UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

Uma Proposta Para Detecção de Buracos com Aprendizagem de Máquina na Borda

Jordão Paulino Cassiano da Silva

Orientador: Prof. Dr. Ivanovitch Medeiros Dantas da Silva

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica e de Computação da UFRN (área de concentração: Engenharia de Computação) como parte dos requisitos para obtenção do título de Mestre em Ciências.

Número de ordem PPgEEC: 674
Natal, RN, Maio de 2024

Universidade Federal do Rio Grande do Norte - UFRN
Sistema de Bibliotecas - SISBI
Catalogação de Publicação na Fonte. UFRN - Biblioteca Central Zila Mamede

Silva, Jordao Paulino Cassiano da.
Uma proposta para detecção de buracos com aprendizagem de máquina na borda / Jordao Paulino Cassiano da Silva. - 2024.
69f.: il.

Dissertação (Mestrado) - Universidade Federal do Rio Grande do Norte, Centro de Tecnologia, Programa de Pós Graduação em Engenharia Elétrica e de Computação, Natal, 2024.
Orientação: Ivanovitch Medeiros Dantas da Silva.

1. Aprendizado de máquina na borda - Dissertação. 2. Infraestrutura urbana - Dissertação. 3. Detecção de buracos - Dissertação. 4. Otimização de modelos - Dissertação. 5. TinyML - Dissertação. I. Silva, Ivanovitch Medeiros Dantas da. II. Título.

RN/UF/BCZM CDU 621.3

Elaborado por Raimundo Muniz de Oliveira - CRB-15/429

"Não me preocupo com o que os computadores serão capazes de fazer no futuro. Me preocupo mais com o que os humanos farão com eles."— Grace Hopper

Agradecimentos

A Deus, por Sua presença eterna e graciosa, que guia e ilumina o meu caminhar.

À minha família, por um amor e sacrifício inestimáveis, fundamentais para cada conquista minha.

À minha amada esposa, Luanny, pelo amor constante, apoio inabalável e inspiração diária.

Ao meu orientador, Ivanovitch Silva, pela orientação precisa, paciência incansável e sabedoria essencial.

Aos colegas do grupo de pesquisa Conect2AI, pela colaboração enriquecedora e pelo companheirismo constante.

Aos colegas e mestres da pós-graduação, por abrir novos horizontes de conhecimento e enriquecer minha jornada acadêmica.

A todos os meus amigos e colegas, por caminharem comigo, compartilhando alegrias e desafios.

Resumo

Buracos em vias urbanas representam um problema significativo, afetando tanto a segurança dos usuários quanto a durabilidade dos veículos. Este estudo surge da necessidade urgente de soluções eficazes para a detecção de buracos, que possam ser implementadas em tempo real utilizando dispositivos com recursos computacionais limitados. Foi desenvolvida uma abordagem inovadora que integra aprendizado de máquina em dispositivos de borda, com ênfase nos modelos YOLOv8 e FOMO no contexto do TinyML. Utilizou-se um conjunto de dados especializado, contendo imagens anotadas, para treinar esses modelos na detecção precisa de buracos. A otimização do desempenho dos modelos YOLOv8 e FOMO para dispositivos de borda assegura eficiência em tempo real. Este trabalho não apenas fornece modelos eficazmente treinados, mas também apresenta um framework adaptável para a detecção de buracos, garantindo uma implementação prática e eficiente. Ademais, é proposto um pipeline completo para a detecção de buracos, validando a precisão e eficiência dos modelos. Esta abordagem oferece uma solução robusta para o reconhecimento automático de buracos, contribuindo significativamente para melhorias na manutenção de infraestruturas urbanas.

Palavras-chave: infraestrutura urbana, aprendizado de máquina na borda, YOLOv8, FOMO, TinyML, detecção de buracos, otimização de modelos, processamento em tempo real.

Abstract

Potholes in urban roads represent a significant problem, affecting both user safety and vehicle durability. This study addresses the urgent need for effective solutions for pothole detection that can be implemented in real-time using devices with limited computational resources. An innovative approach was developed, integrating machine learning on edge devices, with an emphasis on YOLOv8 and FOMO models within the TinyML context.

A specialized dataset, containing annotated images, was used to train these models for accurate pothole detection. The optimization of YOLOv8 and FOMO models' performance for edge devices ensures real-time efficiency. This work not only provides effectively trained models but also presents an adaptable framework for pothole detection, ensuring practical and efficient implementation. Additionally, a complete pipeline for pothole detection is proposed, validating the models' accuracy and efficiency. This approach offers a robust solution for the automatic recognition of potholes, significantly contributing to improvements in urban infrastructure maintenance.

Keywords: urban infrastructure, edge machine learning, YOLOv8, FOMO, TinyML, pothole detection, model optimization, real-time processing.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Símbolos e Abreviaturas

n	Total de imagens
n_{train}	Divisão de treinamento
n_{val}	Divisão de validação
θ	Parâmetros do Modelo
CLI	Command Line Interface
CNN:	Rede Neural Convolucional
COCO	Common Objects in Context
CPU:	Unidade Central de Processamento
FOMO	Faster Objects More Objets - Modelo de detecção de objetos desenvolvido pela Edge Impulse
FP	Falsos Positivos
FP32	Ponto flutuante de 32 bits, padrão IEEE 754.
GLCM:	Matriz de Coocorrência de Níveis de Cinza
GPDR:	Regulamento Geral sobre a Proteção de Dados
GPS:	Sistema de Posicionamento Global
GPU:	Unidade de Processamento Gráfico
IA	Inteligência Artificial
IEEE	Institute of Electrical and Electronics Engineers
InceptionV3:	Arquitetura avançada de rede neural desenvolvida pelo Google, otimizada para eficiência computacional em visão computacional.
INT8	Inteiro de 8 bits, -128 a 127
IoT	Internet das coisas
KNN:	k-Vizinhos Mais Próximos

LBP: Padrão Binário Local

LGPD: Lei Geral de Proteção de Dados

LiDAR: Light Detection and Ranging

MCU: Unidade de Microcontrolador

ML Machine Learning

MLP: Perceptron Multicamada

MobileNet: Rede neural convolucional projetada para eficiência em dispositivos móveis e embarcados, adequada para aplicações com recursos limitados.

ONNX Open Neural Network Exchange

OTA Over-the-air

PSD: Processamento de Sinal Digital

SVM: Máquina de Vetores de Suporte

TinyML: Tiny Machine Learning

TP Verdadeiros Positivos

UAV: Veículo Aéreo Não Tripulado

VGG16: Modelo de rede neural convolucional desenvolvido pela Visual Geometry Group para tarefas de visão computacional.

YOLO: You Only Look Once - Modelo de rede neural para detecção de objetos em tempo real, conhecido por sua velocidade e eficiência.

Capítulo 1

Introdução

A problemática da identificação de buracos nas vias urbanas é um desafio de grande magnitude, que afeta diretamente o bem-estar e a segurança dos condutores (Fan et al. 2020). Além disso, a manutenção dessas vias requer um investimento significativo em termos de recursos financeiros e de mão de obra (Koch et al. 2015). A título de exemplo, em 2022, a cidade de Belo Horizonte no Brasil despendeu R$ 40 milhões em ações de reparação de buracos (Prefeitura de Belo Horizonte 2022), enquanto que, em 2017, conselhos municipais na Nova Zelândia alocaram vários milhões de dólares para a detecção e reparo de buracos, sendo que só em Christchurch foram dedicados $525 mil para esses fins (*Christchurch: The pothole capital of New Zealand* 2023). Neste contexto, a adoção de sistemas automatizados para a detecção de buracos emerge como uma solução valiosa para auxiliar as autoridades locais a manterem as estradas em condições ideais. Esta necessidade torna-se ainda mais premente considerando o rápido desenvolvimento da computação de borda e a incorporação de modelos avançados de inteligência artificial em pequenos dispositivos eletrônicos, o que exige a formulação de estratégias eficazes e com bom custo-benefício que tirem partido dessas inovações tecnológicas.

A detecção automatizada de buracos não é apenas uma otimização dos serviços públicos; ela é essencial para o desenvolvimento urbano sustentável. Esta abordagem abrange aspectos sociais, econômicos e ambientais, alinhando-se com os princípios de sustentabilidade urbana, fundamentais no planejamento de cidades modernas (Kim et al. 2022). Através da melhoria na conservação das vias, a detecção de buracos promove uma melhor qualidade de vida, um uso mais eficiente dos recursos públicos e uma diminuição do impacto ambiental (Xin et al. 2023).

A evolução da computação de borda, integrando modelos avançados de inteligência artificial diretamente nos dispositivos, é crucial para a modernização da gestão de infraestruturas urbanas. Ela permite o processamento de dados em tempo real no local de origem, como em veículos ou sensores de estrada, superando os desafios de latência, consumo elevado de banda larga e questões de privacidade enfrentados pelos modelos convencionais baseados na nuvem (Bourechak et al. 2023).

Neste contexto, o Tiny Machine Learning (TinyML) surge como um avanço significativo, otimizando modelos de aprendizado de máquina para dispositivos de borda limitados. Esta tecnologia não só acelera a detecção de buracos através da redução de latência e aumento da eficiência, mas também se alinha com regulamentos de privacidade de da-

dos, como a LGPD (Lei Geral de Proteção de Dados) no Brasil e o GDPR (Regulamento Geral sobre a Proteção de Dados) na União Europeia, processando dados localmente e minimizando a transmissão de dados externos. O rápido desenvolvimento em otimização de algoritmos e aceleração de hardware no campo do TinyML abre novas perspectivas para melhorias na manutenção de vias urbanas, assegurando vias mais seguras e bem mantidas. (Kallimani et al. 2023)

1.1 Motivação

A evolução acelerada da computação de borda, evidenciada pela crescente disponibilidade de dispositivos compactos e ultracompactos com arquiteturas tipo MCU (*Microcontroller Unit - Unidade de Microcontrolador*), CPUs (*Central Processing Units - Unidades de Processamento Central*) e até sistemas integrados CPU/GPU (*Graphics Processing Unit - Unidade de Processamento Gráfico*), oferece uma oportunidade ímpar para aplicar esses avanços em benefício da sociedade. A acessibilidade e redução de custos desses hardwares abrem caminho para soluções inovadoras no monitoramento e manutenção de infraestruturas urbanas, especialmente rodovias. Este desenvolvimento tecnológico é particularmente pertinente no contexto das vias urbanas frequentemente esburacadas, que não só causam danos significativos aos veículos, mas também aumentam o risco de acidentes para os usuários da estrada.

Adicionalmente, a natureza pervasiva dos veículos que trafegam diariamente por essas vias, incluindo os de transporte público com rotas pré-definidas e outros que circulam repetidamente pela mesma região, representa uma plataforma ideal para a implantação de sistemas de detecção de buracos automatizados. Estes veículos podem ser equipados com sensores avançados que, ao operar em conjunto com tecnologias de borda, permitem a coleta e análise de dados em tempo real, maximizando a eficácia na identificação e mapeamento de irregularidades nas superfícies das estradas. A implementação dessas tecnologias não só contribui para uma gestão mais eficiente e proativa das vias urbanas, mas também alinha-se com práticas sustentáveis de manutenção, reduzindo a necessidade de intervenções frequentes e dispendiosas.

Esta abordagem, ao tirar partido da integração de hardware avançado e algoritmos otimizados em dispositivos móveis, não apenas fortalece a infraestrutura urbana, mas também promove uma maior segurança e qualidade de vida para a população. Assim, a combinação de hardware acessível, software inteligente e a capacidade de processamento local transforma o panorama de manutenção de vias, potencializando a adoção de estratégias mais inteligentes e responsivas para enfrentar um dos problemas mais persistentes e dispendiosos enfrentados por cidades em todo o mundo.

1.2 Contribuições

Esta dissertação visa contribuir para a área de detecção de buracos em vias urbanas, explorando o uso de tecnologias de inteligência artificial em dispositivos de borda. As contribuições destacadas adiante refletem os esforços e resultados do estudo, abordando desde revisões bibliográficas até implementações práticas, visando aprimorar a compreensão e a aplicabilidade das tecnologias emergentes na manutenção de infraestruturas

urbanas:

- **Aplicação de Revisão Bibliográfica Atualizada:** Foi realizada uma revisão das metodologias mais recentes para a detecção automatizada de buracos, destacando-se os avanços tecnológicos no campo da visão computacional aplicados em infraestruturas urbanas.
- **Implementação de Solução na Borda:** Foi desenvolvida uma solução prática utilizando modelos de inteligência artificial destinada aos dispositivos de borda, demonstrando a viabilidade de técnicas modernas para resolução de problemas em tempo real.
- **Demonstração de Implementação em Dispositivos de Borda:** Foi detalhado o processo de adaptação e implementação dos modelos de inteligência artificial em dispositivos de borda utilizando a ferramenta Edge Impulse, evidenciando como soluções modernas podem ser eficazes em contextos reais.
- **Disponibilização de Estrutura Reproduzível:** Com base na experiência adquirida, foi proposta uma estrutura de pipeline adaptável e reproduzível para a implementação de tecnologias similares em diferentes modelos e hardwares, ressaltando a aplicabilidade e flexibilidade das técnicas modernas em monitoramento de rodovias.

1.3 Estrutura do Trabalho

Este trabalho está estruturado em seis capítulos, incluindo este, que é introdutório. No Capítulo 2, realiza-se uma revisão do estado da arte das estratégias atuais desenvolvidas e disponíveis para a detecção automatizada de buracos, a fim de posicionar o objetivo deste trabalho. O Capítulo 3 traz o ferramental teórico utilizado no desenvolvimento da proposta deste trabalho. O Capítulo 4 descreve a metodologia proposta, desde a obtenção dos dados até a obtenção e implementação dos modelos. O Capítulo 5 apresenta os experimentos e resultados e discute-os. O Capítulo 6 apresenta as principais conclusões e direcionamentos futuros.

Capítulo 2

Estado da Arte

Esta seção tem como objetivo revisar as técnicas e métodos contemporâneos para a detecção de buracos em rodovias. Esta revisão é estruturada em subseções que abordam os principais domínios da pesquisa e prática atuais, suas principais vantagens e desvantagens, delineando o cenário em que nossa proposta se insere.

2.1 Visão Computacional e Processamento de Imagens

A visão computacional e o processamento de imagens são campos tecnológicos avançados que permitem às máquinas interpretar informações visuais de imagens ou vídeos. Combinando métodos clássicos e modernas estratégias de deep learning, esses campos se destacam pela extração de características, onde algoritmos avançados identificam elementos significativos nas imagens para emular a visão humana. Enquanto os métodos clássicos, como detecção de bordas e reconhecimento de padrões, processam diretamente as imagens, o deep learning aplica redes neurais profundas para aprender e representar dados em complexidade crescente (Szeliski 2022).

Essenciais em diversas aplicações, desde inspeção industrial até melhorias na interação humano-computador, essas técnicas também são cruciais na detecção automática de anomalias em estradas, otimizando a segurança e manutenção de infraestruturas (Gonzalez & Woods 2018). Constantemente evoluindo, o campo integra abordagens inovadoras para solucionar desafios complexos na análise de informações visuais, destacando-se como uma área de crescente importância e aplicação prática (Goodfellow et al. 2016).

