Universidade de Brasília

Faculdade de Tecnologia

Detecção de Áreas com Atividades de Vulnerabilidade Humana utilizando Imagens Públicas de Satélites e Aprendizagem Profunda

Júlia Passos Pontes

Laura Maciel Neves Franco

TRABALHO DE CONCLUSÃO DE CURSO

ENGENHARIA DE CONTROLE E AUTOMAÇÃO

Brasília

2024

Universidade de Brasília

Faculdade de Tecnologia

Detecção de Áreas com Atividades de Vulnerabilidade Humana utilizando Imagens Públicas de Satélites e Aprendizagem Profunda

Júlia Passos Pontes

Laura Maciel Neves Franco

Trabalho de Conclusão de Curso submetido como requisito parcial para obtenção do grau de Engenheiro de Controle e Automação.

Orientador: Prof. Dr. Flávio de Barros Vidal

Brasília

2024

FICHA CATALOGRÁFICA

Passos Pontes, Júlia.

Detecção de Áreas com Atividades de Vulnerabilidade Humana utilizando Imagens Públicas de Satélites e Aprendizagem Profunda / Júlia Passos Pontes; Laura Maciel Neves Franco; orientador Flávio de Barros Vidal. –– Brasília, 2024.

70 p.

Trabalho de Conclusão de Curso (Engenharia de Controle e Automação) –– Universidade de Brasília, 2024.

1. Aprendizado Profundo. 2. Segmentação de imagens. 3. Vulnerabilidade humana. 4. Imagens de satélite. I. Maciel Neves Franco, Laura. II. Vidal, Flávio de Barros, orient. III. Título.

Universidade de Brasília
Faculdade de Tecnologia

Detecção de Áreas com Atividades de Vulnerabilidade Humana utilizando Imagens Públicas de Satélites e Aprendizagem Profunda

Júlia Passos Pontes
Laura Maciel Neves Franco

Trabalho de Conclusão de Curso submetido como requisito parcial para obtenção do grau de Engenheiro de Controle e Automação.

Trabalho aprovado. Brasília, 19 de Agosto de 2024:

Prof. Dr. Flávio de Barros Vidal,
UnB/CIC
Orientador

Prof. Dr. Carla M. C. Cavalcante Koike,
UnB/CIC
Examinador interno

Prof. Dr. Dianne Magalhães Viana,
UnB/FT/ENM
Examinador interno

Este trabalho é dedicado a todos que lutam por um mundo onde cada vida é valorizada.

Júlia Passos Pontes

Dedico este trabalho a todos que sonham e batalham para uma sociedade mais justa, igualitária e digna.

Laura Maciel Neves Franco

Agradecimentos

Primeiramente, agradeço a todos os meus familiares que foram minha base em toda a minha vida, permitindo que eu chegasse até aqui. Amo todos vocês imensamente.

Agradeço à minha mãe, Cláudia, cujo espírito guerreiro e determinação incansável para alcançar seus objetivos são inspiração e guia para mim. Obrigada por estar sempre me incentivando e me mostrando que sou capaz, mesmo em momentos em que tudo parece estar perdido. Sou muito grata por toda a educação e carinho que você sempre me deu e pela vida maravilhosa que sempre faz questão de me proporcionar. Agradeço ao meu pai, Marcelo, cuja integridade e dedicação como profissional e ser humano são exemplos para mim. Obrigada por sempre ter os melhores conselhos, os melhores almoços de quintas-feiras, as melhores piadas bobas e por simplesmente abrir meu olhar para as oportunidades que a vida traz. Sei que sempre poderei contar com vocês em qualquer situação. Amo vocês incondicionalmente e sou muito grata em poder chamá-los de meus pais.

Agradeço também aos meus irmãos, Thais, Vítor e Luisa, e à Tati, por sempre torcerem por mim e por me proporcionarem momentos de alegria e distração únicos. Em especial, minha gratidão à Tatá por todos esses anos em que compartilhamos o mesmo quarto e amadurecemos juntas, sempre apoiando e fortalecendo uma à outra. Obrigada por me oferecer colo até mesmo quando estava ocupada e por trazer leveza para minha vida. Obrigada, Vítor, por todas as dicas sobre a UnB, pelas noites que viramos tentando resolver algum problema juntos e por todos os ensinamentos. Você foi o monitor mais dedicado que tive e sua inteligência me inspira. Amo muito vocês quatro.

Gostaria de agradecer profundamente à Laura Maciel Neves Franco por todos esses longos, porém rápidos, anos de faculdade. Minha dupla neste trabalho e minha amiga para toda a vida. Sou imensamente grata pela sua confiança e dedicação ao longo de todos esses anos. Agradeço por sempre estar ao meu lado, mesmo quando distante fisicamente, e por sempre me trazer tanta segurança de que tudo vai dar certo. Minha experiência na UnB foi muito mais leve e descontraída com você. É incrível aprender algo novo com você todos os dias. Você me inspira.

Agradeço também ao nosso orientador, Dr. Flávio de Barros Vidal, que foi extremamente atencioso e nos ajudou em todas as etapas deste trabalho de forma excepcional, sempre trazendo novas ideias e discussões relevantes.

Também gostaria de agradecer ao Matheus Mello por ser meu maior apoio há mais de 8 anos. Obrigada por sempre me acolher nos momentos difíceis e por ser meu fã número 1. Mesmo quando eu não acreditei em mim, você estava lá para me provar o contrário, arrancando minhas melhores e mais sinceras risadas. Sou muito grata por te ter ao meu lado

e por ter tido a oportunidade de amadurecer junto com você. Te amo sempre e muito.

Agradeço também a todos os meus colegas de curso, em especial, agradeço ao Paulo Victor, Matheus Felipe, Lucas Manrique, Pedro Trajano, Caio Moreira, Heisson Willen, Gustavo Joshua e Lucas Corrêa. Tive a oportunidade de fazer parte desse grupo de amigos desde minha época de caloura e poder aprender com vocês fez esses anos todos de faculdade ainda mais especiais. Acho que não sou capaz de dizer o tanto que vocês fizeram a diferença e me fizeram ser uma pessoa melhor. Cada conversa, cada trabalho em grupo, cada risada, cada discussão construíram a pessoa que sou hoje. Mais do que colegas, vocês são amigos que levarei para a vida inteira.

Agradeço à empresa júnior EngNet Consultoria, que fez parte da maior parcela da minha trajetória na UnB e me fez evoluir muito profissionalmente. Sei que minha experiência dentro da faculdade foi incrível em grande parte por conta da oportunidade de participar da EngNet. Além disso, nessa empresa pude fazer amizades fantásticas, que são exemplos de profissionais e pessoas. Em especial, deixo meus agradecimentos à Nina, ao Coelhinho e à Juju Ramos. Tenho uma profunda admiração por todos vocês.

Gostaria de agradecer ao time de vôlei feminino da UnB, no qual eu entrei no meu primeiro semestre e sigo até hoje. O vôlei me trouxe amizades maravilhosas e também um sentimento de pertencimento muito grande. Representar a Universidade de Brasília foi sempre um prazer enorme. Agradeço aos meus técnicos, Glauco e Kaio, e a todas as meninas do time por acreditarem no meu potencial.

Agradeço às amizades e irmãs que a vida me trouxe muito antes da UnB, Bruna Turene, Isa e Cath, vocês são meu porto seguro e exemplo de pessoas bondosas e dedicadas. A amizade de vocês me traz conforto e alegria, independentemente de distância e tempo.

Por fim, agradeço à Aninha Reis, à Lu Contaifer, ao Jean e ao Samuks, outros amigos que a UnB trouxe como presente para minha vida. Amo sair e estar com vocês, obrigada por serem amigos tão fiéis e por confiarem tanto em mim. Obrigada por trazerem mais cor para esses últimos anos de UnB.

Júlia Passos Pontes

Gostaria de agradecer, inicialmente, aos meus pais, Sílvia Neves Maciel Franco e José Franco de Sá, que me ensinaram, desde pequena, o poder transformador da educação. São eles que me incentivam a voar cada vez mais alto e me apoiam a cada decisão tomada, por mais difícil que seja. Tenho muito orgulho de ser filha de vocês. Agradeço também ao meu irmão, Pedro Inácio Maciel Neves Franco, em quem eu me espelho desde pequena, mesmo sem ele saber.

Agradeço ao meu orientador, Dr. Flávio de Barros Vidal, que não poupou esforços para nos auxiliar nessa etapa e foi essencial para que este trabalho pudesse ser desenvolvido com maior riqueza técnica. Agradeço também aos meus amigos e futuros colegas de profissão: Paulo Victor, Lucas Manrique, Matheus Felipe, Heisson Willen, Gustavo Joshua, Caio Moreira e Pedro Trajano. Obrigada por serem uma família para mim em Brasília e formarem as minhas melhores memórias da UnB.

Por fim, deixo os meus maiores agradecimentos a minha amiga e dupla neste trabalho: Júlia Passos Pontes. Obrigada por me acompanhar em cada passo da graduação e deixar a jornada muito mais leve e divertida. Foi um prazer aprender tanto com você, partilhar inúmeras histórias e concluir esta fase ao seu lado.

Laura Maciel Neves Franco

Resumo

Indivíduos sujeitos a condições de vida precárias e tendo suas necessidades básicas negligenciadas é uma infeliz realidade no Brasil. Esse cenário, que será abordado neste trabalho segundo o conceito de "vulnerabilidade humana", pode ser exemplificado por meio de famílias que habitam abrigos inadequados, sem estruturas básicas e em meio às ruas de centros urbanos ou rurais. Nesse sentido, tomando o Distrito Federal como escopo de pesquisa, o presente projeto se propõe a desenvolver duas bases de dados inéditas a ser disponibilizadas publicamente e compostas por imagens de satélite, considerando escalas de 50m e 100m, de regiões de vulnerabilidade humana, áreas consideradas tradicionais e resíduos dispostos inadequadamente. Além disso, fazendo-se uso dessas bases de imagens, realizou-se treinamentos com os modelos de aprendizado profundo da YOLOv7 e outros modelos de aprendizado profundo do Estado da Arte, estes sendo utilizados na segmentação de imagens. Adotando-se uma abordagem exploratória, este trabalho compara os resultados dos diferentes modelos de segmentação de imagens e as estratégias de treinamento com inicialização aleatória de pesos (*from scratch*) e com carregamento de pesos pré-treinados (*transfer learning*) aplicadas em cada um deles. Desse modo, o presente trabalho foi capaz de atingir valores máximos de F1 *score* de 0,55 para a YOLOv7 e 0,64 para modelos do *Segmentation Models*.

Palavras-chave: Aprendizado Profundo. Segmentação de imagens. Vulnerabilidade humana. Imagens de satélite.

Abstract

People living in precarious conditions and with their basic needs neglected is an unfortunate reality in Brazil. This scenario will be approached in this work according to the concept of "human vulnerability" and can be exemplified through families who live in inadequate shelters, without basic structures and on the streets of urban or rural centers. Therefore, assuming the Federal District as the research scope, this project proposes to develop two new databases to be made available publicly, considering the map scales of 50m and 100m, and composed by satellite images of human vulnerability areas, regions treated as traditional and waste disposed inadequately. Furthermore, using these image bases, trainings were done with the YOLOv7 model and other deep learning models for image segmentation. By adopting an exploratory approach, this work compares the results of different image segmentation models and training strategies, using random weight initialization (from scratch) and pre-trained weights (transfer learning). Thus, the present work was able to reach maximum F1 score values of 0.55 for YOLOv7 and 0.64 for other segmentation models.

Keywords: Deep Learning. Image segmentation. Human vulnerability. Satellite images.

Lista de figuras

Lista de tabelas

Lista de abreviaturas e siglas

CNN	*Convolutional Neural Network*
COCO	*Common Objects in Context*
CUDA	*Compute Unified Device Architecture*
DF	Distrito Federal
DNN	*Deep Neural Network*
DPN	*Dual Path Networks*
E-ELAN	*Extended Efficient Layer Aggregation*
ELAN	*Efficient Layer Aggregation*
FPN	*Feature Pyramid Networks*
GPU	*Graphics Processing Unit*
IA	Inteligência Artificial
IoU	*Intersection over Union*
IVS-DF	Índice de Vulnerabilidade Social do Distrito Federal
MAnet	*Multi-Scale Attention Network*
mAP	*Mean Average Precision*
MIT	*Massachusetts Institute of Technology*
PAN	*Pyramid Attention Network*
POC	*Prova de Conceito*
PSPNet	*Pyramid Scene Parsing Network*
R-CNN	*Region Based Convolutional Neural Networks*
RA	Regiões Administrativa
ResNet	*Residual Neural Network*
RPN	*Region Proposal Network*
SE-Net	*Squeeze-and-Excitation Networks*
SK-ResNet	*Selective Kernel Residual Neural Network*
SPP	*Spatial Pyramid Pooling*
SSD	*Single Shot multibox Detection*
Unet	*U-Shaped Network*
VGG	*Visual Geometry Group*
WoS	*Web of Science*
YOLO	*You Only Look Once*

Sumário

1 Introdução

A América Latina e o Caribe representam 40% da biodiversidade mundial de acordo com Walter (2016), sendo uma região bastante rica quando se trata de diversidade ecológica, biomas e climas. Contudo, toda essa riqueza natural contrasta fortemente com os desafios socioeconômicos enfrentados pela área, que é uma das mais subdesenvolvidas do mundo. O Brasil, como maior país dessa região, impacta profundamente nos indicadores e equilíbrio socioeconômico da América Latina e do Caribe. Faz-se necessário, desta maneira, discutir formas de superar os desafios associados à vulnerabilidade humana no Brasil e como atuar de forma sustentável.

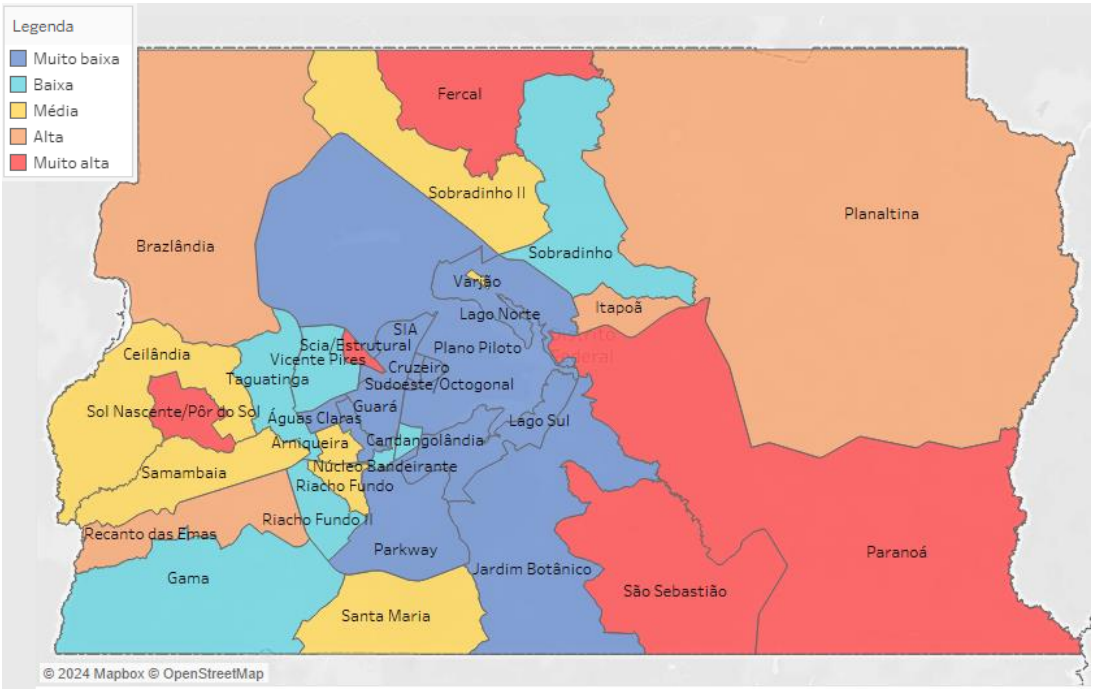
A sustentabilidade, neste trabalho, será entendida como a habilidade de atender às necessidades presentes sem comprometer as necessidades das gerações futuras (Coeckelbergh, 2021), sendo caracterizada, de acordo com Mensah (2019), pela influência de 3 pilares: economia, sociedade e meio ambiente. A vulnerabilidade humana, por sua vez, possui uma definição mais fluida e diversos fatores associados como mostrado em Caliari *et al.* (2021). Neste trabalho, essa vulnerabilidade será explorada no âmbito do comportamento das pessoas como entidades sociais interagindo com um ambiente frágil e superexplorado (Salas; Patterson; Vidal, 2022).

De maneira geral, o conceito de vulnerabilidade está associado à ideia de precariedade, sendo o vulnerável aquele que está em uma situação de desvantagem quanto a algum critério, podendo ser renda, serviços, qualidade de vida, condições de saúde, entre outros, conforme explicado em (IPEDF – Instituto de Pesquisa e Estatística do Distrito Federal, 2024). Assim, de acordo com o detalhado em Salas, Patterson e Vidal (2022), a vulnerabilidade humana vai muito além de apenas aspectos sociais relacionados à qualidade de vida e renda, mas envolve também aspectos associados ao meio ambiente e às mudanças climáticas nas quais as pessoas estão submetidas. Contudo, é válido ressaltar que neste trabalho a vulnerabilidade humana será associada principalmente ao aspecto habitacional da população, ou seja, pessoas que possuem moradias sem boa parte ou sem nenhuma estrutura básica em meio a ambientes urbanos ou rurais.

Tomando-se como base o Distrito Federal (DF), pode-se fazer uma análise da vulnerabilidade humana por meio do Índice de Vulnerabilidade Social do Distrito Federal (IVS-DF), que busca mensurar a vulnerabilidade social do DF em uma perspectiva multidimensional (IPEDF – Instituto de Pesquisa e Estatística do Distrito Federal, 2024). Esse índice considera 4 dimensões: Infraestrutura e Ambiência Urbana, Capital Humano, Renda e Trabalho e Habitacional. Dessa forma, apesar de representar a vulnerabilidade social no DF, esse índice engloba também o que retrataremos aqui como vulnerabilidade humana sob a perspectiva habitacional.

Nesse sentido, nota-se que, mesmo que dos anos de 2018 para 2021 esse índice tenha permanecido em uma faixa de vulnerabilidade social considerada média (0,32 em 2018 e 0,33 em 2021), conforme IPEDF – Instituto de Pesquisa e Estatística do Distrito Federal (2024) pode-se perceber que em 2021 ainda existiam muitas Regiões Administrativas (RAs) com IVS-DF na faixa entre 0,4 e 1, ou seja, índices considerados altos ou muito altos, demonstrando altos resultados para os indicadores das 4 dimensões citadas anteriormente. Com a Figura 1.1, é possível verificar as RAs do DF que apresentaram maiores índices de vulnerabilidade social em 2021. Dentre elas destaca-se Paranoá, São Sebastião, SCIA/Estrutural, Fercal e Sol Nascente/Pôr do Sol como as regiões com os maiores IVS-DF.

Figura 1.1 – Índice de Vulnerabilidade Social do Distrito Federal em 2021.



Fonte: ©2021 PDAD - Pesquisa Distrital por Amostra de Domicílios (CODEPLAN, 2021).

Em vista do supracitado, tanto os pilares da sustentabilidade quanto a vulnerabilidade humana foram utilizados como termos de consulta para fazer uma busca por artigos de pesquisa adequados em um contexto geográfico mundial e, também, em um contexto restrito ao Brasil. Dessa forma, percebe-se com a Tabela 1.1 que, mesmo existindo uma quantidade aceitável de trabalhos escritos associados à união das áreas de Inteligência Artificial (IA) e de sustentabilidade, verifica-se uma redução grande na quantidade de trabalhos quando se filtra pelo Brasil e uma redução ainda maior quando esse filtro é feito considerando-se a vulnerabilidade humana no Brasil. A Tabela 1.1 mostra os dados coletados utilizando a base de dados do site *Web of Science* (WoS) (https://www.webofscience.com).

Tabela 1.1 – Número de trabalhos escritos de acordo com cada filtro usando a base de dados do site Web of Science (WoS).

