UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO DE CIÊNCIAS, TECNOLOGIAS E SAÚDE DO CAMPUS ARARANGUÁ
CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

Vinícius Amaro da Rosa

Detecção Automática de Criadouros de Aedes Aegypti Utilizando Técnicas de Aprendizado Profundo

Araranguá
2024

Vinícius Amaro da Rosa

Detecção Automática de Criadouros de Aedes Aegypti Utilizando Técnicas de Aprendizado Profundo

Trabalho de Conclusão de Curso do Curso de Graduação em Engenharia de Computação do Centro de Ciências, Tecnologias e Saúde do Campus Araranguá da Universidade Federal de Santa Catarina para a obtenção do título de Bacharel em Engenharia de Computação.
Orientador: Prof. Alexandre Leopoldo Gonçalves, Dr.

Araranguá
2024

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Rosa, Vinícius Amaro da
Detecção Automática de Criadouros de Aedes Aegypti Utilizando Técnicas de Aprendizado Profundo / Vinícius Amaro da Rosa ; orientador, Alexandre Leopoldo Gonçalves, 2024.
40 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Campus Araranguá, Graduação em Engenharia de Computação, Araranguá, 2024.

Inclui referências.

1. Engenharia de Computação. 2. Detecção Automática de Criadouros. 3. Processamento de Imagens. 4. Aprendizado Profundo. 5. YOLO. I. Gonçalves, Alexandre Leopoldo . II. Universidade Federal de Santa Catarina. Graduação em Engenharia de Computação. III. Título.

Vinícius Amaro da Rosa

Detecção Automática de Criadouros de Aedes Aegypti
Utilizando Técnicas de Aprendizado Profundo

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do Título de "Bacharel em Engenharia de Computação" e aprovado em sua forma final pelo Curso de Graduação em Engenharia de Computação.

Araranguá, 25 de Novembro de 2024.

Documento assinado digitalmente
Jim Lau
Data: 13/12/2024 12:24:46-0300
CPF: ***.464.702-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Jim Lau, Dr.
Coordenador do Curso

Banca Examinadora:

Documento assinado digitalmente
Alexandre Leopoldo Goncalves
Data: 13/12/2024 14:04:38-0300
CPF: ***.740.349-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Alexandre Leopoldo Gonçalves, Dr.
Orientador

Documento assinado digitalmente
Jim Lau
Data: 13/12/2024 14:06:25-0300
CPF: ***.464.702-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Jim Lau, Dr.
Avaliador
Universidade Federal de Santa Catarina

Documento assinado digitalmente
Alison Roberto Panisson
Data: 13/12/2024 12:19:56-0300
CPF: ***.887.420-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Alison Roberto Panisson, Dr.
Avaliador
Universidade Federal de Santa Catarina

Detecção Automática de Criadouros de Aedes Aegypti Utilizando Técnicas de Aprendizado Profundo

Vinícius Amaro da Rosa* Alexandre Leopoldo Gonçalves †

2024, NOVEMBRO

Resumo

Doenças transmitidas por mosquitos, como Malária, Dengue e a Zika, infectam milhares de pessoas anualmente. A abordagem mais eficaz para combater tais doenças é evitar a reprodução dos mosquitos, identificando e removendo potenciais criadouros. Estratégias de controle enfrentam dificuldades devido a locais de reprodução inacessíveis, resultando em elevados custos governamentais. Para aprimorar a eficiência dessas estratégias, áreas como a Visão Computacional e o Aprendizado Profundo podem ser utilizados para agilizar a detecção automática de possíveis locais de reprodução. Neste contexto, o presente trabalho propõe um método estruturado em seis etapas: aquisição de dados, pré-processamento, definição de arquitetura para criação do modelo de classificação, treinamento e validação do modelo, inferência em imagens aéreas de drones para detecção dos criadouros e segmentação para melhor visualização dos objetos detectados. Com o conjunto de dados inicial definido e pré-processado, utilizou-se a arquitetura de detecção de objetos YOLO, baseada em Redes Neurais Convolucionais. O modelo YOLOv8x foi escolhido para realizar o treinamento, seguido por uma avaliação das métricas de desempenho e classificação dos criadouros de mosquito. Por fim, a segmentação dos criadouros foi aplicada para aprimorar a identificação dos contornos sobre o objeto identificado nas imagens. Os resultados alcançados indicam um bom desempenho para as detecções, com uma média de precisão (mAP) de 89% e um *F1-score* de 98%.

Palavras-chave: Áreas de reprodução, Detecção de objetos, Aprendizado Profundo.

*vinicius.rosa@grad.ufsc.br
†a.l.goncalves@ufsc.br

Automatic Detection of Aedes Aegypti Breeding Grounds Using Deep Learning Techniques

Vinícius Amaro da Rosa* Alexandre Leopoldo Gonçalves †

2024, NOVEMBER

Abstract

Mosquito-borne diseases such as malaria, dengue and Zika infect thousands of people every year. The most effective approach to combating these diseases is to prevent mosquito breeding by identifying and removing potential breeding sites. Control strategies face difficulties due to inaccessible breeding sites, resulting in high government costs. To improve the efficiency of these strategies, areas such as Computer Vision and Deep Learning can be used to speed up the automatic detection of possible breeding sites. In this context, this paper proposes a method structured in six stages: data acquisition, pre-processing, definition of the architecture for creating the classification model, training and validation of the model, inference in aerial drone images for detecting breeding sites and segmentation for better visualization of the detected objects. With the initial data set defined and pre-processed, the YOLO object detection architecture, based on Convolutional Neural Networks, was used. The YOLOv8x model was chosen for training, followed by an evaluation of the performance metrics and classification of mosquito breeding sites. Finally, the segmentation of the breeding sites was applied to improve the identification of the outlines on the object identified in the images. The results indicate adequate detection performance, with an average accuracy (mAP) of 89% and an F1-score of 98%.

Key-words: Breeding ground, Object detection, Deep learning.

*vinicius.rosa@grad.ufsc.br
†a.l.goncalves@ufsc.br

1 Introdução

Doenças epidêmicas transmitidas por vetores (DTVs), como Malária e Dengue e outras, são responsáveis por mais de 17% das doenças infecciosas no mundo, ocasionando mais de 700.000 mortes por ano (PAHO; OMS, 2018). Segundo Joshi e Miller (2021), enfrentar essas enfermidades continuará sendo um desafio nas próximas décadas, devido ao crescente processo de urbanização, rápida globalização e expansão das populações de mosquitos. Doenças transmitidas por mosquitos representam uma séria ameaça à saúde pública global, sendo o mosquito Aedes aegypti o vetor principal de arboviroses como Dengue, Zika, Chikungunya e Febre-Amarela urbana (RüCKERT *et al.*, 2017). Conforme o Boletim Epidemiológico (2024), realizado pelo Ministério da Saúde, até o mês de julho de 2024 ocorreram 6.215.201 milhões de casos prováveis de Dengue no Brasil (taxa de incidência de 3.060,7 casos por 100 mil habitantes), representando um aumento de 344,5% no número de casos em comparação com o mesmo período de 2023. Além disso, foram registrados 233.225 possíveis casos de Chikungunya (taxa de incidência de 114,9 casos por 100 mil habitantes).

No Brasil, o Governo Federal implementa a obrigatoriedade do levantamento anual da infestação por Aedes aegypti desde 2017. O Ministério da Saúde estabeleceu que todas as cidades brasileiras enviem anualmente dois levantamentos, o Levantamento de Índice Amostral (LIA) e o Levantamento Rápido de Índice de Infestação por Aedes aegypti (LIRAa) (CONASS, 2017). Estes relatórios são utilizados pelos gestores de saúde locais para decidir a alocação dos agentes. Segundo Passos (2023), o Aedes aegypti se reproduz em água limpa acumulada em recipientes artificiais, como caixas d'água, garrafas, baldes, pneus descartados e ralos externos. Portanto, os agentes realizam buscas diárias por esses objetos como principal método de controle do mosquito. Considerando a onipresença desses objetos, monitorar e controlar o mosquito torna-se dispendioso e de difícil execução.

As estratégias de controle de mosquitos variam em níveis individual, comunitário e regional para diferentes ambientes (JOSHI; MILLER, 2021). Cunha *et al.* (2021) destacam que o desafio reside no fato de que muitos pontos de reprodução são localizados em áreas de difícil acesso. Tais locais, não são frequentemente identificados nos relatórios por estarem na parte superior de construções ou dentro de propriedades privadas, fora do alcance visual ou de acesso dos agentes de solo. De acordo com Sousa e Paranaiba (2017), o governo brasileiro ampliou para R$ 1,5 bilhão os recursos para emergências, valor destinado para medidas de prevenção, controle e contenção de riscos, danos e agravos à saúde pública em situações que podem ser epidemiológicas, de desastres, ou de desassistência à população.

Esses fatos tornam os arbovírus transmitidos pelo Aedes aegypti um dos principais problemas de saúde global. Assim, conforme Bhutad e Patil (2023), para um controle efetivo de doenças transmitidas por vetores, é essencial realizar um manejo ambiental e ecossistêmico direcionado. No entanto, fornecer informações ambientais detalhadas em grande escala para o controle de doenças transmitidas por vetores continua sendo um desafio. De acordo com Perumal *et al.* (2023), métodos tradicionais de identificação de locais de reprodução, como inspeção visual, são frequentemente demorados e exigem muito trabalho. Por outro lado, imagens de satélite não são consideradas uma alternativa viável devido às limitações de resolução espacial e temporal, além dos custos elevados (GRUBESIC *et al.*, 2018).

Sendo assim, Passos (2023) propõe o uso de veículos aéreos não tripulados (do inglês, *Unmanned Aerial Vehicles* - UAVs), também conhecidos como *drones*, que apresentam diversas vantagens. Primeiro, os UAVs possibilitam a captura de imagens e vídeos com

resolução espacial e temporal superiores, oferecendo diversas perspectivas de altitude. Segundo, o uso de UAVs programáveis aumenta a segurança dos profissionais, minimizando sua exposição a situações ou locais perigosos. Por fim, embora a adoção desta abordagem envolva um investimento inicial relativamente elevado, isso se traduz em benefícios, uma vez que é possível cobrir uma grande área com uma equipe reduzida e custos operacionais baixos. Conforme Truong e Clavel (2022), para aumentar a eficiência no controle de mosquitos, são necessários métodos automáticos, precisos e eficazes na identificação de locais de reprodução do mosquito.

Ademais, levando-se em conta os avanços tecnológicos que viabilizam a detecção de objetos em diversos cenários, incluindo detecção de múltiplos objetos de interesse, o presente trabalho objetiva a proposição de um método voltado à detecção automática de criadouros de mosquitos. Para tal, se utiliza de um conjunto de imagens capturadas por UAVs para identificar possíveis locais de risco, visando oferecer suporte aos agentes de saúde. Dessa forma, o problema será abordado empregando uma arquitetura de Rede Neural voltada ao reconhecimento de padrões em imagens, assim como modelos para a detecção, classificação e segmentação de objetos em imagens.

Este trabalho, além desta seção, está estruturado da seguinte forma: a Seção 2 apresenta a fundamentação dos principais conceitos que permitem estabelecer as bases desta pesquisa. A Seção 3 apresenta os trabalhos correlatos que tratam da detecção de criadouros de mosquito utilizando técnicas de Aprendizado Profundo e Processamento de Imagens. Na Seção 4 é apresentado o método proposto neste trabalho, descrevendo de maneira geral como as diversas etapas são executadas. Já na Seção 5, é descrito o cenário de estudo e elencados os materiais e métodos utilizados, assim como é efetuada a apresentação dos resultados. Por fim, a Seção 6 apresenta as considerações finais, destacando as aplicações futuras que podem ser desenvolvidas a partir deste trabalho.

2 Fundamentação Teórica

2.1 Detecção de Criadouros de Mosquito

O Aedes aegypti é um dos principais vetores de arboviroses[1], como a Dengue, Zika e Chikungunya (RüCKERT *et al.*, 2017). Essas doenças são transmitidas aos seres humanos através da picada do mosquito fêmea infectado. Durante os primeiro seis meses de 2024, a Organização Mundial de Saúde (OMS) registrou nas Américas um total de 9,3 milhões de casos de dengue. Esse valor superou o número de casos notificados em um ano de todos os anos anteriores. E, dobrando o número de casos notificados durante todo o ano de 2023, com 4,6 milhões de casos (PAHO, 2024).

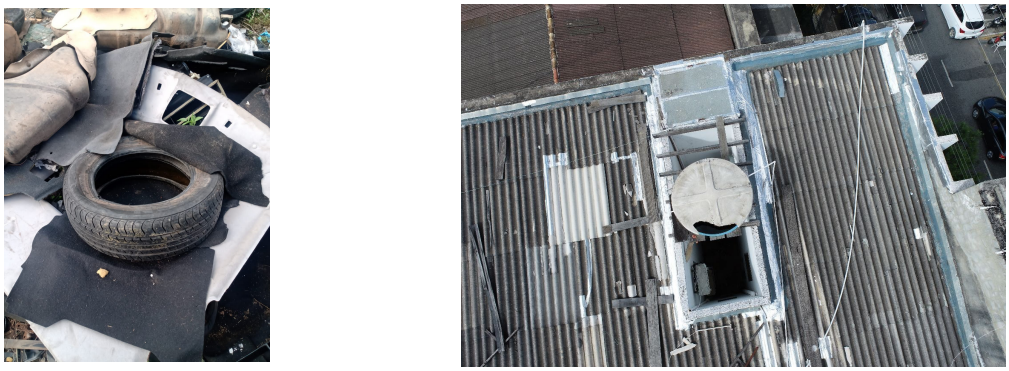
Globalmente, estima-se que existam 200.000 casos de febre-amarela anualmente, resultando em 30.000 óbitos (OMS, 2023). Já a infecção pelo vírus Zika durante a gravidez pode causar microcefalia e outras malformações congênitas ao bebê. Crianças nessas condições de anormalidades raramente possuem um desenvolvimento cognitivo e motor adequados (OMS, 2022). Soma-se isso o forte impacto econômico dessas doenças. Em agosto de 2016, uma pesquisa elaborada em 39 países da América Central e América Latina, avaliou que o custo das epidemias de dengue ultrapasse US$3 bilhões anualmente, sendo US$1,4 bilhão só no Brasil (LASERNA *et al.*, 2018). Esses dados fazem dos arbovírus transmitidos pelo Aedes aegypti um dos principais problemas de saúde global. Exceto para

[1] Sobre as arboviroses

a Febre Amarela e Dengue, não existem vacinas nem medicamentos antivirais específicos para as doenças transmitidas pelo Aedes aegypti.Portanto, a forma atual mais eficaz de combate ainda é por meio do controle e eliminação de possíveis focos de mosquitos (LAMBRECHTS; FAILLOUX, 2012).