2.1.1 Técnicas e Algoritmos

Em Murali et al. (2022), a detecção de buracos é realizada com a ajuda da visão computacional, utilizando especificamente o algoritmo de detecção borda de Canny para realizar a segmentação e agrupamento das imagens. A aplicação desse método permite a identificação dos buracos, que são destacados ao delinear os contornos detectados.

Em Fernandes et al. (2023), a detecção de buracos é realizada a partir da extração de características da imagem a partir de descritores de textura como a Matriz de Coocorrência de Níveis de Cinza (GLCM), o Padrão Binário Local (LBP) e os Filtros de Gabor, essas características, então, são avaliadas utilizando técnicas de aprendizado de máquina como SVM (Máquina de Vetores de Suporte), KNN (k-Vizinhos Mais Próximos) e MLP

(Perceptron Multicamada).

No estudo de Dhoundiyal et al. (2023) é abordada uma aplicação de técnicas de aprendizado profundo para a detecção automática de buracos em estradas, visando substituir os métodos tradicionais de inspeção visual e uso de maquinário dispendioso. O objetivo é identificar o modelo de aprendizado profundo mais eficiente e preciso para essa tarefa, e para isso são utilizados modelos de *deep learning* InceptionV3, VGG16 e MobileNet.

2.1.2 Perspectivas da Tecnologia de Visão Computacional e Processamento de Imagens para Detecção de Buracos

A utilização da tecnologia de visão computacional oferece uma série de benefícios significativos para a detecção de buracos nas estradas. Ela possibilita a automação do processo de detecção, reduzindo a necessidade de inspeções manuais demoradas e aumentando a eficiência operacional. Além disso, com o processamento de imagens avançado, é viável realizar a detecção em tempo real, o que é crucial para a integração em sistemas de veículos autônomos e para a manutenção proativa das estradas. Técnicas avançadas de visão computacional podem alcançar uma alta precisão na identificação de buracos, especialmente em condições controladas ou com iluminação adequada.

Por outro lado, apesar das vantagens, a tecnologia de visão computacional também apresenta algumas limitações e desafios. A precisão da detecção pode ser prejudicada por condições ambientais adversas, como iluminação insuficiente, chuva ou neve, o que pode afetar negativamente o desempenho do sistema. Além disso, a qualidade da detecção depende significativamente da resolução e da qualidade das câmeras utilizadas, o que pode exigir um investimento considerável em hardware de alta qualidade. O desenvolvimento de sistemas eficazes de visão computacional também requer expertise técnica em processamento de imagens e aprendizado de máquina, além de ser, muitas vezes, computacionalmente intensivo, o que pode representar um desafio adicional para a implementação dessas soluções.

2.2 Sistemas de Detecção Baseados em Sensores

Sistemas de detecção baseados em sensores também têm representado uma alternativa para identificar buracos e irregularidades em estradas, utilizando uma variedade de tecnologias sensoriais para capturar informações físicas e ambientais. Esses sistemas podem incluir sensores acústicos, vibracionais, inerciais, e até LiDAR (Light Detection and Ranging), cada um com suas especificidades e aplicações.

2.2.1 Tecnologias e Aplicações

Sensores Acústicos

No trabalho desenvolvido por Mednis et al. (2010) foi proposta uma metodologia em que se utilizam de microfones embarcados em veículos, associados a um GPS, para detecção de buracos e outros tipos de danos em rodovias. Os veículos são dirigidos em ruas públicas e medem sinais sonoros induzidos por buracos, em seguida o sinal sonoro captado passa por um processo de discretização e, relacionado à informação de localiza-

ção do GPS, é varrido por uma função de detecção de evento em um processo de PSD (processamento de sinal digital), o trabalho.

Em Ozoglu & Gökgöz (2023) é proposto um sistema inovador para a detecção de buracos em estradas, empregando sensores inerciais e GPS já integrados aos smartphones, eliminando a necessidade de equipamentos adicionais nos veículos. Através do uso de redes neurais convolucionais, o sistema analisa os dados de vibração capturados pelos sensores inerciais (acelerômetros e giroscópios) dos smartphones, transformando-os em imagens para a detecção eficaz de irregularidades na superfície da estrada. Os resultados demonstraram alta precisão na identificação de buracos, validados por testes em diferentes rotas, comprovando a efetividade do método sem incorrer em custos extras para os usuários.

LiDAR

No estudo desenvolvido por Kang & Choi (2017) foi desenvolvido um sistema e um método para detectar buracos em pavimentos asfálticos, utilizando uma combinação de LiDAR 2D e câmera, visando uma manutenção de estradas mais eficaz. O sistema emprega um algoritmo avançado que inclui várias etapas de processamento tanto para dados LiDAR quanto para imagens, aprimorando a precisão na identificação e caracterização dos buracos.

2.2.2 Perspectivas da tecnologia baseada em sensores

A tecnologia baseada em sensores oferece uma série de benefícios significativos. Em primeiro lugar, destaca-se a alta sensibilidade dos sensores específicos, os quais podem detectar até mesmo as menores variações na superfície da estrada, permitindo uma detecção precisa de buracos e irregularidades, o que é crucial para a segurança viária. Além disso, a versatilidade é uma característica essencial desses sistemas, pois podem ser configurados para utilizar diferentes tipos de sensores, adaptando-se assim às características específicas da estrada e às condições ambientais. A combinação de dados provenientes de múltiplos sensores possibilita uma visão mais abrangente da condição da estrada, o que melhora significativamente a precisão da detecção de problemas.

No entanto, as tecnologias baseadas em sensores também apresentam desafios e desvantagens importantes que precisam ser considerados. O principal entre eles é o custo associado à implementação desses sistemas, especialmente quando se trata de tecnologias mais avançadas como o LiDAR. A complexidade de integração é outro aspecto crítico, já que a criação de um sistema coeso que integre diversos tipos de sensores requer uma infraestrutura complexa e análise de dados avançada. Além disso, a manutenção dos sensores expostos ao ambiente e ao desgaste pode ser uma preocupação, demandando manutenção frequente para garantir a precisão contínua do sistema.

2.3 Técnicas Híbridas

No trabalho de Fan et al. (2019) é proposto um algoritmo avançado para a detecção de buracos em estradas, combinando técnicas de visão computacional em 2D e 3D para melhorar a precisão e eficiência. Inicialmente, utiliza-se um mapa de disparidade denso

que é transformado para diferenciar áreas danificadas das não danificadas. A eficiência na transformação da disparidade é otimizada com a busca da seção áurea e programação dinâmica. Utiliza-se o método de limiarização de Otsu para identificar áreas potencialmente intactas, e modela-se estas áreas com uma superfície quadrática ajustada por mínimos quadrados, integrando a normal da superfície para aumentar a robustez. O consenso de amostra aleatória ajuda a minimizar o impacto de dados discrepantes. A detecção dos buracos é realizada pela comparação das disparidades reais e modeladas, e as nuvens de pontos dos buracos detectados são extraídas de uma superfície de estrada 3D reconstruída.

Em Xu (2019) foi desenvolvido um novo algorítmo para a detecção automática de buracos em estradas, combinando imagens e nuvem de pontos móvel. O processo envolveu três etapas principais: extração de buracos candidatos em 2D usando aprendizado profundo, extração em 3D através de uma nuvem de pontos, e confirmação dos buracos através de análise de profundidade. Utilizou-se o sistema DeepLabv3+ para identificar e classificar os buracos em imagens, e a nuvem de pontos foi empregada para definir a forma 3D e validar a presença do buraco.

2.3.1 Perspectivas da tecnologia baseada em técnicas híbridas

As técnicas híbridas representam uma abordagem inovadora para a detecção de problemas nas estradas. Uma das principais vantagens desses sistemas é a precisão aprimorada obtida pela combinação de múltiplas fontes de dados e tecnologias. Isso resulta em uma detecção de buracos mais precisa e confiável, contribuindo para uma melhor manutenção das estradas. Além disso, a resiliência é uma característica marcante das técnicas híbridas. Projetadas para serem robustas, essas soluções lidam eficazmente com falhas individuais e condições adversas, garantindo uma detecção confiável em diferentes cenários. Outra vantagem significativa é a capacidade de previsão proporcionada pela integração de aprendizado de máquina com outras tecnologias. Isso permite prever futuras deteriorações nas estradas, possibilitando a implementação de medidas preventivas e reduzindo custos a longo prazo.

No entanto, as técnicas híbridas também enfrentam desafios e desvantagens. A complexidade é um dos principais obstáculos. A integração de diferentes tecnologias aumenta a complexidade do sistema, exigindo uma infraestrutura de suporte mais sofisticada e conhecimento técnico especializado para operação e manutenção adequadas. Além disso, o custo pode ser uma preocupação significativa. A implementação de sistemas híbridos geralmente envolve investimentos substanciais devido à necessidade de múltiplos tipos de sensores e tecnologias avançadas. Por fim, a manutenção desses sistemas pode ser mais desafiadora e requer conhecimento especializado em diversas tecnologias, o que pode aumentar os custos operacionais a longo prazo.

Ao avaliar a viabilidade e eficácia das técnicas híbridas para o monitoramento de condições viárias, é essencial considerar cuidadosamente esses aspectos e buscar soluções que maximizem os benefícios e minimizem os desafios associados a essa abordagem.

2.4 Aplicativos Móveis e Crowdsourcing

A incorporação de aplicativos móveis e técnicas de *crowdsourcing* na detecção de buracos em vias públicas tem representado uma evolução significativa na gestão da infraestrutura urbana, essa metodologia de detecção de buracos e qualidade de rodovias, envolve a participação ativa da população.

No trabalho apresentado por Carrera et al. (2013) é proposto um modelo de aplicação que, fazendo uso dos sensores do celular do usuário e sob sua autorização, coleta, subliminarmente, dados de localização e vibração fornecidos pelo smartphone, que ao serem analisados na nuvem, permitem identificar a presença de buracos, lombadas e outras características de qualidade da via.

2.4.1 Perspectivas da tecnologia baseada em Aplicativos Móveis e Crowdsourcing

A natureza distribuída do crowdsourcing proporciona uma cobertura geográfica ampla e atualizações em tempo real sobre as condições das estradas. Isso não apenas fornece informações precisas, mas também possibilita uma resposta ágil a mudanças nas condições. Além disso, os custos são significativamente reduzidos ao envolver a comunidade na coleta de dados, tornando o processo mais acessível e escalável. Mais do que apenas coletar dados, essa abordagem promove o engajamento comunitário, incentivando os cidadãos a participarem ativamente da manutenção e melhoria das estradas em suas áreas locais.

Entretanto, há desafios a serem superados. A qualidade e veracidade dos dados podem ser comprometidas devido à natureza aberta do crowdsourcing, exigindo sistemas robustos de verificação para garantir a confiabilidade das informações coletadas. Além disso, a participação pode ser desigual, com áreas menos povoadas sendo potencialmente sub-representadas nos dados coletados. Questões de privacidade e segurança também precisam ser cuidadosamente consideradas ao lidar com informações pessoais dos usuários, garantindo que seus dados sejam protegidos adequadamente.

Ao considerar a aplicabilidade da tecnologia baseada em aplicativos móveis e crowdsourcing para o monitoramento de condições viárias, é essencial compreender os benefícios únicos dessa abordagem, ao mesmo tempo em que abordamos os desafios específicos que ela enfrenta. Ao fazê-lo, pode-se aproveitar ao máximo as vantagens oferecidas por essa tecnologia inovadora, implementando medidas para mitigar os desafios identificados e garantir sua eficácia e relevância no contexto do monitoramento de infraestrutura viária.

2.5 Sistemas Baseados em UAV (Veículos Aéreos Não Tripulados)

O uso de Sistemas Baseados em UAV, comumente conhecidos como *drones*, na detecção de buracos em vias públicas, representa um avanço tecnológico significativo na monitoração e manutenção de infraestruturas.

O trabalho de Furusho Becker et al. (2019) utiliza imagens capturadas por drones, e experimenta a utilização de Redes Neurais Convolucionais (CNNs), em particular o modelo Faster-RCNN Inception ResNet, ajustando parâmetros como o passo da caixa de

ancoragem e implementando aumento de imagem de modo à detectar buracos na imagem. De forma similar, o trabalho de B.H et al. (2018) propõe um sistema baseado em um drone de funcionamento autônomo, equipado com câmeras e equipamentos de geolocalização, que captura imagens aéreas das estradas e as processa utilizando um modelo de detecção de objetos YOLOv3 treinado para a detecção de buracos.

2.5.1 Perspectivas da tecnologia baseada em UAV

Os UAVs oferecem uma série de vantagens significativas para o monitoramento de estradas. Sua capacidade de sobrevoar áreas inacessíveis ou perigosas permite coletar dados de forma rápida e segura, superando as limitações das inspeções terrestres convencionais. Além disso, a alta resolução das imagens capturadas e os dados topográficos fornecem uma análise detalhada das condições das estradas, facilitando a identificação precisa de buracos e outras irregularidades. A flexibilidade operacional dos UAVs também é uma vantagem importante, pois podem ser rapidamente mobilizados e adaptados a diferentes condições e áreas, tornando-os ferramentas eficientes para o monitoramento contínuo.

No entanto, a adoção de UAVs para o monitoramento de estradas apresenta desafios específicos. As regulamentações e restrições de voo são uma preocupação, pois o uso de UAVs está sujeito a regulamentações estritas que variam conforme a região, afetando a flexibilidade operacional. Além disso, a análise dos dados coletados requer sistemas avançados de processamento e análise, representando um desafio técnico e financeiro. Os custos iniciais de aquisição e manutenção dos UAVs também podem ser consideráveis, apesar de sua eficiência operacional.

Ao avaliar a viabilidade e eficácia da tecnologia baseada em UAVs para o monitoramento de condições viárias, é crucial considerar tanto as vantagens quanto os desafios associados a essa abordagem, buscando soluções que maximizem os benefícios e minimizem os riscos.

2.6 Conclusão

Este capítulo proporcionou uma revisão abrangente das metodologias atuais empregadas na detecção de buracos em rodovias, explorando desde técnicas de visão computacional e sistemas baseados em sensores até abordagens híbridas, aplicações de crowdsourcing e a utilização de UAVs. O propósito foi esboçar o cenário tecnológico existente, delineando o contexto em que o presente trabalho se insere.

A contribuição deste presente estudo consiste na introdução de uma proposta inovadora baseada em aprendizado de máquina, utilizando modelos de redes neurais focados na detecção de buracos, com especial atenção ao processamento na borda. Essa abordagem enfatiza a otimização e a redução dos modelos para assegurar sua portabilidade e adequação ao processamento local, visando uma implementação de baixo custo, essencial para a viabilidade de aplicação em larga escala.

