UNIVERSIDADE FEDERAL DO AMAPÁ
DEPARTAMENTO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

LUCAS FERRO ZAMPAR

MY BIRDS - UTILIZANDO INTELIGÊNCIA ARTIFICIAL PARA A DETECÇÃO DE PÁSSAROS AMAZÔNICOS

MACAPÁ
AGOSTO 2023

LUCAS FERRO ZAMPAR

MY BIRDS - UTILIZANDO INTELIGÊNCIA ARTIFICIAL PARA A DETECÇÃO DE PÁSSAROS AMAZÔNICOS

Trabalho de Conclusão de Curso apresentado a Universidade Federal do Amapá como requisito parcial para obtenção do título de Bacharel em Ciência da Computação. Área de concentração: Bacharelado em Ciência da Computação.

Orientador: Prof. Dr. Clay Palmeira da Silva

MACAPÁ

AGOSTO 2023

Dados Internacionais de Catalogação na Publicação (CIP)
Biblioteca Central/UNIFAP-Macapá-AP
Elaborado por Mário das Graças Carvalho Lima Júnior – CRB-2 / 1451

Z26 Zampar, Lucas Ferro.
My birds: utilizando inteligência artificial para a detecção de pássaros amazônicos / Lucas Ferro Zampar. - Macapá, 2023.
1 recurso eletrônico. 82 folhas.

Trabalho de Conclusão de Curso (Graduação) - Universidade Federal do Amapá, Coordenação do Curso de Ciência da Computação, Macapá, 2023.
Orientador: Clay Palmeira da Silva.

Modo de acesso: World Wide Web.
Formato de arquivo: Portable Document Format (PDF).

1. Aprendizado profundo. 2. Detecção de objetos. 3. Pássaros amazônicos. I. Silva, Clay Palmeira da, orientador. II. Universidade Federal do Amapá. III. Título.

CDD 23. ed. – 004

ZAMPAR, Lucas Ferro. **My birds**: utilizando inteligência artificial para a detecção de pássaros amazônicos. Orientador: Clay Palmeira da Silva. 2023. 82 f. Trabalho de Conclusão de Curso (Graduação) - Coordenação do Curso de Ciência da Computação. Universidade Federal do Amapá, Macapá, 2023.

UNIVERSIDADE FEDERAL DO AMAPÁ
DEPARTAMENTO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
COORDENAÇÃO DO CURSO DE CIÊNCIA DA COMPUTAÇÃO

ATA DE DEFESA DE TCC

Realizou-se, no dia 15 de agosto de 2023, às 17h, no laboratório de redes, a defesa do TCC intitulado: **"MyBirds - Utilizando inteligência artificial para a detecção de pássaros amazônicos"**, do discente **Lucas Ferro Zampar**, matrícula 2017002908. A Banca Examinadora foi composta pelo Prof. Dr. Clay Palmeira da Silva, presidente da banca e orientadora; Prof. Dr. José Walter Cárdenas Sotil e Prof. Esp. Ivson Monteiro Viana, examinadores. Concluída a defesa, foram realizadas as arguições e comentários. Em seguida, procedeu-se o julgamento pelos membros da Banca Examinadora, tendo o trabalho sido APROVADO.

E, para constar, eu, Prof. Dr. Clay Palmeira da Silva, orientador e presidente da Banca Examinadora, lavrei a presente ata que, após lida e achada conforme, foi assinada por mim e demais membros da Banca Examinadora.

Macapá-Ap., 15 de agosto de 2023.

gov.br Documento assinado digitalmente
CLAY PALMEIRA DA SILVA
Data: 22/08/2023 17:11:20-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Clay Palmeira da Silva
Orientador do Projeto de TCC

gov.br Documento assinado digitalmente
JOSE WALTER CARDENAS SOTIL
Data: 16/08/2023 09:43:13-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. José Walter Cárdenas Sotil
Examinador (UNIFAP)

gov.br Documento assinado digitalmente
IVSON MONTEIRO VIANA
Data: 22/08/2023 22:21:03-0300
Verifique em https://validar.iti.gov.br

Prof. Esp. Ivson Monteiro Viana
Examinador (UNIFAP)

UNIVERSIDADE FEDERAL DO AMAPÁ
DEPARTAMENTO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
COORDENAÇÃO DO CURSO DE CIÊNCIA DA COMPUTAÇÃO

FICHA DE AVALIAÇÃO DE TCC

Discente: **Lucas Ferro Zampar**, matrícula 2017002908.

Título: **"MyBirds - Utilizando inteligência artificial para a detecção de pássaros amazônicos".**

Orientador (a): Prof. Dr. Clay Palmeira da Silva

Examinador(a) 1 da Banca Avaliadora: Prof. Dr. José Walter Cárdenas Sotil

Examinador(a) 2 da Banca Avaliadora: Prof. Esp. Ivson Monteiro Viana

	Avaliador 1	Avaliador 2
Trabalho Escrito (0 a 5)	5	5
Apresentação Oral (0 a 5)	5	5
Nota Final	NF1 = 10	NF2 = 10

No item TRABALHO ESCRITO, a banca examinadora deverá avaliar: organização sequencial, argumentação, profundidade do tema, relevância e contribuição acadêmica da pesquisa, correção gramatical, clareza, apresentação estética, adequação aos aspectos formais às normas da ABNT.

No item APRESENTAÇÃO ORAL, a banca examinadora deverá avaliar: domínio do conteúdo, organização da apresentação, habilidades de comunicação e expressão, capacidade de argumentação, uso dos recursos audiovisuais, correção gramatical e apresentação estética do trabalho.

MÉDIA FINAL: A média final será calculada pela soma das duas notas finais (NF1 e NF2) dividida por dois.

$$\text{MÉDIA FINAL} = \frac{NF1 + NF2}{2} = 10.$$

gov.br Documento assinado digitalmente
CLAY PALMEIRA DA SILVA
Data: 15/08/2023 18:37:25-0300
Verifique em https://validar.iti.gov.br

Macapá-Ap., 15 de agosto de 2023.

_____
Prof. Dr. Clay Palmeira da Silva
Orientador do Projeto de TCC

gov.br Documento assinado digitalmente
JOSE WALTER CARDENAS SOTIL
Data: 16/08/2023 09:39:41-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. José Walter Cárdenas Sotil
Examinador (UNIFAP)

gov.br Documento assinado digitalmente
IVSON MONTEIRO VIANA
Data: 16/08/2023 10:51:20-0300
Verifique em https://validar.iti.gov.br

_____
Prof. Esp. Ivson Monteiro Viana
Examinador (UNIFAP)

Dedico este trabalho aos meus pais que sempre me apoiaram.

AGRADECIMENTOS

Agradeço aos meus pais que sempre ofereceram todo suporte necessário a fim de alcançar meus objetivos, tanto pessoais quanto acadêmicos. Gostaria também de agradecer ao meu orientador pela grande compreensão que teve comigo durante essa jornada, além de ter disponibilizado recursos essenciais para a conclusão deste trabalho.

"Estude muito o que mais lhe interessa da maneira mais indisciplinada, irreverente e original possível." (Richard Feynman)

RESUMO

Pássaros despertam a atenção humana pela sua beleza e diversidade, o que estimula admiradores a praticar a atividade recreativa de avistá-los, bem como de registrá-los em imagens. O compartilhamento desses registros em plataformas de ciência cidadã, como WikiAves e eBirds, pode contribuir significativamente com pesquisas científicas que visam compreender e preservar essas espécies. Nesse contexto, o avistamento de pássaros pode ocorrer por meio de comedouros localizados em jardins de residências, o que contribui com o bem-estar dos moradores locais. Diante disso, notou-se a oportunidade de registar pássaros em comedouros através de webcams. Além disso, questionou-se se seria possível detectar automaticamente suas espécies por meio do aprendizado profundo, técnica amplamente adotada na visão computacional. Dessa forma, o presente trabalho visa levantar uma abordagem baseada em aprendizado profundo para detectar espécies de pássaros que visitam comedouros residenciais. Para tanto, o estudo teve acesso ao comedouro de uma residência do município de Santana, no Amapá, região que está inserida no contexto amazônico. Levantou-se um conjunto de dados disponibilizado publicamente que é composto por 940 imagens de pássaros e 1.836 anotações distribuídas entre 5 classes referentes às espécies identificadas. As imagens foram obtidas ao extrair frames das gravações feitas pelas webcams, além de terem sido anotadas para a tarefa de detecção de objetos. O conjunto foi empregado para o treinamento de diferentes modelos do tipo Faster R-CNN ao longo de duas fases consecutivas denominadas de preliminar e de final. Na primeira, uma porção menor dos dados foi utilizada para definir a configuração de treinamento e uma baseline. Na segunda, um único modelo chamado de definitivo foi treinado com a totalidade dos dados através da configuração definida, além de ter sido comparado com a baseline. A avaliação dos modelos ocorreu por meio das métricas mAP, precisão e revocação, além do emprego de matrizes de confusão. Ao considerar a Intersection over Union em 50%, o modelo definitivo conseguiu alcançar mAP de 98,33%, precisão média de 95,96% e revocação média de 98,82%. Até onde se sabe, My Birds é o primeiro trabalho a propor a detecção de espécies de pássaros amazônicos em comedouros residenciais, além de levantar um conjunto anotado dessas espécies. Dessa forma, é possível visualizar trabalhos futuros que foquem na coleta automatizada de mais imagens em outras residências de modo a contribuir com o levantamento de dados sobre essas espécies.

Palavras-chave: Aprendizado profundo; detecção de objetos; espécies de pássaros; Faster R-CNN;

ABSTRACT

Birds arouse human attention for their beauty and diversity, encouraging admirers to practice the recreational activity of spotting them and registering them in images. Sharing these records on citizen science platforms, such as WikiAves and eBirds, can significantly contribute to scientific research to understand and preserve these species. In this context, the sighting of birds can occur through feeders located in the gardens of residences, which contributes to the well-being of residents. The opportunity to record birds in feeders through webcams was noted. In addition, it was questioned whether it would be possible to automatically detect their species through deep learning, a technique widely adopted in computer vision. In this way, the present work aims to raise an approach based on deep learning to detect bird species that visit residential feeders. Therefore, the study had access to the feeder of a residence in the municipality of Santana, in Amapá, a region inserted in the Amazonian context. A publicly available dataset was collected, consisting of 940 images of birds and 1,836 notes distributed among 5 classes referring to the identified species. The images were obtained by extracting frames from the recordings made by the webcams, in addition to being annotated for the object detection task. The set was used to train different models of the Faster R-CNN type over two consecutive phases called preliminary and final. In the first, a smaller portion of the data was used to define the training configuration and a baseline. In the second, a single model called definitive was trained with all the data through the specified configuration, in addition to being compared with the baseline. The models were evaluated using the mAP, precision, and recall metrics, in addition to the use of confusion matrices. When considering the Intersection over Union at 50%, the definitive model achieved an mAP of 98.33%, a mean precision of 95.96%, and a mean recall of 98.82%. To the best of our knowledge, My Birds is the first work to propose detecting Amazonian bird species in residential feeders and raising an annotated set of these species. In this way, it is possible to visualize future works focusing on the automated collection of more images in other residences to contribute to data collection on these species.

Keywords: Deep learning; object detection; bird species; Faster R-CNN;

LISTA DE EQUAÇÕES

Equação 1 — Valores do mapa de característica H gerados pelo n-ésimo filtro de convolução da camada. 25
Equação 2 — Definição matemática da função ReLU. 25
Equação 3 — Valor de saída de um neurônio artificial. 26
Equação 4 — Atualização dos parâmetros de um modelo. 28
Equação 5 — Médias dos gradientes da função de perda em relação aos parâmetros do modelo. 28
Equação 6 — Definição da métrica Intersection over Union. 30
Equação 7 — Definição da métrica precisão 31
Equação 8 — Definição da métrica revocação 32

LISTA DE FIGURAS

Figura 1 — Resumo das etapas de implementação do projeto. 18
Figura 2 — Exemplo de detecção de objetos na qual dois pássaros da espécie sanhaço-do-coqueiro são localizados e classificados. 20
Figura 3 — Arquitetura da LeNet-5. 23
Figura 4 — Ilustração da operação de convolução aplicada sobre uma matriz de entrada. 24
Figura 5 — Última camada da LeNet-5 referente aos 10 dígitos. 27
Figura 6 — Curva de precisão por revocação. 32
Figura 7 — Exemplo de matriz de confusão. 35
Figura 8 — Diagrama relacionando o conjunto total e parcial. 40
Figura 9 — Matriz de confusão para o modelo 5 eleito na etapa de seleção do backbone. 44
Figura 10 — Matriz de confusão para a instância 5.14 eleita na etapa de ajuste de hiperparâmetros. 50
Figura 11 — Comedouro da residência onde eram dispostos alimentos para atrair os pássaros. 53
Figura 12 — Esquema para gravação dos pássaros. 54
Figura 13 — Interface da aplicação desenvolvida em Streamlit para extração dos frames dos vídeos. 55
Figura 14 — Fluxo de trabalho para o treinamento de um modelo de detecção de objetos por meio do framework IceVision. 57
Figura 15 — Exemplos de transformações em imagens. 58
Figura 16 — Detecções realizadas para dados não empregados durante o treinamento e avaliação. 61
Figura 17 — Matriz de confusão para o modelo definitivo. 65

LISTA DE GRÁFICOS

Gráfico 1 — Proporção de anotação por espécie da parcela de teste do conjunto de dados parcial. 41
Gráfico 2 — Proporção de anotação por espécie da parcela de teste do conjunto de dados total. 52
Gráfico 3 — Proporção de anotações por espécie do conjunto de dados total. . . . 56

LISTA DE TABELAS

Tabela 1 — Resumo das categorias de avaliação de uma detecção. 31
Tabela 2 — Comparação deste trabalho com os demais trabalhos relacionados 38
Tabela 3 — Configuração de treinamento dos modelos na etapa de seleção do backbone. 42
Tabela 4 — Desempenho alcançado pelos modelos com cada rede de backbone. 42
Tabela 5 — Métricas finais de precisão e de revocação do modelo 5 para cada espécie ao se considerar 50% como o limiar de IoU. 43
Tabela 6 — Descrição dos cincos testes realizados na etapa de ajuste de hiperparâmetros. 45
Tabela 7 — Métricas finais no ajuste do número de epochs e do embaralhamento do conjunto de treinamento em cada uma. 46
Tabela 8 — Métricas finais no ajuste do tamanho do batch. 46
Tabela 9 — Métricas finais no ajuste da taxa de aprendizagem com 20 epochs de treinamento. 46
Tabela 10 — Métricas finais no ajuste do tamanho da imagem. 47
Tabela 11 — Métricas finais no ajuste do tamanho da imagem acompanhado de presizing. 47
Tabela 12 — Métricas finais de precisão e de revocação da instância 5.10 para cada espécie ao se considerar 50% como o limiar de IoU. 48
Tabela 13 — Métricas finais de precisão e de revocação da instância 5.14 para cada espécie ao se considerar 50% como o limiar de IoU. 48
Tabela 14 — Configuração de treinamento na fase final. 51
Tabela 15 — Comparação dos valores das métricas mAP alcançadas pela baseline e pelo modelo definitivo. 63
Tabela 16 — Métricas finais de precisão e de revocação do modelo definitivo para cada espécie ao se considerar 50% como o limiar de IoU. . . . 63
Tabela 17 — Métricas finais de precisão e de revocação da baseline e do modelo definitivo ao se considerar 50% como o limiar de IoU. 64

LISTA DE ABREVIATURAS E SIGLAS

AP	Average Precision
CNN	Convolutional Neural Network
FN	Falso Negativo
FP	Falso Positivo
FPN	Feature Pyramid Head
GPU	Graphical Processing Unit
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
IoU	Intersection over Union
mAP	mean Average Precision
RoI	Region of Interest
VP	Verdadeiro Positivo

SUMÁRIO

1 **INTRODUÇÃO** 15
1.1 PROBLEMA 16
1.2 JUSTIFICATIVA 16
1.3 OBJETIVOS 16
1.3.1 **OBJETIVO GERAL** 16
1.3.2 **OBJETIVOS ESPECÍFICOS** 17
1.4 METODOLOGIA 17
2 **REFERENCIAL TEÓRICO** 19
2.1 DETECÇÃO DE OBJETOS 19
2.2 REDES NEURAIS CONVOLUCIONAIS 22
2.3 TREINAMENTO 27
2.4 MÉTRICAS DE PERFORMANCE 30
2.4.1 **Average Precision** 32
2.4.2 **Matriz de Confusão** 33
2.5 TRABALHOS RELACIONADOS 35
3 **CONSTRUÇÃO DO MODELO** 39
3.1 FASE PRELIMINAR 39
3.1.1 **Seleção do *Backbone*** 41
3.1.1.1 Resumo da subseção 44
3.1.2 **Ajuste de Hiperparâmetros** 45
3.1.2.1 Resumo da subseção 50
3.2 FASE FINAL 51
4 **PROPOSTA** 53
4.1 CONJUNTO DE DADOS 53
4.2 TREINAMENTO 57
4.3 APLICAÇÃO 60
4.4 AVALIAÇÃO 61
4.5 RESULTADO FINAL 62
5 **CONCLUSÃO** 66
REFERÊNCIAS 68
APÊNDICE A — EXEMPLIFICAÇÃO DE VP, FP E FN 72
APÊNDICE B — CONFUSÕES DO MODELO DEFINIDO NA FASE DE SELEÇÃO DA BACKBONE 75
APÊNDICE C — CONFUSÕES DO MODELO DEFINIDO NA FASE DE AJUSTE DE HIPERPARÂMETROS 78
APÊNDICE D — CONFUSÕES DO MODELO DEFINITIVO 80

1 **INTRODUÇÃO**

Apesar de ser uma tarefa que humanos realizam naturalmente, contemplar computadores com a capacidade de interpretar imagens é um grande desafio enfrentado pela visão computacional. No entanto, na última década, o campo de aprendizado profundo sofreu um rápido crescimento e tornou essa atividade, como muitas outras, cada vez mais viável por meio de complexos modelos computacionais usualmente denominados de redes neurais.

Nesse contexto, o aprendizado profundo é empregado de forma recorrente em diversas aplicações atuais de visão computacional tais como reconhecimento facial, localização de veículos, bem como de pedestres, por carros autônomos e identificação de anomalias em imagens médicas (HOWARD; GUGGER, 2020). Além disso, a visão computacional também é aplicada na observação de animais silvestres, por exemplo, na detecção automática de pássaros em imagens (MIRUGWE; NYIRENDA; DUFOURQ, 2022).

