuais das classes também demonstram alta precisão em toda a faixa de confiança, com pequenas variações nas classes maçã(apple) e laranja(orange), mas sempre mantendo valores muito próximos de 1,0.

Por outro lado, no gráfico à direita, referente ao YOLOv8n, a linha azul escura atinge uma precisão média máxima de 1,00, porém, com uma confiança significativamente mais baixa, de 0,543. As curvas das classes individuais apresentam maior variabilidade em níveis de confiança mais altos, especialmente na classe maçã(apple), que apresenta oscilações mais notáveis. Apesar disso, o modelo mantém uma precisão elevada na maior parte dos níveis de confiança analisados.

Esses resultados indicam que, embora ambos os modelos atinjam precisão máxima em níveis específicos de confiança, o YOLOv8x se destaca pela maior estabilidade, atingindo sua melhor performance em valores mais elevados de confiança.

Figura 27 – Precision vs Confidence curve - comparativo entre modelos do YOLOv8



Fonte: O Autor

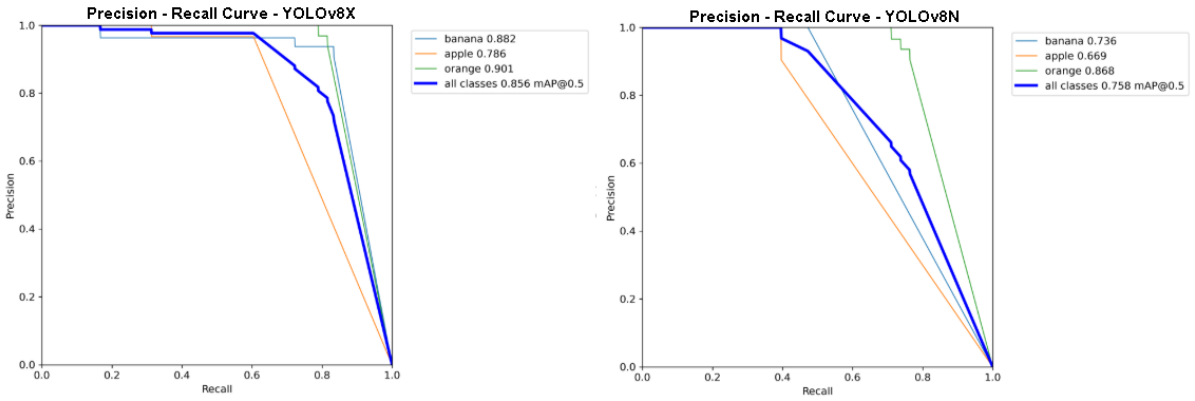
.

Os gráficos de Precisão-Recall comparam os modelos YOLOv8x e YOLOv8n em termos de equilíbrio entre precisão e recall. O YOLOv8x (à esquerda) alcança um mAP@0.5 de 0,856, destacando-se pelo desempenho elevado e consistente em todas as classes, com a classe laranja(orange) liderando com precisão de 0,901 e maçã(apple) apresentando o menor valor de 0,786.

Por outro lado, o YOLOv8n (à direita) atinge um mAP@0.5 de 0,758, inferior ao YOLOv8x, com maior variabilidade e quedas mais acentuadas no desempenho. A classe laranja(orange) novamente lidera com um valor de 0,868, enquanto maçã(apple) apresenta o menor valor de 0,669. O modelo mostra maior instabilidade, especialmente em altos valores de recall.

Os resultados apresentados na Figura 28 reforçam que o YOLOv8x é mais robusto e eficaz em tarefas que exigem alta precisão e recall, enquanto o YOLOv8n, apesar de ser uma opção mais leve, compromete o desempenho em equilíbrio de detecção.

Figura 28 – Precision vs Recall Curve - comparativo entre modelos do YOLOv8



Fonte: O Autor

.