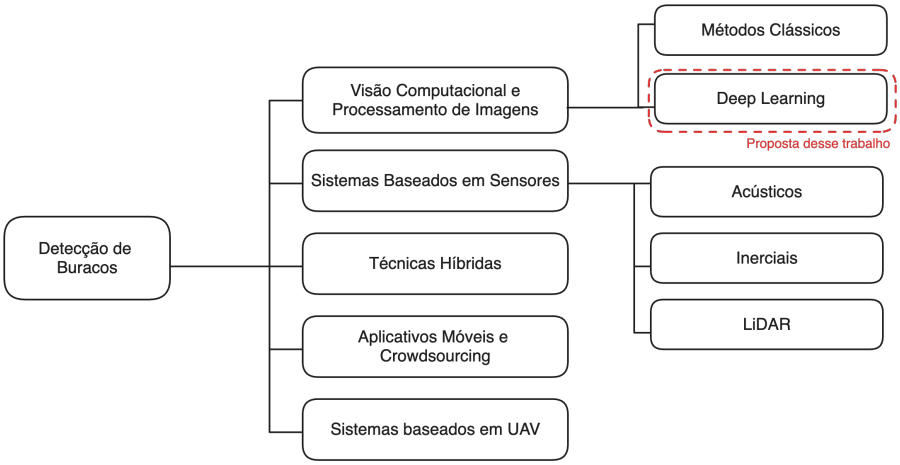

Figura 2.1: Fluxograma das principais metodologias para detecção de buracos com destaque para onde se encontra a proposta do presente trabalho
Fonte: Autoria própria

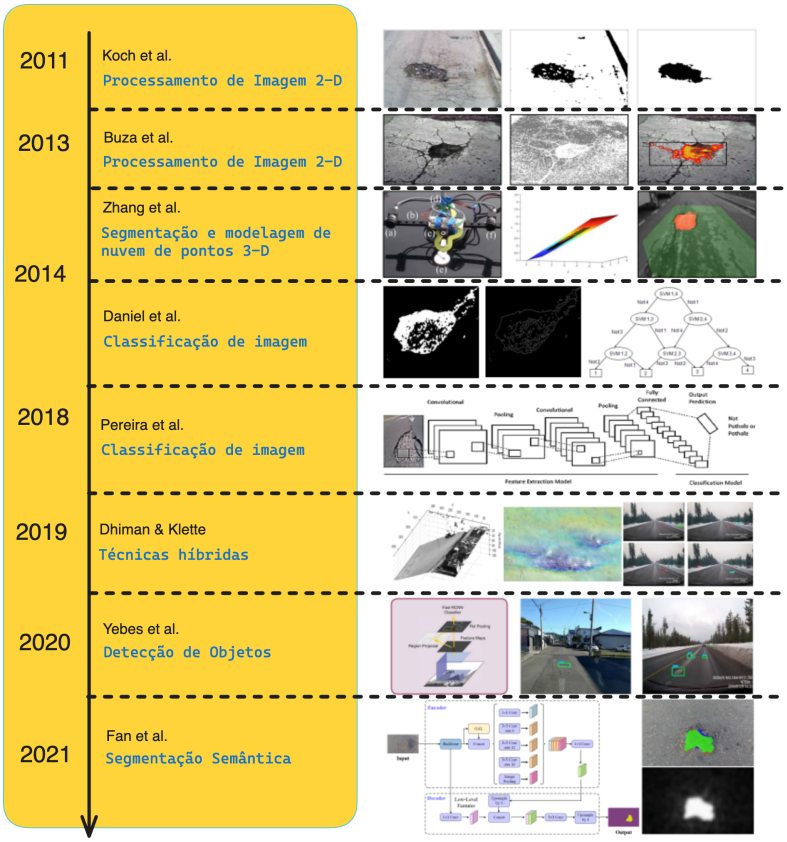

Figura 2.2: Os algoritmos de detecção de buracos em estradas mais representativos desenvolvidos entre 2011 e 2021
Fonte: Adaptado de Ma et al. (2022)

A figura 2.1 exibe um fluxograma das principais metodologias empregadas na detecção de buracos, salientando como as diversas técnicas revisadas se integram e destacando a posição da proposta atual neste contexto. Sob um panorama ainda mais amplo, a figura 2.2 elenca os principais algoritmos desenvolvidos para detecção de buracos de forma cronológica.

Capítulo 3

Fundamentação Teórica

Este capítulo desdobra a fundamentação teórica vital para nossa metodologia, centrando-se nas redes neurais convolucionais (CNNs) e suas expansões inovadoras. Inicia-se com as CNNs, essenciais para o processamento avançado de dados visuais, destacando-se em reconhecimento de imagens e detecção de objetos. A seguir, será detalhada a arquitetura YOLO (You Only Look Once), que reformula a detecção e classificação de objetos em uma operação rápida e integrada, ideal para aplicações em tempo real. Examinamos também a arquitetura FOMO (Faster Objects More Objects), focada na detecção ágil de múltiplos objetos em dispositivos de recursos muito limitados. Adicionalmente, será abordado o TinyML e a plataforma Edge Impulse, sublinhando sua significância para implementações de *machine learning* em dispositivos de capacidade reduzida. Este capítulo estabelece um elo entre os fundamentos teóricos e as aplicações práticas, preparando o terreno para a apresentação detalhada da metodologia proposta.

3.1 Redes Neurais Convolucionais

As Redes Neurais Convolucionais (CNNs, do inglês *Convolutional Neural Networks*) constituem uma subcategoria de redes neurais profundas otimizadas para análise de dados visuais, evidenciando-se por sua superioridade em tarefas de visão computacional como reconhecimento de imagens, detecção de objetos e análise de vídeos. Esta seção dedica-se ao estudo aprofundado das bases teóricas e funcionais das CNNs, estabelecendo um alicerce teórico robusto para a compreensão de arquiteturas complexas, incluindo a YOLO (*You Only Look Once*) e a FOMO (*Faster Objects More Objects*), cuja discussão será desenvolvida nas seções 3.2 e 3.3 respectivamente.

Inspiradas pela organização do córtex visual de mamíferos, onde neurônios específicos reagem a estímulos visuais distintos em diferentes regiões do campo visual, as CNNs mimetizam esta capacidade de resposta seletiva por meio de operações de convolução. Este paralelo biológico promoveu o desenvolvimento de modelos computacionais que, diferentemente das redes neurais tradicionais totalmente conectadas, preservam a integridade espacial dos dados de entrada. Esta característica é particularmente pertinente para o processamento de informações visuais, facilitando a análise e interpretação de imagens e vídeos com alta eficiência (Lecun et al. 1998, Goodfellow et al. 2016).

A essencialidade das CNNs reside em suas camadas de convolução, que empregam filtros (ou kernels) para extrair características específicas dos dados de entrada, como

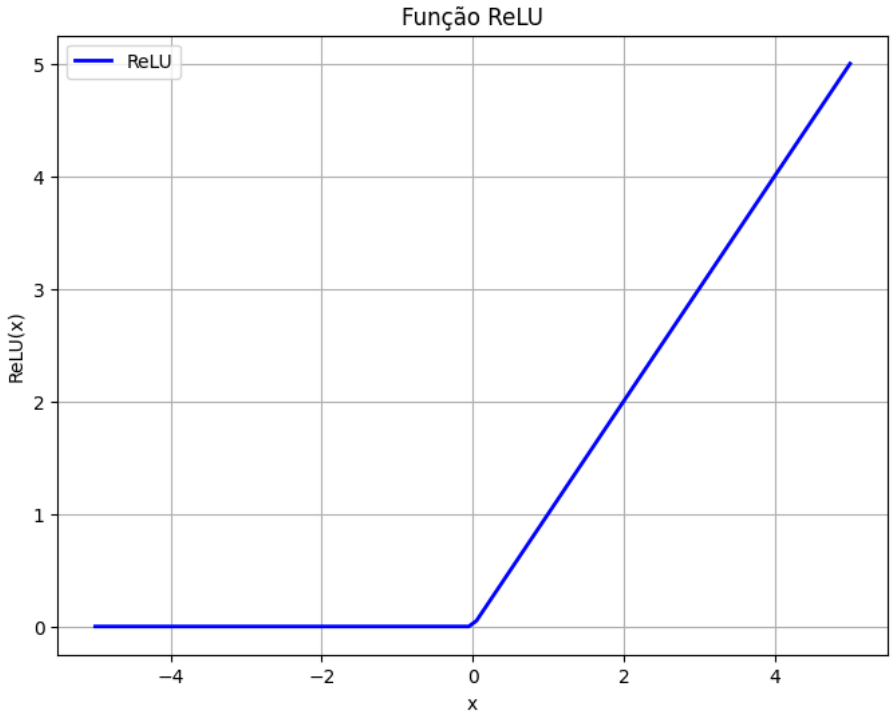

Figura 3.1: Gráfico da função ReLU
Fonte: Autoria própria

bordas, texturas e padrões. A operação de convolução, fundamento matemático destas redes, é descrita pela seguinte equação (Krizhevsky et al. 2012):

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n), \tag{3.1}$$

onde S denota o mapa de características resultante, I a imagem de entrada, e K o kernel ou filtro aplicado. Esta operação é sistematicamente realizada em toda a extensão da entrada para gerar um conjunto completo de mapas de características.

Subsequentemente, a aplicação de funções de ativação, tal como a ReLU (*Rectified Linear Unit*), introduz não-linearidades no modelo, permitindo a captura de relações complexas nos dados (Nair & Hinton 2010). A função ReLU é expressa matematicamente por:

$$f(x) = \max(0,x), \tag{3.2}$$

facilitando a propagação de valores positivos enquanto anula negativos, contribuindo assim para a eficácia e eficiência do aprendizado da rede.

A operação de *pooling*, especificamente o *max pooling*, é utilizada para reduzir a dimensionalidade espacial dos mapas de características, simplificando a estrutura da rede ao diminuir o volume de parâmetros e computações necessárias. A operação de *max pooling* é matematicamente representada por:

$$P(i,j) = \max_{a,b \in W} I(a,b), \tag{3.3}$$

onde W é uma janela definida no mapa de características I, e P é o mapa resultante após a aplicação do *pooling*.

Após o processamento por múltiplas camadas de convolução e *pooling*, as CNNs incorporam camadas totalmente conectadas, que sintetizam as características aprendidas para executar tarefas específicas, como a classificação de objetos em imagens (Lecun et al. 1998).

A aplicação de CNNs transcende a simples classificação de imagens, abrangendo a detecção e classificação de múltiplos objetos dentro de um campo visual, demonstrando assim sua versatilidade e capacidade de adaptação a diversos contextos de visão computacional (Krizhevsky et al. 2012).

A capacidade intrínseca das CNNs de aprender e representar hierarquias complexas de características visuais, através da combinação estratégica de operações de convolução, *pooling* e ativação não-linear, as torna ferramentas poderosas para a análise efetiva e precisa de dados visuais. A seção subsequente focará na arquitetura YOLO, apresentando-a como uma inovação metodológica na detecção de objetos.

3.2 Arquitetura YOLO: You Only Look Once

A arquitetura YOLO (*You Only Look Once*) representa um marco na evolução das técnicas de detecção de objetos, introduzindo uma abordagem inovadora que contrasta com os métodos tradicionais baseados em classificadores ou localizadores aplicados de forma sequencial sobre diferentes regiões de uma imagem. Esta arquitetura, desenvolvida por Redmon et al. (2016), integra-se ao corpo de conhecimento sobre redes neurais convolucionais (CNNs), conforme discutido na seção 3.1, aproveitando-se de sua capacidade de extrair características visuais complexas para realizar tanto a localização quanto a classificação de objetos em uma única etapa de inferência.

A inovação fundamental da YOLO reside em sua capacidade de analisar a imagem completa (daí o nome *"You Only Look Once"*) para prever as classes dos objetos e suas localizações simultaneamente. Este método difere significativamente das abordagens anteriores, que tipicamente analisam múltiplas regiões da imagem de forma independente, resultando em uma eficiência computacional substancialmente menor. A YOLO, ao integrar a detecção e a classificação em um único passo, maximiza a eficiência do processamento, tornando-se particularmente adequada para aplicações em tempo real (Redmon et al. 2016).

Esta abordagem unificada é especialmente pertinente para a detecção de buracos em imagens, uma aplicação crítica em campos como a robótica autônoma e a manutenção preditiva. A capacidade da YOLO de processar imagens de forma rápida e eficaz permite a identificação de irregularidades e defeitos em superfícies ou estruturas, facilitando intervenções corretivas com precisão e agilidade.

A aplicabilidade da YOLO na detecção de buracos beneficia-se diretamente das características aprendidas pelas CNNs, que incluem bordas, texturas e padrões geométricos. Essas características são essenciais para diferenciar buracos e defeitos de outros elemen-

tos visuais numa imagem. A otimização dos parâmetros da rede, através de um conjunto de dados adequadamente anotado com a localização e classificação de buracos, permite à YOLO desenvolver uma compreensão espacial que facilita a identificação precisa dessas anomalias.

$$\text{Precisão}_{\text{detecção}} = \frac{TP}{TP+FP}, \tag{3.4}$$

onde:

- TP representa os Verdadeiros Positivos (*True Positives*),
- FP representa os Falsos Positivos (*False Positives*).

A precisão na detecção de buracos, medida pela proporção de verdadeiros positivos sobre a soma de verdadeiros positivos e falsos positivos, emerge como um indicador chave da eficácia da YOLO nesta aplicação específica. A otimização contínua da rede, por meio do ajuste de seus parâmetros e da seleção criteriosa de características relevantes, é fundamental para maximizar essa precisão. Em resumo, a arquitetura YOLO estabelece-se como uma poderosa ferramenta na detecção de objetos, incluindo buracos em imagens, alavancando a eficiência e a precisão proporcionadas pelas redes neurais convolucionais.

3.3 Arquitetura FOMO: Faster Objects More Objects

O modelo FOMO (*Faster objects, more objects*) do inglês "Objetos mais rápidos, mais Objetos"trata-se de um algoritmo de aprendizado de máquina, projetado pelos engenheiros da Edge Impulse (2024) para possibilitar a detecção de objetos em dispositivos com recursos extremamente limitados. Esse método permite a contagem de objetos, a identificação de suas localizações aproximadas em uma imagem e o rastreamento de múltiplos objetos em tempo real, utilizando até 30 vezes menos capacidade de processamento e memória em comparação com os modelos MobileNet SSD ou YOLOv5. Este algoritmo é particularmente significativo para aprimorar a eficiência e a funcionalidade em sistemas embarcados ou dispositivos IoT (do inglês, Internet das Coisas), onde o poder de computação e o espaço de armazenamento são restritos (Edge Impulse 2024).

A arquitetura do modelo FOMO, concebida para a detecção de objetos em dispositivos com capacidades computacionais limitadas, é uma inovação derivada da adaptação da arquitetura MobileNetV2 desenvolvida por Sandler et al. (2019). Este modelo utiliza blocos residuais para processar imagens através de transformações que intensificam a extração de características, como partes do rosto de um animal, por exemplo. Na configuração do FOMO, contudo, a arquitetura original da MobileNetV2 é simplificada pela interrupção do processo após o terceiro bloco residual. Esta modificação estratégica permite a criação de classificadores dedicados a segmentos menores da imagem, referidos como células, dentro dos mapas de características resultantes deste ponto de corte. A figura 3.2 ilustra a arquitetura simplificada da MobileNet V2.