Termo de Consulta	Expressão Lógica	N° de Trabalhos Publicados
TC1	("Artificial Intelligence" OR "Machine Learning")	871.435
TC2	TC1 AND ("Sustainability" OR "Sustainable" OR "Environment" OR "Economy" OR "Society")	133.679
TC3	TC1 AND ("Human Vulnerability"OR "Social Vulnerability")	75
TC4	TC1 AND ("Brazil")	16.792
TC5	TC2 AND ("Brazil")	2.809
TC6	TC3 AND ("Brazil")	2

1.1 Motivação

Como visto anteriormente, a vulnerabilidade humana é um fenômeno multidimensional que envolve a exposição de indivíduos ou grupos a riscos que podem comprometer a saúde física, social e econômica dessas pessoas. Conforme evidenciado em (Salas; Patterson; Vidal, 2022), as pessoas que estão mais suscetíveis aos efeitos de alterações climáticas vivem e trabalham em ambientes com condições ruins, não possuem o conhecimento necessário de adaptação e, portanto, não são capazes de lidar com eventuais impactos ambientais ou acidentes que as comprometam. Além de tudo, percebe-se, com o auxílio da Tabela 1.1, que ainda há muito espaço para estudos relacionados à vulnerabilidade no Brasil, sendo um tema de extrema relevância quando o assunto é a formulação de políticas públicas eficientes e inclusivas.

Dessa forma, o presente trabalho utiliza a região do DF, como escopo de estudo para analisar de forma exploratória a situação de vulnerabilidade humana no Brasil. Assim sendo, este estudo é fundamentado na detecção de áreas com atividades de vulnerabilidade humana utilizando imagens públicas de satélites e técnicas de aprendizagem profunda. Dito isso, o motivo principal baseia-se na necessidade de identificar regiões no DF onde a população se encontra em situação de vulnerabilidade, com o intuito de fornecer informações precisas e impactantes aos gestores públicos. A detecção dessas áreas, apesar de sua natureza higienista, não tem o propósito de desabrigar as pessoas de suas casas sem qualquer amparo. Pelo contrário, o objetivo é equipar os tomadores de decisão com dados concretos para que possam implementar ações que realmente melhorem a condição de vida das populações vulneráveis e, assim, ser possível melhorar o futuro da capital do Brasil.

1.2 Justificativas

A tecnologia pode ser uma aliada bastante poderosa quando o foco é detectar e mitigar a vulnerabilidade humana em uma localidade. Primeiro, as técnicas de aprendizagem profunda podem contribuir para diversas frentes da esfera da sustentabilidade como no planejamento urbano, segurança alimentar, conservação da biodiversidade e detecção de resíduos em locais inadequados (Salas; Patterson; Vidal, 2022). Nesse sentido, a aprendizagem profunda associada à detecção da vulnerabilidade pode ajudar a identificar áreas de risco e suscetíveis a desastres naturais, auxiliar na análise da distribuição eficiente de recursos, identificar ameaças a ecossistemas e espécies e detectar áreas com a presença de resíduos que podem degradar o meio ambiente.

Em segundo lugar, a construção de novas bases de dados públicas e anotadas fermenta novas oportunidades de estudo, sendo de grande utilidade para próximos trabalhos na área e para a expansão desses estudos no Brasil. Portanto, este trabalho é um passo a mais para impactar positivamente a sociedade no contexto da vulnerabilidade humana por meio da aplicação de técnicas de inteligência artificial.

1.3 Objetivos

Neste trabalho, buscou-se atingir os objetivos geral e específicos descritos a seguir.

1.3.1 Objetivo Geral

Este trabalho tem como objetivo geral avaliar a capacidade de modelos de segmentação que utilizam redes neurais profundas para a detecção de áreas de vulnerabilidade humana a partir de imagens de satélite públicas e do desenvolvimento de duas bases de dados inéditas.

1.3.2 Objetivos Específicos

Este trabalho tem como objetivos específicos:

- Construção de uma base de dados: coleta, desenvolvimento e disponibilização de uma base de dados com imagens geoestacionárias com entidades classificadas nas categorias vulnerabilidade humana, resíduos depositados inadequadamente e áreas consideradas tradicionais.
- Definição de um modelo de treinamento: pesquisa, compreensão e teste de diferentes modelos convolucionais de aprendizagem profunda para segmentação de imagens. Para este objetivo, é necessário avaliar as estratégias de treinamento disponíveis e os parâmetros adequados para cada modelo.

- Avaliação e ajustes dos modelos escolhidos: verificação dos resultados obtidos para cada modelo e discussão exploratória dos resultados.

1.4 Organização do Manuscrito

Este trabalho está organizado nos seguintes capítulos. No Capítulo 2, apresenta-se a teoria fundamentadora deste estudo, incorporando-se um breve contexto histórico, conceitos básicos e específicos das tecnologias trabalhadas. No Capítulo 3 aborda-se a forma como a teoria será aplicada na prática a fim de se atingir os objetivos definidos anteriormente. Todo o processo prático do trabalho é descrito nessa etapa. Depois, no Capítulo 4, os resultados obtidos a partir das aplicações anteriores são apresentados e discutidos. Por fim, no Capítulo 5 apresentam-se as conclusões do trabalho.

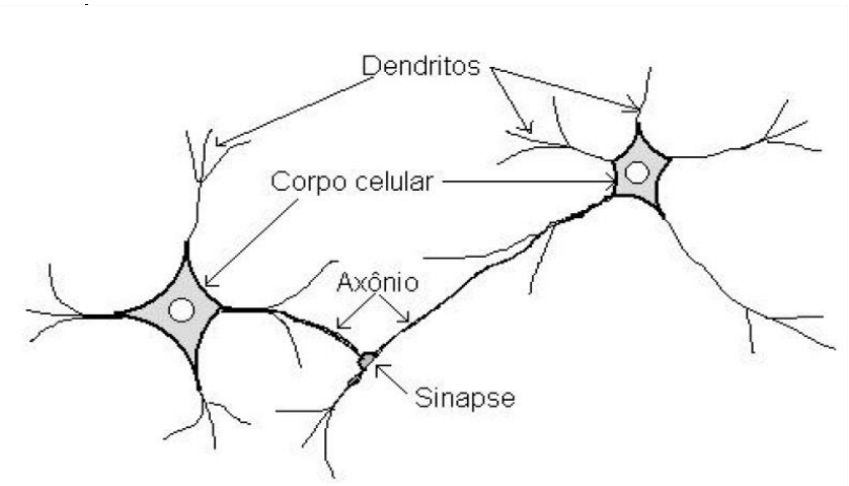
2 Fundamentação Teórica

Nesta seção, serão apresentados alguns conceitos teóricos fundamentais para a elaboração e entendimento da metodologia aplicada neste trabalho. O presente capítulo traz as principais definições a respeito de aprendizagem profunda, redes neurais convolucionais e suas aplicações em segmentação de imagens e detecção de objetos. Aborda-se, também, modelos de redes neurais totalmente conectadas, seus principais parâmetros e métricas utilizadas para análise dos resultados.

2.1 Redes Neurais Artificiais

Os seres humanos possuem um sistema nervoso composto por células denominadas neurônios que se conectam em ligações intituladas sinapses, conforme a Figura 2.1. Esse processo é responsável pela habilidade humana de raciocinar situações e responder a estímulos, sendo, portanto, fundamental para a aprendizagem.

Figura 2.1 – Representação de um neurônio biológico.

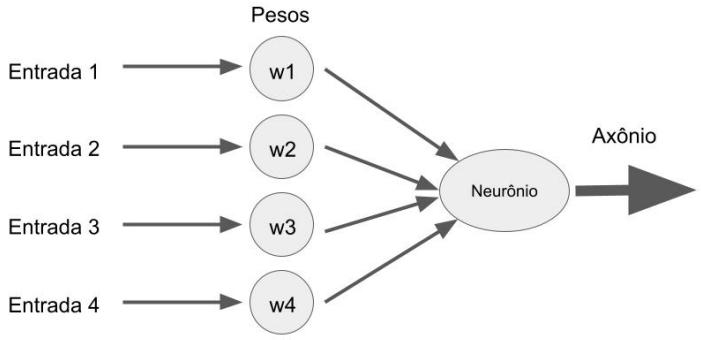


Fonte: ©2014 Alfredo BONINI NETO, Carolina dos Santos Batista BONINI (Neto; Bonini, 2014).

Baseado na ideia do funcionamento de neurônios humanos e buscando-se replicar resultados semelhantes em termos de aprendizagem, surgiram as primeiras ideias de redes neurais artificiais. Essas técnicas de aprendizado de máquina buscam simular o mecanismo de aprendizado em organismos biológicos. Conforme descrito por Aggarwal (2018), a teoria baseia-se em unidades computacionais que se conectam através de pesos que simulam a força das conexões de sinapse. Nesse sentido, a aprendizagem se daria pela alteração dos pesos a partir de "estímulos externos" como dados de treinamento. De acordo com a ideia

exposta acima, a arquitetura das Redes Neurais Artificiais poderia ser representada a partir de uma analogia com as redes neurais biológicas, conforme demonstra a Figura 2.2.

Figura 2.2 – Representação da arquitetura de uma rede neural artificial. Adaptado de (Aggarwal, 2018)



2.1.1 Breve Histórico

Como explicado em Goodfellow, Bengio e Courville (2016), a história das redes neurais tem seu início em 1943 com Warren McCulloch e Walter Pitts, em WS (1943), que se juntaram para desenvolver um neurônio artificial a partir de um modelo matemático. Essas ideias foram exploradas de forma mais ampla em 1949 por Donald Hebb. Em seu livro Hebb (1949), Donald apresenta a ideia de que as conexões neurais são fortalecidas por meio de usos progressivos.

Em seguida, em Rosenblatt (1958), Frank Rosenblatt prosseguiu com esse estudo a fim de desenvolver a primeira máquina capaz de aprender sem nenhum controle humano. Frank construiu um sistema de entrada e saída chamado *Perceptron*. Os pesos que envolviam esse sistema a fim de gerar uma saída a partir de uma entrada eram "aprendidos" pelo sistema a partir de entradas sucessivas, gerando saídas cada vez mais compatíveis com o resultado esperado.

Em 1959, Bernard Widrow e Marcian Hoff (Widrow e Hoff (1988)) desenvolveram os sistemas denominados ADALINE e MADALINE, o segundo sendo utilizado para a eliminação de ruídos em linhas telefônicas.

Em 1985, Paul Werbos publicou um relatório do seu trabalho sobre retro-propagação no MIT, Werbos (1990), o que re-estimulou os estudos em torno das redes neurais artificiais. A partir do trabalho de Werbos, diversos outros estudos foram desenvolvidos a fim de permitir a formação das redes neurais de hoje em dia.

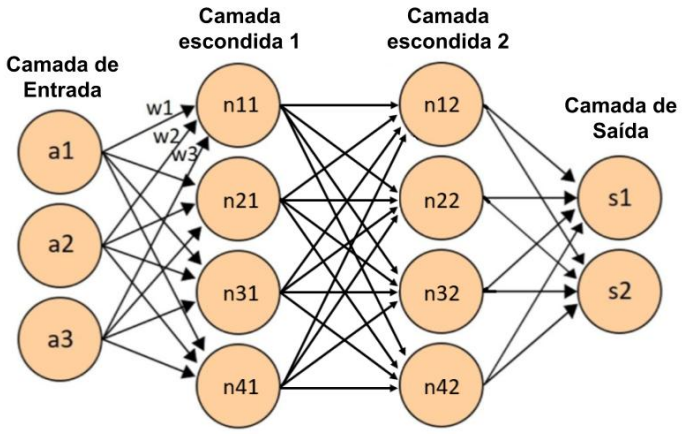
2.1.2 Conceitos Básicos Iniciais

A seguir serão apresentadas algumas ideias básicas a fim de permitir uma compreensão assertiva de conceitos posteriores.

2.1.2.1 Camadas de uma Rede Neural Artificial

A arquitetura de uma rede neural artificial é dividida em camadas. Tipicamente, tem-se pelo menos três camadas: uma de entrada, que contém os dados iniciais, uma ou mais camadas ocultas, onde os dados são computados, e uma camada de saída, em que os resultados são produzidos. As unidades são conectadas com diferentes resistências (ou pesos) de conexão, conforme a Figura 2.3.

Figura 2.3 – Camadas de uma rede neural artificial. Adaptado de (Aggarwal, 2018)



2.1.2.2 Função de Ativação

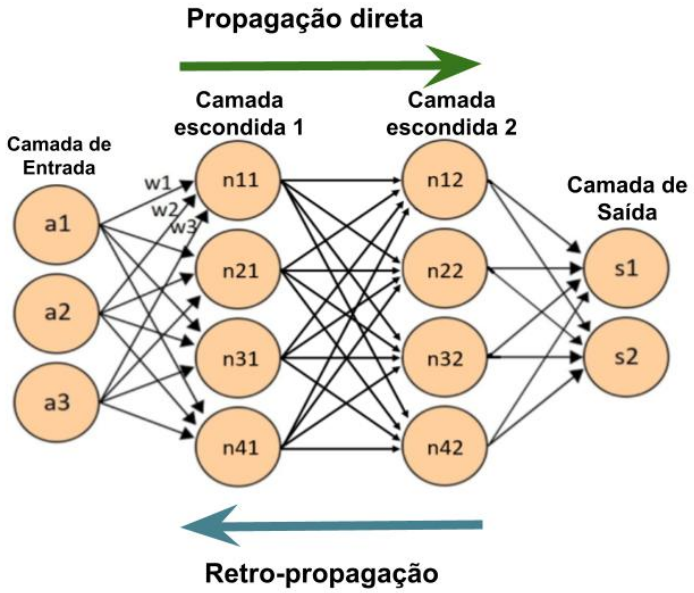
Durante o processamento dos dados de entrada, cálculos lineares ou não lineares são feitos nas camadas escondidas e de saída. Esses cálculos não lineares acontecem por meio das chamadas funções de ativação. Em suma, essas funções determinam quando um neurônio deve ou não ser ativado, permitindo a introdução da não-linearidade na rede neural. Geralmente, as camadas escondidas possuem a mesma função de ativação, porém essa pode variar em relação à camada de saída.

2.1.2.3 Propagação Direta e Retro-propagação

Analisando a forma como os dados fluem dentro da rede neural, observamos dois tipos distintos de movimento: a propagação direta e a retro-propagação, conforme ilustrado na Figura 2.4. O primeiro movimento se refere ao fluxo de dados da camada de entrada para a camada de saída, enquanto o segundo se refere ao caminho oposto, em que os dados de entrada são retro-alimentados com os valores obtidos da saída. A retro-propagação é

fundamental para melhorar a precisão da previsão por meio de derivadas calculadas com os dados de saída.

Figura 2.4 – Formas de propagação de dados em uma rede neural artificial.



2.1.2.4 Função de Perda

As redes neurais profundas, do inglês *Deep Neural Networks* (DNNs), utilizam as funções de perda para quantificar o desempenho da rede em uma tarefa específica e permitir que ela aprenda e se aprimore ao longo do tempo. Esse é um método de medir a discrepância entre as previsões feitas pela rede neural e os valores verdadeiros associados aos dados de treinamento. Nesse sentido, como explicado em Salas, Vidal e Martínez-Trinidad (2019), é possível atestar quantitativamente se o algoritmo implementado está, de fato, modelando a base de dados utilizada.

Sempre que são realizados treinamentos da base de dados, o objetivo é minimizar a perda entre os resultados previstos e os reais. Para isso, os pesos (ou valores de parâmetro) da rede neural são ajustados por meio de algoritmos de otimização. Os cálculos feitos para as funções de perda dependem do tipo de tarefa que a rede está realizando. No aprendizado supervisionado, existem dois tipos principais de funções de perda: as de regressão e as de classificação. A primeira é empregada em redes neurais de regressão, nas quais dado um determinado valor de entrada, o modelo prevê um valor de saída correspondente. Já a segunda função de perda age de tal modo que dada uma entrada, a rede neural produz um vetor de probabilidades dessa entrada pertencer a diversas categorias predefinidas. Assim, em seguida, a função de perda consegue selecionar a categoria com a maior probabilidade de pertencimento.

Sabe-se que para questões relacionadas a classificações de imagens, a função de perda mais comumente utilizada é a entropia cruzada (*cross-entropy*). Para o caso de existirem apenas duas classes de entidades, utiliza-se a entropia cruzada binária, já quando existem 3 ou mais classes, utiliza-se a entropia cruzada categórica.

2.1.2.5 *Overfitting* e *Underfitting*

O correto ajuste do modelo aos dados de treinamento é uma etapa fundamental para se obter resultados adequados de um algoritmo de aprendizagem profunda. Modelos interessantes são aqueles complexos o suficiente para poder produzir uma generalização do aprendizado, porém sem gerar vieses. Ao mesmo tempo, modelos extremamente simplificados tendem a gerar resultados escassos. Essas duas situações constituem o sobre-ajuste e o sob-ajuste e são problemas muito comuns no estado de treinamento, conforme abordado por Smith (2018).

Quando um modelo se torna capaz de se ajustar minuciosamente aos dados de treinamento, diz-se que ocorreu um sobre-ajuste, ou do inglês *overfitting*, do modelo. O *overfitting* é uma condição inadequada para os modelos de aprendizagem profunda, uma vez que o modelo se torna muito acurado para aquele conjunto de dados de treinamento, mas há uma brusca redução na performance com os dados de teste, pois perde sua capacidade de generalização. Conforme revisado por Zhang, Zhang e Jiang (2019), apesar de os estudos clássicos relacionarem a alta complexidade de alguns modelos à ocorrência de sobre-ajuste, estudos mais recentes divergem dessa abordagem, mostrando a capacidade de generalização, mesmo em modelos mais complexos.

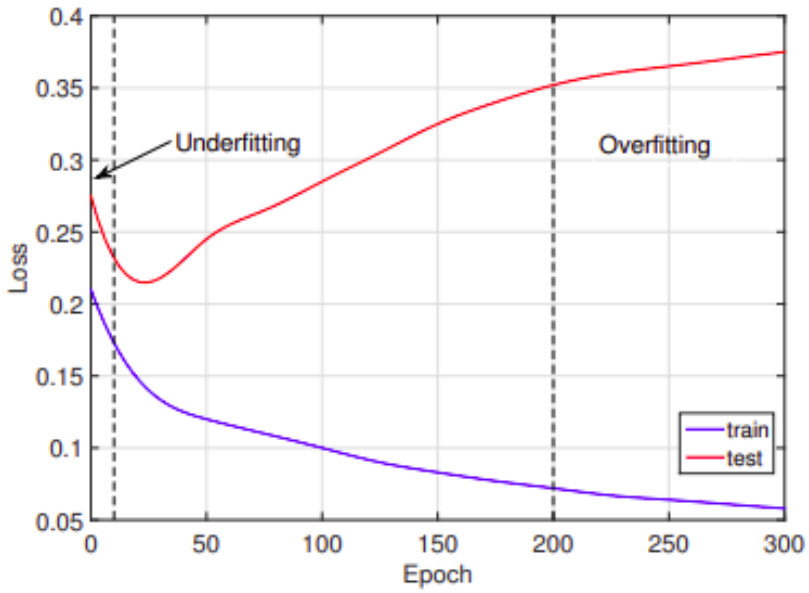
O sob-ajuste, ou do inglês *underfitting*, por sua vez, ocorre quando o modelo apresenta resultados fracos para os dados de treinamento e também de teste. Smith (2018) associa esse problema à incapacidade do modelo de reduzir o erro para o conjunto de treinamento ou teste, não sendo possível atender às complexidades dos dados. Muitas vezes, essa situação está associada à pequena quantidade de dados de treinamento, impossibilitando o modelo de reconhecer as características da base de dados.

Zhang, Zhang e Jiang (2019) ainda define esses dois conceitos baseando-se na curva de perda. Nesse sentido, os autores descrevem a ocorrência de *underfitting* quando há incapacidade do modelo em obter erro suficientemente baixo nos dados de treinamento. Por sua vez, é descrita a ocorrência de *overfitting* quando há uma grande diferença entre o erro de treinamento e de teste. Ambas as situações podem ser exemplificadas na Figura 2.5.

2.1.3 Segmentação de Imagens Utilizando Modelos Profundos

A segmentação de imagens, uma técnica primordial da visão computacional, tem sido revolucionada pela ascensão das redes neurais profundas. Nesse sentido, Goodfellow, Bengio

Figura 2.5 – Exemplo de problemas de *overfitting* e *underfitting*.



Fonte: ©2019 IEEE (Zhang; Zhang; Jiang, 2019).

e Courville (2016) mostra como as redes profundas são ferramentas bastante poderosas na análise, compreensão e, em especial, na segmentação de imagens complexas. Essa área de segmentação baseia-se na fragmentação da imagem como forma de separar as regiões de interesse e poder, por exemplo, realizar a detecção de determinado objeto, conforme será explicado em detalhes nesta seção.