Hoje, uma fonte importante de levantamento de dados sobre esses animais ocorre por meio dos praticantes da atividade de avistamento de pássaros. Segundo Barbosa et al. (2021), ao disponibilizarem registros em plataformas de ciência cidadã, como o WikiAves e eBirds, os avistadores podem contribuir significativamente com estudos ornitológicos que envolvam a observação de aves migratórios em regiões urbanas, comportamento de nidificação, mudanças climáticas e no uso da terra, distribuição de espécies bem como a conservação delas.

Nesse sentido, um dos métodos para a promoção da experiência de avistamento de pássaros é o emprego de comedouros nos quais são dispostos alimentos que atraem as aves para observação (ALEXANDRINO et al., 2022). Tais comedouros podem ser estabelecidos, por exemplo, nos jardins de residências, sendo que os moradores locais podem contribuir com o levantamento de dados ao reportarem as espécies e quantidade de indivíduos que avistaram (BONTER; GREIG, 2021).

Vale destacar também que observar os pássaros se alimentando em jardins pode ter um impacto positivo no bem-estar e na sensação de conexão com a natureza dos residentes, assim como na redução da ansiedade (WHITE et al., 2023). Além disso, a atividade de alimentá-los pode estimular a percepção de contribuição com a vida desses animais, bem como a tomada de atitudes que resolvam problemas observados no ambiente (DAYER et al., 2019).

1.1 PROBLEMA

Diante disso, percebeu-se a oportunidade de empregar comedouros residenciais a fim de registrar os pássaros que os frequentam, bem como de associar inteligência artificial, por meio do aprendizado profundo, na detecção automática de suas espécies. Além disso, surgiu a dúvida se o modelo resultante poderia auxiliar de alguma forma no estudo da frequência e dos hábitos alimentares das espécias da região do Amapá, especificamente no município de Santana. Dessa forma, o presente trabalho estabelece como problema de pesquisa: como o aprendizado profundo pode ser aplicado na detecção automática de espécies de pássaros em imagens a partir de um contexto residencial?

1.2 JUSTIFICATIVA

Organizações interessadas em estudos ornitológicos poderiam se beneficiar com um sistema capaz de registrar pássaros em comedouros localizados nos jardins de residências por meio de imagens, bem como em detectar automaticamente suas espécies. Além de servirem como fonte de dados em uma iniciativa de ciência cidadã, o processo de reconhecimento e anotação dessas espécies poderia ser automatizado parcialmente, o que reduziria a carga de trabalho e facilitaria o monitoramento delas ao longo do tempo.

Vale destacar que os moradores locais também se aproveitariam com a detecção automática das espécies na medida que conheceriam melhor aquelas que se alimentam em seus jardins, bem como a frequência com que aparecem. Além disso, enquanto cidadãos cientistas, eles poderiam colaborar com pesquisas científicas que visam compreender e preservar essas espécies. Por exemplo, por meio dessa pesquisa, foi possível construir uma base de dados anotada com registros em imagem de pássaros de diferentes espécies que frequentam um comedouro residencial e que será disponibilizada publicamente.

1.3 OBJETIVOS

1.3.1 **OBJETIVO GERAL**

Diante do que foi exposto, o presente trabalho tem como objetivo geral levantar uma abordagem baseada em aprendizado profundo com a finalidade de detectar automaticamente espécies de pássaros em imagens a partir de um contexto residencial.

1.3.2 **OBJETIVOS ESPECÍFICOS**

De modo alcançar o objetivo geral colocado, estabelecem-se os seguintes objetivos específicos:

- Abordar como o aprendizado profundo pode ser aplicado na detecção de objetos;
- Revisar trabalhos que empreguem o aprendizado profundo na detecção de pássaros;
- Construir um conjunto de imagens anotadas dos pássaros que frequentam o comedouro de uma residência;
- Treinar e avaliar um modelo de aprendizado profundo na tarefa de detectar as espécies desses pássaros.

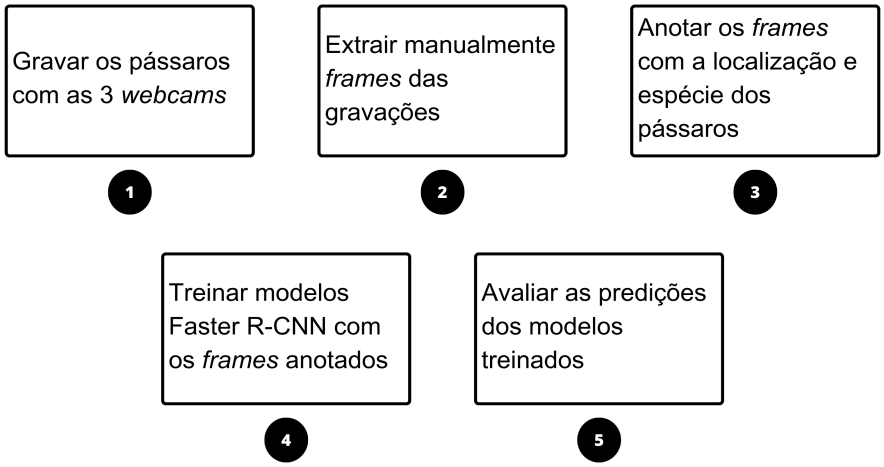
1.4 METODOLOGIA

De modo a alcançar o primeiro e o segundo objetivos específicos, será revisado o estado-da-arte na aplicação e uso do aprendizado profundo para a detecção de objetos em imagens a partir de discussões em fontes secundárias encontradas na literatura. Já para os dois últimos, serão empregadas fontes primárias através de dados coletados em campo e produzidos a partir da análise dos modelos treinados com eles.

Nesse contexto, o tipo de pesquisa será descritiva ao se concentrar na apresentação e discussão dos resultados obtidos a partir do treinamento dos modelos com os dados levantados. Além disso, a abordagem será quantitativa, já que se destina em apresentar resultados numéricos do treinamento desses modelos com base em métricas de avaliação.

A fim de implementar o projeto, os pássaros serão gravados em vídeo por meio de 3 *webcams* inseridas no comedouro presente no jardim de uma residência que está localizada no município de Santana no estado do Amapá. Então, *frames* serão extraídos desses vídeos a fim de comporem as imagens do conjunto de dados, as quais serão anotadas para a tarefa de detecção de objetos em seguida, conforme colocado na Figura 1.

Figura 1 — Resumo das etapas de implementação do projeto.



Fonte: De autoria própria.

De modo a determinar as espécies dos indivíduos registrados, eles serão comparados com registros encontrados em plataformas de ciência cidadã, como o WIKIAVES (2023) e EBIRD (2023), bem como em manuais. Por fim, os dados coletados serão empregados para o treinamento de modelos Faster R-CNN através do IceVision[1] os quais serão avaliados por meio do FiftyOne[2].

Vale destacar que a partir das avaliações realizadas pelo FiftyOne, a análise dos resultados obtidos será dada pela comparação de desempenho dos modelos com o auxílio de tabelas. De modo a quantificar a performance geral do modelo, será utilizada a métrica *mean Average Precision*, enquanto que o comportamento específico em relação a cada espécie será avaliado por meio da precisão e da revocação com o suporte da matriz de confusão.

1 IceVision é um framework disponível em Python que disponibiliza diversos modelos voltados para tarefas de visão computacional, especialmente a detecção de objetos, além de facilitar o treinamento deles.
2 FiftyOne é um framework disponível em Python empregada para visualizar e avaliar as predições realizadas por um modelo em um conjunto de dados voltado para tarefas de visão computacional.

2 **REFERENCIAL TEÓRICO**

Neste capítulo, a tarefa de detecção de objetos será apresentada, bem como sua relação atual com o aprendizado profundo. Nesse contexto, haverá a contextualização do modelo empregado neste trabalho que é o Faster R-CNN. Em seguida, a arquitetura geral das redes neurais convolucionais será discutida, uma vez que ela serve como base para a Faster R-CNN, além de ser utilizada frequentemente pela visão computacional.

Adiante, serão levantados os principais componentes envolvidos no treinamento de um modelo de aprendizado profundo, uma vez que parte do trabalho consiste em treinar o modelo Faster R-CNN com o conjunto de dados construído. Por fim, serão expostas as principais métricas para a avaliação de um detector de objetos de modo que a performance dos modelos treinados possa ser descrita por meio delas.

2.1 DETECÇÃO DE OBJETOS

Entre as tarefas de visão computacional, encontram-se a classificação de imagens e a detecção de objetos. Na classificação, o objeto presente em uma imagem deve ser associado a sua classe mais provável através de um rótulo (Elgendy, 2020). Como exemplo é possível citar a tarefa de atribuir à imagem de um pássaro o rótulo que melhor classifica sua espécie.

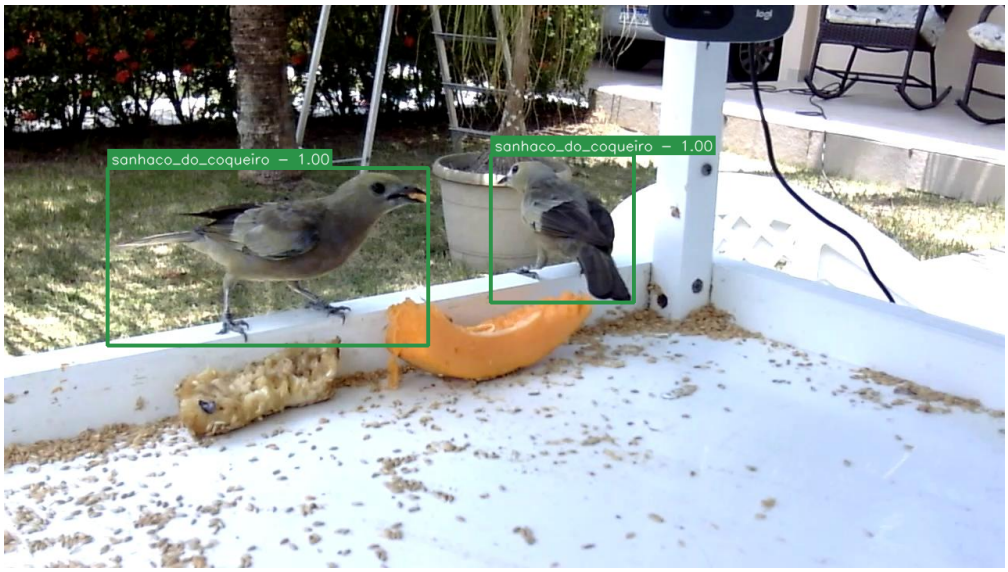
A classificação geralmente se limita a reconhecer um único objeto por imagem. Porém, cenas naturais podem apresentar múltiplos objetos de diferentes classes. Pinheiro e Soares (2021) destacam a dificuldade em classificar imagens com mais de um objeto. Em seu trabalho, os autores empregaram uma rede neural convolucional na classificação de espécies de pássaros na região do Espírito Santo.

Nesse contexto, a detecção de objetos se apresenta como uma evolução da classificação, uma vez que os diversos objetos presentes na imagem devem ser localizados por meio de uma caixa delimitadora, além de serem associados corretamente com suas respectivas classes (Zaidi *et al.*, 2021). Enquanto a classificação tem como objetivo predizer qual objeto se encontra em uma imagem, a detecção prediz onde múltiplos objetos se encontram e o que são tais objetos (Russell; Norvig, 2021).

Dessa forma, um modelo de detecção de objetos deve predizer as coordenadas das caixas delimitadoras que localizam os objetos na imagem, bem como suas respectivas classes. Além disso, ele deve retornar também o nível de confiança que é a probabilidade da detecção realizada estar correta. Na Figura 2, são apresentados exemplos de detecções que localizam e classificam dois pássaros

da espécie sanhaço-do-coqueiro. Ao lado do nome da espécie, é possível visualizar também o nível de confiança das detecções, onde 1.00 representa 100% de confiança.

Figura 2 — Exemplo de detecção de objetos na qual dois pássaros da espécie sanhaço-do-coqueiro são localizados e classificados.



Fonte: De autoria própria.

O progresso do campo de detecção de objetos pode ser dividido em dois períodos em que o primeiro é representado por métodos tradicionais de aprendizagem de máquina e o segundo pela adoção de abordagens baseadas em aprendizado profundo (Zou *et al.*, 2019). Segundo Zaidi *et al.* (2021), em relação ao primeiro período, é possível citar detectores com o Viola-Jones, *Histogram of Oriented Gradients* e *Deformable Parts Model*.

No entanto, métodos convencionais de aprendizagem de máquina possuem limitações na análise de dados naturais como imagens. De acordo com Lecun, Bengio e Hinton (2015), seus modelos necessitam de representações adequadas geradas a partir de extratores de características implementados muitas vezes com conhecimento de especialistas, enquanto os modelos de aprendizado profundo compostos por múltiplas camadas são capazes de aprender automaticamente essas representações em diferentes níveis de abstração.

Nesse contexto, as redes neurais convolucionais, usualmente conhecidas como *convolutional neural networks* (CNNs), são modelos de aprendizado profundo que se destacam na visão computacional. Nelas, características básicas de imagens, como arestas, são aprendidas nas primeiras camadas e gradualmente combinadas até se tornarem características mais complexas, como a forma de um

rosto humano, pelas últimas camadas (Elgendy, 2020).

Desde que as CNNs alcançaram a vitória na competição *ImageNet Large Scale Visual Recognition Challenge* (ILSVRC) de 2012 ao reduzir quase pela metade a taxa de erro na tarefa de classificar objetos entre 1000 categorias em cenas naturais, a comunidade de visão computacional passou a adotá-las amplamente (BENGIO; LECUN; HINTON, 2021). Por exemplo, elas foram incluídas também em tarefas de detecção de objetos com trabalhos como os de Girshick et al. (2014) que apresentaram o método denominado de R-CNN.

No método R-CNN, a partir da imagem de entrada, são propostas regiões de interesse, ou *regions of interest* (RoI) em inglês, por meio do algoritmo de busca seletiva. Vale destacar que tais regiões são representadas por caixas delimitadoras contendo possíveis objetos. Em seguida, cada RoI é repassada para uma CNN pré-treinada com objetivo de extrair características aprendidas durante o treinamento. As características extraídas são fornecidas para uma máquina de vetores de suporte a fim de classificar os objetos. Enquanto isso, um regressor fica encarregado de ajustar as caixas delimitadoras que os localizam. Vale ressaltar que regressores são modelos que retornam valores numéricos como as coordenadas dos pontos que definem a caixa delimitadora.

Apesar de ser uma das primeiras aplicações bem-sucedidas de CNNs na detecção de objetos, a R-CNN guardava algumas desvantagens como a lentidão ao extrair características das RoIs separadamente, além de treinar diferentes modelos de aprendizagem de máquina (Elgendy, 2020). Nesse sentido, Girshick (2015) propôs um algoritmo aperfeiçoado denominado Fast R-CNN, no qual as RoIs são projetadas nos últimos mapas de característica que são extraídos uma única vez a partir da imagem de entrada. Em seguida, elas são redimensionadas para um tamanho fixo e fornecidas para camadas totalmente conectadas que são responsáveis pela classificação e localização dos objetos.

Isso eliminou a necessidade de treinar diferentes modelos de aprendizagem de máquina, além de ter tornado o modelo mais rápido. Porém as RoIs ainda eram propostas por algoritmos como os de busca seletiva. Dessa forma, Ren *et al.* (2015) apresentaram uma modificação chamada de Faster R-CNN de modo que as RoIs passaram a ser propostas dentro do próprio modelo por meio de um módulo, que chamaram de *region proposal network* (RPN), aplicado sobre os últimos mapas de característica extraídos.

Modelos de detecção modernos, como os da família Faster-RCNN, são categorizados como de dois estágios, uma vez que possuem um módulo separado para propor as RoIs (Zaidi *et al.*, 2021). Vale destacar que existem também os de único estágio em que as detecções ocorrem diretamente a partir da imagem por amostragem densa, como a família de detectores YOLO iniciada por

Redmon *et al.* (2015). Nesse sentido, modelos de dois estágios tendem a possuir performance melhor do que os de único estágio, porém ao custo do tempo de inferência maior.

Em geral, a arquitetura de um detector de objetos moderno é constituída por uma rede chamada de *backbone*, responsável pela extração de características a partir da imagem, e por outra denominada de *head*, responsável pela localização e pela classificação (Bochkovskiy; Wang; Liao, 2020). Além disso, é possível encontrar também uma rede intermediária denominada de *neck*, como a *feature pyramid head* (FPN) proposta pelos autores Lin *et al.* (2016) a fim de contribuir com a detecção de objetos em diferentes escalas.

Vale ressaltar que os métodos citados podem predizer diversas caixas delimitadoras. Dessa forma, a técnica de *non-maximum suppression* pode ser empregada a fim de eliminar caixas da mesma classe que se sobrepõem ao deixar apenas aquela com o maior nível de confiança (ELGENDY, 2020).

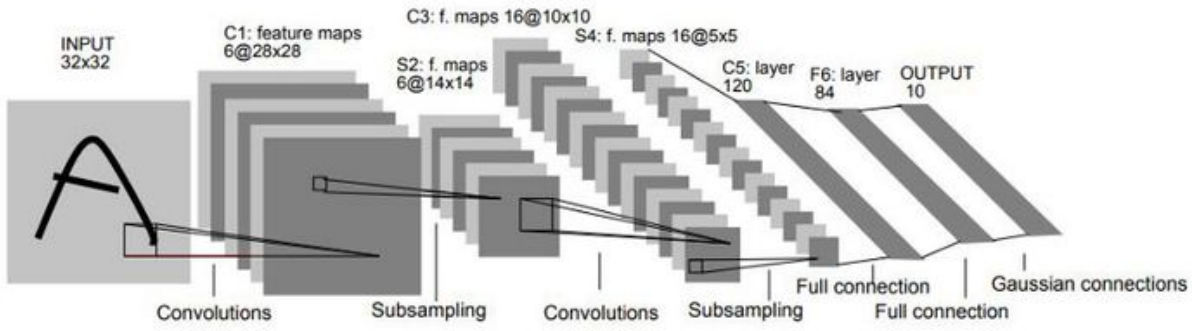
2.2 REDES NEURAIS CONVOLUCIONAIS

A origem do campo de aprendizado profundo está fundamentada nos trabalhos de McCulloch e Pitts que, em 1943, tentaram modelar redes de neurônios no cérebro através de circuitos computacionais (Russell; Norvig, 2021). Por essa forte inspiração na neurociência, os modelos empregados no aprendizado profundo são usualmente conhecidos como redes neurais.