Na prática, o FOMO opera com uma imagem de entrada que, dependendo da resolução escolhida (por exemplo, 240x240), é subdividida através de múltiplas reduções até formar células de 30x30, se considerarmos a entrada de 240x240. Cada elemento dentro desses mapas de características - ou cada pixel, em termos simplificados - é então analisado

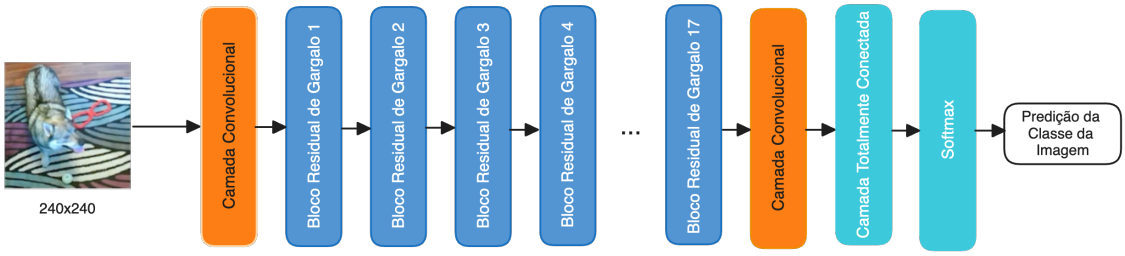

Figura 3.2: Arquitetura simplificada da MobileNET V2
Fonte: Adaptado de Edge Impulse (2023)

individualmente. A figura 3.3 apresenta a arquitetura do FOMO, evidenciando como a estrutura original da MobileNet V2 é adaptada para segmentar e processar essas células.

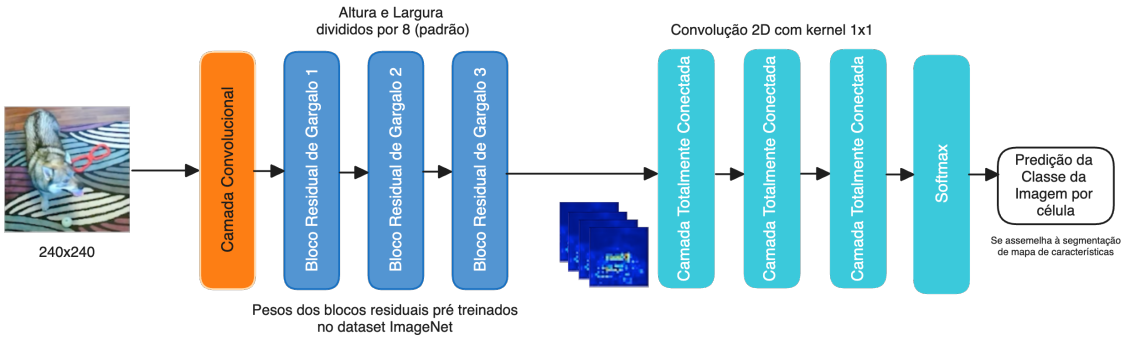

Figura 3.3: Arquitetura simplificada da FOMO
Fonte: Adaptado de Edge Impulse (2023)

A abordagem de segmentação empregada pelo FOMO transforma cada uma dessas células em uma unidade de análise distinta, sobre a qual se aplicam camadas completamente conectadas através de convoluções 2D com núcleos de 1x1, simulando a matemática de uma rede neural densa.

Este processo resulta numa forma de segmentação que não apenas identifica a classe de objeto presente em cada célula (como fundo, bola, cachorro ou brinquedo), mas também localiza aproximadamente esses objetos dentro da imagem. Este método de classificação granular é ideal para aplicações em sistemas onde os recursos são limitados, como microcontroladores, permitindo a identificação e localização de objetos sem necessidade de informações adicionais complexas, como caixas delimitadoras precisas. A figura 3.4 ilustra um exemplo de imagem de entrada subdividida em elementos (mapas de características) com as probabilidades das classes exemplificadas, fundo, bola, cachorro ou brinquedo.

Além disso, a arquitetura do FOMO incorpora técnicas para aprimorar a precisão da localização dos objetos, como a eliminação de células adjacentes com pontuações inferiores. Essa abordagem é particularmente útil em cenários onde objetos similares aparecem próximos uns dos outros, como demonstrado no caso prático da contagem de abelhas, em Palm (2018), por um engenheiro que desenvolveu o FOMO para a plataforma Edge

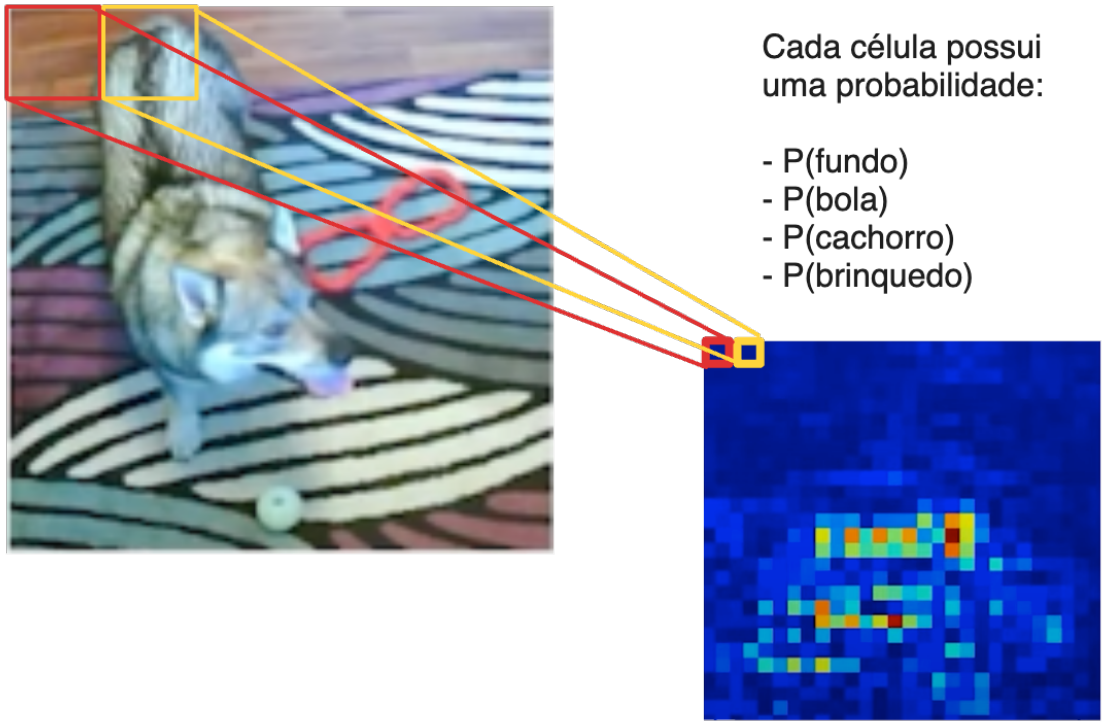

Figura 3.4: Exemplo de classificação por células da FOMO
Fonte: Adaptado de Edge Impulse (2023)

Impulse. Este caso ilustra a aplicabilidade do modelo em condições reais, onde a câmera, posicionada lateralmente, capta imagens de abelhas que entram e saem de uma colmeia. O FOMO processa essas imagens para estimar o número de abelhas, adaptando-se perfeitamente às limitações de processamento de um microcontrolador, o que é crítico para operações contínuas e autônomas em ambientes rurais. O sistema foi implementado em um microcontrolador Cortex-M7 operando a 480MHz, capaz de processar vídeos a 30fps com uma resolução de 240x240 e apenas 245kB de RAM, demonstrando a eficiência do FOMO em ambientes computacionalmente restritos.

Em Impulse (2023) o FOMO é apresentado sendo aplicado em um projeto que envolve a contagem de parafusos e arruelas em uma esteira transportadora, evidenciando sua versatilidade e capacidade de funcionamento em dispositivos extremamente limitados em termos de capacidade de processamento e energia. Neste cenário, o modelo não apenas conta os itens com eficácia, mas também envia esses dados em tempo real para uma plataforma IoT, permitindo monitoramento e controle contínuos, fundamentais para processos industriais e de manufatura.

Finalmente, o modelo FOMO, surge como uma possibilidade promissora para a detecção de irregularidades em estradas, operando eficientemente em microcontroladores. Esta aplicabilidade permite o uso em dispositivos móveis ou embarcados, como drones ou veículos de manutenção, facilitando a monitorização contínua das condições das vias em tempo real.

Explorar a utilização do FOMO em contextos de infraestrutura rodoviária poderia não apenas melhorar a manutenção e segurança das estradas, mas também demonstrar o impacto transformador da inteligência artificial em aplicações práticas de engenharia, destacando a necessidade de investigações futuras sobre sua eficácia e viabilidade.

3.4 TinyML

O TinyML (do inglês, aprendizado de máquina minúsculo) emerge como um campo interdisciplinar no cruzamento entre aprendizado de máquina, hardware e software, visando a implementação de modelos de aprendizado profundo em dispositivos embarcados, como microcontroladores. Este avanço tecnológico é impulsionado pela necessidade de processamento local de dados em dispositivos de borda, que operam com recursos limitados de energia e capacidade computacional. A filosofia do TinyML é fundamentada na execução eficiente de tarefas de inferência em sistemas embarcados, onde o custo, o espaço e o consumo energético são restritivos (Yelchuri 2022).

TinyML também pode ser visto como a interseção entre Aprendizado de Máquinas e Sistemas Embarcados, ilustrando como esses campos se complementam para otimizar o processamento em dispositivos de borda. Esse relacionamento é claramente delineado na figura 3.5, destacando a integração e aplicação prática dessas tecnologias.

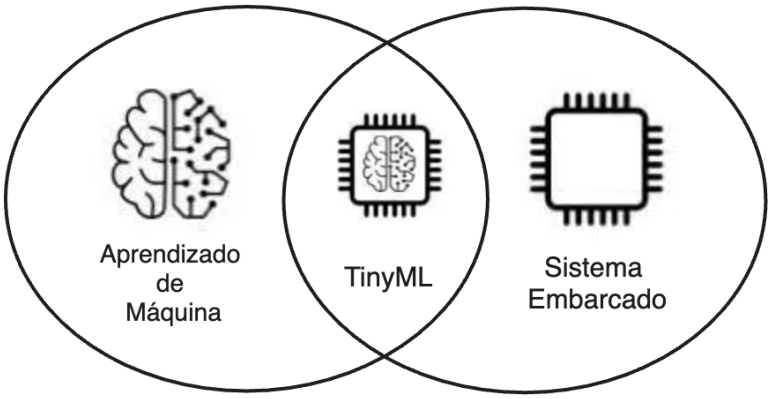

Figura 3.5: TinyML: Intercessão de áreas do conhecimento
Fonte: Autoria própria

As aplicações do TinyML são diversas e impactam múltiplos setores, incluindo saúde, segurança, manutenção industrial e, principalmente, IoT. Em ambientes onde a privacidade dos dados, a rapidez na tomada de decisões e a eficiência energética são críticas, o TinyML se apresenta como uma solução ideal. O processamento local de dados elimina a necessidade de transmissão constante para a nuvem, reduzindo latências e protegendo informações sensíveis. Com isso, dispositivos de borda podem operar de maneira autônoma, executando tarefas de reconhecimento de padrões e processamento sensorial sem dependências externas (Yelchuri 2022).

Apesar de suas promessas, a implementação de TinyML enfrenta desafios significativos relacionados à limitação de recursos. Os principais obstáculos incluem a capacidade

de processamento reduzida, a memória limitada e a necessidade de otimização intensiva dos modelos de aprendizado de máquina para que sejam compatíveis com os microcontroladores. Além disso, a heterogeneidade do hardware em que o TinyML pode ser implementado exige soluções personalizadas, o que pode complicar o desenvolvimento e a padronização de aplicações (Han & Siebert 2022).

A detecção de buracos é uma aplicação prática onde o TinyML pode ter um impacto significativo. Utilizando sensores integrados em veículos ou implantados ao longo de estradas, sistemas baseados em TinyML podem processar dados de forma local para identificar irregularidades no pavimento em tempo real. Esta aplicação não só melhora a segurança dos motoristas e passageiros, mas também auxilia os órgãos de gestão de tráfego e manutenção urbana a receberem informações precisas e atualizadas sobre as condições das vias.

Em suma, o TinyML se configura como uma tecnologia revolucionária para a implementação de inteligência artificial em dispositivos de borda. Sua aplicação na detecção de buracos é apenas um exemplo de como essa tecnologia pode transformar setores inteiros, tornando-os mais eficientes, seguros e proativos. Conforme o TinyML continua a evoluir, espera-se uma ampliação ainda maior de suas aplicações, melhorando significativamente a interação entre seres humanos e máquinas no ambiente físico.

3.5 A Plataforma Edge Impulse

A Plataforma Edge Impulse emerge como uma plataforma integral dedicada ao desenvolvimento e à implantação de modelos de machine learning em dispositivos de borda, evidenciando-se por sua abordagem completa e sistemática em todo o ciclo de vida de modelos de ML, como observa-se na figura 3.6 (Edge Impulse 2023).

A plataforma oferece funcionalidades essenciais como pré-processamento de dados, treinamento de modelos, avaliação de desempenho e compilação para dispositivos específicos (Edge Impulse, 2021). Esses recursos permitem uma adaptação e otimização dos modelos para operar sob restrições de recursos, fundamentais em aplicações de Internet das Coisas (IoT) e contextos similares onde eficiência de processamento e consumo de energia são críticos.

A interface baseada em blocos da Edge Impulse descomplica o processo de desenvolvimento de ML, tornando a experimentação acessível até para usuários com limitado conhecimento técnico em aprendizado de máquina. Esta abordagem facilita a visualização e a manipulação dos componentes do modelo, incentivando uma interação mais intuitiva e produtiva com a plataforma.

Apesar das vantagens, a versão gratuita da plataforma impõe restrições como um limite de tempo de treinamento de até 20 minutos, o que pode ser insuficiente para projetos que exigem sessões de treinamento mais extensas e intensivas. Essa limitação pode necessitar a migração para planos pagos, que oferecem capacidades de treinamento ampliadas.

Portanto, considerando suas funcionalidades robustas, a Edge Impulse foi escolhida para o desenvolvimento do modelo de detecção de buracos proposto neste trabalho. A escolha dessa plataforma é motivada pela sua capacidade de processamento eficiente em dispositivos de borda, sua interface intuitiva que facilita a experimentação e desenvolvimento de modelos, e o suporte abrangente para análise e teste de desempenho em condi-

Figura 3.6: Ciclo da plataforma Edge Impulse
Fonte: Autoria própria

ções reais. Esses atributos tornam a Edge Impulse uma escolha valiosa para abordar as complexidades associadas ao desenvolvimento de modelos de machine learning destinados a operar em ambientes dinâmicos e com recursos limitados.

Capítulo 4

Metodologia Proposta

Este capítulo descreve a metodologia utilizada nesta pesquisa, abordando desde a seleção e uso do Conjunto de Dados até as especificações da Arquitetura Proposta. Serão detalhados os métodos de Treinamento de Modelos, com ênfase nas arquiteturas YOLO e FOMO, e as Configurações Experimentais, que incluem escolhas de Hardware e técnicas de Quantização de Modelos. Esta seção visa estabelecer uma base sólida para a implementação e validação dos modelos desenvolvidos.

4.1 Conjunto de Dados

Na construção de modelos de inteligência artificial, especialmente em aplicações de visão computacional, a qualidade e a representatividade do conjunto de dados são fundamentais para garantir a eficácia e a precisão dos algoritmos de detecção de objetos. Modelos de aprendizado profundo, que formam a base das técnicas mais avançadas em visão computacional, dependem notavelmente da diversidade e da abrangência dos dados fornecidos durante o treinamento. Essa dependência ocorre porque esses modelos aprendem a identificar padrões e características essenciais ao problema em questão, como exposto no Capítulo 3, através do processo de otimização que minimiza uma função de custo, geralmente representada como $L(\theta)$, onde θ são os parâmetros do modelo. No desenvolvimento deste trabalho, foi empregado um conjunto de dados de domínio público (vide Figura 4.1 e 4.2) consistindo em imagens de buracos em vias, acompanhadas de anotações em forma de caixas delimitadoras, para o treinamento dos modelos. A seleção desse conjunto de dados é estratégica, visto que a representatividade e a variabilidade das condições rodoviárias capturadas nas imagens são essenciais para que o modelo de detecção de objetos possa operar eficazmente em um ambiente real. O conjunto de dados, Aegis (2022), obtido através da plataforma Roboflow, é composto por 1.388 imagens registradas sob diversas condições rodoviárias.

