CENTRO DE INFORMÁTICA
UNIVERSIDADE FEDERAL DA PARAÍBA (UFPB)

LUCAS MOREIRA E SILVA ALVES

Análise da Acurácia de Detecção de Placa de Veículos com Yolov3 em Imagens com Características de Brilho, Resolução e Desfoque Alteradas

JOÃO PESSOA - PB
2023

LUCAS MOREIRA E SILVA ALVES

Análise da Acurácia de Detecção de Placa de Veículos com Yolov3 em Imagens com Características de Brilho, Resolução e Desfoque Alteradas

Monografia apresentada ao curso Engenharia de Computação do Centro de Informática, da Universidade Federal da Paraíba, como requisito para a obtenção do grau de Bacharel em Engenharia de Computação

Orientador: Thaís Gaudencio do Rêgo
Coorientador: Yuri de Almeida Malheiros Barbosa

JOÃO PESSOA - PB
2023

LUCAS MOREIRA E SILVA ALVES

Análise da Acurácia de Detecção de Placa de Veículos com Yolov3 em Imagens com Características de Brilho, Resolução e Desfoque Alteradas

Monografia apresentada ao curso Engenharia de Computação do Centro de Informática, da Universidade Federal da Paraíba, como requisito para a obtenção do grau de Bacharel em Engenharia de Computação

João Pessoa, 20 de junho de 2023

BANCA EXAMINADORA

Leonardo Vidal Batista
Universidade Federal da Paraíba

Yuri de Almeida Malheiros Barbosa
Universidade Federal da Paraíba

Thaís Gaudencio do Rêgo
Universidade Federal da Paraíba

Dedico este trabalho à minha família e amigos, cujo amor, apoio e incentivo foram fundamentais para a realização deste projeto.

AGRADECIMENTOS

Gostaria de expressar minha sincera gratidão a todas as pessoas que contribuíram para o sucesso da minha pesquisa e elaboração deste trabalho. Primeiramente, gostaria de agradecer a minha família por todo o apoio, incentivo e recursos que me permitiram seguir meus objetivos acadêmicos. Sem o apoio deles, eu não teria chegado até aqui.

Também quero agradecer aos meus amigos, em especial a Thomas, que trabalhou em conjunto comigo no desenvolvimento deste projeto e talvez continue a pesquisa. Thomas foi um parceiro de trabalho excepcional, sempre demonstrando proatividade, altruísmo e colaboração, além de ser um grande amigo. Sua ajuda foi inestimável e estou muito grato por tê-lo em minha vida.

Por fim, gostaria de agradecer a todos os professores, orientadores e funcionários da universidade que me ajudaram ao longo do caminho. Suas orientações, conhecimentos e insights foram cruciais para o desenvolvimento da minha pesquisa e para o meu crescimento acadêmico. Mais uma vez, gostaria de expressar minha gratidão a todos que me apoiaram durante este processo.

"A verdadeira arte é perigosa. Não é feita para agradar a todos, mas para questionar tudo."
Chuck D

RESUMO

A detecção de placas de veículos usando técnicas de detecção de objetos é de grande importância em diversas aplicações, como segurança, monitoramento de tráfego e gerenciamento de estacionamentos. É uma atividade desafiadora na área visão computacional quando imagens apresentam características de brilho, desfoque e resolução alteradas. Este estudo investigou a influência de variações no brilho, resolução e desfoque na acurácia de detecção utilizando o modelo YOLOv3. Imagens do banco de dados de veículos da Universidade Federal de Ouro Preto (UFOP) foram utilizadas, e filtros foram aplicados para reduzir a qualidade e dificultar a detecção. Gráficos de acurácia média e quantidade de erros foram gerados para identificar os pontos onde o modelo YOLOv3 apresenta baixa acurácia ou falha na detecção. Foi observado que a acurácia do modelo diminui com o valor de 15 como o kernel, 70% de escurecimento em relação à imagem original e 20% da resolução original (800 pixels x 600 pixels). Esses resultados podem ser úteis para o desenvolvimento de estratégias de pré-processamento que melhorem a detecção de placas de veículos em condições de brilho, desfoque e resolução variadas.

Palavras-chave: YOLOv3, Detecção de placas de veículos, Processamento de imagens, Inteligência Artificial

ABSTRACT

Vehicle license plate detection is crucial in various applications such as security, traffic monitoring, and parking management. This study investigates the effect of brightness, resolution, and blur variations on the detection accuracy of the YOLOv3 object detection model. Images from the Universidade Federal de Ouro Preto (UFOP) vehicle database were used, with filters applied to degrade the quality and challenge detection. Average accuracy and error rate graphs were generated to identify points where YOLOv3 model performance declines or fails. It was found that the model's accuracy decreases with a value of 15 as the blur filter kernel, 70% darkening compared to the original image, and 20% of the original resolution (800 pixels x 600 pixels). These findings can aid in developing preprocessing strategies to improve vehicle license plate detection under varying brightness, blur, and resolution conditions.

Keywords: YOLOv3, Plate detection, Digital Image Processing, Artificial Intelligence

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

CSV	Comma-Separated values (do português, valores delimitados por vírgulas)
ID	Sigla para identity, representa um identificador único de uma entidade
PDI	Processamento Digital de Imagens
TCC	Trabalho de Conclusão de Curso
YOLO	You Only Look Once (do português, você só vê uma vez)

SUMÁRIO

1 **INTRODUÇÃO**

A detecção de objetos é uma tarefa fundamental na área da visão computacional e tem aplicações em diversos campos. Ela consiste em identificar e localizar objetos de interesse em imagens ou vídeos, o que pode ser utilizado para tomar decisões ou realizar ações específicas. Por exemplo, a detecção de placas de veículos pode ser utilizada para a segurança de trânsito, enquanto a detecção de objetos perigosos ou ilegais pode ser encontrada em aplicações de vigilância.

O tema supracitado constitui um problema desafiador, visto que os objetos podem se apresentar em diferentes tamanhos, formas e orientações. Ademais, podem ser afetados por condições adversas, como iluminação ruim, desfoque e baixa resolução, fatores que são observados em imagens de trânsito, tendo em vista a qualidade das câmeras, velocidade dos veículos e iluminação nas ruas.

Métodos e algoritmos têm sido desenvolvidos para abordar esse problema, incluindo redes neurais profundas e outras técnicas de aprendizagem de máquina. Tendo em vista a evolução no desenvolvimento de carros autônomos, além do constante monitoramento de trânsito, faz-se necessário a utilização de aplicações que possam detectar objetos de maneira rápida e eficiente, levando em consideração a necessidade de resposta em tempo real para a tomada de decisão nesses problemas.

1.1 DEFINIÇÃO DO PROBLEMA

Este trabalho tem como objetivo avaliar a acurácia da detecção de placas de veículos em imagens com características que dificultam a identificação, utilizando técnicas de aprendizagem profunda. Essa detecção se torna uma tarefa desafiadora quando as imagens apresentam características desfavoráveis, como baixa resolução, baixo contraste e desfoque. A aplicação de técnicas de aprendizagem profunda pode melhorar a acurácia da detecção nessas condições.

Neste trabalho, será realizada uma análise da acurácia da detecção de placas de veículos utilizando técnicas de aprendizagem profunda em imagens com características adversas. Para isso, será utilizada uma base de dados de imagens contendo veículos em diferentes condições de iluminação, quantidade e tipos de veículos em cada imagem. Será avaliado o desempenho do modelo para cada filtro aplicado, com o objetivo de identificar as condições em que a detecção é mais difícil.

1.2 OBJETIVO GERAL

O objetivo é compreender melhor os desafios associados à detecção de objetos em condições de brilho, desfoque e resolução alterados e identificar as condições em que a identificação é mais difícil. A partir dos resultados obtidos, será possível propor soluções para melhorar a acurácia da detecção de placas de veículos nessas condições desfavoráveis, contribuindo para o avanço da área de visão computacional.

1.3 OBJETIVOS ESPECÍFICOS

- Definir o modelo de detecção objetos e os parâmetros que foram utilizados no projeto para realizar a análise dos dados obtidos a partir do dataset de imagens de trânsito
- Avaliar o impacto do aumento do nível de desfoque na imagem na detecção de placas de veículos
- Avaliar o impacto da diminuição dos níveis de brilho na imagem na detecção de placas de veículos
    - Comparar os resultados obtidos na detecção de placas de veículos em imagens com níveis de brilho originais e com brilho equalizado
- Avaliar o impacto da diminuição da resolução da imagem na detecção de placas e veículos

1.4 MOTIVAÇÃO

A detecção de placas de veículos em condições de baixa luminosidade, desfoque e baixa resolução é uma tarefa complexa e desafiadora para sistemas de vigilância e fiscalização de trânsito. Essa dificuldade é amplificada devido à baixa qualidade das câmeras de trânsito e às condições climáticas adversas, como tempo nublado ou chuvoso. Portanto, é importante desenvolver e aprimorar técnicas de detecção de placas de veículos capazes de lidar com essas condições, a fim de garantir a segurança no trânsito e a efetividade dos sistemas de fiscalização.

1.5 ESTRUTURA DO TRABALHO

Este trabalho está dividido em seis capítulos. Na capítulo de Introdução, é apresentado o contexto em que a pesquisa foi desenvolvida, bem como a motivação e os objetivos do estudo.

No Capítulo 2, é apresentada a Fundamentação Teórica que aborda os conceitos

teóricos que foram utilizados no desenvolvimento da pesquisa, incluindo os fundamentos do aprendizado profundo, as arquiteturas de redes neurais convolucionais (do inglês, *Convolutional Neural Networks* - CNN) e a técnica de detecção de objetos utilizada na pesquisa.

No capítulo de Trabalhos Relacionados, serão apresentados estudos semelhantes ao presente trabalho, com o objetivo de contextualizar a pesquisa dentro do estado atual da arte.

O Capítulo 4, Metodologia, trata dos métodos e técnicas utilizados na pesquisa, incluindo a descrição do banco de dados utilizado, os parâmetros da rede neural usados, bem como a descrição dos métodos aplicados para avaliar a detecção de placas de veículos.

No capítulo de Resultados, são apresentados os achados dos experimentos realizados, incluindo a avaliação da acurácia da detecção de placas de veículos em imagens com características de desfoque, resolução e brilho alteradas.

Por fim, o capítulo de Conclusão apresenta uma síntese dos resultados obtidos e discute as implicações da pesquisa, bem como sugestões para trabalhos futuros.

2 **FUNDAMENTAÇÃO TEÓRICA**

A Seção de Fundamentação Teórica apresenta os conceitos e técnicas relacionados à visão computacional e detecção de objetos. A visão computacional é uma área da inteligência artificial que busca desenvolver algoritmos e técnicas para processar e interpretar imagens, com o objetivo de extrair informações contidas nelas (AWARI, 2023).

Já a detecção de objetos é uma área específica da visão computacional, voltada para o desenvolvimento e aplicação de algoritmos capazes de identificar objetos em imagens ou vídeos. Neste trabalho, foi utilizado o algoritmo YOLO (You Only Look Once) (DARKNET), uma rede neural de detecção de objetos eficiente e rápida, capaz de identificar múltiplas classes em imagens e vídeos em tempo real.

2.1 VISÃO COMPUTACIONAL

Com a popularidade das imagens digitais surgiu a necessidade de estudar técnicas para processar imagens, originando um campo de estudo, conhecido como Processamento Digital de Imagens (PDI). O PDI é todo processo de análise e manipulação de imagens digitais, seja para identificar, extrair informações ou transformar uma imagem (QUEIROZ; GOMES, 2006).

A visão computacional engloba um processo complexo que vai além da simples captura da imagem. Ela utiliza modelos de processamento de imagens (PDI) para aprimorar a qualidade da imagem, eliminando ruídos, ajustando o contraste e detectando objetos ou regiões de interesse humano na cena (LOPES, 2019).

2.2 DETECÇÃO DE OBJETOS

Essa pesquisa está inserida em uma área específica da visão computacional, a detecção de objetos, que é composta por duas partes: classificação de imagem e localização de imagem (ZHAO; XU; WU, 2019). Além de proporcionar uma solução para problemas específicos, a visão computacional tem um grande potencial para aplicações futuras, como automação de condução de veículos (DAUMAS, 2020) e vigilância (MOURA; CLARO; GONDIM, 2021).

Pode ser utilizado para localização de objetos em um *frame*, classificação em diferentes categorias e funcionam diante do reconhecimento de objetos através de características individuais (ALVES, 2020).

Tradicionalmente, a detecção de objetos é feita por meio de técnicas de processamento de imagens, como detecção de bordas, segmentação e extração de características.

Essas técnicas dependem fortemente das características da imagem, como resolução, iluminação, contraste, desfoque e cores (PIEMONTEZ, 2022). Com o advento de técnicas de

aprendizado profundo, os computadores podem utilizar dessas técnicas para processar imagens de forma similar aos humanos, reconhecendo padrões complexos em imagens, texto, sons, e outros dados para produzir previsões precisas.