2.1.3.1 Redes Neurais Convolucionais (CNNs)

Antes de tudo, dentro do contexto das Redes Neurais Artificiais, tem-se as Redes Neurais Convolucionais ou CNNs, do inglês *Convolutional Neural Networks*, que consistem em um tipo específico de modelo de aprendizado de máquina projetado para processar dados com uma estrutura de grade, como imagens ou dados de séries temporais, conforme exposto em Goodfellow, Bengio e Courville (2016). O nome dessa rede foi inspirado justamente nos neurônios animais reais, além de significar o emprego da operação matemática chamada de convolução.

Em LeCun Léon Bottou e Haffner (1998), os princípios dessa rede são apresentados através do modelo LeNet-5, considerado um marco muito importante na história das CNNs. O LeNet-5 foi projetado para reconhecimento de dígitos manuscritos e foi amplamente utilizado em sistemas de reconhecimento de caracteres. Com isso, LeCun e seus colaboradores desenvolveram e introduziram técnicas bastante relevantes para o treinamento eficiente das CNNs, como a ideia de compartilhamento de pesos.

No contexto das redes convolucionais, a convolução é uma operação discreta definida

como evidenciado na Equação 2.1:

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a), \tag{2.1}$$

onde t assume apenas valores inteiros, o argumento x é chamado de entrada, o w de *kernel* e a saída $s(t)$ de mapa de características, do inglês *feature map*.

Geralmente, a entrada é um vetor multidimensional de dados e o *kernel* é um vetor multidimensional de parâmetros que são ajustados de acordo com o algoritmo de aprendizado. Esses vetores multidimensionais recebem o nome de tensores.

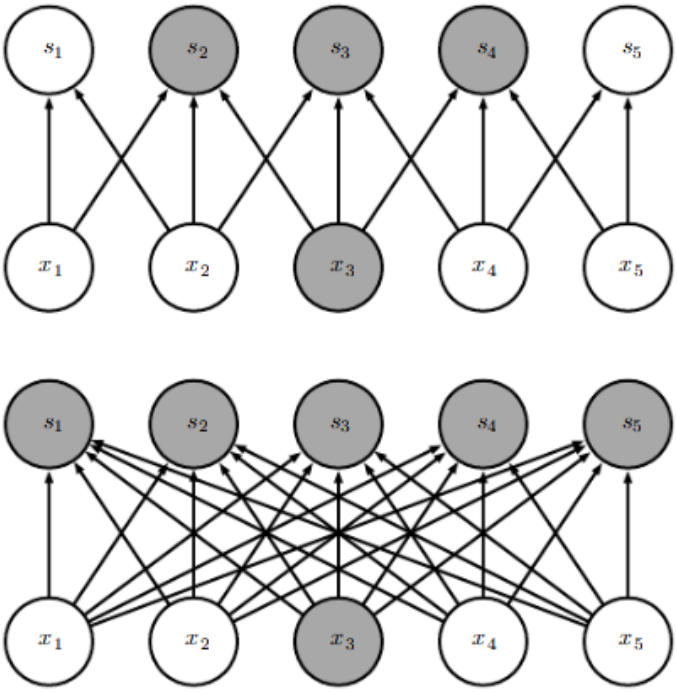
A convolução é, portanto, fundamental para a extração de características locais e a preservação da estrutura espacial das imagens, permitindo que as redes aprendam padrões visuais importantes como texturas, bordas e formas diferentes. Ademais, Goodfellow, Bengio e Courville (2016) explica que sua utilização implica em três conceitos bastante importantes quando o assunto é melhorar um sistema de aprendizado de máquina: interações esparsas, compartilhamento de parâmetros e representações equivariantes.

Sabe-se que as camadas de redes neurais tradicionais utilizam uma multiplicação matricial na qual cada unidade de saída interage com cada unidade de entrada. Já as redes convolucionais funcionam de forma distinta, possuindo, no geral, interações esparsas, também conhecidas como conectividade esparsa ou pesos esparsos. Isto é possível por meio do emprego de um *kernel* menor que a entrada. Desse modo, é necessário armazenar menos parâmetros, reduzindo, então, os requisitos de memória do modelo e a quantidade de operações para o cálculo da saída, melhorando sua eficiência estatística.

A Figura 2.6 retrata uma interação esparsa vista de baixo. É feita a representação da diferença entre uma rede com interação esparsa (parte superior da figura) e uma rede formada por multiplicação matricial (parte inferior da figura). Percebe-se que quando a saída s é obtida por meio de uma interação esparsa, apenas três saídas são afetadas por x. Diferentemente do que acontece com a saída s em redes tradicionais, nas quais a conectividade deixa de ser esparsa e, portanto, todas as saídas são afetadas por x.

Ao falar-se da utilização do mesmo parâmetro para mais de uma função em um determinado modelo, refere-se ao conceito de compartilhamento de parâmetros. Nas redes neurais tradicionais, ao computar a saída de uma camada, cada elemento da matriz de pesos é usado apenas uma vez. No que diz respeito às redes neurais convolucionais, a situação muda. Cada membro do *kernel* é usado em todas as posições de entrada, salvo determinados *pixels* nas bordas de imagens, que podem depender das decisões de projeto tomadas com relação a essas bordas. Esse compartilhamento de parâmetros utilizado pela convolução implica no aprendizado de apenas um conjunto, em oposição ao aprendizado de um conjunto separado de parâmetros para cada localização. Isso reduz de forma significativa os requisitos de armazenamento do modelo.

Figura 2.6 – Exemplo de interação esparsa vista de baixo na parte superior e exemplo de interação não esparsa na parte inferior da figura.



Fonte: ©2016 MIT Press (Goodfellow; Bengio; Courville, 2016).

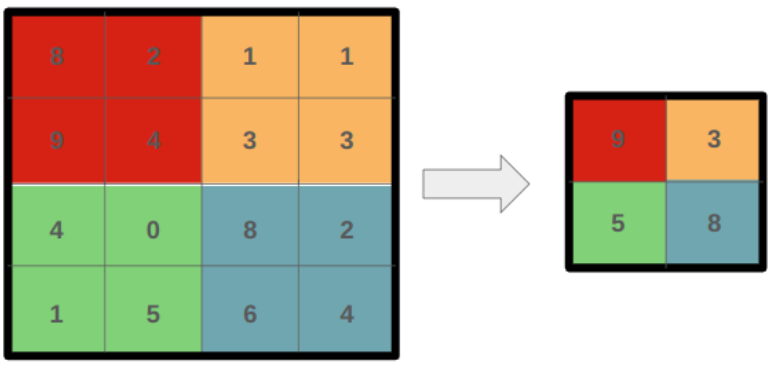
Na convolução, o compartilhamento de parâmetros faz com que a camada possua uma propriedade chamada equivariância à translação. Falar sobre equivariância significa dizer que, se a entrada de uma função muda, a saída também mudará da mesma maneira. Ao mover-se um evento mais tarde na entrada, a mesma representação dele aparecerá na saída, contudo mais tarde. Assim, quando o assunto é imagem, a convolução cria um mapa bidimensional no qual apenas certos recursos aparecem na entrada e, ao mover-se o objeto na entrada, sua representação se moverá na mesma quantidade na saída. Naturalmente, a convolução não é equivalente a todos os tipos de transformações, como mudanças na escala ou rotação de uma imagem, logo, para tratar esses casos, deve-se utilizar outros mecanismos.

Além da camada de convolução, as CNNs também costumam ter uma camada denominada *pooling*. Nesta camada ocorre o processo de subamostragem, ou seja, a redução das dimensões espaciais dos *feature maps*, sem se desfazer das informações primordiais. Essa operação utiliza de um filtro 2x2 que é processado sobre cada canal do *feature map* e é fundamental para a redução de parâmetros e cálculos na rede, controlando a complexidade do modelo.

A Figura 2.7 representa um exemplo de aplicação de uma das funções que podem ser realizadas na camada de *pooling*. A função *Max Pooling* consiste em selecionar o elemento de maior valor dentro da região coberta pelo filtro binário.

No geral, as CNNs oferecem uma abordagem poderosa e eficaz para a análise de

Figura 2.7 – Exemplo da operação *Max Pooling* realizada na camada de *pooling*.



imagens e problemas de visão computacional. A capacidade que elas têm de aprender automaticamente características relevantes, preservar a estrutura espacial das imagens e lidar com invariâncias tornam-nas altamente adequadas para uma ampla gama de aplicações, desde classificação e detecção de objetos até segmentação de imagens e reconhecimento facial. Nesse sentido, em seguida será explicado com mais detalhes sobre as aplicações das CNNs relevantes para este trabalho.

2.1.3.2 Redes Neurais Convolucionais para Segmentação de Imagens

A segmentação de imagens consiste no processo de identificação e separação de objetos de interesse em uma imagem, de forma a obter-se uma representação compacta deles (Barros. *et al.*, 2022; Abade *et al.*, 2022). De acordo com Forsyth e Ponce (2002), os estudos iniciais a respeito da segmentação de imagens se basearam na forma humana de percepção e agrupamento de imagens, isto é, o raciocínio humano para identificar imagens a partir da conexão de elementos seguindo critérios como proximidade, similaridade, paralelismo e regiões em comum.

Partindo da ideia anterior, torna-se claro que o contexto é um fator essencial para a segmentação de imagens. Nesse sentido, faz-se necessário ressaltar a ideia de "fundo de imagem", que, neste cenário, trata-se de todas as frações da imagem que não são objetos de interesse. Por mais que essas regiões não sejam saída de um algoritmo de segmentação de imagem, elas são fundamentais para o processo de identificação dos elementos buscados.

Desde as primeiras ideias de segmentação de objetos, o conceito de "fundo de imagem" se fez presente, inicialmente, para constituir os chamados algoritmos de subtração de fundo de imagem, vastamente utilizados em aplicações de vídeo. Esses algoritmos se baseiam na subtração do fundo de imagem entre os diversos *frames* de um vídeo, segmentando os objetos de interesse entre os *frames*.

Outro fator fundamental e bastante explorado desde os primeiros estudos são as bordas dos objetos. Forsyth e Ponce (2002) aborda algumas técnicas de detecção de bordas

baseadas em diferença de *frames*, histogramas, comparação de blocos e diferença de arestas.

Conforme revisado por Ali, Kako e Abdi (2022), as técnicas de segmentação de imagem foram se aprimorando ao longo dos anos. Inicialmente, os métodos aplicados consistiam em pacotes baseados em histogramas e crescimento de região, de forma a se identificar as bordas de objetos em imagens. Hoje, no entanto, a segmentação baseada em algoritmos de aprendizagem profunda é cada vez mais explorada.

2.1.4 Segmentação de Imagens para Detecção de Objetos

Uma forte aplicação de segmentação de imagens é a detecção de objetos. As técnicas de detecção de objetos, inicialmente, se baseavam em histogramas de gradientes orientados e métodos rudimentares de aprendizagem de máquina. Com o avanço da tecnologia, as técnicas de aprendizagem profunda passaram a dominar o escopo de detecção de imagens pela sua acurácia e performance.

Zhiqiang e Jun (2017) categoriza os modelos de aprendizagem profunda para detecção de objetos em detecções de dois estágios e de um estágio. Na primeira, o modelo utiliza uma etapa para gerar propostas de regiões ou objetos e uma segunda etapa para classificação e detecção dessas propostas, enquanto na segunda categoria, o modelo realiza a geração das regiões de interesse, detecção e classificação em uma única etapa.

Dentre os modelos que se enquadram na primeira categoria, há, por exemplo, as Redes Neurais Convolucionais Baseadas em Região, ou do inglês *Region Based Convolutional Neural Networks* (R-CNN). Sendo o primeiro modelo dessa categoria, ele ainda é utilizado atualmente, porém apresenta limitações quanto as dimensões das imagens de entrada, uma vez que o tamanho deve ser fixo. Derivado da R-CNN, surgiu o modelo *SPP-Net*, que uniu a técnica anterior ao agrupamento de pirâmide espacial, ou do inglês *Spatial Pyramid Pooling (SPP)*, esse método solucionou o problema anterior, permitindo o uso de imagens de entrada de dimensões variáveis. Ainda, vale citar o modelo *Fast* R-CNN, que trouxe vantagens comparativas em relação aos dois modelos anteriores em termos de performance e custo.

Em relação aos modelos de estágio único, a *You Only Look Once* (YOLO) é um forte representante da categoria e será abordada detalhadamente no tópico a seguir. Ainda podemos citar os modelos Detecção de Disparo Único (*Single Shot multibox Detection* - SSD) e RetinaNet.

2.1.4.1 Principais Modelos

Nesta seção, serão abordados alguns dos principais modelos de redes neurais convolucionais para segmentação de imagens, com foco em duas categorias principais: os modelos com arquitetura *encoder-decoder* e modelos totalmente conectados. Para a primeira categoria,

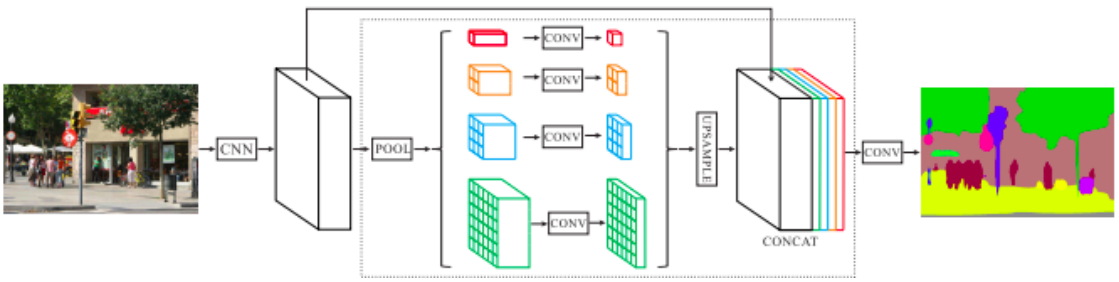
haverá destaque para o modelo PSPNet, enquanto para a segunda categoria, o foco será a YOLO.

2.1.4.1.1 Arquitetura *Encoder-Decoder*

Em Iakubovskii (2019), o autor constrói uma biblioteca que compila 9 das principais arquiteturas de modelos para segmentação binária e multi-classe. Os modelos definidos pelo autor são: Unet (Ronneberger; Fischer; Brox, 2015), Unet++ (Zhou *et al.*, 2018), MAnet (Fan *et al.*, 2020), Linknet (Chaurasia; Culurciello, 2017), FPN (Kirillov *et al.*, 2017), PSPNet (Zhao *et al.*, 2017), PAN (Li *et al.*, 2018), DeepLabV3 (Chen *et al.*, 2017) e DeepLabV3+ (Chen *et al.*, 2018).

Neste ponto, a arquitetura do modelo *Pyramid Scene Parsing Network* (PSPNet) será mais aprofundada. Esse algoritmo é apresentado em Zhao *et al.* (2017) e faz o uso de uma técnica denominada módulo de análise de pirâmide, do inglês *pyramid parsing module*. Essa técnica consiste em processar a imagem de entrada em diferentes tamanhos de kernel pelas camadas de convolução e *pooling*, conforme pode ser observado nas camadas de convolução representadas na Figura 2.8. Esse método permite que a imagem seja analisada tanto em nível detalhado, quanto em contexto geral, ocasionando ganhos significativos no processo de classificação de objetos.

Figura 2.8 – Exemplo da implementação da análise de pirâmides na arquitetura do PSPNet.



Fonte: ©2017 IEEE (Zhao *et al.*, 2017).

O PSPNet utiliza de um tipo de arquitetura comum para algoritmos de segmentação de imagens: a *Encoder-Decoder*. Neste tipo de arquitetura, o *encoder* constitui a etapa de extração de características da imagem, enquanto o *decoder* é responsável por prever a classe do pixel no final. No seu *encoder*, o PSPNet faz uso de redes convolucionais dilatadas e da análise de pirâmides, com o intuito de absorver informações de cenário ao mesmo tempo em que possibilita um foco em detalhes. Para a sua etapa de *encoder*, o PSPNet pode ser combinado com diversas famílias de redes de *encoders* como a ResNet e a EfficientNet.

Em Iakubovskii (2019), o autor também faz uma seleção das principais famílias de *encoders* que são suportadas pela API construída. Esses *encoders* são: ResNet (He *et al.*,

2016), ResNeXt (Xie *et al.*, 2017), ResNeSt (Zhang *et al.*, 2020), Res2Net (Gao *et al.*, 2021), RegNet(x/y) (Radosavovic *et al.*, 2020), GERNet (Lin *et al.*, 2020), SE-Net (Hu *et al.*, 2019), SK-ResNet (Li *et al.*, 2019), DenseNet (Huang *et al.*, 2018), Inception (Szegedy *et al.*, 2014), EfficientNet (Tan; Le, 2020), MobileNet (Howard *et al.*, 2017), DPN (Chen *et al.*, 2017), VGG (Simonyan; Zisserman, 2015), Mix Vision Transformer (Chen *et al.*, 2021) e MobileOne (Vasu *et al.*, 2023)

A ResNet, do inglês, *Residual Network*, foi inicialmente apresentada em He *et al.* (2016) e consiste em uma arquitetura de rede neural convolucional que implementa uma quantidade grande de camadas. Uma de suas maiores contribuições foi a apresentação de um *framework* de aprendizado residual com o intuito de melhorar a performance de arquiteturas com muitas camadas e evitar o problema de *vanishing gradient*, isto é, a perda do sinal de entrada ao longo das diversas camadas. O aprendizado residual consiste em treinar a rede de acordo com a diferença entre o sinal de entrada e o de saída para cada camada, isto é, o resíduo, em contraponto com o treinamento de acordo com a saída desejada.

A ResNet possui diversas variantes que são denominadas de acordo com o seu número de camadas, ou seja, a ResNet-50 implica que para essa arquitetura são utilizadas 50 camadas.

Ainda, vale ressaltar a EfficientNet, uma arquitetura explicada em Tan e Le (2020) que apresenta uma técnica de escala uniformizada para cada dimensão da rede (profundidade, largura e resolução), a partir de um coeficiente composto. A técnica se baseia no balanceamento das dimensões da rede a partir de escalas fixas para cada dimensão que são calculadas a partir do coeficiente composto, permitindo, assim, que o modelo seja escalado uniformemente. Nesse sentido, para o aumento em 2^N do uso computacional, deve-se aumentar a profundidade da rede em α^N, a largura em β^N e a resolução em γ^N, em que α, β e γ são coeficientes fixos.

2.1.4.1.2 Redes Neurais Totalmente Conectadas para Segmentação de Imagens

Dentro do contexto de detecção de imagens, existem as redes totalmente conectadas como a *You Only Look Once (YOLO)*. A YOLO surgiu a partir do desenvolvimento de Redmon *et al.* (2015) e utiliza como espinha dorsal o *framework Darknet*. Indo de encontro aos métodos de RPN (*Region Proposal Network*) e pipelines complexos comumente utilizados em abordagens como a *Region Based Convolutional Neural Networks (R-CNN)*, a YOLO se destacou por sua rapidez e acurácia em comparação com as técnicas anteriores (Barreto; Lambert; Vidal, 2019).

O acrônimo YOLO originou-se da ideia de uma técnica baseada no uso de uma rede neural para analisar uma imagem e todos seus objetos uma única vez. Nesse sentido, sua metodologia consiste em analisar o problema de detecção de imagens como um problema de regressão, de forma que, uma única CNN consiga, simultaneamente, prever e classificar os objetos definidos pelas regiões de interesse retangulares, ou do inglês *bounding boxes*.

O algoritmo atua segmentando a imagem de entrada em uma grade SxS, desse modo, o centro de um objeto pode estar dentro de uma das células da grade, fazendo com que essa célula seja responsável pela identificação do objeto. Cada célula é avaliada e, caso haja um objeto associado à célula em análise, constrói-se um vetor composto por 5 parâmetros: as distâncias (x e y) do centro da caixa em relação aos limites da célula da grade, a altura e largura em relação a toda a imagem e o grau de confiança. Por fim, adiciona-se a probabilidade de o objeto pertencer a uma determinada classe. O conjunto de análise de todas as células resulta em um tensor de previsões.

Ainda, a detecção é otimizada, uma vez que a imagem inteira é analisada. Desse modo, a classificação leva em consideração informações de contexto, reduzindo significativamente erros situacionais em comparação com métodos como *Fast* R-CNN.

Alguns anos depois, em 2016, os autores apresentaram uma segunda versão da YOLO, a YOLOv2 por Redmon e Farhadi (2016). A ideia principal dessa nova versão era aumentar a acurácia do modelo e diversificar as classes capazes de serem identificadas, garantindo, ainda, a rapidez da ferramenta. Desse modo, os autores investiram em estratégias como a clusterização de dimensões das regiões de interesse retangulares, treinamento multi-escala, aplicação de caixas de âncora ou *anchor boxes* e classificação hierárquica. Vale destacar que as *anchor boxes* representam regiões retangulares pré-definidas usadas como referência para a predição dos objetos e servem como ponto de partida para o modelo chegar às *bounding boxes* finais.