O Aedes aegypti se reproduz em água limpa e parada, proliferando-se em recipientes que armazenam água, como tanques, baldes, fontes ornamentais, pratos para plantas, pneus e outros objetos comuns que podem ser encontrados em muitos lugares. A Figura 1 demonstra um exemplo de local de proliferação e um local com difícil acesso. Desta forma, locais de difícil acesso tornam o monitoramento e controle do mosquito um desafio, sobretudo sem o suporte técnico adequado, sendo essencial a atuação de especialistas para controlar os focos mais críticos e planejar intervenções eficazes. No entanto, o monitoramento manual é frequentemente caro, demorado e ineficaz. Combinando o conhecimento especializado com ferramentas tecnológicas, as ações de controle podem se tornar mais eficientes e abrangentes. Assim, seguindo as regulamentações sanitárias vigentes, que requerem visitas de agentes de saúde para eliminar os criadouros CONASS (2017), é possível potencializar o impacto nessas visitas ao integrar tecnologia e conhecimento.

Figura 1 – Imagem de locais de proliferação



(a) Local de proliferação

(b) Local de difícil acesso

Fonte: Elaborado pelo autor (2024)

Em consonância com tais desafios, em 2015 a Organização das Nações Unidas (ONU) estabeleceu 17 objetivos que compõem uma agenda mundial para a construção e implementação de políticas públicas que visam guiar a humanidade até 2030, visando erradicar a pobreza, proteger o planeta e garantir que todas as pessoas desfrutem de paz e prosperidade. A agenda inclui um plano de ação internacional para o alcance dos 17 Objetivos de Desenvolvimento Sustentável (do inglês, *Sustainable Development Goals* - SGDs), envolvendo diversas temáticas como igualdade de gênero, acabar com a pobreza, segurança alimentar e agricultura, saúde, educação, energia, água e saneamento, crescimento econômico inclusivo, infraestrutura e industrialização, padrões sustentáveis de produção e de consumo, mudança do clima, cidades sustentáveis, proteção e uso sustentável dos oceanos e dos ecossistemas terrestres, governança, e meios de implementação. Estas são ainda desdobrados em 169 metas, que abordam diversos temas fundamentais para o desenvolvimento humano (ODS, 2022). Aderente a este trabalho, o objetivo 3 trata sobre saúde e bem-estar, que aborda os riscos com que a população encontra ao longo do seu ciclo de vida e da mortalidade na infância às doenças transmissíveis

O foco sobre os determinantes sociais dialoga com os fatores que afetam a saúde dos mais desfavorecidos. Por exemplo, a meta 3.3 do ODS 3 determina o seguinte requisito

"Até 2030, acabar com as epidemias de AIDS, tuberculose, malária e doenças tropicais negligenciadas, e combater a hepatite, doenças transmitidas pela água, e outras doenças transmissíveis" (ODS, 2022). Ao reduzir a proliferação desses vetores, a ação direcionada para a identificação e eliminação de criadouros constrói um componente integral na prevenção de doenças transmitidas por mosquitos. Isto acaba por mitigar a incidência de doenças como Dengue, Zika, Chikungunya e Febre Amarela, assim como a detecção precoce de criadouros contribui diretamente para a meta 3.3. Ademais, promove uma maior equidade no acesso à saúde, em acordo com os princípios fundamentais dos SGDs.

De forma complementar, cabe consignar a importância do trabalho dos agentes de saúde que inspecionam propriedades para identificar e eliminar possíveis locais de reprodução de mosquitos. No entanto, tem-se desvantagens nessa estratégia, como despesas financeiras, restrições de tempo (PERUMAL *et al.*, 2023). Para fortalecer o trabalho desses profissionais, estudos exploram a detecção dos objetos associados a áreas que contribuem a reprodução dos mosquitos. Sendo assim, o uso de UAVs exibem diversas vantagens quando comparados às abordagens usadas pelo monitoramento e atuação de profissionais da saúde pública. Passos *et al.* (2022) afirmam que o uso dos UAVs não apenas permitem a captura de imagens e vídeos com maior resolução espacial ou temporal, mas também em múltiplos ângulos e altitudes. Ainda, trazem um sistema para automatizar o processo de análise aplicando técnicas de Aprendizado de Máquina (do inglês, *Machine Learning* - ML) e Visão Computacional (do inglês, *Computer Vision* - CV) para auxiliar os especialistas na localização de focos relevantes de mosquitos.

2.2 Processamento de Imagem (IP)

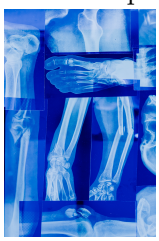
A observação sempre teve um papel importante na vida humana e, a partir desta, ocorreu ao longo dos anos a documentação de diferentes formas. Contudo, com a invenção da fotografia, o registro de momentos únicos passou por uma transformação abrindo caminho para diversas aplicações. No presente, as imagens demonstram um papel importante na vida cotidiana e, com os avanços nas Tecnologias da Informação e Comunicação (do inglês, *Information and Communication Technology*-ICT), proporcionou-se coletar e armazenar quantidades expressivas de imagens. No entanto, o volume crescente de imagens coletadas promove dificuldades de processamentos de forma manual. Segundo Huang *et al.* (2016), o Processamento de Imagem (do inglês, *Image Processing*-IP) torna-se essencial, pois permite que grande parte dessas informações visuais seja representada e processada digitalmente. Com o avanço dos computadores e processadores de sinais a partir dos anos 2000, o IP tornou-se a técnica comumente aplicada em imagens médicas, de sensoriamento remoto e da natureza, como representadas na Figura 2.

Uma imagem digital é um retrato discreto de dados que abrange informações sobre o formato e a intensidade da imagem. Conforme explicado por Gonzalez e Woods (2008), uma imagem é uma função bidimensional, $f(x, y)$, onde x e y representam as coordenadas espaciais, e a amplitude de f em qualquer ponto é referida como a intensidade ou o nível de cinza. Quando x, y e os valores de nível de cinza de f são finitos e discretos, contém-se uma imagem cinza. Os elementos da imagem são denominados de *pixels*, uma abreviação de *picture element*. Eles retratam a menor unidade em uma imagem digital, com um valor numérico que representa a informação básica da imagem, determinado pela resolução e nível de quantização específicos (BRECKON, 2010). Uma imagem é, essencialmente, um sinal 2D digitalizado, representada como um *grid* de *pixels*, como se pode analisar na Figura 3. Esses valores não só indicam cor e intensidade de luz, mas também fornecem outras propriedades, como textura, granularidade e profundidade de cor. A informação

contida em cada *pixel* pode variar significativamente dependendo do tipo de imagem sendo processada.

Figura 2 – Exemplos de imagens de interesse para diferentes áreas de aplicação de IP

(a) Sensoriamento Remoto



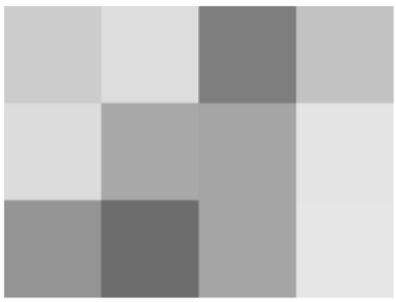
(b) Medicina

(c) Natureza

Fonte: Elaborado pelo autor (2024)

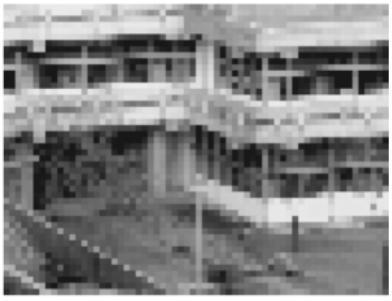
Segundo Jiao e Zhao (2019), o processamento de imagem tem exercido um papel cada vez mais vital em diversos sistemas de informação, facilitando a tomada de decisão e melhorando a cognição. Em reconhecimento de padrões e ML, o processamento de imagem popularmente utiliza processos como a compressão e a codificação, a geração de imagem, a desfocagem, a super-resolução, a segmentação de imagem, a classificação e o reconhecimento de objetos, entre outros. Em particular, técnicas de ML são frequentemente utilizadas e aplicadas com sucesso à pesquisa em processamento de imagem. Equiparado com os métodos não baseados em aprendizado, que podem não traduzir com precisão o conhecimento de domínio em regras ou características, o ML adquire seu conhecimento a partir das representações dos dados.

Figura 3 – Representação de uma imagem com diferentes quantidades de *pixels*



(a) 3 x 4

(b) 12 x 16



(c) 48 x 64

(d) 192 x 256

Fonte: Jähne (2002).

O PI, segundo ZHENG (2022), é uma técnica contemporânea que converte informa-

ções visuais em informações digitais, permitindo que sejam melhor conhecidas e processadas por um computador. Somando os processos citados, o IP garante que imagens sejam suficientemente claras e que as informações sejam identificadas corretamente, no qual o processamento de imagem deve ser suportado por vários métodos e técnicas para aprimorar a resolução e qualidade da imagem. Para Zhang e Dahu (2019), o desenvolvimento da Inteligência Artificial (do inglês, *Artificial Intelligence*-AI) também favorece o avanço da tecnologia de processamento de imagem, tornando-a amplamente utilizada em áreas como visão computacional, reconhecimento de padrões e tecnologia multimídia.

No entanto, devido às limitações e à subjetividade de algoritmos tradicionais de reconhecimento de imagem, muitos requisitos de aplicação não podem ser atendidos. Tais limitações criaram espaço para a evolução do Aprendizado Profundo (do inglês, *Deep Learning*-DL) voltadas ao reconhecimento e processamento de imagens ZHENG (2022), principalmente por meio de arquiteturas de redes neurais para este propósito. Entre estas arquiteturas, destacam-se tanto na comunidade científica quanto na indústria, as Redes Neurais Convolucionais (do inglês, *Convolucional Neural Networks*-CNNs). Desde sua proposição inicial, muitas variações de CNNs foram implementadas para diferentes tarefas em processamento de imagens, citam-se algumas: LeNet, AlexNet, GoogleNet, VGGNet, R-CNN, YOLO, SSD, SqueezeNet, ResNet, DenseNet, SegNet e DCGAN (GU *et al.*, 2018; ZHANG *et al.*, 2019).

2.3 Aprendizado Profundo

A AI é um campo de estudo da Ciência da Computação que se dedica ao desenvolvimento de programas computacionais capazes de reproduzir o comportamento humano na tomada de decisões. Sendo definida como a ciência e a engenharia a fim de criar máquinas inteligentes, especialmente softwares inteligentes (MCCARTHY, 2007). Ainda, segundo McCarthy, a AI está relacionada à tarefa de usar computadores para entender a inteligência humana, mas não se restringe a métodos que sejam biologicamente observáveis. Conforme LeCun, Bengio e Hinton (2015), a comunidade de AI vem, por muitos anos, efetuando avanços na resolução de problemas que desafiam a humanidade. Nesta direção, o Aprendizado Profundo (do inglês, *Deep Learning*-DL), tem se mostrado adequado em descobrir estruturas complexas em dados de alta dimensionalidade, tornando-se aplicável nos mais variados domínios da ciência ou dos negócios.

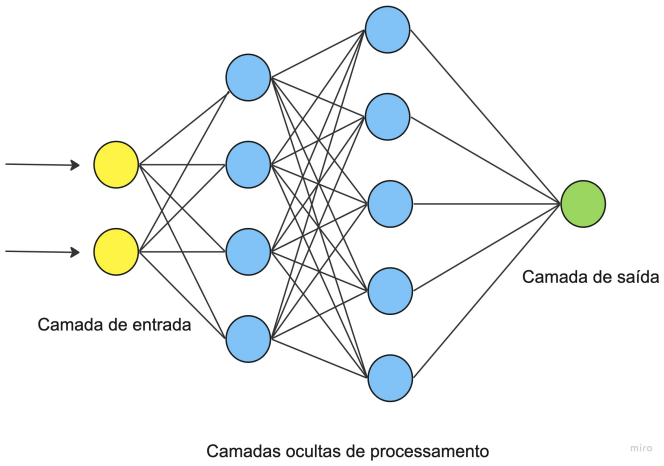
Em sua essência, DL é um subconjunto do ML que usa rede neurais de várias camadas, chamadas de Rede Neurais Profundas (do inglês, *Deep Neural Networks*-DNNs). Além disso, para construir modelos de aprendizado em múltiplas camadas, o DL emprega transformações e tecnologias de grafos simultaneamente. De modo geral, têm obtido êxito em uma variedade de aplicações, principalmente àquelas voltadas ao processamento de áudio e fala, processamento de dados visuais, processamento de linguagem natural (ALZUBAIDI *et al.*, 2021). É particularmente útil em domínios com grandes volumes de dados e de alta dimensionalidade, o que explica por que DNNs, tais como CNNs, superarem algoritmos tradicionais de ML na maioria das aplicações em que dados de texto, imagem, vídeo, fala e áudio precisam ser processados. Conforme mencionado por Heinrich (2021), o DL já demonstra desempenho superior obtido pelo ser humano em muitas aplicações específicas.

Conforme LeCun, Bengio e Hinton (2015), as Redes Neurais Artificiais (do inglês, *Artificial Neural Network*-ANNs) foram um dos avanços significativos da AI, alinhadas às arquiteturas de redes neurais cada vez mais profundas, com capacidades de aprendizado

aprimoradas. Inspiradas pelo princípio de processamento de informação em sistemas biológicos, as ANNs simulam estrutura em referência aos neurônios biológicos que compõem as diferentes camadas do cérebro humano, associadas entre si. A Figura 4 representa uma ANN composta por nós (representação de neurônios) e arestas (conexão entre os neurônios) organizada em camadas. Ainda, são interpretadas como técnicas de AI que buscam reproduzir a rede de neurônios que compõem o cérebro humano, proporcionando que os processadores reconheçam padrões complexos, como sinais cerebrais, e tomem decisões semelhantes às seres humanos (KUMAR, 2023).

De modo geral, consistem em representações matemáticas de unidades de processamento conectadas chamadas neurônios artificiais. Assim como as sinapses no cérebro, cada conexão entre neurônios transmite sinais cuja intensidade pode ser amplificada por um peso, continuamente ajustado durante o processo de aprendizado. Tipicamente, os neurônios são organizados em redes com diferentes camadas. Uma camada de entrada recebe os dados de entrada, e uma camada de saída produz os resultados (HEINRICH, 2021).

Figura 4 – Representação de uma Rede Neura Artificial



Fonte: Elaborado pelo autor (2024)

Ao longo do tempo, diversas áreas técnicas e aplicações têm sido desenvolvidas a fim de tornar mais acessível à prática da AI. Nesse contexto, ANNs e DL se destacam possuindo grande aplicabilidade no processamento de imagem. Como mencionado anteriormente, o IP é uma área que desempenha um papel vital em vários sistemas computacionais, aprimorando os níveis de cognição e facilitando os processos de tomada de decisão. Segundo Alzubaidi *et al.* (2021), as ANNs viabilizam uma variedade de tarefas, como a geração de imagem, a compressão, a desfocagem, a segmentação, a classificação e reconhecimento de objetos.