Seus algoritmos contém camadas de neurônios artificiais interconectados que trabalham juntos para aprender e processar informações. Esses neurônios são chamados nós e utilizam cálculos matemáticos para processar os dados.

Redes de aprendizado profundo possuem os seguintes componentes: Camada de entrada, que representa os nós que inserem dados na rede neural. Uma série de camadas ocultas, que transmitem e processam as informações, adaptando o seu comportamento à medida que a rede é atualizada. As redes neurais podem ser de múltiplas camadas e possuir diferentes maneiras de se analisar um problema, onde cada uma delas processa uma característica diferente, visando categorizá-la com precisão. A camada de saída são os nós emissores de dados e representa a resposta do sistema, que pode ser binária ou possuir diferentes tipos de respostas, possuindo mais nós nesse caso (EMMERT-STREIB *et al.*, 2020).

2.2.1 **Introdução sobre o algoritmo YOLO**

YOLO é uma a rede neural de código aberto de detecção de objetos utilizada para a identificação de múltiplas classes em imagens e vídeos em tempo real (REDMON *et al.*, 2016). É um algoritmo eficiente e rápido, possibilitando o uso em dispositivos de baixo poder computacional, como smartphones e tablets (VERMA, 2019).

A arquitetura do modelo é uma Rede Convolucional Profunda. O *frame* é dividido em diversas caixas de detecção, em que cada uma delas é verificada pelo algoritmo para identificar se há um objeto existente nela. Caso um objeto de uma das classes pré-definidas esteja presente nessa caixa, o YOLO atribui a esse uma das classes estabelecidas, as classes variam de acordo com o problema.

Existem diversas versões da YOLO devido à evolução contínua e às contribuições de vários pesquisadores e autores desde o lançamento da versão inicial em 2016. Essas versões incluem YOLO, YOLOv2, YOLOv3, YOLOv4, YOLOv5, YOLOv6, YOLOv7 e YOLOv8 (SHARMA, 2022).

A YOLOv3, lançada em 2018, é a última versão lançada pelo autor original Joseph Redmon. Ela melhorou os modelos anteriores adicionando uma pontuação de "objetividade" à previsão da caixa delimitadora, conectando as camadas da rede *backbone* e fazendo previsões em três níveis de granularidade diferentes para melhorar o desempenho na detecção de objetos menores.

As versões posteriores, como YOLOv4, YOLOv5, YOLOv6 e YOLOv7, foram desenvolvidas por autores diferentes e não são consideradas lançamentos sequenciais estritos

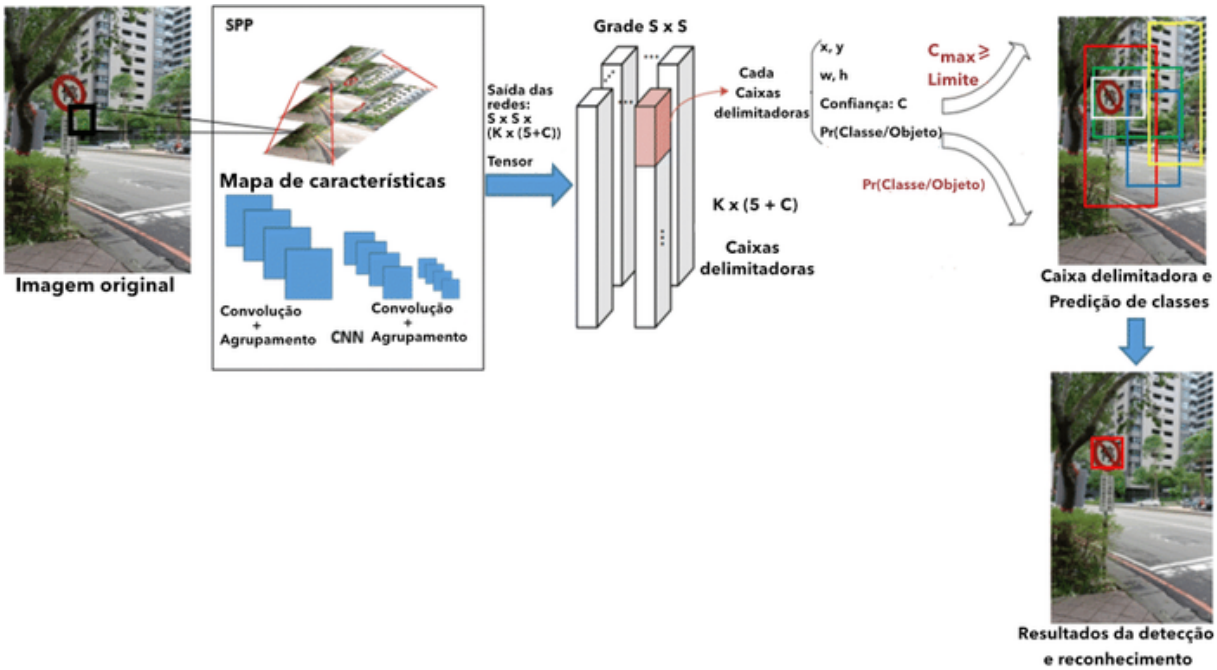
em relação à YOLOv3. Em vez disso, essas versões têm objetivos variados baseados nas metas dos autores que as lançaram, com várias melhorias na arquitetura e nas técnicas utilizadas.

Essas melhorias incluem, mas não estão limitadas a agregação de recursos aprimorada, ativação *mish*, redes *Cross Stage Partial* e outras técnicas que visam aumentar a precisão e diminuir a latência. Essas versões também apresentam diferentes *frameworks*, como PyTorch (no caso da YOLOv5) facilitando o treinamento e uso em produção dos modelos, sendo mais focadas na escalabilidade e na eficiência computacional, tornando-as adequadas para dispositivos com recursos computacionais limitados (NELSON, 2021).

A arquitetura do modelo YOLOv3 funciona da seguinte forma: a imagem de entrada é redimensionada para 448x448 antes de passar pela rede convolucional. Na arquitetura YOLO, várias camadas convolucionais são empregadas, cada uma com dezenas de filtros. A função de ativação utilizada é a ReLU, exceto na camada final, que usa uma função de ativação linear. Algumas técnicas adicionais, como a normalização em lote e a técnica de *dropout* (método de regularização em redes neurais que envolve o desligamento aleatório de neurônios durante o treinamento, ajudando a prevenir o *overfitting* e melhorando a generalização do model).

Na etapa final, o modelo faz previsões em três escalas diferentes, gerando múltiplas *bounding boxes* com diferentes níveis de precisão para capturar objetos de variados tamanhos e aspectos, melhorando assim a detecção de objetos pequenos e grandes na imagem (KEITA, 2022). A Figura 1 representa a arquitetura do modelo YOLO utilizado neste trabalho.

Figura 1 — iImagem da arquitetura do modelo YOLOv3.



Fonte: Adaptado de Dewi, Chen e Yu (2020).

O algoritmo foi inicialmente publicado em 2016 (KUNDU, 2023) e é bastante usado em uma miríade de aplicações de visão computacional, como análise de tráfego de veículos, identificação de animais e vigilância por câmeras.

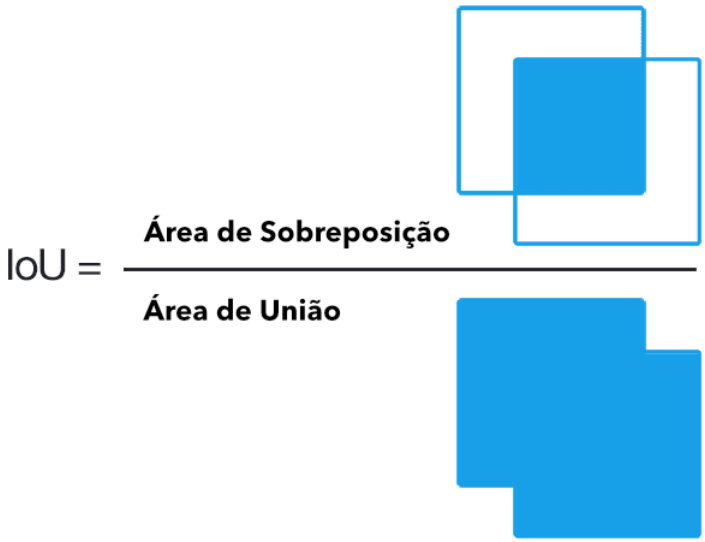
2.2.1.1 Métricas de avaliação

Para avaliar modelos de detecção de objetos com a YOLO, é comum utilizar a métrica de média de precisão (do inglês, *Mean Average Precision* - *mAP*). Para o cálculo da *mAP*, outras submétricas são utilizadas como base. A matriz de confusão é usada para avaliar a detecção das classes do modelo, possuindo 4 tipos diferentes de variáveis:

- **Verdadeiro Positivo (do inglês, *True Positive* - *TP*):** Este termo é utilizado quando o modelo é bem-sucedido em classificar corretamente uma observação como positiva. São as instâncias da classe que estamos interessados que foram corretamente identificadas pelo modelo.
- **Verdadeiro Negativo (do inglês, *True Negative* - *TN*):** Isso se refere às instâncias que foram corretamente identificadas pelo modelo como negativas e eram negativas, Ou seja, o modelo fez uma classificação acertada.
- **Falso Positivo (do inglês, *False Positive* - *FP*):** Este termo descreve as situações em que o modelo previu um resultado positivo para observações que na realidade eram negativas. Em outras palavras, o modelo fez uma estimativa equivocada da classe que estávamos tentando prever.
- **Falso Negativo (do inglês, *False Negative* - *FN*):** Esta é a situação em que o modelo categorizou uma observação como negativa, quando na verdade era positiva. Sendo assim, o modelo estimou incorretamente.

A Intersecção sobre União (do inglês, *Intersection over Union* - *IoU*) é utilizada para verificar áreas em que há uma sobreposição das coordenadas das *bounding boxes* previstas para a *bounding box* padrão-ouro (representando a área correta do objeto a ser detectado). A Fórmula 1 define como calcular a IoU:

Figura 2 — Representação esquemática da fórmula para calcular o IoU.



Fonte: Adaptado de Rosebrock (2017).

A precisão mede a eficiência do modelo em encontrar *TPs* dentre todas as previsões positivas, a precisão é calculada por:

$$Precisão = \frac{TP}{TP + FP} \tag{1}$$

Para o cálculo da precisão é necessário um valor limite para a IoU, por exemplo, se o limite da IoU for 0,5 (limiar de Intersecção sobre União), valores da IoU abaixo desse limite (área de sobreposição divida pela área de união menores que 0,5) isso será considerado um *FP*, caso seja superior a esse valor é um *TP*.

Revocação analisa a capacidade do modelo em encontrar *TPs* dentre os *TPs* e *FNs*, a revocação é calculada por:

$$Revocação = \frac{TP}{TP + FN} \tag{2}$$

Para efetuar o cálculo da Precisão média (do inglês, *Average Precision* - *AP*), primeiro, deve-se traçar a curva de precisão-revocação (*Precision-Recall curve*) mudando o limiar de confiança do modelo e calcular as métricas de precisão e revocação em cada limiar. Posteriormente, deve calcular a área sob a curva de precisão-revocação, que representa a média ponderada de precisões em cada limiar. O peso é dado pelo aumento na revocação do

limiar anterior (SHAH, 2022). A AP é calculada através da seguinte Fórmula (em que *n* representa a quantidade de limiares de confiança e *k* é usado para se referir a um limiar específico):

$$AP = \sum_{k=0}^{k=n-1} [Revocação(k) - Revocação(k+1)] * Precisão(k) \tag{3}$$

Por fim, a mAP é calculada pela Fórmula 4 (em que *APk* representa a precisão média para uma classe *k*), Soma-se todas as *APs* calculadas previamente e dividindo-as pelo número total de classes:

$$mAP = \frac{1}{N} \sum_{k=0}^{N} APk \tag{4}$$

Essa métrica é muito importante para a avaliação e comparação de diferentes modelos de detecção de objetos (GAD, 2021). Ela é usada para a avaliação dos modelos YOLO, como é apresentado no Capítulo 3. O projeto atual utilizou os níveis de confiança (métrica que determina a probabilidade de uma *bounding box* conter um objeto de interesse), fazendo a média desses níveis de confiança da detecção de placas de veículos entre as imagens testadas.