A partir dessas técnicas, os autores foram capazes de melhorar a capacidade da rede em aprender a ajustar melhor as dimensões das regiões de interesse retangulares e fazer melhores previsões. Ainda, foi possível treinar o modelo para lidar com diferentes tamanhos de imagens de entrada e combinar bases de dados de detecção e classificação. Todos esses fatores contribuíram para o aumento da acurácia do modelo e otimização da detecção e classificação, reduzindo a lacuna entre essas duas funções e tornando o modelo apto a classificar mais de 9000 classes

Mais tarde, a YOLOv3 surgiu em Redmon e Farhadi (2018) com melhorias em relação à detecção de pequenos objetos. Depois, a YOLOv4 foi lançada em Bochkovskiy, Wang e Liao (2020), que introduziu técnicas para melhorar a precisão da detecção, como a reorganização do *framework* de detecção em diferentes partes. A versão 5, por sua vez, foi estruturada por outros autores, mas não há um artigo publicado especificando seu desenvolvimento. Ainda, foi lançada a YOLOv6 em Li *et al.* (2022), que ampliou as aplicações industriais e, finalmente, tem-se a YOLOv7.

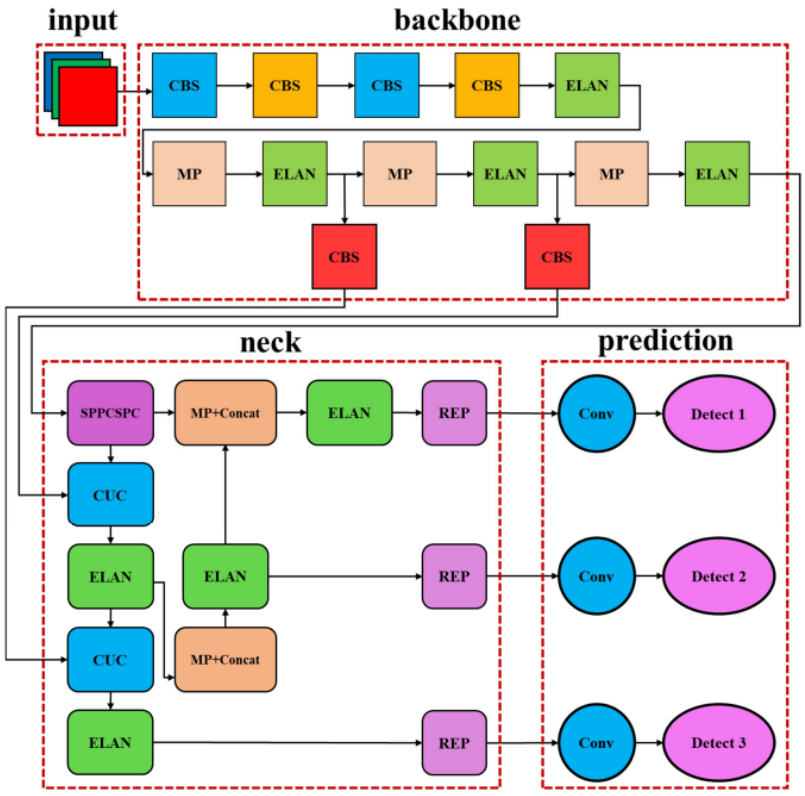
Em Wang, Bochkovskiy e Liao (2022), que trata da versão 7 da YOLO, os autores focaram no desenvolvimento de métodos de treinamento com intuito de aumentar a acurácia de detecção, sem interferir no custo de treinamento. Além disso, os autores expuseram e apresentaram métodos para lidar com os problemas associados à técnica de re-parametrização

de modelos e à estratégia de atribuição de rótulos com diferentes camadas de saída. Por fim, os autores propuseram métodos capazes de economizar parâmetros e custo computacional para o treinamento, associando essas melhorias ao que foi chamado de *Bag of Freebies*.

A arquitetura da YOLO pode ser divida em três componentes principais: a cabeça, o *backbone* e o pescoço. O processamento se inicia no *backbone*, onde os dados de entrada são inseridos e os parâmetros fundamentais das imagens são identificados nas camadas convolucionais. Em seguida, no pescoço ou, do inglês, *neck*, esses parâmetros são processados em redes totalmente conectadas, a fim de se obter as previsões para as coordenadas das *bounding boxes* e as classificações. O último estado é a cabeça, em que podem ser aplicadas técnicas de transferência de aprendizado ou *transfer learning*, conceito que será abordado em maior detalhes em tópicos à frente.

A seguir, a Figura 2.9 ilustra a arquitetura da YOLOv7. Como parte da sua arquitetura, tem-se um *backbone* denominado *Extended Efficient Layer Aggregation* (E-ELAN), que consiste em uma versão modificada da estrutura ELAN . Essa arquitetura modificada usa convolução de grupo para expandir os canais e a cardinalidade do bloco computacional. Além disso, na YOLOv7, a profundidade e a largura possuem fatores de escala definidos na arquitetura do modelo, o que favorece o chamado dimensionamento de modelo.

Figura 2.9 – Ilustração da arquitetura da YOLOv7.



Fonte: (Zhao *et al.*, 2017).

2.2 Hiper-parâmetros de Treinamento

Conhecer e entender os hiper-parâmetros que interferem no processo de treinamento de um modelo é fundamental para a obtenção do ajuste ótimo do modelo. Esses parâmetros geralmente são definidos durante a etapa de treinamento e influenciam diretamente na resposta do modelo em termos de acurácia, tempo de processamento, recursos necessários e projeção da curva de perda. Nesse sentido, serão apresentados a seguir os principais hiper-parâmetros analisados e utilizados neste trabalho.

2.2.1 Taxa de Aprendizagem

A taxa de aprendizagem consiste em um coeficiente capaz de determinar a velocidade com a qual a rede neural aprende e atualiza os parâmetros internos. Busca-se ajustar esse valor para que seja grande o suficiente para evitar o sob-ajuste e acelerar o treinamento, mas também pequena o suficiente para evitar divergência.

Vários pesquisadores estudam a melhor forma de se encontrar o valor ótimo da taxa de aprendizagem. Nesse sentido, Smith (2018), por exemplo, aborda algumas formas de se fazer uma escolha mais eficiente desse parâmetro.

2.2.2 Tamanho de Lote

O tamanho de lote, do inglês *batch size*, é o parâmetro de treinamento que define a quantidade de amostras a serem trabalhadas em uma única iteração, ou seja, previamente à atualização dos parâmetros internos do modelo. Esse parâmetro é limitado pela memória do *hardware*, dificultando o seu ajuste ótimo em alguns casos.

Diversos estudos abordam a relação entre o tamanho de lote e a taxa de aprendizagem, de forma a se permitir combinações de ajustes desses parâmetros para o melhor resultado em performance e tempo. Em Smith *et al.* (2018), por exemplo, os autores defendem o aumento do *batch size* em detrimento da diminuição da taxa de aprendizagem obtendo-se os mesmos resultados de treinamento. Smith (2018), por outro lado, ressalta a limitação de memória em torno do tamanho de lote, demonstrando ser mais viável, em determinadas situações, diminuir a taxa de aprendizagem, evitando-se problemas de recursos.

2.2.3 Épocas

As épocas se referem ao número total de iterações sobre todos os dados de treinamento em um ciclo. Cada época contém etapas de atualização de parâmetros internos do modelo. Nesse sentido, realizar a escolha correta do número de épocas, alinhado com os resultados das funções de perda, é fundamental para se evitar o sobre-ajuste e o sob-ajuste.

2.3 Métricas para Avaliação dos Resultados

É importante escolher métricas e parâmetros adequados para avaliar a performance de um modelo de aprendizagem profunda, de forma a se compreender seu desempenho e fazer os ajustes necessários para o desenvolvimento do algoritmo. No campo de detecção de objetos, há situações como a da Figura 2.10, por exemplo, em que se nota que a marcação do algoritmo, em vermelho, é diferente da marcação ideal, em verde, mas não necessariamente, o algoritmo está performando incorretamente, pois essa avaliação depende das referências definidas. Nesse sentido, serão apresentadas, a seguir, as principais métricas utilizadas neste projeto.

Figura 2.10 – Exemplo de marcações em detecção de objetos.

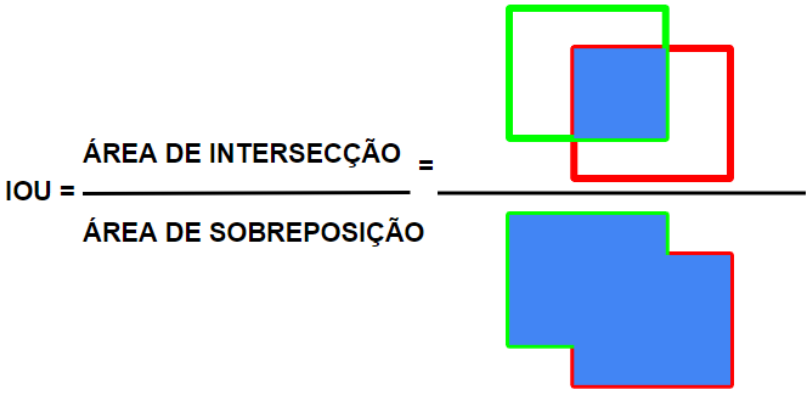
Inicialmente é necessário ressaltar alguns conceitos básicos que envolvem as métricas abordadas. Conforme apresentado por Padilla, Netto e Silva (2020), em algoritmos de detecção de objetos é comum se definir a Intersecção sobre a União, ou *Intersection over Union (IoU)*, isto é, o coeficiente de similaridade entre o valor previsto e o valor real, conforme representado na Figura 2.11. A classificação das detecções entre um objeto corretamente detectado, incorretamente detectado, corretamente não detectado e incorretamente não detectado se baseia nesse coeficiente.

Além disso, os conceitos de precisão e sensibilidade são fundamentais para o entendimento das métricas que serão abordadas. Precisão se refere a quantidade de previsões positivas corretas dentre todas as previsões positivas, conforme a Equação 2.2:

$$P = \frac{TP}{TP + FP}, \tag{2.2}$$

onde P é a precisão, TP se refere às previsões positivas verdadeiras (*true positive*) e FP às previsões falsas positivas (*false positive*).

Figura 2.11 – Representação da definição de *IOU*.



Enquanto isso, sensibilidade é a porcentagem de previsões positivas corretas dentre as previsões detectadas corretamente e não detectadas corretamente, conforme a Equação 2.3:

$$S = \frac{TP}{TP + FN}, \tag{2.3}$$

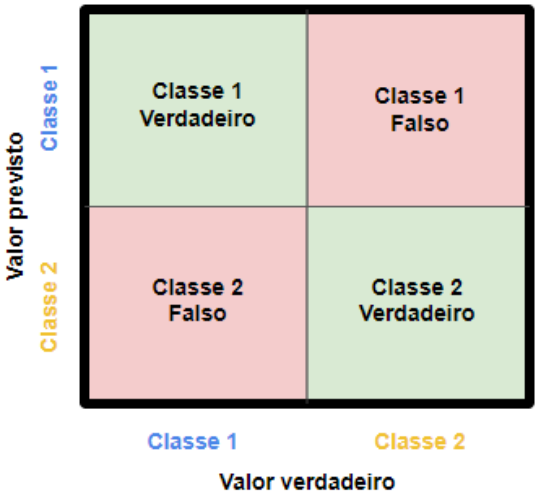
onde S é a sensibilidade, TP se refere às previsões positivas verdadeiras (*true positive*) e FN às previsões falsas negativas (*false negative*).

2.3.1 Matriz de Confusão

A matriz de confusão é uma ferramenta muito utilizada para avaliação de modelos de aprendizagem profunda, pois concatena em uma única tabela a correlação entre as previsões e os valores verdadeiros. Desse modo, é possível verificar a porcentagem de previsões corretas e incorretas e, em um cenário multi-classes, observar em quais classes há maior índice de confusão do modelo, ou seja, a porcentagem de previsões de objetos em classes erradas, conforme a Figura 2.12.

Para a detecção de objetos, além das classes definidas no modelo, a matriz de confusão apresenta ainda a comparação das previsões em relação às partes consideradas fundo de imagem, ou seja, que não estão associadas a nenhuma das classes. Desse modo, a matriz apresenta a porcentagem de previsões das classes para o fundo da imagem, ou seja, objetos que não deveriam ser classificados, mas foram, erroneamente. Na YOLOv7, essa porcentagem é representada na última coluna, chamada de *background* FP. De forma, análoga, ela apresenta também as previsões que não foram detectadas, e, portanto, foram consideradas objetos de fundo da imagem, mas, na verdade, pertenciam a alguma das classes. Na YOLOv7, essa porcentagem é representada na última linha, chamada de *background* FN.

Figura 2.12 – Representação de uma matriz de confusão padrão.



2.3.2 Curva de Precisão x Sensibilidade

Conforme abordado anteriormente, a precisão e a sensibilidade são parâmetros fundamentais para a avaliação da performance de um modelo. Nesse sentido, a curva precisão x sensibilidade é capaz de representar graficamente o desempenho do modelo e a variação da precisão conforme a sensibilidade vai se alterando para um certo valor de confiança. Idealmente, busca-se para uma precisão alta, conforme a sensibilidade aumenta.

2.3.3 *Mean Average Precision*

A *Mean Average Precision* ou *mAP* é uma métrica diretamente relacionada aos conceitos de precisão, sensibilidade e *IoU*, abordados anteriormente. O cálculo da *Average Precision* leva em consideração a tolerância permitida entre as caixas de demarcação preditas e verdadeiras, ou seja, a tolerância para qual o coeficiente obtido para uma determinada *IoU* é suficiente para ser considerada uma previsão correta ou incorreta. Nesse sentido, a *mAP* considera várias tolerâncias para o cálculo da *Average Precision* de cada classe e tira a média total, conforme a Equação 2.4:

$$mAP = \frac{1}{n} \sum_{k=1}^{k=n} AP_k, \tag{2.4}$$

onde n é o número de classes e k a classe em questão.

Neste trabalho, serão abordados duas diferentes métricas de *mAP*, a *mAP@0.5* e *mAP@0.5:0.95*. A primeira, utiliza como referência uma tolerância de *IoU* de 0.5 para o cálculo, enquanto a segunda, leva em consideração uma faixa de valores para o *IoU*, de 0.5 à 0.95, incrementando de 0.05.

2.3.4 F1 *Score*

A F1 *Score* ou F *Score* é uma métrica de avaliação que consiste em uma média harmônica entre os valores de precisão e sensibilidade, podendo ser representada pela Equação 2.5:

$$F1 = \frac{2PS}{P+S}, \tag{2.5}$$

onde P é a precisão e S a sensibilidade.

Fazendo-se uso da média harmônica, a F1 *Score* é capaz de produzir uma métrica balanceada em relação aos valores de precisão e sensibilidade, o que se torna vantajoso para a análise da acurácia do modelo.

Os valores para essa métrica variam de 0 a 1, sendo que quanto mais próximo de zero, pior a performance do modelo, demonstrando a dificuldade de se encontrar balanço entre precisão e sensibilidade. Por outro lado, quanto mais próximo de um, melhor é o desempenho do modelo em acertar verdadeiros positivos e falsos negativos.

2.4 Trabalhos Relacionados

A compreensão refinada do conceito de vulnerabilidade humana é relativamente recente e, portanto, ainda há muito o que ser explorado nesse campo de pesquisa. Assim, nesta seção, serão apresentados estudos que serviram de base para a construção e desenvolvimento deste trabalho. Nesse sentido, considerou-se utilizar como base artigos que abordam vulnerabilidade humana como conceito e artigos associados à detecção de objetos em imagens por meio da aprendizagem profunda.

O artigo de Salas, Patterson e Vidal (2022) apresenta um mapeamento sistemático das soluções de inteligência artificial aplicadas aos desafios de sustentabilidade na América Latina e no Caribe. Os autores destacam a utilização de IA em diversas áreas, como monitoramento ambiental, agricultura, detecção de áreas de vulnerabilidade humana, evidenciando a capacidade dessa tecnologia em possibilitar adaptação às mudanças climáticas. Assim sendo, esse estudo identifica que há um aumento significativo na precisão e na eficiência das análises ambientais ao usar-se técnicas de aprendizado profundo, permitindo que áreas vulneráveis fossem precocemente identificadas e medidas preventivas implementadas. Adicionalmente, o artigo aborda o conceito de vulnerabilidade humana como a exposição e interação das populações com ambientes frágeis e superexplorados, estimulando o estudo e a avaliação dessa vulnerabilidade por meio do uso de imagens de satélites.

Em Salas, Vidal e Martínez-Trinidad (2019) é feita uma abordagem geral e abrangente dos principais conceitos associados à aprendizagem profunda, além de descrever as principais arquiteturas de redes neurais, técnicas de treinamento e apresentar aplicações associadas

a cada arquitetura. Nesse contexto, a detecção de objetos em imagens é dada como uma aplicação bastante relevante do uso das redes neurais convolucionais ou CNNs.

As estratégias empregadas em Salas *et al.* (2021) revelam um bom desempenho médio atingido com o uso de CNNs na detecção de vulnerabilidade humana a nível nacional. Ao utilizarem imagens de satélite e indicadores obtidos a partir de dados do censo populacional do México de 2010, os autores desenvolveram um método para avaliar rapidamente a pobreza do México em detalhes e com ampla cobertura. Esse método fez o uso de CNNs e técnicas de aprendizado profundo para encontrar uma relação entre as características das imagens de satélites e os indicadores de vulnerabilidade do censo populacional, revelando a *EfficientNet* como a arquitetura com melhor performance quando comparada às demais e propiciando a oportunidade de explorar esse estudo a nível regional, estadual e municipal.

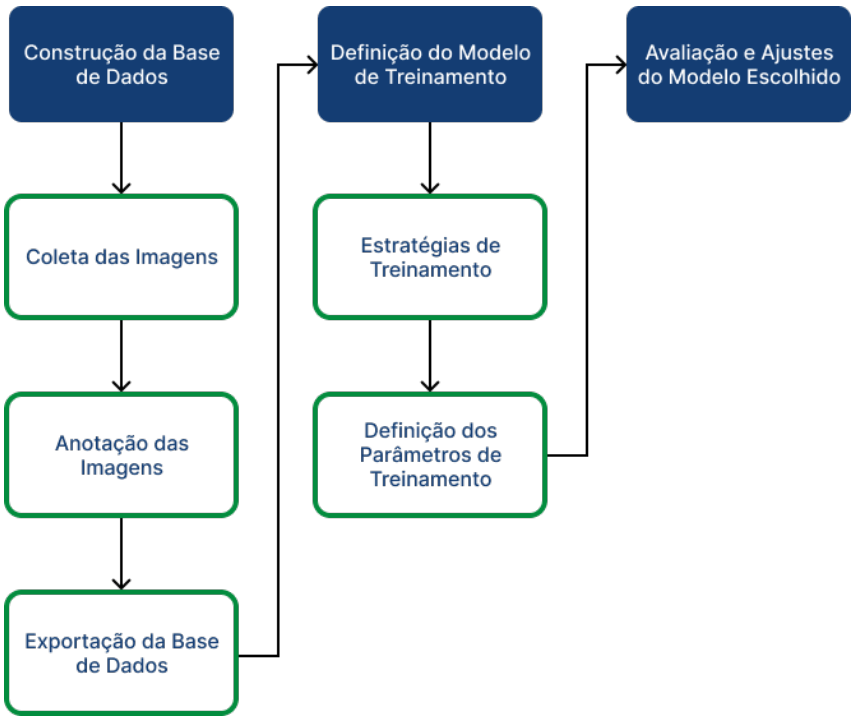
Assim sendo, vale destacar que o uso de CNNs para a detecção de objetos tem sido recorrente nos estudos recentes devido a sua ótima performance para realizar esta tarefa, o que sugere que as CNNs tornaram-se uma escolha natural quando trata-se de tarefas associadas a análises de imagens.

3 Metodologia

A metodologia adotada neste trabalho, a ser detalhada a seguir, seguirá a proposta do fluxograma apresentado na Figura 3.1. Inicialmente, será feita uma cuidadosa coleta de imagens públicas de satélites, ou seja, serão realizadas capturas de tela das regiões de estudo e será construída uma base de dados a partir da anotação de áreas de interesse em cada imagem. Em seguida, serão aplicadas técnicas de pré-processamento e serão definidas as estratégias e os parâmetros de treinamento. Por fim, será realizada a avaliação do modelo escolhido.