Por fim, Druzhkov e Kustikova (2016) relatam que nos últimos anos, abordagens de DL, mais especificamente às baseadas em DNNs, têm despertado especial interesse e desafios. O número elevado de parâmetros tem conduzido à necessidade de restrição da topologia da rede e das funções de ativação, bem como ao desenvolvimento de algoritmos altamente paralelos para o aprendizado. Portanto, considera-se um grande esforço na busca de arquiteturas/topologias de redes que melhor se adequem ao processamento eficiente de imagens quanto ao tempo e recursos computacionais. Neste contexto, de acordo com Alzubaidi *et al.* (2021), a arquitetura CNN se tornou uma das redes de DL mais populares e utilizadas no cenário de processamento de imagens.

2.4 Redes Neurais Convolucionais (CNNs)

As Redes Neurais Convolucionais (do inglês, *Convolutional Neural Network* - CNNs) foram inicialmente propostas por Fukushima (1980), sendo compostas por uma camada de entrada, seguida por uma conexão em cascata de várias estruturas modulares, cada uma constituída por duas camadas de células com pesos e viéses ajustáveis. Essa proposta se destacou como uma solução significativa para os desafios de reconhecimento de padrões em máquinas, especialmente em relação à variação de posição e distorções nos padrões de entrada, problemas comuns em métodos tradicionais de reconhecimento de caracteres ópticos da época. Desde então, muitos avanços têm sido alcançados, permitindo que as CNNs incorporassem o conceito de DL. Li *et al.* (2022) descrevem que a visão computacional baseada em CNNs progrediu muito nas últimas décadas, possibilitando a implementação de aplicações como o reconhecimento facial, o controle autônomo de veículos, a automação de supermercados e fábricas e o tratamento médico inteligente.

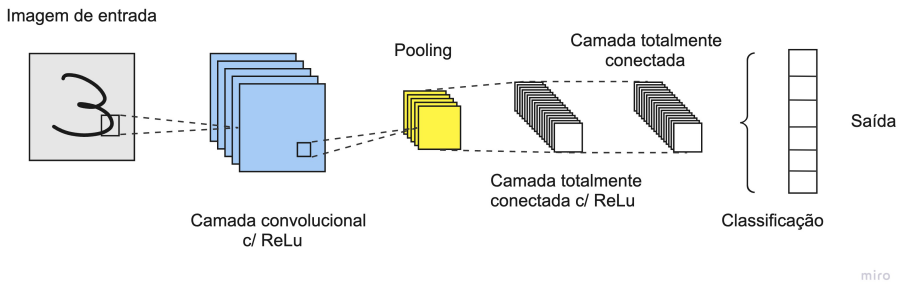
De acordo com Albawi, Mohammed e Al-Zawi (2017), o aspecto mais benéfico das CNNs é a redução do número de parâmetros, que se referem aos pesos utilizados para extrair características das imagens de entrada. Esse feito estimulou tanto pesquisadores quanto desenvolvedores a adotar modelos maiores para resolver tarefas complexas, algo que não era possível com as ANNs clássicas. De acordo com Khan *et al.* (2020), essas redes apresentam bons desempenhos em diversas aplicações relacionadas à visão computacional e ao processamento de imagens.

Segundo O'Shea e Nash (2015), às CNNs são análogas às ANNs no que tange a composição baseada em neurônios que se auto-organizam visando o aprendizado. Cada neurônio ainda receberá uma entrada e realizará uma operação (como um produto escalar seguido de uma função não linear) o que representa a base de inúmeras ANNs. De acordo com Li *et al.* (2022), ANNs como as Perceptron Multicamadas (do inglês, *Multilayer Perceptron* - MLP) são de uso mais geral e voltadas para uma ampla variedade de tarefas, mas podem não ser tão eficazes quanto as CNNs em tarefas que envolvem dados espaciais e visuais. Ademais, as CNNs geralmente têm menos parâmetros do que outras arquiteturas de ANNs com propósito geral ou similar, o que as tornam menos propensas ao reduzir o sobreajuste (do inglês, *overfitting*), ou seja, a um sobreajuste dos pesos da rede, reduzindo a capacidade de generalizar determinado problema.

A CNN é classificada como uma rede neural progressiva (do inglês, *Feedforward Neural Network*-FNN) capaz de extrair características dos dados por meio de estruturas de convolução. Diferentemente dos métodos tradicionais de extração de características, uma CNN extrai características automaticamente, identificando padrões básicos nas camadas iniciais e aumentando a complexidade dos padrões nas camadas mais profundas, sendo inspirada no córtex visual dos seres humanos. Conforme mencionado por O'Shea e Nash (2015), as CNNs se concentram principalmente na premissa de que a entrada será composta por imagens. Isso direciona a configuração da arquitetura possibilitando atender melhor as necessidades de lidar com esse tipo específico de dado. Uma das principais diferenças está no fato de que os neurônios, nas camadas internas, são organizados em três dimensões: a dimensionalidade espacial da entrada, altura e largura, e a profundidade. Na prática, isso significa que, o 'volume' de entrada terá uma dimensionalidade de $64 \times 64 \times 3$ (64×64 representa altura e largura e 3 a profundidade). Conforme passa pelas camadas de convolução e de subamostragem (do inglês, *pooling*), essa entrada é gradualmente reduzida e processada, de modo que a saída final da rede apresente uma forma simplificada, $1 \times 1 \times n$ (onde n representa o possível número de classes).

A arquitetura das CNNs é composta por três tipos de camadas: camada convolucional, camada de *pooling* e camadas totalmente conectadas. Essas camadas, quando empilhadas, constituem a arquitetura de CNN, caracterizada pelas repetições de sequências de camadas de convolução seguidas por uma camada de *pooling*. Uma arquitetura de CNN simplificada para classificação do conjunto de dados MNIST (composto por imagens de dígitos manuscritos) é visualizada na Figura 5.

Figura 5 – Uma arquitetura simples de CNN, composta por cinco camadas.



Fonte: Adaptado de O'Shea e Nash (2015)

O'Shea e Nash (2015) descrevem a funcionalidade básica da CNN de exemplo acima dividindo em quatro áreas-chave.

1. Como encontrado em outras arquiteturas de ANN, a camada de entrada conterá os valores dos *pixels* da imagem.
2. A **camada convolucional** determinará a saída dos neurônios conectados a regiões locais da entrada por meio do cálculo do produto escalar entre seus pesos e a região conectada ao volume de entrada. A **Unidade Linear Retificada** (do inglês, *Rectified Linear Units* - ReLu) visa aplicar uma função de ativação '*element-wise*', como a *sigmoide*, à saída da ativação produzida pela camada anterior.
3. A **camada de *pooling***, por sua vez, realizará simplesmente o *downsampling* ao longo da dimensionalidade espacial da entrada fornecida, reduzindo o número de parâmetros dentro dessa ativação.
4. As **camadas totalmente conectadas**, então, realizam as mesmas funções encontradas em ANNs padrão com o intuito de produzir pontuações de classe a partir das ativações, para então serem utilizadas na classificação. Também é sugerido que ReLu possa ser utilizado entre essas camadas para melhorar o desempenho.

Através deste simples método de transformação, as CNNs conseguem modificar camada por camada a entrada original usando técnicas de convolução e *downsampling* para produzir pontuações de classe. Ou seja, valores numéricos que indicam a probabilidade da entrada pertencer a cada uma das classes possíveis ao se analisar a tarefa de classificação.

2.4.1 Camada de Convolução

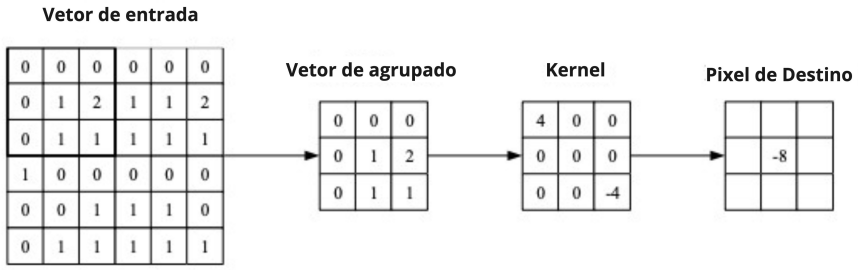
A camada convolucional desempenha um papel central na operação das CNNs. Os parâmetros da camada concentram-se no uso de *kernels*. Também conhecidos como

filtros, são pequenas matrizes utilizadas na operação de convolução, que se espalham por toda a profundidade da entrada realizando multiplicações elemento a elemento com os *pixels* correspondentes (O'SHEA; NASH, 2015). Visto que auxiliam na extração de características, *kernels* são utilizados sobre cada posição dos dados de entrada realizando somas para obter o valor resultante, o qual é empregado para a construção dos mapas de características (do inglês, *feature maps*) (YAMASHITA *et al.*, 2018). A Equação 1, que demonstra tal operação, pode ser vista abaixo.

$$FeatureMap[i,j] = \sum_{m}\sum_{n} \text{Input}[i+m, j+n] \times Kernel[m,n] \tag{1}$$

Onde $FeatureMap$ é o valor na posição (i, j) do mapa de características resultante, *Input* representa o valor na posição ($i + m$, $j + n$) dos dados de entrada, e *Kernel* o valor na posição (i, j) do *kernel* de convolução.O *kernel* de convolução reduz a dimensão dos dados durante a formação dos mapas de características (ZHANG *et al.*, 2019). Como pode ser observado no exemplo da Figura 6, à medida que o vetor de entrada é percorrido, o produto escalar é calculado para cada valor em determinado *kernel*. A partir disso, a rede aprenderá *kernels* que são ativados quando identificam uma característica específica em uma posição espacial da entrada. Isso é comumente conhecido como função de ativação.

Figura 6 – Representação visual de uma camada convolucional



Fonte: Adaptado de O'Shea e Nash (2015)

O elemento central do *kernel* é posicionado sobre o vetor de entrada, o qual é então calculado e substituído por uma soma ponderada de si e de quaisquer *pixels* próximos. Gu *et al.* (2018) menciona que cada *kernel* terá um mapa de ativação correspondente, sendo empilhado ao longo da dimensão de profundidade para formar o volume de saída completo da camada convolucional. Esses mapas de ativação têm uma função crucial, pois determinam, por meio do uso de funções de ativação (como a ReLU), se um neurônio será ativado ou não, ou seja, se a informação será propagada para a próxima camada ou não. Entre as funções típicas utilizadas para ativação encontram-se a tangente hiperbólica, a *sigmoid* e a ReLU.

2.4.2 Camada de Pooling

A camada de subamostragem ou *pooling* fornece uma operação típica de redução de amostragem, que diminui a dimensionalidade em plano dos mapas de características, a fim de introduzir invariância à translação para pequenos deslocamentos e distorções, e diminuir o número de parâmetros aprendidos subsequentes (YAMASHITA *et al.*, 2018). Na maioria das CNNs, isso é realizado por meio de *layers* de *max-pooling* com *kernels* de uma dimensionalidade de 2 × 2, aplicados com um passo (*stride*) de 2 ao longo das dimensões

espaciais da entrada. Ademais, a redução da amostragem em camadas acelera o treinamento ao diminuir os parâmetros. Isso impacta positivamente para evitar o $overfitting$.

Além do *max-pooling*, outra abordagem comum de subamostragem é o *global average pooling*, que realiza um tipo extremo de redução de amostragem, onde um mapa de características com dimensões de altura × largura é reduzido para uma matriz de 1 × 1. Isto ocorre, simplesmente aplicando a média de todos os elementos em cada mapa de características, enquanto a profundidade dos mapas de características é mantida. Essa operação é geralmente aplicada apenas uma vez antes das camadas totalmente conectadas (YAMASHITA *et al.*, 2018).

2.4.3 Camada Totalmente Conectada

A camada totalmente conectada contém neurônios que são, diretamente, conectados aos neurônios nas duas camadas adjacentes, sem serem conectados a quaisquer camadas dentro delas. Isso é análogo à maneira como os neurônios são organizados em arquiteturas tradicionais de ANNs (O'SHEA; NASH, 2015). Esta camada recebe os mapas de características, visto que após serem processados pela camada de convolução ou *pooling*, os mapas são convertidos de duas dimensões (2D) para uma dimensão (1D). Neste processo, cada valor é conectado a todos os neurônios da camada densa e cada uma das conexões gera um peso treinável (YAMASHITA *et al.*, 2018).

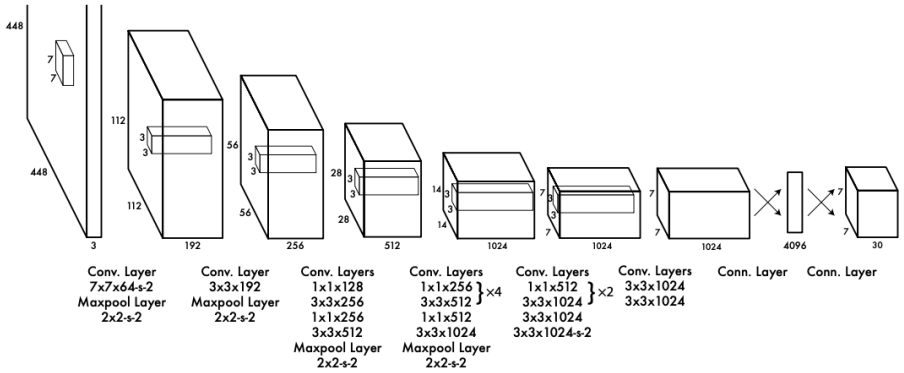
2.4.4 YOLO

A detecção de objetos envolve tanto a localização quanto a classificação de uma ou mais instâncias de objetos em imagens. Para cada imagem, o detector atribui um conjunto de caixas delimitadoras com suas respectivas coordenadas, rótulos e pontuações de confiança (PASSOS, 2023). Uma das abordagens mais eficientes e recentes aplicadas para detecção de objetos é chamada de "*You Only Look Once*" (YOLO), tem-se seu significado como uma imagem que pode ser prevista apontando quais são os objetos de interesse e onde eles estão, analisando uma imagem de relance (DU, 2018).

Desenvolvido inicialmente por Redmon *et al.* (2016), o YOLO é um modelo de passagem única que, a partir de uma imagem de entrada, utiliza uma CNN para detectar um objeto nesta imagem. Outros métodos se utilizam da detecção em duas etapas. Assim, o YOLO também pode realizar a detecção de objetos em tempo real em vídeos. Ademais, contrariando métodos como o Fast R-CNN, o YOLO utiliza uma saída mais direta baseada apenas em regressão para prever as saídas de detecção. Com o passar dos anos, foram desenvolvidas várias versões, entre elas, YOLOv2, YOLOv3, YOLOv4, YOLOv5, YOLOv7, YOLOv8, YOLOv9, YOLOv10 e, mais recente, YOLOv11.