2.2.2 **DEFINIÇÃO DAS CARACTERÍSTICAS AVALIADAS**

Neste trabalho as características de brilho, resolução e desfoque foram alteradas para verificar a eficácia do modelo na detecção de imagens com essas mudanças. Segue a definição dos métodos utilizados e como estes alteram a matriz de pixels da imagem:

- **Resolução:** Ao ajustar a resolução de uma imagem, estamos alterando o número de pixels presentes na imagem. Reduzir a resolução implica diminuir a quantidade de pixels na largura e altura da imagem, enquanto aumentar a resolução acrescenta mais pixels. Essa alteração afeta diretamente a matriz de pixels, pois a quantidade de pixels em cada dimensão é modificada (KHANAL, 2020).
- **Brilho:** Alterar o brilho de uma imagem afeta a intensidade de luz de cada pixel na matriz de pixels. Aumentar o brilho da imagem significa aumentar a intensidade de luz dos pixels, enquanto diminuir o brilho reduz a intensidade de luz dos pixels. Portanto, a matriz de pixels é afetada pela mudança no valor de intensidade de cada pixel individual (ARAR, 2019).
- **Desfoque:** Ao aplicar desfoque a uma imagem, estamos suavizando as bordas e detalhes na imagem, o que afeta a matriz de pixels ao misturar os valores dos pixels

> adjacentes. Quando um desfoque é aplicado, os pixels próximos compartilham informações de cor e intensidade entre si, resultando em uma suavização dos detalhes. Esse efeito altera a matriz de pixels, pois os valores dos pixels são modificados com base nas informações dos pixels vizinhos, diminuindo a nitidez da imagem, deixando-a com um aspecto mais borrado (NAUFAL, 2020).

Essas mudanças na matriz de pixels causadas pelos ajustes de resolução, brilho e desfoque influenciam diretamente o desempenho do método de detecção YOLOv3, uma vez que o algoritmo depende das informações presentes nos pixels da imagem para identificar e localizar objetos.

3 **TRABALHOS RELACIONADOS**

Diversos trabalhos têm sido realizados com o objetivo de investigar e melhorar o desempenho dos modelos de detecção de objetos em condições de resolução, brilho e desfoque variadas. Esses trabalhos são importantes para aprimorar os modelos existentes, desenvolver novas abordagens e fornecer diretrizes para futuras pesquisas na área.

Neste capítulo, serão apresentados trabalhos relacionados que abordam a detecção de objetos em imagens com características variadas, destacando os avanços e desafios enfrentados pelos pesquisadores e profissionais da área. Além disso, será fornecido um panorama geral das técnicas e metodologias utilizadas para lidar com esses desafios, bem como os resultados obtidos em diferentes cenários e aplicações. Dessa forma, será possível compreender o estado da arte na detecção de objetos em condições adversas e identificar possíveis oportunidades para futuras pesquisas e desenvolvimento de novas soluções.

Um estudo relevante para o projeto atual foi realizado por pesquisadores da Universidade de Stanford (JIANG; WANG, 2016), que investigaram a robustez de algoritmos de detecção de objetos em imagens de baixa qualidade, comumente encontradas em contextos de webcam, como câmeras de robôs. O estudo destacou a importância de desenvolver algoritmos que possam tolerar imagens de baixa qualidade e realizar detecções em tempo real. No projeto em questão, foram testados diferentes tipos de desfoque (gaussiano e de movimento) e resoluções, bem como a aplicação de um filtro de pixelização nas imagens, utilizando o mesmo conjunto de tecnologias *Python* e *OpenCV*. Tal abordagem é diferente do método utilizado no presente trabalho, que usou o desfoque de caixa e uma redução da resolução da imagem sem pixelização.

Os resultados desse estudo apontaram para uma diminuição da precisão dos modelos de detecção de objetos em imagens com características alteradas, mas também mostraram melhorias, quando os modelos foram re-treinados com imagens de baixa qualidade específicas para cada categoria de degradação. No entanto, a precisão em imagens originais diminuiu com o modelo re-treinado. Esses achados indicam que a utilização de técnicas de pré-processamento de imagem e re-treinamento de modelos pode ser útil para melhorar a detecção de objetos em condições adversas, embora seja necessário um equilíbrio entre a precisão em imagens degradadas e a de imagens originais. Essas conclusões são relevantes para o presente projeto, pois utiliza técnicas semelhantes que visam aprimorar a detecção de placas e veículos em imagens com características degradadas.

O artigo de Xu, Li e Shi (2022) aborda a detecção de navios em imagens de satélite de baixa resolução. Os autores identificaram que os modelos de detecção de navios baseados em redes neurais convolucionais atuais enfrentam dificuldades em imagens de baixa resolução, principalmente devido à perda de informações importantes sobre os navios e à variação do contraste entre objeto e plano de fundo.

Para resolver esses problemas, os autores propuseram o modelo LMO-YOLO, construído a partir do YOLOv4. Eles desenvolveram um esquema de redimensionamento linear múltiplo para quantizar as imagens de satélite originais em imagens de 8 bits. Além disso, convoluções dilatadas foram incluídas na rede YOLOv4 para extrair características dos objetos e contrastes do objeto, em relação ao plano de fundo. Por último, um esquema de ponderação adaptativa foi projetado para equilibrar a perda entre navios fáceis e difíceis de detectar.

Os resultados experimentais mostraram que o LMO-YOLO superou outros métodos de ponta. No entanto, alguns alvos cobertos por nuvens espessas não puderam ser detectados, e alguns ruídos do mar ou nuvens fragmentadas foram identificados como falsos alarmes.

Comparando com o trabalho atual, ambos lidam com a detecção de objetos em imagens com características de resolução baixa e utilizam o modelo YOLO (apesar do presente projeto utilizar o YOLOv3). No entanto, o artigo foca na detecção de navios em imagens de satélite de baixa resolução, enquanto este trabalho se concentra na detecção de placas de veículos. O LMO-YOLO utiliza técnicas diferentes para melhorar a detecção, como o esquema de redimensionamento linear múltiplo e convoluções dilatadas.

Um estudo relacionado propôs o algoritmo RA-RetinexNet para melhorar o YOLOv4 na detecção de porcos em condições de pouca luz e com névoa (YIN, 2022). Os autores treinaram o modelo YOLOv4 com imagens de baixa iluminação, iluminação normal e névoa com baixa iluminação. Posteriormente, utilizaram a rede RetinexNet com conexões residuais e mecanismo de atenção para aprimorar a qualidade das imagens e aumentar a precisão da detecção dos alvos.

Embora este estudo esteja focado na detecção de porcos em um ambiente de fazenda, a abordagem RA-RetinexNet poderia ser aplicada como uma etapa de pré-processamento no trabalho atual para melhorar a qualidade das imagens em condições de pouca luz e desfoque antes de aplicar o YOLOv3 na detecção de placas de veículos. Imagens desfocadas poderiam potencialmente ser beneficiadas pois o artigo também investigou imagens com névoa, obtendo resultados de precisão melhores que o modelo original.

O artigo "Deblur-YOLO: Real-Time Object Detection with Efficient Blind Motion Deblurring" aborda a detecção de objetos em imagens desfocadas (ZHENG *et al.*, 2021). Os autores identificaram que os algoritmos de detecção de objetos de última geração falham ao operar em imagens desfocadas. Para superar essa limitação, eles propuseram o Deblur-YOLO, uma abordagem eficiente e robusta baseada no YOLOv3 para lidar com imagens borradas causadas por movimento.

O Deblur-YOLO utiliza uma rede adversarial generativa com um gerador de pirâmide de recursos dilatados e um par de discriminadores multi-escala com normalização espectral, além de um discriminador de detecção. Os autores também desenvolveram uma nova métrica de qualidade de imagem chamada *Smooth Peak Signal-to-Noise Ratio* (SPSNR) para medir a

suavidade da imagem reconstruída.

Comparando com o trabalho atual, ambos lidam com a detecção de objetos em imagens com características de desfoque alteradas e utilizam o modelo YOLOv3. No entanto, o artigo se concentra em imagens com desfoque de movimento, enquanto o trabalho atual investiga a influência de variações de desfoque de caixa na detecção de placas de veículos. O Deblur-YOLO emprega técnicas específicas, como a rede adversarial generativa e o gerador de pirâmide de recursos dilatados, que podem ser consideradas como inspiração para melhorar a detecção de placas de veículos em condições adversas de desfoque.

No artigo Bhandari *et al.* (2020) , os autores abordam o desafio da detecção e reconhecimento de objetos em ambientes de baixa luminosidade, típicos em sistemas de vigilância noturna. Eles utilizam imagens capturadas com uma câmera infravermelha em condições de baixa luz e aplicam diferentes algoritmos de aprimoramento de imagem baseados no domínio espacial. Essas imagens aprimoradas são então enviadas para um processo de classificação usando uma rede neural convolucional seguida por uma camada totalmente conectada de neurônios.

O estudo compara a precisão da classificação após a implementação de diferentes algoritmos de aprimoramento de imagem: sem aprimoramento (58%), equalização de histograma (71%), equalização adaptativa de histograma (74%) e equalização adaptativa com limitação de contraste de histograma (85%). A pesquisa conclui que as técnicas de aprimoramento de imagem podem ser usadas para melhorar a precisão da classificação produzida por uma CNN especializada desenvolvida usando a técnica de transferência de aprendizado.

Embora esse trabalho também explore a detecção de objetos em imagens degradadas, ele se concentra especificamente em ambientes de baixa luminosidade e utiliza câmeras infravermelhas. O artigo utiliza diferentes algoritmos de aprimoramento de imagem e uma abordagem de transferência de aprendizado para treinar a CNN, enquanto o trabalho atual utiliza o modelo YOLOv3. Ainda assim, ambos os trabalhos compartilham o objetivo comum de análise de acurácia em ambientes de baixa iluminação, além da tentativa de utilizar técnicas para melhorar a acurácia. No trabalho atual a técnica equalização de histograma foi utilizada.

O Quadro 1 possui a descrição do nome, objetivo, metodologia, métricas e resultados dos trabalhos supracitados:

Quadro 1 — Descrição dos artigos relacionados. (continua)

Artigo	**Objetivo**	**Metodologia**	**Métricas e resultados**
"Object Detection and Counting with Low	Investigar a robustez de algoritmos de detecção	Testes com desfoque gaussiano e de	Os resultados indicam que o modelo re-treinado

Quadro 1 — Descrição dos artigos relacionados. (continuação)

Artigo	Objetivo	Metodologia	Métricas e resultados
Quality Videos", Jiang; Wang, 2016.	de objetos em imagens de baixa qualidade, como em webcams e câmeras de robôs.	movimento, variação de resolução e filtro de pixelização usando Python, OpenCV, YOLOv3-tiny, re-treinando modelos com imagens de baixa qualidade.	conta objetos com mais eficácia em diversos vídeos, avaliado através de métricas como mAP, acurácia de classificação e IOU acima de 50%.
"LMO-YOLO: A Ship Detection Model for Low-Resolution Optical Satellite Imagery", Xu, Li e Shi, 2022.	Detecção de navios em imagens de satélite de baixa resolução, melhorando o desempenho de modelos baseados em CNNs.	Modelo LMO-YOLO construído a partir do YOLOv4, redimensionamento linear múltiplo para quantização de imagens, e convoluções dilatadas para extração de características.	O LMO-YOLO, método proposto, ultrapassa alternativas recentes em detecção, alcançando 92,32% de precisão média no conjunto WFV e 93,07% no PMS, além de exibir altas taxas de revocação e precisão, com eficácia em métricas de sensoriamento remoto.
"An Improved Algorithm for Target Detection in Low Light Conditions", Yin, 2022.	Melhorar a detecção de porcos em condições de pouca luz e névoa, aprimorando a qualidade das imagens e a precisão da detecção dos alvos.	Algoritmo RA-RetinexNet aplicado ao modelo YOLOv4, treinamento com imagens de baixa iluminação e névoa com baixa iluminação, rede RetinexNet com conexões residuais e mecanismo de atenção.	A integração da RA-RetinexNet com o YOLO v4 resultou em melhorias significativas no desempenho da detecção de objetos em imagens com pouca luz, incluindo um aumento de 5,4% no recall e 2,04% na precisão. Além disso, para imagens com baixa luminosidade e nevoeiro, houve um aumento de 4,84% na precisão, 2,38% no recall, e a precisão foi 3,85% maior após a otimização.
"Deblur-YOLO: Real-Time Object Detection with Efficient Blind Motion Deblurring",	Detecção de objetos em imagens desfocadas causadas por movimento,	Deblur-YOLO baseado no YOLOv3, rede adversarial generativa com um gerador de pirâmide de	Uma métrica de qualidade de imagem equilibrada chamada Smooth Peak

Quadro 1 — Descrição dos artigos relacionados. (conclusão)

Artigo	Objetivo	Metodologia	Métricas e resultados
Zheng *et. al*, 2021.	desenvolvendo uma abordagem eficiente e robusta baseada no YOLOv3.	recursos dilatados e discriminadores multi-escala, e discriminador de detecção.	Signal-to-Noise Ratio (SPSNR) foi proposta e utilizada para avaliar o desempenho do Deblur-YOLO. Este modelo apresentou excelentes resultados e melhor qualidade visual nos conjuntos de dados Set 5, Set 14 e no COCO 2014
"Image Enhancement and Object Recognition for Night Vision Surveillance", Bhandari *et al*, 2020	Detecção e reconhecimento de objetos em ambientes de baixa luminosidade, típicos em sistemas de vigilância noturna, utilizando câmeras infravermelhas.	Aprimoramento de imagem com diferentes algoritmos e classificação usando uma rede neural convolucional (CNN) seguida por uma camada totalmente conectada de neurônios, treinamento da CNN com transferência de aprendizado	As técnicas de equalização de histograma HE, AHE e CLAHE foram avaliadas com base em entropia, MSE, PSNR e PSNR modificado com variância. A técnica CLAHE superou as outras, alcançando 85% de precisão na classificação, em comparação com 74% para AHE, 71% para HE e 58% para imagens sem aprimoramento.