Portanto, este capítulo desenvolverá o processo de construção da base de dados, definição do modelo de treinamento com o melhor desempenho, além de todos os ajustes e validações necessários para definir o modelo mais adequado para a detecção de áreas de vulnerabilidade humana.

Figura 3.1 – Fluxograma base para a realização deste trabalho.



3.1 Construção da Base de Dados

Uma base de dados robusta e equilibrada é fundamental para um treinamento supervisionado de qualidade e uma validação com resultados satisfatórios. Uma base ampla permite que o modelo seja treinado de forma complexa, se atentando às diversas caracterísi-

cas das classes em análise e, desse modo, evitando-se o subajuste. Ao mesmo tempo, uma base equilibrada evita que o modelo seja fortemente influenciado por vieses, não permitindo a ocorrência do sobreajuste.

Não foi possível encontrar uma base de dados pública e consolidada de imagens de satélite durante o processo de revisão bibliográfica que retratassem zonas de vulnerabilidade e resíduos inadequados em centros urbanos, desse modo, optou-se por realizar a coleta e classificação das imagens de forma manual. Para a execução desse processo, os fatores quantidade e qualidade dos dados são determinísticos para o desenvolvimento do conjunto de dados utilizado neste projeto, de forma a se obter como resultado uma base sólida e significante. A seguir, esse processo será retratado com detalhes.

3.1.1 Coleta das Imagens de Satélite

Inicialmente, delimita-se as regiões geográficas de coleta das imagens. Como este trabalho trata-se de uma Prova de Conceito (POC) , optou-se por definir o escopo de coleta das imagens dentro da região do Distrito Federal. Esta escolha garante uma maior integridade dos dados e direcionamento dos esforços, contribuindo para uma validação mais consistente da ideia proposta. Serão coletadas imagens de diferentes regiões do Distrito Federal, englobando as Regiões Administrativas, porém, devido a limitação de recursos, não é possível garantir que todo o DF será abrangido.

Será utilizada a plataforma *Google Earth* (Google, 2023) como fonte de imagens de satélite deste trabalho por sua disponibilidade e fácil consulta, porém ressalta-se que a metodologia pode ser aplicada a outras fontes. A definição do escopo em torno do Distrito Federal, também é afetada pela escolha da plataforma, uma vez que, considerando a resolução proporcionada pelo *Google Earth* e a presença de áreas amplas no DF, identificou-se nesse cenário uma oportunidade de verificação dos dados coletados de forma facilitada.

Todas as imagens deverão ser rigorosamente coletadas a partir de capturas de telas na visualização 2D disponível e em escalas de 100 metros e 50 metros ajustadas a partir dos seletores da plataforma. Para obter-se imagens mais limpas e sem informações desnecessárias, serão desativadas as configurações de marcadores de locais, ruas e edificações.

3.1.2 Anotação das Imagens

Após a coleta das imagens, será utilizada a plataforma *Roboflow* (Dwyer, B., Nelson, J., Hansen, T., 2024) para a estruturação da base de dados. O *Roboflow* é uma aplicação fortemente utilizada no desenvolvimento de modelos de visão computacional e processamento de imagens por permitir o armazenamento das imagens em diferentes dimensões, anotação das imagens e exportação da base de dados. Nesse sentido, realiza-se o *upload* das imagens

de cada escala em um repositório, de forma a obter-se uma base de dados para cada escala, 50m e 100m.

Em seguida, o processo de categorização dos elementos na imagem para cada base de dados será desenvolvido seguindo três categorias ou classes: área de vulnerabilidade humana, resíduos inadequados e áreas urbanas tradicionais. As anotações serão realizadas de forma manual utilizando a opção *bounding boxes* (forma retangular de anotação) da plataforma para selecionar cada objeto. Para garantir uma maior confiabilidade nas anotações, pode ser utilizada a opção *Street View* do *Google Earth* para verificar as regiões com maior clareza de detalhes e realizar anotações mais assertivas.

Além disso, é importante enfatizar que objetos incompletos nas bordas das imagens não serão selecionados, pois tomou-se como premissa que esses objetos seriam capturados de forma completa em outras imagens do satélite. Nesse sentido, pode-se observar pela Figura 3.2, na escala de 50m, exemplos de objetos incompletos que não serão anotados nas imagens utilizadas para a criação da base de dados. Esses objetos são evidenciados na figura por meio de retângulos vermelhos de forma a realçar que as áreas cortadas nas capturas de tela não serão anotadas para a realização deste trabalho.

Figura 3.2 – Objetos incompletos que não serão considerados para a anotação das imagens, representados pelos retângulos vermelhos.

Fonte: Software Google Earth® (Google, 2023).

3.1.3 Exportação da Base de Dados

Como dito anteriormente, a plataforma *Roboflow* será utilizada para realizar as anotações das imagens de toda a base de dados deste trabalho. Essa plataforma permite a exportação da base de dados desenvolvida em diversos formatos como .txt, .xml e .csv. Portanto, a etapa de exportação da base também será realizada por meio do *Roboflow*.

Nessa etapa, antes da exportação, o *Roboflow* permite a realização de alguns pré-processamentos como alteração no tamanho de imagem, aplicação de técnicas de aumento de dados também conhecidas como *data augmentation* e a divisão da base de dados em treino, teste e validação.

Após a exportação da base de acordo com os formatos disponibilizados pelo *Roboflow*, os ajustes necessários serão realizados fora dessa plataforma para viabilizar o treinamento do modelo com a base exportada.

3.2 Definição do Modelo de Treinamento

A seguir serão apresentados os detalhes dos modelos de treinamento utilizados para a detecção de áreas com atividades de vulnerabilidade humana. Nesse sentido, serão expostos os fatores e estratégias utilizados na definição do modelo de treinamento.

3.2.1 Estratégias de Treinamento

Neste trabalho, duas categorias de modelos de detecção de objetos serão o foco: YOLOv7 e *Segmentation Models*. Para a realização dos treinamentos serão adotadas duas estratégias diferentes para que seja possível analisar fatores como o desempenho da rede neural e suas métricas. Nesse sentido, serão aplicadas as técnicas de treinamento com inicialização aleatória de pesos, também conhecida como *from scratch*, e de treinamento com carregamento de pesos pré-treinados ou por transferência de aprendizado, chamada de *transfer learning*.

O treinamento *from scratch* possui a premissa de que todo o treinamento é realizado usando os dados específicos do problema em questão, sem a necessidade de se basear em modelos previamente treinados. Isto significa que os pesos das camadas convolucionais da rede neural são inicializados de forma aleatória para evitar a existência de um viés inicial. Nesse sentido, treinar uma rede neural do zero traz a flexibilidade e controle total sobre todo o processo, desde a inicialização dos pesos até otimização dos hiperparâmetros, entretanto costuma requerer uma grande quantidade de dados para obter-se bons resultados, além de exigir muitos recursos computacionais e tempo para finalizar o treinamento.

Já a estratégia de transferência de aprendizado parte do pressuposto de que o treinamento é iniciado com um modelo pré-treinado em um conjunto de dados genérico e, em

seguida, é feito o ajuste do modelo para um novo conjunto de dados específico. Ou seja, os pesos das camadas convolucionais da rede neural são inicializados com os pesos aprendidos no decorrer do treinamento prévio. Dessa maneira, a técnica de *transfer learning* é bastante eficiente quando a base de dados é pequena e demanda menos tempo de treinamento, já que a maior parte de todo o processo de aprendizado já foi realizada anteriormente. Contudo, essa estratégia requer cuidados especiais dado que pode haver uma incompatibilidade entre o modelo pré-treinado e a tarefa alvo a ser realizada. Ademais, o ajuste fino realizado pode aumentar o risco de *overfitting*, conforme explicado em detalhes na subseção 2.1.2.5.

Vale ressaltar que para a realização de ambas as estratégias será utilizado o procedimento de divisão da base de dados em três partes. A primeira parte é o conjunto de imagens de treinamento com 70% das imagens da base de dados, a segunda e a terceira parte são os conjuntos de validação e teste com 15% das imagens cada. Essa divisão será feita de forma igual para a base de dados das imagens de satélite na escala de $50m$ e para base de dados com as imagens na escala de $100m$.

A ideia principal dos treinamentos é justamente comparar os processos e resultados obtidos com ambas as estratégias comentadas, *from scratch* e *transfer learning* e, assim, encontrar a categoria e o modelo com melhor desempenho para a detecção de regiões de vulnerabilidade humana no Distrito Federal.

3.2.2 Definição dos Parâmetros de Treinamento

A etapa de treinamento de qualquer modelo de rede neural é essencial para a garantia de um modelo robusto, uma vez que é nessa etapa que a rede aprende as principais características da base de dados e, portanto, esse processo é decisivo para que o modelo se comporte da maneira mais assertiva possível. São diversos os fatores capazes de afetar a eficácia do treinamento: a qualidade da base de dados, por exemplo, é um desses fatores.

Essa fase é diretamente afetada pelos parâmetros descritos na seção 2.2, sendo que a alteração de um ou mais desses parâmetros pode ser decisiva para um aprendizado satisfatório. Ao início dessa etapa, não é possível saber com convicção qual é a combinação de parâmetros que resultará no melhor desempenho durante o treinamento, por isso, é indispensável a realização do treinamento múltiplas vezes, variando a combinação desses parâmetros entre as tentativas.

Desse modo, durante a fase de treinamento, serão avaliados os desempenhos dos modelos para diferentes valores de épocas, tamanhos de lote e taxa de aprendizagem buscando-se observar para quais valores, o modelo possui o melhor desempenho e, então, definir o melhor conjunto de parâmetros.

3.3 Avaliação e Ajustes do Modelo Escolhido

Em frente aos resultados dos treinamentos, será feita uma avaliação cuidadosa das métricas para que seja possível ajustar e escolher a categoria mais adequada a este trabalho. As métricas que serão coletadas são as seguintes:

- Melhor época
- *F1-score*
- Intersecção sobre a União (*IoU*)

Essas métricas são detalhadas na seção 2.3 e em posse delas será possível avaliar o desempenho de cada modelo em relação à detecção das áreas de interesse. Além disso, também serão avaliados os valores de função de perda para cada treinamento.

Em seguida, por meio das duas estratégias de treinamento, *from scratch* e *transfer learning*, as métricas citadas acima serão comparadas para avaliar como os modelos se comportam em cada uma dessas estratégias. Assim, será possível analisar o desempenho e identificar qual das opções e configurações de treinamento se adapta melhor às necessidades e objetivos do projeto dados na seção 1.3.

Essa escolha será feita por meio da comparação de tabelas com as respectivas métricas. Tendo todos esses dados organizados será viável a comparação e consequente escolha do modelo mais apropriado. Dito isso, será escolhida a estratégia que apresentar o maior valor da métrica *F1-score*, a melhor época e a IoU mais próxima de 1. Evidentemente, há a chance da categoria selecionada não ter todas as métricas superiores ao mesmo tempo, portanto o desempenho será analisado por métrica e por quantidade de métricas com valores superiores.

As análises das métricas são consideradas devido à necessidade das previsões do modelo e estratégia escolhidos se aproximarem o máximo possível dos valores reais e alcançarem a melhor precisão possível para o algoritmo. Os valores abordados no parágrafo anterior são referenciais de cada métrica e representam cenários de ótimo desempenho para o modelo, ou seja, correspondem à meta de alcance do treinamento.

4 Resultados

Este capítulo exibirá os resultados alcançados de acordo com os treinamentos realizados, além da análise detalhada de cada um deles. Inicialmente, os detalhes do *hardware* dos treinamentos e da composição da base de dados serão expostos. Posteriormente, os treinamentos das duas categorias de modelos apresentadas na subseção 3.2.1 serão minuciosamente abordados, passando-se pelos resultados de cada métrica das estratégias de treinamento utilizadas neste trabalho. Por fim, será feita uma discussão em cima da comparação das métricas obtidas por meio da categoria de modelos da YOLOv7 com as métricas alcançadas com a categoria de modelos implementados na biblioteca *Segmentation Models*, salientando o melhor e o pior modelo para cada estratégia empregada.

4.1 Ambiente de Desenvolvimento

Todo o processo de treinamento da base de dados foi realizado em uma Unidade Gráfica de Processamento, do inglês *Graphics Processing Unit* (GPU) NVIDIA-SMI, versão 520.61.05, com um armazenamento máximo de 32768 MiB. Neste *hardware*, utilizou-se a plataforma de computação paralela *Compute Unified Device Architecture* (CUDA) na versão 11.8. Além disso, utilizou-se a linguagem *Python* versão 3.8.10.

4.2 Construção da Base de Dados

Nesta seção, apresenta-se com detalhes como os procedimentos descritos na seção 3.1 foram implementados para obter-se a base de dados de forma consistente e apropriada para os treinamentos desejados[1].

4.2.1 Coleta das Imagens de Satélite

Seguindo os critérios definidos na seção 3.1, foram coletadas 723 imagens na escala de 50 metros e 233 imagens na escala de 100 metros. Essa discrepância entre a quantidade de imagens de cada base de dados se deve ao fato de que o objetivo principal era ter uma quantidade de anotações nivelada entre as bases e não de imagens. Para isso, as bases precisaram ter uma quantidade de imagens um pouco discrepantes considerando que a escala de $100m$ consegue abranger uma maior quantidade de entidades de interesse por imagem que a base de dados de $50m$.

[1] Todos os scripts e base de imagens estão disponíveis no sítio web https://github.com/fbvidal/HumanVulnerabilityDetectionDL.

4.2.2 Anotação das Imagens

Efetuando-se a etapa de anotação das imagens, obteve-se um total de 6202 anotações para a escala de 50 metros, sendo 2087 entidades referentes à classe de área tradicional urbana, 2047 referentes à classe de vulnerabilidade humana e 2068 referentes a resíduos inadequados. Para a escala de 100 metros, obteve-se 6022 anotações, dentre elas, 2008 foram referentes à classe de resíduos inadequados, 2026 referentes à classe de área tradicional urbana e 1988 referentes a vulnerabilidade humana. Vale ressaltar, ainda, que buscou-se balancear a base de dados o máximo possível, desse modo, obtendo-se quantidades similares de entidades para cada classe.

4.2.3 Exportação da Base de Dados

Antes da exportação das bases de dados, selecionou-se a divisão de ambas as bases em 70% para dados de treino, 15% para dados de teste e 15% para dados de validação. Nesta etapa, manteve-se o tamanho original das imagens, sem reajustes no *Roboflow*. Além disso, não foram realizadas operações de *data augmentation* pela plataforma.

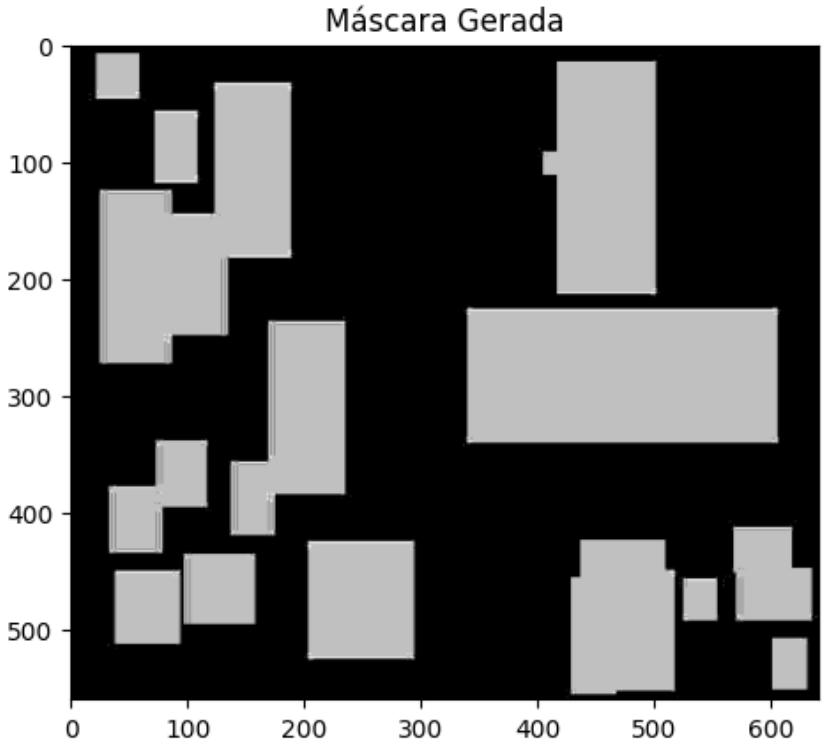
Em seguida, exportou-se as bases de dados nos formatos txt para *YOLOv7 Pytorch* e xml para *Pascal VOC*. Foram feitos dois tipos de exportações devido aos dois tipos de treinamentos de modelos diferentes realizados, conforme detalhado na subseção 3.2.1.

4.2.4 Ajustes e pré-processamento

Para o caso da exportação no formato xml, foi necessário fazer alguns ajustes da base de dados para que ela pudesse ser corretamente utilizada no código de treinamento. Portanto, separou-se a base de dados em arquivos de treinamento, de validação e de teste, os quais foram todos corretamente renomeados.

Em seguida, desenvolveu-se um código em *Python* que converteu cada arquivo xml em uma máscara para cada imagem de forma que a classe de área de vulnerabilidade humana fosse representada visualmente como a escala 1 da cor preta, a classe de resíduos inadequados como a escala 2, a classe de área tradicional urbana como a escala de cor 3 e a escala 0 representou todo o resto da figura que não foi anotado ou também conhecido com o fundo ou *background* da imagem. Dessa forma, foi possível utilizar máscaras para representar as anotações de cada imagem do conjunto de dados em vez de utilizar arquivos no formato xml, como exemplifica a Figura 4.1. Vale ressaltar que a máscara da Figura 4.1 possui uma escala de cor alterada para melhorar a visualização das classes da imagem.

Figura 4.1 – Exemplo de visualização da máscara de uma imagem da base de dados do trabalho.



4.3 Treinamentos dos Modelos

Para a realização dos treinamentos foi utilizada a linguagem de programação *Python*, além de bibliotecas e ferramentas consolidadas na área de aprendizagem profunda.

O *Pytorch* (Paszke *et al.*, 2019) é uma das ferramentas empregadas mais importantes, isso porque esse *framework* de desenvolvimento baseado em tensores traz mais flexibilidade e eficiência aos treinamentos de redes profundas. Assemelhando-se à programação convencional do *Python*, o *Pytorch* se destaca pela sua simplicidade e facilidade de uso e, também, por possuir suporte para execução em unidades de processamento gráfico ou GPUs, o que acelera significativamente os treinamentos dos modelos.

Os treinamentos para todos os modelos de ambas as categorias foram realizados utilizando-se como hiperparâmetro 300 épocas, ou seja, tanto nos treinamentos da YOLOv7 quanto nos treinamentos do *Segmentation Models* o algoritmo percorreu todo o conjunto de dados 300 vezes. A escolha de 300 épocas se deu devido ao fato de ser uma quantidade razoável inicial para que o modelo tenha tempo suficiente de aprender com os dados, mas evitar o treinamento excessivo para não causar o sobre-ajuste.

4.3.1 Treinamentos YOLOv7

Iniciou-se o processo de treinamento a partir dos modelos da YOLOv7. Para cada modelo disponível em Wang, Bochkovskiy e Liao (2022) realizou-se o treinamento *from*

scratch de cada base de dados individualmente. Nas Tabela 4.1 e Tabela 4.2 a seguir, é possível verificar os hiper-parâmetros adotados em cada treinamento para as bases de 50m e 100m, respectivamente.

Vale ressaltar que o tamanho das imagens foi definido a partir da recomendação dos autores. Além disso, devido a limitações de *hardware*, foi necessário reduzir o *batch size* para modelos mais complexos.

Tabela 4.1 – Hiper-parâmetros utilizados para o treinamento da base de 50m com os modelos da YOLOv7.

Modelo	Workers	Batch Size	Tamanho Imagem	Nº Épocas
YOLOv7	8	16	640 x 640	300
YOLOv7x	8	16	640 x 640	300
YOLOv7-w6	8	8	1280 x 1280	300
YOLOv7-d6	8	4	1280 x 1280	300
YOLOv7-e6	8	4	1280 x 1280	300
YOLOv7-e6e	8	4	1280 x 1280	300

Tabela 4.2 – Hiper-parâmetros utilizados para o treinamento da base de 100m com os modelos da YOLOv7.