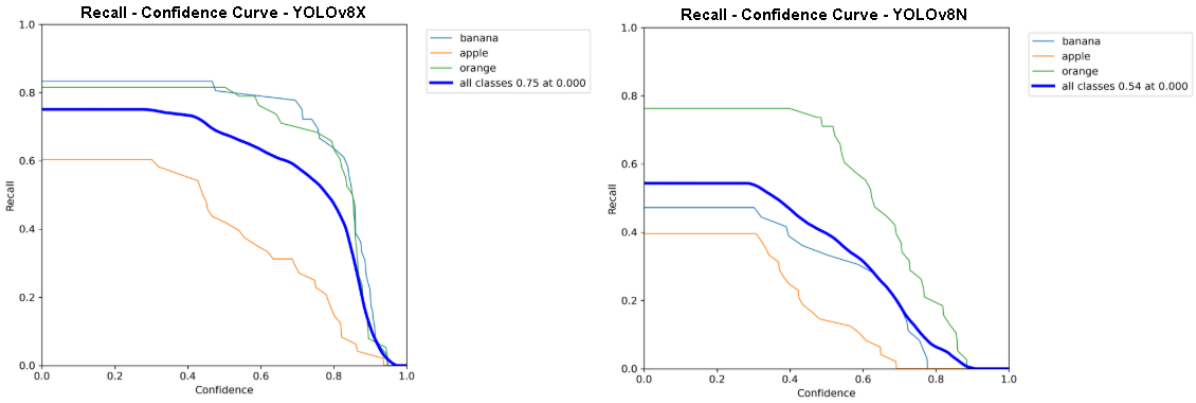
Os gráficos da Figura 29 são comparativos que ilustram o desempenho dos modelos YOLOv8x e YOLOv8n na tarefa de detecção de frutas. No gráfico do YOLOv8x (esquerda), observa-se um recall médio superior, com estabilidade notável em toda a faixa de confiança. A classe banana(banana) apresenta o melhor desempenho, seguida pela maçã(apple) e laranja(orange), que mostram maior variação em altos níveis de confiança.

Já no gráfico do YOLOv8n (direita), o recall médio é significativamente menor, refletindo

menor capacidade de detecção. Além disso, as curvas das classes individuais revelam maior instabilidade, particularmente nas classes maçã(apple) e laranja(orange). Apesar de a classe banana(banana) também liderar em desempenho, o modelo YOLOv8n apresenta maior vulnerabilidade a falhas de detecção em confiança elevada.

Esses resultados indicam que o YOLOv8x é mais robusto e confiável para aplicações que exigem alto desempenho em detecção.

Figura 29 – Recall vs Confidence - comparativo entre modelos do YOLOv8



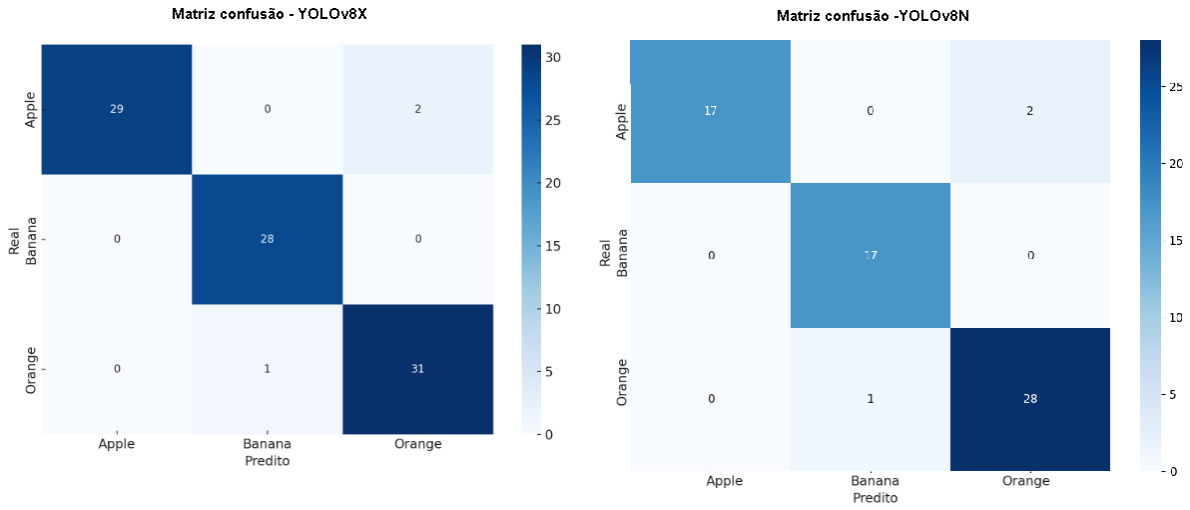
Fonte: O Autor

.