A essência do modelo YOLO está em seu *design* aprimorado, que combina tamanho reduzido do modelo e velocidade de cálculo, podendo produzir diretamente a posição e a categoria da caixa delimitadora por meio de uma rede neural. A velocidade do YOLO é um ponto forte, visto que só precisa inserir a imagem na rede para obter o resultado da detecção, permitindo desta forma a detecção em tempo real em vídeos (JIANG *et al.*, 2022). Conforme descrito por Shinde, Kothari e Gupta (2018), YOLO aplica uma única CNN à imagem inteira, que posteriormente será divida em grades de células. Cada célula é responsável por prever caixas delimitadoras, rótulos e respectivas pontuações de confiança, indicando a presença de determinado objeto. Essas caixas delimitadoras são analisadas conforme a pontuação da confiança prevista. Possui 24 camadas convolucionais e 2 camadas totalmente conectadas conforme Figura 7.

Figura 7 – Uma representação visual da arquitetura YOLO



Fonte: Shinde, Kothari e Gupta (2018)

De acordo com Shinde, Kothari e Gupta (2018), o YOLO recebe uma imagem de entrada e a redimensiona para 448x448 *pixels*. A imagem então passa pela rede convolucional e gera uma saída na forma de um tensor 7x7x30. Este, por sua vez, fornece informações sobre as coordenadas do retângulo da caixa delimitadora e a distribuição de probabilidade sobre todas as classes para as quais o modelo foi treinado. Ao aplicar um limiar a essas pontuações de confiança e, definindo-a como uma probabilidade, rótulos de classe com pontuações menores que 30% são eliminados.

De modo geral, funciona dividindo a imagem de entrada em uma grade de N células não sobrepostas. Desta forma, se o centro de um objeto ficar em uma célula da grade, essa célula da grade é responsável por detectar esse objeto, cada uma com $S \times S$ *pixels*. Para cada célula prevê B caixas delimitadoras (*bounding boxes*) juntamente com suas coordenadas, a pontuação de confiança (*confidence score*) e a probabilidades de classe (PASSOS, 2023). Conforme Redmon *et al.* (2016), cada caixa delimitadora consiste em cinco previsões: as coordenadas (x, y) representam o centro da caixa em relação aos limites da célula da grade. A largura e a altura (w, h) são previstas em relação à imagem inteira e, por fim, a confiança, que indica a previsão de confiança, representa a interseção sobre a união (do inglês, *Intersection Over Union*-IOU) entre a caixa predita e qualquer caixa delimitadora real.

Wu (2018) define formalmente o valor de confiança como $\Pr(\text{Class}_i) \times \text{IOU}\ {}^{\wedge\text{truth}}_{\wedge\text{pred}}$; se não houver objeto previsto na célula da grade em análise, os escores de confiança devem ser zero. Caso contrário, busca-se atingir uma pontuação de confiança igual à IOU entre a caixa predita e a verdadeira. Cada célula da grade também prevê C probabilidades condicionais de classe, $Pr(Class_i|Object)$. Tais probabilidades são condicionadas à célula da grade contendo um objeto, prevendo apenas um conjunto de probabilidades de classe por célula da grade, independentemente do número de caixas delimitadoras.

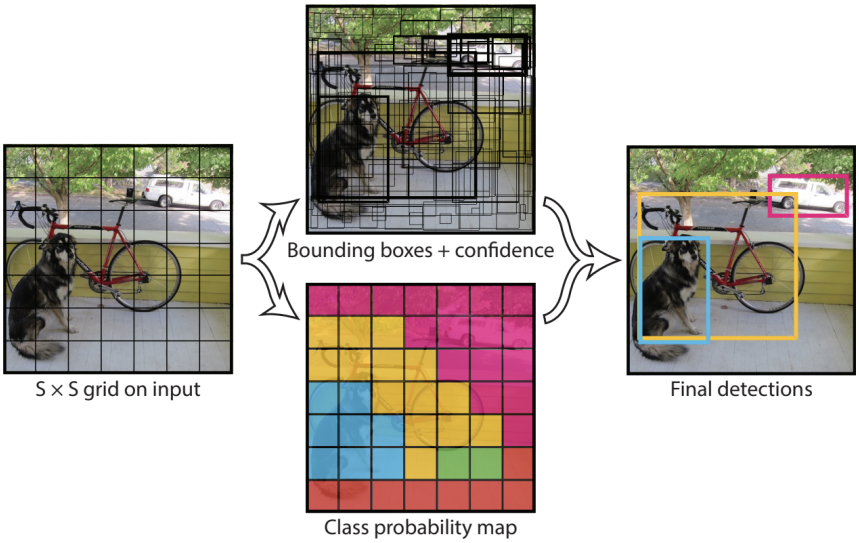
De acordo com Redmon *et al.* (2016), no momento do teste, o modelo YOLO multiplica as probabilidades condicionais das classes e as previsões individuais de confiança das caixas, como descrito na Equação 2.

$$\Pr(Class_i|Object) \times \Pr(Object) \times \text{IOU}\ {}^{\wedge\text{truth}}_{\wedge\text{pred}} = \Pr(Class_i) \times \text{IOU}\ {}^{\wedge\text{truth}}_{\wedge\text{pred}} \tag{2}$$

Ademais, essas pontuações codificam tanto a probabilidade da classe aparecer na caixa quanto o quão bem a caixa predita se ajusta ao objeto. A Figura 8 demonstra que o

sistema modela a detecção como um problema de regressão. Ou seja, a imagem é dividida em uma grade de $S \times S$ e, para cada célula da grade, B caixas delimitadoras são preditas, assim como a confiança associada a essas caixas e as probabilidades de pertencer a C classes. Essas previsões são codificadas como um tensor de $S \times S \times (B * 5 + C)$.

Figura 8 – Modelo de Funcionamento do YOLO



Fonte: Redmon *et al.* (2016)

Em 2023, introduzido pela *Ultralytics*®, o YOLOv8 significou um avanço expressivo na família de modelos YOLO, aprimorando acurácia e eficiência em relação aos modelos anteriores, passando a representar, até o período mencionado, o estado da arte em detecção de objetos em tempo real. Segundo Perumal *et al.* (2023), a arquitetura do YOLOv8 é composta principalmente por duas partes: a espinha dorsal e as camadas de detecção. A arquitetura Darknet, uma rede neural profunda composta por camadas convolucionais que extraem características da imagem de entrada, serve como a espinha dorsal do sistema. Essa espinha dorsal é responsável por processar a imagem de entrada e criar um conjunto de mapas de características utilizados pelas camadas de detecção para prever as caixas delimitadoras dos itens e as probabilidades de classe. As camadas de detecção, formadas por um grupo de camadas convolucionais, prevêem, as caixas delimitadoras para cada objeto na imagem de entrada. Para ocorrer isso, o YOLOv8 faz o uso de *anchor-boxes*, que são caixas predefinidas de várias proporções e tamanhos, servindo como ponto de partida para delimitar a localização dos objetos em uma imagem.

3 Trabalhos Correlatos

Em buscas realizadas na literatura científica foram encontrados diversos trabalhos que tratam sobre detecção automática de possíveis locais de reprodução de mosquitos. Em sua maioria, tais trabalhos se utilizam de conjuntos de dados abrangentes de imagens aéreas para desenvolver um sistema de detecção automática. As buscas foram realizadas utilizando combinações das seguintes palavras-chave: "*breeding ground*", "*breeding grounds*", "*mosquito*", "*Aedes aegypti*", "*object detection*", "*image detection*", "*image segmentation*", "*deep learning*", "*machine learning*", priorizando artigos em inglês ou português publicados nos últimos cinco anos. Os critérios de inclusão objetivaram estudos que discutem a

aplicação de CNNs e a arquitetura de detecção de objetos YOLO. A Tabela 1 apresenta, respectivamente, as bases de dados, a quantidade de trabalhos recuperados e a quantidade de trabalhos selecionados integrados na plataforma Rayyan®.

Tabela 1 – Base de dados utilizadas na identificação de trabalhos correlatos

Nome da Base	**Quantidade de Trabalhos Relacionados**
IEEE Xplore®	21
Web of Science®	11
Springer Link®	20
Periódicos CAPES®	21
Scopus®	14
Total	**87**

Fonte: Elaborado pelo autor (2023)

Na fase de seleção da literatura, foram examinados os títulos e resumos de 87 artigos, filtrando aqueles que eram pertinentes ao desenvolvimento de sistemas que se aproximavam do objeto principal deste trabalho. Esta avaliação permitiu a identificação de 9 artigos com alinhamento aos objetivos deste projeto, destacando-se pela aplicação inovadora de detecção de locais de reprodução de mosquito. Após uma revisão mais rigorosa, procedeu-se à leitura individual de cada artigo, resultando em uma síntese dos trabalhos mais alinhados ao escopo deste trabalho.

- O artigo publicado por Passos *et al.* (2022) apresenta uma solução para a detecção automática de potenciais locais de reprodução do mosquito Aedes aegypti. O sistema proposto é baseado em DNNs com consistência espaço-temporal, que inclui três módulos, sendo um extrator de características, um gerador de propostas de região e um classificador/regressor. O extrator de características utiliza uma rede convolucional ResNet-50 com rede de pirâmide de características (do inglês, *Feature Pyramid Network* - FPN). Já o gerador de propostas de região, produz regiões de interesse com base nos mapas de características. Por sua vez, o módulo de classificação e regressão, identifica as fronteiras precisas da região de interesse (RoI) e as classes de conteúdo, juntamente com suas probabilidades estimadas.

- Em Perumal *et al.* (2023), o método utilizado é baseado em DL, por meio de CNNs e Redes Adversariais Generativas (do inglês, *Generative Adversarial Networks* - GANs) para detectar automaticamente locais de reprodução de mosquitos. O artigo descreve a utilização de diferentes arquiteturas de modelos de DL, incluindo YOLO, Rede Neural Convolucional Baseada em Região de Máscara (do inglês, *Mask Region-based Convolutional Neural Network* - R-CNN), *Transformers*, e a aplicação de vários otimizadores para aumentar a precisão da detecção. Além disso, o artigo descreve a utilização de GANs para gerar um novo conjunto de imagens de treinamento, devido à insuficiência de disponibilidade de algumas classes. O desempenho dos modelos é avaliado utilizando o conjunto de dados locais de reprodução dos mosquitos (do inglês, *Mosquito Breed Ground* - MBG) e a média aritmética das precisões médias (do inglês, *Mean Average Precision* - mAP).

- Cunha *et al.* (2021) desenvolve e aplica um método de detecção de tanques de água e piscinas em áreas urbanas utilizando técnicas de sensoriamento remoto e aprendizado profundo. O pipeline do Faster R-CNN utilizado para realizar a detecção consiste em uma rede neural convolucional para receber a imagem de entrada e fornecer o

mapa de características, uma rede de propostas de região (do inglês, *Region Proposal Network* - RPN) para gerar caixas delimitadoras e prever a possibilidade de elas serem funda ou primeiro plano e uma série de camadas totalmente conectadas para prever as localizações das caixas delimitadoras na imagem e seus respectivos rótulos

- Em Passos (2023), os autores utilizam técnicas de CV e ML para detectar possíveis locais de reprodução de mosquitos Aedes aegypti. O estudo explora o uso de estratégias de aumento de dados, como escalonamento aleatório, rotação, bem como ajustes de cor e brilho, para aprimorar a detecção automática de possíveis locais de reprodução utilizando vídeos capturados por um drone. O estudo emprega dois modelos de detecção de objetos, nomeadamente o Faster R-CNN e o YOLOv5, para identificar objetos relacionados à reprodução em vídeos aéreos. Além disso, a pesquisa investiga o uso de diferentes redes neurais e técnicas de aumento de dados para aumentar o número de amostras de treinamento visando melhorar o desempenho do sistema.

- No artigo de Haddawy *et al.* (2019), os autores descrevem um *pipeline* desenvolvido para detectar potenciais locais de reprodução dos vetores da dengue a partir de imagens georreferenciadas. O *pipeline* utiliza CNNs para o reconhecimento de objetos, visando identificar diversos tipos de recipientes que possam servir como locais de reprodução em imagens do Google Street View®. As contagens de recipientes obtidas das imagens do Google Street View® são então empregadas na criação de mapas de densidade de recipientes. Além disso, os autores se baseiam em modelos multi-lineares simples, utilizando os valores de densidade de recipientes para proporcionar boas estimativas por meio do índice de Breteau, valor numérico que indica a quantidade de larvas do mosquito Aedes aegypti encontrados em residências. O trabalho se utiliza de uma rede de reconhecimento de objetos Faster R-CNN com ResNet-101 (101 camadas residuais de redes neurais).

- Bhutad e Patil (2023) propõem um sistema para indicação e monitoramento de potenciais pontos de reprodução de mosquitos usando Operações de Aprendizado de Máquia (do inglês, *Machine Learning Operations* - MLOps) e uma versão aprimorada do YOLOv3. O sistema é baseado em um algoritmo de aprendizado profundo YOLOv3 e infraestrutura em nuvem. Os autores utilizaram um conjunto de dados obtidos a partir de câmeras de segurança em áreas públicas de água parada para pré-processar os dados. A detecção por vídeo é realizada ao longo do dia e, ao encontrar a presença de água parada, imagens georreferenciadas são enviadas para notificar as autoridades locais, possibilitando o subsequente monitoramento da situação. Os autores também conduziram uma pesquisa utilizando um questionário do Google Forms® visando compreender as necessidades e demandas da população pesquisada.

- O artigo Truong e Clavel (2022) descreve uma abordagem de ML para identificar corpos d'água temporários a partir de imagens de drones em tempo real. Especificamente, os autores utilizaram o algoritmo YOLOv4, devido à sua precisão e eficiência na detecção de objetos, para treinar modelos capazes de identificar corpos d'água com base em seu tamanho e forma. Os modelos foram treinados em um conjunto de dados composto por 782 imagens de drones, as quais foram coletadas para os experimentos realizados em um estudo anterior. Os autores avaliaram o desempenho do modelo em termos de precisão, sensibilidade e pontuação, sendo este capaz de identificar pequenas poças de água em tempo real em imagens capturadas a 120 metros acima do nível da superfície. Os autores sugerem que esta tecnologia poderia

ser utilizada para controlar doenças transmitidas por mosquitos, como a malária, ao identificar e tratar habitats de larvas em tempo real.

