UNIVERSIDADE FEDERAL DO PARANÁ

FELIPPE DE OLIVEIRA LIMA

UMA ABORDAGEM DE DETECÇÃO DE PALMEIRAS EM IMAGENS AÉREAS COM YOLO E DARKNET

CURITIBA

2022

FELIPPE DE OLIVEIRA LIMA

UMA ABORDAGEM DE DETECÇÃO DE PALMEIRAS EM IMAGENS AÉREAS COM YOLO E DARKNET

Dissertação apresentada como requisito à obtenção do grau de Mestre em Ciências Geodésicas no programa de Pós-Graduação em Ciências Geodésicas, Setor de Ciências da Terra, Universidade Federal do Paraná.

Orientador: Prof. Dr. Hideo Araki

CURITIBA

2022

DADOS INTERNACIONAIS DE CATALOGAÇÃO NA PUBLICAÇÃO (CIP)
UNIVERSIDADE FEDERAL DO PARANÁ
SISTEMA DE BIBLIOTECAS – BIBLIOTECA CIÊNCIA E TECNOLOGIA

Lima, Felippe de Oliveira.
Uma abordagem de detecção de palmeiras em imagens aéreas com Yolo e Darknet. / Felippe de Oliveira Lima. – Curitiba, 2022.
1 recurso on-line : PDF.

Dissertação (Mestrado) – Universidade Federal do Paraná, Setor de Ciências da Terra, Programa de Pós-Graduação em Ciências Geodésicas.
Orientador: Prof. Dr. Hideo Arki.

1. Geodésia. 2. Imagens aéreas. 3. Palmeira. 4. Óleo de palma. I. Arki, Hideo. II. Universidade Federal do Paraná. Programa de Pós-Graduação em Ciências Geodésicas. III. Título.

Bibliotecário: Nilson Carlos Vieira Júnior CRB-9/1797

TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação CIÊNCIAS GEODÉSICAS da Universidade Federal do Paraná foram convocados para realizar a arguição da Dissertação de Mestrado de **FELIPPE DE OLIVEIRA LIMA** intitulada: **Uma abordagem de detecção de palmeiras em imagens aéreas com Yolo e Darknet**, sob orientação do Prof. Dr. HIDEO ARAKI, que após terem inquirido o aluno e realizada a avaliação do trabalho, são de parecer pela sua APROVAÇÃO no rito de defesa.

A outorga do título de mestre está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

CURITIBA, 26 de Agosto de 2022.

Assinatura Eletrônica
30/08/2022 10:00:31.0
HIDEO ARAKI
Presidente da Banca Examinadora

Assinatura Eletrônica
02/09/2022 11:32:30.0
LEANDRO ANDREI BESER DE DEUS
Avaliador Externo (UNIVERSIDADE DO ESTADO DO RIO DE JANEIRO)

Assinatura Eletrônica
27/08/2022 16:51:13.0
JORGE ANTONIO SILVA CENTENO
Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)

AGRADECIMENTOS

Agradeço a Deus por ter me dado forças para perseverar e conseguir o objetivo final de toda essa trajetória. Sem ele não estaria aqui.

Ao meu pai, Edson (em memória), por todos os ensinamentos que me proporcionou durante sua passagem nessa vida. O que me tornei hoje é fruto de sua batalha e inúmeros sacrifícios. Sempre estará em meu coração com muita saudade lembrando sempre dos bons momentos.

A minha mãe, Ana Lucia, por ser a melhor mãe que eu poderia ter. Sempre apoiando as minhas decisões e me dando forças para seguir na minha trajetória. Amo você.

A minha irmã, Eduarda, por sempre ter sido a pessoa maravilhosa que ela é. Foi uma surpresa inesperada na minha vida, mas hoje não vivo sem você.

Ao meu irmão, Andson, por todo apoio e companheirismo mesmo estando distante.

A minha namorada e futura esposa, Taísa, por sempre estar ao meu lado em todos os momentos da vida e ser grande incentivadora dos meus sonhos. Sou grato pela sua confiança, amor e carinho. A vida não poderia ter me proporcionado alguém melhor.

Aos meus tios, que são muitos para serem listados aqui, por estarem juntos comigo e terem me apoiado nas decisões da vida. Sei que sempre poderei contar com vocês.

A minha vó, Vera Lucia, pelo seu amor, apoio e carinho. Sei que o seu coração e suas orações sempre terão o meu nome incluído.

Aos meus antigos e melhores amigos do ensino médio, Allan, Lucas, Luidy, Matheus e Pedro, pela companhia presencial ou distante. As noites jogando Playstation 4 ou os churrascos nos finais de semana alegram a minha vida.

Aos meus amigos de graduação da turma 2013-2 de Engenharia de Agrimensura e Cartográfica da Rural. Mesmo distantes, jamais esquecerei cada um de vocês.

Ao Programa de Pós-Graduação em Ciências Geodésicas e seu corpo docente, em especial ao meu orientador Hideo Araki por ter me apoiado, incentivado e ter muita paciência comigo no processo da dissertação.

A minha antiga orientadora de TCC, Alessandra Baptista, por todos os seus ensinamentos e ter me apoiado na trajetória acadêmica. Se sou mestre hoje, muito disso eu devo a você.

Ao meu professor por um dia, coorientador de TCC, orientador de IC, banca de mestrado e amigo, Leandro Andrei. A sua humildade e garra, principalmente no âmbito acadêmico, são louváveis.

A minha única amiga feita na Pós-Graduação, Lissa, por sua amizade, ajuda e apoio durante esse longo período. Parabéns pelo seu título de mestre.

À CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) pelo financiamento desta pesquisa por meio da concessão de bolsa de estudos.

À empresa Brasil BioFuels por gentilmente ter colaborado com os dados para que esta pesquisa fosse feita.

“A ciência nunca resolve um problema sem criar ao menos outros dez.”

(George Bernard Shaw)

RESUMO

O cultivo de árvores de palmeira vem crescendo nos últimos anos devido ao uso do óleo da palma em diversas finalidades econômicas e ambientas. O número exato de palmeiras é uma importante informação para contagem de palmeiras. O trabalho propõe-se a verificar se técnica *You Only Look Once* (YOLO) pode ser utilizada para detecção e avaliação do estado de palmeiras através de imagens aéreas. Para isto, se abordará um estudo da YOLOv4 em conjunto com o framework Darknet para a detecção de palmeiras em imagens de alta resolução espacial. Foi utilizado um *dataset* de imagens de alta resolução de palmeiras para a realização do treinamento. Foi realizada uma evolução na detecção de objetos com diferentes quantidades de imagens (10, 20 e 40) e amostras (261, 605 e 1.301). No *dataset* final (40 imagens) foi obtido um *Mean Avarage Precision* (MAP) de 98,9%. Utilizando o mesmo *dataset* final, foi realizada detecções em diferentes escalas. A detecção mostrou bons resultados com mais de 90% de acerto com as palmeiras reais da imagem. Além disso, 25 imagens no treinamento e 331 amostras foram utilizadas para a detecção de palmeiras não saudáveis a partir do amarelecimento das folhas com o valor de MAP de 78%. Ambos os treinamentos apresentaram valores satisfatórios nas detecções propostas na pesquisa. Os testes também se mostraram satisfatórios quanto a detecção em diferentes escalas em relação ao *dataset* de treinamento.

Palavras-chave: detecção de objetos; palmeiras; *deep learning*.

ABSTRACT

The cultivation of palm trees has been growing in recent years due to the use of palm oil for various economic and environmental purposes. The exact number of palm trees is important information for counting palm trees. The work aims to verify if the You Only Look Once (YOLO) technique can be used to detect and evaluate the status of palm trees through aerial images. For this, a study of YOLOv4 will be approached in conjunction with the Darknet framework for the detection of palm trees in high spatial resolution images. A dataset of high-resolution images of palm trees was used to carry out the training. An evolution was made in the detection of objects with different amounts of images (10, 20 and 40) and samples (261, 605 and 1,301). In the final dataset (40 images) a Mean Avarage Precision (MAP) of 98.9% was obtained. Using the same final dataset, detections were performed at different scales. The detection showed good results with more than 90% accuracy with the real palm trees in the image. In addition, 25 training images and 331 samples were used to detect unhealthy palm trees from leaf yellowing with a MAP value of 78%.

Keywords: object detection; palm trees; deep learning.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS OU SIGLAS

AP	- *Average Precision*
BN	- *Batch Normalization*
CNN	- Convolutional Neural Network
CUDA	- *Compute Unified Device Architecture*
CUDNN	- CUDA *Deep Neural Network*
DOTA	- *Dataset for Object Detection in Aerial Images*
DL	- *Deep Learning*
GPU	- *Graphics Processing Units*
IDE	- *Integrated Development Environment*
IoU	- *Intersection Over Union*
MAP	- *Mean Average Normalization*
OpenCV	- *Open Source Computer Vision Library*
R-CNN	- *Region-based Convolutional Neural Networks*
RNA	- Redes Neurais Artificiais
SSD	- *Single Shot Detector*
SSDD	- SAR *Ship Detection Dataset*
VAID	- *Vehicle Aerial Imaging from Drone*
VOC	- *Pascal Visual Object Classes*
VOC 2007	- *Visual Object Classes Challenge 2007*
YOLO	- *You Only Look Once*

SUMÁRIO

1 INTRODUÇÃO

As palmeiras se tornaram uma das culturas que obteve rápida expansão devido a extração do óleo da palma que possui diversas finalidades, sendo utilizada em comidas, cosméticos e fonte de energia. Além disso, as palmeiras são as maiores produtoras de óleo quando comparada com outras plantas como soja e girassol (YARAK *et al.*, 2021). Este fato atraiu uma atenção considerável para técnicas de detecção e contadores de objetos a partir de imagens aéreas de alta resolução espacial (SHAFRI *et al.*, 2011).

Os frutos da palmeira-de-dendê contêm valor nutricional com benefícios para a saúde. Além disso, os frutos podem ser processados em uma variedade de produtos com finalidades comerciais, como materiais de construção, cosméticos e papel. Portanto, o levantamento de palmeiras, incluindo a contagem, padrão, distribuição e localização, é de importância crucial para prever volumes de produção e para fins de gerenciamento de estoques (JINTASUTTISAK; EDIRISINGHE; ELBATTAY, 2022).

O campo da visão computacional tem como objetivo extrair informações a partir de imagens. Neste âmbito da visão computacional está incluído o pré-processamento e o tratamento de imagens, englobando a detecção de objetos, segmentação de instâncias e classificação de imagens.

A tarefa de detecção de objetos consiste em determinar o local na imagem onde um determinado alvo está presente, bem como classificar esses objetos, ou seja, a localização de um objeto junto com a classe, é chamado de detecção de objetos.

O procedimento padrão de detecção de objetos dentro do campo da visão computacional era feito tradicionalmente em três diferentes passos: (i) detecção da região de interesse, (ii) extração de características das regiões/alvos e (iii) implementação de um classificador supervisionado. Esse procedimento metodológico, apesar de mostrar bons resultados, a maioria das vezes é incapaz de apresentar melhorias (Pham *et al.*, 2020).

Atualmente, a aprendizagem profunda (*Deep Learning*, em inglês) tem sido amplamente aplicada para detecção de objetos devido ao seu sucesso em extração de padrões em imagens. Desse modo, cada vez mais técnicas e arquiteturas são

modeladas para a identificação com o objetivo de melhoria de acurácia e redução de tempo de processamento.

Embora o aprendizado profundo tenha sido aplicado em técnicas de sensoriamento remoto, incluindo detecção de copas de árvores, o tamanho da copa da árvore, por exemplo, pode sobrepor uma à outra, dificultando a diferenciação das árvores. Além disso, a variedade de dados torna difícil a detecção em áreas de diferentes características como densidade e cor da copa.

A técnica *You Only Look Once* (YOLO) é um tipo de método para detecção de objetos a partir de imagens e vídeos. Essa técnica vem sendo utilizada em diversas tarefas de detecção de objetos devido a sua capacidade de detectar objetos em tempo real, como pedestres em vídeos de segurança (MOLCHANOV, 2017). Além disso, a YOLO é capaz de contribuir com o campo de detecção de objetos associado a imagens aéreas de alta resolução, apesar de detecções erradas ou com baixa acurácia ainda acontecerem devido ao tamanho dos objetos em relação a resolução espacial (objetos pequenos), e a base de dados para treinamento ser pequena e limitada. Em geral, a detecção de um determinado objeto depende da correspondente base de dados de imagens de treino para uma melhor performance da detecção.

A YOLO nas 3 primeiras versões teve dificuldade de encontrar pequenos objetos em imagens de entrada de alta resolução, pois a obtenção das amostras é feita em resoluções mais baixas, o que dificulta a detecção de objetos pequenos ou distantes (RŮŽIČKA; FRANCHETTI, 2018). Radovic *et al.* (2017) utilizou a YOLOv2 e mostrou que após 45.000 iterações durante treinamento, foi possível detectar aviões utilizando a técnica YOLO em imagens aéreas provenientes do Google Maps em até 900 dpi.

Entretanto, o trabalho para detecção em palmeiras e outros tipos de árvores ainda necessita de trabalho manual antes da detecção propriamente dita (LI *et al.*, 2018). Nesse sentido, diferentes arquiteturas de *Deep Learning* (DL) podem ser utilizadas em aplicações agrícolas com acurácia e velocidade para o aumento da produtividade (KAMILARIS; PRENAFETA-BOLDÚ, 2018).

Devido a aplicações na esfera da detecção, a presente pesquisa pretende contribuir para a construção de uma abordagem metodológica para a detecção de palmeira de dendê a partir da técnica YOLOv4 e, além disso, detectar mudanças espectrais nas imagens obtidas da região de estudo, tendo assim como um

resultado uma detecção automática baseada em DL servindo de suporte e tendo como consequência o melhoramento no gerenciamento.

2 OBJETIVO GERAL

Desenvolver uma metodologia de detecção de palmeiras, a partir de imagens aéreas utilizando o sistema de detecção YOLOv4 com a arquitetura da Darknet.

2.1 OBJETIVOS ESPECÍFICOS

- Criação de *datasets* de imagens (a partir de caixas delimitadoras de palmeiras para o treinamento da rede;
- Avaliar o desempenho da técnica YOLOv4, variando o número de épocas e imagens;
- Verificar a possibilidade de diferenciação de palmeiras saudáveis e não saudáveis na faixa do visível (RGB);
- Realizar a contagem do número de palmeiras.

3 JUSTIFICATIVA

As aplicações em DL na área de detecção de objetos, são diversas e vão além da detecção de objetos usuais. Com o constante desenvolvimento do sensoriamento remoto, imagens de alta resolução estão cada vez mais disponíveis e aptas a aplicações em detecção, possibilitando extrair padrões e detalhes das imagens.

A produção de palmeiras no Brasil, apesar de possuir uma porcentagem pequena comparada a produção mundial de óleo de palmeiras, dobrou de tamanho entre 2004-2010 e tem a previsão de crescer ainda mais (PEDLOWSKI, 2021). O óleo da palma, no Brasil, é extraído principalmente para gastronomia, lubrificante, cosmético e graxas.

Atrelada a crescente produção, o Brasil possui uma política de incentivo (RenovaBio) que visa a expansão da produção de biocombustíveis, aumentando a demanda por matérias-primas a base de óleos comestíveis, sendo um deles o óleo proveniente das palmeiras.

Sendo assim, técnicas em detecção de objetos utilizando imagens aéreas podem ajudar no sentido de contagem para estimação de futuras colheitas. Nesse sentindo, a YOLO é capaz de contribuir na detecção de objetos por ser uma técnica rápida na questão de processamento e precisão. Ademais, a literatura ainda é escassa em relação a detecção de doença, perturbação ou algum tipo de anomalia que se encontre nas folhas, deixando-as com tons amarelados ou diferentes de uma folha saudável.

Com o rápido desenvolvimento de técnicas de sensoriamento remoto, aumentou-se consideravelmente a quantidade e qualidade de imagens de alta resolução disponíveis para caracterizar objetos na superfície terrestre como aviões, carros e edifícios. Esse fato traz uma possibilidade de observação nos dias atuais, através de uma análise automática e com maior compreensão das imagens aéreas. Desse modo, a detecção de objetos desempenha um papel importante na interpretação de imagens em uma ampla gama de aplicações, como monitoramento, planejamento e atualizações geográficas (LI *et al.*, 2020).

Sendo assim, atualmente a detecção de objetos possui diversos desafios no sensoriamento remoto, como, por exemplo, a variação da aparência dos objetos

causada por mudança no ponto de vista, oclusão iluminação, sombra e entre outros. (CHENG; HAN, 2016).

Além disso, diversos trabalhos (Weinstein *et al.*, 2019; Culman, Delalieux, Van Tricht, 2020) aplicam imagens aéreas na detecção de palmeiras e outros tipos de árvores em situações que os objetos a serem identificados estão espaçados um dos outros, facilitando a detecção com poucos contrastes em alvos semelhantes, algo que em uma plantação de palmeiras usualmente não acontece, pois uma fica sobreposta sobre a outra. Dessa forma, observa-se o surgimento de dois problemas desafiadores que a YOLOv4 no campo de detecção pode superar: (1) o tamanho de cada palmeira (apenas 155 x 155 *pixels* em média) em uma imagem de 7 centímetros de resolução; (2) palmeiras e outras árvores próximas causando sobreposição, o que dificulta a detecção de forma precisa.

4 REVISÃO DE LITERATURA

Serão abordados neste capítulo os conceitos importantes para a realização desta pesquisa. Desse modo, serão apresentados conceitos de detecção de objetos, aprendizado profundo e visão computacional.

4.1 RECONHECIMENTO DE PADRÕES

Reconhecimento de padrões é uma habilidade inata que o ser humano possui e tem relação com a capacidade de separar e/ou classificar objetos em diferentes categorias para que seja possível tomar uma ação desejada. Os objetos de estudo podem variar de acordo com a aplicação, podendo ser imagens, dados financeiros ou alguma variável que possa ser classificada. O campo de estudo em reconhecimento de padrões se desenvolveu significativamente na década de 1960 e tinha um conceito interdisciplinar, tais como engenharia, estatística, psicologia, filosofia, entre outros (WEBB, 2003).

No contexto da computação, o reconhecimento de padrões tem como objetivo receber os dados de entrada e separá-los, classificando-os em categorias específicas (ROPKE; BINELO, 2020). Uma aplicação dessa técnica, por exemplo, são os filtros de *e-mail* de *spam* que aplicam um reconhecedor de padrões em cada *e-mail* recebido. Assim, caso encontre algum padrão classificado como *spam*, automaticamente ele executa uma ação específica (ou seja, mover a mensagem para a caixa de *spam*).