4.1.0.1 Matriz de confusão

A matriz de Confusão mostra a contagem de predições para cada classe, comparando o rótulo real (linha) com a previsão feita pelo modelo (coluna). Essa matriz mostra a contagem exata de predições corretas e incorretas para cada classe como demonstrado na Figura 30.

Figura 30 – Matriz confusão - comparativo entre modelos do YOLOv8



Fonte: O Autor

Em resumo, comparando os resultados dos dados executados com os dois modelos, podemos constatar que para todas as classes de frutas analisadas, o modelo do YOLOv8X obteve um melhor desempenho. Podemos verificar os dados na Tabela 3.

Tabela 3 – Métricas YOLOv8X vs YOLOv8N

Modelo YOLO	Classe	Acurácia	Recall	mAP50	mAP50-95
X	**Maça**	**0,967**	**0,604**	**0,786**	**0,692**
X	**Banana**	**0,909**	**0,833**	**0,882**	**0,732**
X	**Laranja**	**0,939**	**0,816**	**0,901**	**0,775**
Nano	**Maça**	**0,905**	**0,396**	**0,669**	**0,605**
Nano	**Banana**	**1**	**0,472**	**0,736**	**0,607**
Nano	**Laranja**	**0,906**	**0,763**	**0,868**	**0,751**

Fonte: O Autor

4.1.1 Processamento de dados utilizando GPU e CPU

Os mesmos dados foram processados utilizando GPU esla T4e uma CPU (Intel Xeon 2.20GHz). Podemos constatar que os dados de precisão não apresentam alterações significativas ao utilizar uma GPU ou uma CPU. A única variável que é afetada é o tempo de processamento. Podemos verificar essa diferença de processamento por frame/segundo na Figura 31.

Figura 31 – Processametno YOLOv8 - GPU vs CPU

	Pacote YOLO	
	YOLOv8N	YOLOv8X
CPU	**128.6ms**	**2s**
GPU	**45ms**	**77.8ms**

Fonte: O Autor

.

A diferença no tempo de processamento entre GPUs e CPUs deve-se à arquitetura paralela das GPUs, que possibilita a execução de milhares de operações simultaneamente. Elas são projetadas para otimizar operações em vetores e matrizes, possuem maior largura de banda de memória e diminuem a latência em tarefas repetitivas.

5 CONCLUSÃO

O presente trabalho teve como objetivo estudar a aplicação do algoritmo YOLOv8. na identificação de frutas utilizando imagens captadas pelo drone Tello. O estudo abordou a crescente relevância da tecnologia de drones na agricultura, destacando como esses dispositivos podem coletar dados visuais em larga escala e em tempo real, permitindo uma análise mais eficiente das plantações.

Os experimentos realizados em ambientes controlados possibilitaram a comparação entre diferentes versões do algoritmo, como o YOLOv8X, evidenciou a capacidade do algoritmo de detectar frutas com alta precisão. Embora o modelo YOLOv8N tenha apresentado algumas limitações,comparando com o modelo YOLOv8X, os resultados obtidos foram bastante promissores. O YOLOv8X, em particular, se destacou pela sua confiança na classificação dos objetos, indicando que ele é particularmente eficaz em ambientes mais dinâmicos e desafiadores, como os encontrados em plantações ao ar livre.

A aplicação do algoritmo YOLOv8 mostrou não só uma alta taxa de detecção, mas também uma boa capacidade de generalização para diferentes tipos de frutas, como maçã, laranja e banana, mesmo em condições de oclusão e em diferentes distâncias. Essa flexibilidade é um ponto crucial, pois sugere que a tecnologia pode ser adaptada a uma ampla gama de cenários agrícolas. No entanto, a pesquisa também identificou que a precisão do modelo pode ser ainda mais otimizada, especialmente no caso de frutas parcialmente cobertas por folhas ou outros objetos.