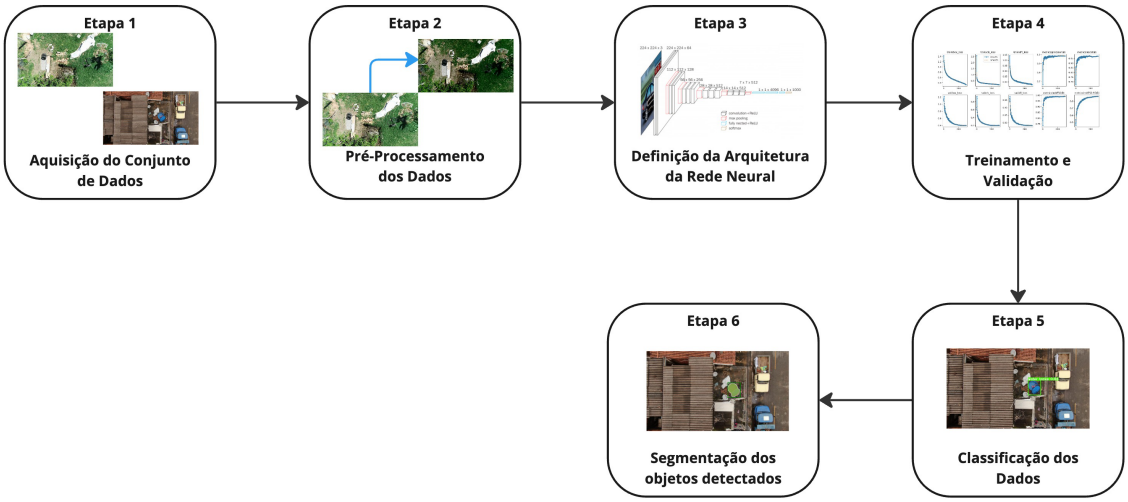
- O artigo publicado por Mylvaganam e Dissanayake (2022) propõe e compara dois métodos diferentes de DL e CNNs, YOLOv4 e Mask-RCNN, para detectar e extrair a região de áreas de acúmulo de água usando imagens aéreas obtidas por UAVs. O modelo proposto utiliza uma abordagem de transferência de aprendizado com os pesos pré-treinados de cada modelo para lidar com a falta de um conjunto de dados adequado no que tange a quantidade de imagens. Posteriormente, os pesos pré-treinados são ajustados para, finalmente, se adequarem ao cenário de aplicação utilizando um conjunto de dados personalizado com imagens aéreas. Por fim, os resultados obtidos a partir dos dois modelos propostos são comparados e analisados em termos de eficiência, eficácia e precisão, a fim de compreender qual modelo é mais adequado para a aplicação.

- Trujillano *et al.* (2023) propõem um fluxo de trabalho e uma metodologia para compilar e processar dados de treinamento rotulados, visando implementar algoritmos de DL para detectar automaticamente potenciais habitats de vetores da malária. O estudo utiliza-se de imagens de alta resolução em RGB para identificar possíveis habitats de mosquitos, aplicando abordagens de aprendizado profundo para classificar os tipos de cobertura do solo. O estudo também avalia o desempenho do classificador utilizando o coeficiente de Dice, que descreve a precisão da classificação ao nível de *píxel*. O estudo destaca ainda a importância de identificar como a informação classificada pode ser utilizada, bem como a necessidade de avaliar como os resultados serão aplicados pelos programas de controle.

4 Método Proposto

O método proposto para este trabalho fundamenta-se na análise de informações não estruturadas na forma de imagens com o intuito de detectar possíveis criadouros de mosquitos. De modo geral, o método é composto por seis etapas, apresentadas na Figura 9. A primeira e segunda etapa representam a aquisição e o pré-processamento do conjunto de dados. A partir disso, na etapa seguinte, ocorre a definição da arquitetura de rede neural adequada para utilizar no projeto. Na sequência, a quarta etapa é responsável pela realização do treinamento e validação do conjunto de dados coletado na etapa anterior, assim como gerar o modelo de predição. Seguindo, a quinta etapa objetiva a classificação de imagens buscando identificar possíveis locais com criadouros de mosquitos. E, por último, é realizado a segmentação dos objetos detectados utilizando um modelo de segmentação de imagens e vídeos.

Figura 9 – Fluxo das etapas disponíveis no método proposto



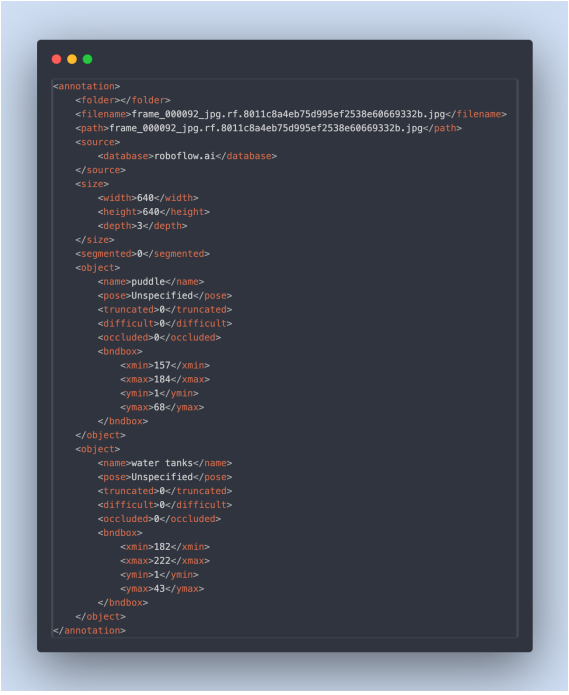
Fonte: Elaborado pelo autor(2024)

4.1 Etapa 1: Aquisição do Conjunto de Dados

A primeira etapa do método proposto consiste na obtenção de um conjunto de dados que permita a obtenção de resultados adequados quanto à precisão no que tange a detecção dos criadouros de mosquitos. Sendo assim, considerando o domínio deste trabalho, o conjunto a ser coletado deve conter um número relevante de amostras para cada classe de objeto, variabilidade de fundo, posição do objeto, luminosidade e altura. Preferencialmente, de terrenos que estejam desocupados, sem construções, preferencialmente com imagens claras e com ângulos que possibilitem a fácil visualização de objetos pequenos.

É fundamental que o conjunto de dados compreenda uma variedade de objetos rotulados associados a cada imagem, visando ensinar o modelo voltado à classificação e segmentação como identificar cada objeto que contenha criadouros de mosquitos. Os rótulos devem ser anotados em formatos padronizados e frequentemente utilizados em ML e CV, como o PASCAL VOC. A Figura 10 apresenta um exemplo neste formato, em que é possível verificar a classe do objeto identificado e suas coordenadas de localização dentro da imagem, normalmente no formato de caixas delimitadoras (*bouding boxes*).

Figura 10 – Anotação em PASCAL VOC XML



```
<annotation>
    <folder></folder>
    <filename>frame_000092_jpg.rf.8011c8a4eb75d995ef2538e60669332b.jpg</filename>
    <path>frame_000092_jpg.rf.8011c8a4eb75d995ef2538e60669332b.jpg</path>
    <source>
        <database>roboflow.ai</database>
    </source>
    <size>
        <width>640</width>
        <height>640</height>
        <depth>3</depth>
    </size>
    <segmented>0</segmented>
    <object>
        <name>puddle</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <occluded>0</occluded>
        <bndbox>
            <xmin>157</xmin>
            <xmax>184</xmax>
            <ymin>1</ymin>
            <ymax>68</ymax>
        </bndbox>
    </object>
    <object>
        <name>water tanks</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <occluded>0</occluded>
        <bndbox>
            <xmin>182</xmin>
            <xmax>222</xmax>
            <ymin>1</ymin>
            <ymax>43</ymax>
        </bndbox>
    </object>
</annotation>
```

Fonte: Elaborado pelo autor(2024).

É importante que o conjunto possua, no contexto de detecção de criadouros de mosquito, uma quantidade considerável de classes a fim de contemplar os ambientes mais comuns de foco desses insetos. Como mencionado por Passos *et al.* (2022), o mosquito se reproduz em água limpa e parada, portanto, qualquer recipiente que armazena água, tais como tanques de água, baldes, fontes ornamentais, pratos de plantas, recipientes de água para animais, pneus, entre outros, são potenciais criadouros.

Pensando no processo de aprendizado do modelo que permite a classificação e detecção de objetos, é indispensável que este seja dividido em subconjuntos de treinamento, validação e teste. O subconjunto de treinamento deve ser o maior entre os três e é usado para treinar o modelo. Já na etapa de validação, são ajustados os hiper parâmetros do modelo aplicando-se uma validação cruzada, auxiliando na capacidade de generalização durante seu uso posterior, quando serão utilizados dados não apresentados durante a fase do treinamento. É, portanto, uma etapa fundamental para se evitar o *overfitting*, ou seja, o ajuste especializado do modelo ao conjunto de dados utilizado no treinamento, reduzindo a capacidade de generalização, podendo impactar na correta previsão de novos objetos.

Por fim, o conjunto de testes tem como foco avaliar o desempenho do modelo. Após a etapa de treinamento com os conjuntos de treinamento e validação, o modelo é testado apenas uma vez a fim de verificar sua habilidade de generalização para dados novos. Sendo assim, esta etapa é totalmente independente, não sendo influência direta no treinamento do modelo, mas que, dependendo do resultado, indicará possíveis ações para ajustes nas etapas anteriores.

4.2 Etapa 2: Pré-processamento dos Dados

Para alcançar um bom aproveitamento dos dados do conjunto inicial, levando as restrições computacionais dos elementos de desenvolvimento, é realizado o pré-processamento

dos dados original. Este processo objetiva criar um novo conjunto de dados, mais eficiente e ajustado para o treinamento do modelo. Primeiro, é explorado como cada classe está distribuída dentro dos arquivos de anotações XML, possibilitando uma visão geral de cada classe pertencente ao conjunto de dados.

Em seguida, é separada uma quantidade específica de imagens para cada classe, de modo a permitir o balanceamento dos dados. Este procedimento garante o equilíbrio de classes, evitando afetar o desempenho do modelo, e reduzindo o risco de uma distribuição desbalanceada das classes comprometer a capacidade de aprendizado do modelo, além de proporcionar um conjunto de dados mais consistente.

As anotações das imagens neste trabalho, com o mapeamento das classes definidas no conjunto de dados, foram realizadas na plataforma RoboFlow®. Para cada imagem, foram realizadas as delimitações das *bounding boxes* para cada objeto de interesse. Após anotar as imagens, o sistema do RoboFlow® permite a escolha do formato para o qual se deseja treinar o modelo, sendo escolhido o formato para o YOLOv8. Essas anotações são utilizadas na etapa de treinamento do modelo.

Posteriormente a esse processo, é realizado o reajuste de imagens e anotações utilizadas no conjunto de dados. A plataforma fornece ainda a possibilidade de separar o conjunto entre treinamento, validação e teste. E, finalmente, utilizam-se métodos voltados ao aumento de imagens. Isto consiste em realizar alterações no conjunto de dados, a fim de aumentar a quantidade e variedade de instâncias, com o intuito de melhorar o desempenho do modelo quanto à capacidade de classificar corretamente em uma imagem, objetos do domínio deste trabalho.

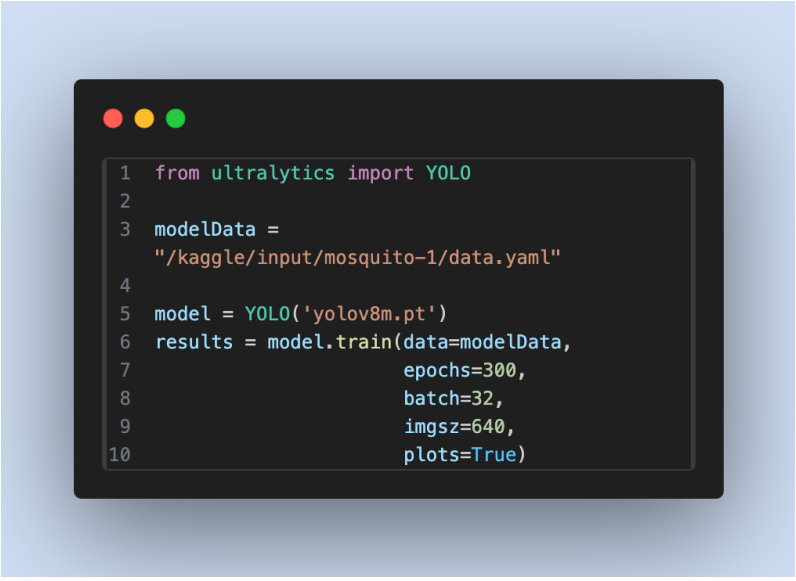
4.3 Etapa 3: Definição da Arquitetura da Rede Neural

Uma estratégia comumente utilizada em problemas de processamento de imagem é empregar o uso de Redes Neurais Convolucionais (CNNs), principalmente por sua capacidade de extrair características dos dados por meio de estruturas de convolução. Considerando o foco deste trabalho na detecção de objetos, abrangendo os diversos tipos de locais provenientes de criadouros de mosquito, optou-se por utilizar o arquitetura YOLO, considerado o estado da arte na detecção de objetos. Pensando em desenvolvimentos e evoluções futuras do método, esta etapa é flexível para que novos modelos, técnicas ou algoritmos sejam incorporados, visando aprimorar o desempenho na tarefa de detectar objetos em imagens em tempo real.

4.4 Etapa 4: Treinamento e Validação

O treinamento constitui-se em uma das etapas mais importantes no desenvolvimento do método proposto. As configurações utilizadas durante o processo de treinamento influenciam o desempenho, velocidade e precisão do modelo de classificação e detecção de objetos. As principais configurações para o treinamento incluem o modelo ("*model*"), os dados ("*data*"), as épocas ("*epochs*"), ou seja, a quantidade de iterações sobre o conjunto de dados, e o tamanho do lote ("*batch*"). A decisão desses parâmetros varia conforme o poder de processamento e quantidade de dados.

Figura 11 – *Script* de treinamento do YOLO



```
from ultralytics import YOLO

modelData =
"/kaggle/input/mosquito-1/data.yaml"

model = YOLO('yolov8m.pt')
results = model.train(data=modelData,
                      epochs=300,
                      batch=32,
                      imgsz=640,
                      plots=True)
```

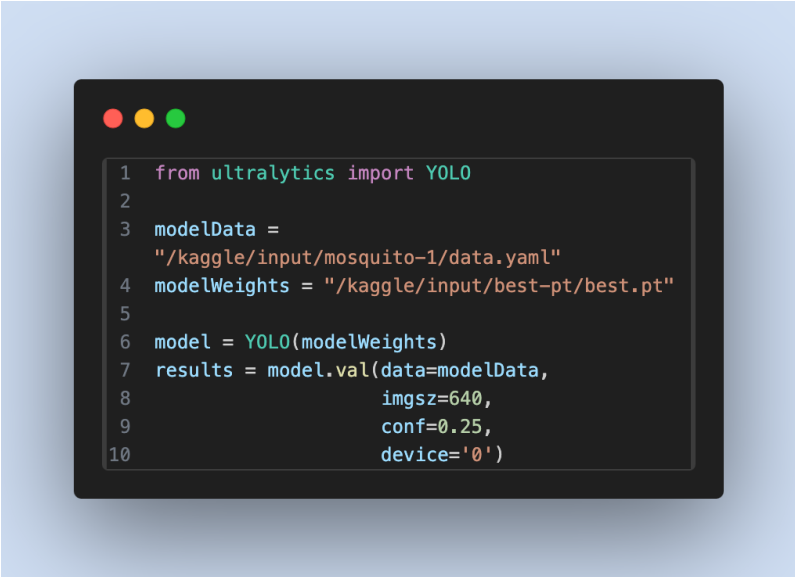
Fonte: Elaborado pelo autor

Com o conjunto de dados pré-processado, a definição da arquitetura de rede neural e os parâmetros para o modelo, pode-se iniciar a etapa de treinamento para permitir a criação do modelo voltado para predições. Durante esta etapa, o modelo aprende a localizar e identificar objetos nas imagens. No presente trabalho, com base no YOLO, utiliza-se o conceito de transferência de aprendizado, em que se parte de um modelo pré-treinado. Essa técnica acelera o processo e melhora a eficácia do treinamento, especialmente em cenários onde há limitação de dados.