Em 1980, inspirado pelo sistema de visão de mamíferos, Fukushima propôs o modelo Neocognitron que serviria como base para as arquiteturas modernas de CNN (Goodfellow; Bengio; Courville, 2016). Porém, um dos problemas enfrentados nesse momento era a falta de um algoritmo de treinamento efetivo (Russell; Norvig, 2021). Então, em 1989, LeCun e seus colaboradores publicaram o primeiro estudo no qual treinaram com sucesso uma CNN a partir do algoritmo de *backpropagation* que é aplicado até os dias atuais (Zhang *et al.*, 2021).

Em 1998, LeCun e sua equipe também propuseram uma CNN denominada de LeNet-5 e demonstraram que ela era capaz de reconhecer dígitos escritos à mão no conjunto de dados MNIST, o que é considerado um dos principais avanços históricos no campo de inteligência artificial (Howard; Gugger, 2020). A arquitetura da LeNet-5 é apresentada na Figura 3, sendo composto por cinco camadas, duas das quais são convolucionais e as outras três totalmente conectadas.

Figura 3 — Arquitetura da LeNet-5.



Fonte: Lecun *et al.* (1998).

Apesar do potencial das CNNs, a viabilidade de treiná-las em conjuntos de dados mais complexos e realistas foi difícil de ser demonstrada. No entanto, isso mudaria quando Krizhevsky, Sutskever e Hinton (2012) alcançaram o primeiro lugar no desafio ILSVRC de 2012 por meio da AlexNet, uma CNN composta por 8 camadas treinada com cerca de 1,2 milhões de imagens entre 1.000 categorias. Vale destacar que a taxa de erro da AlexNet ao se considerar as 5 classes mais prováveis foi de 15,3%, o que superou com grande margem o segundo colocado que obteve 26,2% com métodos de aprendizagem tradicionais (Krizhevsky; Sutskever; Hinton, 2012).

Segundo Zhang et al. (2021), o sucesso da AlexNet está em dois fatores chaves que permitem o treinamento de modelos cada vez maiores e mais profundos. Primeiro, o surgimento de grandes conjuntos de dados como o ImageNet que possui milhões de imagens rotuladas. Segundo, o desenvolvimento de *hardware* adequado para treinar redes neurais de forma eficiente como as *Graphical Processing Units* (GPUs). Além disso, a AlexNet introduz diversas novidades em relação a LeNet-5, como o emprego da função de ativação ReLU, a técnica de *dropout,* o aumento de dados e o treinamento com múltiplas GPUs (ELGENDY, 2020).

A partir disso, a comunidade de visão computacional passou a adotar amplamente as CNNs, além de buscar por modelos cada vez mais profundos capazes de aprender funções mais complexas. Nesse contexto, a arquitetura ResNet proposta por He *et al.* (2015) se destaca ao introduzir os blocos residuais, o que permite o treinamento estável de modelos com 152 camadas, por exemplo, superando os daquele momento como o VGG (Simonyan; Zisserman, 2015) que continha 16 ou 19 camadas apenas.

Outras arquiteturas surgiram como propostas de melhoria da ResNet ao longo do tempo. Por exemplo, Xie *et al.* (2017) propuseram a ResNeXt que, além dos blocos residuais, emprega grupos de convolução aplicados individualmente sobre

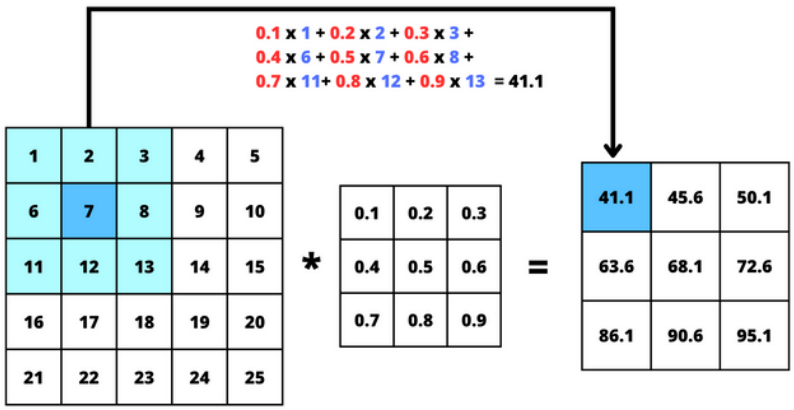
canais específicos da entrada, levando à redução do erro cometido em conjuntos de *benchmark* como o ImageNet quando comparado com a ResNet original.

Porém, a arquitetura geral das CNNs ainda se mantêm semelhante à arquitetura vista em propostas anteriores, como a LeNet-5, levantadas por LeCun e seus colaboradores (BENGIO; LECUN; HINTON, 2021). Dessa forma, elas são compostas por três tipos principais de camadas: camadas convolucionais que são o principal tipo abordado nesta seção, camadas de *pooling* e camadas totalmente conectadas.

Camadas convolucionais são constituídas por múltiplos filtros e são os principais blocos de construção de uma CNN. Tais filtros são responsáveis por rastrear e extrair características importantes a partir de uma entrada por meio da operação de convolução, o que resulta nos chamados mapas de característica (Elgendy, 2020). A partir disso, os mapas gerados servem com entrada para as camadas posteriores.

De modo a ilustrar como esse processo ocorre, a Figura 4 apresenta uma matriz como entrada de uma camada convolucional. Inicialmente, o filtro é posicionado sobre a região superior esquerda da matriz. Então, os elementos presentes nela são multiplicados pelos parâmetros correspondentes do filtro e os valores obtidos são somados. Em seguida, o resultado encontrado é mapeado para a respectiva posição do mapa de característica. O processo se repete conforme o filtro se desloca horizontalmente e verticalmente, sendo que o tamanho do deslocamento é denominado de *stride* (Howard; Gugger, 2020).

Figura 4 — Ilustração da operação de convolução aplicada sobre uma matriz de entrada.



Fonte: De autoria própria baseado em Howard e Gugger (2020).

Em vez de uma matriz, como apresentada na Figura 4, a entrada de uma

camada convolucional geralmente é um tensor tridimensional composto por múltiplos canais que são associados individualmente com um mapa de característica extraído na camada anterior. Isso exige que os filtros também sejam tensores tridimensionais, sendo que seus canais se referem às matrizes de convolução. Nesse sentido, segundo Zhang *et al.* (2021) o valor em um ponto do mapa de característica *H* gerado pelo *n*-ésimo filtro *K* de uma camada convolucional em relação ao tensor de entrada *X* pode ser encontrado pela Equação 1.

$$H_{i,j,n} = g(u_n + \sum_{a=-k}^{k} \sum_{b=-k}^{k} \sum_{c} K_{a,b,c,n} X_{i+a,j+b,c}) \quad (1)$$

Onde:

- *i* e *j* são respectivamente as posições horizontais e verticais do tensor de entrada *X* em que o centro do filtro se posiciona;
- *a e b* são respectivamente as posições horizontais e verticais relativas dentro do filtro *K* com centro em *a=0* e *b=0*;
- *k* representa o limite do intervalo de posições relativas a serem percorridas;
- *c* é o índice dos canais do tensor de entrada X;
- *n* é o índice do filtro da camada convolucional;
- *u* é um parâmetro aprendível geral do filtro denominado de *bias;*
- *g* é uma função de ativação.

Funções de ativação são importantes dentro de um modelo de aprendizado profundo pois elas introduzem o conceito de não-linearidade, o que é essencial para aproximar funções arbitrárias (Russell; Norvig, 2021). Nesse contexto, funções como sigmoide e tangente hiperbólica eram frequentemente empregadas nas camadas escondidas de uma rede neural até a introdução da *rectified linear unit* (ReLU), definida na Equação 2, que se popularizou ao facilitar a otimização dos modelos (Goodfellow; Bengio; Courville, 2016).

$$g(z) = max(0, z) \quad (2)$$

Onde:

- *z* é o argumento da função de ativação. No caso da camada convolucional, *z* é o valor retornado que irá compor alguma posição do mapa de característica conforme colocado na Equação 1;

- *max* é a função que retorna o maior valor entre aqueles repassados como argumento. No caso da ReLU, os argumentos são 0 e *z*.

Uma vez que os mapas de característica tenham sido gerados, é comum que eles passem pela camada de *pooling*, ou *subsampling*, a fim de terem sua dimensão reduzida. Para tanto, uma janela fica responsável por deslizar sobre cada mapa e aplicar algum resumo estatístico em relação aos valores encontrados nela (Elgendy, 2020). Nesse contexto, a operação de resumo mais empregada atualmente é feita por camadas de *max pooling* que consistem em selecionar apenas os valores máximos dentro da janela (Zhang *et al.*, 2021).

Por fim, os últimos mapas de característica gerados são entregues a camadas totalmente conectadas na forma de um vetor para que a classificação ocorra. Tais camadas são o tipo predominante da arquitetura *multilayer perceptron*, sendo compostas por diversas unidades de processamento, denominadas muitas vezes de neurônios artificiais, cujas saídas são entregues a todas as unidades da próxima camada (Goodfellow; Bengio; Courville, 2016). Dessa forma, as saídas dessas unidades podem ser definidas pela Equação 3.

$$h = g(w^{\intercal}x + b) \tag{3}$$

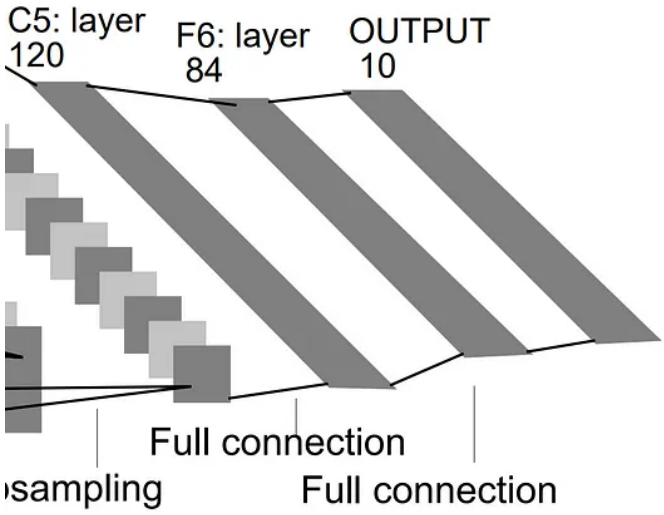
Onde:

- *x* é o vetor de entrada;
- *w* é o vetor de pesos sinápticos do neurônio;
- *b* é o termo denominado de *bias;*
- *g* é a função de ativação empregada.

No caso da tarefa de classificação, a última camada de uma CNN geralmente é composto por um número de neurônios equivalente ao total de classes, cada um associado com uma respectiva categoria. Muitas vezes, a saída deles é repassada para alguma função, como a *softmax*, responsável por mapeá-las para probabilidades, sendo a classe mais provável selecionada como o resultado da classificação (Howard; Gugger, 2020).

Por exemplo, a última camada da LeNet-5, apresentada na Figura 5, é constituída por 10 neurônios. Uma vez que seu objetivo é o de classificar imagens de dígitos manuscritos que variam de 0 a 9, cada neurônio está associado com um dígito. Apesar de não ter sido empregado no trabalho original, é possível repassar os valores de saída de cada neurônio para a função *softmax* de modo que eles sejam mapeados para probabilidades. Assim, é possível definir o dígito mais provável ao qual a imagem de entrada se refere.

Figura 5 — Última camada da LeNet-5 referente aos 10 dígitos.



Fonte: Lecun *et al.* (1998).

2.3 TREINAMENTO

Segundo Howard e Gugger (2020), o treinamento consiste no ajuste automático dos parâmetros de um modelo, inicializados aleatoriamente, através de algum algoritmo de otimização. Para tanto, o modelo realiza sucessivas predições ao ser alimentado com os dados fornecidos que são avaliadas por uma função de perda. Nesse sentido, um algoritmo de aprendizado profundo é constituído basicamente por quatro elementos: um modelo, um conjunto de dados, uma função de perda e um processo de otimização (Goodfellow; Bengio; Courville, 2016).

No aprendizado profundo, o termo modelo refere-se ao maquinário computacional por meio do qual um dado de entrada é transformado por sucessivas camadas até ser mapeado para uma predição (Zhang *et al.*, 2021). Por exemplo, em uma CNN, a imagem de entrada é transformada em sucessivos mapas de característica até ser classificada. Tais transformações são condicionadas pelos parâmetros presentes nas camadas e que são aprendidos com base no conjunto de dados visto no treinamento.

A especificação do conjunto de dados pode variar de acordo com o tipo de aprendizado adotado. No aprendizado supervisionado, o conjunto de dados é formado por exemplos associados com suas respectivas anotações. Ao final do treinamento, espera-se que o modelo aprenda a anotar novos dados com base nos pares de exemplos e anotações vistos (Goodfellow; Bengio; Courville, 2016).

O tipo de aprendizado utilizado para o treinamento de modelos de detecção

de objetos é supervisionado, o que exige que as imagens do conjunto de dados sejam anotadas. Assim, anotar é o ato de especificar as coordenadas das caixas delimitadoras que podem ser obtidas, por exemplo, a partir do canto superior esquerdo até o ponto inferior direito da caixa. Além disso, é necessário determinar também as classes dos objetos, isto é, a categoria a qual eles pertencem. A partir disso, o modelo aprende a localizar e a classificar novos objetos com base nas imagens e anotações fornecidas ao longo do treinamento.

Apesar de ser apenas um entre outros paradigmas, o aprendizado supervisionado corresponde a grande parte do sucesso alcançado pela aprendizagem de máquina em aplicações na indústria. Por exemplo, ele pode ser empregado em tarefas de classificação, na qual a entrada deve ser associada com sua categoria, ou na regressão, onde ela deve ser mapeada para um valor numérico (ZHANG et al., 2021).

Nesse contexto, a detecção de objetos pode ser vista como uma tarefa que envolve tanto a regressão, ao predizer coordenadas de caixas delimitadores que localizam os objetos, quanto a classificação, ao associar o objeto localizado com sua respectiva categoria.

Outro componente importante do treinamento é a função de perda que quantifica a diferença entre o valor previsto para um dado de entrada e o seu respectivo rótulo (Howard; Gugger, 2020). Por exemplo, o modelo pode inferir que a probabilidade de um pássaro da espécie sanhaço-do-coqueiro ser dessa espécie é de 20%, enquanto o ideal é que seja o mais próximo de 100%. Assim, o objetivo do treinamento é o de encontrar os parâmetros ideais que minimizem essa função, isto é, que reduzam o erro cometido, o que pode ser feito por algum método de otimização.

Um dos principais algoritmos nesse sentido é o gradiente estocástico descendente. Nesse método, o conjunto de treinamento é dividido em grupos menores denominados de *mini-batches*, os quais são empregados individualmente para calcular a média dos gradientes da função de perda em relação aos parâmetros do modelo (Goodfellow; Bengio; Courville, 2016). Dessa forma, tais parâmetros podem ser atualizados cada vez que a média dos gradientes for computada conforme colocado na Equação 4 e na Equação 5.

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha g \tag{4}$$

$$g = \frac{1}{m} \nabla_{\boldsymbol{\theta}} \sum_{i=1}^{m} L(x^{(i)}, y^{(i)}, \boldsymbol{\theta}) \tag{5}$$

Onde:

- **θ** é o vetor que armazena os parâmetros do modelo;
- *α* é a taxa de aprendizagem;
- *g* é a média dos gradientes*;*
- *m* é a quantidade de exemplos do *mini-batch;*
- *L* é a função de perda que recebe como argumentos $x^{(i)}$, $y^{(i)}$ e **θ.**
- $x^{(i)}$ é o *i*-ésimo exemplo do *mini-batch;*
- $y^{(i)}$ é o rótulo do *i*-ésimo exemplo do *mini-batch;*

Enquanto parâmetros podem ser aprendidos, hiperparâmetros são escolhas que refletem algum aspecto do treinamento, como o tamanho do *batch* ou o número de *epochs,* que é o número de vezes que o conjunto de treinamento será percorrido em sua totalidade (Howard; Gugger, 2020). Na Equação 4, o termo *α* refere-se à taxa de aprendizagem, sendo um dos hiperparâmetros mais influentes do treinamento, já que uma taxa muito baixa pode levar à lentidão na convergência dos parâmetros, já uma taxa muito alta pode levar a sua divergência (Elgendy, 2020).

Assim, o processo de treinamento pode ser resumido nos seguintes passos: inicializar o modelo com parâmetros aleatórios, realizar predições com base nesses parâmetros, avaliar as predições com uma função de perda, calcular os gradientes, atualizar os parâmetros a partir dos gradientes, repetir o processo após a inicialização até que se tenha atingido um critério de parada.

Vale destacar que raramente uma CNN é treinada do princípio, uma vez que isso exige muitos dados, além de recursos computacionais capazes de conduzir o treinamento nessas condições (ELGENDY, 2020). Dessa forma, geralmente é empregada a técnica de transferência de aprendizagem na qual um modelo é pré-treinado em um conjunto de dados maior como o ImageNet. Em seguida, ele pode ser treinado para a tarefa desejada, porém com um conjunto de dados menor e com menos recursos computacionais (HOWARD; GUGGER, 2020).

Vale ressaltar também que é uma prática comum expandir o tamanho do conjunto de treinamento ao empregar técnicas de aumento de dados. Nelas, transformações aleatórias são aplicadas aos dados originais a fim de gerar variações similares desses. No contexto de imagens, essas transformações podem se dar, por exemplo, pelo recorte aleatório de uma área menor da imagem seguido pelo seu redimensionamento, além de mudanças nos valores de brilho, contraste, saturação e tonalidade das cores (ZHANG et al., 2021).

2.4 MÉTRICAS DE PERFORMANCE

A performance de um modelo treinado não pode ser avaliada diretamente no conjunto de treinamento, uma vez que ele pode ter passado pelo fenômeno de *overfitting* (sobreajuste). Isso ocorre quando o modelo é incapaz de fazer previsões acuradas para novos pontos de dados por ter se ajustado excessivamente ao conjunto de treinamento, comprometendo assim sua generalização (Howard; Gugger, 2020). Dessa forma, é separada uma parte dos dados em um conjunto denominado de teste para que predições sejam realizadas a partir deles e avaliadas por meio de alguma métrica.