Fonte: O autor (2023).

Em conclusão, os trabalhos relacionados apresentados nesta seção exploram diferentes abordagens e técnicas para lidar com imagens com níveis de resolução, desfoque e brilho alterados. Proporcionando lições valiosas para o desenvolvimento e aprimoramento de modelos de detecção de objetos em condições adversas. Esses estudos demonstram a importância de adaptar e melhorar os modelos existentes, como o YOLO, para enfrentar os desafios específicos que surgem em cenários reais. Ao revisar esses trabalhos e aprender com suas abordagens e técnicas, podemos identificar oportunidades para aprimorar ainda mais o desempenho do modelo YOLOv3 na detecção de placas de veículos e, assim, contribuir para a eficácia e confiabilidade dos sistemas de detecção em ambientes de trânsito desafiadores.

4 **METODOLOGIA**

O trabalho possui uma metodologia exploratória, em que o tema de teste de imagens com características alteradas foi pesquisado em conjunto com a análise do fenômeno da pesquisa, gerando como resultados gráficos e observações sobre o temo oriundas do próprio trabalho.

Por se tratar de uma pesquisa que lida com dados exatos, gerando resultados numéricos e gráficos para cada tipo de degradação da imagem (desfoque, resolução e brilho), possui abordagem quantitativa.

Busca-se a partir dos gráficos detectar pontos de ruptura na detecção do modelo, porcentagens dos atributos das imagens em que o veículo não é mais identificado, além de traçar graficamente as mudanças de acurácia do YOLOv3, ao alterar as características das imagens. A partir disto será possível avaliar a efetividade do modelo na detecção de veículos e identificar possíveis limitações ou pontos a serem melhorados.

4.1 BANCOS DE DADOS

A escolha dos dados é um dos fatores críticos para o sucesso de um modelo de detecção de objetos eficaz. Para tanto foram utilizados 2 bancos de dados para o trabalho.

Os dados relacionados aos veículos da UFOP em conjunto com o aplicativo *Label Img* foram utilizados para o treinamento de detecção de placas de veículos com o modelo YOLOv3. Mais detalhes serão apresentado na Subseção 4.1.1.

As informações do banco de dados COCO são utilizadas como base para o treinamento do modelo YOLOv3 (MANTRIPRAGADA, 2020) para a detecção de veículos (carros, motos, ônibus, caminhões). Diferentes pesos e configurações foram utilizadas para os modelos YOLOv3 no projeto.

O conjunto de dados utilizado no desenvolvimento dessa pesquisa, é disponibilizado pela Universidade Federal de Ouro Preto (UFOP). Este banco de dados possui imagens de veículos com proporções 800x600 *pixels*, em diferentes horários matutinos e vespertinos. Possui um total de 376 imagens fotografadas no campus da UFOP (MENDES JÚNIOR *et al.*, 2020).

É um conjunto de dados de imagens para treinar e testar algoritmos de detecção de objetos, como o YOLOv3. Ele contém um conjunto de dados de imagens para treinamento e teste de algoritmos de visão computacional, com cerca de 330 mil imagens de treinamento e validação e mais de 200 mil imagens de teste.

As imagens incluem uma grande variedade de objetos, cenas e fundos complexos, além de anotações para 80 categorias de objetos, como pessoas, animais, veículos, móveis, entre outros. O banco de dados COCO também fornece anotações de segmentação para objetos individuais em cada imagem. O tamanho total dos dados é de aproximadamente 25GB

(STERLING, 2020). O conjunto de dados é amplamente utilizado para tarefas de detecção de objetos, segmentação semântica e outros desafios de visão computacional. O modelo YOLOv3 usa a base de dados COCO para o treinamento e validação de seu algoritmo de detecção de objetos.

Além disso, a base de dados fornece informações de rótulos para cada objeto nas imagens, incluindo suas coordenadas, tamanho e classe. Isso permite que os algoritmos possam ser avaliados e comparados com maior precisão (LIN *et al.*, 2014).

4.2 FERRAMENTAS

LabelImg foi utilizado para fazer as anotações de coordenadas das placas dos veículos nas imagens do banco de dados da UFOP. A aplicação é uma ferramenta gráfica de anotação de imagens, rotulando diferentes classes personalizáveis, possibilitando ao usuário desenhar caixas delimitadoras em torno dos objetos de interesse de cada imagem (HEARTEX, 2022).

OpenCV foi usada para a edição de brilho, resolução e desfoque, apresentação dos resultados de acurácia, utilização da rede neural, além da leitura e exposição de arquivos de imagem. É uma biblioteca de computação visual de código aberto, que fornece uma série de algoritmos para processamento de imagens e vídeos (BOESCH, 2023). Essa ferramenta suporta uma variedade de linguagens de programação, incluindo Python 3.9.12, que foi utilizado nessa pesquisa.

A biblioteca possui uma ampla gama de funcionalidades, tais como detecção de rostos, reconhecimento e rastreamento de objetos, reconstrução 3D, entre outras. É bastante utilizada em aplicações de visão computacional, como robótica, inteligência artificial, segurança e vigilância e análise de dados.

Numpy foi utilizado para realizar operações com *arrays* no projeto, como encontrar as placas de veículos que possuem maior acurácia, transformação da imagem para *array* (formato reconhecido pela biblioteca *opencv*). Além de ser usado como alicerce para a ferramenta *Panda*s.

É uma biblioteca de computação científica que oferece suporte a *arrays* multi-dimensionais, possibilitando a realização de operações matemáticas rápidas e eficientes sobre essas estruturas.

Os *arrays* dessa ferramenta permitem o armazenamento e manipulação de uma grande quantidade de dados numéricos, com diferentes funções matemáticas disponibilizadas como métodos dos conjuntos de dados, relacionados a trigonometria, aritmética e estatística.

Pandas foi utilizado para agregar dados em *DataFrames*, possibilitando a geração de arquivos *csv* com informações relativas a precisão de detecção das placas dos veículos, além de agregar dados de erros. A partir dessas informações, o agrupamento desses dados, extração da média e geração de gráficos também foi feito utilizando *pandas*.

4.3 UTILIZAÇÃO DO YOLO NA PESQUISA

A rede neural YOLO foi aplicada em duas etapas do trabalho, *a priori* para a detecção de carros, motos, ônibus e caminhões. As imagens utilizadas possuem classes pré-definidas, reconhecidas pelo algoritmo através do uso de pesos, modelo e configurações do YOLOv3-spp (DARKNET) treinado com o banco de dados *coco*. Cada veículo corretamente detectado pelo modelo tem sua *bounding box* delimitada (coordenadas representando a localização da classe localizada na imagem).

Uma rede neural YOLO é usada para a detecção de veículos na imagem e outra para detectar placas dos veículos. A rede de detecção de placas é utilizada após a *bounding box* delimitada. Apenas essa região é usada para a extração das placas. O banco de dados de veículos foi utilizado visando obter a área da placa do veículo em cada imagem. *A posteriori*, as imagens do banco de dados foram aplicadas no software. A cada imagem anotada, novos arquivos com coordenadas da caixa delimitadora são gerados, com isso, é possível treinar o modelo YOLO utilizando esses dados.

4.3.1 **Parâmetros do YOLO**

Os parâmetros do YOLOv3 variam dependendo da implementação do algoritmo, mas existem alguns parâmetros comuns, entre eles:

- **classes:** O número de classes de objetos que o YOLO está treinado para detectar.
- **width** e **height:** O tamanho da imagem de entrada, que geralmente é redimensionado para um tamanho fixo.
- **mask:** O número de *bounding boxes* que são atribuídas a cada célula da grade.
- **anchor:** As dimensões das caixas delimitadoras pré-definidas (*anchors*) que o YOLO usa para detectar objetos de diferentes formas e tamanhos.
- **jitter:** O grau de aleatoriedade aplicado ao redimensionamento e recorte das imagens durante o treinamento, para aumentar a robustez do modelo.
- **Filters:** Quantidade de filtros utilizados pelas camadas convolucionais do modelo, utilizados para extrair características mais importantes das imagens.
- **subdivisions:** Se refere ao número de vezes que a imagem é dividida em sub-regiões para detectar objetos.
- **max_batches:** Número máximo de iterações que o modelo irá executar.

Esses parâmetros podem ser ajustados para otimizar o desempenho do YOLO em diferentes aplicações e cenários. Os parâmetros de configuração utilizados no modelo do YOLO para a detecção de veículos foram obtidos pelo repositório oficial, sem alterações. Para a detecção de placas os valores de

batches, *subdivisions*, *max_batches*, *filters* e *classes* foram modificados.

4.3.1.1 Parâmetros para a detecção de veículos

Na detecção de veículos utilizamos o banco de dados COCO mencionado na Seção 4.1.2, usando as configurações e parâmetros padrão da versão YOLOv3-spp, obtidos a partir do site oficial do *darknet* (DARKNET). Os pesos do modelo pré-treinado foram utilizados.

4.3.1.2 Parâmetros para a detecção de placas

A detecção de placas de veículos foi feita utilizando o arquivo de configuração do *yolov3.cfg* do repositório oficial da *Darknet* (ALEXEYAB). Foi utilizado o banco de dados de veículos da UFOP mencionado na Seção 4.1.1. O *software LabelImg*, citado na Seção 4.2 foi usado para extrair as regiões de placas de veículos que serão usados no treinamento da rede neural. As informações de coordenadas verticais e horizontais de cada imagem são salvas em um arquivo de texto que será utilizado pelo modelo no momento de treinamento.

O treinamento do modelo foi feito com a plataforma *Google Colab* (GOOGLE COLAB) e os parâmetros do YOLOv3 foram alterados e detalhados em seguida. As alterações de parâmetros foram feitas com base no artigo de Radečić (2020). Não foram testados outros valores por limitações de poder de processamento e disponibilidade de recursos para contas gratuitas. As seguintes mudanças de parâmetros foram realizados no arquivo *Makefile*:

GPU: Mudanças nessa configuração determinam se o modelo vai rodar utilizando uma Unidade de Processamento Gráfico (*GPU*) ou não. Neste trabalho foi utilizado o valor 1, denotando que o modelo vai rodar executar com a *GPU*, tornando o processo mais veloz comparado a sem *GPU (CHARAN, 2021)*.

CUDNN: Corresponde a configuração de CUDA Deep Neural Network (cuDNN), que utiliza implementações otimizadas de operações de aprendizado profundo para *GPU*'s da NVIDIA. Neste trabalho foi utilizado o valor 1, aumentando a velocidade do treinamento da rede neural.

OPENCV: Denota se o modelo vai utilizar a biblioteca *Opencv*, citada na Seção 4.3. Neste trabalho foi utilizado o valor 1, fazendo com que o modelo YOLOv3 use recursos de processamento de imagem da biblioteca *Opencv*, melhorando a precisão e eficiência da detecção de objetos.

Ademais, alterações no arquivo de configuração da rede neural (*yolov3.cfg*) também foram realizadas:

batches: Neste trabalho foi usado o valor 64.

subdivisions: Neste trabalho foi usado o valor 64, permitindo que o modelo detecte objetos em uma área maior da imagem.

max_batches: O valor de 2000 foi utilizado, tornando o treinamento mais rápido, o que pode afetar a precisão do modelo, pois quanto menor o número de iterações, menor a chance do modelo aprender características mais complexas da imagem. Dadas as limitações de *hardware* para executar o modelo, esse valor foi estabelecido.

filters: Neste trabalho foi utilizado o valor 18, diminuindo-o por causa de limitações de *hardware*.

classes: Diminuiu-se de 80 para 1, pois, a intenção é detectar apenas a placa do veículo em questão.

4.4 FLUXO DA APLICAÇÃO

4.4.1 **Seleção de imagens do dataset**

A priori são selecionadas imagens do banco de dados da UFOP (Seção 4.1.1) que serão utilizadas para analisar resultados relativos a acurácia do modelo na detecção de placas de veículos. Foi realizada a seleção de 300 imagens do banco de dados, de maneira aleatória utilizando o módulo *random* da linguagem Python com o método *choices (DOCS.PYTHON)*. Foram utilizadas menos imagens que o total do banco pois a extração dos dados é mais rápida e mais teste puderam ser realizados. Uma função foi criada para realizar os tratamentos de brilho, resolução ou desfoque. Esta função recebe uma lista com todas as imagens aleatoriamente selecionadas e o tipo de filtro que deve ser aplicado nas imagens, após a aplicação do filtro, cada imagem passa pelo processo de detecção de placas e veículos.