Esses resultados não apenas reforçam a importância da integração de tecnologias modernas na agricultura, mas também abrem portas para futuras pesquisas. Existem muitas possibilidades para aprimorar os algoritmos de detecção, como a implementação de técnicas para melhorar a precisão em condições de oclusão e em ambientes mais desafiadores. Além disso, o desenvolvimento de novos modelos, treinados com dados mais diversos, pode ampliar a capacidade de identificação e proporcionar.

Vale destacar que para otimizar o controle do drone tello durante as operações, é fundamental ter uma boa noção de voo, pois pilotar com precisão não é uma tarefa simples. A habilidade do operador é crucial para manter o drone tello estável e garantir que ele capture imagens de qualidade, mesmo a distâncias maiores.

Para trabalhos futuros, sugere-se:

- Implementação de sistemas embarcados com YOLO: Avaliar o desempenho do YOLO integrado a sistemas embarcados, coletando métricas de detecção, como tempo de processamento e precisão.
- Treinamento de pacotes personalizados: Desenvolver pacotes específicos para diferentes classes de frutas, a fim de melhorar o desempenho do modelo.
- Expansão para Outras Culturas: A aplicação da tecnologia poderia ser expandida para incluir a identificação de outras culturas agrícolas, como vegetais e grãos, validando a versatilidade da abordagem de visão computacional em diferentes contextos agrícolas.
- Desenvolvimento de Aplicativos Móveis: Outra linha de pesquisa seria o desenvolvimento de um aplicativo móvel para facilitar o uso da tecnologia de drones e visão computacional por agricultores. O aplicativo poderia fornecer análises em tempo real e recomendações baseadas nos dados coletados, otimizando a tomada de decisões no campo.

O estudo demonstrou a eficácia do algoritmo YOLO na detecção de frutas em diferentes condições e ressaltou o potencial transformador que a tecnologia pode ter na modernização das práticas agrícolas. Os resultados obtidos abrem caminho para futuras investigações e aplicações, destacando a importância de integrar inovações tecnológicas no setor agrícola para promover maior eficiência, precisão e sustentabilidade.

REFERÊNCIAS

AGGARWAL, C. C. et al. *Data mining: the textbook*. [S.l.]: Springer, 2015. v. 1. Citado na página 36.

BEZERRA, P. C. *Processamento de Imagens 2D*. 2005. ://www.dca.fee.unicamp.br/courses/IA369P/2s2009/slides/imagem2D.pdf. Acessado em 13 de julho de 2024. Citado na página 22.

COLAB, G. *Google Colab: Docs*. 2024. Https://medium.com/google-colab. Acessado em 03 de agosto de 2024. Citado na página 30.

DEMŠAR, J. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine learning research*, JMLR. org, v. 7, p. 1–30, 2006. Citado na página 35.

EMBRAPA. *Frutas: desempenho recente do agro nacional*. 2021. <https://www.embrapa.br/visao-de-futuro/trajetoria-do-agro/desempenho-recente-do-agro/frutas>. Acesso em: 09 jul. 2024. Citado na página 19.

FAWCETT, T. An introduction to roc analysis. *Pattern recognition letters*, Elsevier, v. 27, n. 8, p. 861–874, 2006. Citado na página 36.

FELGUEIRAS, C. A. *Teorias de Processamento Digital de Imagens*. 2005. <https://www.dpi.inpe.br/~carlos/Academicos/Cursos/Pdi/pdi_teorias.html>. Acessado em 9 de julho de 2024. Citado na página 21.

FILHO, O. M.; NETO, H. V. *Processamento Digital de Imagens*. Rio de Janeiro: Brasport, 1999. Citado na página 21.

FORSYTH, D. A.; PONCE, J. A modern approach. *Computer vision: a modern approach*, Prentice-Hall, v. 17, p. 21–48, 2003. Citado na página 19.

FUJIYOSHI, H.; HIRAKAWA, T.; YAMASHITA, T. Deep learning-based image recognition for autonomous driving. *IATSS research*, Elsevier, v. 43, n. 4, p. 244–252, 2019. Citado 3 vezes nas páginas 19, 22 e 23.