Modelo	Workers	Batch Size	Tamanho Imagem	Nº Épocas
YOLOv7	8	32	640 x 640	300
YOLOv7x	8	32	640 x 640	300
YOLOv7-w6	8	16	1280 x 1280	300
YOLOv7-d6	8	8	1280 x 1280	300
YOLOv7-e6	8	8	1280 x 1280	300
YOLOv7-e6e	8	4	1280 x 1280	300

Ao final de cada treinamento, foram gerados indicadores de performance como gráficos de F1, curvas de precisão, sensibilidade, precisão x sensibilidade e matriz de confusão. Além disso, ainda são gerados os valores de mAP@0.5 e mAP@0.5:0.95 para cada época.

Na Figura 4.2, pode-se observar um exemplo de imagem da base de teste de 50m com a saída esperada e a previsão efetivamente realizada por um dos modelos da YOLOv7. Já na Figura 4.3 pode-se observar um exemplo também da base de teste para um dos modelos da YOLOv7, contudo para a base de dados de 100m.

Em seguida, para fins de comparação de performance, realizou-se novos treinamentos para cada um dos modelos em ambas as bases utilizando a técnica *transfer learning* explicada anteriormente. Utilizou-se os pesos de treinamentos prévios de cada modelo com a base de dados da COCO, que estão disponíveis no repositório da YOLOv7. Os mesmos hiper-parâmetros da Tabela 4.1 e da Tabela 4.2 foram adotados para esses treinamentos.

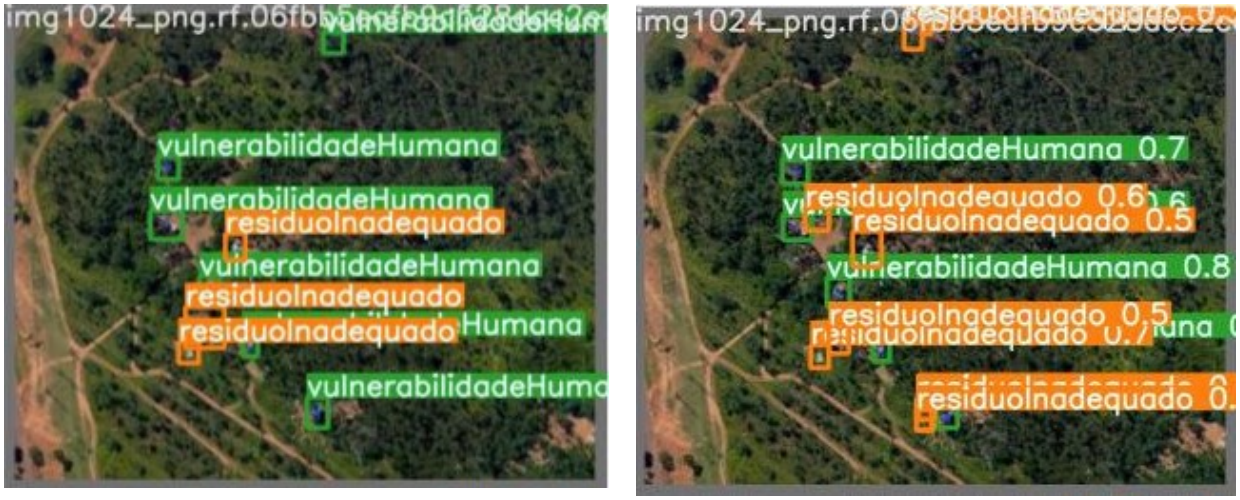
A COCO é uma base de dados robusta para detecção e segmentação de imagens, sendo amplamente utilizada para teste e validação de modelos emergentes. Devido a sua

Figura 4.2 – Exemplo de uma imagem de teste para a base de 50m.

(a) Saída esperada (b) Saída prevista

Figura 4.3 – Exemplo de uma imagem de teste para a base de 100m.

(a) Saída esperada (b) Saída prevista



magnitude e compatibilidade com a YOLOv7, optou-se por utilizar os pesos do treinamento com essa base de dados para a técnica do *transfer learning*, utilizando-se da recomendação dos próprios autores da YOLOv7.

4.3.2 Treinamentos *Segmentation Models*

Em seguida, a fim de comparar as performances entre diferentes estratégias e modelos, estudou-se a biblioteca *Segmentation Models* (Iakubovskii, 2019) e, a partir dos exemplos fornecidos, foi elaborado um código de treinamento que iterou sobre cada uma das com-

binações de arquitetura e *encoder* disponíveis. Cada combinação foi formada por uma das arquiteturas e um dos *encoders* descritos na subseção 2.1.4.1.1.

De forma análoga ao que foi feito com a YOLOv7, para as diferentes combinações da *Segmentation Models*, realizou-se inicialmente um treinamento *from scratch* para cada uma das bases e, depois, um treinamento do tipo *transfer learning*. Mais uma vez, optou-se por seguir a recomendação dos autores e utilizou-se a base de dados *ImageNet* para o processo de *transfer learning* com os modelos disponíveis na biblioteca *Segmentation Models*. A *ImageNet* também se trata de uma base de dados bastante robusta e desenvolvida, sendo, portanto, apropriada para essa técnica.

Mais uma vez, de forma similar à estratégia anterior, ao final de cada treinamento, foram gerados indicadores de performance. Para esses modelos, foram obtidos valores para a função de perda, F1 e *IoU* para cada época.

Devido a limitações de *hardware*, não foi possível realizar o treinamento utilizando todas as combinações para cada base de dados. Nesse sentido, foi possível realizar treinamentos com a base de 50*m* para as combinações da Tabela 4.3, enquanto para a base de 100*m* foi possível realizar treinamentos para as combinações da Tabela 4.4.

Tabela 4.3 – Combinações arquitetura + *encoder* para os treinamentos da base de 50m com a biblioteca *Segmentation Models*

Combinação	Arquitetura	Encoder
PSPNet e resnet50	PSPNet	resnet50
PSPNet e resnet101	PSPNet	resnet101
PSPNet e efficientnet-b3	PSPNet	efficientnet-b3
PSPNet e timm-gernet_m	PSPNet	timm-gernet_m

Tabela 4.4 – Combinações arquitetura + *encoder* para os treinamentos da base de 100m com a biblioteca *Segmentation Models*

Combinação	Arquitetura	Encoder
PSPNet e resnet50	PSPNet	resnet50
PSPNet e resnet101	PSPNet	resnet101
PSPNet e efficientnet-b3	PSPNet	efficientnet-b3
PSPNet e efficientnet-b7	PSPNet	efficientnet-b7
PSPNet e timm-gernet_m	PSPNet	timm-gernet_m
DeepLabv3 e resnet50	DeepLabv3	resnet50
DeepLabv3 e resnet101	DeepLabv3	resnet101
DeepLabv3 e timm-gernet_m	DeepLabv3	timm-gernet_m

Observa-se que a base de 100*m* conseguiu rodar mais modelos do que a base de 50*m*. Considerando que o mesmo código e o mesmo procedimento foi utilizado em ambos os treinamentos das duas bases, essa diferença se deve ao fato de que quando os treinamentos de 100*m* foram rodados o servidor utilizado possuía mais espaço de memória disponível

do que quando a base de 50*m* foi treinada. Nesse sentido, buscando-se manter consistência entre as análises das duas bases de dados, serão analisadas a seguir apenas as combinações de arquitetura e *encoder* que satisfizeram ambas as bases.

4.4 Análise dos Resultados Obtidos

Conforme explicitado na seção anterior, foi possível extrair varias métricas de performance para cada um dos treinamentos realizados. Cada um desses indicadores foram cuidadosamente analisados e os mais relevantes foram selecionados e serão analisados a seguir.

4.4.1 Resultados YOLOv7

Em primeiro lugar, os resultados da YOLO serão apresentados.

4.4.1.1 Resultados YOLOv7 para a Base de Dados de 50m

Inicialmente, serão analisados os resultados para a base de 50*m*. Na Tabela 4.5 e na Tabela 4.6, podemos observar os resultados *from scracth* e *transfer learning*, respectivamente. Para fins de visualização dos resultados, optou-se por apresentar a melhor época de cada modelo considerando-se a métrica de mAP@0.5.

Tabela 4.5 – Resultados obtidos para o treinamento *from scratch* com a YOLOv7 para a base de dados de 50*m*.

Modelo	Época	Precisão	Sensibilidade	mAP@0.5	mAP@0.5:0.95	F1
YOLOv7	256	0,4946	0,4548	0,4383	0,1898	0,4739
YOLOv7x	239	0,4485	0,4965	0,4194	0,1802	0,4713
YOLOV7-w6	170	0,4959	0,4590	0,4326	0,1912	0,4767
YOLOv7-e6	257	0,5040	0,4435	0,4351	0,1965	0,4718
YOLOV7-e6e	177	0,4279	0,4836	0,4160	0,1771	0,4540
YOLOV7-d6	236	0,4349	0,4507	0,3995	0,1783	0,4427

Tabela 4.6 – Resultados obtidos para o treinamento *transfer learning* com a YOLOv7 para a base de dados de 50*m*.

Modelo	Época	Precisão	Sensibilidade	mAP@0.5	mAP@0.5:0.95	F1
YOLOv7	58	0,4968	0,5466	0,4778	0,2209	0,5205
YOLOv7x	257	0,4891	0,5507	0,4751	0,2242	0,5181
YOLOV7-w6	72	0,5189	0,5274	0,4664	0,2181	0,5231
YOLOv7-e6	139	0,5473	0,5042	0,4691	0,2145	0,5249
YOLOV7-e6e	24	0,4940	0,5141	0,4699	0,2182	0,5038
YOLOV7-d6	98	0,5279	0,4958	0,4667	0,2191	0,5113

Para o treinamento *from scratch*, observando-se, inicialmente, os valores de precisão, nota-se que os valores variaram entre 0,5 e 0,42, sendo que o modelo YOLOv7-e6 apresentou a melhor performance para esse indicador. Em relação à sensibilidade, nota-se um comportamento semelhante, em que os valores ficaram entre 0,44 e 0,50, aproximadamente, com destaque para o modelo da YOLOv7x.

Analisando-se, agora, os valores de mAP@0.5 e mAP@0.5:0.95 da Tabela 4.5, pode-se notar que os modelos desempenharam de forma similar, com um valor de mAP@0.5 entre 0,40 e 0,44, indicando que, para uma tolerância de IoU de 0,5, o modelo obteve uma taxa de acerto média de aproximadamente 42% para a área de detecção. Ao avaliar-se a mAP@0.5:0.95, no entanto, a taxa de acerto diminui, mostrando que ao se reduzir gradualmente a tolerância de IoU até 95% de compatibilidade, o modelo perde em performance. Esse decréscimo, no entanto, já era esperado e será observado no treinamento de *transfer learning* também, pois a redução da tolerância exige que o modelo seja mais assertivo na delimitação da região de interesse.

Por fim, observando-se a métrica F1, nota-se que o modelo YOLOV7-w6 obteve o melhor valor. Essa métrica mostra que esse modelo performou de forma consistente tanto para a precisão, quanto para a sensibilidade. Nota-se que seus resultados não foram os mais altos para nenhuma das duas métricas de precisão e sensibilidade, no entanto, seu destaque para a F1 demonstra que em uma análise geral considerando ambas as métricas anteriores, esse modelo se destacou.

Uma consistência semelhante foi observada no treinamento com *transfer learning* na Tabela 4.6. No entanto, para esse treinamento, nota-se uma performance mais satisfatória em todas as métricas. Obteve-se valores de mAP@0.5 em torno de 0,47 e mAP@0.5:0.95 acima de 0,2, demonstrando que houve melhorias tanto para o limite de IoU médio, quanto para as maiores restrições de acurácia. Esse comportamento era esperado pois com a técnica de *transfer learning* os pesos de um treinamento prévio são reaproveitados e reutiliza-se de um aprendizado anterior, geralmente, trazendo melhores resultados.

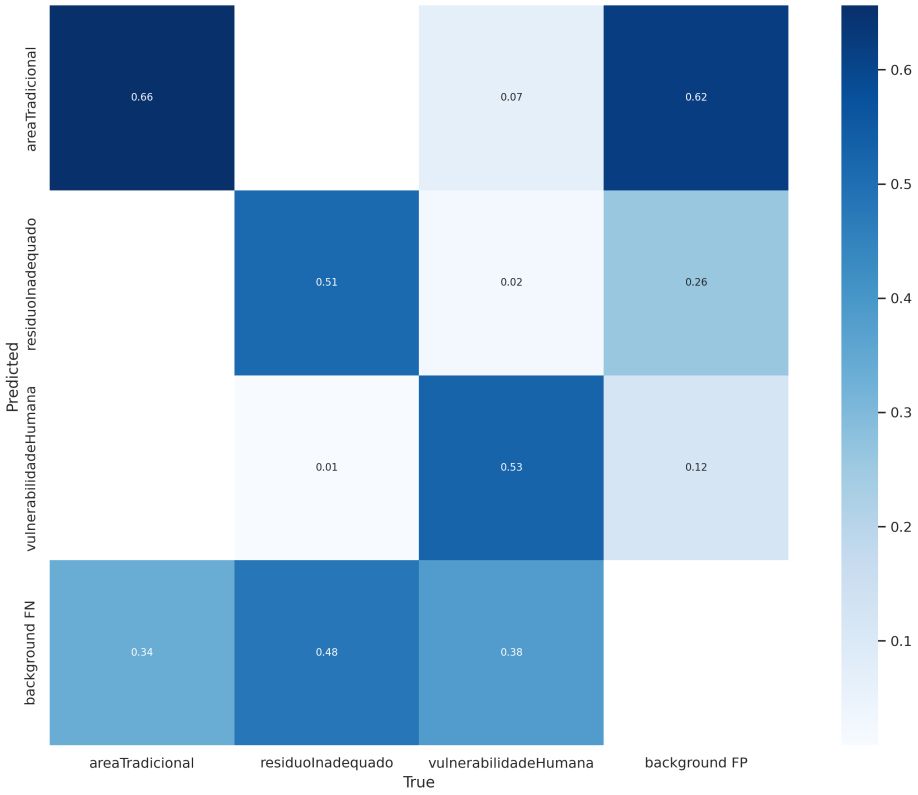
Para ambas as técnicas, nota-se que há uma variação de performance não só entre os modelos mas entre as métricas para um modelo quando comparado com outros. Faz-se importante ressaltar isso, pois apesar de serem analisados diferentes indicadores, será selecionado um deles para fins de comparação entre os modelos da YOLOv7 e do *Segmentation Models*.

Além das tabelas acima, foram obtidas matrizes de confusão nas saídas dos treinamentos da YOLOv7. Abaixo, na Figura 4.4 e na Figura 4.5 podem ser vistas as matrizes de confusão do modelo YOLOv7x para o treinamento *from scratch* e *transfer learning*, respectivamente. Conforme explorado na seção 2.3, a matriz de confusão permite a visualização dos resultados de uma forma gráfica e de fácil compreensão.

Analisando as imagens abaixo, nota-se uma melhoria nas previsões positivas para

as três classes, comparando-se os valores obtidos para o treinamento *transfer learning* em relação ao *from scratch*. Ademais, vale notar os valores obtidos para as previsões falso positivas (FP) e falso negativas (FN) do fundo de imagem. A tendência observada comparando-se o treinamento *transfer learning* em relação ao *from scratch* foi a de redução das previsões FP e FN do fundo da imagem. A diminuição do indicador FP para o fundo contribui para a melhor performance no indicador de precisão do modelo, enquanto a diminuição de FN para o fundo contribui para a melhoria da sensibilidade. Esses fatores, alinhados com a melhoria das previsões positivas, condizem com o aperfeiçoamento do modelo já verificado anteriormente na Tabela 4.5 e na Tabela 4.6.

Figura 4.4 – Matriz de confusão do treinamento *from scratch* da YOLOv7x da base de dados de 50*m*.



4.4.1.2 Resultados YOLOv7 para a Base de Dados de 100m

Agora, serão analisados os resultados para a base de 100*m*. Pode-se observar na Tabela 4.7 e na Tabela 4.8 os resultados para o treinamento *from scratch* e *transfer learning*, respectivamente. Assim como no caso da base de 50*m*, optou-se por ordenar a tabela dos resultados de acordo com a métrica de mAP@0.5.

Primeiro, analisando-se os valores de precisão da base de 100*m*, tem-se que os valores variaram entre 0,46 e 0,61 para o treinamento *from scratch* e entre 0,44 e 0,56 para o *transfer learning*, valores ligeiramente maiores quando comparados aos de precisão da base de 50*m*

Figura 4.5 – Matriz de confusão do treinamento *transfer learning* da YOLOv7x da base de dados de 50m.

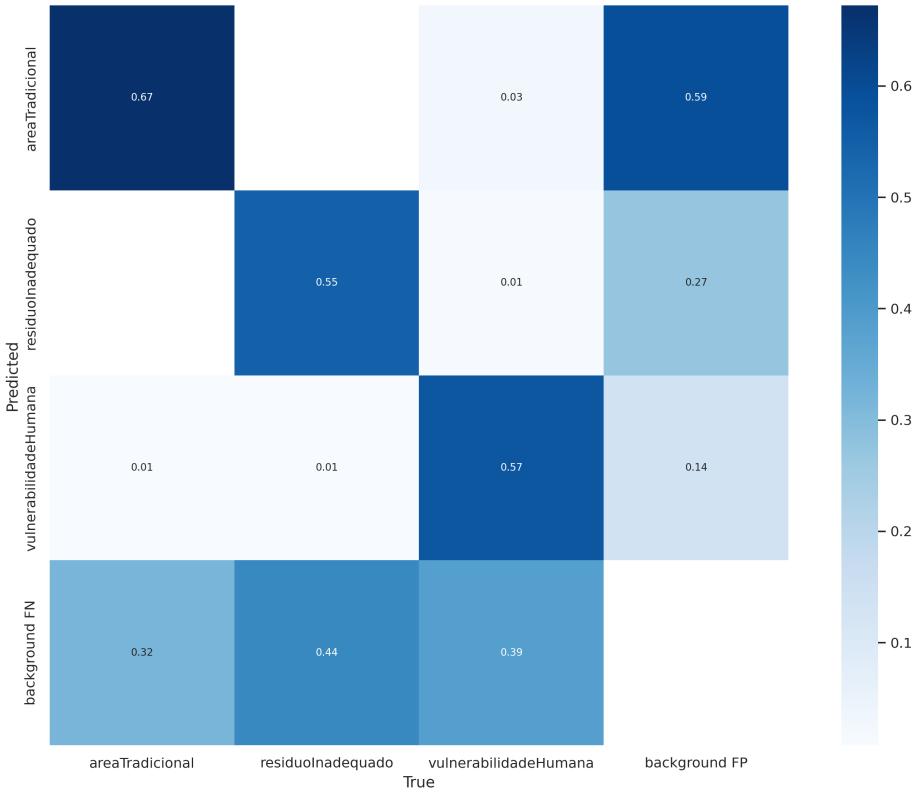


Tabela 4.7 – Resultados obtidos para o treinamento *from scratch* com a YOLOv7 para a base de dados de 100m.

Modelo	Época	Precisão	Sensibilidade	mAP@0.5	mAP@0.5:0.95	F1
YOLOv7-w6	252	0,6023	0,5613	0,5170	0,2585	0,5811
YOLOV7-e6e	293	0,5595	0,5661	0,5059	0,2638	0,5628
YOLOV7	260	0,4761	0,6084	0,4989	0,2388	0,5342
YOLOv7x	290	0,4817	0,5969	0,4984	0,2501	0,5331
YOLOv7-e6	299	0,4673	0,6289	0,4981	0,2675	0,5362
YOLOV7-d6	296	0,5567	0,5406	0,4787	0,2343	0,5485

para esses dois treinamentos. O modelo que apresentou melhor performance para esse indicador foi o YOLOv7-w6 tanto para o treinamento *from scratch* quanto para o *transfer learning*, ainda que no caso do *transfer learning* essa métrica tenha sido igual para os modelos YOLOv7x e YOLOv7-w6, considerou-se o modelo YOLOv7-w6 por apresentar uma menor melhor época. Já em relação à sensibilidade, os valores ficaram entre aproximadamente 0,54 e 0,63 para o *from scratch* e entre 0,47 e 0,63 para o *transfer learning*, destaca-se o modelo YOLOv7-e6 para o primeiro caso, enquanto para o segundo caso destaca-se o modelo YOLOv7x.

Em seguida, analisando-se os valores de mAP@0.5 da Tabela 4.7, assim como no caso da base de 50m, houve pouca variação nos números, mostrando que os modelos treinados

Tabela 4.8 – Resultados obtidos para o treinamento *transfer learning* com a YOLOv7 para a base de dados de 100*m*.