Nesse contexto, o primeiro passo na avaliação individual das predições de um detector de objetos é reconhecer se a caixa delimitadora estimada se sobrepõe à caixa anotada de algum objeto da mesma classe prevista. Para tanto, é necessário quantificar o nível de sobreposição entre a caixa prevista com a caixa anotada para o objeto em questão, o que pode ser feito por meio da métrica *Intersection over Union* (IoU), definida na Equação 6 (Elgendy, 2020).

$$IoU = \frac{\text{area}\left(B_p \cap B_{gt}\right)}{\text{area}\left(B_p \cup B_{gt}\right)} \tag{6}$$

Onde:

- *area* é a função que calcula a área da região delimitada pelo argumento.
- B_p é a caixa delimitadora prevista pelo modelo;
- B_{gt} é a caixa delimitadora anotada para o objeto;

Inicialmente, devem ser calculadas as áreas de intersecção e de união entre a caixa delimitadora prevista e anotada. Então, a IoU é dada pela razão entre a área de intersecção e a área de união. Vale destacar que ela é computada apenas se a classe prevista pela detecção for equivalente a do objeto.

De modo a avaliar se uma detecção está correta, é necessário definir um valor de limiar para a IoU. Nesse contexto, caso a IoU calculada seja maior ou igual a esse limiar, então a detecção é considerada correta sendo chamada de verdadeiro positivo (VP), caso contrário, está incorreta recebendo o nome de falso positivo (FP) (Padilla; Netto; Silva, 2020). Vale destacar que FPs podem ocorrer também quando a detecção se referir a um objeto inexistente, por exemplo, quando a caixa delimitadora se sobrepor ao plano de fundo da imagem.

Por exemplo, caso a caixa delimitadora prevista pelo modelo se sobreponha a

caixa anotada de um objeto da mesma classe com IoU em 60%, se o limiar de IoU for definido em 50%, então a detecção estará correta, isto é, ela será um VP. Por outro lado, se o limiar for de 75%, então a detecção será considerada incorreta, ou seja, será um FP. Isso ocorre pois, no primeiro caso, a IoU de 60% é maior do que o limiar de 50%, já no segundo, ela é menor que o limiar de 75%.

Durante a avaliação, há ainda a existência do falso negativo (FN) que ocorre quando um objeto anotado deixa de ser detectado, isto é, quando sobre ele não houver qualquer VP. Vale ressaltar que verdadeiros negativos não ocorrem na detecção de objetos, pois isso significaria dizer que o plano de fundo deixou de ser detectado, o que pode ocorrer de infinitas maneiras (PADILLA et al., 2021).

Tabela 1 — Resumo das categorias de avaliação de uma detecção.

Categoria	Descrição
VP	A caixa prevista se sobrepõe a caixa anotada para um objeto da mesma classe com valor de IoU superior ao limiar definido.
FP	A caixa prevista se sobrepõe a caixa anotada para um objeto mesma classe, porém com valor de IoU inferior ao limiar definido. Além disso, pode se dar pela detecção de um objeto inexistente.
FN	O objeto anotado deixa de ser detectado corretamente, já que não há qualquer VP associado com ele.

Fonte: De autoria própria.

A partir de tais conceitos, a métrica precisão pode ser definida formalmente a partir da equação Equação 7. Ela visa quantificar a proporção de acertos que o modelo teve em relação a todas suas detecções. Para tanto, calcula a razão entre o total de VPs e a quantidade de detecções feitas, a qual pode ser encontrada ao somar o total VPs com o total de FPs.

Já a revocação pode ser expressa matematicamente por meio da Equação 8. Seu objetivo é o de quantificar a proporção de objetos detectados corretamente pelo modelo em relação a todos os objetos que deveriam ter sido. Para tanto, calcula a razão entre o total de VPs e a quantidade de objetos anotados, a qual pode ser encontrada ao somar o total de VPs com o total de FNs.

$$P = \frac{\sum VP}{\sum VP + \sum FP} \tag{7}$$

$$R = \frac{\sum VP}{\sum VP + \sum FN} \tag{8}$$

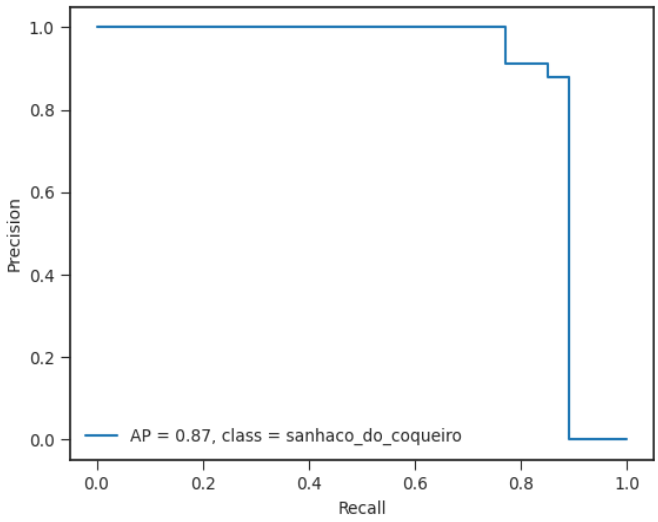
Onde:

- ∑ VP é o total de detecções feitas pelo modelo que são consideradas verdadeiros positivos, ou seja, a quantidade de predições corretas.
- ∑ FP é o total de detecções feitas pelo modelo que são consideradas falsos positivos, ou seja, a quantidade de predições incorretas.
- ∑ FN é o total de detecções feitas pelo modelo que são consideradas falsos negativos, ou seja, a quantidade de objetos não detectados.

2.4.1 **Average Precision**

Precisão e revocação podem ser condicionadas também por meio do nível de confiança ao considerar como VPs apenas as detecções acima de determinados valores. Nesse contexto, é possível construir uma curva com os valores de precisão por revocação para diferentes níveis de confiança (Elgendy, 2020). Segundo Padilla *et al.* (2021), um bom detector é aquele que consegue manter tanto precisão quanto revocação altas, mesmo se o nível de confiança decrescer, o que pode ser indicado pela área sob a curva mencionada, que é exemplificada pela Figura 6.

Figura 6 — Curva de precisão por revocação.



Fonte: De autoria própria.

Vale observar que o limiar de IoU não deve ser confundindo com o nível de confiança. O limiar de IoU visa definir um valor mínimo de sobreposição que deve existir entre uma caixa prevista e uma caixa anotada. Já o nível de confiança refere-se a certeza que o modelo tem sobre sua detecção. Nesse caso, ambos podem ser utilizadas como critério para limitar os VPs ao se considerar apenas detecções acima de uma certo limiar de IoU e de um determinado nível de confiança.

Assim, a métrica *Average Precision* (AP) pode ser definida como a área estimada sob a curva de precisão por revocação de uma determinada classe, sendo a principal métrica de performance utilizada nas competições da área de detecção de objetos (Padilla; Netto; Silva, 2020). No entanto, seu cálculo pode variar de acordo com a referência adotada, por exemplo, a avaliação de detectores no conjunto MS COCO (Lin *et al.*, 2014) pode empregar três variações denominadas de AP, $AP^{IoU=.50}$ e $AP^{IoU=.75}$.

Na primeira forma (AP), são construídas 10 curvas de precisão por revocação para cada limiar de IoU no intervalo de 50% até 95% espaçados igualmente por 5%, sendo que a métrica é calculada como a média das áreas estimadas para cada uma. Já na segunda ($AP^{IoU=.50}$) e na terceira ($AP^{IoU=.75}$), é empregado um único limiar de IoU na construção das curvas nos valores de 50% e de 75% respectivamente.

Vale ressaltar que como a maioria dos conjuntos possuem múltiplas classes, então a métrica final de desempenho do modelo, denominada de *mean Average Precision* (mAP), é o resultado da média aritmética entre os valores de AP de cada classe.

Neste trabalho, será empregada a nomenclatura **mAP**, **mAP@.50** e **mAP@.75** para se referir ao cálculo da métrica final de desempenho com base nas respectivas formas AP, $AP^{IoU=.50}$ e $AP^{IoU=.75}$ citadas anteriormente.

2.4.2 **Matriz de Confusão**

A matriz de confusão é um recurso útil para visualizar a quantidade de detecções nas quais o modelo classificou erroneamente um objeto como sendo de outra categoria. Em uma matriz de confusão, linhas e colunas são associadas com as classes do conjunto de dados, sendo que as classes das linhas representam o valor verdadeiro da anotação do objeto, enquanto as colunas representam as classes previstas (Howard; Gugger, 2020).

Nesse sentido, os elementos de uma matriz de confusão guardam o total de ocorrências em que um objeto da classe indicada pela linha foi detectado como pertencente a classe associada com a coluna. A partir disso, espera-se que os

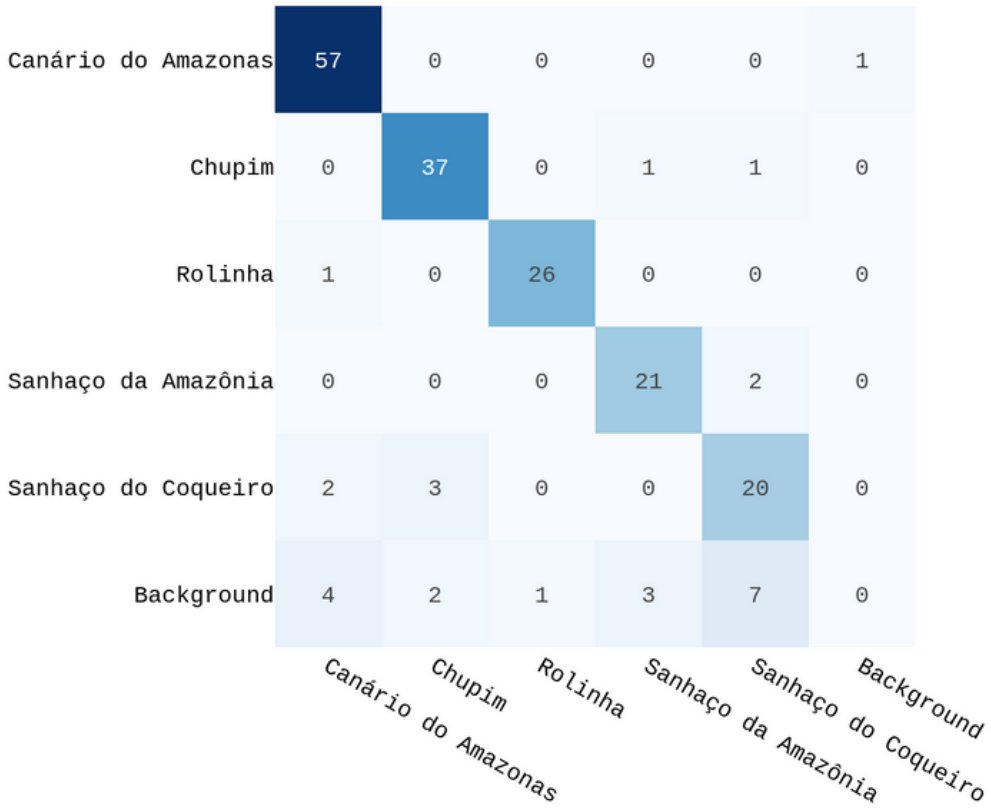
maiores números se encontrem na diagonal da matriz, onde a classe prevista para o objeto é equivalente a de sua anotação.

De modo a construir a matriz de confusão no contexto de detecção de objetos, é necessário permitir que a IoU seja computada mesmo que a classe prevista pela detecção seja diferente da do objeto em questão. Nesse contexto, múltiplas detecções de diferentes classes podem ocorrer sobre um mesmo objeto, sendo que esse tipo de empate pode ser contornado ao se considerar que o objeto foi detectado pela detecção com maior nível de confiança (Padilla; Netto; Silva, 2020).

Porém, por meio desse critério, mesmo que haja alguma detecção que se enquadre como um VP, se o seu nível de confiança for inferior ao de outra detecção sobre o mesmo objeto, então ela passa ser interpretada como um FP. Assim, é considerado que ocorreu a detecção do plano de fundo da imagem, já que o modelo se encontra mais confiante de que o pássaro em questão seja de outra espécie.

Na detecção de objetos, é comum incluir também uma linha e uma coluna adicionais na matriz de confusão. A linha, denominada de *background*, indica as detecções de plano de fundo da imagem. Já a coluna, denominada também de *background*, indica objetos sobre os quais não ocorreram qualquer detecção. Na Figura 7, é possível visualizar um exemplo de matriz de confusão. Além disso, esses conceitos são ilustrados no Apêndice A.

Figura 7 — Exemplo de matriz de confusão.



Fonte: De autoria própria.

2.5 TRABALHOS RELACIONADOS

Na literatura científica, é possível encontrar trabalhos que focam exclusivamente na classificação de espécies de pássaros. Por exemplo, Srijan, Samriddhi e Gupta (2021) investigaram 6 arquiteturas de redes neurais convolucionais na classificação de 260 espécies, além de desenvolverem uma aplicação mobile empregando o modelo EfficientNet-Lite0. Já Huang e Basanta (2021) estudaram o emprego do modelo Inception-ResNet v2 para classificar 29 espécies endêmicas de Taiwan, além de desenvolverem também uma aplicação mobile.

Em relação aos estudos nacionais, Pinheiro e Soares (2021) levantaram um

conjunto de imagens de 64 espécies de pássaros avistados em Linhares, município do Espírito Santo. Em seguida, treinaram o modelo ResNet101 v2 para a classificação dessas espécies. No final, o modelo alcançou acurácia de 70,12% para os dados de validação e apresentou dificuldade em cenas com a presença de múltiplos pássaros.

No que diz respeito à detecção de objetos, foram encontrados trabalhos relacionados ao contexto de prevenção de acidentes ou falhas. Por exemplo, Alqaysi *et al.* (2021) estudaram 3 configurações do modelo YOLOv4 com o intuito de detectar pássaros voando próximos de fazendas eólicas. Já Shi *et al.* (2021) empregaram um modelo YOLOv5 adaptado a fim de detectar pássaros voando ao redor de aeroportos. Em ambos os casos, as espécies dos pássaros não foram levadas em consideração.

Entre os trabalhos que focam na detecção das espécies, encontram-se o de Qiu *et al.* (2021) que utilizaram o modelo YOLOv4-tiny de modo a detectar pássaros de 20 espécies relacionadas a falhas em linhas de transmissão energéticas. Além disso, há trabalhos dentro do contexto de cenas naturais, como o de Mao *et al.* (2021) que treinaram o modelo Faster R-CNN com a técnica de randomização de domínio para detectar 2 espécies frequentemente encontradas em um parque ecológico. Já Xiang *et al.* (2022) propuseram uma versão aperfeiçoada do modelo Faster R-CNN para detectar 10 espécies de pássaros em pequenas escalas localizados também em um parque ecológico.

Mirugwe, Nyirenda e Dufourq (2022) realizaram o estudo mais próximo da proposta levantada neste trabalho ao empregarem webcams na tarefa de detectar pássaros que visitam comedouros, porém sem levar em consideração suas espécies. Para tanto, coletaram imagens a partir de transmissões ao vivo feitas por webcams pelo instituto Cornell Lab of Ornithology apresentando pássaros se alimentando em seis localizações distintas ao redor do mundo.

A partir disso, investigaram a aplicação dos detectores Faster R-CNN e SSD com diferentes redes de backbone como MobileNet, ResNet50, ResNet101, ResNet152 e Inception-ResNet v2. Nesse caso, o detector Faster R-CNN com o backbone ResNet152 obteve maior mAP de 92,3%, porém acompanhado com o maior tempo de inferência de 283ms por imagem. Já o detector SSD com o backbone MobileNet obteve o melhor tempo de inferência de 110ms por imagem, porém acompanhado da pior mAP de 67,5%.

A proposta deste trabalho se diferencia ao focar na detecção das espécies dos pássaros que frequentam um comedouro exclusivamente residencial localizado dentro de um contexto amazônico. Até onde se sabe, o presente trabalho é o primeiro a levantar essa proposta, bem como de produzir um conjunto de imagens

dessas espécies anotado para a tarefa de detecção de objetos. Dessa forma, a contribuição deste trabalho pode ser comparada com a dos demais trabalhos pela Tabela 2.

Tabela 2 — Comparação deste trabalho com os demais trabalhos relacionados.

	Detecta múltiplos pássaros?	Reconhece as espécies?	Conjunto formado por imagens originais?	Disponibiliza publicamente o conjunto levantado?	mAP@.50 é superior a 90%?	Emprega webcams?	Estudo nacional?	Espécies inseridas no contexto amazônico?
MyBirds	✔	✔	✔	✔	✔	✔	✔	✔
Srijan, Samriddhi e Gupta (2021)	✘	✔	✘	✘	**NA**	✘	✘	✘
Huang e Basanta (2021)	✘	✔	✘	✘	**NA**	✘	✘	✘
Pinheiro e Soares (2021)	✘	✔	**P**	✘	**NA**	✘	✔	✘
Alqaysi et al. (2021)	✔	✘	✔	✔	✔	✘	✘	✘
Shi et al. (2021)	✔	✘	**P**	✘	✘	✘	✘	✘
Qui et al. (2022)	✔	✔	**P**	✘	✔	✘	✘	✘
Mao et al. (2021)	✔	✔	✔	✘	✘	✘	✘	✘
Xiang et al. (2022)	✔	✔	✔	✘	✘	✘	✘	✘
Mirugwe, Nyirenda e Dufourq (2022)	✔	✘	✘	✘	✔	✔	✘	✘

✔: Sim ✘: não **P**: parcialmente **NA**: não se aplica

Fonte: De autoria própria.

3 **CONSTRUÇÃO DO MODELO**

Neste capítulo, será apresentado o processo de construção do modelo Faster R-CNN que foi dividido em duas fases. Assim, na seção 3.1, será abordada a primeira fase na qual uma porção menor dos dados foi empregada para definir uma *baseline*, isto é, um modelo que serve como base de análise, além da configuração de treinamento. Já na seção 3.2, será discutida a segunda fase, na qual um único modelo foi treinado com a totalidade dos dados, além de ter sido comparado com a *baseline*.