Para maximizar o desempenho do modelo e sua capacidade de generalização, foi procedida a divisão das imagens em uma proporção de 80:20 para treinamento e validação, respectivamente. Essa divisão, $n_{train} : n_{val} = 0.8n : 0.2n$ onde n representa o número total de imagens, é uma prática comum em aprendizado de máquina para equilibrar a necessidade de aprender com a maior quantidade de dados possível, enquanto se reserva uma parcela significativa para a validação independente do aprendizado.

Figura 4.1: Exemplo de imagens do dataset utilizado
Fonte: Autoria própria.

4.2 Treinamento de Modelos

Uma vez adquirido, limpo e ajustado às dimensões de entrada de 320x320 pixels, o conjunto de dados foi utilizado para o treinamento dos modelos propostos. O modelo YOLO foi treinado utilizando a linguagem Python juntamente com a biblioteca Ultralytics, que permite a implementação do treinamento com poucas linhas de código. Enquanto que o modelo FOMO foi treinado na plataforma Edge Impulse, destacando-se pela sua eficiência e adequação para o processamento de imagens em dispositivos de borda.

4.2.1 YOLO

Para o treinamento do modelo YOLO, foi utilizada a arquitetura YOLOv8-nano, menor arquitetura YOLOv8, que possui 3,2 milhões de parâmetros e realiza 8,7 bilhões de operações de ponto flutuante por segundo (FLOPS). Este modelo foi inicialmente carregado com pesos pré-treinados no conjunto de dados COCO (Common Objects in Context – Objetos Comuns em Contexto) desenvolvido por Lin et al. (2015). Os hiperparâmetros foram configurados conforme especificado no apêndice A. O treinamento foi conduzido durante 200 épocas, sendo uma época definida como uma passagem completa pelo conjunto de dados de treinamento. Além disso, foi utilizado um mecanismo de autobatch para otimizar dinamicamente o tamanho do batch size, adaptando-se às capacidades de memória da GPU NVIDIA T4 (14.7 GB). Esta GPU foi disponibilizada através do Goo-

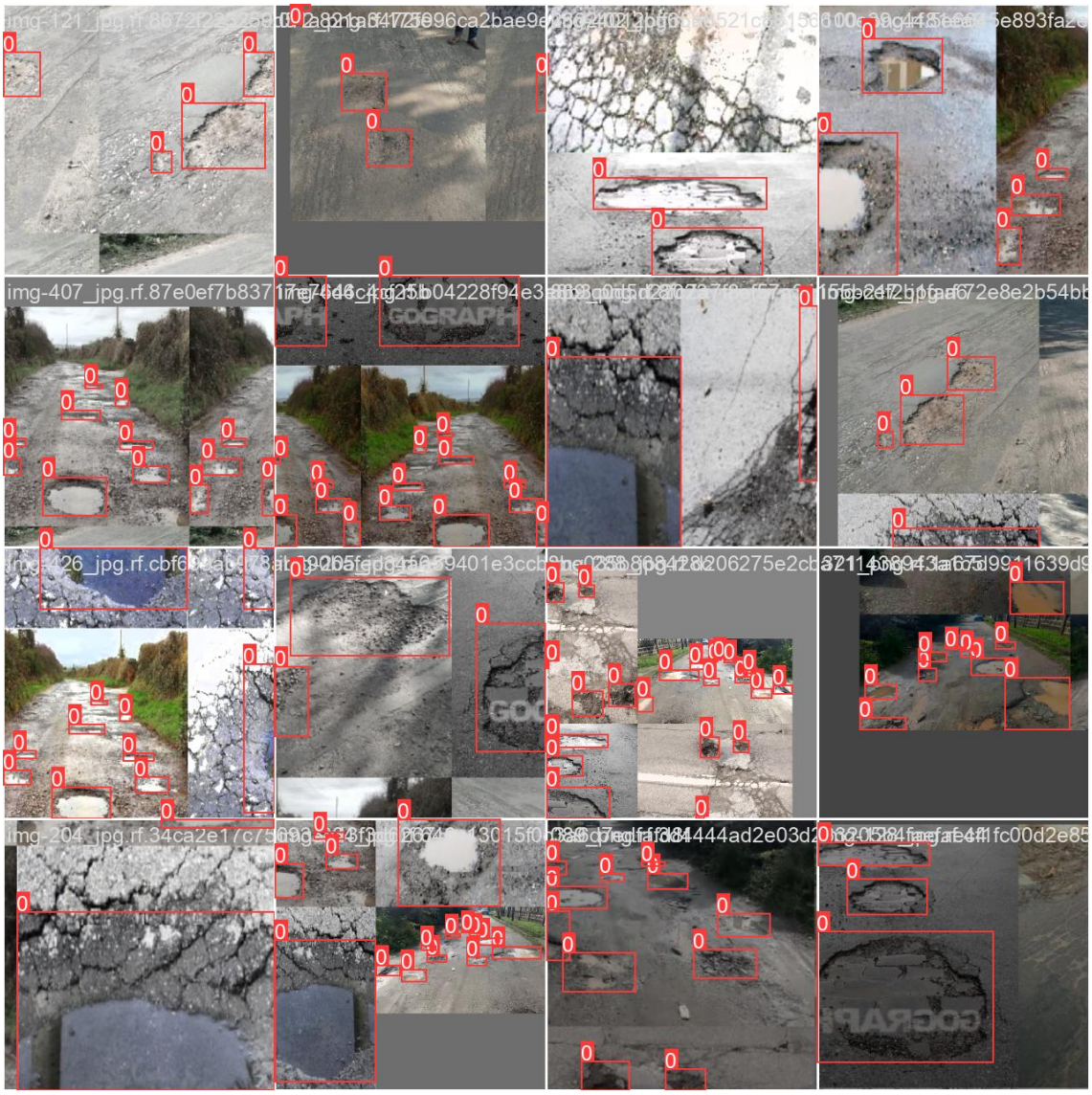

Figura 4.2: Exemplo do dataset anotado, organizado para treinamento
Fonte: Autoria própria.

gle Colab, uma plataforma da Google que fornece acesso a máquinas virtuais, com Jupyter Notebook, configuráveis para computação na nuvem, facilitando assim o acesso a recursos computacionais avançados.

Na figura 4.3 pode-se observar como, com poucas linhas de código em Python, o treinamento do modelo YOLOv8n é executado.

O arquivo `data_pothole.yaml` é crucial para iniciar o treinamento de modelos, configurando as especificações do conjunto de dados, incluindo a classe anotada *pothole*. Este arquivo especifica os diretórios `image` e `label` usados no treinamento e na validação, com imagens armazenadas como `.jpg` e rótulos como `.txt`, detalhando a classe e as coordenadas de *bounding box* no formato YOLO, essenciais para o modelo aprender a localização dos objetos.

```python
from ultralytics import YOLO

# Carrega o modelo
model = YOLO('yolov8n.pt')

# Treina o modelo
model.train(data= 'data_pothole.yaml', epochs=20, batch = -1, imgsz=320, plots=True)
```

Figura 4.3: Código em Python para treinamento do modelo YOLO
Fonte: Autoria própria

4.2.2 FOMO

Para o treinamento do modelo FOMO, empregou-se a plataforma Edge Impulse, que integra um módulo de "data acquisition". As imagens do conjunto de dados, já anotadas com bounding boxes, foram carregadas neste módulo, conforme ilustrado na figura 4.5. Estas foram divididas entre treinamento e validação na proporção de 80:20. Incluiu-se também um arquivo .json com todas as anotações em bounding boxes. Embora o modelo FOMO retorne apenas os centroides dos objetos, é necessário que o treinamento seja realizado com um dataset anotado com bounding boxes completas.

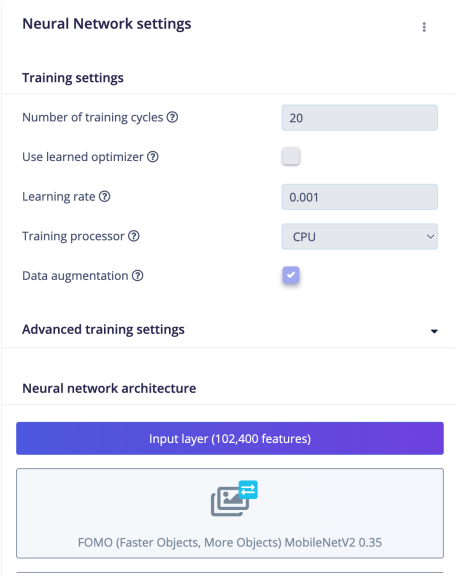

Figura 4.4: Módulo detecção de objetos.
Fonte: Edge Impulse, 2024

Segue-se então à etapa de "impulse design", figura 4.6 na qual os dados passam por uma série de pré-processamentos. Nele as imagens são redimensionadas para as dimensões especificadas, e procede-se à definição do tipo de dados, da tarefa a ser realizada e

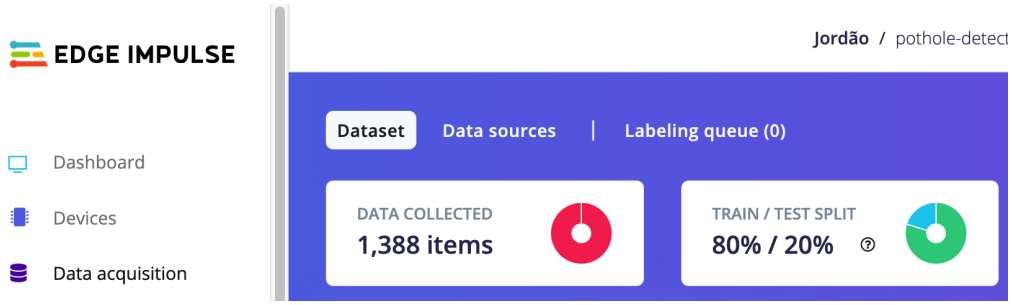

Figura 4.5: Módulo de aquisicão de dados.
Fonte: Edge Impulse, 2024

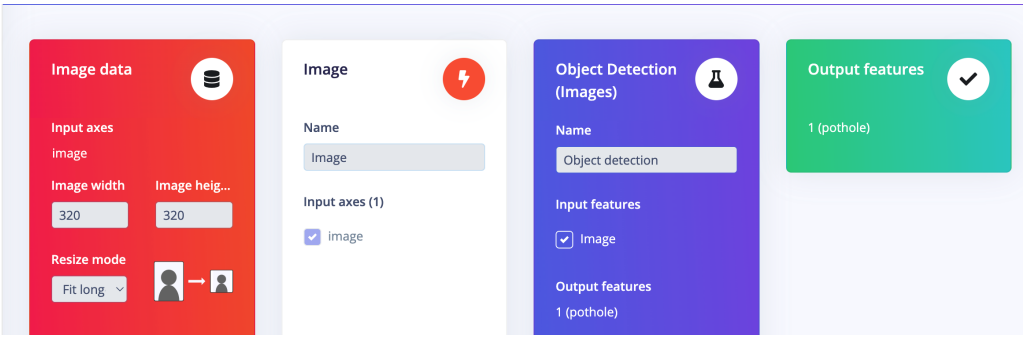

Figura 4.6: Módulo de impulse design.
Fonte: Edge Impulse, 2024

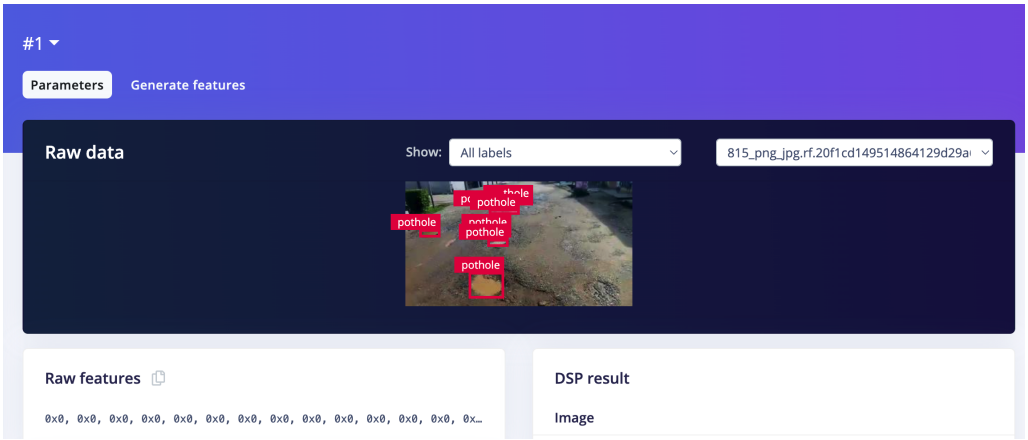

Figura 4.7: Módulo image.
Fonte: Edge Impulse, 2024

das classes correspondentes. No módulo "image", figura 4.7, os dados são normalizados, convertendo os valores de cada canal dos pixels para um intervalo flutuante entre 0 e 1. No módulo de detecção de objetos, figura 4.4, define-se o modelo FOMO para treinamento, estabelecendo-se 20 épocas, uma taxa de aprendizagem de 0.001 e a CPU como dispositivo de treinamento. O processo de treinamento é executado por meio do framework

TensorFlow, em um container Docker especialmente configurado para tal fim.

4.3 Configurações Experimentais

Esta seção introduz os dispositivos de hardware escolhidos para a implementação dos modelos desenvolvidos e explora o processo de quantização de modelos. Este último é crucial para aprimorar a eficiência e o desempenho, permitindo que os modelos operem de maneira mais eficaz em ambientes de hardware restritos.

4.3.1 Hardware

Para a arquitetura proposta neste projeto, optou-se pela seleção de três plataformas de hardware distintas, com base em suas características e capacidades de processamento na borda. A escolha destes dispositivos visa explorar diferentes níveis de capacidade computacional e eficiência energética, adequados para a implementação de modelos de inteligência artificial em aplicações de *edge computing*.

O primeiro dispositivo escolhido foi a NVIDIA Jetson Orin Nano, que integra uma CPU de 6 núcleos Arm® Cortex®-A78AE v8.2 e uma GPU com arquitetura NVIDIA Ampere de 512 núcleos, incluindo 16 núcleos tensoriais. Este hardware destaca-se por sua alta capacidade de processamento e otimização para modelos de redes neurais, apesar de seu custo relativamente mais alto e dimensões compactas. A capacidade integrada de processamento de GPU permite a realização de operações intensivas em paralelo, o que é ideal para aplicações que requerem alto desempenho em processamento gráfico e computacional.

Em seguida, selecionou-se o Raspberry Pi 4, equipado com uma CPU ARMv7 Cortex-A72 de 1.5 GHz. Diferente da NVIDIA Jetson, o Raspberry Pi foca no processamento baseado em CPU, oferecendo uma solução de custo-benefício atrativo e um desempenho robusto para uma ampla gama de aplicações em computação na borda. Sua arquitetura permite que seja utilizado em projetos que demandam uma boa capacidade de processamento, mas que não necessitam das capacidades avançadas de uma GPU.