4.4.2 **Filtros aplicados às imagens**

Os filtros e níveis percentuais ou de intensidade aplicados nas imagens estão listados na Tabela 2:

Tabela 1 — Filtros e níveis percentuais ou de intensidade aplicados. 5% para brilho representa 0,05 de gamma aplicado no cálculo de correção de gamma. 3% representa uma diminuição nas dimensões da imagem para 3% de largura e altura originais e 5 de desfoque denota o tamanho do kernel

Filtro	**Níveis percentuais/intensidade aplicados**
Brilho	5%, 10%, 15%, 22%, 25%, 28%, 30%, 33%, 35%, 40%, 50%, 60%, 70%, 80%, 90%
Desfoque	5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 65, 75, 85, 95
Resolução	3%, 5%, 7%, 10%, 20%, 25%, 35%, 45%, 60%, 75%, 90%, 95%

Fonte: O autor (2023).

Os filtros foram testados em cada imagem selecionada (cada filtro foi aplicado de maneira isolada). Os níveis e passos foram escolhidos através de testes, diminuindo o intervalo quando haviam quedas mais acentuadas de acurácia para identificar o valor do filtro em que o modelo deixa de ser eficaz (com baixa acurácia na detecção das placas ou apresentando erros ao detectá-las), sendo possível observar o comportamento do modelo em diferentes cenários.

4.4.2.1 Resolução

O filtro de resolução de imagem é aplicado da seguinte forma: A largura da nova imagem é calculada como a largura da imagem original multiplicada pela porcentagem de escala passada na função dividido por 100. A altura da nova imagem é calculada da mesma maneira.

Após a obtenção de novos dados de altura e largura da imagem, foi utilizada a função *resize* da biblioteca *opencv* para efetuar o redimensionamento, utilizando o método de interpolação de área *cv2.INTER_AREA* (CHADRICK, 2018). O método realiza a reamostragem baseado na área dos *pixels*. É preferível em métodos de diminuição de resolução por preservar detalhes da imagem original.

O banco de dados apresenta esses dados com as mesmas dimensões, 800 de largura por 600 de altura, sendo assim, já apresenta essas características equalizadas nas imagens originais. Pode-se observar o filtro de resolução aplicado na Figura 3.

Figura 3 — Imagem original ao lado da mesma com a aplicação do filtro de 20% da resolução original.

Fonte: O autor (2023).

4.4.2.2 Desfoque

Para efetuar o filtro de desfoque na imagem, o método *blur* do *opencv* foi utilizado. O método criado recebe uma intensidade de desfoque que por fim, será aplicado a imagem original.

A intensidade de desfoque define o tamanho do *kernel* que será utilizado para realizar o filtro na imagem. Quanto maior o *kernel*, mais borrada a imagem filtrada (ROSEBROCK, 2021). Pode-se observar o filtro de desfoque aplicado na Figura 4.

Figura 4 — Imagem original ao lado da mesma com aplicação de filtro de 20 pixels de tamanho do kernel de desfoque.

Fonte: O autor (2023).

4.4.2.3 Brilho

Ademais, filtros também foram aplicados para o brilho da imagem original visando escurecê-la. Para tanto, um nível de *gamma* é utilizado como parâmetro e determinará o nível de escurecimento da imagem. Quanto menor o valor de *gamma*, mais escura será a imagem

A técnica de correção de *gamma* é utilizada para alterar a luminosidade original da imagem, por meio do cálculo do inverso do fator *gamma* e da criação de uma tabela de conversão de cores. A partir dessa tabela, é possível aplicar a correção de *gamma* à imagem original utilizando a função LUT do *OpenCV* (JOSHI; ESCRIVA; GODOY, 2016). Pode-se observar o filtro de brilho aplicado na Figura 5.

Figura 5 — Imagem original ao lado da mesma com aplicação de filtro de 0,4 como denominador do inverso da gamma da imagem original, gerando um valor de 2,5 de gamma.

Fonte: O autor (2023).

4.4.2.4 Equalização de brilho

Outrossim, dado que a luminosidade das imagens do banco de dados não é homogênea, testes com níveis de luminância uniformizados foram realizados. Para esse propósito, o algoritmo, detalhado a seguir, foi implementado.

A princípio, o cálculo do histograma (distribuição de frequências de níveis de escala de cinza) da imagem em preto e branco é calculado usando os métodos *convert* e *histogram* da biblioteca *Pillow*. Os valores obtidos do histograma são percorridos (a tabela criada possui o tamanho 256 e representa a quantidade total de *pixels* para cada faixa nível de cinza) calculando um passo da equalização em cada etapa. Em seguida, uma lista de valores de cinza uniformizados foi preenchida. A lista criada é então aplicada a imagem original utilizando a função *point* da biblioteca *Pillow* efetivando a equalização da imagem.

A técnica utilizada é chamada equalização de histograma e é feita para realizar ajustes no contraste e na luminosidade de uma imagem, distribuindo os níveis de cinza de maneira mais uniforme ao longo do histograma da imagem. Isso pode ajudar a melhorar a visualização de detalhes na imagem e aumentar o seu contraste. Pode-se observar o efeito de equalização aplicado na Figura 6.

Figura 6 — Imagem original ao lado da mesma com filtro de equalização aplicado.

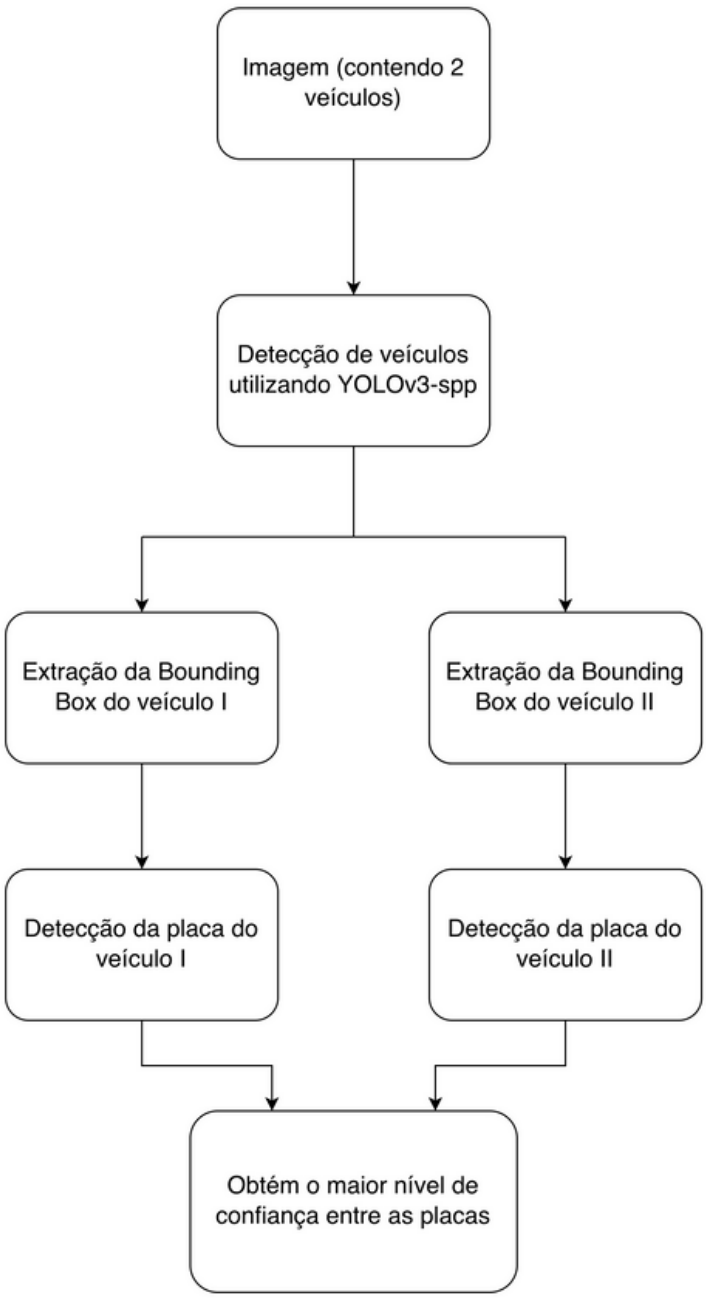
Fonte: O autor (2023).

4.4.3 **Detecção de veículos e placas na imagem**

Tendo em vista que mais de um veículo e placa de veículo pode ser identificado na imagem, foi necessário um tratamento para esses casos, em que suas *bounding boxes* e placas são detectadas. O valor de precisão utilizado para os resultados é o da placa com maior nível de confiança.

Essa escolha foi feita pois imagens com veículos distantes da câmera poderiam distorcer o nível de confiança do veículo mais próximo, caso a média fosse calculada. Utilizando o valor mais alto de confiança de detecção, o veículo mais próximo da câmera será utilizado para a obtenção dos resultados, que, no caso do banco de dados da UFOP, representa uma distância similar a de imagens com apenas 1 veículo. Um fluxograma representando como a aplicação se comporta em casos de imagens contendo 2 veículos pode ser observado no Fluxograma 1. Fluxograma da aplicação para uma imagem com 2 veículos.

Figura 7 — Fluxograma da aplicação para uma imagem com 2 veículos.



Fonte: O autor (2023).

Após a aplicação de algum dos filtros à imagem, um processo de detecção de veículos utilizando o YOLOv3 foi utilizado. *A priori*, um modelo da rede neural é criado utilizando a função do *opencv* chamada *dnn.readNetFromDarknet*, passando como parâmetros as configurações e pesos do YOLOv3 para a detecção de veículos.

Para a identificação dos pontos de interesse na imagem, é feita uma cópia para a classe de detecção de veículos. Essa versão é alterada com as demarcações das *bounding boxes* dos

veículos e placas, bem como os respectivos níveis de confiança, sendo assim essa cópia é manuseada ao longo do processo de detecção.

A partir da cópia da imagem, é criado um *blob* (representação numérica de uma imagem). Isso é feito a partir da função do *opencv dnn.blobFromImage,* possibilitando a normalização e redimensionamento da imagem. O elemento *blob* é usado como entrada para a rede neural com a função *setInput* do modelo da rede previamente criada (ROSEBROCK, 2017).

Os nomes das camadas da rede neural são recuperados utilizando a função *getLayerNames* da mesma. Com isso uma lista de nomes contendo as camadas de saída são utilizadas com o retorno do método *getUnconnectedOutLayers* obtido a partir da rede neural, recuperando as camadas de saída não conectadas a rede.

Por fim, é utilizado o método *forward* chamado pela instância da rede neural passando como parâmetro os resultados das camadas de saída obtidas através do retorno da função anterior. Esse método é responsável por passar as camadas de saída pela rede, armazenando os resultados de camadas identificadas.

Destarte, após os dados de saída com os resultados do modelo serem aplicados a imagem, esses são percorridos, obtendo as pontuações de confiança de cada uma das classes selecionadas, que foram detectadas ao atravessar a imagem. Sendo assim, caso a classe retornada esteja entre as classes de veículos previamente estabelecidas (carro, moto, ônibus, caminhão) e o nível de confiança for maior que o limiar (limite inferior de detecção de veículo) determinado - 20% de confiança que algum dos veículos foram detectados -, esses veículos têm suas *bounding boxes*, ID's das classes dos veículos e valores de pontuação de confiança salvos. Esse valores são usados para alterações na cópia da imagem, demarcando-a com esses dados, além de gerar gráficos de resultados de acurácia.

Dessa forma, através dos resultados obtidos, delimitando as coordenadas das *bounding boxes* e confiança dos veículos da imagem, esses valores, bem como o limiar de confiança e o limiar da *NMS* (*Non-maximum Suppression*) são passados como parâmetro para a função da biblioteca *opencv dnn.NMSBoxes*, sendo responsável por remover as *bounding boxes* que estão muito próximas entre si e possuem pontuações de confiança semelhantes (SAMBASIVARAO, 2019).

Após o filtro realizado nas *bounding boxes*, o código verifica se alguma foi identificada, se não for, uma exceção será lançada, retornando que nenhum veículo foi encontrado na imagem. Caso contrário, cada um dos veículos delimitados será verificado, extraindo sua respectiva classe.

A partir da imagem demarcada do veículo - utilizando o pontos centrais x e y (pontos relativos a imagem original), altura e largura da *bounding box* -, é possível extrair uma sub-imagem a partir da original contendo apenas o veículo que deve ser analisado. Utilizando a sub-imagem, o detector de placas de veículos é acionado (o arquivo de configurações e

classes identificadas são diferentes, como foi citado na Seção 4.6). A estrutura da classe e funções foram as mesmas usadas para a detecção de veículos.

O valor das coordenadas da placa e o nível de confiança confiança (caso a mesma tenha sido encontrada), são retornados pela chamada do método de identificação de placas. Dessa forma, os valores de confiança dos veículos, placas e coordenadas podem ser pintados na imagem.