4.2 VISÃO COMPUTACIONAL

O propósito de visão computacional é dar à máquina a capacidade de “enxergar”. Isto é, analisar, entender e interpretar visualmente as informações. A visão computacional se desenvolve para a construção de sistemas artificiais que obtém informações de imagens, ou seja, faz com que o computador reconheça padrões de uma maneira próxima ou até melhor que o ser humano, a partir de imagens. Um exemplo de problema solucionado por essa técnica, é a capacidade do computador de detectar algum tipo de alvo (MILANO; HONORATO, 2010).

A visão computacional trabalha com uma imagem de entrada e uma saída da interpretação dessa imagem, com informações ou dados multidimensionais (MARENGONI STRINGHINI, 2009). Com o desenvolvimento e crescimento da inteligência artificial e a aplicação de redes neurais, o campo continuou em desenvolvimento e ganhando novas aplicabilidades, de forma que o computador recebesse informações a partir de imagens e vídeos, executando aplicações de forma inteligente (DE MILANO; HONORATO, 2014).

Uma das principais bibliotecas para aplicações em visão computacional é o *Open Source Computer Vision Library* (OpenCV). O OpenCV é uma biblioteca de visão computacional de código livre que foi escrita em C e C++ e é aplicável no Linux, Windows e Mac que foi criada para eficiência computacional, com foco em aplicações em tempo real (BRADSKI; KAEHLER, 2008).

Um dos objetivos do OpenCV é fornecer uma estrutura de visão computacional fácil de se utilizar que ajude os usuários a construir aplicações de forma rápida. Além disso, possui uma vasta possibilidade de bibliotecas que pode ser aplicada em diversas áreas, incluindo como interface de usuários, na medicina, na inspeção de produtos, na calibração de câmeras e robótica. No total, são mais de 500 funções disponíveis (ZELINSKY, 2009).

O OpenCV engloba as áreas de segmentação, detecção, reconhecimento e rastreamento de objetos/humanos/faces, calibração e reconstrução de formas 2D e 3D. Além disso, o OpenCV possui pacotes de redes neurais profundas para treinamento de detecção de objetos que permite inserir parâmetros para processamentos de imagens de entrada (FARHODOV *et al.*, 2019).

Com o desenvolvimento computacional, o DL está cada vez mais interligado ao campo da visão computacional. Porém, seguramente o DL não irá resolver todos os problemas da visão computacional. Para alguns problemas as técnicas tradicionais são a melhor solução, mas, para problemas mais atuais como a classificação de imagens, segmentação e detecção de objetos, ou seja, coisas que exigem maior esforço computacional, o DL irá permitir que usuários de visão computacional atinjam melhor acurácia (O'MAHONY *et al.*, 2019). O item 4.3 introduzirá conceitos de aprendizado com redes neurais e de DL.

4.3 REDES NEURAIS ARTIFICIAIS

As Redes Neurais Artificiais (RNA) têm sido amplamente utilizadas e popularizadas como uma técnica útil para classificação, segmentação, reconhecimento de padrões e previsão. As RNAs são técnicas computacionais que apresentam um modelo matemático similar ao comportamento neural de organismos inteligente que adquirem aprendizado através da experiência. Além disso, essa técnica pode ser enquadrada como um tipo de modelo de aprendizado em que não há suposições sobre distribuição dos dados, possuindo, assim, uma maior aplicabilidade prática, além de abrangerem modelos de regressão não linear e modelos discriminantes (ABIODUN *et al.*, 2018).

A RNA tem origem no século 19 a partir de observações dos neurônios biológicos. A inteligência humana, derivada do cérebro, tem como entidade básica os neurônios que são interconectados em redes, o que permite a troca de informação entre si e que tem um papel fundamental na capacidade de raciocínio e comportamento do corpo (KOVÁCS, 2002).

O modelo de uma RNA, que foi introduzido por McCulloch e Pitts em 1943, é composto por unidades, as quais geralmente são conectadas por canais de entrada que estão associadas a um peso correspondente. A função de ativação irá realizar a combinação linear entre as entradas e os pesos da rede neural para dar saída ao neurônio. Se o resultado dessa combinação linear ultrapassar um limiar, no caso de funções de ativação com decisão abrupta, o neurônio ativa o valor 1 da função binária, caso não ultrapasse, a saída fica em 0. Já a função de ativação, calcula as respostas geradas pelas unidades. As mais usuais são a Degrau, *Sigmoid* e *Hard Limiter* (RAUBER, 2005).

O que diferencia e potencializa o uso das RNAs em relação a outras técnicas é a capacidade de processamento em alta velocidade proporcionada por uma implementação paralela massiva e isso tem aumentado a necessidade de pesquisas nesse campo (IZEBOUDJEN *et al.*, 2014). As RNAs são impulsionadas pelas análises de dados e se mostram eficientes e bem-sucedidas, com um alto nível de capacidade no que diz respeito ao tratamento de problemas complexos. Esta técnica pode também ser aplicadas no sensoriamento remoto como mapeamento de agricultura, planejamento urbano e ambiental, além de ser eficaz na ciência médica, finanças, gestão, segurança, engenharia e arte (ABIODUN *et al.*, 2018).

A taxa de velocidade em que uma rede neural vai aprender, depende de uma série de fatores como, complexidade da rede, número de camadas, arquitetura escolhida, algoritmo de aprendizado e regras de decisão empregadas. Além disso, vários fatores precisam ser abordados com relação a tarefa de aprendizado e treinamento de uma rede neural artificial (TORDEUX *et al.*, 2017).

O desenvolvimento das *Convolutional Neural Network* (CNN) teve uma significativa influência no campo da visão computacional e é responsável por um grande salto na capacidade de reconhecer padrões (Voulodimos *et al.*, 2018). No contexto do sensoriamento remoto, recursos da imagem podem ser aprendidos usando CNN de acordo com sua arquitetura específica (LONG *et al.*, 2017).

Tendo em vista as aplicações mencionadas, incluindo desafios que não podem ser resolvidos por procedimentos computacionais e matemática convencional, há uma extensa necessidade de abordar o problema de forma mais sistemática na fase de desenvolvimento com o intuito de melhorar seu desempenho. Os avanços dos procedimentos eram lentos e tiveram avanço a partir dos anos 80 devido ao avanço da tecnologia, principalmente, pelas propriedades associativas das RNAs (OSÓRIO e BITTENCOURT, 2000).

4.4 *DEEP LEARNING*

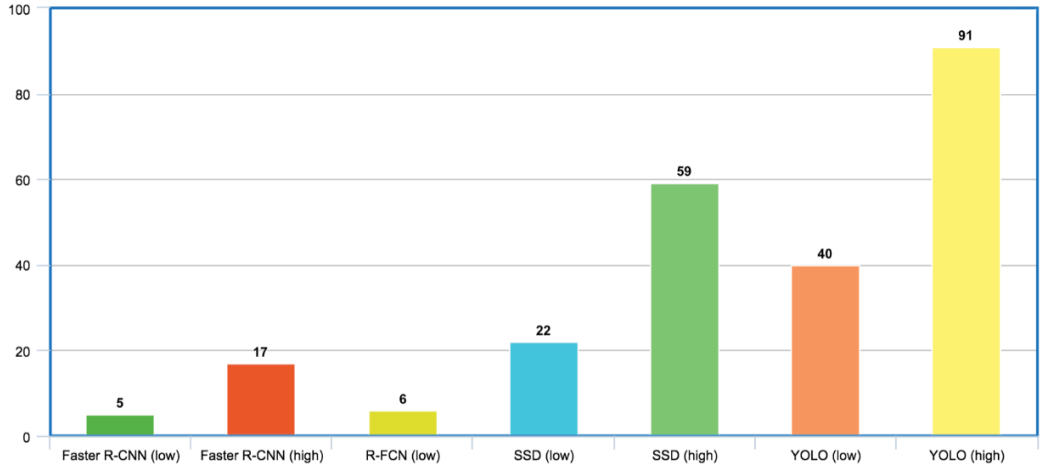
O DL já é utilizado em todo o mundo e sua aplicabilidade no âmbito da detecção de objetos vem aumentando. Outros trabalhos se assemelharam a essa pesquisa utilizando imagens aéreas para detectar objetos terrestres em tempo real a partir de imagens de drones, como foi o caso do estudo de Meng *et al.* (2020), com objetivo de identificar em tempo real escavadeiras para segurança de dutos a partir de imagens de drone. O DL opera em lugares onde muitas pessoas não percebem e, nesse sentido, é um campo do *Machine Learning* que basicamente treina computadores para executar tarefas como seres humanos, baseado em redes neurais profundas (SAITOH, 2021).

O DL possui técnicas e redes principais, como a SSD (*Single Shot Detector*) e a YOLO, que só precisam de uma passagem pela imagem para detectar vários objetos na imagem. Enquanto isso, outras redes, como a R-CNN (*Region-based Convolutional Neural Networks)* e a *Faster* R-CNN precisam de mais de uma passagem para detectar objetos de cada proposta de região (FU *et al.*, 2017).

Devido ao seu sucesso, o DL vem sendo mais estudado nos últimos anos colocando como foco a segmentação e classificação de imagens, bem como a detecção de objetos. Antes da era do DL, a detecção de objetos tinha o objetivo de procurar localidades que podiam conter os objetos que eram chamados de “regiões de interesse” usando a técnica de janela deslizante, que permite detectar objetos em uma imagem. Com o objetivo de capturar aspectos dos objetos da imagem, elas eram redimensionadas em diferentes escalas e janelas eram deslizadas por essas imagens (WU *et al.*, 2020).

Atualmente, a detecção de objetos baseada em DL pode ser dividida em dois grupos distintos: (i) detectores em dois estágios, como o R-CNN e as suas variantes e (ii) os detectores de um único estágio, como a YOLO e suas versões, e a SSD, sendo essa última categoria os mais rápidos e com um custo-benefício maior em relação ao tempo, além de ter uma grande aplicabilidade em detecção de objetos em tempo real. Os detectores de um único estágio fazem de forma direta a previsão categórica de objetos, são mais eficientes em relação ao tempo e possui aplicações em detecções em tempo real (WU *et al.*, 2020). A Figura 1 mostra os resultados de frames por segundo em diferentes arquiteturas utilizando o conjunto de dados *Visual Object Classes Challenge* 2007 (VOC 2007).

Figura 1 - Frames por segundo de teste em diferentes técnicas. As técnicas de passada única (YOLO e SSD) se sobressaem na velocidade em comparação com outras técnicas.



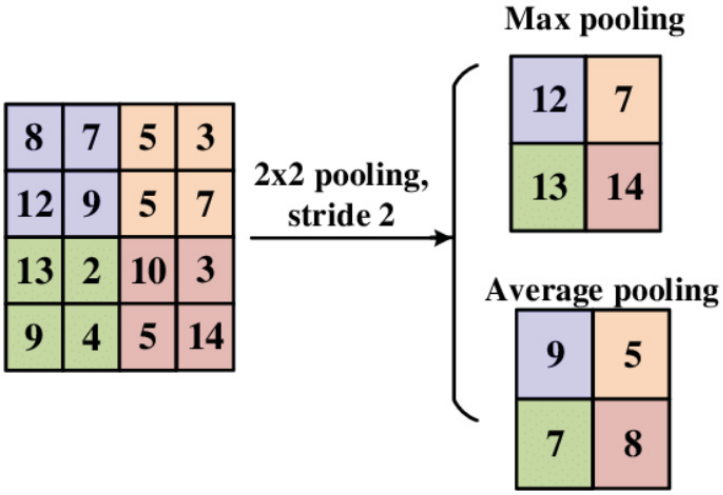
Fonte: Hui (2018)

Uma CNN consiste em uma classe de rede neural no estilo *feed-forward*, que pode ser aplicada no processamento de imagens digitais e que possui uma variação de redes de *perceptrons* de várias camadas. Um *perceptron* é representação

matemática de um neurônio, com os neurônios dispostos em camadas totalmente conectadas, em que os pesos podem ser treinados para produzir um vetor alvo (RUCK; ROGERS; KABRISKY, 1990).

Em uma CNN são aplicados filtros em imagens ou vídeos, assegurando uma relação de vizinhança entre os *pixels* ao longo do processamento (VARGAS; PAES; VASCONCELOS, 2016). Agregando na CNN, a aplicação de *pooling* visa agrupar os *pixels* das imagens e reduzir a sua dimensionalidade em um subconjunto. O objetivo é reduzir a amostra de uma representação de entrada e, consequentemente, sua dimensionalidade. Além disso, houve diminuição do custo computacional, de forma que reduziu o número de parâmetros (NAGI *et al.*, 2011). A Figura 2 mostra o processo de uma camada de *pooling*.

Figura 2 - Camada de pooling



Fonte: Yingee, Ali e Lee (2020)

A operação chamada de *Max pooling* se refere ao valor máximo do *pixel* do *batch* que é selecionado. A *Average pooling* significa que o valor médio é selecionado.

4.4.1 Treinamento em Detecções de Objetos

O conjunto de dados mais comuns como de animais, objetos e pessoas são também os mais fáceis de encontrar anotações para suas respectivas imagens. Porém, o conjunto de dados mais específicos diferem de *datasets* usais em visão

computacional, pois há poucos trabalhos disponibilizados e, consequentemente, há pouca variedade de imagens com alta resolução disponíveis (XIAO et al., 2010).

Diferentes conjuntos de dados são aplicados no campo da visão computacional para o treinamento em DL (DUARTE, 2017). Para se construir um modelo de aprendizado robusto para visão computacional, é necessário aplicar conjuntos de dados de alta qualidade na fase de treinamento, sendo que, alguns desses conjuntos são amplamente aplicados, como o CIFAR-10 que possui 60.000 imagens em 10 diferentes classes (BÖHM, 2021). Outro exemplo é o *dataset* MPII *Human Pose* que possui cerca de 25 mil imagens com mais de 40 mil pessoas com articulações corporais anotadas com imagens extraídas diretamente de vídeos do YouTube (ZHANG; ZHU; YE, 2019). O Quadro 1 mostra um resumo dos dados dos principais *datasets*.

Quadro 1: Comparativo dos principais *datasets* em visão computacional

Dataset	Exemplos de Classes	Quantidade de Classes	Número de Imagens	Dimensão das Imagens (*pixels*)
VOC	Pessoa, gato, cachorro, carro e cadeira.	20	11.530	500 x 334
ImageNet	Avião, pássaro, carro, cadeira e capacete	200	1.200.000	64 x 64
COCO	Faca, cachorro, cavalo, barco e bicicleta	80	123.287	Variado
CIFAR – 10	Gato, veado, cavalo, cachorro e caminhão.	10	60.000	32 x 32
MPII Humam Pose	Exercícios de condicionamento, esportes e atividades domésticas	410	25.000	200 x 200

Fonte: Autoria própria

Entretanto, poucos conjuntos de dados estão disponíveis no que se refere a imagens aéreas, como o *Vehicle Aerial Imaging from Drone* (VAID) que contém mais de 6.000 imagens aéreas de diferentes ângulos, iluminações e tipos de veículos. No entanto, devido a variações de sombra, iluminação e altura de voo, a detecção de objetos por imagens aéreas continua a ser um problema desafiador.

Apesar de alguns *datasets* que se utilizam de imagens aéreas estarem disponíveis, como SAR *Ship Detection Dataset* (SSDD) que tem 1160 imagens de

navios e OpenSARShip que trabalha com imagens do Sentinel-1 com mais de 34 mil navios anotados, e o *Dataset for Object Detection in Aerial Images* (DOTA), que trabalha com imagens de diferentes tamanhos (de 800 x 800 até 20.000 x 20.000 *pixels*) de diferentes sensores abrangendo variadas classes como veículos grandes, helicóptero, ponte e campos de futebol, a variedade de classes ainda é pequena e classes específicas necessitam de uma criação de conjunto de dados personalizados. Anotar imagens de forma manual é o método mais comum e acurado, mas pode ser bastante intensivo e fornecer informações novas com generalizações nem sempre contidas em outros conjuntos de dados (KONG *et al.*, 2020).

4.4.2 Aprendizado por Transferência

A disponibilidade para um conjunto de dados de treinamento para a transferência de aprendizado nem sempre está acessível e de fácil acesso para variados conjunto de dados. O aprendizado por transferência é a reutilização de um modelo treinado previamente em uma nova situação, ou seja, o modelo será usado como um ponto de partida para uma aplicação posterior (PIRES DE LIMA; MARFURT, 2020). Ding *et al.* (2020) utilizou transferência de aprendizado profundo e DL para a detecção de rostos trocados. O modelo foi capaz de detectar rostos trocados para impedir falsidade ou roubo de identidade. No trabalho foi construído o maior *dataset* até a data de rostos trocados.

4.5 DETECÇÃO DE OBJETOS

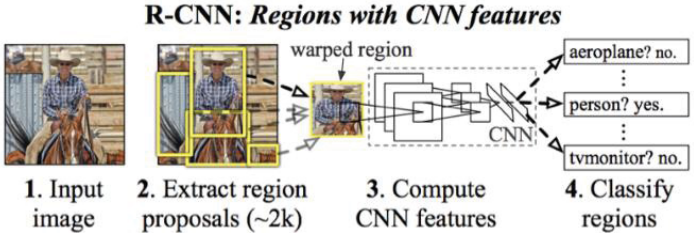
A detecção de objetos envolve a visão computacional no que diz respeito a identificação e localização de alvos em uma imagem ou vídeo, normalmente usando uma caixa delimitadora. O seu objetivo é detectar os objetos de uma ou mais classes conhecidas que podem ser carros, motos, animais ou rostos inseridos em uma imagem (AGARWAL *et al.*, 2018).

Sistemas de detecção de objetos podem ser construídos para uma ou mais classes de objetos a partir de um conjunto de exemplos de treinamento. Para detectar um objeto apenas um exemplo pode ser necessário, mas geralmente vários exemplos são necessários para o treinamento para que, assim, seja capturado e

padronizado aspectos da variabilidade da classe. Em resumo, menos dados de treinamento são necessários quando mais informações sobre a variabilidade da classe está incorporado ao conjunto de dados. Contudo, pode não ser de fácil alcance capturar as variabilidades nas imagens. Um exemplo de abordagem alternativa é usar métodos como redes neurais convolucionais que aprendem a variabilidade das classes em um grande conjunto de dados (AMIT *et al.*, 2020).

Com relação a visão computacional, a detecção de objetos possui diferentes tipos de algoritmos que foram criados para a previsão de objetos e suas caixas delimitadoras. A R-CNN se propõe com um método que usa uma busca seletiva de regiões na imagem, que é chamado de *region proposals*. Portanto, em vez de trabalhar com um grande número de regiões, o algoritmo trabalha apenas com 2.000. A Figura 3 mostra a visão geral do sistema de detecção de objetos da R-CNN dividida em 4 estágios.

Figura 3 - Resumo das camadas do sistema de detecção R-CNN



Fonte: GIRSHICK *et al.* (2014)

A rede é dividida em três módulos principais, como mostra a Figura 3:

- O primeiro gera regiões parciais das classes do modelo e então essas regiões são candidatas para o detector;
- O segundo é uma CNN que irá extrair as características dessas regiões;
- A terceira parte utiliza um classificador para a predição das classes a partir de suas características.