Por último, escolheu-se o MCU Arduino Portenta H7, que possui um microcontrolador STM32H747XI com um núcleo Cortex-M7 rodando a 480 MHz. Este dispositivo é notável por suas dimensões extremamente reduzidas e seu baixo consumo de energia, tornando-o ideal para aplicações onde o espaço físico e a eficiência energética são críticos. O Arduino Portenta H7 é adequado para tarefas de processamento menos intensivas, permitindo ainda a integração em sistemas embarcados onde a simplicidade e a eficácia são essenciais.

A combinação destes três hardwares possibilita uma abordagem versátil e escalável para a implementação de soluções em *edge computing*, adaptando-se a diferentes necessidades de processamento e eficiência operacional em variados cenários de aplicação.

4.3.2 Quantização de Modelos

Na busca por otimizar o desempenho dos modelos de inteligência artificial em dispositivos de hardware na borda, a quantização emerge como uma técnica essencial. A quantização visa reduzir a precisão computacional dos modelos, diminuindo o tamanho

do modelo e acelerando a inferência, sem comprometer significativamente a precisão. Este processo envolve converter os tipos de dados de ponto flutuante, usados durante o treinamento, para tipos de dados de menor precisão ou inteiros, utilizados durante a execução do modelo.

A precisão computacional dos modelos pode variar amplamente. Por exemplo, um modelo pode ser treinado com precisão de ponto flutuante de 32 bits (FP32), mas para aplicações em dispositivos com recursos limitados, a quantização para 8 bits inteiros (INT8) pode ser mais apropriada. Matematicamente, isto pode ser representado como:

$$q(w) = \text{round}\left(\frac{w - \min(W)}{\max(W) - \min(W)} \times 255\right) \tag{4.1}$$

onde w é um peso específico e W representa o conjunto de todos os pesos no modelo.

Considerando o modelo YOLO, treinado utilizando a biblioteca PyTorch na plataforma Ultralytics, este inicialmente está no formato `.pt`. O formato `.pt` (PyTorch) encapsula tanto a arquitetura do modelo quanto os pesos treinados em um único arquivo, utilizando tensores PyTorch e serialização binária. Após o treinamento, o modelo é submetido a um processo de quantização usando ferramentas da própria Ultralytics, onde é primeiro convertido para o formato ONNX (Open Neural Network Exchange), que é um formato interoperável que permite que o modelo seja usado em várias plataformas e frameworks.

Após a conversão para ONNX, o modelo é então convertido para TFLite FP32, mantendo a precisão de ponto flutuante mas em um formato otimizado para dispositivos móveis e embarcados oferecido pelo TensorFlow Lite. Finalmente, o modelo é quantizado para TFLite INT8, o que reduz significativamente o tamanho do modelo e melhora a velocidade de inferência, tornando-o adequado para execução em tempo real em dispositivos com recursos limitados.

Similarmente, o modelo FOMO, que originalmente está no formato Keras (`.h5`), segue um caminho de conversão semelhante. Após ser exportado para o formato TFLite, ele também é disponibilizado em versões FP32 e posteriormente em INT8, seguindo o processo de quantização para adequação ao uso em dispositivos com restrições de capacidade de processamento e memória.

Após a obtenção da versão final quantizada, o modelo é então compilado especificamente para o hardware alvo, utilizando as ferramentas apropriadas para garantir que a execução seja otimizada para aproveitar ao máximo as capacidades do dispositivo. Este passo é crucial para garantir que o desempenho seja maximizado, oferecendo a melhor resposta possível dentro das limitações do hardware.

Esses processos de conversão e quantização, ilustrados na figura 4.8, implicam considerações computacionais importantes incluindo a escolha do grau de quantização que equilibra a eficiência computacional com a precisão do modelo. Esse equilíbrio é fundamental porque, embora a quantização reduza o uso de memória e acelere o processamento, também pode levar a uma perda de precisão do modelo. Portanto, é crucial determinar o nível de quantização adequado que maximiza o desempenho sem comprometer a eficácia do modelo em suas aplicações práticas. Este *trade-off* é especialmente crítico em aplicações de computação na borda, onde as capacidades de processamento e memória são

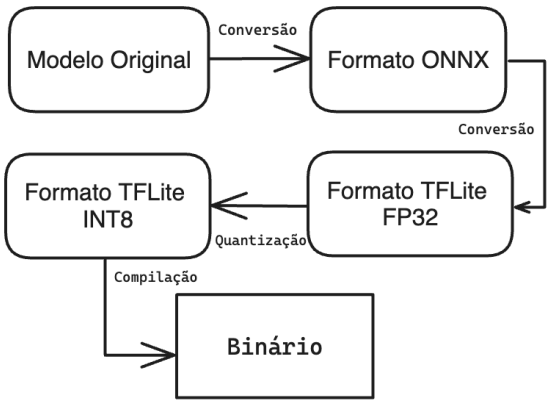

Figura 4.8: Processo de conversão e quantização dos modelos
Fonte: Autoria própria

limitadas, mas a precisão e a velocidade de resposta são essenciais para a funcionalidade do sistema.

4.4 Arquitetura Proposta

A arquitetura proposta define um framework metodológico para a detecção de buracos em rodovias usando inteligência artificial, projetado para oferecer uma abstração flexível e adaptável do sistema de hardware e modelos de IA. Este sistema permite a intercambialidade desses componentes, facilitando a customização de acordo com os requisitos específicos de preço, acurácia e desempenho.

Essa abstração sistêmica é fundamentada em uma estrutura modular, onde diferentes dispositivos de hardware podem ser selecionados e acoplados a uma variedade de modelos de IA otimizados. A otimização é alcançada por meio de técnicas como a quantização, adaptando os modelos para operar eficientemente nos recursos disponíveis em cada dispositivo escolhido. Os três dispositivos utilizados nos experimentos variam em custo e complexidade de processamento, buscando demonstrar a capacidade do framework de atender a diversas necessidades operacionais.

O processo desenvolvido para os modelos segue uma sequência bem definida que abstrai a complexidade subjacente: Aquisição de Imagens, Segregação de Dados, Treinamento dos Modelos, Conversão dos Modelos, Avaliação dos Modelos e Implantação, conforme ilustrado na figura 4.9. Esta abordagem sistematizada garante não só a eficácia na detecção de buracos, mas também a viabilidade econômica e o desempenho adequado nos dispositivos alvo.

Para o sistema em si, propõe-se uma representação de alto nível na Figura 5.1, que inclui uma câmera como dispositivo responsável pela captura de dados a serem processados. A figura também ilustra três dispositivos de borda, refletindo os hardwares utilizados nos experimentos, que demonstram a capacidade de intercambialidade do sistema. Esses dispositivos são conectados à nuvem através de uma API em um servidor, que facilita o envio de ocorrências detectadas e, potencialmente, permite receber atualizações automáticas de modelos por meio de atualizações *Over-the-Air* (OTA). OTA, que significa

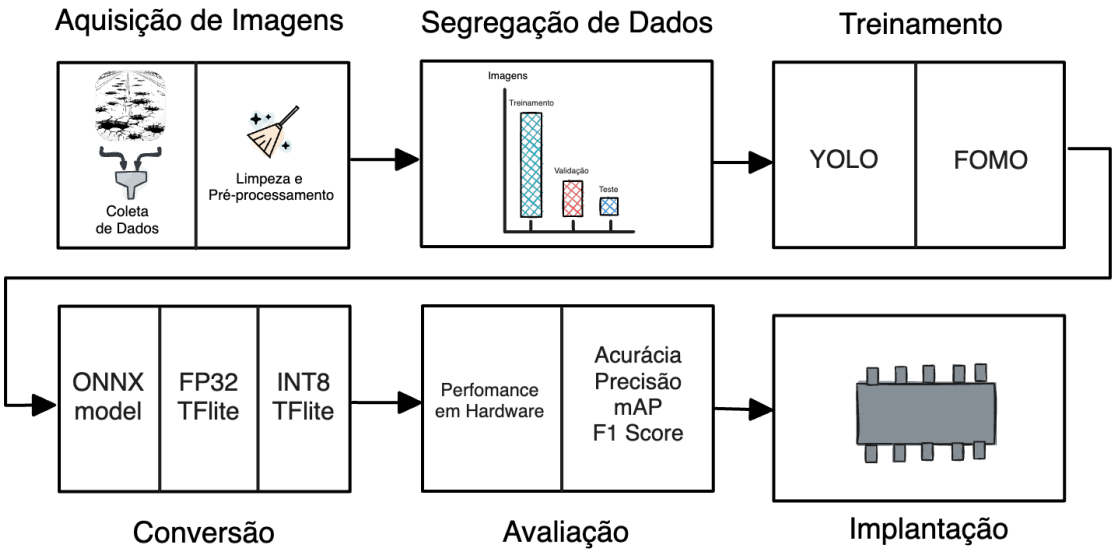

Figura 4.9: Fluxo de trabalho até a implantação do modelo
Fonte: Adaptado de Da Silva et al. (2023)

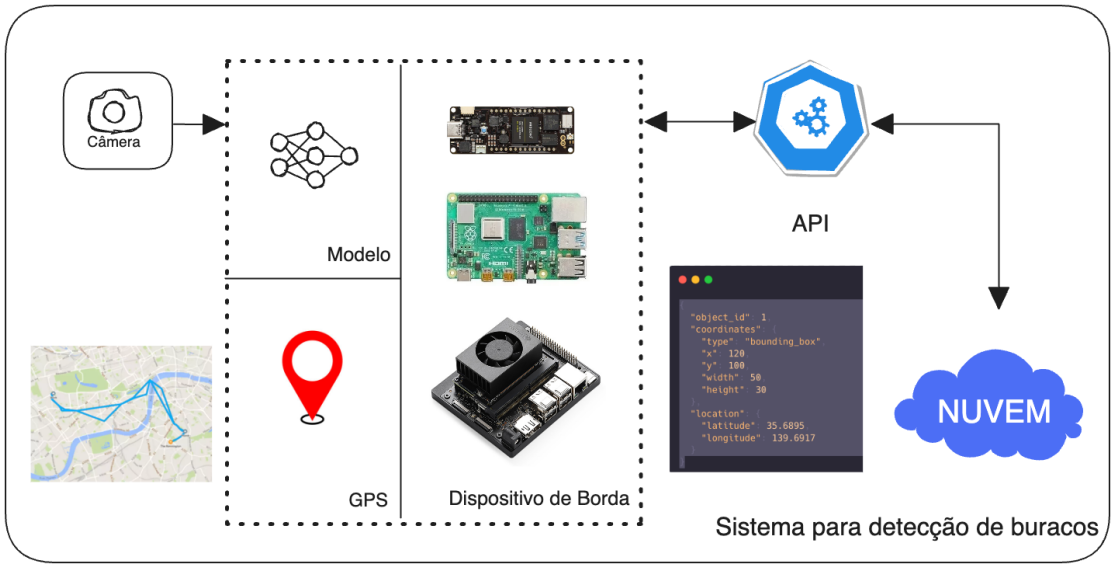

Figura 4.10: Representação em alto nível do sistema de detecção de buracos
Fonte: Autoria própria

"Atualizações via Rede", refere-se ao processo de enviar atualizações de software, configurações ou modelos de inteligência artificial diretamente para dispositivos operacionais no campo, sem necessidade de conexão física ou intervenção manual.

Assim, a proposta não apenas atende às necessidades atuais, mas também estabelece uma plataforma escalável e adaptável que promove um ciclo contínuo de inovação e melhoria na aplicação de inteligência artificial para a manutenção de infraestruturas viárias.

Esta estratégia garante que o sistema possa rapidamente adaptar-se e melhorar, respondendo eficazmente às evoluções tecnológicas e às alterações nas condições das vias.

4.5 Implementação

A implementação dos modelos para o sistema de detecção automática de buracos foi concebida utilizando-se o módulo de *Deployment* da plataforma Edge Impulse. Esse módulo prepara e otimiza os modelos de *machine learning* — desenvolvidos e treinados na própria plataforma ou importados após o treinamento — para simulação de execução em dispositivos de borda específicos. A otimização utiliza o *EON Compiler*, adaptando os modelos para uma avaliação virtual em uma variedade de hardwares.

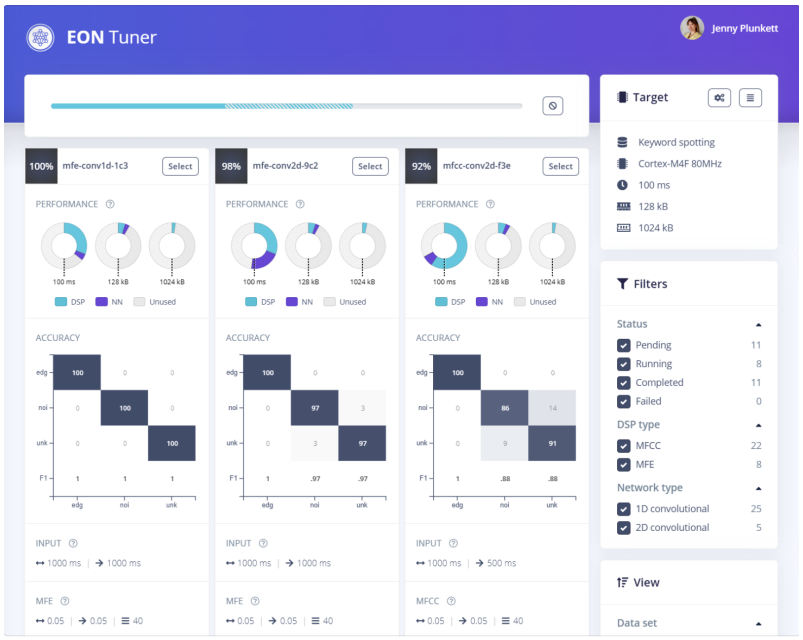

Figura 4.11: Exemplo da interface do EON Tuner na plataforma Edge Impulse
Fonte: Edge Impulse, 2024

Utilizando o *EON Tuner*, figura 4.11 a simulação do desempenho do modelo é realizada em diferentes dispositivos de borda. Este processo permite uma análise detalhada do comportamento do modelo em dispositivos como Arduino Portenta H7, Nvidia Jetson Orin Nano e Raspberry Pi 4, sem necessitar da presença física dos hardwares. A simulação engloba a geração de binários de *firmware*, arquivos de bibliotecas otimizadas e a integração com sistemas operacionais específicos. Assim, a plataforma proporciona uma visão realística e ajustada do desempenho esperado do modelo em diversos contextos operacionais.

Por exemplo, para o Arduino Portenta H7, o simulador compila um binário de *firmware ready-to-go* que integra todos os componentes necessários para a operação imediata. Para a Nvidia Jetson Orin Nano, é gerado um arquivo da biblioteca TensorRT, otimizado para a GPU do dispositivo, enquanto para o Raspberry Pi 4, cria-se uma biblioteca C++ integrada ao Edge Impulse CLI.

A funcionalidade do *Deployment* na plataforma permite a virtualização do processo de envio e configuração do modelo otimizado, simulando a implementação direta nos dispositivos configurados. Isso maximiza a eficiência operacional e simplifica o processo de validação e instalação do sistema de detecção de buracos, explorando as capacidades específicas de cada hardware escolhido para otimizar o desempenho do sistema.