O treinamento em si ocorre pela execução de um *script* Python, Figura 11, em um ambiente de execução com Unidades de Processamento Gráfico (do inglês, *Graphics Processing Units*-GPUs) dedicadas para esse tipo de tarefa. Agregada à etapa de treinamento, é efetuada a avaliação do modelo, com o conjunto de validação, provendo métricas de avaliação ao final de cada época, de modo que se consiga mensurar de maneira iterativa o processo de treinamento.

Para esta etapa, é disponibilizado pela classe YOLO um modo de validação a ser utilizado com o conjunto de treinamento para qualificar efetivamente a qualidade do modelo. Para evitar o *overfitting* e acompanhar o desempenho, essa etapa é indispensável, pois avalia a capacidade do modelo de generalizar novos dados. Para tal, podem ser empregadas diferentes métricas, tais como precisão, recall, mAP50-95 e IoU (*Intersection over Union*). Novamente, executa-se um *script* Python para a etapa de validação do modelo, como exemplificado na Figura 12.

Figura 12 – *Script* de validação do YOLO



```
from ultralytics import YOLO

modelData =
"/kaggle/input/mosquito-1/data.yaml"
modelWeights = "/kaggle/input/best-pt/best.pt"

model = YOLO(modelWeights)
results = model.val(data=modelData,
                    imgsz=640,
                    conf=0.25,
                    device='0')
```

Fonte: Elaborado pelo autor (2024)

4.5 Etapa 5: Classificação dos Dados

Depois da fase de treinamento e validação, o modelo deve ser utilizado para inferir imagens de locais de possíveis criadouros, possibilitando observar como o modelo classifica as imagens. Para tal, a avaliação ocorre por meio do modo de predição do YOLO, em que novas instâncias, ou seja, imagens não utilizadas na produção do modelo, são empregadas para verificar se o mesmo consegue realizar adequadamente a tarefa de detecção.

Figura 13 – Exemplo *Script* de predição do YOLO

```
from ultralytics import YOLO

modelWeights = "/kaggle/input/best-pt/best.pt"
model = YOLO(modelWeights)


# Inferência em lista de imagens, configurações
padrão
results = model(['/images/drone5.jpg',
                 '/images/frame_229.jpg',
                 '/images/frame_549.jpg'])

# Inferência em uma única imagem, configurações
específicas
model.predict('/images/drone5.jpg',
              save=True,
              mgsz=640,
              conf=0.25)

```

Fonte: Elaborado pelo autor.

O modo de predição realiza uma análise das imagens de entrada, identificando diferentes classes de objetos nelas. De modo geral, prevê as coordenadas que indicam a localização e o tamanho do retângulo que envolve o objeto, além de atribuir uma pontuação de confiança. Isto indica o nível de certeza do modelo de que a caixa delimitadora realmente contém um objeto, à detecção, distinguindo entre detecções verdadeiras e falsos positivos. Além disso, o modo também estima as probabilidades de pertencimento do objeto detectado a cada classe previamente treinada pelo modelo. Para aplicar a predição e obter os resultados conforme descrito, é executado o *script* Python exemplificado na Figura 13.

4.6 Etapa 6: Segmentação dos Objetos Detectados

Após o processamento das predições, as imagens exibem caixas delimitadoras a qual identificam as classes dos objetos detectados, sendo acompanhadas das probabilidades de cada detecção. Essas caixas delimitadoras permitem visualizar claramente a localização e o tipo de objeto identificado, enquanto as probabilidades de certeza fornecem uma medida de confiança em relação a cada predição. Como é possível ver na Figura 14, (a) é um tanque d'água detectado em uma residência, com 94% de certeza; enquanto (b), apresenta um pneu em meio ao um local aberto, com 84% de certeza. Com os resultados obtidos a partir das predições realizadas, as coordenadas das caixas delimitadoras são extraídas para servir como entrada para o processo de segmentação, sendo utilizadas como uma 'máscara' para destacar as regiões de interesse.

Figura 14 – Imagens de inferência de teste

(a) Local residencial (b) Local aberto

Fonte: Elaborado pelo autor (2024)

Segundo Yu *et al.* (2023), esse processo refina a detecção, segmentando as áreas de interesse com base nas características do objeto, permitindo uma análise mais detalhada e contextualizada da área que está sendo monitorada, bem como promovendo vantagens, entre elas: a) a delimitação mais precisa das áreas inspecionadas; b) uma visão mais clara de aspectos referentes a tamanho e distribuição espacial; c) a redução da confusão no processo de detecção e classificação quando os objetos estão próximos ou sobrepostos; e d) a determinação mais clara da evolução do possível criadouro, por exemplo, no caso de poças d'água, a secagem da mesma pode ocorrer depois de algum tempo. Para exemplificação, a Figura 15a mostra a extração das coordenadas do resultado de uma detecção, enquanto a Figura 15b ilustra como a segmentação é realizada utilizando o modelo SAM2.

Figura 15 – Código-fonte referente à extração de coordenadas e segmentação

```python
from pathlib import Path
# Array para armazenar as coordenadas das detecções
box = []
results = []
#Predição realizada com YOLOv8 em cima de imagens da imagem drone.jpg
# Itera sobre as detecções no resultado
for result in results:
    image_path = Path(result.path)
    if "drone5.jpg" in image_path.name:  # Filtra a imagem específica
        print(f"Detecções para {image_path.name}:")
        if result.boxes:
            for detection in result.boxes:
                # Extrai coordenadas de cada box (xmin, ymin, xmax, ymax)
                x_min, y_min, x_max, y_max = detection.xyxy[0].tolist()
                # Adiciona as coordenadas ao array box
                box.append((x_min, y_min, x_max, y_max))

                print(f" - Coordenadas: ({x_min}, {y_min}, {x_max}, {y_max})")
        else:
            print(" - Nenhuma detecção encontrada.")
# Exibe o array com as coordenadas das caixas
print(box)
```

(a) Extração das coordenadas

```python
from sam2.build_sam import build_sam2
from sam2.sam2_image_predictor import SAM2ImagePredictor

# Inicializa o modelo SAM2
checkpoint = f"{HOME}/checkpoints/sam2_hiera_large.pt"
model_cfg = "sam2_hiera_l.yaml"

# Builda e carrega o modelo SAM2
sam2_model = build_sam2(model_cfg, checkpoint, device=device)
predictor = SAM2ImagePredictor(sam2_model)

# Le e carrega a imagem
image_path = '/kaggle/input/fotos-internet/drone5.jpg'
image = Image.open(image_path)
predictor.set_image(image)

for bbox in box:
    input_box = np.array(bbox)
    predictor.set_image(image)
    masks, scores, logits = predictor.predict(
        point_coords=None,
        point_labels=None,
        box=input_box[None, :],
        multimask_output=True
    )
    # Mostra as mascaras e a caixa
    for mask in masks:
        show_mask(mask, plt.gca(), random_color=True)
    show_box(input_box, plt.gca())
# retorna a imagem
plt.axis('off')
plt.savefig("output_image.jpg", bbox_inches='tight', pad_inches=0)
plt.show()
```

(b) Segmentação com o SAM2

Fonte: Elaborado pelo autor (2024)

5 Discussão e Análise dos Resultados

5.1 Cenário de Estudo

O cenário de estudo deste trabalho envolve a detecção de criadouros de Aedes aegypti, dada a alta relevância da proliferação do mosquito no Brasil, utilizando-se de um conjunto de dados contendo imagens de locais com maior incidência de proliferação do mosquito.

Para a execução do trabalho foi utilizado um conjunto de imagens disponibilizadas por Silva e Murciego (2022). Para a aquisição das imagens, os autores utilizaram o banco de dados de vídeos anotados, chamado Locais de Reprodução de Mosquitos (do inglês, *Mosquito Breeding Grounds*-MBG). Foi especificamente criado para detectar possíveis criadouros do mosquito Aedes aegypti utilizando imagens aéreas capturadas por drones acoplados com câmeras de alta definição. O MBG possui anotações de caixas delimitadoras cobrindo vários objetos de interesse, como garrafas, baldes, piscinas, poças d'água, pneus e tanques de água.

O banco de dados possui 13 (treze) vídeos, os quais foram processados e transformados em imagens (*frames*). Em razão da quantidade de vídeos e do trabalho para extrair as imagens, Silva e Murciego (2022) selecionaram somente os vídeos 2 (dois), 6 (seis), 9 (nove) e 12 (doze) para gerar as imagens, resultando em um total de 5,094 imagens. Destas, 4,488 foram separadas para treinamento, 404 para validação e 202 para teste. Isso permitiu alcançar um conjunto representativo e variado para treinamento e avaliação do modelo.

Ainda em Silva e Murciego (2022), o pré-processamento dos dados foi produzido na plataforma RoboFlow®, uma ferramenta que simplifica tarefas de visão computacional no campo do aprendizado profundo, com suporte a modelos de detecção e classificação de objetos. Para obter melhores resultados de treinamento, validação e teste e evitar o sobreajuste do modelo, Silva e Murciego (2022) realizaram as seguintes etapas no processo de aumento dos dados (*augumentation*), foram geradas duas saídas, com transformações que incluíram rotação da caixa delimitadora entre 15° e +15°, além de um cisalhamento horizontal e vertical de ±15° da caixa delimitadora. As configurações para realizar os

experimentos do estudo envolveu o uso do Sistema Operacional Ubuntu 18.04, com um total de 128 GB de memória. A GPU utilizada foi uma RTX 3090, com a versão CUDA 11.02 instalada, e os experimentos foram conduzidos usando Python 3.7.

5.2 Materiais e Métodos

Para o desenvolvimento do método proposto foi utilizada a linguagem de programação Python®, dada a quantidade de bibliotecas que facilitam o desenvolvimento de soluções em CV, como a *OpenCV*[2]. Nesse contexto, foi utilizado o modelo YOLOv8, criado pela Ultralytics®, construído sobre avanços em aprendizado profundo e visão computacional, oferecendo desempenho adequado em termos de velocidade e precisão. A decisão pelo YOLOv8 deveu-se ao equilíbrio que o modelo oferece, comportando-se relativamente melhor para a proposta deste trabalho do que modelos mais recentes disponíveis até então.

Sabendo que para treinar redes convolucionais são realizados muitos cálculos, optou-se por realizar a implementação no Kaggle®, uma plataforma *online* para cientistas de dados e entusiastas de ML. Esta plataforma permite utilizar duas GPUs NVIDIA T4 2x, com 16GB de memória e 2,560 núcleos CUDA para acelerar o processamento, garantindo acesso a uma infraestrutura com bom desempenho. Possui um limite de 30h de uso semanais das GPUs, mas sendo limitado por 12h de uso contínua quando executando determinado *notebook*.

Após a coleta dos dados, foram realizadas as etapas de treinamento, ajustando as bibliotecas importadas da Ultralytics®. Em seguida, o conjunto de imagens é importado como entrada e para realizar o treinamento um *script* Python é executado. Para o modelo utilizado neste trabalho considerou-se a maior variante do YOLOv8, o YOLOv8x, com 68,2 milhões de parâmetros. Em seguida, a validação e teste foram executadas para avaliar o desempenho do modelo, permitindo ajustes adicionais. Com isso, um conjunto de imagens separado daquele utilizado para as etapas anteriores foi reservado para uma avaliação final do desempenho do modelo, garantindo uma análise mais abrangente do trabalho realizado.

E por fim, com as detecções realizadas, foi realizado a segmentação de imagens utilizando o modelo de segmentação SAM2, desenvolvido pela Meta®, que é um modelo unificado para segmentação de vídeos e imagens com foco na tarefa de segmentação visual a partir de *prompts* (do ingês, *Promptable Visual Segmentation*-PVS). O PVS é uma tarefa que generaliza a segmentação de imagens para vídeos, recebendo como entrada pontos, caixas ou máscaras em qualquer quadro do vídeo para definir e rastrear um segmento de interesse ao longo dos demais quadros.

Para avaliar o método proposto, é necessário realizar uma análise detalhada dos resultados obtidos após o treinamento, a fim de discutir a acurácia do modelo usado para detecção de objetos, mas também entender sua eficácia para cada classe de objetos utilizada. Para tal fim, são consideradas as métricas de avaliação *Precision*, *Recall*, *F1-Score*, *IoU* e o *Mean Average Precision* (mAP). A Equação 3 define o *Precision*, que calcula a razão entre os verdadeiros positivos (TP), número de amostras corretamente previstas, e o número total de positivos previstos entre todas as previsões positivas, avaliando a capacidade do modelo em evitar falsos positivos (FP), o número de amostras marcadas incorretamente.

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

[2] https://docs.opencv.org/4.x/d1/dfb/intro.html

A Equação 4 determina o *Recall*, cálculo da proporção de verdadeiros positivos (TP) entre todos os positivos reais, medindo a capacidade do modelo de detectar todas as instâncias de uma classe. Ou seja, avalia a capacidade do modelo de evitar falsos negativos (FN), que ilustra o número de amostras erroneamente marcadas como amostras negativas.