Modelo	Época	Precisão	Sensibilidade	mAP@0.5	mAP@0.5:0.95	F1
YOLOv7x	261	0,5538	0,6266	0,5136	0,2915	0,5880
YOLOV7-w6	197	0,5538	0,5696	0,4962	0,2683	0,5616
YOLOv7	117	0,5203	0,6051	0,4946	0,2751	0,5595
YOLOv7-e6	293	0,5211	0,5230	0,4687	0,2422	0,5220
YOLOV7-e6e	194	0,5120	0,4922	0,4439	0,2308	0,5019
YOLOV7-d6	242	0,4418	0,4783	0,3933	0,1850	0,4593

da base de 100*m* tiveram um desempenho parecido entre si. Essa métrica variou entre 0,47 e 0,52, evidenciando que para a tolerância de IoU de 0,5, a base de 100*m* apresentou uma taxa de acerto média de aproximadamente 50% para a área de detecção, demostrando uma melhora de 8% em relação à taxa de acerto média da base de 50*m* para esse mesmo treinamento. Já para o treinamento por *transfer learning* exibido na Tabela 4.8, essa métrica alcançou resultados similares, porém um pouco inferiores aos do treinamento *from scratch*, apresentando uma taxa de acerto média de aproximadamente 47% para a área de detecção, 3% a menos que a taxa média do treinamento *from scratch* para a base de 100*m*.

No que se refere à métrica mAP@0.5:0.95 da Tabela 4.7, a taxa de acerto média reduziu de forma significativa. Esse comportamento para a mAP@0.5:0.95 também ocorreu para o treinamento *transfer learning* da Tabela 4.8 e era esperado considerando que é uma métrica calculada com valores de IoU cada vez maiores, representando diferentes níveis de sobreposição entre as regiões de interesse previstas e verdadeiras, como explicado anteriormente para a base de 50*m*.

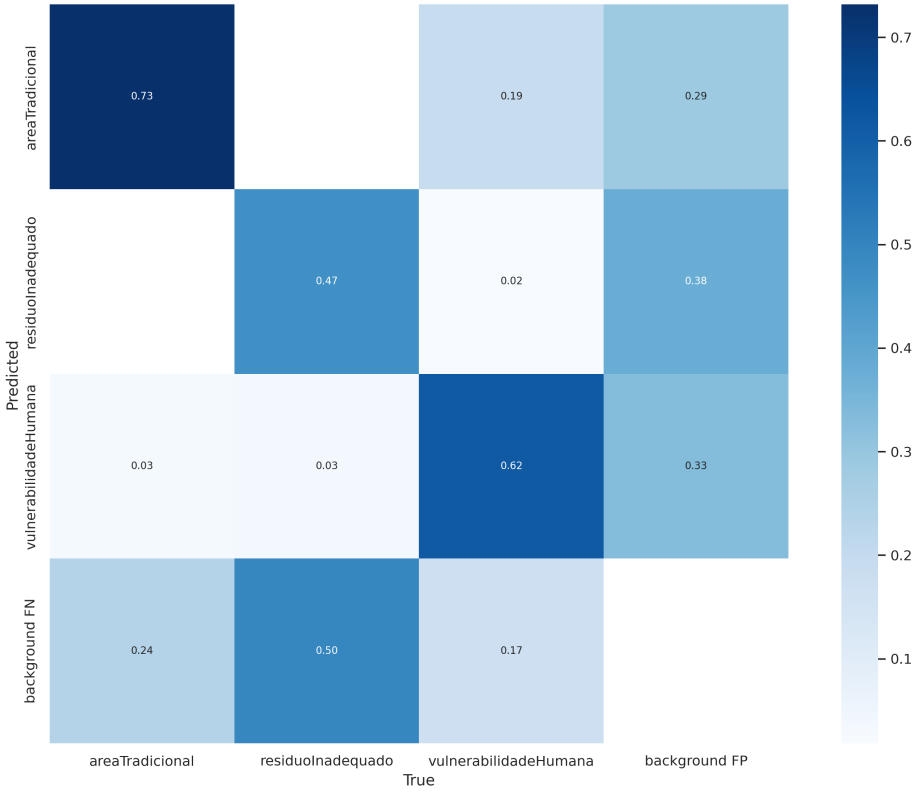
Tomando-se agora os valores da métrica F1 apresentados na Tabela 4.7, observa-se que, apesar de serem próximos entre si, o modelo YOLOv7-w6 se destaca, sendo também o modelo que obteve o maior valor de F1 para a base de 50*m*. Isso pode indicar que o treinamento *from scratch* com o modelo YOLOv7-w6 apresenta certa robustez, já que mesmo com variações de resolução entre as bases de dados foi possível manter um desempenho consistente para a F1. Em relação ao treinamento por *transfer learning*, como observado na Tabela 4.8, o modelo de destaque é o YOLOv7x, que apresenta um valor de F1 ligeiramente superior ao melhor valor de F1 obtido com o treinamento *from scratch*.

Assim como feito com a base de dados de 50m, foram coletadas matrizes de confusão nas saídas dos treinamentos da YOLOv7 da base de dados de 100*m*. Nesse sentido, a título de exemplificação, pode-se observar na Figura 4.6 e na Figura 4.7 as matrizes de confusão do modelo YOLOv7-e6e para o treinamento *from scratch* e para o *transfer learning*, respectivamente.

De imediato, por meio da análise da diagonal principal de ambas as matrizes, percebe-se que o treinamento *from scratch* apresentou valores superiores nas previsões verdadeiras

positivas das três classes analisadas quando compara-se com a diagonal principal da matriz de confusão do treinamento por *transfer learning*. Além disso, analisando-se também os valores da última linha e da última coluna, que representam as previsões de falsos negativos (FN) e as previsões de falsos positivos (FP) do fundo de imagem respectivamente, percebe-se que do treinamento *from scratch* para o treinamento *transfer learning* houve uma tendência de aumento nos valores das previsões de FN e FP. Nesse sentido, considerando que o ideal é que os valores de FN e FP sejam menores, alinhados com a necessidade de valores maiores na diagonal principal da matriz de confusão, conclui-se que o treinamento *from scratch* demonstrou um desempenho superior ao treinamento por *transfer learning*, conforme já verificado anteriormente na Tabela 4.7 e na Tabela 4.8.

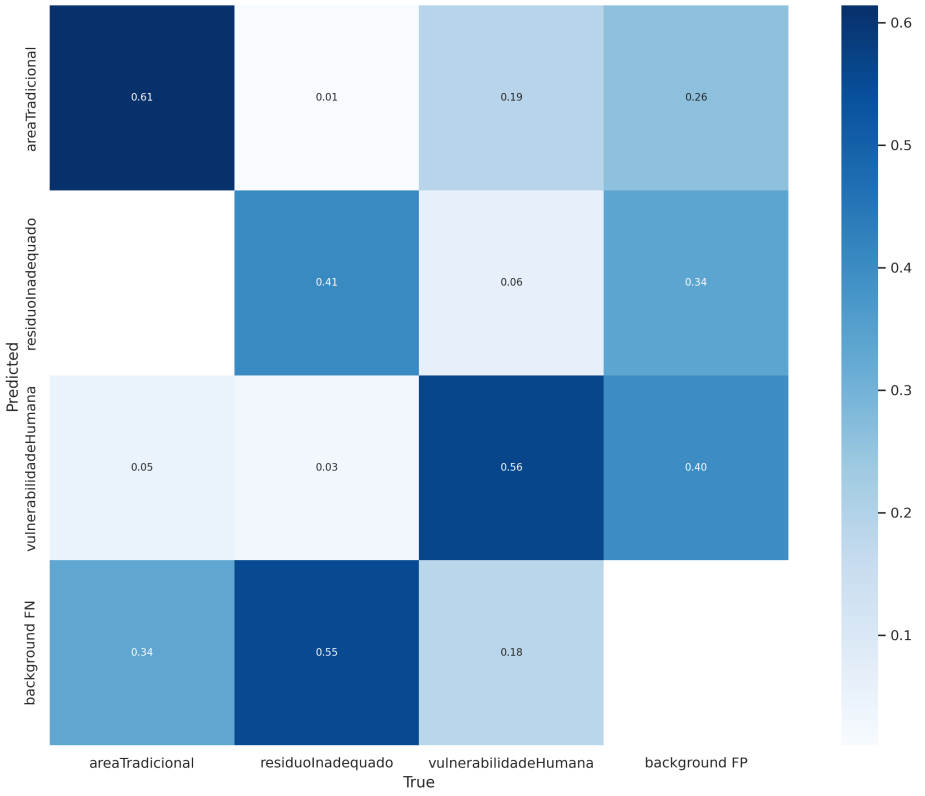
Figura 4.6 – Matriz de confusão do treinamento *from scratch* da YOLOv7-e6e da base de dados de 100m.



4.4.1.3 Resultados YOLOv7 para as Bases de 50m e de 100m

Em suma, percebe-se que no caso da base de 50m, a estratégia do *transfer learning* apresentou melhores resultados para todas as métricas, conforme evidenciado na Tabela 4.9. Para a base de 100m, contudo, a estratégia *from scratch* revelou-se superior em todas as métricas, como pode-se observar pela Tabela 4.10, que compila os valores médios de cada métrica para essa base, considerando cada estratégia empregada. Tanto para a Tabela 4.9 quanto para a Tabela 4.10 tem-se que P_m representa o valor médio de precisão, S_m a sensibi-

Figura 4.7 – Matriz de confusão do treinamento *transfer learning* da YOLOv7-e6e da base de dados de 100m.



lidade média, mAP@0.5$_m$ a mAP@0.5 média, mAP@0.5:0.95$_m$ a mAP@0.5:0.95 média e F1 a F1 média.

Tabela 4.9 – Valores médios de cada métrica do treinamento YOLOv7 por estratégia para a base de dados de 50m.

Estratégia	P_m	S_m	mAP@0.5$_m$	mAP@0.5:0.95$_m$	$F1_m$
From Scratch	0,4676	0,4647	0,4235	0,1855	0,4651
Transfer Learning	0,5123	0,5231	0,4708	0,2192	0,5170

Tabela 4.10 – Valores médios de cada métrica do treinamento YOLOv7 por estratégia para a base de dados de 100m.

Estratégia	P_m	S_m	mAP@0.5$_m$	mAP@0.5:0.95$_m$	$F1_m$
From Scratch	0,5239	0,5837	0,4995	0,2522	0,5493
Transfer Learning	0,5171	0,5491	0,4684	0,2488	0,5321

4.4.2 Resultados *Segmentation Models*

Agora, serão apresentados os resultados para os treinamentos com a biblioteca *Segmentation Models*.

4.4.2.1 Resultados *Segmentation Models* para a Base de Dados de 50m

Em relação aos modelos do *Segmentation Models*, pode-se observar na Tabela 4.11 e na Tabela 4.12 os treinamentos *from scratch* e *transfer learning* para a base de 50m. Novamente, para fins de visualização, foi escolhida a melhor época de cada modelo associada ao melhor resultado da métrica F1.

Tabela 4.11 – Resultados obtidos para o treinamento *from scratch* com o *Segmentation Models* para a base de dados de 50m.

Modelo e arq	Época	F1	IoU	Função de Perda
PSPNet e resnet50	112	0,6400	0,4790	0,2202
PSPNet e resnet101	67	0,6399	0,4768	0,2203
PSPNet e efficientnet-b3	60	0,6313	0,4749	0,2270
PSPNet e timm-gernet_m	74	0,6361	0,4761	0,2235

Tabela 4.12 – Resultados obtidos para o treinamento *transfer learning* com o *Segmentation Models* para a base de dados de 50m.

Modelo e arq	Época	F1	IoU	Função de Perda
PSPNet e resnet50	26	0,6391	0,4774	0,2215
PSPNet e resnet101	19	0,6422	0,4776	0,2193
PSPNet e efficientnet-b3	63	0,6448	0,4761	0,2167
PSPNet e timm-gernet_m	21	0,6350	0,4725	0,2247

Para o resultado *from scratch*, nota-se uma forte consistência entre os resultados das diferentes combinações de *encoders* com o modelo PSPNet, sendo que o resultado geral da função de perda ficou em torno de 0,22. A F1 apresentou um resultado médio de 0,64 e o valor aproximado da *IoU* para as 4 combinações foi de 0,48. Nota-se apenas ligeiras diferenças entre cada um dos resultados para cada *encoder*, podendo-se interpretá-las como ruídos.

A mesma interpretação acima pode ser feita para os resultados obtidos para o treinamento com a técnica de *transfer learning*. Ambas as técnicas trouxeram resultados muito semelhantes e, apesar de os resultados do *transfer learning* apresentarem melhorias mínimas na segunda casa decimal, não pode-se afirmar que houve um aumento de performance considerável. Nesse sentido, aproveitar-se dos pesos e treinamento da base *ImageNet* não apresentou resultados mais favoráveis, podendo indicar que a base escolhida não foi apropriada para o problema analisado neste projeto.

4.4.2.2 Resultados *Segmentation Models* para a Base de Dados de 100m

Já para a base de 100m, pode-se observar os resultados para os treinamentos *from scratch* e *transfer learning* na Tabela 4.13 e na Tabela 4.14, respectivamente. Analogamente

ao que foi feito com a base de 50*m*, aqui também considerou-se a escolha da melhor época como sendo aquela associada ao melhor resultado da métrica F1.

Tabela 4.13 – Resultados obtidos para o treinamento *from scratch* com o *Segmentation Models* para a base de dados de 100*m*.

Modelo e arq	Época	F1	IoU	Função de Perda
PSPNet e resnet50	274	0,5239	0,6490	0,2130
PSPNet e resnet101	287	0,5238	0,6561	0,2079
PSPNet e efficientnet-b3	33	0,5240	0,6536	0,2116
PSPNet e efficientnet-b7	39	0,5238	0,6624	0,2039
PSPNet e timm-gernet_m	231	0,5241	0,6572	0,2073
DeepLabv3 e resnet50	13	0,5243	0,6662	0,2031
DeepLabv3 e resnet101	160	0,5238	0,6465	0,2155
DeepLabv3 e timm-gernet_m	35	0,5239	0,6651	0,2026

Tabela 4.14 – Resultados obtidos para o treinamento *transfer learning* com o *Segmentation Models* para a base de dados de 100*m*.

Modelo e arq	Época	F1	IoU	Função de Perda
PSPNet e resnet50	0	0,5246	0,6275	0,2501
PSPNet e resnet101	290	0,5241	0,6599	0,2059
PSPNet e efficientnet-b3	258	0,5241	0,6605	0,2049
PSPNet e efficientnet-b7	53	0,5241	0,6733	0,1961
PSPNet e timm-gernet_m	261	0,5243	0,6493	0,2133
DeepLabv3 e resnet50	245	0,5236	0,6838	0,1876
DeepLabv3 e resnet101	79	0,5239	0,6821	0,1895
DeepLabv3 e timm-gernet_m	289	0,5239	0,6790	0,1913

De imediato, nota-se que, para a base de 100*m*, tanto o treinamento *from scratch* quanto o treinamento por *transfer learning* possibilitaram a execução de uma maior variedade de combinações de modelos e arquiteturas em comparação com a base de 50*m*. Para ambas as estratégias de treinamento, assim como ocorreu para os treinamentos da base de 50*m*, observa-se uma constância significativa entre os resultados. Nesse sentido, mesmo variando-se o modelo e a arquitetura analisadas, os valores finais de cada métrica ficaram bastante próximos entre si.

Em relação ao treinamento *from scratch* da Tabela 4.13, tem-se um resultado médio aproximado de 0,52 para a métrica F1, 0,66 para a métrica IoU e 0,21 para a função de perda. Já em relação ao treinamento utilizando-se a estratégia de *transfer learning* da Tabela 4.14, percebe-se que os valores médios aproximados de cada métrica são praticamente os mesmos que os encontrados com o treinamento *from scratch*, sendo a métrica da função de perda a única com um valor médio diferente, mas mesmo assim ainda muito próximo do valor da função de perda do treinamento *from scratch*. Portanto, no *transfer learning* os valores

médios de cada métrica foram de 0,52 para a F1, 0,66 para a IoU e 0,20 para a função de perda.

Em suma, de maneira semelhante ao ocorrido com a base de 50m, os resultados do treinamento *transfer learning* da base de 100m apresentaram melhorias extremamente pequenas em alguns casos de combinações de modelo e arquitetura e, portanto, podem ser considerados apenas ruídos.

4.4.2.3 Resultados *Segmentation Models* para as Bases de 50m e de 100m

Comparando-se ambas as bases, percebe-se que a base de 50m performou melhor em relação à métrica F1, contudo a base de 100m obteve melhores resultados para a métrica IoU. Já a função de perda apresentou resultados bastante semelhantes para ambas as bases, com valores médios finais muito próximos, conforme observado na Tabela 4.15 e na Tabela 4.16. Nessas tabelas, $F1_m$ representa a F1 média, IoU_m representa a IoU média e Função de Perda$_m$ representa a Função de Perda média.

Tabela 4.15 – Valores médios de cada métrica do treinamento *Segmentation Models* por estratégia para a base de dados de 50m.

Estratégia	$F1_m$	IoU_m	Função de Perda$_m$
From Scratch	0,6368	0,4767	0,2228
Transfer Learning	0,6403	0,4759	0,2206

Tabela 4.16 – Valores médios de cada métrica do treinamento *Segmentation Models* por estratégia para a base de dados de 100m.

Estratégia	$F1_m$	IoU_m	Função de Perda$_m$
From Scratch	0,5240	0,6570	0,2081
Transfer Learning	0,5241	0,6644	0,2048

4.4.3 Comparação das Estratégias Adotadas

A partir das análises anteriores, nota-se que os indicadores de performance para cada estratégia são diferentes. Nesse sentido, para essa seção, será considerada apenas a métrica F1 *Score*, uma vez que é o índice em comum entre ambas as estratégias.

Avaliando-se, inicialmente, os resultados da base de 50m, nota-se que, tanto para o treinamento *from scratch*, quanto para o *transfer learning*, os valores da F1 para os modelos do *Segmentation Models* foram maiores do que os valores para os modelos da YOLOv7, demonstrando que o modelo se adaptou melhor em termos de precisão e sensibilidade de forma conjunta. Enquanto os valores dos modelos de *Segmentation Models* variaram em torno de 0,64 para ambas as técnicas de treinamento, os valores de F1 para a YOLOv7 ficaram em torno de 0,52 na sua melhor técnica. Pode-se observar, portanto, que no contexto

avaliado, uma abordagem usando arquitetura *encoder-decoder* foi mais vantajosa em relação à abordagem com os modelos da YOLOv7.

Em relação aos resultados da base de 100*m*, contudo, percebe-se que para ambas as estratégias de treinamento, os valores médios da F1 utilizando-se os modelos da YOLOv7 foram superiores que o valor médio da F1 com a abordagem *encoder-decoder* do *Segmentation Models*. Assim, tanto o treinamento *from scratch* da YOLOv7, que obteve uma F1 média de 0,5493, quanto o *transfer learning* da YOLOv7, que apresentou F1 média de 0,5321, foram superiores ao valor médio da F1 de 0,52 para ambas as estratégias com a abordagem do *Segmentation Models*. Assim sendo, pode-se dizer que, para essa base dados específica de 100m, a YOLOv7 pode ser uma escolha mais adequada para tarefas que requerem um equilíbrio melhor entre precisão e sensibilidade.

4.5 Discussões Finais

Com base nas análises anteriores e fazendo-se as considerações necessárias para cada modelo, os modelos da biblioteca *Segmentation Models* demonstraram melhor perfomance para a base de 50*m* enquanto a YOLOv7 teve destaque para a base de 100*m*. Nota-se que os melhores resultados da base de 50*m* para a YOLOv7 foram obtidos com a estratégia *transfer learning*, já os melhores resultados da base de 100*m* foram obtidos com a estratégia *from scratch*. Nesse sentido, comparando-se os valores médios de cada base para o treinamento da YOLOv7 dados pela Tabela 4.9 e pela Tabela 4.10, percebe-se que a base de 100*m* apresentou melhores resultados para este modelo em ambas as estratégias utilizadas.

Dessa forma, apesar de a base de dados de 50*m* possuir imagens com maior riqueza de detalhes devido a sua melhor resolução, o fato da base de 100*m* ter performado melhor neste caso, pode significar que os modelos da YOLOv7 se beneficiam de imagens menos detalhadas devido à redução da complexidade de processamento. Logo, imagens menos detalhadas podem estar fornecendo uma visão mais geral e limpa dos objetos, o que proporciona um melhor equilíbrio entre informação relevante e ruído. Assim, no caso da YOLOv7 a base de dados de 100*m* tem maior capacidade de generalização para novos dados que a base de dados de 50*m*.