No entanto, ainda se leva muito tempo para treinar a rede, pois deve-se classificar 2.000 *region proposals*. Além disso, o sistema não pode ser implementado em tempo real, pois leva cerca de 47 segundos até mesmo com a *Graphics Processing Units* (GPU) para cada imagem de teste (GIRSHICK, 2015).

O mesmo autor da R-CNN resolveu algumas das desvantagens para construir um sistema de detecção de objetos mais rápido e foi nomeado de *Fast* R-CNN. A

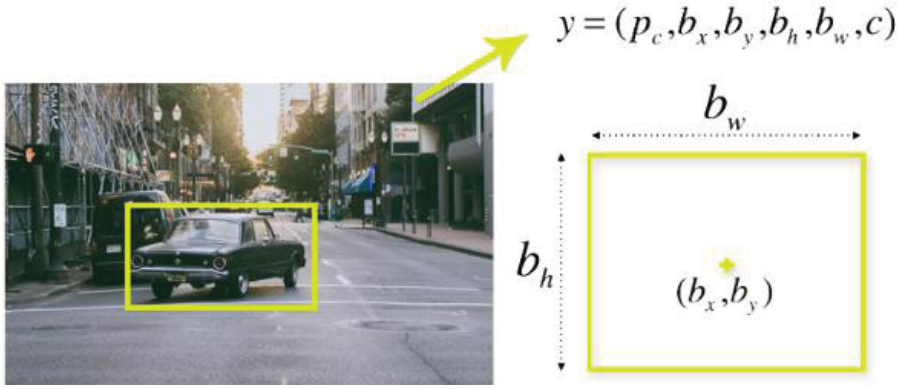
abordagem é semelhante ao algoritmo anterior, porém, a imagem de entrada é passada para a CNN para gerar um mapa convolucional de recursos a partir de uma repetida aplicação do mesmo filtro da convolução. A partir disso, verifica-se as *region proposals*, transforma-se em quadrados e, usando uma camada de *pooling*, remodela-se em um tamanho fixo para que possam ser alimentadas em uma camada totalmente conectada. Ao final, uma camada *softmax* é utilizada para prever a classe de cada região proposta. Ela se torna mais rápida porque não é necessário alimentar 2000 *region proposals* para a CNN todas as vezes (GIRSHICK, 2015).

A busca seletiva para descobrir as *regions proposals* é um processo lento e demorado que afeta o desempenho de toda rede. Nesse sentido, Ren *et al.* (2015) criou o *Faster* R-CNN que é um algoritmo que acaba com a busca seletiva. Assim, uma rede separada é utilizada para identificar as *region proposals*, que são remodeladas usando uma camada *pooling*. A *Faster* R-CNN é mais rápidas que os seus antecessores e pode ser utilizada para detecção de objetos em tempo real.

4.6 YOLO

A ideia principal da rede YOLO é otimizar o cálculo de previsões em várias posições da imagem de entrada, sem utilizar o método de janelas deslizantes (ZAFAR *et al.*, 2018). A YOLO, faz todo o processo com uma rede única, assim como indica o nome. Dessa forma, essa rede promove a divisão da imagem em uma grade S x S e cada grade faz a previsão de um determinado número de caixas delimitadoras com 4 componentes: as coordenadas (bx, by) representam o centro da caixa, em relação à localização da célula da grade, as dimensões da caixa (bw, by), a probabilidade de existir um objeto dentro da caixa delimitadora (pc) e a classe (c) correspondente ao objeto (HUANG; PEDOEEM; CHEN, 2018). A Figura 4 mostra como é formada a caixa delimitadora e seus descritores.

Figura 4 - Objeto representado pela caixa delimitadora

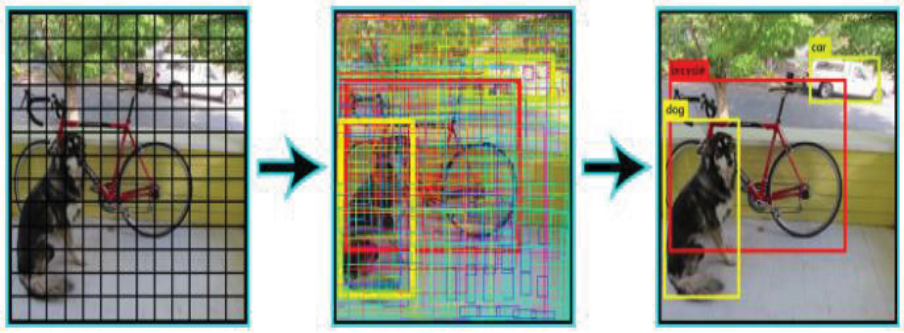


Fonte: Swiezewski, 2020

Onde p_c significa a probabilidade da classe; b_x e b_y são as coordenadas centrais da caixa delimitadora; b_h e b_w significam a altura e largura da caixa delimitadora, respectivamente; e c é o nome da classe em si.

Independentemente do número de caixas delimitadoras encontradas, ela detecta apenas um objeto e prevê uma probabilidade condicional por classe. Assim, na representação elas vão ter a espessura (largura) modificada de acordo com o seu grau de confiança de detecção de um objeto (Figura 5), ou seja, a grossura da linha de detecção serve para visualizar o grau de confiança. Portanto, pode ser observado na Figura 5 que as caixas delimitadoras que estão nos objetos, estão mais grossas que as demais, então maior é sua pontuação de confiança (REDMON *et al.*, 2016).

Figura 5 - Modelo de detecção



Fonte: Readmon (2016)

Para cada caixa, a célula também faz uma previsão de uma classe correspondente, fornecendo um valor de probabilidade para cada uma das classes possíveis. Então, se o conjunto de imagens tiver uma grade 13x13, há, portanto, 169 células e para cada uma das células são detectadas 5 caixas delimitadoras, o que resulta em 845 caixas no total. Portanto, a maioria dessas caixas delimitadores terá

um valor demasiadamente baixo, pois em diversas células não terão nenhum objeto. Por isso, geralmente é definido um limiar para que só apareçam as que tem um valor consideravelmente bom. Para avaliar o desempenho da YOLO, Everingham *et al.* (2015) utilizou um conjunto de dados chamado Pascal *Visual Object Classes* (VOC), que pode identificar 20 diferentes classes (carro, pessoa, gato, etc).

O que acontece é que diversas caixas delimitadoras terão um nível de confiança muito baixo, portanto, pode-se definir um limiar onde se exige um nível de confiança a partir de certa porcentagem considerada satisfatória (nesse exemplo foi de 85%). No final, foram identificados apenas os objetos cachorro, bicicleta e carro como pode ser visto na Figura 5.

4.6.1 Darknet

O YOLO utiliza uma rede neural profunda, cuja arquitetura é chamada de Darknet, a qual tem o mesmo nome do *framework* utilizado para a implementação (SETIYONO; AMINI; SULISTYANINGRUM, 2021). Esse tipo de rede neural tem o código aberto escrito em C e *Compute Unified Device Architecture* (CUDA). A extensão para a linguagem C CUDA foi criado pela NVIDIA para plataforma de computação e interface de programação de aplicativos que permite aumentos no desempenho de computação. A arquitetura do Darknet é composta de camadas convolucionais, filtros e funções de ativação (APTE; MANGAT; SEKHAR, 2017).

O Darknet funciona de forma rápida e permite fazer os processamentos em uma GPU. A sua execução e instalação também pode ser feita de maneira simples com poucas linhas de código e permite utilizar um modelo já pré treinado (*transfer learning*) ou construir outro totalmente do zero com os próprios dados de treinamento, desde que tenha o arquivo de treinamento na extensão *weights* já configurado para os dados em questão.

4.6.2 YOLO v1, v2, v3. O que mudou?

A YOLO consiste de uma única CNN o que a torna rápida durante todo processo. Ela trata a detecção de objetos como um problema de regressão ao invés de um problema de classificação (YANG; JIACHUN, 2018). Sendo assim, ela “olha” para a imagem de entrada de forma global ao invés de localmente e isto foi de fato a

inovação em relação a técnicas anteriores como *Faster* R-CNN e suas variações, onde se dividia a detecção em partes, gerava uma região candidata e, em seguida, classificava a região.

Os desenvolvedores Redmon e Ali Farhadi lançaram até a presente data 5 versões diferentes, sendo que a YOLO v1 estabeleceu-se como base para todas as outras. A primeira versão mostrava vantagens em relação aos outros algoritmos pela rapidez do treinamento, ao contrário de técnicas pretéritas como base em janelas deslizantes. O fato de ver a imagem completa (224 x 244 *pixels*), ocorre em detrimento ao aprendizado da YOLO em representações gerais de objetos, tornando melhor a capacidade de generalização. Portanto, quando os tipos de dados de treinamento e teste são diferentes (ideal), a YOLO v1 tem desempenho melhor que a R-CNN, por exemplo.

A YOLO v2 obteve inúmeras melhorias sobre a primeira versão, através da proposta de um método de combinação dos dados ImageNet e COCO, resultando em um modelo chamado de YOLO9000 após o treinamento com a versão dois.

Um dos primeiros avanços sobre a YOLOv1 foi o *Batch Normalization* (BN) que teve um aumento de 2% do *Mean Avarage Normalization* (MAP) (READMON; FARHADI, 2017). O BN é utilizado para deixar as redes mais rápidas e leva melhorias significativas na convergência dos dados (IOFFE; SZEGEDY, 2015). Sua rapidez está atrelada ao aumento da velocidade de treinamento pela normalização dos dados de entrada, regulando, assim, o modelo de estudo, a partir disso, as camadas de BN foram adicionadas em todas as camadas convolucionais. Outro enriquecimento que ocorreu foi quanto a classificação de alta resolução, que aumentou o MAP em 4%; convolução com caixas de âncora e o *recall* para 88%; dimensão de clusters que elevou as pontuações de *Intersection Over Union* (IoU) para caixas de âncora para 67,2%. A Figura 6 mostra o resumo das melhorias.

Figura 6 - Resumo das melhorias incrementadas

	YOLO							YOLOv2
batch norm?		✓	✓	✓	✓	✓	✓	✓
hi-res classifier?			✓	✓	✓	✓	✓	✓
convolutional?				✓	✓	✓	✓	✓
anchor boxes?				✓	✓			
new network?					✓	✓	✓	✓
dimension priors?						✓	✓	✓
location prediction?						✓	✓	✓
passthrough?							✓	✓
multi-scale?							✓	✓
hi-res detector?								✓
VOC2007 mAP	63.4	65.8	69.5	69.2	69.6	74.4	75.4	76.8	**78.6**

Fonte: Readmon e Farhadi (2017)

A YOLO v3 não pretendeu resolver nenhum problema de fato. O artigo de Redmdon e Farhadi (2018) na verdade mostra um relatório técnico com as melhorias em relação a versão anterior. A YOLO v3 usa regressão logística para prever a pontuação da caixa delimitadora que é baseada no nível de sobreposição entre o objeto e a caixa delimitadora de previsão (READMON; FARHADI, 2018). Além disso, foi utilizada uma nova rede para aprimorar a extração de camadas. A nova rede. A nova rede tem uma nova abordagem em relação a Darknet-19, utilizada na YOLO v2. A Darknet-53 utiliza sucessivas camadas convolucionais 3x3 e 1x1 com convoluções curtas, além de ser razoavelmente maior. A rede possui 53 camadas, por isso o nome Darknet-53 (READMON; FARHADI, 2018). A Figura 7 mostra a arquitetura da rede. A Darknet-53 é usada como um extrator de recursos e possui filtros 3 x 3 e 1 x 1 com conexões residuais.

Figura 7 - Arquitetura da Darknet-53

	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: Readmon e Farhadi (2018)

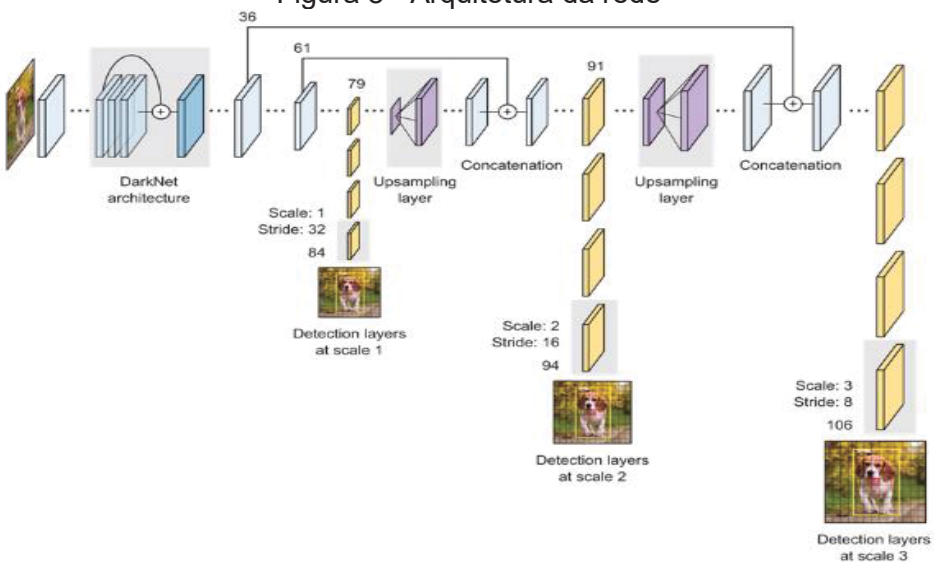
Atualmente já estão disponíveis as versões cinco e seis da YOLO. A YOLOv5 possui algumas mudanças na sua arquitetura e novas funções de ativação foram inseridas. Além disso, essa versão usa o *framework* PyTorch ao invés da Darknet (NEPAL; ESLAMIAT, 2022). Já a YOLOv6 foi dedicada a aplicações industriais e superou as outras versões em termos de precisão e velocidade (BEN, 2022). Para esta pesquisa foi utilizada a YOLOv4 devido a disponibilidade no início do projeto.

4.6.3 YOLOv4

A YOLO v4 foi criada por três desenvolvedores Alexey Bochkovskiy, Chien-Yao Wang e Hong-Yuan Mark Liao. O desenvolvedor inicial Joseph Readmon não quis mais participar devido ao uso indevido da tecnologia, sobretudo quanto as aplicações militares e proteção de dados (KHAN, 2021).

Desde que a YOLO foi lançada em 2015, ele segue uma arquitetura típica, mas houve inovações no que diz respeito a saída. A detecção geralmente requer informações visuais mais refinadas, por esse motivo foi aumentada a resolução de entrada da rede. Antes era 224 x 224, agora 448 x 448 (REDMON *et al.*, 2016). A Figura 8 mostra com mais detalhes a arquitetura da rede.

Figura 8 - Arquitetura da rede



Fonte: Kathuria (2018)

A YOLOv4 consiste de (WANG *et al.*, 2020):

- *Backbone* – CSPDarknet 53;

- *Neck* (conecta o *backbone* com o *head*) – SPP (*Spatial Pyramid Pooling*) e PAN (*Path Agression Network*);
- *Head* – O mesmo do YOLO v3.

O CSPDarknet 53 é um novo *backbone* aprimorado para aprendizado em relação as versões anteriores. A precisão média aumentou em 10% comparado ao YOLOv3 (*backbone* Darknet 53), utilizando o conjunto de dados ImageNet que possui mais de 1000 classes (BOCHKOVSKIY; WANG; LIAO, 2020).

A arquitetura possui uma rede neural convolucional em que é feito o processamento com várias camadas convolucionais. Um diferencial desta arquitetura para outras, é que o YOLO não trabalha com camadas densas das redes neurais clássicas. Por isso, há uma concatenação que ao chegar na camada 36, faz com que a rede envie a informação para as próximas camadas, porém também envia a informação para o final. O mesmo processo ocorre na camada 61 (MANTRIPRAGADA, 2020).

Durante o processamento são passados filtros onde a arquitetura muda a dimensão dos dados e são não só enviadas cópias para o final da rede, como também há o recebimento de uma cópia de todo o processamento. Esse fato faz com que o desempenho do YOLO seja diferente e melhor.

A detecção do YOLO, como mostra a Figura 8, é feita em 3 escalas a partir das camadas de detecção. Na escala 1 a imagem é a menor e vai aumentando de escala até a 3, isso acontece para que o algoritmo consiga detectar objetos menores na imagem, servindo tanto para imagens grandes, quanto para objetos pequenos. Esse fator demonstra um avanço, visto que em versões anteriores o YOLO não conseguia fazer essa detecção. Outra parte importante no processo são as camadas de *upsampling* (79 e 91) que vão aumentar a dimensionalidade da imagem antes da detecção na próxima escala maior.

A técnica YOLO obteve um melhoramento na versão YOLOv4 em termos de performance e velocidade comparado a outras técnicas que utilizam regiões para localizar objetos na imagem devido ao fato de processar a imagem inteira em uma única rede neural. Desde o início, a YOLO tem sido melhorada o que aumentou a acurácia usando uma arquitetura maior e, ao mesmo tempo, não sacrificando sua velocidade computacional (REDMON; ALI, 2018).

A função de perda na YOLO é utilizada no treinamento para melhorar as previsões das redes, dessa forma ela prevê as caixas delimitadoras por célula da

grade para, só então, calcular a perda do verdadeiro positivo, onde apenas uma das caixas delimitadoras será responsável pelo objeto. Essa técnica usa a soma dos quadrados dos resíduos entre as previsões e a verdade de campo para o cálculo da perda. Mediante a isso, a função de perda é composta pela perda de classificação, perda de localização e perda de confiança (MICHELUCCI, 2019). A Figura 9 mostra a função de perda total.

$$
\begin{aligned}
&\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(x_i - \hat{x_\iota})^2 + (y_i - \hat{y_\iota})^2] \\
&+ \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left[\left(\sqrt{w_i} - \sqrt{\widehat{w_\iota}} \right)^2 + (\sqrt{h_i} - \sqrt{\hat{h_\iota}})^2 \right] \\
&+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left(C_i - \widehat{C_\iota} \right)^2 + \lambda_{noobj \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{noobj} (C_i - \widehat{C_\iota})^2} \\
&+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{obj} \sum_{c \in classes} \left(p_i(c) - \widehat{p_\iota}(c) \right)^2
\end{aligned}
\tag{1}
$$

Fonte: Michelucci (2019)

A função de perda do treinamento realiza uma comparação de uma caixa delimitadora real do conjunto de treinamento pra saber se aquela âncora está se “encaixando” no objeto. Ou seja, se a âncora não está pequena, grande ou distante do objeto.