Por fim, a funcionalidade de detecção de veículos retorna a placa com maior nível de confiança entre os veículos detectados na imagem, como é demonstrado no Fluxograma 1.

4.5 EXTRAÇÃO DE RESULTADOS

Após a extração dos dados de confiança dos veículos das imagens, para cada um dos níveis diferentes de brilho, resolução e desfoque, gerou-se um novo arquivo do tipo *csv*. Os arquivos são gerados separadamente para cada tipo de filtro na imagem (um arquivo contém as informações de um dos filtros aplicados a todas as imagens testadas). O arquivo possui os seguintes dados: nome da imagem, intensidade do filtro e nível de confiança.

Isso é feito a partir do pacote *pandas*, utilizando a função *DataFrame*, com o conjunto de dados supracitados em formato de dicionário. Por fim, a tabela é salva - individualizada para cada filtro - através da função *to_csv* com o *DataFrame* criado.

Com base no arquivo *csv* criado, que contém estatísticas relacionadas à intensidade do filtro, imagem e porcentagem de acurácia, são gerados dois gráficos distintos. Um gráfico apresenta os erros de detecção de veículos, já o outro ilustra a curva de níveis de acurácia na detecção de placas na imagem.

Para gerar os gráficos de erro (falha na detecção do veículo), as linhas com valores de erro na coluna de porcentagem de acurácia são filtradas, armazenando as frequências desses valores. Em seguida, é criado um novo *DataFrame* com duas colunas, contendo os valores únicos de intensidades testadas e a quantidade de erros para cada intensidade. A partir desses valores, um novo gráfico de barras é gerado.

Gráficos relacionados aos níveis de acurácia associados a intensidade de filtro também serão criados. A princípio, os valores de erro serão filtrados na detecção de veículos, alterando seus níveis de confiança para zero. Posteriormente, a média da acurácia para cada nível é calculada para a visualização gráfica. Assim, um *DataFrame* é criado com os valores de intensidade única e acurácia média para cada intensidade. O *DataFrame* é salvo produzindo um gráfico de linha como sua representação.

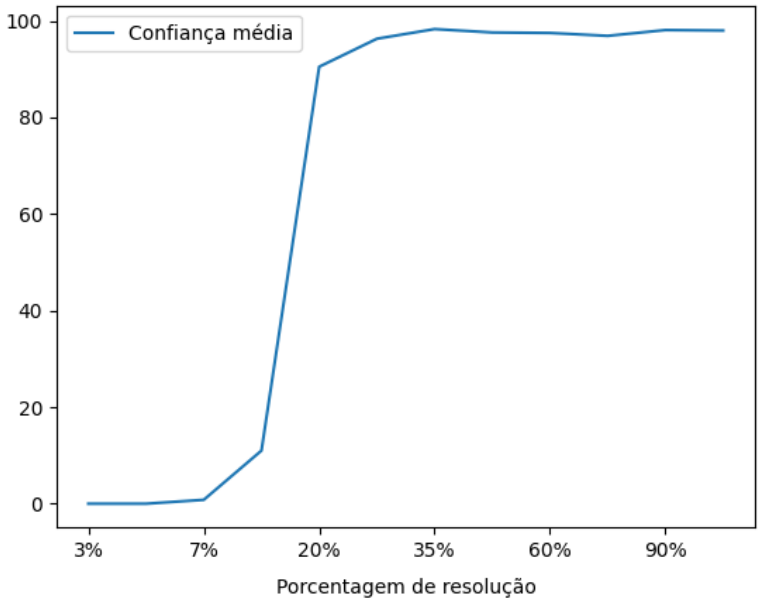
5 **RESULTADOS**

Neste capítulo, serão apresentados os resultados de acurácia obtidos a partir dos testes realizados com 300 imagens do banco de dados da UFOP, mencionado na Seção 4.1.1. Foram avaliadas as variações de acurácia ao modificar características como brilho, resolução e desfoque dessas imagens.

5.1 RESOLUÇÃO

As imagens testadas, possuem uma resolução de 800x600. Reduzimos um percentual da imagem original, iniciando com um valor de 3% da resolução inicial, equivalente a 24x18, até 95% da resolução, que equivale a 760x570, pode ser observado o impacto da mudança da resolução na média da acurácia de detecção no Gráfico 1.

Gráfico 1 — Média dos níveis de confiança da detecção de placas de veículo baseado na resolução da imagem.



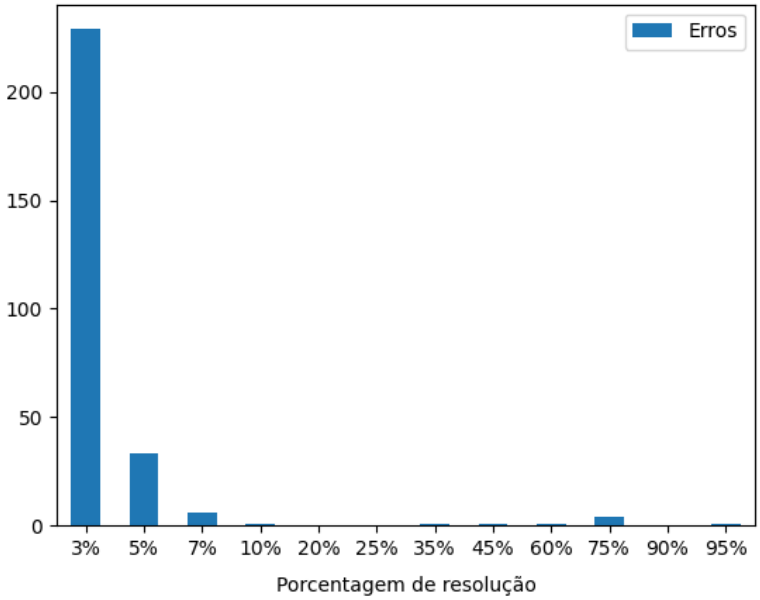
Fonte: O autor (2023).

De acordo com o gráfico apresentado, é possível observar que a redução percentual da resolução das imagens impacta significativamente na acurácia da detecção de placas. Quando a resolução é mantida acima de 20% da original (160x120), a acurácia se mostra relativamente estável.

Entretanto, quando a resolução é reduzida abaixo desse valor, é possível observar uma queda acentuada na acurácia. Isto indica que a qualidade da imagem, medida pela sua

resolução, é um fator crítico para o desempenho do modelo de detecção utilizado, demonstrado pelo gráfico no intervalo de 3% até 20%. A quantidade de erros (falha na detecção de veículos) por cada porcentagem de resolução testada está apresentada no Gráfico 2.

Gráfico 2 — Quantidade de erros de veículo baseado na resolução da imagem



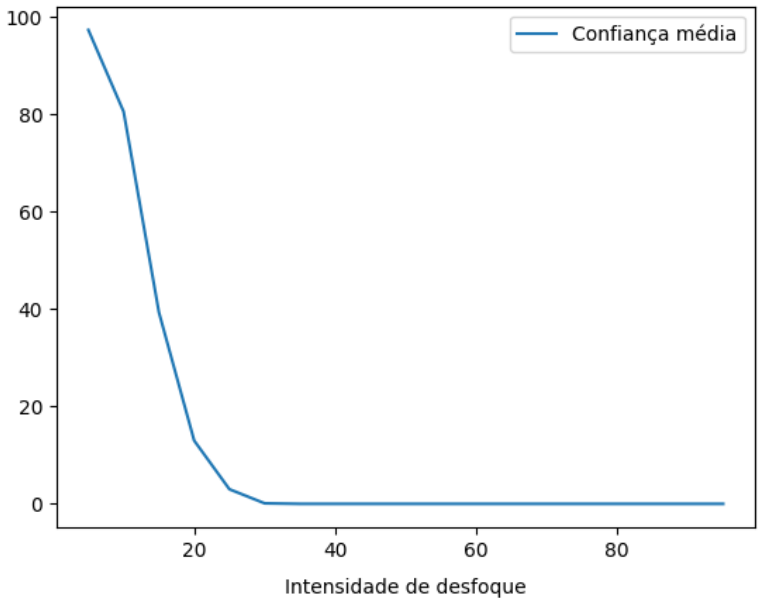
Fonte: O autor (2023).

O Gráfico 2 evidencia que a resolução das imagens, abaixo de 160x120, tem um impacto negativo na capacidade de detecção. É possível observar que níveis mais baixos de resolução estão associados a uma taxa de erro de detecção de veículos mais elevada.

5.2 DESFOQUE

O nível de desfoque varia de 0 - 100, onde quanto menor o valor, menor o nível de desfoque. Através do método mencionado na Seção 4.7.2.2, iniciou-se com um valor de 5 até 95, os passos são apresentados na Tabela 2. Pode-se observar no Gráfico 3 o comportamento da acurácia, através da mudança no tamanho no *Kernel* de desfoque, representando a intensidade de desfoque aplicado na imagem.

Gráfico 3 — Média dos níveis de confiança na detecção de placas de veículo baseado nos níveis de desfoque da imagem



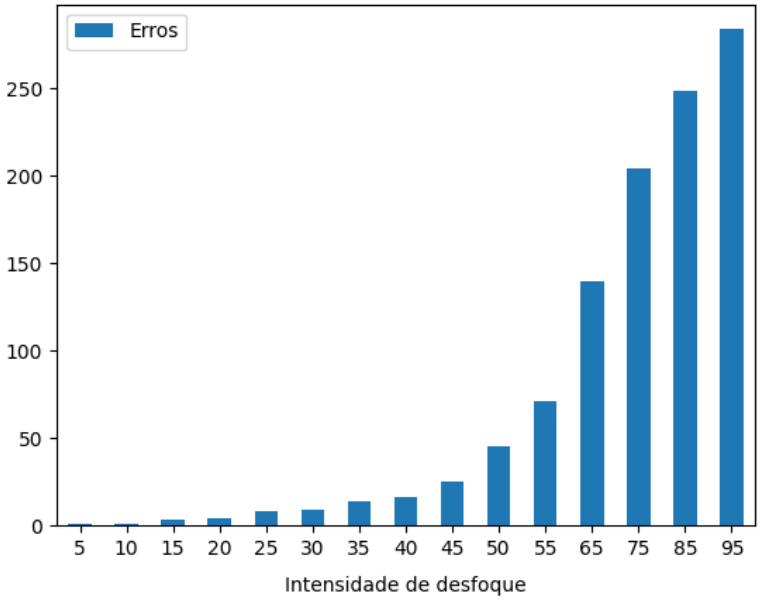
Fonte: O autor (2023).

É evidente que o grau de desfoque tem um impacto significativo no nível de acurácia. Esse impacto no contexto da YOLO pode ser explicado pelo fato de que o algoritmo utiliza características visuais presentes nas imagens, como bordas, contornos e padrões locais, para identificar objetos e suas respectivas classes. O desfoque, por sua natureza, suaviza essas características visuais e diminui o contraste entre diferentes regiões da imagem. Como resultado, a capacidade do YOLO de identificar e distinguir as características visuais específicas das placas de veículos é afetada negativamente à medida que o grau de desfoque aumenta.

Como é possível observar na imagem, valores acima de 15 já são suficientes para que o algoritmo de detecção de placas se torne ineficiente, obtendo níveis de acurácia próximos de zero.

Isso demonstra como uma foto desfocada pode ser um fator crítico para o desempenho do modelo de detecção utilizado. A quantidade de erros por cada nível de desfoque testado é apresentado no Gráfico 4.

Gráfico 4 — Quantidade de erros de detecção de veículos baseado nos níveis de desfoque da imagem



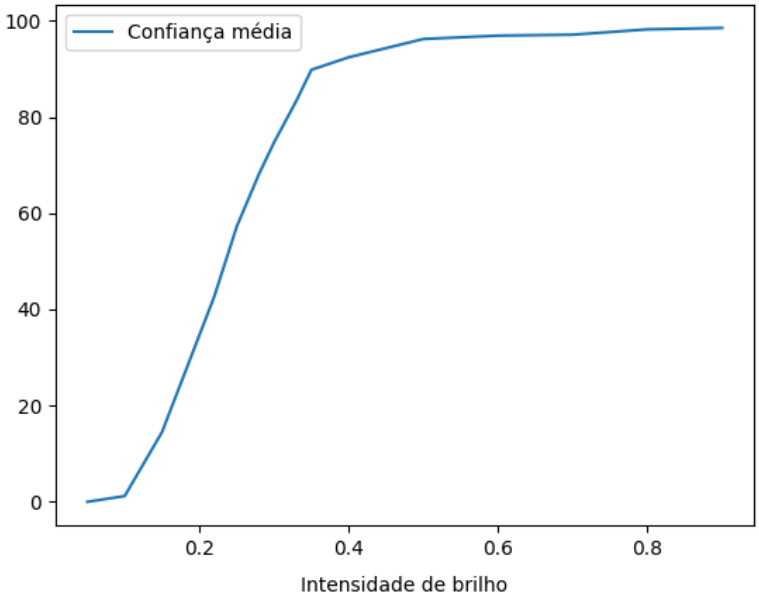
Fonte: O autor (2023).