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

O *F1-score* fornece uma avaliação balanceada do desempenho de um modelo, calculando a média harmônica de *Precision* e *Recall*, considerando falsos positivos e falsos negativos, como mostrado na Equação 5.

$$F1\text{-}score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{5}$$

Já a *Intersection over Union* (IOU), calcula a sobreposição entre uma caixa delimitadora (*bouding boxes*) e uma caixa delimitadora real (*ground truth*). O IoU é demonstrado na Equação 6.

$$IoU = \frac{Área\ de\ intersecção}{Precision + União} \tag{6}$$

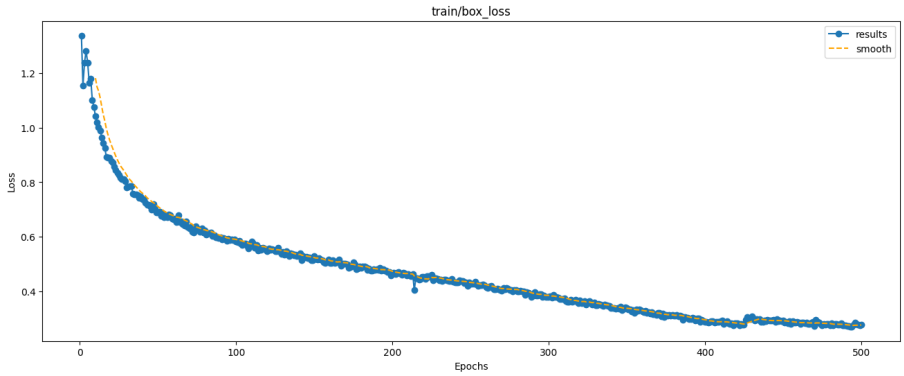
Por fim, a média da precisão (*Average-Precision*), calcula a área sob a curva *precision-recall*, fornecendo um único valor que encapsula o desempenho de *Precision* e *Recall* do modelo. A métrica mAP, demonstrada na Equação 7, estende o conceito de AP calculando os valores médios de AP entre múltiplas classes de objetos. Isso é útil em cenários de detecção de objetos multi-classe, fornecendo uma avaliação abrangente do desempenho do modelo.

$$mAP = \frac{1}{nc} \sum_{i=0}^{nc} APi \tag{7}$$

5.3 Apresentação dos Resultados

Esta seção objetiva discutir os resultados obtidos com o método proposto considerando o cenário estabelecido e o conjunto de dados deste estudo. Comenta-se inicialmente que o treinamento do modelo, baseado em YOLOv8x, durou aproximadamente 26 horas, treinados por 500 épocas e considerando o tamanho do lote (*batch*) de 16. A Figura 16 representa o desempenho do modelo associado às previsões das caixas delimitadoras durante o treinamento até sua parada na iteração 500. Percebe-se que, a medida que o treinamento avança, a perda (*loss*) decai rapidamente no início, mostrando que o modelo está aprendendo e melhorando suas previsões. Após o treinamento, modelo foi avaliado utilizando o conjunto de validação, e os resultados são apresentados na Tabela 2. A tabela revela as métricas de desempenho para cada classe individualmente, bem como para o conjunto geral para todas as classes, representado pela classe *all*.

Figura 16 – Variação da perda (*loss*) do modelo associado às previsões das caixas delimitadoras para 500 épocas



Fonte: Elaborado pelo autor(2024)

Nota-se, para todas as classes, um resultado em *mAP@50* de 99%. Este resultado demonstra a precisão média calculada em um limiar de interseção sobre união (IoU) de 0,50. É uma medida da precisão do modelo considerando apenas as detecções consideradas "fáceis". Já para a métrica *mAP@50∼95*, calculado em diferentes limiares de IoU, variando de 0,50 a 0,95, tem-se uma visão abrangente do desempenho do modelo em diferentes níveis de dificuldade de detecção. O modelo treinado chegou a um índice de 88%. Todavia, esse valor pode ser considerado baixo, tendo sido influenciado principalmente pela classe "*bottle*" com valor de 73%, visto que este é um objeto pequeno para detecções feitas por imagens aéreas. Em contrapartida, pode-se observar na Tabela 2 que, em geral, as demais classes demonstram um resultado elevado, ou seja, acima de 90%, demonstrando a eficácia do modelo em detectar tais objetos. Também é possível observar que a coluna *Images* indica o número total de imagens usadas para validação de cada classe, enquanto *Instances* mostra a quantidade de ocorrências detectadas de cada classe específica dentro dessas imagens.

Tabela 2 – Resultados da validação do modelo

Class	**Images**	**Instances**	**Precision**	**Recall**	**mAP@50**	**mAP@50∼95**
all	404	1229	0.987	0.975	0.992	0.888
bottle	404	101	0.989	0.917	0.985	0.732
bucket	404	161	0.994	0.958	0.992	0.889
pool	404	116	0.973	0.983	0.988	0.865
puddle	404	25	0.983	1.000	0.995	0.978
tire	404	189	0.989	0.994	0.994	0.904
water tanks	404	637	0.994	0.997	0.995	0.961

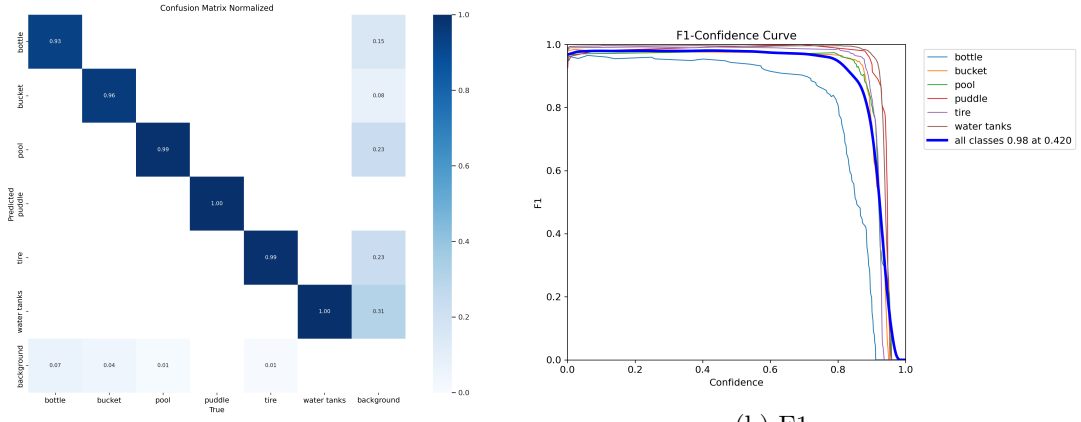
Fonte: Elaborado pelo autor (2024)

Pode-se observar que há uma variação de desempenho entre cada classe. Analisando a tabela, é evidente que as classes *"puddle"* e "water tanks" mostram a pontuação de *mAP@50∼95* maior entre todas as classes., indicando uma consistência na detecção desses objetos. Em contrapartida, a classe *"bottle"* tem uma pontuação relativamente menor quando comparada com as demais classes. Já *"bucket"*, *"pool"* e *"tire"*, exibem valores relevantes na pontuação de *mAP@50∼95*, alcançando valores próximos a 90%. Essas variações ressaltam que o modelo pode ter um desempenho diferente dependendo do tipo de objeto que está sendo identificado.

Além disso, uma maneira de analisar o equilíbrio do modelo entre falsos positivos

e falsos negativos em diferentes limiares é utilizando o gráfico da curva da pontuação de confiança F1, representada na Figura 17b. O gráfico ilustra a pontuação F1 do modelo ao longo do tempo, equilibrando *precision* e *recall*. É notável que a pontuação atribuída à classe *bottle* é relativamente menor, como representado no gráfico de confiança, onde a curva se afasta dos limiares das outras classes. Como já mencionado, isto ocorre devido às pequenas dimensões das garrafas quando comparadas aos outros objetos nos diferentes cenários de detecção.

Figura 17 – Informações gerais sobre os resultados do treinamento



(a) Matriz de confusão normalizada

(b) F1-score

Fonte: Elaborado pelo autor (2024)

Para os cálculos das métricas utilizadas neste trabalho é essencial a elaboração de uma matriz de confusão, sendo necessária para analisar o desempenho de modelos de classificação. Ela exibe o número de previsões geradas pelo modelo, incluindo verdadeiros positivos (TP), verdadeiros negativos (TN), falsos positivos (FP) e falsos negativos (FN). Analisando a Figura 17a, é possível verificar que cada linha representa uma classe e cada coluna representa a classe predita. Ao examinar as classes *"puddle"* e *"water tanks"*, *"buckets"*, *"pools"* e *"tires"*, verifica-se que o modelo exibe alta precisão na identificação desses objetos, variando de 96% a 100%. Entretanto, o modelo apresenta dificuldades com a categoria fundo (*"background"*), visto que esta classe tende a absorver um número significativo de classificações incorretas de outras classes, especialmente as classes *"water tanks"* e *"bottle"*.

A Figura 18a exibe uma amostra de imagens do conjunto de treinamento com as previsões do modelo YOLOv8 sobrepostas. Cada retângulo colorido demonstra uma detecção realizada pelo modelo, com rótulos indicando a classe detectada. Analisando a imagem, verifica-se que, ao aplicar as técnicas de *augmentation* como rotação, cisalhamento e ajustes de escala, o YOLOv8 aumenta a diversidade dos dados utilizados para treinamento, contribuindo para evitar o *"augmentation"* e melhorar a capacidade de generalização do modelo. Com isso, o modelo pode lidar melhor com variações nas imagens, como mudanças de ângulo, iluminação e perspectiva, comuns em imagens aéreas.

A ideia geral do método proposto é utilizar o algoritmo em várias situações que, com a detecção aérea, identifique os locais dos criadouros. Com isso em mente, imagens foram selecionadas do banco de dados para o conjunto de teste, visando representar situações complexas e cenários que há pouca visibilidade do terreno. A Figura 18b representa o resultado obtido em uma das imagens em que foi realizado o teste do modelo. Observa-se

Figura 18 – Informações gerais sobre os resultados do treinamento

(a) Processo de *augmentation* (b) Resultado de uma predição

Fonte: Elaborado pelo autor (2024)

uma distinção das cores para diferentes classes do modelo, com a pontuação de confiança e o nome das classes *"puddle"* e *"water tanks"*.

A Tabela 3 apresenta o resultado da avaliação do conjunto de teste, que contém um total de 202 imagens diferentes das imagens usadas para treinamento. Com isso, analisando a classificação geral do modelo, verifica-se que o mesmo é eficaz na identificação das instâncias. Com um *mAP@50∼95* de 89% para o conjunto geral para todas as classes, *all*. Assim como na validação, a situação de maior destaque refere-se à classe *bottle*. Embora tenha uma precisão de 100%, o *recall* está abaixo das demais classes, demonstrando que algumas garrafas não foram identificadas. Percebe-se que este tipo de objeto, em função de seu tamanho, traz desafios para a detecção em diferentes escalas e situações. Todavia, os resultados alcançados nas etapas de validação e teste, demonstram que o modelo produzido para a detecção dos objetos se comporta adequadamente.

Tabela 3 – Resultados da avaliação considerando o conjunto de teste

Class	**Images**	**Instances**	**Precision**	**Recall**	**mAP@50**	**mAP@50∼95**
all	202	601	0.984	0.970	0.989	0.890
bottle	202	55	1.000	0.890	0.968	0.731
bucket	202	82	0.988	0.976	0.995	0.889
pool	202	52	0.962	0.961	0.986	0.844
puddle	202	14	0.959	1.000	0.995	0.985
tire	202	87	1.000	0.993	0.995	0.920
water tanks	202	311	0.995	1.000	0.995	0.970

Fonte: Elaborado pelo autor (2024)

5.4 Experimentação da Etapa de Segmentação

A segmentação é uma tarefa da visão computacional que envolve a identificação de objetos em uma imagem e atribuição de um envelope sobre a forma desses objetos ao invés de apontar uma caixa delimitadora aproximada. Constitui-se, assim, em um desafio quando os objetos estão sobrepostos ou próximos uns dos outros. Desta forma, modelos de

segmentação de imagens devem ser capazes de distinguir entre os objetos e identificando-os corretamente quanto a sua forma. Utilizando a segmentação de instâncias com a detecção dos criadouros de mosquitos, é possível ter uma análise mais detalhada das estruturas identificadas por drone, facilitando a classificação e monitoramento dos objetos.

Para implementar a segmentação dos criadouros de mosquitos utilizou-se primeiramente o modelo YOLOv8, gerado após o treinamento, para identificar as áreas de interesse onde mosquitos podem estar localizados. Em seguida, para cada caixa delimitadora localizada, utiliza-se as coordenadas da mesma para inserir no modelo SAM2, que efetua a segmentação baseada nas características visuais do objeto.

Essa combinação permite uma melhor identificação dos contornos sobre o objeto identificado na imagem e, desta forma, cada possível criadouro é isolado visualmente, proporcionando uma avaliação detalhada e diminui a interferência de objetos próximos ou de elementos de fundo. De modo geral, tal abordagem reduz a quantidade de falsos positivos que podem ser confundidos com as classes utilizadas durante a detecção. A Figura 19 apresenta duas imagens, sendo (a) o objeto detectado e (b) o mesmo objetivo segmentado, o que possibilita visualizar o contorno do mesmo, distinguindo-o de elementos de fundo próximos. Representa, assim, uma análise detalhada da detecção que pode, por exemplo, ser utilizada de maneira mais precisa em tarefas de localização de objetos. Cabe mencionar que, diferentemente da Subseção 5.3 em que se verificou a acurácia da classificação dos objetos, nesta etapa não foram elaboradas avaliações adicionais, sendo essas reservadas para evoluções futuras do método.

Figura 19 – Resultado da segmentação sobre objetos previamente detectados

(a) Detecção YOLOv8 (b) Segmentação com SAM2

Fonte: Elaborado pelo autor (2024)

6 Considerações Finais

Considerando o grande aumento de casos de doenças transmitidas por mosquitos no Brasil, é essencial que esforços em saúde pública façam parte do planejamento de gestores públicos. Em um cenário de constante evolução tecnológica, as técnicas computacionais propiciam soluções que podem desempenhar um papel crucial ao impactar no sistema de saúde pública do país, acelerando ações preventivas, por exemplo.

Uma solução para diminuir a incidência desses casos é adotar algumas medidas que eliminam locais em que há proliferação dos mosquitos transmissores de doenças. Todavia, muitos Agentes de Controle de Endemias (ACEs) passam por desafios ao tentarem acessar lugares de difícil acesso que possam representar criadouros, comprometendo o combate efetivo de mosquitos. Neste sentido, este trabalho propõe um método baseado em DL,

mais especificamente, utilizando a arquitetura CNN para a detecção automática de locais propícios à proliferação de mosquito, visando facilitar o processo de monitoramento e eliminação de criadouros.

Para desenvolver o método proposto, utilizou-se um conjunto de dados de imagens conforme descrito na Subseção 5.1. Este conjunto de dados possui 5040 imagens e, mesmo sendo relativamente pequeno, proporcionou uma avaliação geral do modelo. Menciona-se que, para se alcançar resultados mais expressivos em algumas classes, seria necessário aumentar o conjunto de imagens utilizado neste trabalho. Todavia, os resultados obtidos pelo método proposto são relevantes para a detecção dos criadouros. O modelo consegue diferenciar diversos exemplos de classes de criadouro com um mAP de 89%, o que pode ser visto como um valor adequado em um cenário multi-classe.

É importante ressaltar que o ambiente de desenvolvimento estabelecido para este trabalho não levou em consideração diferentes realidades, onde se tem menos controle das variáveis, principalmente no que se refere às condições de clima e luminosidade. Por exemplo, a obtenção de vídeos e imagens em dia chuvosos, ou com pouca luminosidade, produzirá conteúdos que promovem desafios quanto a correta classificação, detecção e segmentação de objetos. Neste sentido, pensando na continuidade do projeto, seria interessante o estabelecimento de parcerias com empresas ou governo, tanto para o acesso a tecnologias quanto para autorizações de monitoramento, a fim de se obter vídeos e imagens de qualidade e de forma legal.