Onde B é o número total de âncoras; S é o número de pontos de grade (exemplo: 7 x 7); $(x_i - \hat{x_\iota})^2 + (y_i - \hat{y_\iota})^2$ representa a distância do centro do retângulo para saber se está próximo ou não do retângulo da âncora para o exemplo real; $\left(\sqrt{w_i} - \sqrt{\widehat{w_\iota}} \right)^2 - (\sqrt{h_i} - \sqrt{\hat{h_\iota}})^2$ representa a distância da altura e largura. A raiz reduz o efeito do tamanho do objeto evitando que um objeto grande tenha uma perda maior ou um objeto pequeno ter uma perda menor; λ_{coord} é o peso para o erro de localização; $\left(C_i - \widehat{C_\iota} \right)^2$diz respeito à probabilidade de ser um objeto; λ_{noobj} representa quando a âncora se refere ao fundo (quando não contém um objeto) com

um peso menor; $\left(p_i(c) - \widehat{p_i}(c)\right)^2$é a probabilidade de pertencer a uma classe. $\sum_{c \in classes}\left(p_i(c) - \widehat{p_i}(c)\right)^2$.

4.7 DETECÇÃO DE OBJETOS EM SENSORIAMENTO REMOTO

O DL vem sendo usado no sensoriamento remoto no âmbito de aplicações em segmentação, classificação e restauração de imagens. No campo de detecção de objetos, o DL ainda está limitado em alguns casos específicos quando se trata de imagens de alta resolução espacial. Além disso, ele ainda tem mostrado aplicações pertinentes a outros tipos de imagens em sensoriamento remoto como: radar, imagens hiperespectrais, imagens de alta resolução, entre outros (CRESSON, 2020). Porém, o sucesso depende altamente do método que é aplicado (MA *et al.*, 2019).

O CNN, a base do DL, é utilizada pela comunidade do sensoriamento remoto há anos, no entanto, recentemente mais precisamente desde 2014, a comunidade de sensoriamento remoto mudou suas atenções para o DL e seus algoritmos conseguiram bons resultados em análise de imagens de sensoriamento remoto (CHEN *et al.*, 2014; MA *et al.*, 2019). Vetrivel *et al.* (2019) utilizou imagens de alta resolução para detecção de dados provenientes de desastres em terremotos. Já Kussul *et al.* (2017) combinou imagens orbitais (Landsat-8) e imagens de radar (Sentinel-1) para a classificação de tipos de agricultura com uma acurácia acima de 85%.

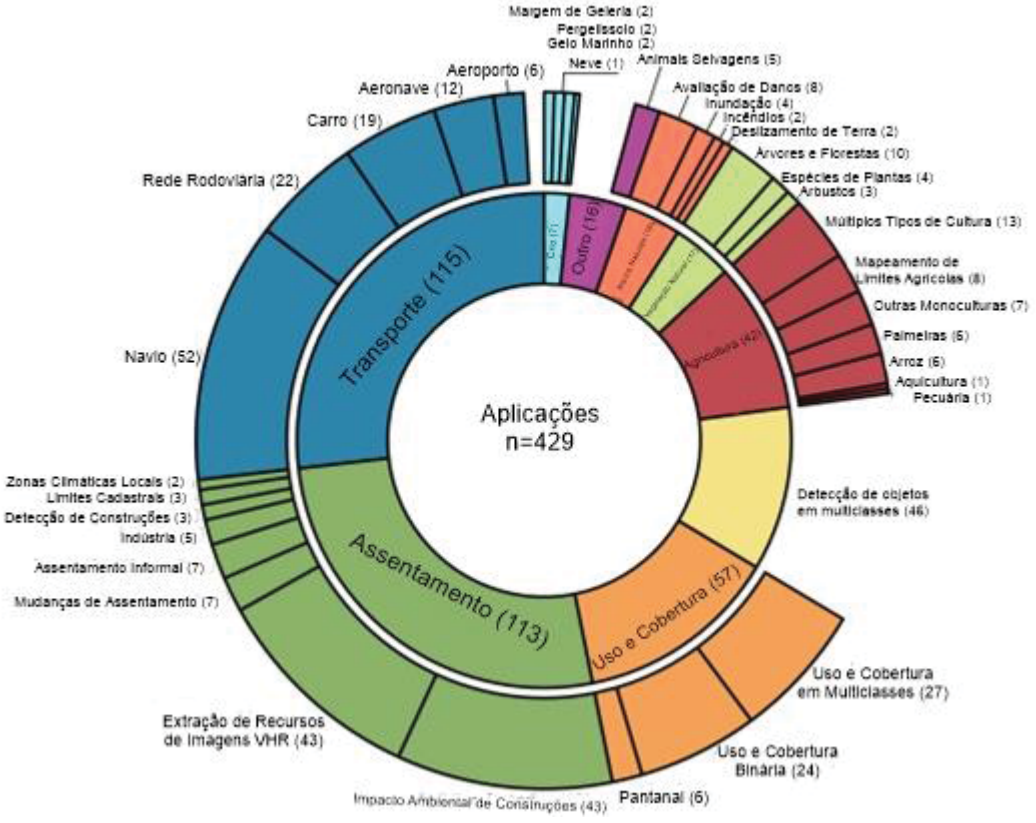
No campo da detecção de objetos, o avanço foi permitido devido a imagens de satélite de alta resolução espacial (CHENG; HAN, 2016). Jin e Davis (2007) propuseram uma abordagem de detecção de veículos utilizando imagens IKONOS de 1 m de resolução espacial e seus resultados atingiram a 86% de *score* de detecção. Outras abordagens envolvendo detecção de veículos foram utilizadas, um exemplo é Jiang *et al.* (2015) que propuseram uma CNN em que uma segmentação de *superpixel* é usada para extrair trechos da imagem e uma outra CNN para prever se o trecho contém algum tipo de veículo.

Hoeser e Bachofer (2020) fizeram um levantamento de dados de aplicações, técnicas e arquiteturas de redes neurais convolucionais sobre trabalhos aplicando DL no âmbito do sensoriamento remoto. No total foram levantados um total de 429

trabalhos divididos em aplicações distintas como agricultura, mudança de uso e cobertura da terra e transporte. Construções e transporte são as principais aplicações encontradas correspondendo a mais de metade nas pesquisas publicadas. Aplicações na agricultura também são mencionadas com aplicações em palmeiras (6), arroz (6) e outros tipos de plantações (13).

A Figura 9 mostra o levantamento geral dos trabalhos. Nota-se que dentre as principais aplicações, a área de transporte se sobressai com aproximadamente 27% dos estudos. Por outro lado, menos de 10% dos trabalhos são aplicados em agricultura. No total, apenas aproximadamente 1% foi aplicado em palmeiras no levantamento do artigo até o ano de 2020.

Figura 9 - Resumo detalhado de todas 429 aplicações encontradas e suas categorias



Fonte: Hoeser e Bachofer (2020)

4.8 MÉTRICAS DE AVALIAÇÃO EM DETECÇÃO DE OBJETOS

A detecção de objetos do YOLO com o Darknet produz diferentes métricas de avaliação para a conferência da qualidade do treinamento em uma tarefa de

detecção de objetos. Além disso, permite comparar diferentes sistemas ou comparar o comportamento do treinamento em diferentes épocas.

4.8.1 Precisão e *Recall*

A precisão mede o quão acurado estão as predições, ou seja, a porcentagem de quanto as predições estão corretas. O recall mede o quão bem encontra os positivos. A precisão irá responder a seguinte questão: qual é a proporção de identificações positivas que estavam realmente corretas? Para o recall, a seguinte pergunta deve ser relacionada: qual a proporção de positivos reais foi identificada corretamente? Sendo assim, matematicamente ambos são apresentados na Equações 1 e 2 (PADILLA; NETTO; DA SILVA, 2020).

$$\text{Precisão} = \frac{\text{TP}}{(\text{TP} + \text{FP})} \tag{2}$$

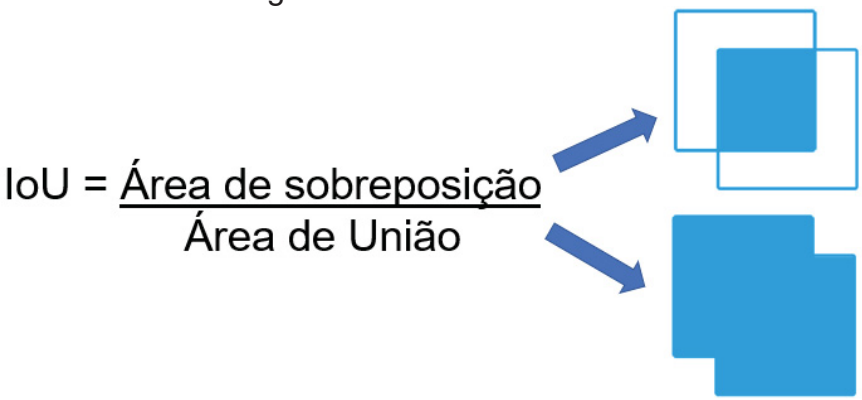
$$\text{Recall} = \frac{\text{TP}}{(\text{TP} + \text{FN})} \tag{3}$$

Com TP = Verdadeiro Positivo; FP = False Positivo; FN = Falso Negativo.

4.8.2 *Intersection Over Union*

A métrica IoU mede a sobreposição entre dois limites (Figura 10). No caso de detecção de objetos, usa-se para medir o quanto a caixa delimitadora prevista, se sobrepõe sobre a caixa delimitadora real. Em alguns casos pode ser usado um valor limiar para classificar a previsão como um verdadeiro positivo ou falso positivo (PADILLA; NETTO; DA SILVA, 2020). A Figura 10 mostra como computar o valor de IoU.

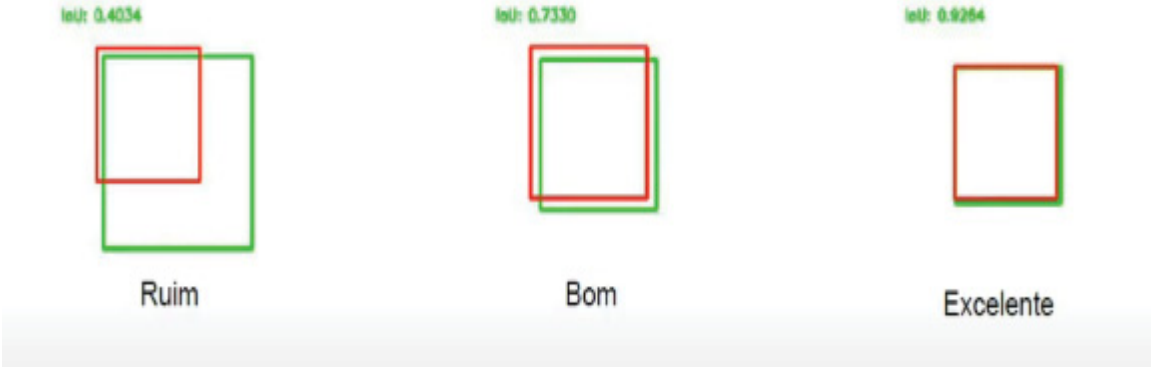
Figura 10 - Cálculo de IoU



Fonte: Autoria própria

A Figura 11 mostra a linha verde que seria a caixa delimitadora do objeto e a linha vermelha a predição calculada. Os valores representam os IOU calculados.

Figura 11 - Exemplo de IoU calculado em diferentes detecções



Fonte: Adrian Rosenbrock (2007)

O valor 1 (100%) seria o valor perfeito para a predição, porém isso é quase impossível, pois a métrica vai penalizar qualquer deslocamento na imagem.

4.8.3 F1 *Score*

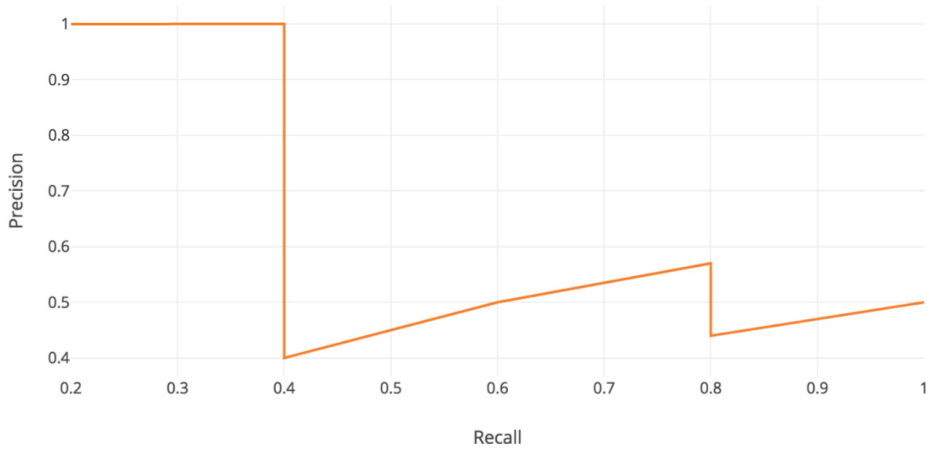
O F1 *Score* é a média harmônica entre a precisão e o recall. Os valores variam entre 0 e 1 (PADILLA; NETTO; DA SILVA, 2020). Como F1 é a média de precisão e *recall* (Equação 3), isso significa que o F1 dá peso igual a ambos.

$$\mathrm{F1} = \frac{2 * \mathrm{Precisão} * \mathrm{Recall}}{(\mathrm{Precisão} + \mathrm{Recall})} \quad (4)$$

4.8.4 *Mean Average Precision* (MAP) e *Avarage Precision* (AP)

A métrica de AP pode ser encontrada calculando a área abaixo da curva entre precisão e *recall* (cor laranja) para cada classe do conjunto de dados. O valor de AP é usado para avaliar a performance do treinamento (IVAŠIĆ-KOS; KRIŠTO; POBAR, 2019). O valor varia entre 0 e 1 e quanto mais próximo de 1, maior a precisão. A Figura 12 mostra o gráfico entre precisão e o recall de um treinamento.

Figura 12 - Gráfico Precisão – Recall



Fonte: Hui (2018)

Como observado, a Figura 12 plota os valores no gráfico de recall contra precisão à medida que vai passando o número de épocas. A curva é feita para que seja calculado o valor de AP. O cálculo da área pode ser observado melhor quando dividido em partes. Além disso, é possível perceber que na primeira parte há uma queda brusca da precisão, formando, assim, um retângulo de lado 1 e 0,4 e nesse instante, a curva apresenta um AP de 0,4 (1 x 0,4). Portanto, como foi mencionado anteriormente a definição geral para a AP é encontrando a área sob toda a curva precisão (p) – *recall* (r).

$$\mathrm{AP} = \int_0^1 \mathrm{p(r)dr} \tag{5}$$

Com p = precisão; r = recall.

O MAP é a precisão média tomando o AP médio de todas as classes. Ambos servem para calcular a performance do modelo aplicado no âmbito da segmentação

e detecção de objetos. Na YOLO o valor do MAP, assim como as outras métricas envolvidas, é obtida a cada cem interações.

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP\ i \tag{6}$$

4.9 PALMEIRA DE DENDÊ

A palmeira de dendê é um dos cultivos com maior expansão nas últimas décadas no Brasil devido ao seu uso recorrente na área da indústria. Apesar desse crescimento, o óleo da palma tinha um número modesto de produção até meados de 2010, quando o então presidente Luiz Inácio Lula da Silva incentivou a expansão do setor devido a pretensão de um polo de geração de biocombustíveis no Estado do Pará (MONGABAY, 2021).

A Indonésia e Malásia sozinhas correspondem a mais de 90% de todo óleo de palma consumidos no mundo. Porém, as monoculturas são sinônimos de desmatamento no Sudeste Asiático e está se expandindo para a Amazônia, onde os mesmos problemas estão ocorrendo (FAO, 2015). A produção de palmeiras de dendê tem transformado o uso e ocupação do solo modificando suas paisagens naturais, especialmente em países do Sudeste Asiático e América do Sul (HENDERS *et al.*, 2015).

Embora globalmente a maior parte do aumento de atividades de agricultura se dê pelo aumento da produtividade, em países tropicais a expansão das terras agrícolas e o aumento da produtividade contribuíram para melhores resultados. Porém, a maior parte dessa expansão ocorreu às custas de florestas tropicas antes intactas, de modo que a demanda global se tornou um impulsionador cada vez mais importante das mudanças do uso, da ocupação do solo e do desmatamento tropical (RUDEL *et al.*, 2009).

4.9.1 Localização e Detecção de Palmeiras com DL

Alguns trabalhos se dedicaram na localização e detecção de palmeiras, utilizando diferentes técnicas, redes e metodologias. Ammar e Koubaa (2020)

usaram a DL para contagem e geolocalização de palmeiras a partir de imagens aéreas usando CNN. Para isso, eles coletaram imagens utilizando drones em uma fazenda na Arábia Saudita e construíram um conjunto de dados com 10.000 amostras de palmeiras. Para a detecção, foi utilizada a R-CNN.

Além disso, o uso do sensoriamento remoto tem se expandido em larga escala, o que possibilitou uma redução do custo de detecção de objetos utilizando drones, satélites e imagens aéreas em geral (CULMAN *et al.*, 2020). O uso de visão computacional aplicada em sensoriamento remoto, sobretudo com a arquitetura em DL, tem provado um sucesso em detecção de árvores em florestas (WEINSTEIN *et al.*, 2019)

Em relação ao inventário de palmeiras, Culma, Delalieux e Tricht (2020) estabeleceram uma metodologia a partir de imagens RGB, utilizando CNN para contabilizar o número de palmeiras na Espanha em diferentes regiões, agrupando locais urbanos e rurais. Para isso, foi utilizada uma SSD chamada RetinaNet, que alcançou mais de 500.000 palmeiras detectadas, com probabilidade acima de 50% e com um MAP de 0.861.

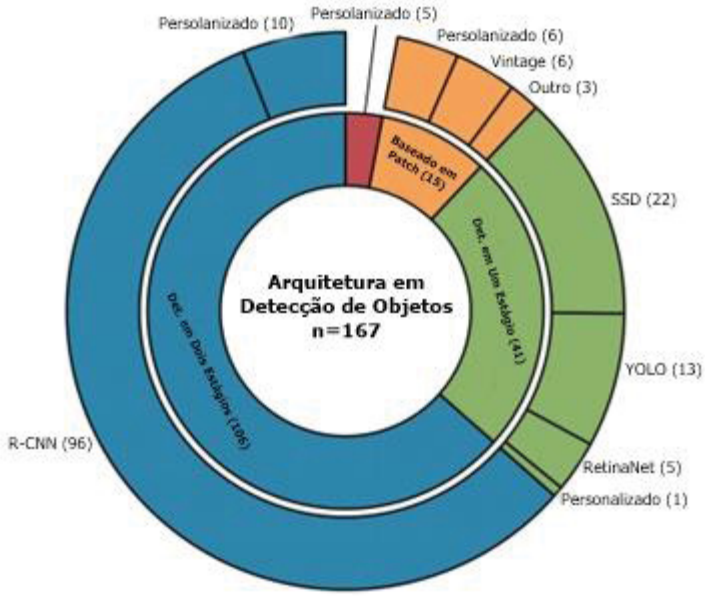
O estudo de detecção de objetos em DL pode ser ainda mais profundo e detalhista para extração de informações em uma imagem. Yarak *et al.* (2021) explorou a capacidade detectar palmeiras saudáveis e não saudáveis utilizando a *Faste*r-RCNN e a capacidade de detecção em diferentes alturas de voo. O conjunto de dados foi separado em 10% para teste e 90% para treinamento usando a Resnet-50 e VGG-16, alcançando um F1-Score de mais de 95% de detecção de palmeiras.