No gráfico acima, observa-se a quantidade de veículos não detectados a medida que o nível de desfoque aumenta, demonstrando um aumento na quantidade de erros na detecção quanto maior o nível de desfoque.

5.3 BRILHO

O nível de brilho foi outra característica avaliada, para verificar seus impactos na detecção de placas de veículos. Os níveis intensidade vão variar entre 5% e 90%, os passos são apresentados na Tabela 2. Quanto menor o valor percentual, menor o brilho da imagem. O método usado para alterar os níveis de brilho na imagem está descrito na Seção 4.7.2.3. O Gráfico 5 apresenta a curva de acurácia da detecção de placas de veículos com a alteração da aplicação de valores para inversão de *gamma* na imagem, representado pela intensidade de brilho.

Gráfico 5 — Média da níveis de desfoque na detecção de placas de veículo baseado nos níveis de gamma da imagem.

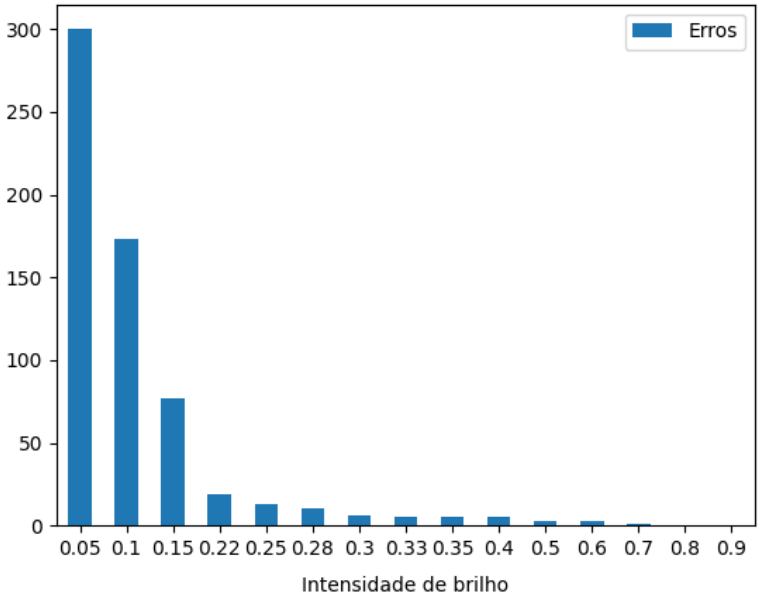


Fonte: O autor (2023).

A imagem apresentada ilustra como a variação de brilho afeta a detecção de placas de veículos. É possível observar que, para níveis percentuais de brilho acima de 0,4, a acurácia mantém-se em níveis mais elevados, acima de 80%.

Para níveis de intensidade abaixo de 0,4, a acurácia começa a diminuir, indicando que a detecção não é mais eficaz. O Gráfico 6 representa a quantidade de erros para cada nível de brilho testado.

Gráfico 6 — Quantidade de erros de detecção de veículos baseado nos níveis de gamma da imagem



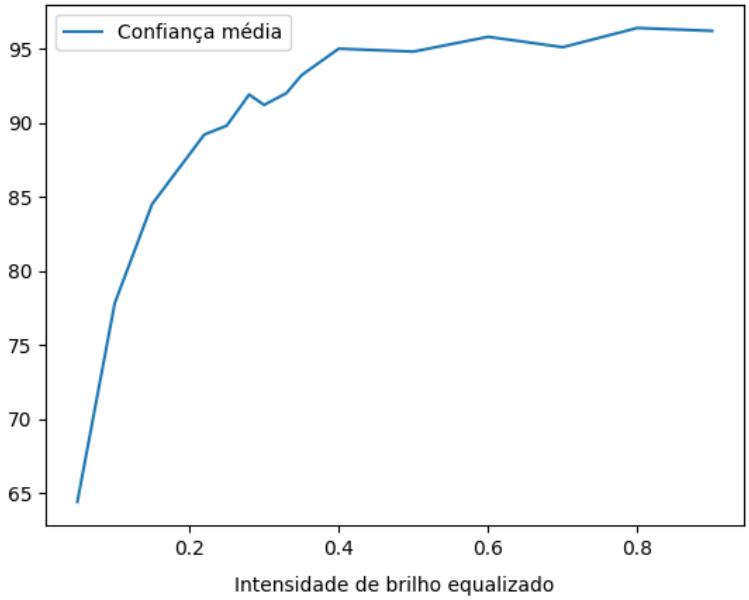
Fonte: O autor (2023).

Como mencionado acima, valores abaixo de 0,4 possuem níveis de confiança menores, ocasionando, também, mais erros de falha de detecção de veículos e placas de veículos. Pode-se observar que a medida que a imagem escurece, o nível de acurácia reduz e, consequentemente, aparecem mais cenários de não detecção de veículos.

5.3.1 **BRILHO EQUALIZADO**

Uma nova avaliação realizada a fim de diminuir vieses da análise (dado que imagens apresentam diferentes níveis de brilho), foi equalizar o nível de brilho de cada imagem, pois as imagens recuperadas em horários diferentes podem possuir um nível de brilho mais escuro ou mais claro devido a iluminação, mencionado na Seção 4.7.2.4. O Gráfico 7 representa a curva de acurácia na detecção de placas com os níveis de brilho equalizados.

Gráfico 7 — Média dos níveis de confiança de detecção de placas de veículo baseado nos níveis de brilho da imagem, utilizando imagens com brilho equalizado



Fonte: O autor (2023).

O Gráfico 7 ilustra uma acurácia reduzindo mais bruscamente em intensidades abaixo de 0,2, mostrando que mesmo com o brilho equalizado ainda há um decréscimo em acurácia. Em comparação com o Gráfico 5, a acurácia do modelo é maior para valores próximos de 0,2, denotando que modelo consegue manter níveis melhores de acurácia de detecção mesmo com baixos níveis de brilho. Ademais, os níveis variam de 65 a 95 de acurácia média, demonstrando que o uso de métodos de equalização de brilho de imagens resultaram em melhorias na detecção de placas de veículos.

5.4 ANÁLISE COMPARATIVA COM OUTROS TRABALHOS

Os resultados de acurácia obtidos estão de acordo com o que é observado na literatura, como é apresentado no Capítulo 3.

Neste trabalho, foi analisada a acurácia do modelo YOLOv3 na detecção de placas de veículos, comparando-a com diversos estudos que utilizaram técnicas semelhantes para melhorar a eficácia de modelos de detecção de objetos em imagens com características alteradas, como desfoque, baixa resolução e brilho.

Bhandari *et al.* (2020) aplicaram técnicas de equalização de histograma para melhorar a acurácia das redes neurais convolucionais, utilizando o erro médio quadrático (do inglês, - MSE) como métrica. Zheng *et al.* (2021) propuseram o método Deblur-YOLO para melhorar

a detecção de objetos em imagens desfocadas com YOLOv3 e criaram a métrica Smooth Peak Signal-to-Noise Ratio para avaliar a qualidade das imagens reconstruídas após o desembaçamento.

Jiang e Wang (2016) e Yin (2022) utilizaram imagens com características alteradas no treinamento do modelo, observando que a eficácia de YOLOv3 e YOLOv4 era inferior em imagens com desfoque gaussiano, de movimento e pixelização. Eles aplicaram a métrica mAP para avaliar o desempenho do modelo re-treinado.

Xu, Li e Shi (2022) identificaram que a precisão da detecção de navios com YOLOv4 em imagens de baixa resolução era significativamente inferior, propondo um novo modelo chamado LMO-YOLO para lidar com essa situação, apresentando melhor desempenho em termos de AP.

Embora as métricas utilizadas nos estudos mencionados sejam diferentes das empregadas neste trabalho - sendo mAP, F1 escore e AP usadas nos artigos relacionados, enquanto o trabalho atual avalia a média dos níveis de confiança -. Além disso, alguns dos trabalhos relacionados exploram outras versões do YOLO. De forma consistente com os estudos, observou-se uma diminuição na acurácia em imagens com características alteradas de brilho, desfoque e resolução, bem como uma melhoria na acurácia ao aplicar métodos de equalização de histograma.

6 **CONCLUSÃO**

Através da análise realizada sobre a detecção de placas e veículos com o modelo Yolov3 em imagens com características variadas, especificamente desfoque, brilho e resolução, foi possível observar que a acurácia do modelo diminuiu consideravelmente ao diminuir a qualidade das imagens através do uso dos filtros mencionados.

É importante destacar que essas características variadas são comuns em situações reais, como, por exemplo, em imagens capturadas por câmeras de trânsito com baixa qualidade ou em situações climáticas desfavoráveis. Dessa forma, esses resultados sugerem que ainda há espaço para melhorias no modelo, especialmente na detecção de placas e veículos em condições adversas.

No entanto, é possível destacar que ainda assim o modelo apresentou resultados satisfatórios. A intensidade de desfoque possuiu um limiar de acurácia com o valor 20 como tamanho do *kernel*, resolução apresentou uma curva acentuada de diminuição da confiança a partir de 20% da resolução da imagem original e brilho teve uma piora na capacidade de detecção a partir do valor de 0,4 de *gamma*, possuindo uma melhora considerável para níveis de brilho equalizados, com uma curva de diminuição de acurácia mais brusca para valores abaixo de 0,1.

6.1 TRABALHOS FUTUROS

Durante o desenvolvimento desse trabalho foram identificados vários tópicos que podem ser realizados como trabalhos futuros, como aplicar técnicas de melhoria de imagem com baixa resolução, alto desfoque e baixo brilho, como equalização de histograma e filtros de suavização, antes de passar as imagens para o modelo YOLOv3.

Além disso, outra possibilidade é aplicar técnicas de aprendizado profundo, como o uso de redes neurais adicionais, uso de técnicas de transferência de aprendizado, redes neural adversariais, utilização de *RetinexNet* como uma etapa de pré-processamento, além de incluir dados com características de brilho, desfoque, resolução alterados podem ajudar a aprimorar o desempenho do modelo.

Ao longo da realização deste projeto, notou-se que uma limitação importante é o *hardware* disponível para a realização de treinamento e testes do modelo YOLOv3. Com a disponibilidade de uma infraestrutura mais robusta e um conjunto de dados maior e mais diversificado, seria possível explorar melhorias significativas no desempenho do modelo em situações com brilho, desfoque e resolução alteradas.

Em trabalhos futuros, tendo acesso a uma infraestrutura aprimorada e a conjuntos de dados mais abrangentes, podem ser realizados testes com arquiteturas de redes neurais mais complexas, treinamento com maior número de épocas e experimentação de diferentes técnicas

de pré-processamento e pós-processamento de imagens. Esses avanços, combinados com as sugestões de aprimoramento mencionadas anteriormente, têm o potencial de melhorar significativamente a acurácia da detecção de placas de veículos em condições adversas, tornando o uso do YOLOv3 ainda mais eficaz em sistemas de segurança e fiscalização de trânsito.

Em geral, a utilização do YOLOv3 para detecção de placas de veículos pode ser aplicada em diversos sistemas de segurança e fiscalização de trânsito, mas ainda assim é necessário realizar testes e melhorias para aprimorar ainda mais a acurácia da detecção em condições de brilho, desfoque e resolução alteradas.

REFERÊNCIAS

ALEXEYAB. **darknet**. Github. Disponível em: https://github.com/AlexeyAB/darknet. Acesso em: 18 abr. 2023.

ALVES RODRIGUES, Diego. **DEEP LEARNING E REDES NEURAIS CONVOLUCIONAIS: RECONHECIMENTO AUTOMÁTICO DE CARACTERES EM PLACAS DE LICENCIAMENTO AUTOMOTIVO**. 2018 Trabalho de Conclusão de Curso (Ciência da Computação) - Universidade Federal da Paraíba, 2018.

ALVES, Gabriel. **Detecção de Objetos com YOLO – Uma abordagem moderna**. IA Expert Academy. 2020. Disponível em: https://iaexpert.academy/2020/10/13/deteccao-de-objetos-com-yolo-uma-abordagem-moderna/. Acesso em: 25 jun. 2023.

APPLICATION of Pandas in Data Science Industry. Study Tonight. Disponível em: https://www.studytonight.com/pandas/application-of-pandas-in-data-science-industry. Acesso em: 17 abr. 2023.

ARAR, Steve. **Digital Image Processing**: Point Operations to Adjust Brightness and Contrast. All About Circuits. 2019. Disponível em: https://www.allaboutcircuits.com/technical-articles/digital-image-processing-point-operations/. Acesso em: 25 jun. 2023.

AWARI. **Visão computacional**: o que é e como funciona esta inteligência artificial. 2023. Disponível em: https://awari.com.br/visao-computacional. Acesso em: 25 jun. 2023.