Embora o propósito deste trabalho tenha sido alcançado, pesquisas e desenvolvimentos futuros podem conduzir a resultados mais expressivos, tanto em acurácia do classificador, quanto em sua aplicabilidade. Entre as possibilidades, é viável o desenvolvimento de um sistema integrado que utilize drones ou câmeras fixas para monitorar áreas urbanas em tempo real. Para tal, o sistema precisaria de drones e câmeras de alta precisão que fizessem a coleta das imagens, assim como o armazenando, o processamento e a disponibilização dos dados para consultas. Tal sistema, poderia alertar agentes e tomadores de decisão no âmbito da saúde pública sobre os possíveis focos de criadouros de mosquito. Ainda, a participação comunitária como estratégia de coleta de dados, permitiria que cidadãos reportassem possíveis criadouros, por exemplo, através do envio de fotos e as localizações dos focos.

Em síntese, as possibilidades de enfrentamento deste problema por meio da tecnologia são diversas, possuindo como foco primordial o aumento da efetividade do trabalho de agentes de controle de endemias, buscando a redução do número de criadouros e as doenças ocasionadas por mosquitos. Ademais, ao se utilizar estratégias e tecnologias que possibilitem o enfrentamento deste desafio em nosso país, obtém-se uma melhoria na qualidade de vida da população em geral, bem como a redução de custos de investimentos no tratamento de doenças ocasionadas por mosquitos.

Referências

ALBAWI, S.; MOHAMMED, T. A.; AL-ZAWI, S. Understanding of a convolutional neural network. IEEE, p. 1–6, ago. 2017. Disponível em: <https://ieeexplore.ieee.org/document/8308186/>. Citado na página [14].

ALZUBAIDI, L. *et al.* Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data*, v. 8, n. 1, p. 53, mar. 2021. ISSN 2196-1115. Disponível em: <https://journalofbigdata.springeropen.com/articles/10.1186/s40537-021-00444-8>. Citado (2) vezes nas páginas [12 e 13].

BHUTAD, S.; PATIL, K. A Novel System for Potential Mosquito Breeding Hotspot Intimation and Monitoring Using MLOps and Improved YoloV3. *Instrumentation Mesure Metrologie*, v. 22, n. 1, p. 35–40, 2023. Citado (2) vezes nas páginas [7 e 21].

Boletim Epidemiológico. *Monitoramento das arboviroses e balanço de encerramento do Comitê de Operações de Emergência (COE) Dengue e outras Arboviroses 2024*. 2024. Boletim Epidemiológico, volume 55, número 11, Brasília: Ministério da Saúde, 2024. Acesso em:[12/12/2024]. Disponível em: <https://www.gov.br/saude/pt-br/centrais-de-conteudo/publicacoes/boletins/epidemiologicos/edicoes/2024/boletim-epidemiologico-volume-55-no-11.pdf/view>. Citado na página [7].

BRECKON, S. *Pixels*. [s.n.], 2010. 49-83 p. ISBN 9780470689776. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470689776.ch3>. Citado na página [10].

CONASS. *CONASS*. 2017. Disponível em: <https://www.conass.org.br/wp-content/uploads/2017/02/CIT12-2017.pdf>. Citado (2) vezes nas páginas [7 e 9].

CUNHA, H. *et al.* Water tank and swimming pool detection based on remote sensing and deep learning: Relationship with socioeconomic level and applications in dengue control. *PLoS ONE*, v. 16, n. 12, 2021. Citado (2) vezes nas páginas [7 e 20].

DRUZHKOV, P. N.; KUSTIKOVA, V. D. A survey of deep learning methods and software tools for image classification and object detection. *Pattern Recognition and Image Analysis*, v. 26, n. 1, p. 9–15, jan. 2016. ISSN 1555-6212. Disponível em: <https://doi.org/10.1134/S1054661816010065>. Citado na página [13].

DU, J. Understanding of Object Detection Based on CNN Family and YOLO. *Journal of Physics: Conference Series*, v. 1004, p. 012029, abr. 2018. ISSN 1742-6588, 1742-6596. Disponível em: <https://iopscience.iop.org/article/10.1088/1742-6596/1004/1/012029>. Citado na página [17].

FUKUSHIMA, K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, v. 36, p. 193–202, 1980. Citado na página [14].

GONZALEZ, R. C.; WOODS, R. E. *Digital image processing*. Upper Saddle River, N.J.: Prentice Hall, 2008. – p. ISBN 9780131687288 013168728X 9780135052679 013505267X. Disponível em: <http://www.worldcat.org/search?qt=worldcat_org_all&q=013505267X>. Citado na página [10].

GRUBESIC, T. H. *et al.* Using unmanned aerial systems (UAS) for remotely sensing physical disorder in neighborhoods. *Landscape and Urban Planning*, v. 169, p. 148–159, jan. 2018. ISSN 01692046. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0169204617302025>. Citado na página [7].

GU, J. *et al.* Recent advances in convolutional neural networks. *Pattern Recognition*, v. 77, p. 354–377, maio 2018. ISSN 00313203. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0031320317304120>. Citado (2) vezes nas páginas [12 e 16].

HADDAWY, P. *et al.* Large scale detailed mapping of dengue vector breeding sites using street view images. *PLoS neglected tropical diseases*, Public Library of Science, United States, v. 13, n. 7, p. e0007555, 2019. ISSN 1935-2735. Citado na página [21].

HEINRICH, J. Z. Machine learning and deep learning. *Electronic Markets*, v. 31, n. 3, p. 685–695, set. 2021. ISSN 1019-6781, 1422-8890. Disponível em: <https://link.springer.com/10.1007/s12525-021-00475-2>. Citado (2) vezes nas páginas [12 e 13].

HUANG, X. *et al.* Computational Imaging for Cultural Heritage: Recent developments in spectral imaging, 3-D surface measurement, image relighting, and X-ray mapping. *IEEE Signal Processing Magazine*, v. 33, n. 5, p. 130–138, set. 2016. ISSN 1053-5888. Disponível em: <http://ieeexplore.ieee.org/document/7560020/>. Citado na página [10].

JIANG, P. *et al.* A Review of Yolo Algorithm Developments. *Procedia Computer Science*, v. 199, p. 1066–1073, 2022. ISSN 18770509. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S1877050922001363>. Citado na página [17].

JIAO, L.; ZHAO, J. A Survey on the New Generation of Deep Learning in Image Processing. *IEEE Access*, v. 7, p. 172231–172263, 2019. ISSN 2169-3536. Disponível em: <https://ieeexplore.ieee.org/document/8917633/>. Citado na página [11].

JOSHI, A.; MILLER, C. Review of machine learning techniques for mosquito control in urban environments. *Ecological Informatics*, v. 61, p. 101241, mar. 2021. ISSN 15749541. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S1574954121000327>. Citado na página [7].

JäHNE, B. Digital Image Processing, 5th revised and extended edition. *Measurement Science and Technology*, v. 13, n. 9, p. 1503–1503, set. 2002. ISSN 09570233. Disponível em: <https://iopscience.iop.org/article/10.1088/0957-0233/13/9/711>. Citado na página [11].

KHAN, A. *et al.* A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*, v. 53, n. 8, p. 5455–5516, dez. 2020. ISSN 1573-7462. Disponível em: <https://doi.org/10.1007/s10462-020-09825-6>. Citado na página [14].

KUMAR, G. G. The role of artificial neural network and machine learning in utilizing spatial information. *Spatial Information Research*, v. 31, n. 3, p. 275–285, jun. 2023. ISSN 2366-3286, 2366-3294. Disponível em: <https://link.springer.com/10.1007/s41324-022-00494-x>. Citado na página [13].

LAMBRECHTS, L.; FAILLOUX, A.-B. Vector biology prospects in dengue research. *Memórias do Instituto Oswaldo Cruz*, v. 107, n. 8, p. 1080–1082, dez. 2012. ISSN 0074-0276. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0074-02762012000800022&lng=en&tlng=en>. Citado na página [9].

LASERNA, A. *et al.* Economic impact of dengue fever in Latin America and the Caribbean: a systematic review. *Revista Panamericana de Salud Pública*, 2018. Disponível em: <http://iris.paho.org/xmlui/handle/123456789/49454>. Citado na página [8].

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *Nature*, v. 521, p. 436–44, 05 2015. Citado na página [12].

LI, Z. *et al.* A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects. *IEEE Transactions on Neural Networks and Learning Systems*, v. 33, n. 12, p.

6999–7019, dez. 2022. ISSN 2162-237X, 2162-2388. Disponível em: <https://ieeexplore.ieee.org/document/9451544/>. Citado na página [14].

MCCARTHY, J. WHAT IS ARTIFICIAL INTELLIGENCE? 2007. Citado na página [12].

Mylvaganam; Dissanayake. Deep Learning for Arbitrary-Shaped Water Pooling Region Detection on Aerial Images - 2022 Moratuwa Engineering Research Conference (MERCon). p. 1–5, 7 2022. Journal Abbreviation: 2022 Moratuwa Engineering Research Conference (MERCon). Citado na página [22].

ODS. *NAÇÕES UNIDAS BRASIL*. 2022. Disponível em: <https://brasil.un.org/pt-br/sdgs/3>. Citado (2) vezes nas páginas [9 e 10].

OMS. *Zika virus*. 2022. Disponível em: <https://www.who.int/news-room/fact-sheets/detail/zika-virus>. Citado na página [8].

OMS. *Yellow fever*. 2023. Disponível em: <https://www.who.int/news-room/fact-sheets/detail/yellow-fever>. Citado na página [8].

O'SHEA, K.; NASH, R. An Introduction to Convolutional Neural Networks. arXiv, dez. 2015. ArXiv:1511.08458 [cs]. Disponível em: <http://arxiv.org/abs/1511.08458>. Citado (4) vezes nas páginas [14, 15, 16 e 17].

PAHO. *Reported cases of dengue fever in the Americas*. 2024. Disponível em: <https://www.paho.org/pt/documentos/atualizacao-epidemiologica-aumento-casos-dengue-na-regiao-das-americas-18-junho-2024>. Citado na página [8].

PAHO; OMS. *CD56/11 Plan of Action on Entomology and Vector Control 2018-2023*. 2018. Disponível em: <https://www.paho.org/en/documents/cd5611-plan-action-entomology-and-vector-control-2018-2023>. Citado na página [7].

PASSOS. Toward improved surveillance of aedes aegypti breeding grounds through artificially augmented data. *ENGINEERING APPLICATIONS OF ARTIFICIAL INTELLIGENCE*, v. 123, 8 2023. ISSN 0952-1976. Citado (4) vezes nas páginas [7, 17, 18 e 21].

PASSOS, W. L. *et al.* Automatic detection of aedes aegypti breeding grounds based on deep networks with spatio-temporal consistency. *Computers, environment and urban systems*, v. 93, p. 101754, 2022. ISSN 0198-9715. Citado (3) vezes nas páginas [10, 20 e 24].

PERUMAL, V. *et al.* Mosquito breeding grounds detection using deep learning techniques. p. 1–8, 2023. Citado (4) vezes nas páginas [7, 10, 19 e 20].

REDMON, J. *et al.* You Only Look Once: Unified, Real-Time Object Detection. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, 2016. p. 779–788. ISBN 978-1-4673-8851-1. Disponível em: <http://ieeexplore.ieee.org/document/7780460/>. Citado (3) vezes nas páginas [17, 18 e 19].

RüCKERT, C. *et al.* Impact of simultaneous exposure to arboviruses on infection and transmission by Aedes aegypti mosquitoes. *Nature Communications*, maio 2017. Disponível em: <https://www.nature.com/articles/ncomms15412>. Citado (2) vezes nas páginas [7 e 8].

SHINDE, S.; KOTHARI, A.; GUPTA, V. YOLO based Human Action Recognition and Localization. *Procedia Computer Science*, v. 133, p. 831–838, 2018. ISSN 18770509. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S1877050918310652>. Citado (2) vezes nas páginas [17 e 18].

SILVA, L. A.; MURCIEGO, A. L. Improved YOLOV7 with transformer predictions head for automated detection of mosquito breeding grounds. 2022. Citado na página [29].

SOUSA, T. D. C.; PARANAIBA, A. D. C. Análise Econômica do Combate ao Aedes aegypti no Brasil segundo a Perspectiva da Eficiência Dinâmica. *MISES: Interdisciplinary Journal of Philosophy, Law and Economics*, v. 5, n. 1, p. 29–41, dez. 2017. ISSN 2594-9187, 2318-0811. Disponível em: <https://www.misesjournal.org.br/misesjournal/article/view/39>. Citado na página [7].

TRUJILLANO, F. *et al.* Mapping Malaria Vector Habitats in West Africa: Drone Imagery and Deep Learning Analysis for Targeted Vector Surveillance. *Remote Sensing*, v. 15, n. 11, 2023. Citado na página [22].

Truong; Clavel. Identifying temporary water bodies from drone images at real-time using deep-learning techniques - 2022 International Conference on Advanced Computing and Analytics (ACOMPA). p. 12–19, 11 2022. Journal Abbreviation: 2022 International Conference on Advanced Computing and Analytics (ACOMPA). Citado (2) vezes nas páginas [8 e 21].

WU, J. Complexity and accuracy analysis of common artificial neural networks on pedestrian detection. *MATEC Web of Conferences*, v. 232, p. 01003, 2018. ISSN 2261-236X. Disponível em: <https://www.matec-conferences.org/10.1051/matecconf/201823201003>. Citado na página [18].

YAMASHITA, R. *et al.* Convolutional neural networks: an overview and application in radiology. *Insights into Imaging*, v. 9, n. 4, p. 611–629, ago. 2018. ISSN 1869-4101. Disponível em: <https://insightsimaging.springeropen.com/articles/10.1007/s13244-018-0639-9>. Citado (2) vezes nas páginas [16 e 17].

YU, Y. *et al.* Techniques and challenges of image segmentation: A review. *Electronics*, v. 12, n. 5, 2023. ISSN 2079-9292. Disponível em: <https://www.mdpi.com/2079-9292/12/5/1199>. Citado na página [28].

ZHANG, Q. *et al.* Recent advances in convolutional neural network acceleration. *Neurocomputing*, v. 323, p. 37–51, jan. 2019. ISSN 09252312. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0925231218311007>. Citado (2) vezes nas páginas [12 e 16].

ZHANG, X.; DAHU, W. Application of artificial intelligence algorithms in image processing. *J. Vis. Comun. Image Represent.*, Academic Press, Inc., USA, v. 61, n. C, p. 42–49, maio 2019. ISSN 1047-3203. Disponível em: <https://doi.org/10.1016/j.jvcir.2019.03.004>. Citado na página [12].

ZHENG, Z. Development of Image Processing Based on Deep Learning Algorithm. IEEE, Dalian, China, p. 1226–1228, abr. 2022. Disponível em: <https://ieeexplore.ieee.org/document/9777479/>. Citado (2) vezes nas páginas [11 e 12].