Algumas pesquisas têm o objetivo de comparar algumas diferentes arquiteturas e metodologias. Freudenberg *et al.* (2019) utilizou diferentes arquiteturas da rede U-Net (U-Net A e U-Net B) para comparar o desempenho com a AlexNet na questão de detecção de palmeiras na Índia. O método detectou palmeiras com acurácias entre 89% e 92% utilizando imagens de satélite de 0,46 m de resolução. O modelo encontrou dificuldades, pois as imagens continham regiões sombreadas em algumas áreas de palmeiras criando, assim, falsos positivos.

No entanto, a técnica YOLO v4 apesar de ainda ser pouco explorada em sensoriamento remoto (menos de 10% em técnicas de detecção de objetos), é a que se sobressai na questão de desempenho de tempo no campo da visão computacional por suas características mencionadas (detecção em um único estágio). Por isso, a verificação da YOLOv4 na questão de detecção de palmeiras

possibilita a exploração da técnica em conjunto com a Darknet-53 em *datasets* de grande volume de imagens e anotações. A Figura 13 mostra um levantamento geral das arquiteturas utilizadas em detecção de objetos (HOESER; BACHOFER, 2020).

Figura 13 - Resumo de arquiteturas aplicadas para detecção de objetos. 63% são utilizadas fora do âmbito das *One Stage Detector*



Fonte: Hoeser e Bachofer (2020)

A detecção de objetos aplicadas à detecção de palmeiras tem sido pesquisada de forma mais pontual em países asiáticos. Isso acontece pela alta produção do óleo retirado da palmeira, tendo em vista que países como Indonésia e Malásia produzem mais de 80% de óleo de palma no mundo (SHAHBANDEH, 2022). O Brasil, apesar de possuir uma produção pequena mundial, elevou sua produção de 0,57% em 2011 para 0,7% em 2020, o que representa aproximadamente 550.000 toneladas anuais.

Sendo assim, a YOLOv4 apresenta uma evolução na detecção de objetos no que diz respeito a acurácia e velocidade de detecção de objetos (ZAKRIA *et al.*, 2022). Portanto, a YOLOv4 e outras versões e variações estão sendo aplicadas no âmbito do sensoriamento, como por exemplo em detecções de navios (YILDIRIM; KAVZOGLU, 2021; JIANG *et al.*, 2021) e na aplicação na agricultura (SHI *et al.*, 2021). Comparada a outras técnicas, a YOLOv4 obtém melhores resultados em diferentes condições de cobertura densa (LI; WANG; HUANG, 2022).

5 MATERIAIS E MÉTODOS

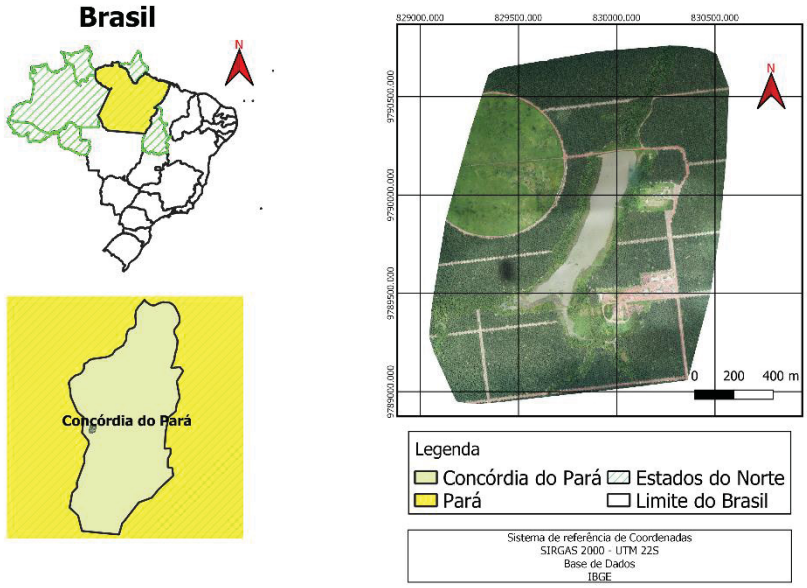
Os materiais que serão utilizados na realização desta pesquisa estão especificados nos tópicos 5.1 e 5.2.

5.1 ÁREA DE ESTUDO

Para o treinamento inicial, foi utilizado um conjunto de dados de palmeiras disponibilizados pela empresa Brasil BioFuels.

A área de estudo é dividida em duas cidades do estado do Pará. Na detecção pura de palmeiras foi utilizada uma região do município de Concórdia do Pará (Figura 14). A espécie das palmeiras nessa região é um cruzamento genético das espécies Deli x Ghana.

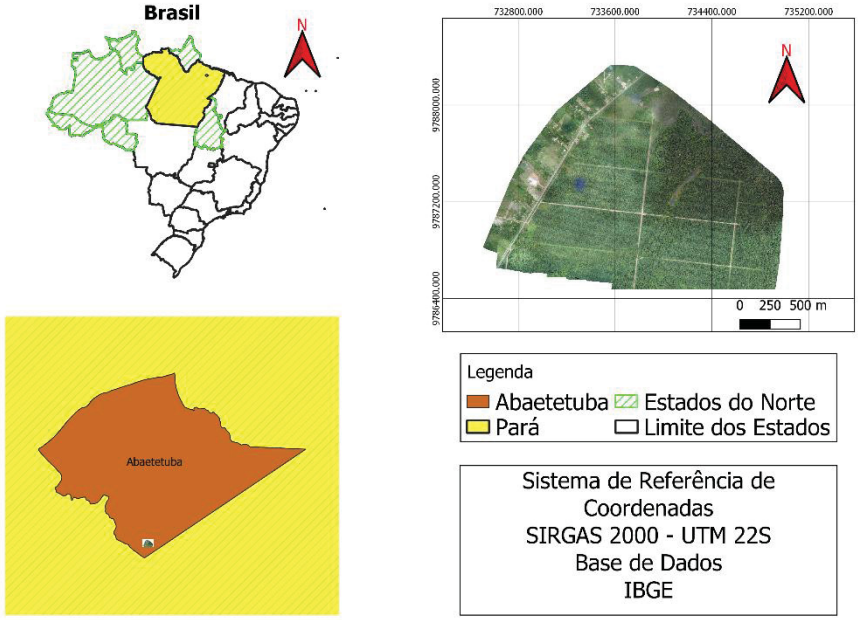
Figura 14 - Área de estudo para detecção de palmeiras



Fonte: Autoria própria

A segunda área de estudo envolve o estudo de detecção de palmeiras não saudáveis. A região corresponde ao município de Abaetetuba que também está localizado no estado do Pará e a espécie das palmeiras na localidade também é um cruzamento genético, mas das espécies Deli x Nigéria. A Figura 15 mostra o mapa de localização.

Figura 15 - Mapa de localização do município de Abaetetuba para detecção de palmeiras não saudáveis



Fonte: Autoria própria

5.2 AMBIENTE

Para a utilização do *framework* Darknet é necessário realizar o download do *framework* de mesmo nome e que pode ser obtido no endereço eletrônico https://github.com/AlexeyAB/Darknet.

Um ponto importante para a agilidade no processo dos dados é utilizar a GPU do *Google Colab*. Tendo em vista o difícil acesso de computadores de alta capacidade de processamento de dados, o *Google Colab* surge como opção com acesso gratuito as GPUs com uma simples configuração interna. Assim, se torna mais ágil o desempenho do processamento em cálculos avançados de DL.

As bibliotecas utilizadas para o processamento dos dados envolveram a visualização dos dados, conexão com o google drive e bibliotecas de aprendizagem de máquina. A seguir são listadas as bibliotecas importadas no processamento:

- Tensorflow – utilização da GPU;
- Google Drive – armazenamento dos dados de entrada;
- OpenCV – visualização da imagem gerada com a detecção;
- Matplotlib – plotagem da imagem para visualização;

O *framework* Darknet possui um arquivo *Makefile* para a compilação do arquivo. Para a utilização da GPU, é necessário mudar o valor de GPU de 0 para 1. A ideia é que o valor 0 é para falso e 1 para positivo. Essa mudança também será feita no valor da OpenCV e CUDA *Deep Neural Network* (CUDNN), que é o tipo de GPU utilizado. Essas mudanças são necessárias para que o arquivo consiga executar de acordo com as mudanças que serão feitas nas suas dependências.

Para a utilização da OpenCV, deve-se primeiramente realizar a instalação da biblioteca para em seguida compilar com o *framework* para que o mesmo consiga reconhecer as mudanças no arquivo Makefile.

O *Google Colab* é uma *Integrated Development Environment* (IDE) com um ambiente *Jupyter notebook* livre. É armazenado na nuvem, não requer nenhuma instalação e é hospedado pela Google para incentivar pesquisas na área de Inteligência Artificial (GUNAWAN *et al.*, 2020).

5.3 PREPARAÇÃO DO *DATASET*

Em ambas as imagens de detecção propriamente dita e detecção de palmeiras não saudáveis foram feitos recortes da região para a criação do *dataset*. As regiões dos recortes foram escolhidas de forma espaçada, de modo que representasse toda região. Na região que foi feito o *dataset* de palmeiras não saudáveis foram feitos recortes nas áreas onde se observava melhor as palmeiras com tons amarelados, facilitando, assim, a criação do *dataset*.

A partir do *software* MultiSpec, que é um *software* que foi desenvolvido para analisar dados de imagens multiespectrais, as imagens da região foram visualizadas na escala real. Sendo assim, recortes das imagens foram escolhidos de modo que diversas áreas com palmeiras fossem selecionadas, oferecendo uma ampla distribuição de características contidas na imagem.

A coleta de dados para o treinamento do modelo envolve primeiramente o LabelImg que consiste em uma ferramenta de anotação em imagens para rotular as amostras. Rotular uma imagem é a primeira e mais significativa parte para a detecção de objetos, além disso, é um processo lento e manual, mas que quanto maior a precisão, melhor será o modelo.

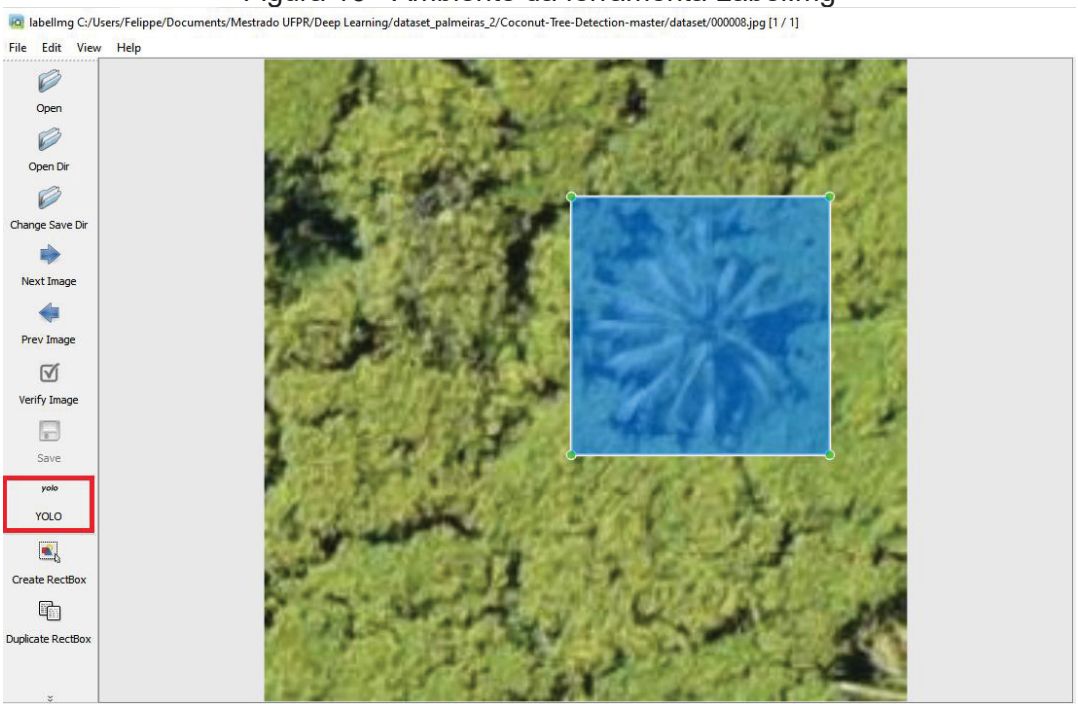
Alguns cuidados devem ser tomados na coleta de imagens para a criação do conjunto de dados (*datasets*). A parte manual não se limita apenas nas coletas de

imagens de forma aleatória, é necessário um trabalho de padronização dos tamanhos das imagens para que se tenham dimensões parecidas e de um tamanho que comporte o treinamento de acordo com a técnica. Assim sendo, as imagens devem ser catalogadas de formas diferentes, não agregando valor em imagens repetidas.

5.3.1 Rotulação dos objetos

Vale salientar a importância de se realizar a rotação dos objetos alvos para uma melhor generalização dos dados. A Figura 16 mostra o ambiente da ferramenta LabelImg.

Figura 16 - Ambiente da ferramenta LabelImg



Fonte: Autoria própria

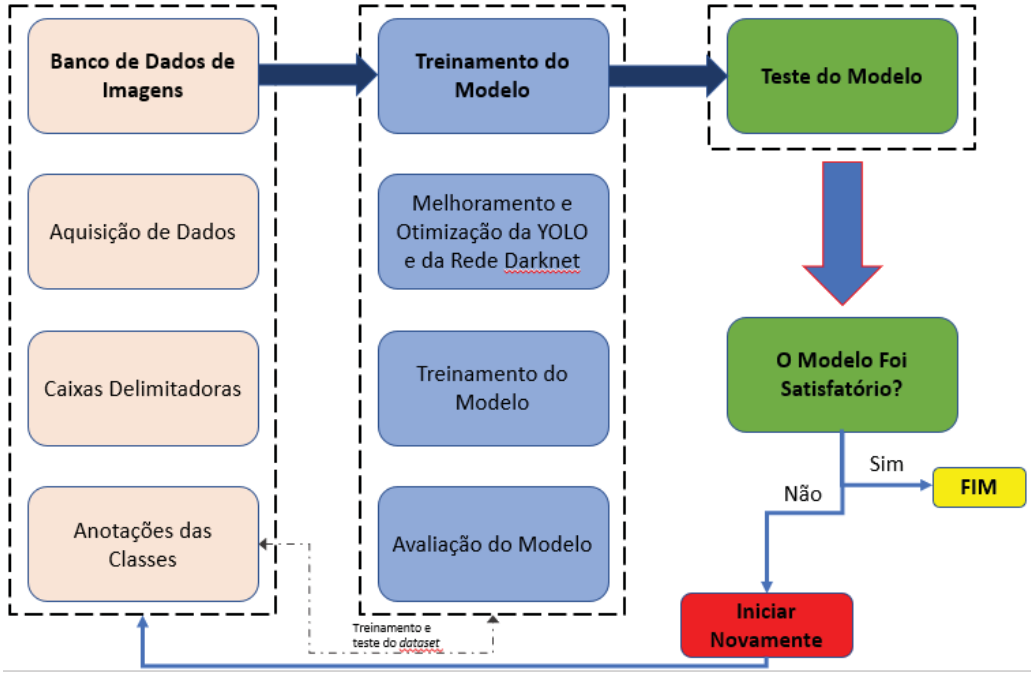
A ferramenta permite a criação de um retângulo para a criação da caixa delimitadora do objeto já no padrão de saída da YOLO, ou seja, no arquivo de saída cada linha corresponde a um objeto catalogado com seus respectivos valores de classe, coordenadas x e y do centro da caixa delimitadora, altura e largura.

5.4 METODOLOGIA

A presente pesquisa propõe o desenvolvimento de um sistema de detecção automatizado aplicado no sensoriamento remoto para detecção de palmeiras, capaz de caracterizar as particularidades dos alvos de estudo em imagens na faixa do RGB e avaliar seu respectivo desempenho, além da contagem do número de palmeiras em diferentes *datasets*.

O modelo de detecção das palmeiras foi desenvolvido em três etapas principais (Figura 18). Em primeiro lugar, foi criado o *dataset* com as imagens das palmeiras em imagens aéreas de alta resolução. Os objetos (palmeiras) foram anotados através de caixas delimitadoras nas suas respectivas classes. Já o segundo *dataset* foi criado com palmeiras para a detecção de teste em diferentes escalas. O terceiro *dataset* foi criado para a detecção de palmeiras catalogadas como “não saudáveis”, ou seja, que se apresentavam em tonalidades totalmente ou parcialmente amareladas. Em seguida, a rede YOLO foi treinada e otimizada nos dados desenvolvidos. A cada 100 etapas de treinamento, as métricas de avaliação foram calculadas para validar o desempenho da detecção. Finalmente, o melhor conjunto de peso foi selecionado para a detecção de palmeiras a partir de imagens aéreas de alta resolução. A Figura 17 mostra o fluxo geral.

Figura 17 - Fluxo geral da metodologia proposta



Fonte: Autoria própria

A primeira parte do fluxo metodológico se refere a preparação dos dados de entrada para o treinamento da rede. É necessário um banco de imagens de resolução adequada para atender a detecção de palmeiras. As caixas delimitadoras são feitas pelo operador a partir das imagens adquiridas de forma manual, fazendo a delimitação dos alvos de interesse e anotando a(s) classe(s) correspondente(s).

A segunda parte do processo se refere ao treinamento em si do modelo. O *Google Colab* permite uma conexão com o google drive onde os dados podem ser armazenados. Portanto, o treinamento é feito com o YOLOv4 em conjunto com os dados do Darknet armazenados na nuvem. A cada 100 épocas de treinamento é avaliado o treinamento do modelo através das métricas geradas.

A terceira e última parte do processo realiza a verificação do conjunto de dados de teste. Após o treinamento, podemos testar e visualizar em uma imagem de mesmas características que fazem parte do conjunto de teste (validação utilizando o OpenCV). Isso fornecerá o resultado visual da detecção através de caixas delimitadoras nas palmeiras. Além disso, a avaliação será feita através do *score* da caixa delimitadora, fornecendo a probabilidade de uma palmeira ter sido encontrada de forma correta.