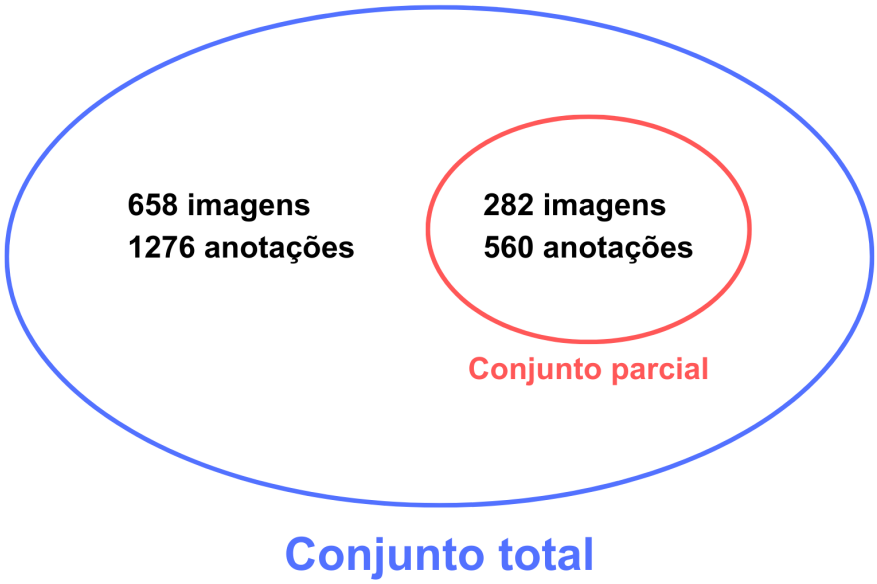
3.1 FASE PRELIMINAR

De modo a construir o modelo Faster R-CNN, foi necessário levantar um conjunto de dados[3] de imagens anotadas. O conjunto conta com 940 imagens e 1.836 anotações distribuídas entre 5 classes. Cada classe se refere a uma espécie de pássaro identificada. A construção desse conjunto, chamado de total, será detalhada no capítulo 4.

Antes de realizar o treinamento com o conjunto total, 30% dos dados foram alocadas aleatoriamente em um novo conjunto, denominado de parcial. Nesse sentido, o conjunto parcial conta com 282 imagens e 560 anotações distribuídas também entre 5 espécies. A relação entre os conjuntos total e parcial é descrita pelo diagrama de Venn da Figura 8.

3 O conjunto de dados pode ser acessado em: https://github.com/Lucas-Zampar/detector_de_passaros_amazonicos/tree/main/dataset

Figura 8 — Diagrama relacionando o conjunto total e parcial.



Fonte: De autoria própria.

Durante a fase preliminar, o conjunto de dados parcial foi empregado para o treinamento de modelos Faster R-CNN com diferentes configurações a fim de:

- Determinar a configuração de treinamento do modelo em construção.
- Definir uma *baseline* para comparação.

Com o propósito de alcançar esses objetivos, a fase preliminar foi subdividida ainda em duas etapas. Na primeira etapa, redes de *backbone* dos tipos ResNet e ResNeXt foram experimentadas com a finalidade de selecionar aquela com o melhor desempenho. Já na segunda etapa, instâncias do modelo com o *backbone* selecionado foram treinadas com diferentes valores de hiperparâmetro de modo a melhorar o desempenho atingido.

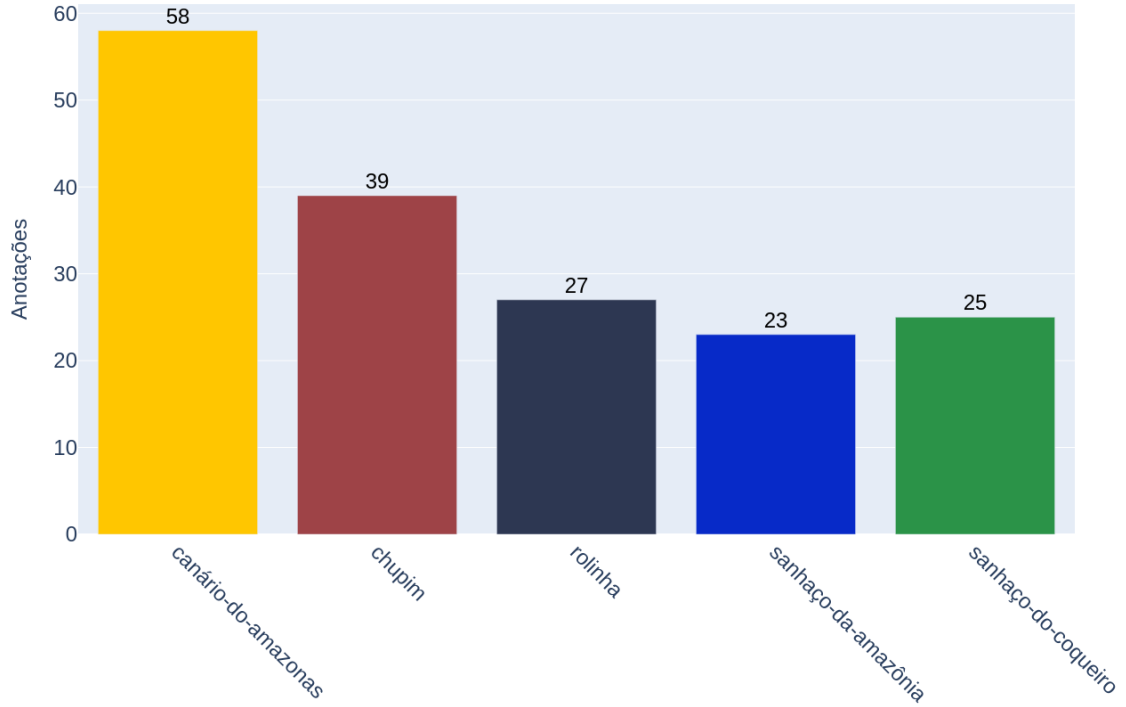
Entre os hiperparâmetros ajustados, encontram-se o número de *epochs,* a ocorrência de embaralhamento em cada *epoch*, o tamanho do *batch*, a taxa de aprendizagem, o tamanho da imagem e o tamanho de redimensionamento no *presizing*. Vale destacar que, na técnica de *presizing*, as transformações na imagem são aplicadas em sequência na GPU, sendo necessária uma única interpolação no final. Para tanto, a imagem deve ser redimensionada inicialmente para um tamanho maior do que aquele utilizado no treinamento (Howard; Gugger, 2020).

Durante as duas etapas, 70% do conjunto parcial foi reservado para o treinamento dos modelos pelo *framework* IceVision, enquanto os 30% restantes

foram destinados para teste pelo *framework* FiftyOne. Nesse contexto, 197 images foram empregadas para treinar os modelos da fase preliminar, o que representa aproximadamente 20,96% do conjunto de dados total.

Além disso, foram utilizadas 85 imagens para teste, o que reflete cerca de 9,04% da totalidade dos dados. Em relação às anotações das imagens de teste, foram levantadas 58 anotações para a espécie canário-do-amazonas, 39 para chupim, 27 para rolinha, 23 para sanhaço-da-amazônia e 25 para sanhaço-do-coqueiro conforme apresentado pelo Gráfico 1.

Gráfico 1 — Proporção de anotação por espécie da parcela de teste do conjunto de dados parcial.



Fonte: De autoria própria.

3.1.1 **Seleção do *Backbone***

De modo a realizar o treinamento dos modelos, é necessário definir o valor inicial dos hiperparâmetros. Nesse contexto, a configuração de treinamento dos modelos nesta etapa pode ser visualizada na Tabela 3.

Tabela 3 — Configuração de treinamento dos modelos na etapa de seleção do backbone.

Hiperparâmetro	Valor
Total de *epochs*	20
Embaralhamento do conjunto de treinamento	Não
Tamanho do *batch*	1
Taxa de aprendizagem	10-4
Tamanho da imagem	512x512
Emprego da técnica de *presizing*	Não

Fonte: De autoria própria.

A Tabela 4 exibe o desempenho final alcançado por cada modelo treinado com essa configuração. A partir dela, é possível perceber que o maior desempenho foi alcançado pelo modelo 5 ao registrar um percentual de 71,33% para a métrica mAP. Além disso, ele também alcançou os maiores valores para as métricas de referência mAP@.50 e mAP@.75. Dessa forma, o *backbone* ResNeXt101 32x4d FPN 1x do modelo 5 será utilizado também no modelo em construção.

Tabela 4 — Desempenho alcançado pelos modelos com cada rede de backbone.

Nº do modelo	*Backbone*	mAP	mAP@.50	mAP@.75
1	ResNet50 FPN 1x	0,6287	0,8711	0,7887
2	ResNet50 FPN 2x	0,6234	0,8718	0,7628
3	ResNet101 FPN 1x	0,6489	0,8723	0,8209
4	ResNet101 FPN 2x	0,6518	0,8767	0,8381
5	ResNeXt101 32x4d FPN 1x	0,7133	0,9417	0,8865
6	ResNeXt101 32x4d FPN 2x	0,6739	0,8870	0,8433
7	ResNeXt101 64x4d FPN 1x	0,6960	0,9280	0,8648
8	ResNeXt101 64x4d FPN 2x	0,6779	0,8874	0,8393

Fonte: De autoria própria.

Vale destacar que empregar redes do tipo ResNeXt levou a melhores resultados, uma vez que os modelos de 5 até 8 obtiveram desempenhos superiores do que os modelos de 1 até 4 que utilizaram apenas redes do tipo ResNet. Talvez, isso seja explicado pelo fato da ResNeXt ser uma proposta de melhoria das primeiras redes ResNet.

A fim de verificar como o modelo 5 se comporta em relação a cada espécie de

pássaro, a Tabela 5 exibe os valores das métricas precisão e revocação obtidas para cada classe ao se considerar 50% como o limiar de *Intersection over Union* (IoU). Além disso, também são apresentados os totais de VPs, de FPs e de FNs como referência.

Tabela 5 — Métricas finais de precisão e de revocação do modelo 5 para cada espécie ao se considerar 50% como o limiar de IoU.

Espécie	Precisão	Revocação	VP	FP	FN
canário-do-amazonas	0,8750	0,9655	56	8	2
chupim	0,9500	0,9744	38	2	1
rolinha	0,8966	0,9630	26	3	1
sanhaço-da-amazônia	0,9200	1,0000	23	2	0
sanhaço-do-coqueiro	0,8800	0,8800	22	3	3
Média	0,9043	0,9566			

Fonte: De autoria própria.

Com a análise dos resultados, percebeu-se que a menor precisão ocorre para a espécie canário-do-amazonas, já que apenas 87,5% das predições feitas para essa espécie estavam corretas. Já a menor revocação ocorre para a espécie sanhaço-do-coqueiro, uma vez que apenas 88% dos objetos anotados dessa espécie foram detectados corretamente. Em geral, a precisão média do modelo 5 foi de 90,43%, enquanto a revocação média ficou em 95,66%.

Ao analisar esses dados por meio da matriz de confusão, na Figura 9, é possível verificar entre quais espécies o modelo 5 apresenta maior dificuldade de distinção. No Apêndice B, são apresentadas as detecções destacadas pela matriz. Vale ressaltar que para alguns objetos, ocorreram detecções que os classificaram corretamente, porém com um valor inferior em termos de nível de confiança do que as detecções incorretas.

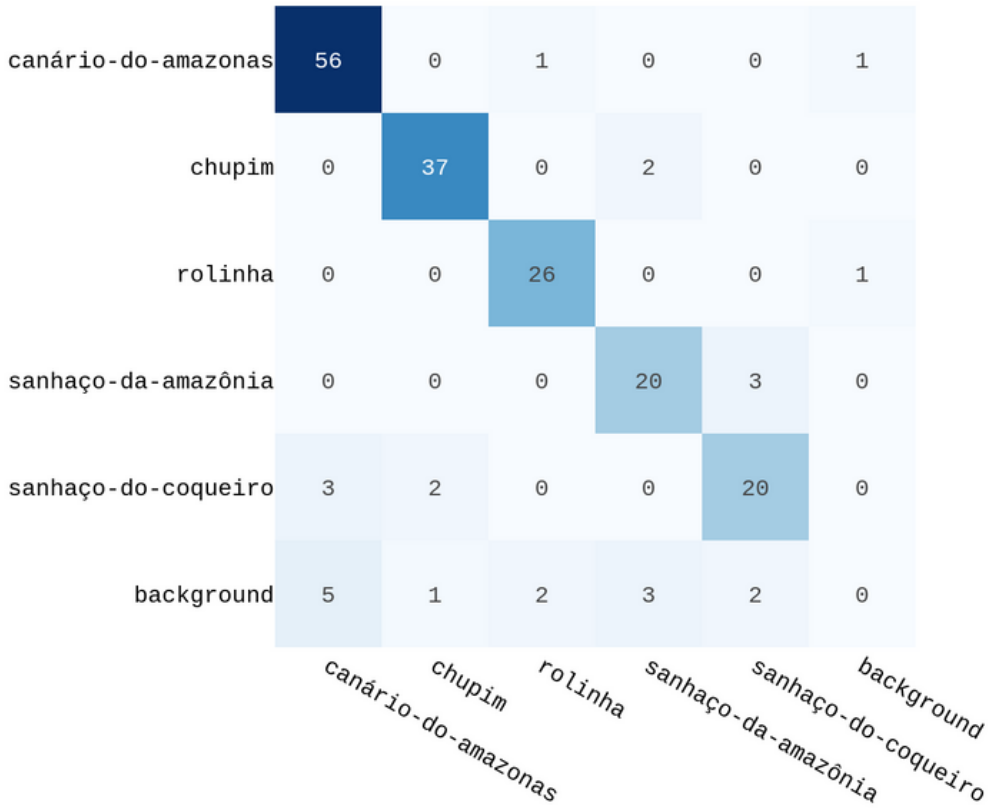
Nesse contexto, nota-se que os indivíduos da espécie sanhaço-do-coqueiro foram os mais confundidos, uma vez que 3 deles foram detectados como canário-do-amazonas e outros 2 como chupim. Ainda não existe uma explicação clara de tal ocorrência, já que as espécies são diferentes fisicamente.

Além disso, ocorreram também confusões entre as espécies de sanhaços, dado que 3 indivíduos sanhaço-da-amazônia foram detectados como sanhaço-do-coqueiro, o que pode ter ocorrido pelas semelhanças físicas que as espécies compartilham.

Por fim, 2 pássaros da espécie chupim foram detectados como

sanhaço-da-amazônia. Vale destacar que a plumagem do chupim macho pode apresentar um tom azulado de acordo com a posição e incidência luminosa, sendo que ela também é uma das principais características visuais do sanhaço-da-amazônia, o que pode ter levado à confusão.

Figura 9 — Matriz de confusão para o modelo 5 eleito na etapa de seleção do backbone.



Fonte: De autoria própria.

3.1.1.1 Resumo da subseção

Nesta etapa, foram treinados modelos Faster R-CNN enumerados de 1 a 8 com diferentes *backbones* dos tipos ResNet e ResNext. O modelo 5, que contava com a rede de *backbone* ResNeXt101 32x4d FPN 1x, foi o que mais se destacou ao alcançar mAP de 71,33%. Além disso, ele obteve precisão média de 90,43% e

revocação média de 95,66% ao se considerar 50% como o limiar de IoU. Assim, a rede ResNeXt101 32x4d FPN 1x foi selecionada como o *backbone* do modelo em desenvolvimento.

3.1.2 **Ajuste de Hiperparâmetros**

O ajuste de hiperparâmetros permite ao modelo em construção ser submetido a modificação de diferentes propriedades que normalmente levam a mudanças no resultado final. Assim, houve o treinamento de novos modelos, enumerados de 5.1 até 5.14, que representam instâncias do modelo 5, porém com hiperparâmetros modificados. Nesse contexto, foram conduzidos cinco testes consecutivos a fim de verificar se seria possível alcançar um desempenho superior ao do modelo 5 original através do ajuste de hiperparâmetros. A Tabela 6 resume os testes realizados.

Tabela 6 — Descrição dos cincos testes realizados na etapa de ajuste de hiperparâmetros.

Teste	Descrição
Primeiro teste	Variar a quantidade de *epochs* em 10 unidades, bem como o embaralhamento do conjunto de treinamento em cada uma.
Segundo teste	Experimentar tamanho de *batch* 4.
Terceiro teste	Experimentar taxas de aprendizagem 10^{-2}, 10^{-3} e 10^{-5} respectivamente.
Quarto teste	Experimentar os tamanhos de imagem 640x640, 768x768 e 896x896 respectivamente.
Quinto teste	Experimentar os tamanhos de imagem 512x512, 640x640, 768x768 e 896x896 acompanhados de *presing* com redimensionamento em 640x640, 786x786, 896x896 e 1024x1024 respectivamente.

Fonte: De autoria própria.

Por meio da Tabela 7, é possível perceber que o melhor desempenho no primeiro teste foi alcançado pela instância 5.3, que registrou o valor de 71,11% em relação a mAP. No entanto, não houve melhoria de performance em relação ao modelo 5. Dessa forma, foi descartada a adoção do embaralhamento do conjunto de treinamento, além de terem sido mantidas as 20 *epochs* originais.

Tabela 7 — Métricas finais no ajuste do número de epochs e do embaralhamento do conjunto de treinamento em cada uma.

Nº da instância	*Epochs*	Embaralhamento	mAP	mAP@.50	mAP@.75
5.1	30	Não	0,6904	0,9241	0,8274
5.2	20	Sim	0,6799	0,9162	0,8278
5.3	30	Sim	0,7111	0,9412	0,8678

Fonte: De autoria própria.

Como pode ser visto na Tabela 8, a instância 5.4 alcançou mAP de apenas 56,35% no segundo teste, o que representa o menor desempenho visto até então. Dessa forma, foi mantido o tamanho do *batch* em 1 conforme visto na configuração original de treinamento. É válido salientar que a memória da GPU foi insuficiente para experimentar tamanhos maiores que 4.

Tabela 8 — Métricas finais no ajuste do tamanho do batch.

Nº da instância	Tamanho do Batch	mAP	mAP@.50	mAP@.75
5.4	4	0,5635	0,7973	0,7058

Fonte: De autoria própria.

De acordo com a Tabela 9, observa-se que, no terceiro teste, a maior taxa de aprendizagem levou à provável divergência da instância 5.5, já a menor levou a um resultado inferior para a instância 5.7 com apenas 20 *epochs* de treinamento. Por fim, treinar a instância 5.6 com a taxa de 10^{-3} resultou no valor de 71,22% para a mAP, o que, apesar de próximo, ainda é inferior ao resultado do modelo 5. Portanto, a configuração de treinamento original que emprega a taxa de 10^{-4} foi mantida.