Capítulo 5

Experimentos e Resultados

Nesta seção serão apresentados o experimento realizado e os resultados obtidos. A análise compreenderá tanto os aspectos visuais, por meio de inferências realizadas com os diferentes modelos de detecção de buracos, quanto o desempenho desses modelos em hardware específico. Os resultados serão exibidos em sequência e discutidos detalhadamente, proporcionando uma compreensão abrangente do impacto das escolhas técnicas e tecnológicas adotadas neste estudo.

5.1 Desempenho dos Modelos

Para avaliar o desempenho dos modelos de detecção de buracos obtidos, adotou-se a métrica F1 Score, uma medida que combina precisão e sensibilidade em uma única métrica harmônica. A precisão (precision) indica a proporção de identificações positivas que foram corretamente classificadas como buracos, enquanto a sensibilidade (recall) reflete a proporção de buracos reais corretamente detectados pelo modelo. O F1 Score é útil para avaliar a eficácia do modelo na detecção precisa da única classe de interesse: buracos.

Os resultados obtidos são apresentados na Tabela 5.1, compreendendo os F1 Scores calculados após o treinamento e validação dos modelos. Utilizou-se o mesmo conjunto de validação para ambos, o que permite uma comparação objetiva do desempenho dos modelos. A uniformidade do dataset empregado no treinamento de ambos os modelos assegura uma análise comparativa coerente.

É importante notar que o tamanho do lote de processamento (batch size) utilizado no treinamento dos modelos não foi uniforme, o que pode influenciar os resultados. As variações no batch size podem alterar quais imagens são processadas em cada época de treinamento, afetando diretamente o processo de aprendizagem do modelo.

Modelo	**F1 Score (%)**
FOMO	82.6
YOLOv8n	84.6

Tabela 5.1: F1 Scores dos modelos FOMO e YOLOv8n para o tamanho de entrada 320x320.

Fonte: Autoria própria

Classe Real	Buraco Predito	Fundo Predito
Buraco	100%	0%
Fundo	20.8%	79.2%

Tabela 5.2: Matriz de Confusão para o Modelo FOMO
Fonte: Autoria própria

Classe Real	Buraco Predito	Fundo Predito
Buraco	86%	14%
Fundo	0%	100%

Tabela 5.3: Matriz de Confusão para o Modelo YOLOv8n
Fonte: Autoria própria

Além disso, uma diferença significativa entre os modelos YOLO e FOMO deve ser considerada. O modelo YOLO fornece a localização exata dos buracos por meio de bounding boxes, enquanto o modelo FOMO, conforme detalhado na seção 3.3, identifica apenas o centroide do objeto. Isso pode levar o modelo FOMO a generalizar dois objetos próximos como um único, detectando apenas um centroide, mesmo quando treinado com imagens anotadas com bounding boxes precisas. Este aspecto é crucial para entender as variações nos resultados e na comparação da eficácia dos modelos.

Nas tabelas 5.2 e 5.3, as matrizes de confusão para os modelos FOMO e YOLO revelam diferenças notáveis na taxa de falsos positivos entre eles, ilustrando comportamentos distintos na detecção de objetos. Embora as matrizes de confusão sejam frequentemente associadas a modelos de classificação, sua aplicação em modelos de detecção de objetos, como neste estudo, permite avaliar a eficácia com que esses modelos diferenciam a presença e a ausência de objetos específicos, neste caso, buracos.

O modelo FOMO registrou uma taxa de falsos positivos de 20.8%, indicando que aproximadamente um quinto das áreas marcadas como contendo buracos eram, na verdade, fundo. Essa taxa elevada de superdetecção sugere que o modelo FOMO é propenso a errar no lado da cautela, preferindo marcar potenciais não-buracos como buracos, o que poderia ser vantajoso em aplicações onde a não detecção de um buraco real pudesse ser grave.

Por outro lado, o modelo YOLO apresentou uma taxa de falsos positivos de 14%, que, embora menor que a do FOMO, ainda representa uma significativa quantidade de identificações incorretas de buracos. Esta taxa mais baixa pode refletir uma melhor capacidade do YOLO em discriminar entre buracos e fundo, ajustando-se de forma mais eficaz à variabilidade das imagens.

A avaliação desses modelos por meio de matrizes de confusão é crucial para entender o equilíbrio entre sensibilidade e especificidade que cada modelo alcança. A sensibilidade é fundamental para assegurar que todos os buracos reais sejam detectados, enquanto a especificidade minimiza a marcação errônea de áreas sem buracos como contendo buracos.

É importante destacar que, embora os dados da matriz de confusão forneçam uma vi-

são valiosa do desempenho dos modelos, essas informações pressupõem a precisão das anotações no conjunto de validação. Considerando que as anotações são realizadas por humanos, podem existir erros que influenciem a interpretação dos resultados. Essa possibilidade sublinha a necessidade de revisões cuidadosas das anotações e de considerar margens de erro ao avaliar a eficácia dos modelos de detecção.

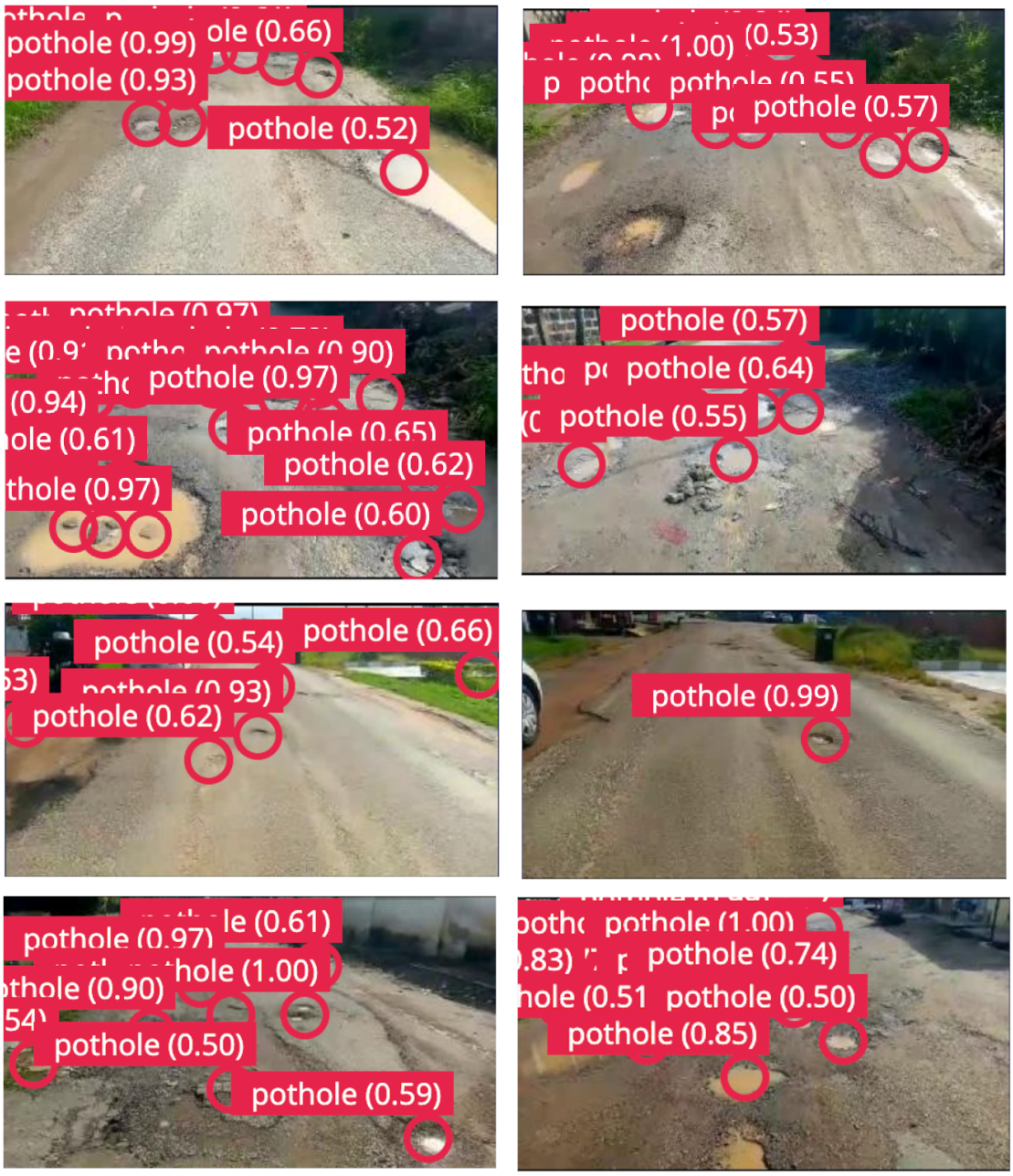

Figura 5.1: Predições de validação com modelo FOMO após treinamento
Fonte: Autoria própria

As figuras 5.1 e 5.2 ilustram os resultados de predição, com rótulos ‘pothole’ (buraco

Figura 5.2: Predições de validação com modelo YOLOv8n após treinamento
Fonte: Autoria própria

em inglês) e o grau de confiança, realizados pelos modelos, pós-quantização, em um conjunto de 8 imagens do trecho de validação do dataset. Ambos os modelos, FOMO e YOLO, foram aplicados às mesmas 8 imagens, que foram selecionadas por conter buracos considerados de difícil detecção devido à qualidade não uniforme da via.

Na figura 5.1, os resultados para o modelo FOMO são apresentados. O modelo FOMO utiliza um círculo para representar o centroide do objeto detectado. Foi observada a de-

tecção de buracos em todas as imagens, no entanto, em algumas delas ocorreram falsos negativos. Um exemplo é visível na primeira imagem da segunda coluna, onde um buraco presente na imagem não foi marcado com o centroide. Além disso, na segunda imagem da primeira coluna, um conjunto de pedras dentro de um grande buraco foi erroneamente identificado como vários buracos menores.

Por outro lado, na figura 5.2, observamos as detecções realizadas pelo modelo YOLO, que emprega bounding boxes para indicar a localização exata dos buracos. Este método demonstrou maior precisão, evitando a generalização de múltiplos buracos como um único buraco. Contudo, ainda foi possível notar uma falha de detecção, como na quarta imagem da primeira coluna, onde um buraco não foi identificado pelo YOLO, apesar de ter sido detectado pelo modelo FOMO na Figura 5.1.

Uma análise visual comparativa revela que, embora o modelo FOMO seja pelo menos 30 vezes mais leve que o modelo YOLO, ele consegue realizar a detecção de maneira satisfatória, mesmo com precisão reduzida. Portanto, dependendo do grau de exigência da tarefa, o modelo FOMO pode ser uma opção viável devido à sua eficiência e menor requisito de recursos computacionais.

Esses resultados sublinham a importância de selecionar o modelo apropriado com base nas características específicas do ambiente de aplicação, além de destacar as limitações inerentes de cada abordagem tecnológica na detecção de buracos em condições desafiadoras.

5.2 Desempenho em Hardware

Tabela 5.4: Tempos de inferência para os modelos FOMO e YOLOv8n em diferentes dispositivos

Dispositivo	Modelo	FP32 (ms)	INT8 (ms)	FP32 (fps)	INT8 (fps)
Raspberry Pi 4	FOMO	32	27	31.25	37.04
	YOLOv8n	243	166	4.12	6.02
Jetson Orin Nano	FOMO	5	1	200.00	>900
	YOLOv8n	14	5	71	200
Arduino Portenta H7	FOMO	1442	626	0.69	1.60
	FOMO (128x128)	190	67	5.26	14.93
	YOLOv8n	-	-	-	-

Fonte: Autoria própria

O desempenho em hardware dos modelos propostos para a detecção de buracos foi avaliado considerando o tempo de resposta, medido como o inverso da taxa de frames por segundo (fps), além do uso de RAM e armazenamento FLASH nos dispositivos. Os resultados, apresentados na Tabela 5.4 e 5.5, refletem os desempenhos em duas precisões numéricas, FP32 e INT8, em três dispositivos distintos.

No dispositivo Raspberry Pi 4, o modelo FOMO alcançou um excelente desempenho, com taxas superiores a 30fps em ambas as precisões, sugerindo a capacidade de realizar

Modelo	Tamanho FP32 (MB)	Tamanho INT8 (MB)
YOLOv8n	12	3.1
FOMO	2.4	0.09
FOMO 128x128	0.6	0.07

Tabela 5.5: Tamanhos dos modelos YOLOv8n, FOMO e FOMO 128x128
Fonte: Autoria própria

inferências em tempo real sem perda significativa de frames, adequado para câmeras que operam a partir de 23fps. Em contraste, o modelo YOLOv8n atingiu aproximadamente 4fps em FP32 e 6fps após a quantização. Embora esta taxa esteja abaixo do ideal para captura de vídeo em tempo real, dependendo da prioridade dada à precisão das bounding boxes, o modelo ainda pode ser útil em aplicações menos exigentes em termos de tempo real.

No Jetson Orin Nano, ambos os modelos demonstraram altas taxas de fps, beneficiando-se do poder computacional superior do dispositivo. Nesse cenário, o uso do modelo YOLOv8n é particularmente vantajoso, dado sua maior precisão e acurácia, como discutido na seção anterior. Essa combinação de alto desempenho e e detalhamento nas detecções faz do YOLOv8n uma escolha robusta para aplicações que exigem um alto grau de confiabilidade.

No Arduino Portenta H7, o modelo YOLOv8n foi excluído da avaliação devido à insuficiência de recursos do hardware, particularmente em termos de RAM e FLASH. Para o modelo FOMO, foi necessário ajustar o tamanho da entrada de 320x320 para 128x128 para viabilizar a operação, resultando numa taxa de 14fps após quantização. Essa modificação, embora permita a execução em tempo real, pode comprometer a capacidade do modelo de capturar detalhes finos em imagens complexas, o que é uma consideração crítica para aplicações onde a detecção precisa de buracos é essencial.

Essa análise de desempenho em hardware não só sublinha as capacidades e limitações dos modelos em ambientes operacionais diversos, mas também enfatiza a importância de escolher o hardware adequado ao perfil de exigência da aplicação de detecção de buracos.

5.3 Avaliação de custo

Após a avaliação do desempenho dos modelos e do hardware, uma análise do custo dos dispositivos se faz necessária para determinar a opção mais viável economicamente. A tabela 5.6 apresenta os preços dos três dispositivos avaliados.

O Raspberry Pi 4 emerge como uma opção particularmente atraente devido ao seu custo baixo em comparação com os demais dispositivos. Embora o Jetson Orin Nano ofereça capacidades de processamento superiores, seu custo elevado pode ser uma barreira, especialmente em projetos onde o desempenho extra não se traduz em benefícios proporcionais. O Raspberry Pi 4, por outro lado, não apenas se adapta bem a muitas aplicações de detecção de buracos em tempo real devido ao seu desempenho adequado, mas também representa uma excelente relação custo-benefício, sendo ideal para projetos com orçamento restrito ou para implementações em larga escala. Contudo, ao selecionar

Dispositivo	Preço (USD)
Raspberry Pi 4	61
Jetson Orin Nano	270
Arduino Portenta H7	113

Tabela 5.6: Preço aproximado dos dispositivos em dólares
Fonte: Autoria própria

um dispositivo, fatores como suporte técnico, a comunidade de usuários, facilidade de integração e durabilidade também devem ser levados em conta. Esses aspectos asseguram que a escolha do hardware seja sustentável a longo prazo e alinhada com as exigências específicas do projeto.