5.5 TREINAMENTO DO MODELO

Antes de treinar a YOLO é necessário modificar alguns arquivos que são adquiridos a partir do descarregamento dos seus dados, de sua página do GitHub. A mudança é necessária, pois os arquivos estão com as configurações padrões, sendo preciso mudar para cada dado personalizado. Os arquivos necessários para o treinamento e que serão editados conforme o conjunto de dados são:

- Imagens dos objetos (palmeiras) já catalogadas junto com os arquivos das anotações das caixas delimitadoras;
- Arquivo com a extensão com informações do número de classe e local do arquivo;
- Arquivo com a extensão com o nome das classes dos objetos de estudo;
- Arquivo customizado com as alterações feitas de acordo com os dados a serem detectados;

- Arquivo com o nome das imagens para treino e das imagens de teste (opcional).

O processo de desenvolvimento do modelo para treinamento e teste da detecção através da YOLO com a Darknet foi feito no ambiente do *Google Colab* em conjunto com o *Google Drive* para o armazenamento das imagens, anotações e arquivos da rede Darknet.

Para realizar um treinamento com um bom desempenho, é necessário que se tenha um conjunto numeroso de imagens. Porém, o número ideal de imagens é incerto e varia para cada caso específico. Li *et al.* (2020) em seu trabalho para detecção de objetos a partir de imagens orbitais, utilizou mais de 23.000 imagens para 20 classes. Já Xia *et al.* (2018) utiliza 2.806 imagens, porém com mais de 188.000 caixas delimitadoras. Portanto, a quantidade de imagens e objetos variam para cada estudo, sendo importante conseguir o maior número de objetos detectáveis possível. Para este trabalho foram utilizados três *datasets* com 10, 20 e 40 imagens, com 261, 605 e 1301 amostras respectivamente.

Outro quesito importante para que se tenha um treinamento satisfatório é o cuidado no momento de demarcar as caixas delimitadoras dos objetos de estudo. O conjunto de dados de palmeiras normalmente se apresenta em um cenário de difícil visualização com palmeiras sobrepostas e semelhante às regiões próximas das palmeiras (outras árvores de parecida tonalidade, cor e formato). Para superar esse obstáculo, deve-se considerar uma intensa generalização dos dados de treinamento.

A próxima etapa para o treinamento é a configuração do arquivo da rede neural (cgf) para ajustar ao modelo o conjunto de dados. Para uma configuração padrão, serão alterados os seguintes parâmetros:

- *Subdivisions*. Resumidamente, esse parâmetro diz respeito ao consumo de memória. Se o valor for baixo, ele irá consumir mais memória e talvez tenha dificuldade de executar no *Google Colab*;
- *Width* e *Height*. Se refere ao tamanho da imagem. Na YOLOv4 o padrão é 608x608. Porém, caso o usuário queira um melhor desempenho, pode-se mudar o valor para 416x416 que era o valor recomendável para o YOLOv3;
- Max_batches que é o valor total de registros em cada um dos *batches.* A fórmula para calcular o parâmetro é: 2000 * número de classes;

- *Step*. Esse parâmetro diz respeito ao treinamento da rede. A cada época da rede neural ela vai executar um determinado número de *steps*. Como são dois valores, o primeiro será 80% do valor do Max_batches e o segundo será de 90% do Max_batches;
- Classes. Será apenas o número de classes que será igual a 2 (classe saudável e não-saudável). Ou igual a 1 no treinamento para a detecção geral das palmeiras;
- Filters = (classes + 5) * 3. A fórmula está explicada na documentação da YOLO e do Darknet. O 5 se refere aos parâmetros *width*, *height*, x, y e confiança (classes+1+1+1+1+1), e o número 3 se refere ao número de boxes que a YOLO detecta por célula.

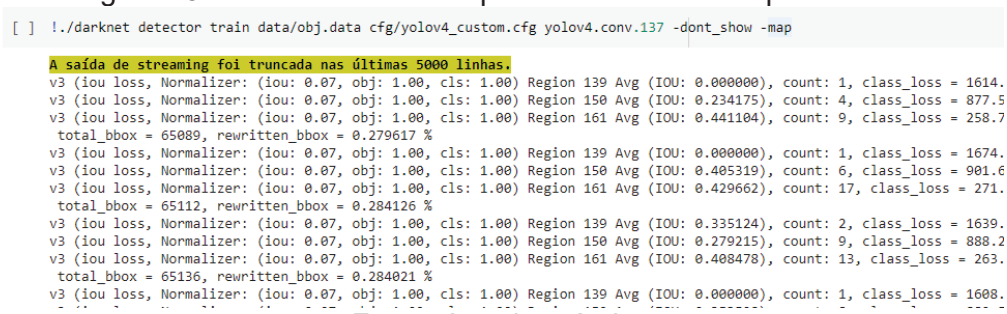
Para a detecção de palmeiras não saudáveis outra mudança necessária no arquivo da rede neural é a troca do valor de *hue* para 0. A YOLO é capaz de distinguir cores na detecção, basta uma separação correta das imagens para que cada cor fique em uma classe diferente. Assim, o parâmetro *hue* diz respeito ao color *augmentation*, que é utilizado para definir que na etapa de treinamento sejam criadas novas amostras (*data augmentation*) que variam minimamente em certas características da imagem (escala, rotação, luminosidade, etc) e uma dessas características seria o próprio tom de cores da imagem. O *data augmentation* é feito para tornar o detector mais robusto e preparado. Portanto, para distinguir melhor as cores dos objetos a serem detectados, não é desejável manter ativo o *augmentation*, ou seja, desabilitá-lo para o valor 0.

Após a configuração da rede e todo o conjunto de dados necessários mencionados anteriormente, existe uma necessidade de um processo de *transfer learning* para o processamento inicial e geração dos primeiros pesos da rede. O arquivo de pesos nessa fase é o yolov4.conv.137, obtido através do repositório do Darknet no site GitHub. Esse arquivo contém pesos de apenas algumas camadas convolucionais, mais precisamente até a camada com o id igual a 137. Ou seja, será utilizado um conjunto de pesos aleatórios para o conjunto de dados de treinamento de palmeiras.

O arquivo de peso inicial (yolov4.conv.137) funciona como uma sugestão. Para qualquer tipo de objeto, a detecção inicial irá encontrar bordas, linhas, cores e características padrões que estão contidas em qualquer tipo de imagem ou objeto, ou seja, ele apenas irá detectar informações mais básicas do objeto, nas primeiras

camadas de convolução em que se encontram essas características. A Figura 18 mostra a execução através do *framework* Darknet.

Figura 18 - Treinamento inicial por transferência de aprendizado



Fonte: Autoria própria

Usualmente, são necessárias pelo menos 2000 iterações para cada classe (objeto). Porém, a cada 100 iterações é gerado um novo arquivo de peso com as métricas de avaliações associadas. Assim, fica mais nítido a percepção do desenvolvimento do treinamento. Para as 100 primeiras épocas é comum ter uma perda média (*average loss*) alta, pois o treinamento ainda está muito no início. Para cada conjunto de dados o valor ideal de perda média irá variar. O valor médio é 3 para dados mais complexos, porém essa medida dependerá de diversas variáveis do treinamento (como o tamanho do conjunto de dados e a quantidade de anotações).

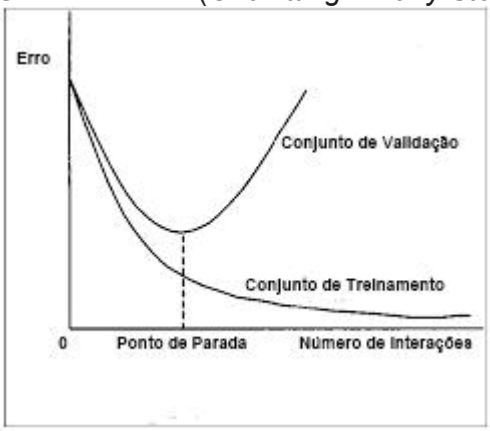
Outro resultado importante são as métricas de avaliação geradas pelo comando *map*. Além de fornecer o MAP, ele fornece dados de precisão, *recall*, F-1 *Score*, IoU e o tempo total de detecção.

Conforme mencionado anteriormente, é corriqueiro ter um treinamento de épocas de acordo com o número de classes multiplicado por 2000. Porém, é possível que após esse número o treinamento ainda tenha chance de melhorar e deva ser continuado.

Entretanto, erros podem acontecer no treinamento e nem sempre com mais iterações o sistema de detecção apresenta melhoras. Um dos erros mais comuns é o *Overfitting*. O treinamento pode se tornar tão especialista com a grande quantidade de épocas que pode causar confusão no momento da detecção, ou seja, pode-se treinar para 10.000 épocas, mas o melhor resultado pode ser de pesos anteriores (5.000 ou 6.000). O *Overfitting* ocorre quando se consegue detectar objetos nas imagens de treino, mas não consegue detectar nas imagens de teste. O

melhor ponto de parar o treinamento e onde são obtidos os melhores pesos é no *Early Stopping Point* (Figura 19).

Figura 19 - Gráfico de erro (*Overfitting* e *Early Stopping Point*)



Fonte: AlexyAB (2016)

6 RESULTADOS

A área da pesquisa apresenta palmeiras próximas umas das outras, o que dificulta para a detecção de alvos próximos. Srestasathiern e Rakwatin (2014) desenvolveram uma pesquisa para detecção de palmeiras com êxito, mas a região apresentava palmeiras espaçadas, facilitando a detecção e diminuindo a probabilidade de encontrar falsos negativos, por exemplo.

Após os resultados de detecção do *dataset* de teste (validação), será avaliado o número de palmeiras reconhecidas pelo treinamento com a YOLO v4, de acordo o número de verdadeiros positivos encontrados em relação aos falsos positivos e falsos negativos e com ajuda da verdade de campo. Este resultado irá servir como base para a criação do inventário da mesma. Além disso, fornecerá o *score* atribuído a cada um dos alvos.

6.1 EXPERIMENTO DE DETECÇÃO DE PALMEIRAS

O primeiro experimento focou na parte de detecção de palmeiras propriamente dita, na análise das métricas de acordo com o número de épocas e no número de imagens do *dataset*. O segundo experimento teve o objetivo de detectar palmeiras em escalas diferentes daquela da imagem original (1:400). Já o terceiro experimento foi concentrado na detecção de palmeiras não saudáveis (tonalidade amarelada) dentro de uma região com palmeiras saudáveis.

6.1.1 Experimento 1

O primeiro experimento foi obtido das imagens em resolução verdadeira. A detecção de palmeiras nesta etapa foi apreciada variando a quantidade de imagens de treinamento, número de épocas e palmeiras.

6.1.1.1 Experimento com 10 imagens

O *dataset* foi dividido inicialmente em quatro fases para testes do treinamento. A primeira fase incluiu um conjunto de 10 imagens, totalizando em 261 amostras. O

diferencial desta parte do treinamento é que nem todas as palmeiras contidas nas imagens foram catalogadas, indicando que o que não estava com a caixa delimitadora não pertenceria a classe de palmeiras. O treinamento foi até 2000 épocas por não apresentar melhoras significativas. A Tabela 1 mostra os resultados das métricas finais.

Tabela 1 - Métricas da primeira fase do treinamento

Métrica	Valor (%)
IoU	68,55
Precisão	81
Recall	98
F1-Score	89
MAP	**97,9**

Fonte: Autoria própria

6.1.1.2 Experimento com 20 imagens

Na segunda parte do treinamento novas imagens foram adicionadas totalizando em 20 imagens e 605 amostras de palmeiras. O treinamento mostrou melhorias para 2.000 épocas de treinamento como mostra a Tabela 2.

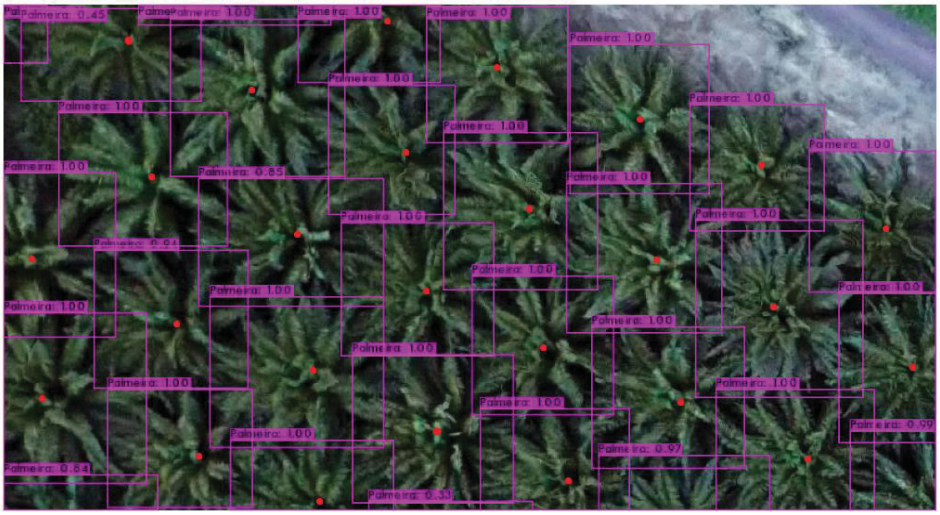
Tabela 2 - Métricas da segunda fase do treinamento

Métrica	Valor (%)
IoU	70,57
Precisão	83
Recall	98
F1-Score	90
MAP	**99**

Fonte: Autoria própria

No final do treinamento, também foi feito um teste de detecção utilizando uma imagem do conjunto de teste para a verificação da qualidade. Na imagem escolhida, continham 31 amostras de palmeiras, sendo 28 totalmente dispostas na imagem e 3 somente parciais cortadas pela imagem. Apesar de algumas estarem sobrepostas em relação as outras, a detecção conseguiu êxito, ou seja, sem falsos positivos ou negativos, detectando 32 palmeiras. A Figura 20 mostra o resultado.

Figura 20 - Exemplo do resultado de detecção de palmeiras geradas pela YOLO em conjunto com a Darknet na região. Anotações manuais estão representadas como circunferências vermelhas



Fonte: Autoria própria

6.1.1.3 Treinamento final utilizando 40 imagens

Na terceira fase do treinamento novas imagens foram incorporadas no conjunto de dados. No total, 40 imagens com 1.301 amostras de palmeiras foram catalogadas, mostrando uma alta no valor de incremento com o intuito de aumentar a espacialidade das amostras. O treinamento foi feito até 2000 épocas para grau de comparação com os treinamentos anteriores. A Tabela 3 apresenta os resultados.

Tabela 3 - Métricas da terceira fase do treinamento

Métrica	Valor (%)
IoU	71,55
Precisão	83
Recall	99
F1-Score	90
MAP	**98,9**

Fonte: Autoria própria

Ao final das três fases do treinamento foi possível observar variações nas métricas em cada um dos *datasets*, porém nada de forma significativa. A diferença do MAP da primeira fase de treinamento (10 imagens) comparado ao da terceira fase (40 imagens) foi de apenas 1%, evidenciando que mesmo após um considerável incremento de amostras (1.040), o valor do MAP não aumentou

proporcionalmente ao que era esperado. Portanto, isto é um indicativo que uma delimitação atenciosa das caixas delimitadoras já se mostra suficiente para ter um treinamento com qualidade pela YOLOv4 com a Darknet, sem necessidade de serem criados volumosos conjuntos de dados. Sendo assim, aproximadamente 300 amostras é um ponto de partida satisfatório para um treinamento em imagens provenientes de sensoriamento remoto.

Além disso, a avaliação das outras métricas, também no treinamento, não mostrou resultados significativos. Assim sendo, tanto o primeiro *dataset* com poucas imagens, quanto o terceiro com mais imagens e amostras de palmeiras, não mostraram nenhum indicador que o número maior de coleta de caixas delimitadoras fosse importante na evolução da qualidade das métricas. A Tabela 4 mostra um comparativo entre os três treinamentos no que diz respeito ao IoU.

Tabela 4 - Comparativo do IoU entre as épocas dos treinamentos

	IoU (%)		
Épocas	**1º Treinamento**	**2º Treinamento**	**3º Treinamento**
500	67,31	67,92	71,95
1000	70,71	70,96	70,47
1500	71,42	69,78	60,22
2000	68,55	70,57	71,55

Fonte: Autoria própria

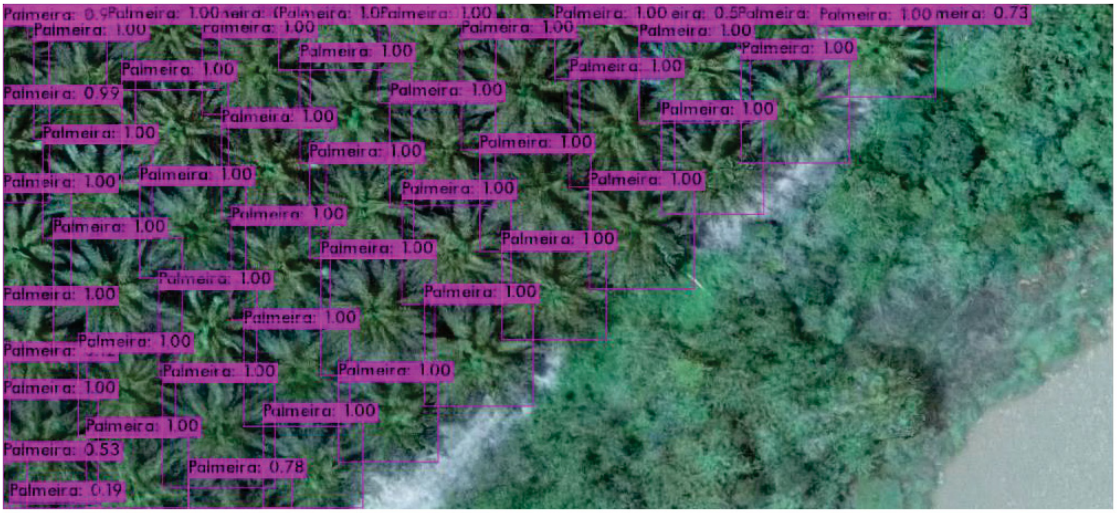
Apesar de algumas métricas apresentarem valores baixos, a detecção mostrou-se eficiente nas imagens de teste. A Figura 21 apresenta a seleção manual das 46 palmeiras que estão na imagem (algumas apenas de forma parcial).

Figura 21 - Palmeiras detectadas de forma manual de uma imagem do conjunto de teste

Fonte: Autoria própria

A mesma imagem foi inserida na detecção automática pela YOLOv4 com um limiar de apenas 10%, para uma melhor verificação de um maior número de palmeiras possíveis. A detecção mostrou bons resultados detectando 51 palmeiras. O número foi maior do que o real pelo fato de que na borda houve uma confusão, criando falsos positivos. A Figura 22 mostra o resultado da detecção automática.