GONZALEZ, R. C.; WOODS, R. E. *Processamento Digital de Imagens*. 3. ed. São Paulo: Pearson Prentice Hall, 2010. Citado 2 vezes nas páginas 21 e 22.

JAIN, H.; NANDY, S. *Incremental Training for Image Classification of Unseen Objects*. 2019. Citado na página 27.

KALMBACH, D. What happens in a deep learning image classifier? 2021. Disponível em: <https://dirk-kalmbach.medium.com/what-happens-in-a-deep-learning-image-classifier-2ce0471eba58>. Citado na página 24.

LIU, W. et al. Ssd: Single shot multibox detector. In: SPRINGER. *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. [S.l.], 2016. p. 21–37. Citado na página 37.

MARENGONI, M.; STRINGHINI, D. Tutorial: Introdução à visão computacional usando opencv. *Revista de Informática Teórica e Aplicada*, v. 16, 2019. Citado 2 vezes nas páginas 21 e 22.

MARENGONI, M.; STRINGHINI, S. Tutorial: Introdução à visão computacional usando opencv. *Revista de Informática Teórica e Aplicada*, v. 16, n. 1, p. 125–160, Mar. 2010. Disponível em: <https://seer.ufrgs.br/index.php/rita/article/view/rita_v16_n1_p125>. Citado 2 vezes nas páginas 21 e 22.

NETO, A.; BONINI, C. D. S. B. Redes neurais artificiais: ApresentaÇÃo e utilizaÇÃo do algoritmo perceptron em biossistemas / artificial neural networks: Introduction and use of perceptron algorithm in biossystems. *Revista Brasileira de Engenharia de Biossistemas*, v. 4, 11 2014. Citado na página 24.

ONU. *Perspectivas da população mundial 2024*. 2024. Acesso em: 08 set. 2024. Disponível em: <https://population.un.org/wpp/>. Citado na página 19.

PACHECO, A. G. Classificação de espécies de peixe utilizando redes neurais convolucional. *arXiv preprint arXiv:1905.03642*, 2019. Citado na página 23.

POWERS, D. M. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. *arXiv preprint arXiv:2010.16061*, 2020. Citado na página 36.

REDMON, J. et al. You only look once: Unified, real-time object detection. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2016. p. 779–788. Citado 5 vezes nas páginas 19, 25, 26, 28 e 29.

REZATOFIGHI, H. et al. Generalized intersection over union: A metric and a loss for bounding box regression. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. [S.l.: s.n.], 2019. p. 658–666. Citado na página 37.

RYZE. *TELLO SPECS: drone tello*. 2024. ://www.ryzerobotics.com/tello/specs. Acessado em 22 de julho de 2024. Citado na página 30.

SALMAN, H.; KALAKECH, A. Image enhancement using convolution neural networks. *Babylonian Journal of Machine Learning*, p. 30–47, 2024. Citado na página 31.

SIMãO, J. E. de L. *Análise Modal Utilizando Câmeras Fotográficas como Sensores de Frequência sem Contato*. Dissertação (Mestrado) — Universidade Tecnológica Federal do Paraná, 2020. Citado na página 30.

TECH, D. *O que são Redes Neurais Convolucionais?* 2020. ://didatica.tech/introducao-a-redes-neurais-convolucionais. Acessado em 14 de julho de 2024. Citado na página 23.

ULTRALYTICS. *Ultralytics: YOLO v8*. 2024. Https://docs.ultralytics.com/. Acessado em 03 de agosto de 2024. Citado na página 29.

VARGAS, A. C. G.; PAES, A.; VASCONCELOS, C. N. Um estudo sobre redes neurais convolucionais e sua aplicação em detecção de pedestres. In: SN. *Proceedings of the xxix conference on graphics, patterns and images*. [S.l.], 2016. v. 1, n. 4. Citado 3 vezes nas páginas 23, 24 e 25.

VERAMENDI, W. N. C. Método para contagem de plantas de milho baseado no processamento digital de imagens multiespectrais utilizando drones em ambiente de campo. Universidade Federal de São Carlos, 2022. Citado 2 vezes nas páginas 31 e 32.