Já considerando-se os modelos da biblioteca *Segmentation Models*, percebe-se que a aplicação da técnica de *transfer learning* não trouxe grandes acréscimos no desempenho dos modelos. Contudo, os resultados médios da métrica F1 foram superiores para a base de dados de 50*m*, ao passo que os resultados médios da métrica IoU foram melhores para a base de dados de 100*m*. Isso pode significar que, por ser mais detalhada, a base de 50*m* permitiu que o modelo da *Segmentation Models* capturasse melhor as características das regiões de interesse, resultando em um maior equilíbrio entre precisão e sensibilidade. Entretanto, é possível que tenha havido uma facilitação na identificação de áreas gerais dos objetos

ao utilizar a base de 100*m*, visto que a IoU dessa base apresentou resultados superiores, evidenciando uma melhor sobreposição entre as regiões previstas e as regiões reais dos objetos. Já o fato de ambas as bases terem performado de maneira parecida para a métrica de função de perda pode implicar que os modelos da *Segmentation Models* conseguiram se ajustar de forma similar aos dados mesmo havendo diferenças nos níveis de detalhe e nas escalas das imagens, apresentando desempenhos quantitativos bastante próximos entre as bases.

As ligeiras diferenças de performance entre os modelos para as duas bases pode estar diretamente associado a fatores de resolução da imagem, tamanho da base, anotação das imagens e *status* do *hardware* no momento de cada treinamento. Vale notar, por exemplo, que devido a maior disponibilidade do *hardware* no momento de treinamento da base de 100*m*, foi possível utilizar *batch sizes* maiores para esse conjunto de dados, o que contribui para a performance do modelo.

Ainda, observa-se que, apesar das considerações explicadas na subseção 3.1.2 e exemplificadas na Figura 3.2, durante a predição com a base de testes, o modelo da YOLOv7 classificou entidades consideradas incompletas para a predição, conforme exemplificado na Figura 4.2, o que contribui para a menor performance do modelo.

5 Conclusões

É notório que a região do Distrito Federal apresenta diversas regiões de vulnerabilidade humana. O indivíduo em situação de vulnerabilidade humana é submetido a uma condição constante de desamparo e possui seus direitos básicos de moradia e segurança limitados, estando muitas vezes em condições sanitárias precárias e ao redor de resíduos descartados inadequadamente. É importante a identificação dessas regiões para que os órgãos responsáveis possam reconhecer a dimensão da situação e buscar a melhor forma de suporte e eventual reversão dessa situação, garantindo condições dignas aos indivíduos.

Nesse sentido, este trabalho explorou o uso de técnicas de aprendizagem profunda em imagens de satélite de regiões de vulnerabilidade humana, depósito inadequado de resíduos e áreas consideradas tradicionais, isto é, sem indícios de vulnerabilidade humana, com o intuito de identificar as características de regiões de interesse. A partir dessas imagens, foi possível demarcar e classificar as regiões de interesse em cada imagem e, dessa forma, construir duas bases de dados sólidas, com imagens de 50m e 100m de resolução, para treinar modelos de aprendizagem profunda.

Com o objetivo de testar diferentes tecnologias, optou-se por estudar e treinar as bases com modelos da YOLOv7, fazendo o uso do *backbone Darknet*, e, também, os modelos da biblioteca *Segmentation Models*, seguindo uma arquitetura *encoder-decoder*. Os parâmetros de treinamento foram mantidos os mais próximos possível, considerando as limitações de *hardware*, a fim de garantir a integridade dos resultados. Além disso, buscou-se avaliar a relevância dos treinamentos *from scratch* e *transfer learning*, buscando-se comparar as performances de cada tipo de treinamento com cada base de dados.

Após o treinamento dos modelos e predição com os dados de teste, foi possível comparar os resultados para cada arquitetura e avaliar os respectivos desempenhos para cada base. Essa etapa foi fundamental para melhor compreender o comportamento de cada modelo e realizar uma avaliação exploratória dos resultados.

Para a base de dados com imagens de 50m, a YOLOv7 apresentou resultados médios em torno de 52% para a métrica F1 em seu melhor treinamento (*transfer learning*), enquanto para a base de 100m, observou-se que o melhor treinamento ocorreu com o carregamento de pesos aleatórios (*from scratch*), com resultados médios em torno de 55% para a métrica F1. Já para os modelos da *Segmentation Models*, observou-se uma taxa média de 64% para a métrica F1 na base de 50m e, para a base de 100m, a taxa foi de 52% para a mesma métrica. Desse modo, a melhor performance observada foi com os modelos da biblioteca *Segmentation Models* para a base de 50m.

Este trabalho permitiu um estudo aprofundado de modelos de aprendizagem pro-

funda para a segmentação de objetos. Este estudo se baseou em imagens complexas: por vezes, sem uma resolução adequada e objetos de interesse sem um formato definido, o que elevou a dificuldade do trabalho. Ainda assim, considera-se que os resultados obtidos foram satisfatórios e permitiram uma investigação preliminar da problemática.

Avaliando-se oportunidades futuras de melhorias, pode-se identificar o aumento da base de dados, uma vez que, por se tratar de uma atividade manual, houve uma limitação de tempo para os autores se dedicarem à atividade de coleta e anotações das imagens para as bases de dados. O aumento do número de imagens coletadas e entidades anotadas pode contribuir para um treinamento mais robusto da base de dados, permitindo resultados potencialmente melhores. Ainda, poderia ser explorada uma técnica de *data augmentation*, que, utilizando da mesma lógica anterior, poderia aumentar a performance do modelo pelo aumento da base de dados.

Outro caminho possível de ser explorado seria buscar uma fonte diferente para retirada das imagens, uma vez que poderia ser possível obter imagens com melhor resolução e poderia-se realizar análises a diferentes escalas, investigando novos intervalos. Além disso, a expansão para modelos mais recentes e poderosos como a YOLOv8 e outros modelos de segmentação mais robustos também pode apresentar incrementos aos resultados obtidos.

As bases de dados construídas e as anotações poderão ser utilizadas em trabalhos futuros. Além disso, acredita-se que a rede neural treinada poderá ser utilizada para treinamento de outros modelos com finalidade semelhante para outras áreas do Brasil, apesar de não ser possível garantir a performance da rede devido aos diferentes perfis geográficos de cada região.

Referências

ABADE, A.; PORTO, L. F.; FERREIRA, P. A.; de Barros Vidal, F. Nemanet: A convolutional neural network model for identification of soybean nematodes. **Biosystems Engineering**, v. 213, p. 39–62, 2022. ISSN 1537-5110. Disponível em: https://www.sciencedirect.com/science/article/pii/S153751102100283X. Citado na p. 28.

AGGARWAL, C. **Neural Networks and Deep Learning: A Textbook**. [*S.l.*: *s.n.*], 2018. ISBN 978-3-319-94462-3. Citado nas pp. 10, 20, 21 e 22.

ALI, N. N.; KAKO, N. A.; ABDI, A. S. Review on image segmentation methods using deep learning. **2022 4th International Conference on Advanced Science and Engineering (ICOASE)**, p. 7–12, 2022. Citado na p. 29.

BARRETO, S. C.; LAMBERT, J. A.; VIDAL, F. de B. Using synthetic images for deep learning recognition process on automatic license plate recognition. *In*: CARRASCO-OCHOA, J. A.; MARTÍNEZ-TRINIDAD, J. F.; OLVERA-LÓPEZ, J. A.; SALAS, J. (Ed.). **Pattern Recognition**. Cham: Springer International Publishing, 2019. p. 115–126. ISBN 978-3-030-21077-9. Citado na p. 31.

BARROS., G. O.; WANDERLEY., D. C.; REBOUçAS., L. O.; SANTOS., W. L. C. dos; DUARTE., A. A.; VIDAL., F. de B. Podnet: Ensemble-based classification of podocytopathy on kidney glomerular images. *In*: INSTICC. **Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2022) - Volume 5: VISAPP**. [*S.l.*]: SciTePress, 2022. p. 405–412. ISBN 978-989-758-555-5. ISSN 2184-4321. Citado na p. 28.

BOCHKOVSKIY, A.; WANG, C.-Y.; LIAO, H.-Y. M. Yolov4: Optimal speed and accuracy of object detection. **ArXiv**, abs/2004.10934, 2020. Citado na p. 32.

CALIARI, A.; BERDEJA, D.; MIRANDA, P.; OSTFELD, S. **Atlas de Vulnerabilidad: La Pandemia en America Latina y el Caribe**. [*S.l.*], 2021. Citado na p. 15.

CHAURASIA, A.; CULURCIELLO, E. Linknet: Exploiting encoder representations for efficient semantic segmentation. *In*: **2017 IEEE Visual Communications and Image Processing (VCIP)**. IEEE, 2017. Disponível em: http://dx.doi.org/10.1109/VCIP.2017.8305148. Citado na p. 30.

CHEN, J.-N.; SUN, S.; HE, J.; TORR, P.; YUILLE, A.; BAI, S. **TransMix: Attend to Mix for Vision Transformers**. 2021. Citado na p. 31.

CHEN, L.-C.; PAPANDREOU, G.; SCHROFF, F.; ADAM, H. **Rethinking Atrous Convolution for Semantic Image Segmentation**. 2017. Citado na p. 30.

CHEN, L.-C.; ZHU, Y.; PAPANDREOU, G.; SCHROFF, F.; ADAM, H. **Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation**. 2018. Citado na p. 30.

CHEN, Y.; LI, J.; XIAO, H.; JIN, X.; YAN, S.; FENG, J. **Dual Path Networks**. 2017. Citado na p. 31.

CODEPLAN. **PDAD 2021**. 2021. https://www.codeplan.df.gov.br/pdad-2021-3/. Acessado em: 17 de junho de 2024. Citado na p. 16.

COECKELBERGH, M. Ai for climate: freedom, justice, and other ethical and political challenges. **AI and Ethics**, v. 1, n. 1, 2021. Citado na p. 15.

Dwyer, B., Nelson, J., Hansen, T. **Roboflow [Software]**. 2024. Disponível em: https://roboflow.com – acesso em 1 mai. 2023. Citado na p. 41.

FAN, T.; WANG, G.; LI, Y.; WANG, H. Ma-net: A multi-scale attention network for liver and tumor segmentation. **IEEE Access**, v. 8, p. 179656–179665, 2020. Citado na p. 30.

FORSYTH, D. A.; PONCE, J. **Computer Vision: A Modern Approach**. [*S.l.*]: Prentice Hall Professional Technical Reference, 2002. ISBN 0130851981. Citado na p. 28.

GAO, S.-H.; CHENG, M.-M.; ZHAO, K.; ZHANG, X.-Y.; YANG, M.-H.; TORR, P. Res2net: A new multi-scale backbone architecture. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, Institute of Electrical and Electronics Engineers (IEEE), v. 43, n. 2, p. 652–662, fev. 2021. ISSN 1939-3539. Disponível em: http://dx.doi.org/10.1109/TPAMI.2019.2938758. Citado na p. 31.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [*S.l.*]: MIT Press, 2016. http://www.deeplearningbook.org. Citado nas pp. 21, 25, 26 e 27.

Google. **Google Earth**. 2023. Disponível em: https://www.google.com/earth/ – acesso em 4 abr. 2023. Citado nas pp. 41 e 42.

HE, K.; ZHANG, X.; REN, S.; SUN, J. Deep residual learning for image recognition. *In*: **2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [*S.l.: s.n.*], 2016. p. 770–778. Citado na p. 31.

HEBB, D. O. **The organization of behavior: A neuropsychological theory**. New York: Wiley, 1949. Hardcover. ISBN 0-8058-4300-0. Citado na p. 21.

HOWARD, A. G.; ZHU, M.; CHEN, B.; KALENICHENKO, D.; WANG, W.; WEYAND, T.; ANDREETTO, M.; ADAM, H. **MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications**. 2017. Citado na p. 31.

HU, J.; SHEN, L.; ALBANIE, S.; SUN, G.; WU, E. **Squeeze-and-Excitation Networks**. 2019. Citado na p. 31.

HUANG, G.; LIU, Z.; MAATEN, L. van der; WEINBERGER, K. Q. **Densely Connected Convolutional Networks**. 2018. Citado na p. 31.

IAKUBOVSKII, P. **Segmentation Models Pytorch**. [*S.l.*]: GitHub, 2019. https://github.com/qubvel/segmentation_models.pytorch. Citado nas pp. 30 e 50.

IPEDF – Instituto de Pesquisa e Estatística do Distrito Federal. **Evolução do Índice de Vulnerabilidade Social do Distrito Federal (IVS-DF) 2018-2021. Relatório**. Brasília, 2024. Citado nas pp. 15 e 16.

KIRILLOV, A.; HE, K.; GIRSHICK, R.; DOLLÁR, P. A unified architecture for instance and semantic segmentation. *In*: CVPR. **Computer Vision and Pattern Recognition Conference**. [*S.l.*], 2017. Citado na p. 30.

LECUN LéON BOTTOU, Y. B. Y.; HAFFNER, P. Gradient-based learning applied to document recognition. **Proceedings of the IEEE**, v. 86, 1998. Citado na p. 25.

LI, C.; LI, L.; JIANG, H.; WENG, K.; GENG, Y.; LI, L.; KE, Z.; LI, Q.; CHENG, M.; NIE, W.; LI, Y.; ZHANG, B.; LIANG, Y.; ZHOU, L.; XU, X.; CHU, X.; WEI, X.; WEI, X. Yolov6: A single-stage object detection framework for industrial applications. **ArXiv**, abs/2209.02976, 2022. Citado na p. 32.

LI, H.; XIONG, P.; AN, J.; WANG, L. **Pyramid Attention Network for Semantic Segmentation**. 2018. Citado na p. 30.

LI, X.; WANG, W.; HU, X.; YANG, J. **Selective Kernel Networks**. 2019. Citado na p. 31.

LIN, M.; CHEN, H.; SUN, X.; QIAN, Q.; LI, H.; JIN, R. **Neural Architecture Design for GPU-Efficient Networks**. 2020. Citado na p. 31.

MENSAH, J. Sustainable development: Meaning, history, principles, pillars, and implications for human action: Literature review. **Cogent Social Sciences**, v. 5, n. 1, 2019. Citado na p. 15.

NETO, A.; BONINI, C. D. S. B. Redes neurais artificiais: ApresentaÇÃo e utilizaÇÃo do algoritmo perceptron em biossistemas / artificial neural networks: Introduction and use of perceptron algorithm in biossystems. **Revista Brasileira de Engenharia de Biossistemas**, v. 4, 11 2014. Citado na p. 20.

PADILLA, R.; NETTO, S. L.; SILVA, E. A. B. da. A survey on performance metrics for object-detection algorithms. *In*: **2020 International Conference on Systems, Signals and Image Processing (IWSSIP)**. [*S.l.: s.n.*], 2020. p. 237–242. Citado na p. 35.

PASZKE, A.; GROSS, S.; MASSA, F.; LERER, A.; BRADBURY, J.; CHANAN, G.; KILLEEN, T.; LIN, Z.; GIMELSHEIN, N.; ANTIGA, L.; DESMAISON, A.; KOPF, A.; YANG, E.; DEVITO, Z.; RAISON, M.; TEJANI, A.; CHILAMKURTHY, S.; STEINER, B.; FANG, L.; BAI, J.; CHINTALA, S. Pytorch: An imperative style, high-performance deep learning library. *In*: **Advances in Neural Information Processing Systems 32**. Curran Associates, Inc., 2019. p. 8024–8035. Disponível em: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf. Citado na p. 48.

RADOSAVOVIC, I.; KOSARAJU, R. P.; GIRSHICK, R.; HE, K.; DOLLáR, P. **Designing Network Design Spaces**. 2020. Citado na p. 31.

REDMON, J.; DIVVALA, S.; GIRSHICK, R. B.; FARHADI, A. You only look once: Unified, real-time object detection. **2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**, 2015. Disponível em: https://www.semanticscholar.org/paper/f8e79ac0ea341056ef20f2616628b3e964764cfd. Citado na p. 31.

REDMON, J.; FARHADI, A. Yolo9000: Better, faster, stronger. **2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**, p. 6517–6525, 2016. Citado na p. 32.

REDMON, J.; FARHADI, A. Yolov3: An incremental improvement. **ArXiv**, abs/1804.02767, 2018. Citado na p. 32.

RONNEBERGER, O.; FISCHER, P.; BROX, T. **U-Net: Convolutional Networks for Biomedical Image Segmentation**. 2015. Citado na p. 30.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. **Psychological review**, 1958. Citado na p. 21.

SALAS, J.; PATTERSON, G.; VIDAL, F. de B. A systematic mapping of artificial intelligence solutions for sustainability challenges in latin america and the caribbean. **IEEE Latin America Transactions**, v. 20, n. 11, 2022. Citado nas pp. 15, 17, 18 e 38.

SALAS, J.; VERA, P.; ZEA-ORTIZ, M.; VILLASEñOR, E.-A.; PULIDO, D.; FIGUEROA, A. Fine-grained large-scale vulnerable communities mapping via satellite imagery and population census using deep learning. **Remote Sensing**, v. 13, 2021. Disponível em: https://www.mdpi.com/2072-4292/13/18/3603. Citado na p. 39.

SALAS, J.; VIDAL, F.; MARTíNEZ-TRINIDAD, J. Deep learning: Current state. **IEEE LATIN AMERICA TRANSACTIONS**, v. 17, 2019. Citado nas pp. 23 e 38.

SIMONYAN, K.; ZISSERMAN, A. **Very Deep Convolutional Networks for Large-Scale Image Recognition**. 2015. Citado na p. 31.

SMITH, L. N. **A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay**. 2018. Citado nas pp. 24 e 34.

SMITH, S. L.; KINDERMANS, P.-J.; YING, C.; LE, Q. V. **Don't Decay the Learning Rate, Increase the Batch Size**. 2018. Citado na p. 34.

SZEGEDY, C.; LIU, W.; JIA, Y.; SERMANET, P.; REED, S.; ANGUELOV, D.; ERHAN, D.; VANHOUCKE, V.; RABINOVICH, A. **Going Deeper with Convolutions**. 2014. Citado na p. 31.

TAN, M.; LE, Q. V. **EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks**. 2020. Citado na p. 31.

VASU, P. K. A.; GABRIEL, J.; ZHU, J.; TUZEL, O.; RANJAN, A. **MobileOne: An Improved One millisecond Mobile Backbone**. 2023. Citado na p. 31.

WALTER, M. **Extractives in Latin America and the Caribbean. The Basics**. [*S.l.*]: Inter-American Development Bank, 2016. Citado na p. 15.

WANG, C.-Y.; BOCHKOVSKIY, A.; LIAO, H.-Y. M. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. **ArXiv**, abs/2207.02696, 2022. Citado nas pp. 32 e 48.

WERBOS, P. Backpropagation through time: what it does and how to do it. **Proceedings of the IEEE**, 1990. Citado na p. 21.

WIDROW, B.; HOFF, M. E. Adaptive switching circuits. *In*: . [*s.n.*], 1988. Disponível em: https://api.semanticscholar.org/CorpusID:60830585. Citado na p. 21.

WS, P. W. M. **A logical calculus of the ideas immanent in nervous activity"**. [*S.l.*]: Bulletin of mathematical biology, 1943. Citado na p. 21.

XIE, S.; GIRSHICK, R.; DOLLáR, P.; TU, Z.; HE, K. **Aggregated Residual Transformations for Deep Neural Networks**. 2017. Citado na p. 31.

ZHANG, H.; WU, C.; ZHANG, Z.; ZHU, Y.; LIN, H.; ZHANG, Z.; SUN, Y.; HE, T.; MUELLER, J.; MANMATHA, R.; LI, M.; SMOLA, A. **ResNeSt: Split-Attention Networks**. 2020. Citado na p. 31.

ZHANG, H.; ZHANG, L.; JIANG, Y. Overfitting and underfitting analysis for deep learning based end-to-end communication systems. *In*: **2019 11th International Conference on Wireless Communications and Signal Processing (WCSP)**. [*S.l.: s.n.*], 2019. p. 1–6. Citado nas pp. 24 e 25.

ZHAO, H.; SHI, J.; QI, X.; WANG, X.; JIA, J. Pyramid scene parsing network. *In*: **2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [*S.l.: s.n.*], 2017. p. 6230–6239. Citado nas pp. 30 e 33.

ZHIQIANG, W.; JUN, L. A review of object detection based on convolutional neural network. *In*: **2017 36th Chinese Control Conference (CCC)**. [*S.l.: s.n.*], 2017. p. 11104–11109. Citado na p. 29.

ZHOU, Z.; SIDDIQUEE, M. M. R.; TAJBAKHSH, N.; LIANG, J. **UNet++: A Nested U-Net Architecture for Medical Image Segmentation**. 2018. Citado na p. 30.