AWS. **What is deep learning**. Amazon. Disponível em: https://aws.amazon.com/what-is/deep-learning/. Acesso em: 12 abr. 2023.

BHANDARI, Aashish *et al*. **Image Enhancement and Object Recognition for Night Vision Surveillance**. 2020 Tese (Electronics and Communication Engineering) - Tribhuvan University, 2020. Disponível em: https://arxiv.org/pdf/2006.05787.pdf. Acesso em: 25 jun. 2023.

BOESCH, Gaudenz. **LabelImg for Image Annotation**. Viso.ai. Disponível em: https://viso.ai/computer-vision/labelimg-for-image-annotation/. Acesso em: 17 abr. 2023.

BOESCH, Gaudenz. **What is OpenCV?**: The Complete Guide. viso.ai. 2023. Disponível em: https://viso.ai/computer-vision/opencv/. Acesso em: 17 abr. 2023.

BROWNLEE, Jason. **A Gentle Introduction to Object Recognition With Deep Learning**. visaocomputacional. 2019. Disponível em: https://machinelearningmastery.com/object-recognition-with-deep-learning/. Acesso em: 12 abr. 2023.

Chadrick. **cv2 resize interpolation methods**. chadrick-kwag. 2018. Disponível em: https://chadrick-kwag.net/cv2-resize-interpolation-methods/. Acesso em: 19 abr. 2023.

CHARAN, Ashish. **Why GPU Can Process Image Much Faster than CPU?**. E2E Cloud. 2021. Disponível em: https://www.e2enetworks.com/blog/why-gpu-can-process-image-much-faster-than-cpu. Acesso em: 18 abr. 2023.

DARKNET. **YOLO**: YOLO: Real-Time Object Detection. pjreddie. Disponível em: https://pjreddie.com/darknet/yolo/. Acesso em: 28 dez. 2022.

DATASCIENCETEST. **NumPy**: the most used Python library in Data Science. DataScientest. Disponível em: https://datascientest.com/en/numpy-the-python-library-in-data-science. Acesso em: 17 abr. 2023.

DAUMAS, Marina . **Computer vision and Self-driving cars**: The challenges of applying artificial intelligence technologies to autonomous vehicles. UFRJ Nautilus. 2020. Disponível em: https://www.ufrjnautilus.com/post/vis%C3%A3o-computacional-e-carros-aut%C3%B4nomos. Acesso em: 11 abr. 2023.

DEWI, Christine; CHEN, Rung-Ching; YU, Hui. **Weight analysis for various prohibitory sign detection and recognition using deep learning**. 2020 Tese - Chaoyang University Of Technology (cyut) And The Higher Education Sprout Project, Ministry Of Education (moe, 2020. Disponível em: https://www.researchgate.net/publication/343973414_Weight_analysis_for_various_prohibitory_sign_detection_and_recognition_using_deep_learning. Acesso em: 25 jun. 2023.

DOCS.PYTHON. **random**: Generate pseudo-random numbers. docs.python. Disponível em: https://docs.python.org/3/library/random.html. Acesso em: 18 abr. 2023.

EDUCATIVE ANSWERS TEAM. **Series vs. DataFrame in Pandas**. educative. Disponível em: https://www.educative.io/answers/series-vs-dataframe-in-pandas. Acesso em: 17 abr. 2023.

EMMERT-STREIB, Frank *et al*. **An Introductory Review of Deep Learning for Prediction Models With Big Data**. 2020 Tese - Sec. Machine Learning And Artificial Intelligence. Disponível em: An Introductory Review of Deep Learning for Prediction Models With Big Data. Acesso em: 25 jun. 2023.

GAD, Ahmed. **Evaluating Object Detection Models Using Mean Average Precision**: Evaluating Object Detection Models Using Mean Average Precision In this article we will see see how precision and recall are used to calculate the Mean Average Precision (mAP).. KD Nuggets. 2021. Disponível em: https://www.kdnuggets.com/2021/03/evaluating-object-detection-models-using-mean-average-precision.html#:~:text=To%20evaluate%20object%20detection%20models,model%20is%20in%20its%20detections.. Acesso em: 15 abr. 2023.

GOOGLE COLAB. **Welcome to Colaboratory**. Google. Disponível em: https://colab.research.google.com/. Acesso em: 18 abr. 2023.

HEARTEX. **LabelImg**. Github. 2022. Disponível em: https://github.com/heartexlabs/labelImg. Acesso em: 17 abr. 2023.

JIANG, Hao; WANG, Shiquan. **Object Detection and Counting with Low Quality Videos**. 2016 Tese - Stanford, 2016. Disponível em: http://cs231n.stanford.edu/reports/2016/pdfs/287_Report.pdf. Acesso em: 25 jun. 2023.

JOSHI, Prateek; ESCRIVA, David Millan; GODOY, Vinicius. **OpenCV By Example**. Packt Publishing Ltd, v. 3, f. 148, 2016. 296 p.

KATHURIA, Ayoosh. **What's new in YOLO v3?**. 2018. Disponível em: https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b. Acesso em: 25 jun. 2023.

KEITA, Zoumana. **YOLO Object Detection Explained**. Dataacamp. 2022. Disponível em: https://www.datacamp.com/blog/yolo-object-detection-explained. Acesso em: 15 abr. 2023.

KHANAL, Samir . **Image Resolution**. Medium. 2020. Disponível em: . Acesso em: 15 mai. 2023.

KUNDU, Rohit. **YOLO: Algorithm for Object Detection Explained [+Examples]**: What is YOLO architecture and how does it work? Learn about different YOLO algorithm versions and start training your own YOLO object detection models.. V7. 2023. Disponível em: https://www.v7labs.com/blog/yolo-object-detection. Acesso em: 15 abr. 2023.

LIN, Tsung-Yi *et al*. **Microsoft COCO**: Common Objects in Context. arxviv. 2014. Disponível em: https://arxiv.org/abs/1405.0312. Acesso em: 17 abr. 2023.

LOPES, Nathan. **DESCRITORES APLICADO AO PROCESSAMENTO DE IMAGENS DIGITAIS.**. Mossoró, 2019 Trabalho de Conclusão de Curso (CIÊNCIA E TECNOLOGIA) - Universidade Federal Rural do Semi-árido, Mossoró, 2019.

MANTRIPRAGADA, Manogna. **Digging deep into YOLO V3 - A hands-on guide Part 1**. 2020. Disponível em: https://towardsdatascience.com/digging-deep-into-yolo-v3-a-hands-on-guide-part-1-78681f2c7e29. Acesso em: 25 jun. 2023.

MENDES JÚNIOR, Pedro *et al*. **Towards an automatic vehicle access control system:** : License plate location. IEEE: International Conference on Systems, Man, and Cybernetics (SMC). 2020. Disponível em: https://ieeexplore.ieee.org/document/6084108. Acesso em: 17 abr. 2023.

METTZER. **O melhor editor para trabalhos acadêmicos já feito no mundo**. Mettzer. Florianópolis, 2016. Disponível em: http://www.mettzer.com/. Acesso em: 21 ago. 2016.

MICROSOFT. **Aprendizado profundo x Aprendizado de máquina em Azure Machine Learning versus Machine Learning**. Microsoft. 2022. Disponível em: https://learn.microsoft.com/pt-br/azure/machine-learning/concept-deep-learning-vs-machine-learning. Acesso em: 11 abr. 2023.

MOURA, Natan; CLARO, Daniela Barreiro; GONDIM, João Medrado. **Análise experimental para a detecção de objetos em vídeos decâmeras de vigilância**: Uma abordagem para porte de arma, incêndio e pichação. Salva, 2021 Trabalho de Conclusão de Curso - Universidade Federal da Bahia. Disponível em: https://doi.org/10.5753/webmedia_estendido.2021.17608. Acesso em: 11 abr. 2023.

NAUFAL, Akeyla. **How Image Blurring Works**. Medium. 2020. Disponível em: https://medium.com/swlh/how-image-blurring-works-652051aee2d1. Acesso em: 25 jun. 2023.

NELSON, Joseph. **Your Comprehensive Guide to the YOLO Family of Models**. roboflow. 2021. Disponível em: https://blog.roboflow.com/guide-to-yolo-models/. Acesso em: 25 jun.

2023.

QUEIROZ, José; GOMES, Herman. **Introdução ao Processamento Digital de Imagens** . Campina Grande, 2006 Tese - Universidade Federal de Campina Grande, Campina Grande. Disponível em: http://www.dsc.ufcg.edu.br/~hmg/disciplinas/graduacao/vc-2016.2/Rita-Tutorial-PDI.pdf. Acesso em: 25 jun. 2023.

RADEČIĆ, Dario. **How to Detect License Plates with Python and YOLO**. Better data science. 2020. Disponível em: https://betterdatascience.com/detect-license-plates-with-yolo/. Acesso em: 18 abr. 2023.

REDMON, Joseph *et al*. **You Only Look Once**: Unified, Real-Time Object Detection. 2016 Tese - University Of Washington, Allen Institute For Ai, Facebook Ai Research, 2016. Disponível em: https://doi.org/10.48550/arXiv.1506.02640. Acesso em: 25 jun. 2023.

RIBEIRO MENDES JÚNIOR, Pedro *et al*. **Dataset for Vehicle License Plate Location**. 2020. Disponível em: https://pedrormjunior.github.io/dataset-VLPL.html. Acesso em: 17 abr. 2023.

RIBEIRO MENDES JÚNIOR, Pedro *et al*. **Dataset for Vehicle License Plate Location**. pedrormjunior.github.io. Campinas. Disponível em: https://pedrormjunior.github.io/dataset-VLPL.html. Acesso em: 10 dez. 2022.

ROSEBROCK, Adrian. **Deep learning: How OpenCV's blobFromImage works**. pyImageSearch. 2017. Disponível em: https://pyimagesearch.com/2017/11/06/deep-learning-opencvs-blobfromimage-works/. Acesso em: 20 abr. 2023.

ROSEBROCK, Adrian. **Intersection over Union (IoU) for object detection**. Pyimagesearch. 2016. Disponível em: https://pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/. Acesso em: 25 jun. 2023.

ROSEBROCK, Adrian. **OpenCV Smoothing and Blurring**. pyimagesearch. 2021. Disponível em: https://pyimagesearch.com/2021/04/28/opencv-smoothing-and-blurring/. Acesso em: 19 abr. 2023.

Sambasivarao. **Non-maximum Suppression (NMS)**. 2019. Disponível em: https://towardsdatascience.com/non-maximum-suppression-nms-93ce178e177c. Acesso em: 20 abr. 2023.

SHAH, Deval. **Mean Average Precision (mAP) Explained**: Everything You Need to Know. V7. 2022. Disponível em: https://www.v7labs.com/blog/mean-average-precision. Acesso em: 25 jun. 2023.

SHARMA, Aditya. **Introduction to the YOLO Family**. pyimagesearch. 2022. Disponível em: https://pyimagesearch.com/2022/04/04/introduction-to-the-yolo-family/. Acesso em: 25 jun. 2023.

STERLING, Bruce. **Datasets for deep learning**. Wired. 2020. Disponível em: https://www.wired.com/beyond-the-beyond/2020/02/datasets-deep-learning/. Acesso em: 17 abr. 2023.

TUTORIALS POINT. **Does pandas depend on NumPy?**. Tutorials Point. Disponível em: https://www.tutorialspoint.com/does-pandas-depend-on-numpy. Acesso em: 17 abr. 2023.

VERMA, Sahil. **YOLOV3-on-Android**. Github. 2019. Disponível em: https://github.com/huuuuusy/YOLOV3-on-Android/blob/master/Readme_English.md. Acesso em: 12 abr. 2023.

XU, Qizhi ; LI, Yuan; SHI, Zhenwei. **LMO-YOLO**: A Ship Detection Model for Low-Resolution Optical Satellite Imagery. 2022 Tese - Ieee, 2022. Disponível em: https://ieeexplore.ieee.org/document/9779558. Acesso em: 25 jun. 2023.

YIN, Dan. **An Improved Algorithm for Target Detection in Low Light Conditions**. 2022 Tese - Journal Of Physics, 2022. Disponível em: https://iopscience.iop.org/article/10.1088/1742-6596/2203/1/012045/pdf. Acesso em: 25 jun. 2023.

ZHAO, Zhong-Qiu; XU, Shou-tao; WU, Xindong. **Object Detection with Deep Learning**: A Review. 2019 Tese - Ieee, 2019. Disponível em: https://arxiv.org/pdf/1807.05511.pdf&usg=ALkJrhhpApwNJOmg83O8p2Ua76PNh6tR8A. Acesso em: 25 jun. 2023.

ZHENG, Shen *et al*. **Deblur-YOLO**: Real-Time Object Detection with Efficient Blind Motion Deblurring. Wenzhou, China, 2021 Tese - Wenzhou-kean University, 2021. Disponível em: https://ieeexplore.ieee.org/abstract/document/9534352. Acesso em: 25 jun. 2023.