Figura 22 - Detecção automática de palmeiras na região



Fonte: Autoria própria

Um ponto interessante que pode ser observado na detecção feita, é que as palmeiras na parte central da imagem obtiveram melhores resultados em relação as que estão na borda da imagem, isso acontece pelo fato de que estas palmeiras aparecem apenas de forma parcial na imagem. A Tabela 5 mostra o *score* de cada palmeira detectada na imagem de estudo.

Tabela 5 - *Score* das palmeiras detectadas na Figura 22

Quantidade	**Score (%)**
38	100
1	99
1	98
1	78
1	74
1	73
1	58
1	57
1	53
1	22
1	19
1	16
2	12

Fonte: Autoria própria

Portanto, 46 palmeiras entre as 51 detectadas conseguiram um *score* acima de 50%, mostrando eficiência na detecção de palmeiras. Além disso, outros tipos de vegetação estão contidos na imagem, mas a detecção não encontrou dificuldades em diferenciá-las das palmeiras.

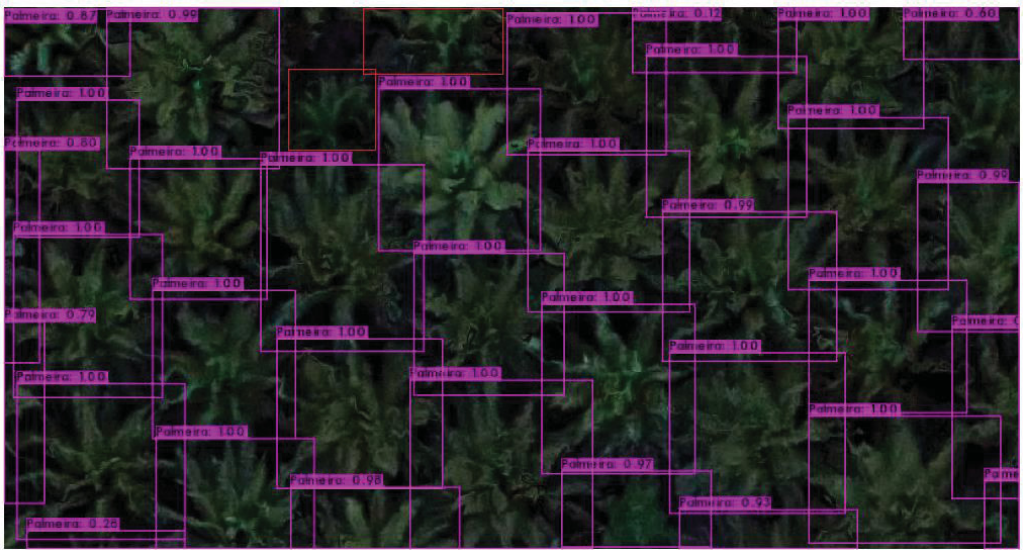
Como os recortes foram adquiridos ao longo da imagem original, alguns deles localizavam-se nas bordas da imagem. Assim, como não há uma sobreposição de imagens nas bordas, as palmeiras que se encontram nesta região ficam mais distorcidas na imagem. Porém, para uma detecção com qualidade de toda a região, foram inseridas no conjunto de dados de treinamento, palmeiras dessa região. A Figura 23 mostra a imagem que contém 36 palmeiras.

Figura 23 - Palmeiras detectadas manualmente na região da imagem

Fonte: Autoria própria

Apesar de bons valores de *scores* encontrados, a detecção apresentou dois valores de falso negativo, detectando 34 palmeiras, duas palmeiras a menos. Uma palmeira pode ter sido não detectada pelo tamanho diferente das demais; e a segunda palmeira por estar na borda imagem, que pode ter tido sua identificação dificultada devido a sua característica diferente em comparação as suas vizinhas. Ambos falsos negativos podem ser resolvidos adquirindo mais amostras para uma maior diversidade dos dados. A Figura 24 mostra o resultado da detecção.

Figura 24 - Resultado da detecção da imagem e seus falos negativos



Fonte: Autoria própria

6.1.2 Experimento 2

Um segundo experimento de detecção foi criado para detecção de palmeiras em escalas diferentes com o mesmo conjunto de dados apresentado (40 imagens). Sabe-se que cada tipo de imagem tem suas particularidades como tonalidade, iluminação e escala, por exemplo, portanto o objetivo é averiguar se a YOLOv4 é capaz de detectar as mesmas palmeiras da região em outros ambientes (em escalas diferentes). Para isso foram feitos recortes nas escalas 1:400, 1:500 e 1:600, ou seja, estavam com escalas diferentes das imagens originais feitas na visão original no *software* MultiSpec. A Figura 25 mostra as imagens originais.

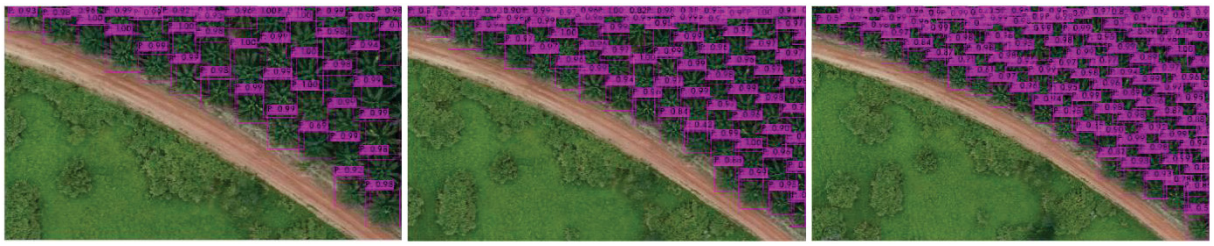
Figura 25 – Recortes na escala: (A) 1:400; (B) 1:500; e (C) 1:600.

Fonte: Autoria Própria

A detecção foi de acordo com um limiar de 30% para uma melhor qualidade das palmeiras detectadas. Outro fator importante é que a contagem manual foi levada em conta apenas para palmeiras que eram reveladas em pelo menos metade da sua folhagem. Ou seja, palmeiras que eram exibidas em pequena parte não foram levadas em consideração na contagem manual para validação dos dados.

A detecção mostrou resultado satisfatório comparado com as palmeiras reais das imagens da Figura 25. O recorte da Figura 25 (C) continha 98 palmeiras e 97 foram detectadas; na área da Figura 25 (B) tinham 68 palmeiras e 71 foram detectadas. Um dos motivos para a detecção encontrar mais palmeiras que o real (Falsos Positivos), é que as palmeiras que estavam parcialmente aparecendo na borda com pequenas folhas foram detectadas; e por fim, a região da Figura 25 (A) foi detectada pela YOLO 40 palmeiras das 42 possíveis. A Figura 26 mostra o resultado.

Figura 26 - (A) detecção na escala 1:400; (B) detecção na escala 1:500; e (C) detecção na escala 1:600.



Fonte: Autoria própria

6.1.3 Experimento 3

Para a detecção de palmeiras não saudáveis foi utilizado o mesmo conjunto de imagens, porém a escala de 1:500 foi escolhida para uma melhor visibilidade da região de interesse. A área apresentava algumas palmeiras com cores amareladas, indicando alguma deficiência na saúde da palmeira, podendo ser um problema fitossanitário (doença) conhecido como “Amarelecimento Fatal” ou problemas de deficiência nutricional. A Figura 27 mostra a região.

Figura 27 - Recorte da região na escala 1:1000

Fonte: Autoria própria

A imagem foi dividida em recortes e foi construído um conjunto de dados com 25 imagens e 331 amostras para o treinamento de 2.000 épocas apenas com uma classe de “palmeiras não saudáveis”. Nesta parte do conjunto de dados, as imagens de entrada tinham o tamanho de 975 x 574 *pixels*. A Figura 28 ilustra exemplos de palmeiras não saudáveis.

Figura 28 - Exemplo de palmeira não saudável

Fonte: Autoria própria

A questão mais importante foi identificar as palmeiras não saudáveis de forma padronizada e com critério definido. A percepção pode mudar de uma imagem para outra, deixando o detector pouco preciso pelas constatações errôneas de quem está anotando as caixas delimitadoras. Dessa forma, uma palmeira com alguns tons de amarelo não pode ser marcada como palmeira em uma imagem, da mesma forma que outra palmeira com as mesmas características em outra imagem também não pode ser marcada. A Tabela 6 mostra o resultado das detecções das palmeiras não saudáveis.

Tabela 6 - Resultado das métricas de avaliação do treinamento para detecção de palmeiras não saudáveis

Métrica de Avaliação	**Valor**
Precisão	0,82
Recall	0,82
MAP	0,78
F1-Score	0,82
IOU	0,70
Avg. Loss	1,32

Fonte: Autoria própria

Ao final do treinamento, foi observado que o detector teve uma tendência de encontrar palmeiras com tons amarelados, evidenciando alguma deficiência. Algumas palmeiras apresentaram tons amarelados, mas algumas foram detectadas e outras não. Porém, o detector mostrou um padrão de não encontrar falsos

positivos, apesar de alguns falsos negativos terem sido encontrados de acordo com o que foi padronizado na construção do *dataset.* A Figura 29 mostra o recorte original da região e Figura 30 sua respectiva detecção.

Figura 29 - Exemplo de palmeira não saudável

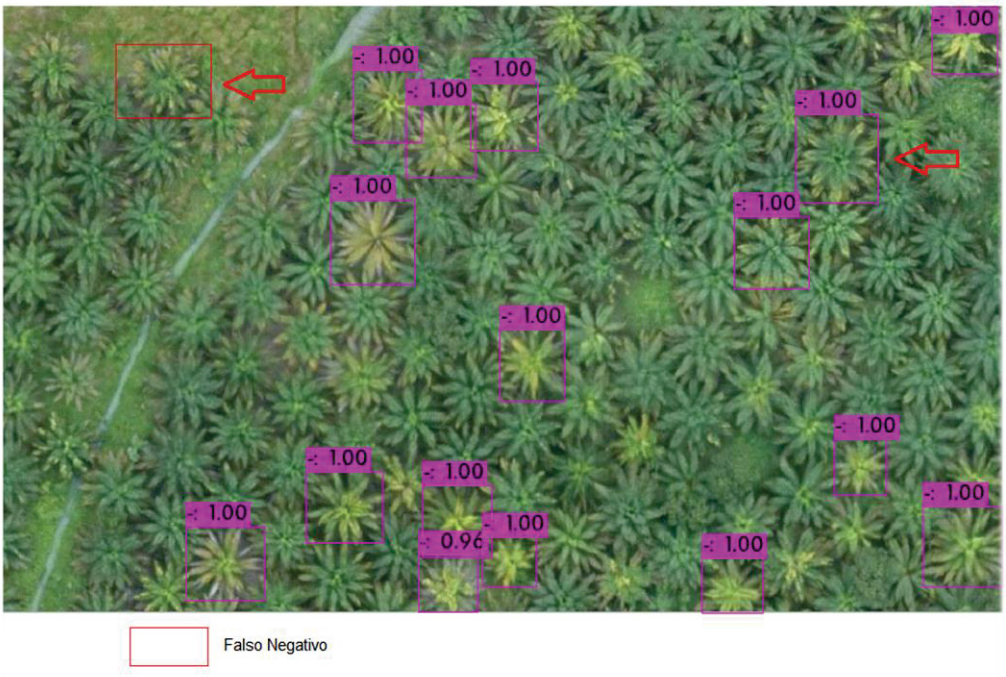
Fonte: Autoria própria

Figura 30 - Recorte de palmeiras não saudáveis detectadas

Fonte: Autoria própria

Outras regiões apresentaram bons resultados na detecção de palmeiras não saudáveis, mas alguns falsos negativos ocorreram como na Figura 31. Portanto, como foi mencionado anteriormente, não foi encontrado nenhum falso positivo.

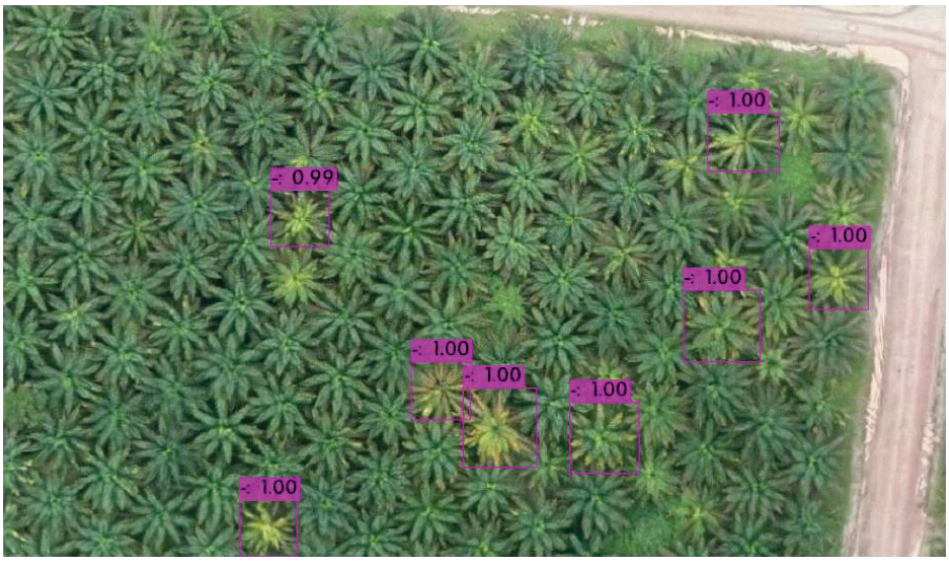
Figura 31 - Resultado de detecção de palmeiras não saudáveis



Fonte: Autoria própria

Como pode ser observado, duas palmeiras de características semelhantes com folhas de tonalidade amarelada nas suas bordas não apresentaram a mesma classe para o detector. Já na Figura 32 o mesmo não ocorreu devido a um grau maior de diferença entre as palmeiras saudáveis e não saudáveis. Assim, a detecção da YOLOv4 com a Darknet, e mais alguns ajustes no arquivo rede, foi capaz de distinguir as cores indicativas de algum tipo de anomalia na saúde da palmeira.

Figura 32 - Detecção de palmeiras não saudáveis



Fonte: Autoria própria

Também foi testada a possibilidade da detecção em escala diferente da que foi submetida ao treinamento. Para isso, foram utilizados recortes na escala 1:400. O desempenho se mostrou promissor como mostra a Figura 33.

Figura 33 - Detecção na escala 1:400

Fonte: Autoria Própria

Este processo é importante por economizar custo e tempo criando um segundo *dataset* para uma escala diferente. Isto mostra que a detecção com a YOLOv4 e a Darknet é capaz de generalizar a detecção em diferentes situações, como por exemplo a altura de voo, luminosidade e escala.

7 CONCLUSÔES

Nesta pesquisa foi proposta uma abordagem para detecção e contagem de palmeiras em diferentes cenários utilizando diferentes escalas e números de amostras para a detecção de objetos. Além disso, foi testada a capacidade da técnica YOLO em conjunto com a rede Darknet-53 em detectar e distinguir palmeiras não saudáveis das saudáveis.

Em relação a detecção de palmeiras, o número de imagens e anotações mostrou-se eficaz para a proposta apresentada nos três experimentos. O experimento 1 alcançou um valor de MAP de 98,9% com 1.301 amostras; o experimento 2, focado em detecção em diferentes escalas, conseguiu detectar mais de 90% das palmeiras; por fim, o experimento 3 detectou a maioria das palmeiras não saudáveis com poucos falsos positivos e falsos negativos, obtendo um valor de MAP de 78%.

As palmeiras do conjunto de dados apresentavam condições necessárias para uma detecção adequada, não possuindo variações na luminosidade e nem sombras pelo fato da proximidade uma das outras. Porém, quando o número foi aumentado, não houve um aumento proporcional em relação às métricas, ou seja, o experimento com 40 imagens não mostrou uma evolução das métricas de avaliação esperada comparado ao experimento com 10 imagens. Assim sendo, a YOLO com a Darknet-53 pode produzir resultados com altos valores das métricas de precisão mesmo com um *dataset* considerado pequeno.

A pesquisa foi capaz de criar *datasets* para uma eficaz detecção e contagem de palmeiras com a YOLOv4 em conjunto com a arquitetura Darknet-53. Além do mais, o algoritmo foi capaz de diferenciar palmeiras saudáveis e não saudáveis a partir da aplicação de uma um *dataset* detalhado para esta proposta específica.

Sob o ponto de vista científico a pesquisa vai ao encontro com outras propostas de pesquisas nacionais e internacionais com um foco na detecção propriamente dita das palmeiras. Porém, a pesquisa apresentada teve um foco mais abrangente na variabilidade de cenários (diferentes escalas e número de amostras). Além disso, foi testada a possibilidade da técnica YOLOv4 distinguir cores, mesmo aquelas que se encontram mais suavizadas.

A literatura apesar de avançada, produz resultados pouco diferentes no que diz respeito à detecção e a saúde das palmeiras e seu amarelecimento. As

pesquisas possuem um foco maior na análise em si e na comparação em diferentes técnicas, não explorando a variabilidade dos dados. Além disso, há baixa exploração da variabilidade de dados e, locais com palmeiras mais espaçadas, são usualmente estudo de caso, facilitando a detecção pela rede.

Com avanços na área de sensoriamento remoto e inteligência artificial, esta pesquisa contribui para o desenvolvimento de sistemas e técnicas de contagem e detecção de palmeiras, facilitando, sobretudo, a identificação das que possuem folhas amareladas. Vale lembrar que a pesquisa pode ainda contribuir para outros tipos de árvore, trazendo impactos econômicos para produtores.

Entende-se que os objetivos da pesquisa foram alcançados. Entre os resultados, está a eficácia em detectar palmeiras em condições adversas da região dos recortes do *dataset* de treinamento, bem como a capacidade de identificar mudanças espectrais das folhas das palmeiras não saudáveis.

Quanto as lições aprendidas sobre a YOLOv4 e a rede Darknet, destaca-se a capacidade da rede detectar com alta precisão palmeiras próximas umas das outras e a capacidade de distinguir detalhes de cores com pequenas modificações. Além disso, todo o processo apresentou altos valores das métricas de precisão sem precisar de um volumoso conjunto de dados.

8 RECOMENDAÇÃO PARA TRABALHOS FUTUROS

Como sugestões futuras para trabalhos envolvendo a temática, pode ser comentado a comparação de detecção nas diferentes versões da YOLO que são mais atuais, verificando se houve melhora na detecção de palmeiras nas versões mais recentes. Além disso, outra questão a ser explorada é a capacidade de detecção em imagens multiespectrais, principalmente para o caso de detecção de palmeiras não saudáveis.

REFERÊNCIAS

ABIODUN, Oludare Isaac *et al.* State-of-the-art in artificial neural network applications: A survey. **Heliyon**, v. 4, n. 11, p. 938, 2018.

AGARWAL, Shivang; TERRAIL, Jean Ogier Du; JURIE, Frédéric. Recent advances in object detection in the age of deep convolutional neural networks. **arXiv:1809.03193**, 2018.