Tabela 9 — Métricas finais no ajuste da taxa de aprendizagem com 20 epochs de treinamento.

Nº da instância	Taxa de Aprendizagem	mAP	mAP@.50	mAP@.75
5.5	10-2	0,0000	0,0000	0,0000
5.6	10-3	0,7122	0,9377	0,8788
5.7	10-5	0,2433	0,4029	0,2485

Fonte: De autoria própria.

A partir da Tabela 9, verifica-se que a instância 5.10 foi a que mais se destacou no quarto teste ao alcançar mAP de 76,05%. O resultado registrado é superior ao do modelo 5. Assim, conclui-se que redimensionar as imagens para o tamanho de 896x896, conforme colocado na Tabela 10, prova-se válido.

Tabela 10 — Métricas finais no ajuste do tamanho da imagem.

Nº da instância	Tamanho da Imagem	mAP	mAP@.50	mAP@.75
5.8	640x640	0,7078	0,9258	0,8594
5.9	768x768	0,7164	0,9325	0,8585
5.10	896x896	0,7605	0,9434	0,8902

Fonte: De autoria própria.

Através da Tabela 11, é possível notar que a instância 5.14 teve o melhor desempenho no quinto teste ao alcançar mAP de 75,29%. O resultado também é superior ao do modelo 5. Dessa forma, conclui-se que redimensionar inicialmente as imagens para o tamanho de 1024x1024 durante o *presizing*, conforme apresentado na Tabela 11, também prova-se válido.

Tabela 11 — Métricas finais no ajuste do tamanho da imagem acompanhado de presizing.

Nº da instância	Tamanho da Imagem	Redimensionamento no *presizing*	mAP	mAP@.50	mAP@.75
5.11	512x512	640x640	0,6969	0,9275	0,8587
5.12	640x640	768x768	0,7043	0,9134	0,8667
5.13	768x768	896x896	0,7405	0,9452	0,9019
5.14	896x896	1024x1024	0,7529	0,9459	0,8851

Fonte: De autoria própria.

Apesar da instância 5.10 alcançar maior mAP do que a instância 5.14, os valores registrados ainda são próximos. Assim, a fim de selecionar o melhor ajuste de hiperparâmetros, foi analisado também como as instâncias se comportam em relação a cada espécie. Para tanto, as métricas de precisão e de revocação foram empregadas ao se considerar 50% como o valor limiar para a IoU. Além disso, foram inclusos o total de VPs, de FPs e de FNs como referência.

Nesse contexto, os resultados das instâncias 5.10 e 5.14 podem ser visualizados respectivamente na Tabela 12 e na Tabela 13. Quando comparados

com os resultados obtidos pelo modelo 5, exibidos na Tabela 5, percebe-se que, em ambos os casos, as métricas aumentaram apenas para as espécies canário-do-amazonas e rolinha. Em relação às demais espécies, elas se mantiveram constantes ou foram reduzidas.

Tabela 12 — Métricas finais de precisão e de revocação da instância 5.10 para cada espécie ao se considerar 50% como o limiar de IoU.

Espécies	Precisão	Revocação	VP	FP	FN
canário-do-amazonas	0,8906	0,9828	57	7	1
chupim	0,8810	0,9487	37	5	2
rolinha	1,0000	1,0000	27	0	0
sanhaço-da-amazônia	0,9200	1,0000	23	2	0
sanhaço-do-coqueiro	0,7333	0,8800	22	8	3
Média	0,8850	0,9623			

Fonte: De autoria própria.

Tabela 13 — Métricas finais de precisão e de revocação da instância 5.14 para cada espécie ao se considerar 50% como o limiar de IoU.

Espécies	Precisão	Revocação	VP	FP	FN
canário-do-amazonas	0,9344	0,9828	57	4	1
chupim	0,9000	0,9231	36	4	3
rolinha	0,9643	1,0000	27	1	0
sanhaço-da-amazônia	0,8519	1,0000	23	4	0
sanhaço-do-coqueiro	0,8462	0,8800	22	4	3
Média	0,8993	0,9572			

Fonte: De autoria própria.

Em relação à instância 5.10, houve uma queda acentuada de precisão para a espécie sanhaço-do-coqueiro quando comparada com o modelo 5 ao passar de 88% para 73,33% dado o acréscimo de 5 falsos positivos. Já em relação à instância 5.14, a redução para a mesma espécie é menor, já que passou de 88% para 84,62% dado o acréscimo de apenas 1 falso positivo. Vale destacar que para as duas instâncias, a espécie sanhaço-do-coqueiro possui tanto a menor precisão, quanto a menor revocação.

Além disso, nota-se que a quantidade de falsos positivos da instância 5.14 está distribuída de forma mais uniforme entre as espécies do que em relação à instância 5.10, os quais se concentram mais nas espécies canário-do-amazonas e sanhaço-do-coqueiro. De fato, o desvio padrão para a coluna de falsos positivo da instância 5.10 é de 3,01, enquanto que da instância 5.14 é de 1,2, revelando de fato maior dispersão no primeiro caso.

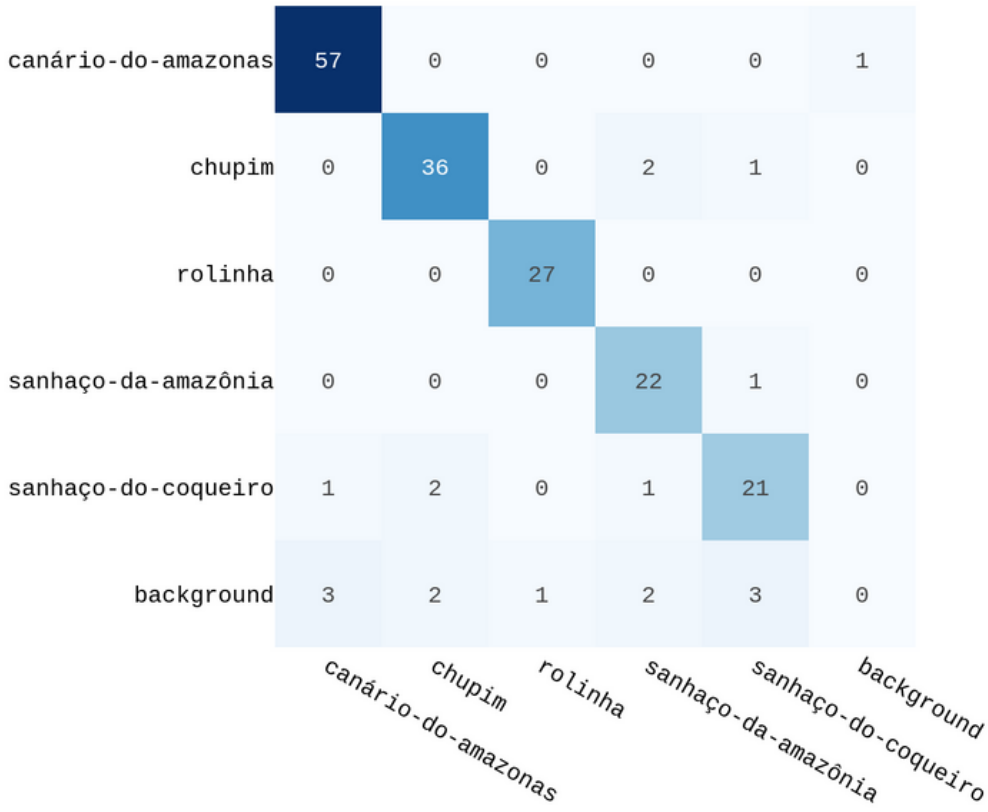
A precisão média da instância 5.14 também é levemente maior do que a da instância 5.10, apesar do oposto ser verdade para a revocação. Nesse caso, como os valores são próximos, priorizou-se a precisão, já que se espera que as detecções do modelo sejam as mais corretas possíveis nesse primeiro momento.

Diante disso, apesar da instância 5.10 apresentar maior mAP, percebe-se que a instância 5.14 se destaca mais em relação à precisão média e à distribuição dos falsos positivos. Além disso, por mais que se tenha notado a queda das métricas de precisão e de revocação para algumas espécies, as duas instâncias ainda superam o desempenho do modelo 5 em termos de mAP.

A Figura 10 apresenta a matriz de confusão da instância 5.14. No Apêndice C, é possível visualizar algumas das detecções ressaltadas. Assim como nos casos anteriores, houve a ocorrência de detecções com nível de confiança menor, mas que classificaram corretamente os indivíduos.

Percebe-se que houve algumas melhorias ao compará-la com a matriz de confusão do modelo 5, já que 2 pássaros da espécie sanhaço-do-coqueiro deixaram de ser detectados como canário-do-amazonas, além de outros 2 da espécie sanhaço-da-amazônia deixarem de ser classificados como sanhaço-do-coqueiro. No entanto, há o acréscimo de novas confusões, já que 1 chupim foi detectado como sanhaço-do-coqueiro, além de outra detecção ter classificado 1 sanhaço-do-coqueiro como sanhaço-da-amazônia. Porém, no total, o número de confusões geral foi reduzido de 11 para 8.

Figura 10 — Matriz de confusão para a instância 5.14 eleita na etapa de ajuste de hiperparâmetros.



Fonte: De autoria própria.

Finalmente, após a análise de cada instância, conclui-se que a instância 5.14 foi a que apresentou melhores resultados. Assim, ela será adotada como uma *baseline* para comparação. Além disso, o ajuste de hiperparâmetros empregado por ela será incorporado na configuração de treinamento do modelo em construção.

3.1.2.1 Resumo da subseção

Nesta etapa, 14 instâncias do modelo 5 foram treinadas com variações nos valores dos hiperparâmetros. Nesse contexto, as instâncias 5.10 e 5.14 foram as que mais se destacaram ao alcançarem mAP de 76,05% e de 75,29% respectivamente, superando assim o desempenho do modelo 5. Dada a proximidade entre os resultados das duas instâncias, elas foram comparadas também em termos de precisão e de revocação ao se considerar 50% com o limiar de IoU. Por meio dessa análise, foi possível concluir que a instância 5.14 apresentou melhores

resultados ao atingir a maior precisão média de 89,93%, além dos falsos positivos estarem distribuídos de forma mais uniforme entre as espécies. Dessa forma, decidiu-se incorporar o ajuste de hiperparâmetros da instância 5.14 na configuração de treinamento do modelo em desenvolvimento. Nesse sentido, as imagens passarão a ser redimensionadas inicialmente para o tamanho de 1024x1024 durante o *presizing*, retornando para o tamanho de 896x896 durante o treinamento. Por fim, a instância 5.14 também foi definida como uma *baseline* para comparação.

3.2 FASE FINAL

Na fase final, a construção do modelo foi concluída. Para tanto, um único modelo, chamado de definitivo, foi treinado com o conjunto de dados total através da configuração de treinamento exposta na Tabela 14. Além disso, o modelo definitivo foi comparado com a *baseline* com o intuito de verificar o aumento de desempenho proporcionado pelo fornecimento de mais dados.

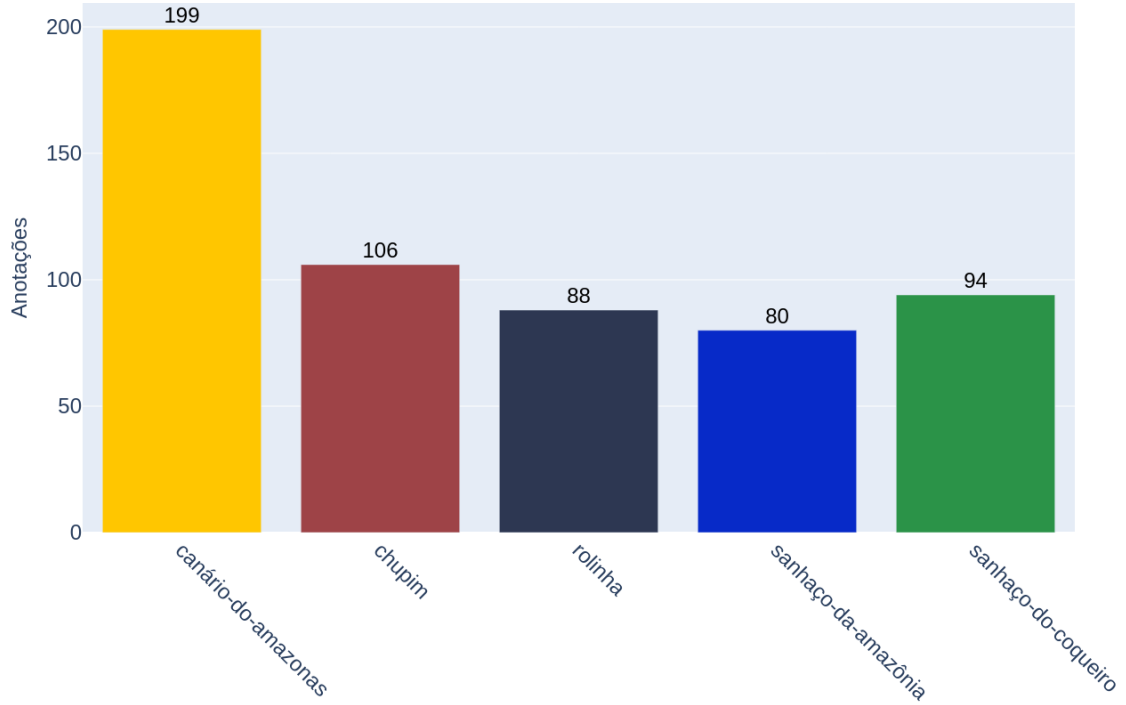
Tabela 14 — Configuração de treinamento na fase final.

Hiperparâmetros	Valores
Backbone	ResNeXt101 32x4d FPN 1x
Número de *epochs*	20
Embaralhamento do conjunto de treinamento	Não
Taxa de aprendizagem	10^{-4}
Tamanho do *batch*	1
Tamanho da imagem	896x896
Tamanho de redimensionamento no *presizing*	1024x1024

Fonte: De autoria própria.

Nesse contexto, 70% do conjunto total foi destinado para treinar o modelo definitivo, enquanto os 30% restantes foi utilizado para teste. Dessa forma, o conjunto de teste desta fase conta com 282 imagens e com 199 anotações para a espécie canário-do-amazonas, 106 para chupim, 88 para rolinha, 80 para sanhaço-da-amazônia e 94 para sanhaço-do-coqueiro conforme colocado pelo Gráfico 2.

Gráfico 2 — Proporção de anotação por espécie da parcela de teste do conjunto de dados total.



Fonte: De autoria própria.

A apresentação dos resultados alcançados pelo modelo definitivo, bem como a comparação com a *baseline* ocorrerão no final do capítulo 4 de modo a encerrar a exposição da proposta levantada neste trabalho.

4 **PROPOSTA**

Neste capítulo, a proposta deste trabalho de conclusão será detalhada. Assim, na seção 4.1 será discutido como o conjunto de dados foi construído, na seção 4.2 será abordado o processo de treinamento do detector Faster R-CNN a partir da biblioteca IceVision, na seção 4.3 será mencionado como aplicar o modelo definitivo para realizar novas detecções, na seção 4.4 serão informados os procedimentos de avaliação com a biblioteca FiftyOne e na seção 4.5 o resultado final do modelo definitivo será apresentado.

4.1 CONJUNTO DE DADOS

O estudo teve acesso a uma residência localizada no município de Santana no estado do Amapá que conta com um comedouro aberto em seu jardim conforme apresentado na Figura 11. De modo a atrair as aves para observação, são depositadas sementes, como alpiste, e frutas, como banana e mamão.

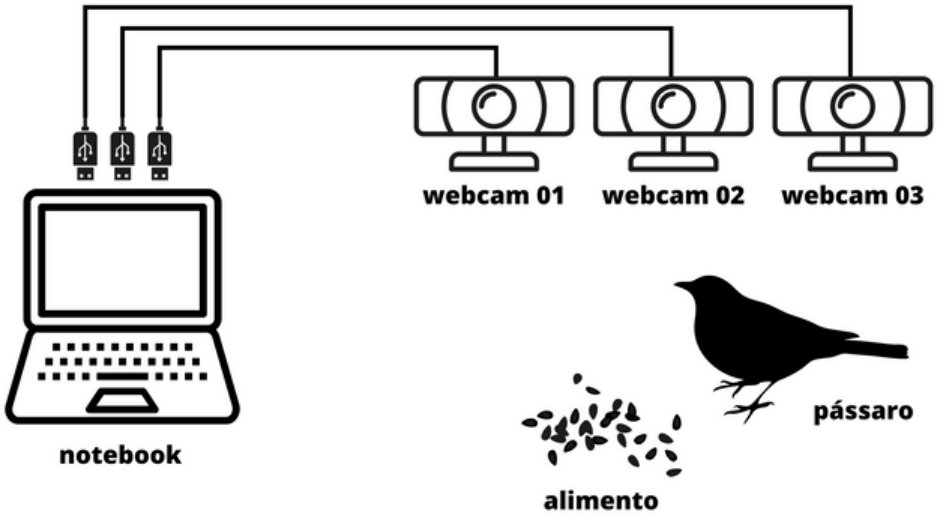
Figura 11 — Comedouro da residência onde eram dispostos alimentos para atrair os pássaros.

Fonte: De autoria própria.

A fim de registrar os pássaros se alimentando, foram posicionadas três *webcams* do modelo Logitech C270, capazes de gravar em 720p com 3 MP de resolução, nas colunas do comedouro conforme esquematizado na Figura 12. Elas se conectavam às entradas USB de um notebook, no qual um script

escrito em Python capturava e armazenava localmente as gravações a cada 5 minutos no formato AVI por meio da biblioteca OpenCV[4]. As capturas ocorreram entre os dias 07/09/2022 e 15/09/2022.

Figura 12 — Esquema para gravação dos pássaros.



Fonte: De autoria própria.

Com o intuito de determinar as espécies, os pássaros presentes nas gravações foram comparados com registros disponibilizados em plataformas de ciência cidadã como WikiAves e eBird. Além disso, foi empregado também o catálogo de aves encontradas na cidade de Manaus dos autores D'affonseca, Macedo e Cohn-haft (2012). A partir disso, foi possível determinar cinco espécies conhecidas pelos nomes populares de canário-do-amazonas, chupim, rolinha, sanhaço-da-amazônia e sanhaço-do-coqueiro.