Capítulo 6

Conclusão

Este trabalho teve como objetivo primordial desenvolver uma proposta para detecção de buracos em vias urbanas, utilizando tecnologias de computação na borda. Esta necessidade surge em resposta à crescente disponibilidade de hardwares com capacidade de processamento adequada para executar modelos de inteligência artificial que, anteriormente, eram considerados demasiadamente complexos para tais aplicações. A relevância de empregar tais tecnologias para abordar problemas concretos e prementes na sociedade é evidente e foi um dos pilares deste estudo.

No decorrer da pesquisa, foram implementados e avaliados dois modelos de estado da arte para a detecção de objetos. O primeiro, o modelo YOLO, é amplamente conhecido por sua precisão e complexidade, embora demande significativos recursos computacionais. O segundo, adaptado de uma aplicação originalmente diferente, destacou-se pela sua eficiência operacional em plataformas com capacidades computacionais limitadas, característica essencial para aplicações em TinyML. Ambos os modelos demonstraram desempenho satisfatório, alcançando métricas de acurácia consideráveis, especialmente quando se leva em conta o volume e a natureza do conjunto de dados utilizado. A facilidade de conversão dos modelos para implementação nos hardwares selecionados foi amplamente facilitada pelo uso da ferramenta Edge Impulse, que se provou extremamente valiosa e eficiente.

Foram explorados três diferentes hardwares, variando em tamanho, capacidade computacional e custo, oferecendo uma gama de opções dependendo das prioridades e requisitos específicos da aplicação pretendida. Isso introduziu um dilema significativo quanto ao custo-benefício na implementação de soluções urbanas para detecção de buracos. A escolha entre utilizar modelos mais precisos e potentes, como o YOLO, em hardwares mais robustos e caros instalados em um número menor de veículos, versus a adoção de modelos menos precisos, porém menos computacionalmente intensos e mais economicamente viáveis, como o FOMO, que podem ser implementados em um número maior de veículos, apresenta uma questão estratégica crucial. Esta decisão influencia diretamente a capilaridade e a eficácia da detecção de irregularidades nas vias.

Este estudo, portanto, não apenas forneceu direcionamentos valiosos sobre a implementação de tecnologias de detecção de buracos em ambiente urbano, mas também abriu caminho para futuras investigações. Encoraja-se a realização de pesquisas adicionais para explorar mais profundamente essas questões estratégicas e para avançar no desenvolvimento e na implementação de soluções práticas em cenários reais. Espera-se que as fun-

dações estabelecidas por esta pesquisa inspirem e informem futuras iniciativas visando a melhoria contínua da infraestrutura urbana.

6.1 Trabalhos Futuros

Diante das diversas avenidas para investigações futuras que podem expandir significativamente o entendimento e a aplicabilidade dos sistemas de detecção de buracos em ambiente urbano, segue-se algumas direções promissoras:

- **Avaliação em Condições Reais de Tráfego**: Um trabalho futuro relevante seria a implementação e teste dos modelos YOLO e FOMO em veículos operando em condições reais de tráfego urbano. Isso incluiria a coleta de dados em tempo real e a análise da eficácia dos modelos em diferentes condições ambientais e de tráfego. A implementação prática ajudaria a identificar desafios não observados em condições controladas e ajustar os modelos para maximizar sua eficiência e precisão no mundo real.
- **Desenvolvimento de Modelos Híbridos**: Outra área de interesse seria o desenvolvimento e teste de modelos híbridos que combinem as características de alta precisão do YOLO com a eficiência de execução do FOMO. Tal abordagem poderia resultar em um modelo robusto, capaz de operar eficazmente em hardware de baixo custo e baixo consumo energético, mantendo uma precisão aceitável para a detecção prática de buracos em vias urbanas.
- **Exploração de LSTM para Otimização Temporal**: Explorar a utilização de LSTM (Long Short-Term Memory) na otimização dos modelos de detecção de buracos, considerando que a sensibilidade à detecção de buracos aumenta ao identificar um buraco, elevando a probabilidade de encontrar outros na mesma estrada. Esta abordagem pode aproveitar o contexto temporal para melhorar a eficácia dos modelos.
- **Estudos de Impacto Econômico e Social**: Seria igualmente valioso conduzir estudos sobre o impacto econômico e social da implementação de sistemas de detecção de buracos em larga escala. Isso incluiria análise de custo-benefício, considerando os custos de implementação dos diferentes modelos e hardwares, e o impacto resultante na redução de danos a veículos e na segurança do tráfego. Esses estudos poderiam fornecer diretrizes claras para autoridades urbanas sobre as opções mais eficazes e eficientes economicamente para melhorar a infraestrutura rodoviária.

6.2 Publicações

Por fim, durante a consolidação da proposta deste trabalho, houve uma publicação intitulada **"TinyML-Based Pothole Detection: A Comparative Analysis of YOLO and FOMO Model Performance"** no congresso internacional *2023 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*. O estudo abordou uma avaliação comparativa dos modelos YOLOv5, YOLOv8 e FOMO para detecção de buracos usando a abordagem Tiny Machine Learning (TinyML).

- J. Da Silva, T. Flores, S. Júnior and I. Silva, “TinyML-Based Pothole Detection: A Comparative Analysis of YOLO and FOMO Model Performance,” 2023 IEEE

Latin American Conference on Computational Intelligence (LA-CCI), Recife-Pe, Brazil, 2023, pp. 1-6, doi: 10.1109/LA-CCI58595.2023.10409357.

Referências Bibliográficas

Aegis (2022), 'Conjunto de dados de detecção de buracos [conjunto de dados]', Roboflow Universe. Disponível em: `https://universe.roboflow.com/aegis/pothole-detection-i00zy`. [Acessado em: 8 de abril de 2023].

B.H, HemaMalini, Akshay Padesur, Manoj V & Atish Shet (2018), 'Detection of potholes on roads using a drone', *EAI Endorsed Transactions on Energy Web* p. 171546.
URL: *http://dx.doi.org/10.4108/eai.19-10-2021.171546*

Bourechak, Amira, Ouarda Zedadra, Mohamed Nadjib Kouahla, Antonio Guerrieri, Hamid Seridi & Giancarlo Fortino (2023), 'At the confluence of artificial intelligence and edge computing in iot-based applications: A review and new perspectives', *Sensors* **23**(3).
URL: *https://www.mdpi.com/1424-8220/23/3/1639*

Carrera, F., S. Guerin & J. B. Thorp (2013), 'By the people, for the people: The crowdsourcing of "streetbump": An automatic pothole mapping app', *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* **XL-4/W1**, 19–23.
URL: *https://isprs-archives.copernicus.org/articles/XL-4-W1/19/2013/*

Christchurch: The pothole capital of New Zealand (2023), `https://www.stuff.co.nz/the-press/news/100847641/christchurch-the-pothole-capital-of-new-zealand`. [Online; accessed May 27, 2023].

Da Silva, Jordão, Thommas Flores, Silvan Júnior & Ivanovitch Silva (2023), Tinyml-based pothole detection: A comparative analysis of yolo and fomo model performance, *em* '2023 IEEE Latin American Conference on Computational Intelligence (LA-CCI)', pp. 1–6.

Dhoundiyal, Parveen, Vikrant Sharma & Satvik Vats (2023), Deep learning framework for automated pothole detection, *em* '2023 International Conference on Sustainable Communication Networks and Application (ICSCNA)', pp. 1382–1387.

Edge Impulse (2023), 'Edge impulse: The leading development platform for machine learning on edge devices', `https://www.edgeimpulse.com/`. Acessado em: 14 de abril de 2023.

Edge Impulse (2024), 'Fomo: Object detection for constrained devices', https://docs.edgeimpulse.com/docs/edge-impulse-studio/learning-blocks/object-detection/fomo-object-detection-for-constrained-devices. Acessado em 03 de março de 2024.

Fan, Rui, Umar Ozgunalp, Brett Hosking, Ming Liu & Ioannis Pitas (2019), 'Pothole detection based on disparity transformation and road surface modeling', *IEEE Transactions on Image Processing* **29**, 897–908.

Fan, Rui, Umut Ozgunalp, Billy Hosking, Ming Liu & Ioannis Pitas (2020), 'Pothole detection based on disparity transformation and road surface modeling', *IEEE Transactions on Image Processing* **29**(1), 897–908.

Fernandes, Anita, Mateus Cassaniga, Bianka Passos, Eros Comunello, Stefano Frizzo Stefenon & Valderi Leithardt (2023), 'Detection and classification of cracks and potholes in road images using texture descriptors', *Journal of Intelligent and Fuzzy Systems* **44**, 10255–10274.

Furusho Becker, Yuri V., Henrique Lopes Siqueira, Edson Takashi Matsubara, Wesley Nunes Gonçalves & Jose Marcato Marcato (2019), Asphalt pothole detection in uav images using convolutional neural networks, *em* 'IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium', pp. 56–58.

Gonzalez, Rafael C & Richard E Woods (2018), *Digital image processing*.

Goodfellow, Ian, Yoshua Bengio & Aaron Courville (2016), *Deep Learning*, MIT Press. http://www.deeplearningbook.org.

Han, Hui & Julien Siebert (2022), Tinyml: A systematic review and synthesis of existing research, *em* '2022 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)', pp. 269–274.

Impulse, Edge (2023), 'Workshop: Fomo azure counter', https://github.com/edgeimpulse/workshop-fomo-azure-counter. Acessado em: 14 de Abril de 2024.

Kallimani, Rakhee, Krishna Pai, Prasoon Raghuwanshi, Sridhar Iyer & Onel L. A. López (2023), 'Tinyml: Tools, applications, challenges, and future research directions', *Multimedia Tools and Applications* **83**(10).
URL: *http://dx.doi.org/10.1007/s11042-023-16740-9*

Kang, Byeong-ho & Su-il Choi (2017), Pothole detection system using 2d lidar and camera, *em* '2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)', pp. 744–746.

Kim, Young-Mok, Young-Gil Kim, Seung-Yong Son, Soo-Yeon Lim, Bong-Yeol Choi & Doo-Hyun Choi (2022), 'Review of recent automated pothole-detection methods', *Applied Sciences* **12**(11).
URL: *https://www.mdpi.com/2076-3417/12/11/5320*

Koch, C., K. Georgieva, V. Kasireddy, B. Akinci & P. Fieguth (2015), 'A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure', *Advanced Engineering Informatics* **29**(2), 196–210.
URL: *https://doi.org/10.1016/j.aei.2015.01.008*

Krizhevsky, Alex, Ilya Sutskever & Geoffrey E Hinton (2012), Imagenet classification with deep convolutional neural networks, *em* F.Pereira, C.Burges, L.Bottou & K.Weinberger, eds., 'Advances in Neural Information Processing Systems', Vol. 25, Curran Associates, Inc.
URL: *https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e92*
Paper.pdf

Lecun, Y., L. Bottou, Y. Bengio & P. Haffner (1998), 'Gradient-based learning applied to document recognition', *Proceedings of the IEEE* **86**(11), 2278–2324.

Lin, Tsung-Yi, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick & Piotr Dollár (2015), 'Microsoft coco: Common objects in context'.

Ma, Nachuan, Jiahe Fan, Wenshuo Wang, Jin Wu, Yu Jiang, Lihua Xie & Rui Fan (2022), 'Computer vision for road imaging and pothole detection: a state-of-the-art review of systems and algorithms', *Transportation Safety and Environment* **4**(4).
URL: *http://dx.doi.org/10.1093/tse/tdac026*

Mednis, Artis, Girts Strazdins, Martins Liepins, Andris Gordjusins & Leo Selavo (2010), Roadmic: Road surface monitoring using vehicular sensor networks with microphones, Vol. 88, pp. 417–429.

Murali, Gunji Bala, V. Santosh Kumar, Dibya Narayan Behera, Kapil Kumar Mohanta, Omkar Tulankar & Sanketh S. Salimath (2022), Pothole detection on roads using canny edge detection algorithm, *em* B. B. V. L.Deepak, D.Parhi, B.Biswal & P. C.Jena, eds., 'Applications of Computational Methods in Manufacturing and Product Design', Springer Nature Singapore, Singapore, pp. 653–661.

Nair, Vinod & Geoffrey Hinton (2010), Rectified linear units improve restricted boltzmann machines vinod nair, Vol. 27, pp. 807–814.

Ozoglu, Furkan & Türkay Gökgöz (2023), 'Detection of road potholes by applying convolutional neural network method based on road vibration data', *Sensors* **23**(22).
URL: *https://www.mdpi.com/1424-8220/23/22/9023*

Palm, Matt (2018), 'Counting bees', `https://matpalm.com/blog/counting_bees/`. Acessado em: 14 de Abril de 2024.

Prefeitura de Belo Horizonte (2022), 'Prefeitura vai investir r$ 40 milhões em serviços de tapa-buraco em 2022'. Acesso em: 8 abr. 2023.
URL: *https://prefeitura.pbh.gov.br/noticias/prefeitura-vai-investir-r-40-milhoes-em-servicos-de-tapa-buraco-em-2022*

Redmon, Joseph, Santosh Divvala, Ross Girshick & Ali Farhadi (2016), 'You only look once: Unified, real-time object detection'.

Sandler, Mark, Andrew Howard, Menglong Zhu, Andrey Zhmoginov & Liang-Chieh Chen (2019), 'Mobilenetv2: Inverted residuals and linear bottlenecks'.

Szeliski, Richard (2022), *Computer Vision: Algorithms and Applications*, Springer International Publishing.
URL: *http://dx.doi.org/10.1007/978-3-030-34372-9*

Xin, Hanyu, Yin Ye, Xiaoxiang Na, Huan Hu, Gaoang Wang, Chao Wu & Simon Hu (2023), 'Sustainable road pothole detection: A crowdsourcing based multi-sensors fusion approach', *Sustainability* **15**(8).
URL: *https://www.mdpi.com/2071-1050/15/8/6610*

Xu, Zeran (2019), 'Road pothole extraction and safety evaluation by integration of point cloud and images derived from mobile mapping sensors', *Advanced Engineering Informatics* **42**.

Yelchuri, Harsha eand others (2022), 'A review of tinyml', *arXiv preprint arXiv:2211.04448* .

Apêndice A

Hiperparâmetros do treinamento do modelo YOLOv8n

- epochs: 200
- patience: 100
- batch: -1
- imgsz: 320
- optimizer: auto
- seed: 0
- deterministic: True
- cos_lr: False
- amp: True
- dropout: 0.0
- iou: 0.7
- lr0 (learning rate initial): 0.01
- lrf (learning rate final): 0.01
- momentum: 0.937
- weight_decay: 0.0005
- warmup_epochs: 3.0
- warmup_momentum: 0.8
- warmup_bias_lr: 0.1
- box: 7.5
- cls: 0.5
- dfl: 1.5
- pose: 12.0
- kobj: 1.0
- label_smoothing: 0.0
- nbs: 64
- hsv_h: 0.015
- hsv_s: 0.7
- hsv_v: 0.4
- degrees: 0.0
- translate: 0.1
- scale: 0.5
- shear: 0.0
- perspective: 0.0
- flipud: 0.0
- fliplr: 0.5
- mosaic: 1.0
- mixup: 0.0
- auto_augment: randaugment
- erasing: 0.4
- crop_fraction: 1.0