AMMAR, Adel; KOUBAA, Anis. Deep-Learning-based Automated Palm Tree Counting and Geolocation in Large Farms from Aerial Geotagged Images. **arXiv:2005.05269**, 2020.

AMIT, Yali; FELZENSZWALB, Pedro; GIRSHICK, Ross. Object detection. **Computer Vision: A Reference Guide**, p. 1-9, 2020.

APTE, Maneesh; MANGAT, Simar; SEKHAR, Priyanka. YOLO Net on iOS. **CS231n. Stanford University**, 2017

BOCHKOVSKIY, Alexey; WANG, Chien-Yao; LIAO, Hong-Yuan Mark. Yolov4: Optimal speed and accuracy of object detection. **arXiv preprint arXiv:2004.10934**, 2020.

BÖHM, Samuel M. et al. Análise de Performance de um Algoritmo de Reconhecimento Facial por Visão Computacional Aplicado a Sistemas Embarcados. 2021.

CHEN, Yushi *et al.* Deep learning-based classification of hyperspectral data**. IEEE Journal of Selected topics in applied earth observations and remote sensing**, v. 7, n. 6, p. 2094-2107, 2014.

CHENG, Gong; HAN, Junwei. A survey on object detection in optical remote sensing images. **ISPRS Journal of Photogrammetry and Remote Sensing**, v. 117, p. 11-28, 2016.

CRESSON, Rémi. **Deep Learning for Remote Sensing Images with Open Source Software**. CRC Press, 2020.

CULMAN, María; DELALIEUX, Stephanie; VAN TRICHT, Kristof. Individual palm tree detection using deep learning on RGB imagery to support tree inventory. **Remote Sensing**, v. 12, n. 21, p. 3476, 2020.

DE MILANO, Danilo; HONORATO, Luciano Barrozo. Visao computacional. 2014.

DING, Xinyi *et al.* Swapped face detection using deep learning and subjective assessment. **EURASIP Journal on Information Security**, v. 2020, p. 1-12, 2020.

DUARTE, Amanda Cardoso. **Geração de conjuntos de dados para aplicações de visão computacional em ambientes subaquáticos**. 2017. Dissertação de Mestrado.

FARHODOV, Xurshedjon et al. Faster RCNN detection based OpenCV CSRT tracker using drone data. In: **2019 International Conference on Information Science and Communications Technologies (ICISCT)**. IEEE, 2019. p. 1-3.

FREUDENBERG, Maximilian *et al.* Large scale palm tree detection in high resolution satellite images using U-Net. **Remote Sensing**, v. 11, n. 3, p. 312, 2019.

FU, Cheng-Yang *et al.* Dssd: Deconvolutional single shot detector. **arXiv preprint arXiv:1701.06659**, 2017.
HUANG, Rachel; PEDOEEM, Jonathan; CHEN, Cuixian. YOLO-LITE: a real-time object detection algorithm optimized for non-GPU computers. In: **2018 IEEE International Conference on Big Data (Big Data)**. IEEE, 2018. p. 2503-2510.

GIRSHICK, Ross *et al.* Rich feature hierarchies for accurate object detection and semantic segmentation. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. 2014. p. 580-587.

GIRSHICK, Ross. Fast r-cnn. In: **Proceedings of the IEEE international conference on computer vision**. 2015. p. 1440-1448.

GUNAWAN, Teddy Surya *et al.* Development of video-based emotion recognition using deep learning with *Google Colab*. **TELKOMNIKA**, v. 18, n. 5, p. 2463-2471, 2020.

HENDERS, Sabine; PERSSON, U. Martin; KASTNER, Thomas. Trading forests: land-use change and carbon emissions embodied in production and exports of forest-risk commodities. **Environmental Research Letters**, v. 10, n. 12, p. 125012, 2015.

HUI, Jonathan. Object Detecction: speed and acuracy comparsion (Faster R-CNN, R-FCN, SSD, FPN, RetinaNet and YOLOv3), 27 de mar. de 2018. Disponível em:< https://jonathan-hui.medium.com/object-detection-speed-and-accuracy-comparison-faster-r-cnn-r-fcn-ssd-and-yolo-5425656ae359>. Acesso em: 12 de jul. de 2021.

HOESER, Thorsten; BACHOFER, Felix; KUENZER, Claudia. Object detection and image segmentation with deep learning on Earth observation data: A review—Part II: Applications. **Remote Sensing**, v. 12, n. 18, p. 3053, 2020.

IOFFE, Sergey; SZEGEDY, Christian. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: **International conference on machine learning**. PMLR, 2015. p. 448-456.

IVAŠIĆ-KOS, Marina; KRIŠTO, Mate; POBAR, Miran. Human detection in thermal imaging using YOLO. In: **Proceedings of the 2019 5th International Conference on Computer and Technology Applications**. 2019. p. 20-24

IZEBOUDJEN, Nouma; LARBES, C.; FARAH, Ahcene. A new classification approach for neural networks hardware: from standards chips to embedded systems on chip. **Artificial Intelligence Review**, v. 41, n. 4, p. 491-534, 2014.

JIANG, Qiling *et al.* Deep neural networks-based vehicle detection in satellite images. In: **2015 International Symposium on Bioelectronics and Bioinformatics (ISBB)**. IEEE, 2015. p. 184-187.

JIANG, Jiahuan *et al.* High-speed lightweight ship detection algorithm based on YOLO-v4 for three-channels RGB SAR image. **Remote Sensing**, v. 13, n. 10, p. 1909, 2021.

JINTASUTTISAK, Thani; EDIRISINGHE, Eran; ELBATTAY, Ali. Deep neural network based date palm tree detection in drone imagery. **Computers and Electronics in Agriculture**, v. 192, p. 106560, 2022.

KAEHLER, Adrian; BRADSKI, Gary**. Learning OpenCV 3: computer vision in C++ with the OpenCV library**. " O'Reilly Media, Inc.", 2016.

KAMILARIS, Andreas; PRENAFETA-BOLDÚ, Francesc X. Deep learning in agriculture: A survey. **Computers and electronics in agriculture**, v. 147, p. 70-90, 2018.

KHAN, Asjad M. **Vehicle and Pedestrian Detection Using YOLOv3 and YOLOv4 for Self-Driving Cars**. 2021. Tese de Doutorado. California State University San Marcos.

KONG, Fanjie *et al.* The Synthinel-1 dataset: a collection of high resolution synthetic overhead imagery for building segmentation. In: **Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision**. 2020. p. 1814-1823.

KOVÁCS, Zsolt László. **Redes neurais artificiais**. Editora Livraria da Fisica, 2002.

KUSSUL, Nataliia *et al.* Deep learning classification of land cover and crop types using remote sensing data. **IEEE Geoscience and Remote Sensing Letters**, v. 14, n. 5, p. 778-782, 2017.

LI, Ke *et al.* Object detection in optical remote sensing images: A survey and a new benchmark. **ISPRS Journal of Photogrammetry and Remote Sensing**, v. 159, p. 296-307, 2020.

LONG, Yang *et al.* Accurate object localization in remote sensing images based on convolutional neural networks. **IEEE Transactions on Geoscience and Remote Sensing**, v. 55, n. 5, p. 2486-2498, 2017.

LI, He; WANG, Peng; HUANG, Chong. Comparison of Deep Learning Methods for Detecting and Counting Sorghum Heads in UAV Imagery. **Remote Sensing**, v. 14, n. 13, p. 3143, 2022.

MA, Lei *et al.* Deep learning in remote sensing applications: A meta-analysis and review. **ISPRS journal of photogrammetry and remote sensing**, v. 152, p. 166-177, 2019

MANTRIPRAGADA, Manogna. **Digging deep into YOLO V3 – A hands-on guide Part 1: An attempt to familiarize ourselves with the YOLO v3 model**, 15 de Ago. de 2020. Disponível em: https://towardsdatascience.com/digging-deep-into-yolo-v3-a-hands-on-guide-part-1-78681f2c7e29. Acesso em 13 de jul. 2021.

MARENGONI, Maurício; STRINGHINI, Stringhini. Tutorial: Introdução à visão computacional usando opencv**. Revista de Informática Teórica e Aplicada**, v. 16, n. 1, p. 125-160, 2009.

MENG, Lingxuan *et al.* Real-time detection of ground objects based on unmanned aerial vehicle remote sensing with deep learning: Application in excavator detection for pipeline safety. **Remote Sensing**, v. 12, n. 1, p. 182, 2020

MICHELUCCI, Umberto. **Advanced Applied Deep Learning: Convolutional Neural Networks and Object Detection**. Apress, 2019. 285p.

MILANO, Danilo de; HONORATO, Luciano Barrozo. Visão computacional. Universidade Estadual de Campinas, p. 1-7, 2010.

MOLCHANOV, V. V. *et al.* Pedestrian detection in video surveillance using fully convolutional yolo neural network. In: **Automated visual inspection and machine vision II**. International Society for Optics and Photonics, 2017. p. 103340Q.

MONGABAY. **Desmatamento e água contaminada: o lado obscuro do óleo de palma 'sustentável' da Amazônia**. Disponível em: https://brasil.mongabay.com/2021/03/desmatamento-e-agua-contaminada-o-lado-obscuro-do-oleo-de-palma-sustentavel-da-amazonia/. Acesso em 31 de mai. 2021.

MOUNTRAKIS, Giorgos; IM, Jungho; OGOLE, Caesar. Support vector machines in remote sensing: A review. **ISPRS Journal of Photogrammetry and Remote Sensing**, v. 66, n. 3, p. 247-259, 2011

NAGI, Jawad *et al.* Max-pooling convolutional neural networks for vision-based hand gesture recognition. In: **2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)**. IEEE, 2011. p. 342-347.

NEPAL, Upesh; ESLAMIAT, Hossein. Comparing YOLOv3, YOLOv4 and YOLOv5 for autonomous landing spot detection in faulty UAVs. **Sensors**, v. 22, n. 2, p. 464, 2022.

O'MAHONY, Niall *et al.* Deep learning vs. traditional computer vision. In: **Science and Information Conference**. Springer, Cham, 2019. p. 128-144.

OSÓRIO, Fernando S.; BITTENCOURT, João R. Sistemas inteligentes baseados em redes neurais artificiais aplicados ao processamento de imagens. In: **I WORKSHOP**

DE INTELIGÊNCIA ARTIFICIAL UNISC–Universidade de Santa Cruz do Sul Departamento de Informática-Junho. 2000.

PADILLA, Rafael; NETTO, Sergio L.; DA SILVA, Eduardo AB. A survey on performance metrics for object-detection algorithms. In: **2020 International Conference on Systems, Signals and Image Processing (IWSSIP)**. IEEE, 2020. p. 237-242.

PEDLOWSKI, Marcos. A produção de óleo de palma na Amazônia brasileira ameaça os compromissos do NDPE contra o desmatamento. **Blog do Pedlowski**, Campos dos Goytacazes, 08 de dez. de 2021. Disponível em: <https://blogdopedlowski.com/2021/12/08/a-producao-de-oleo-de-palma-na-amazonia-brasileira-ameaca-os-compromissos-do-ndpe-contra-o-desmatamento >. Acesso em: 23 de jul. de 2022.

PHAM, Minh-Tan *et al.* YOLO-Fine: One-stage detector of small objects under various backgrounds in remote sensing images. **Remote Sensing**, v. 12, n. 15, p. 2501, 2020.

PIRES DE LIMA, Rafael; MARFURT, Kurt. Convolutional neural network for remote-sensing scene classification: Transfer learning analysis. **Remote Sensing**, v. 12, n. 1, p. 86, 2020.

RADOVIC, Matija; ADARKWA, Offei; WANG, Qiaosong. Object recognition in aerial images using convolutional neural networks. **Journal of Imaging**, v. 3, n. 2, p. 21, 2017.

RAUBER, Thomas Walter. Redes neurais artificiais**. Universidade Federal do Espírito Santo**, v. 29, 2005.

REDMON, Joseph *et al.* You only look once: Unified, real-time object detection. *In:* **Proceedings of the IEEE conference on computer vision and pattern recognition**. 2016. p. 779-788.

REDMON, Joseph; FARHADI, Ali. YOLO9000: better, faster, stronger. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. 2017. p. 7263-7271.

REDMON, Joseph; FARHADI, Ali. Yolov3: An incremental improvement. **arXiv preprint arXiv:1804.02767**, 2018.

REN, Shaoqing *et al.* Faster r-cnn: Towards real-time object detection with region proposal networks. **arXiv preprint arXiv:1506.01497**, 2015.

RÖPKE, Leonardo Luís; BINELO, Manuel Osório. Desenvolvimento de Sistema de Inteligência Artificial (IA) Baseado em Reconhecimento de Padrões Para Análise de Rotas Veiculares. **Inteligencia artificial: Revista Iberoamericana de Inteligencia Artificial**, v. 23, n. 65, p. 67-85, 2020.

ROSENBROCK, Adrian. Intersection over Union (IoU) for object detection. Py image Search, 07 de nov. de 2016. Disponível em:< https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/ >. Acesso em: 17 de jun. de 2021.

RUCK, Dennis W.; ROGERS, Steven K.; KABRISKY, Matthew. Feature selection using a multilayer perceptron. **Journal of Neural Network Computing**, v. 2, n. 2, p. 40-48, 1990.

RUDEL, Thomas K. *et al.* Changing drivers of deforestation and new opportunities for conservation. **Conservation Biology**, v. 23, n. 6, p. 1396-1405, 2009.

RŮŽIČKA, Vít; FRANCHETTI, Franz. Fast and accurate object detection in high resolution 4K and 8K video using GPUs. In: **2018 IEEE High Performance extreme Computing Conference (HPEC)**. IEEE, 2018. p. 1-7.

SAITOH, Koki. Deep Learning from the Basics: Python and Deep Learning: Theory and Implementation. Packt, 2021. 316p.

SETIYONO, Budi; AMINI, Dyah Ayu; SULISTYANINGRUM, Dwi Ratna. Number plate recognition on vehicle using YOLO-Darknet. In: **Journal of Physics: Conference Series**. IOP Publishing, 2021. p. 012049.

SHAFRI, Helmi ZM; HAMDAN, Nasrulhapiza; SARIPAN, M. Iqbal. Semi-automatic detection and counting of oil palm trees from high spatial resolution airborne imagery. **International journal of remote sensing**, v. 32, n. 8, p. 2095-2115, 2011.

SHI, Pengfei *et al.* Oil Well Detection via Large-Scale and High-Resolution Remote Sensing Images Based on Improved YOLO v4. **Remote Sensing**, v. 13, n. 16, p. 3243, 202

SRESTASATHIERN, Panu; RAKWATIN, Preesan. Oil palm tree detection with high resolution multi-spectral satellite imagery. **Remote Sensing**, v. 6, n. 10, p. 9749-9774, 2014.

SWIEZEWSKI, Jedrzej. Intersection over Union (IoU) for object detection. Py Image Search, 07 de nov. de 2016. Disponível em:< https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/ >. Acesso em: 77 de jun. de 2021.

TORDEUX, Antoine *et al.* Prediction of pedestrian speed with artificial neural networks. In: **International Conference on Traffic and Granular Flo**w. Springer, Cham, 2017. p. 327-335.

VARGAS, Ana Caroline Gomes; PAES, Aline; VASCONCELOS, Cristina Nader. Um estudo sobre redes neurais convolucionais e sua aplicação em detecção de pedestres. In: **Proceedings of the XXIX conference on graphics, patterns and images**. sn, 2016.

VETRIVEL, Anand *et al.* Disaster damage detection through synergistic use of deep learning and 3D point cloud features derived from very high resolution oblique aerial images, and multiple-kernel-learning. **ISPRS journal of photogrammetry and remote sensing**, v. 140, p. 45-59, 2018.

VOULODIMOS, Athanasios *et al.* Deep learning for computer vision: A brief review. **Computational intelligence and neuroscience**, v. 2018, 2018.

WANG, Chien-Yao *et al.* CSPNet: A new backbone that can enhance learning capability of CNN. In: **Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops**. 2020. p. 390-391.

WEBB, Andrew R. **Statistical pattern recognition**. John Wiley & Sons, 2003.

WEINSTEIN, Ben G. *et al.* Individual tree-crown detection in RGB imagery using semi-supervised deep learning neural networks. **Remote Sensing**, v. 11, n. 11, p. 1309, 2019.

WOEBBECKE, David M. *et al.* Plant species identification, size, and enumeration using machine vision techniques on near-binary images. In: **Optics in Agriculture and Forestry**. International Society for Optics and Photonics, 1993. p. 208-219.

WU, Xiongwei; SAHOO, Doyen; HOI, Steven CH. Recent advances in deep learning for object detection. **Neurocomputing**, v. 396, p. 39-64, 2020.

XIA, Gui-Song *et al.* DOTA: A large-scale dataset for object detection in aerial images. In**: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. 2018. p. 3974-3983.

XIAO, Jianxiong et al. Sun database: Large-scale scene recognition from abbey to zoo. In: **2010 IEEE computer society conference on computer vision and pattern recognition**. IEEE, 2010. p. 3485-3492.

YANG, Wang; JIACHUN, Zheng. Real-time face detection based on YOLO. In: **2018 1st IEEE international conference on knowledge innovation and invention (ICKII)**. IEEE, 2018. p. 221-224.

YARAK, Kanitta *et al.* Oil Palm Tree Detection and Health Classification on High-Resolution Imagery Using Deep Learning. **Agriculture**, v. 11, n. 2, p. 183, 2021.

YILDIRIM, Esra; KAVZOGLU, Taskin. Ship detection in optical remote sensing images using YOLOv4 and Tiny YOLOv4. In: **The Proceedings of the International Conference on Smart City Applications**. Springer, Cham, 2021. p. 913-924

YINGGE, Huo; ALI, Imran; LEE, Kang-Yoon. Deep Neural Networks on Chip-A Survey. In: **2020 IEEE International Conference on Big Data and Smart Computing (BigComp)**. IEEE, 2020. p. 589-592.

ZAFAR, Iffat; TZANIDOU, Giounona; BURTON, Richard; PATEL, Nimesh; ARAUJO LEONARDO. Hands-On Convolutional Neural Networks with TensorFlow: Solve

Computer Vision Problems with modeling in TensorFlow and Python. Packt, 2018. 274 p.

ZAKRIA, Zakria *et al.* Multiscale and direction target detecting in remote sensing images via modified YOLO-v4. **IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing**, v. 15, p. 1039-1048, 2022.

ZELINSKY, Alex. Learning OpenCV---Computer vision with the OpenCV library (Bradski, GR et al.; 2008)[On the Shelf]. **IEEE Robotics & Automation Magazine**, v. 16, n. 3, p. 100-100, 2009.

ZHANG, Feng; ZHU, Xiatian; YE, Mao. Fast human pose estimation. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. 2019. p. 3517-3526.