Vale destacar que em relação a espécie rolinha, foram identificados indivíduos com características visuais diferentes, porém foi decidido mantê-los na mesma categoria. Além disso, foram registrados prováveis fêmeas e machos da espécie chupim, que também apresentam atributos distintos. Por fim, apesar de terem sido registrados alguns bem-te-vis, eles não foram considerados na composição final do conjunto de dados pela baixa frequência com que aparecem.

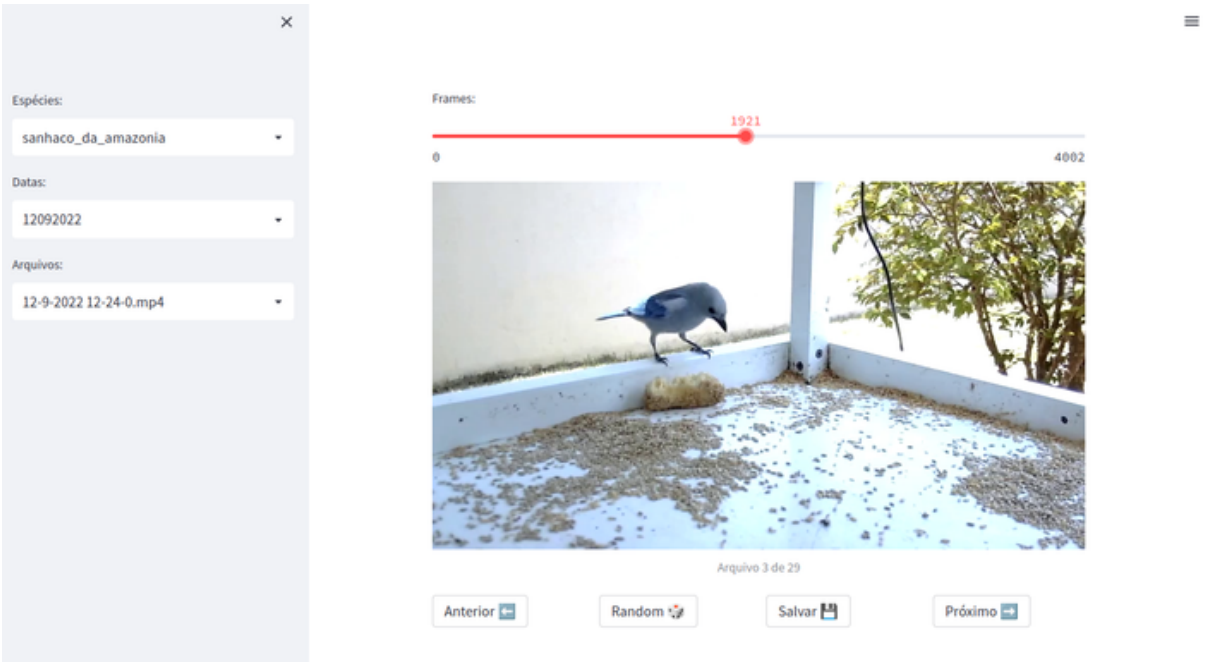
Os vídeos foram convertidos para o formato MP4 com o intuito de reduzir o espaço de armazenamento. Além disso, aqueles entre os dias 07/09 até 13/09 foram recortados manualmente com o objetivo de remover os momentos onde não havia a presença de pássaros, o que levou cerca de uma semana para ser concluído. A

4 OpenCV é uma biblioteca open source destinada ao desenvolvimento de aplicações voltadas para o processamento de imagem e visão computacional.

partir desses recortes, *frames* seriam extraídos para comporem o conjunto de dados.

De modo a facilitar esse processo, os recortes foram organizados em pastas com base na espécie de maior destaque presente neles e na data de gravação. Então, foi desenvolvida uma interface gráfica[5] por meio da qual era possível filtrar os recortes a partir da espécie e da data de gravação, além de extrair um *frame* aleatório ou em alguma posição específica. A interface foi implementada com a linguagem Python através da biblioteca Streamlit[6], podendo ser visualizada na Figura 13.

Figura 13 — Interface da aplicação desenvolvida em Streamlit para extração dos frames dos vídeos.



Fonte: De autoria própria.

Por meio dela, houve a extração de 940 *frames* que foram carregados na plataforma RoboFlow[7] com o objetivo de serem anotados. Vale mencionar que a plataforma conta com ferramentas que facilitam a anotação de dados voltados para tarefas de visão computacional, como a de detecção de objetos, além de contar com outros recursos para transformação dos dados e diversos formatos de anotação.

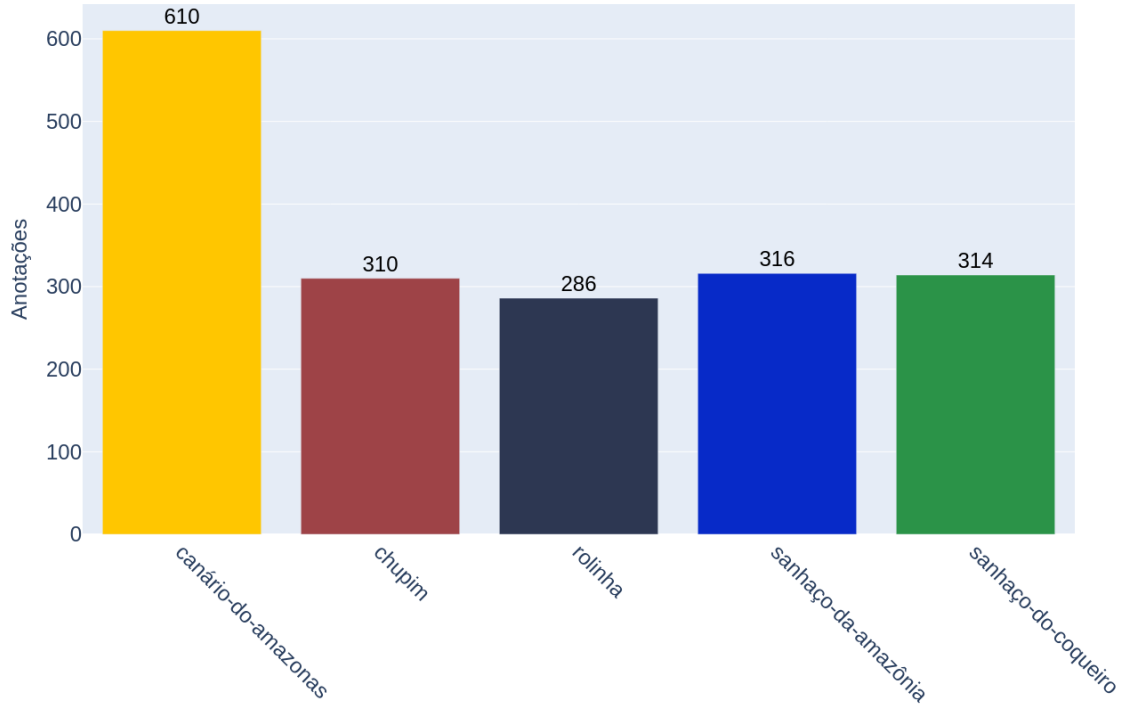
5 O código de implementação da interface pode ser encontrado no seguinte repositório: https://github.com/Lucas-Zampar/detector_de_passaros_amazonicos/tree/main/streamlit_app
6 Streamlit é um framework open source que facilita o desenvolvimento de aplicações web voltadas para a ciência de dados.
7 Roboflow é uma plataforma online na qual é possível anotar imagens para diferentes tarefas de visão computacional.

Nesse contexto, cada imagem foi anotada manualmente ao se desenhar a caixa delimitadora sobre os pássaros presentes nela, bem como de associá-las com as respectivas espécies. Dessa forma, tanto a localização real dos pássaros, representada pela caixa delimitadora, bem como a classe, denotada pela espécie, puderam ser especificadas. Ao todo, foram realizadas 1.836 anotações manuais, o que demandou cerca de uma semana para ser concluído.

A fim de realizar o *download* do conjunto de dados anotado, foi necessário selecionar o formato das anotações entre os disponíveis. Assim, foi adotado o formato do conjunto Pascal VOC[8] por ser um dos mais conhecidos, além de ter sido considerado o mais intuitivo. Nele, as anotações são armazenadas em arquivos XML separados para cada imagem e as caixas delimitadoras são representadas pelas coordenadas retangulares do ponto superior esquerdo e inferior direito.

A proporção de anotações por espécie de pássaro do conjunto de dados total pode ser visualizada no Gráfico 3. Ao todo, foram 610 anotações da espécie canário-do-amazonas, 310 da chupim, 286 da rolinha, 316 da sanhaço-da-amazônia e 314 da sanhaço-do-coqueiro.

Gráfico 3 — Proporção de anotações por espécie do conjunto de dados total.



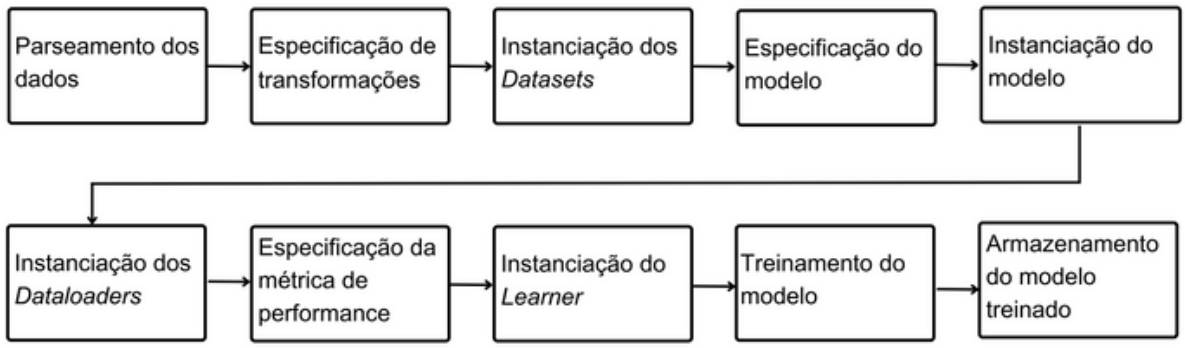
Fonte: De autoria própria.

8 Pascal VOC é um conjunto de benchmark para a tarefa de detecção de objetos.

4.2 TREINAMENTO

Entre os *frameworkers* disponíveis em Python para o treinamento de modelos de detecção de objetos, é possível encontrar o IceVIsion que oferece centenas de modelos pré-treinados para diversas tarefas de visão computacional, além de oferecer um fluxo de trabalho simplificado para o treinamento deles, como o adotado neste trabalho que é apresentado na Figura 14.

Figura 14 — Fluxo de trabalho para o treinamento de um modelo de detecção de objetos por meio do framework IceVision.



Fonte: De autoria própia.

O parseamento dos dados consiste em associar as anotações com suas respectivas imagens através de uma representação interna do IceVision denominada de *record* a partir da qual informações úteis, como coordenadas de caixas delimitadoras e suas classes, podem ser acessadas. Para o formato de anotações Pascal VOC que foi empregado no conjunto de dados, o parseamento é realizado automaticamente pelo *framework*.

Além disso, é nessa etapa que deve ser especificada a proporção de dados reservadas para o treinamento e para a avaliação do modelo. Segundo Elgendy (2020), em projetos de aprendizagem de máquina geralmente são empregados 70% ou 80% dos dados para o treinamento, enquanto 30% ou 20% são destinados para o teste. Neste trabalho, optou-se pela proporção 70% e 30% para os conjuntos de treinamento e de teste respectivamente. Vale destacar que o IceVision gera *records*

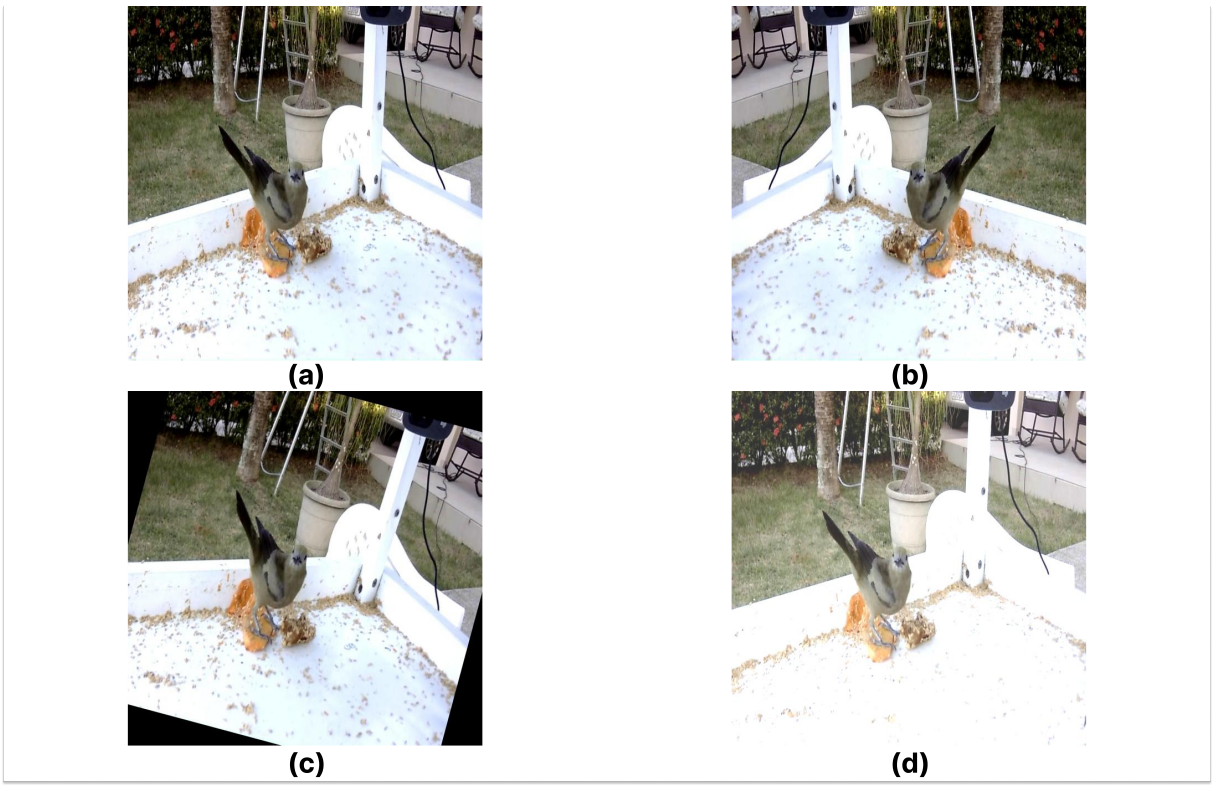
individuais para os dois conjuntos.

Em seguida, devem ser informadas as transformações que o IceVision irá aplicar sobre os dados, o que é feito com auxílio da ferramenta Albumentations[9] especializada em transformações voltadas para a visão computacional. Por exemplo, é comum realizar o redimensionamento da imagem para um tamanho fixo, bem como a normalização dos valores de seus *pixels.*

Nesse sentido, o IceVision disponibiliza também um conjunto de transformações padrões para o aumento de dados como inversão horizontal em relação ao eixo y, translação, alteração de escala, rotação, mudança nos valores de *pixels* nos canais de cores da imagem, alteração no brilho e contraste, adição de borrado, recorte de região seguido de reescalonamento sem comprometimento da caixa delimitadora e adição de *padding* se necessário.

Na Figura 15, é possível observar alguns exemplos de transformações que podem ser aplicadas sobre uma imagem. Em (a) é apresentada a imagem original de tamanho 640x640. Em (b), observa-se a inversão horizontal em relação ao eixo y. Em (c), a imagem é rotacionada por um ângulo de -15°. Por fim, em (d), houve acréscimo de brilho por um fator de 0,2.

Figura 15 — Exemplos de transformações em imagens.



Fonte: De autoria própria.

9 Albumentations é uma biblioteca a partir da qual diversas transformações podem ser aplicadas sobre imagens a fim de aumentar os dados.

Vale destacar que as transformações de aumento de dados são aplicadas apenas quando a imagem é acessada, além da ocorrência delas ser aleatória e dentro de limites pré-definidos. Por exemplo, as imagens só podem ser rotacionados entre os ângulos -15° e 15°, enquanto o brilho e o contraste só podem ser ajustados entre os fatores -0,2 e 0,2.

Dessa forma, cada vez que a imagem é acessada em uma *epoch*, transformações diferentes poderão ser aplicadas, o que leva a imagens ligeiramente diferentes a cada acesso contribuindo assim com o aumento dos dados. Além disso, as transformações ocorrem em tempo de execução, isto é, não são geradas imagens estáticas no processo.

Para o conjunto de treinamento, foram especificadas as transformações de redimensionamento da imagem para um tamanho fixo que é definido manualmente no código, a normalização dos valores de *pixels* e o grupo padrão de transformações de aumento que ocorrem de forma aleatória. Para o conjunto de teste, não é interessante que ocorram transformações de aumento, já que isso pode comprometer a avaliação do modelo em cada *epoch*, logo apenas o redimensionamento definido manualmente e a normalização foram escolhidas.

Uma vez que os *records* foram gerados e as transformações especificadas, é possível instanciar objetos denominados de *datasets* que representam de fato os conjuntos de treinamento e de teste. Assim, o IceVision saberá quais dados pertencem a cada conjunto, além das transformações que deve aplicar neles.

A partir disso, é possível especificar e instanciar o modelo de detecção de objetos acompanhado do seu respectivo *backbone*. Neste trabalho, o detector escolhido foi o Faster R-CNN por ser de dois estágios, o que geralmente leva a uma performance melhor em relação aos de único estágio.

O IceVision disponibiliza o modelo Fastrer R-CNN através da biblioteca MMDetection proposta pelos autores Chen *et al.* (2019). Por meio dela, encontram-se modelos pré-treinados com diferentes redes de *backbone* no conjunto COCO com a técnica de *learning rate scheduling,* na qual a taxa de aprendizagem é reduzida de acordo com uma determinada escala. Nesse contexto, os modelos disponibilizados são treinados na escala 1x por 12 *epcochs,* com redução nas *epochs* 8 e 11, e na escala 2x por 24 *epochs,* com redução nas *epochs* 16 e 22*,* ambas empregando o fator de redução de 10^{-1}.

O próximo passo é a instanciação dos *dataloaders* de treinamento e de teste que são responsáveis por selecionar os itens dos respectivos *datasets*, bem como por agrupá-los em *batches* de acordo com o formado específico de cada modelo. É nesse momento que são definidos hiperparâmetros como tamanho do

batch, que se refere ao número de imagens apresentadas por vez para o cálculo do gradiente*,* e a ocorrência de embaralhamento do conjunto de treinamento em cada *epoch*.

Em seguida, uma métrica pode ser especificada para visualizar evolução da performance em cada *epoch.* Nesse ponto, todos os elementos já foram determinados para a inicialização do treinamento que é controlado por meio de uma instância da classe *Learner,* introduzida inicialmente pela biblioteca Fastai pelos autores Howard e Gugger (2020) e adaptada para os modelos de detecção do IceVision*.*

A fim de instanciá-la, é necessário fornecer ao modelo os *dataloaders* e a métrica. Com base nisso, o treinamento pode ser inicializado, bastando especificar o número de *epochs*, bem como a taxa de aprendizagem. É nesse processo que o modelo pré-treinado será ajustado com o conjunto de dados construído e adaptado para a tarefa de detecção das espécies dos pássaros. Por fim, os parâmetros do modelo treinado, bem como seus metadados, são salvos em um arquivo de *checkpoint* no formato PTH.

De modo a facilitar o aplicação do fluxo de trabalho, foram escritas funções em um arquivo separado responsáveis por invocar os procedimentos do IceVision, funcionando portanto como *wrappers*. A partir disso, tais funções são importadas em um notebook[10] Jupyter, no qual o fluxo pode ser implementado de forma mais organizada e documentada.

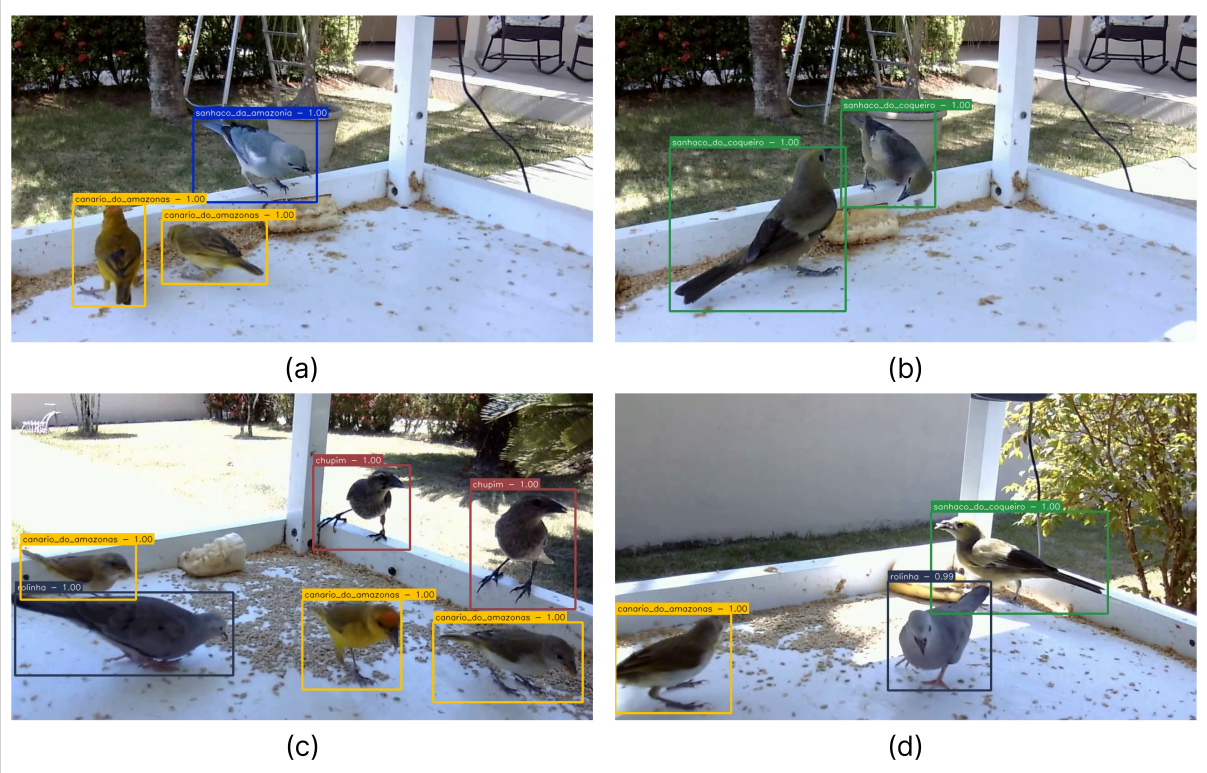
O treinamento de todos os modelos ocorreu em uma máquina local com as seguintes especificações: sistema operacional Linux Mint 21 Cinnamon, processador AMD Ryzen 5 3600, 16 Gb de memória RAM, GPU Nvidia RTX 2060 Super com 8 Gb de VRAM. Além disso, de modo a instalar a biblioteca IceVision e demais dependências, foi criado um ambiente virtual com a versão 3.9.13 do Python através do gerenciador Mamba.

4.3 APLICAÇÃO

Uma vez que o modelo definitivo foi treinado com a base de dados completa, é possível empregá-lo para realizar inferências através do IceVision. Assim, o modelo pode realizar detecções para novas imagens fornecidas no futuro. Por exemplo, a Figura 16 apresenta algumas detecções realizadas em frames extraídos manualmente de segmentos das gravações do dia 14/09/2022 que não foram utilizadas nem para o treinamento, nem para a avaliação.

10 Um notebook é um tipo de documento interativo no qual é possível escrever e executar código Python através da aplicação web chamada Jupyter Notebook.

Figura 16 — Detecções realizadas para dados não empregados durante o treinamento e avaliação.



Fonte: De autoria própria.

De modo a produzir novas detecções com o modelo definitivo, foram desenvolvidos dois notebooks[11]. No primeiro, nomeado *detect_image*, o modelo definitivo pode realizar inferências sobre uma imagem de entrada. Já no segundo, chamado *detect_video*, é possível carregar um vídeo a fim de que as inferências ocorram para cada *frame*[12].Em ambos os casos, as detecções, o que inclui as caixas delimitadoras, classes e níveis de confiança, são anotadas diretamente nos arquivos com o auxílio da biblioteca OpenCV.

4.4 AVALIAÇÃO

A avaliação do desempenho dos modelos foi realizada por meio do FiftyOne, um *framework* que oferece diversos recursos para análise e visualização dos resultados. Para avaliar as predições do modelo, é necessário criar um conjunto de dados próprio do FiftyOne que contenha as imagens de teste, as anotações e as detecções realizadas, o que pode ser realizado por meio do IceVision.

11 Os notebooks podem ser acessados no seguinte repositório: https://github.com/Lucas-Zampar/detector_de_passaros_amazonicos/tree/main/codigos_de_desenvolvimento
12 Alguns vídeos foram anotados e disponibilizados publicamente na seguinte playlist do YouTube: https://www.youtube.com/watch?v=2AbiWK365c0&list=PLRrYiy77L8XEwZ8TIzThM9LNaanQ2eAE0&index=1

De modo a construir tal conjunto, o *checkpoint* gerado após o treinamento do modelo é carregado pelo IceVision. A partir disso, ocorre novamente o parseamento dos dados, a especificação das transformações empregadas no conjunto de teste e a criação do respectivo *dataset*. Então, as predições sobre as imagens de teste são realizadas pelo modelo e repassadas para o método do IceVision responsável por criar o conjunto de dados do FiftyOne.

O FiftyOne oferece diversos métodos para avaliar o desempenho do modelo a partir do conjunto criado. Por exemplo, é possível calcular as métricas mAP, mAP@.50 e mAP@.75 de acordo com os critérios da competição COCO. Além disso, é permitido computar a precisão e revocação por classe, ou seja, por espécie. Por fim, pode ser gerada a matriz de confusão que auxilia a verificar entre quais espécies o modelo apresenta maior dificuldade de distinção.

Com o intuito de empregar os procedimentos citados, foram escritos dois arquivos separados contendo as funções necessárias para a criação do conjunto de dados do FiftyOne e para o cálculo das métricas de performance por ela respectivamente. Tais funções são importadas no mesmo *notebook* onde ocorre o treinamento dos modelos de forma que no final de cada treinamento, as predições dos modelos já sejam armazenadas e avaliadas pelo FiftyOne.

Em relação ao fluxo de avaliação, são gerados dois arquivos CSV denominados de *final_metrics* e *metrics_per_class* nos quais se encontram o resultado das métricas gerais de avaliação de mAP e das específicas por classe respectivamente. Além disso, é gerado também um arquivo PNG contendo a figura da matriz de confusão.

4.5 RESULTADO FINAL

O treinamento do modelo Faster R-CNN com o conjunto de dados total, chamado de modelo definitivo, levou a valores superiores para as métricas de avaliação mAP, mAP@.50 e mAP@.75 do que aqueles alcançados apenas com o conjunto parcial pela *baseline* como pode ser observado na Tabela 15. Nota-se que a mAP, principal métrica de avaliação, passou de 75,29% para 81,89%, representando um aumento percentual de 8,77% no desempenho. Isso indica que o treinamento com mais dados beneficiou a performance do modelo.

Tabela 15 — Comparação dos valores das métricas mAP alcançadas pela baseline e pelo modelo definitivo.

Modelo	mAP	mAP@.50	mAP@.75
Baseline	0,7529	0,9459	0,8851
Modelo Definitivo	0,8189	0,9833	0,9572

Fonte: De autoria própria.

A fim de verificar como ele se comporta em relação às espécies, a Tabela 16 exibe os valores das métricas de precisão e de revocação para cada espécie ao se considerar 50% como o limiar para a *Intersection over Union* (IoU). Também são apresentados o total de VPs, de FPs e de FNs como referência.

Tabela 16 — Métricas finais de precisão e de revocação do modelo definitivo para cada espécie ao se considerar 50% como o limiar de IoU.

Espécies	Precisão	Revocação	VP	FP	FN
canário-do-amazonas	0,9515	0,9849	196	10	3
chupim	0,9725	1,0000	106	3	0
rolinha	0,9663	0,9773	86	3	2
sanhaço-da-amazônia	0,9877	1,0000	80	1	0
sanhaço-do-coqueiro	0,9200	0,9787	92	8	2
Média	0,9596	0,9882			

Fonte: De autoria própria.

Além disso, de modo a facilitar a comparação entre os dois modelos em relação a essas métricas, a Tabela 17 apresenta os valores alcançados por cada um. Dessa forma, percebe-se que o modelo definitivo alcançou valores de precisão e de revocação superiores ao da *baseline* para cada espécie, exceto pela redução da revocação de 100% para 97,73% em relação à espécie rolinha. Além disso, é notável que a menor precisão ainda ocorre para a espécie sanhaço-do-coqueiro entre os dois modelos, apesar de seu valor ter aumentado de 84,62% para 92% o que reflete um crescimento percentual de 8,72% na precisão dessa espécie.

Tabela 17 — Métricas finais de precisão e de revocação da baseline e do modelo definitivo ao se considerar 50% como o limiar de IoU.

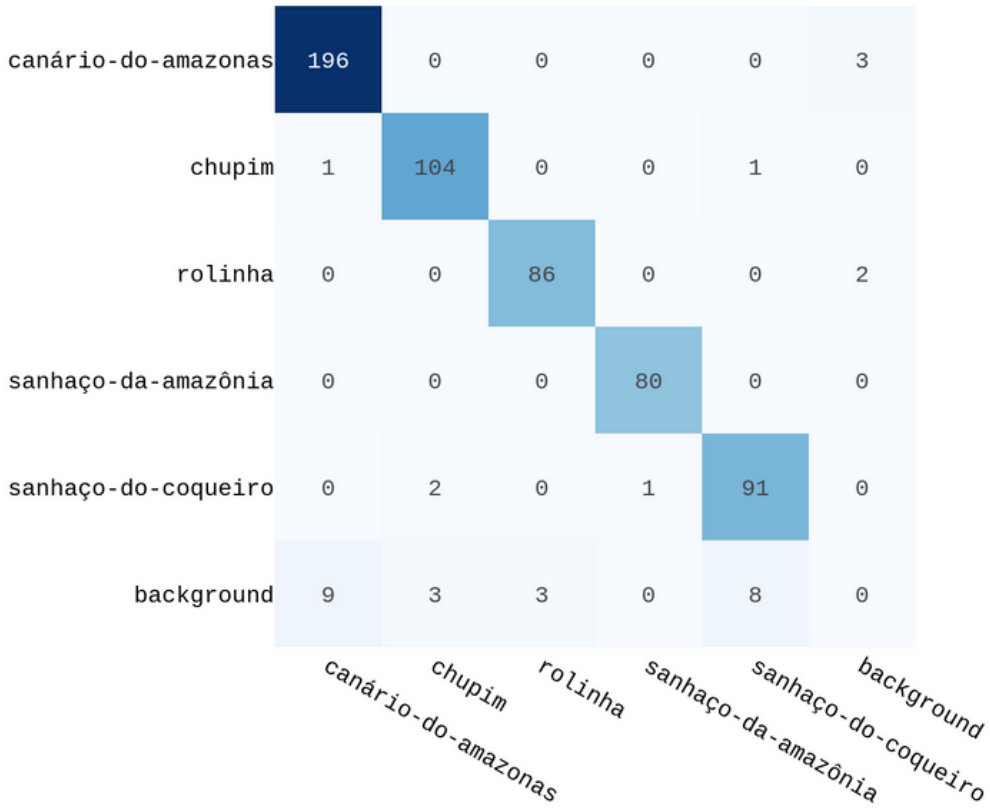
Espécies	Precisão Baseline	Precisão Modelo Definitivo	Revocação Baseline	Revocação Modelo Definitivo
canário-do-amazonas	0,9344	0,9515	0,9828	0,9849
chupim	0,9000	0,9725	0,9231	1,0000
rolinha	0,9643	0,9663	1,0000	0,9773
sanhaço-da-amazônia	0,8519	0,9877	1,0000	1,0000
sanhaço-do-coqueiro	0,8462	0,9200	0,8800	0,9787
Média	0,8993	0,9596	0,9572	0,9882

Fonte: De autoria própria.

Em geral, o modelo definitivo apresentou um ganho percentual de precisão média de 6,7% em comparação com a *baseline*, ao passar de 89,93% para 95,96%. Já a revocação média teve um crescimento percentual menor de 3,24% saindo de 95,72% para 98,82%.

Na Figura 17, é possível visualizar a matriz de confusão do modelo definitivo. No Apêndice D, as detecções destacas nela são apresentadas. Diante disso, nota-se uma clara melhoria, já que o número de confusões principais passou de 8 para 5, levando em consideração que o número de imagens analisadas foi maior. Porém, algumas confusões ainda persistem, como 2 indivíduos sanhaço-do-coqueiro detectados como chupim e outro como sanhaço-da-amazônia, enquanto 1 chupim é classificado como sanhaço-do-coqueiro.

Figura 17 — Matriz de confusão para o modelo definitivo.



Fonte: De autoria própria.

Diante do que foi exposto, é possível perceber que o modelo definitivo superou a *baseline* ao ser treinado com mais dados, alcançando resultados que demonstram a viabilidade de se empregar uma abordagem baseada em aprendizado profundo para a detecção de espécies de pássaros em um contexto residencial.

No entanto, com base nas matrizes de confusão e nos apêndices, é válido destacar que o modelo pode gerar múltiplas detecções sobre um mesmo indivíduo, o que pode comprometer a contagem de espécies na imagem em certos momentos. Por outro lado, espera-se que essa dificuldade possa ser contornada com a apresentação de mais dados de treinamento como ocorreu na passagem da *baseline* para o modelo definitivo.

5 **CONCLUSÃO**

Por meio deste trabalho, foi demostrado que é possível detectar automaticamente espécies de pássaros a partir de um contexto residencial através de uma abordagem baseada em aprendizado profundo. Para tanto, 3 *webcams* foram empregadas para registrar pássaros amazônicos que frequentam o comedouro de uma residência localizada no município de Santana, Amapá.

A partir disso, foi possível construir um conjunto de dados composto por 940 imagens e 1.836 anotações distribuídas entre as espécies conhecidas pelos nomes populares de canário-do-amazonas, chupim, rolinha, sanhaço-da-amazônia e sanhaço-do-coqueiro.

Por meio do conjunto levantado, foram conduzidos experimentos divididos em duas fases principais. Na fase de experimentos preliminares, 30% dos dados foram separados para o treinamento e avaliação de modelos Faster R-CNN em diferentes configurações de modo a selecionar aquele que mais se destacasse como uma *baseline.* Nesse sentido, o modelo definido conseguiu atingir o percentual de 75,29% na métrica de avaliação geral mAP. Além disso, em relação às espécies individuais, ele alcançou precisão média de 89,93% e revocação média de 95,72% respectivamente.

Já na fase final, um único modelo foi treinado com a base de dados total por meio das configurações definidas anteriormente. Tal modelo superou a *baseline* em todas as métricas ao alcançar 81,89% para a mAP. Já em relação às espécies individuais, ele obteve precisão média de 95,96% e revocação média de 98,82%. No entanto, vale destacar que para alguns indivíduos, os dois modelos realizaram múltiplas detecções com classificações distintas, o que provavelmente pode ser contornado futuramente com o emprego de mais dados.

Vale destacar ainda que, de acordo com o melhor de nosso conhecimento, este trabalho é o primeiro a propor a aplicação de aprendizado profundo na detecção automática de espécies de pássaros amazônicos registrados em imagens dentro de um contexto residencial, bem como de anotar um conjunto de imagens dessas espécies para a tarefa de detecção de objetos. Com isso, observou-se a possibilidade de trabalhos futuros.

Nesse contexto, é estabelecido como proposta de melhoria de curto prazo adicionar mais imagens anotadas ao conjunto de dados. Para tanto, novos *frames* serão extraídos das gravações já existentes. Além disso, o modelo construído será empregado para anotar parcialmente esses *frames*, o que reduzirá o trabalho humano de anotação. Diante disso, é esperado construir um novo modelo com desempenho maior, uma vez que foi observado o aumento de performance com

o fornecimento de mais dados ao longo do trabalho.

Já como proposta de melhoria de médio prazo, um novo conjunto de dados será levantado com o objetivo de pré-treinar os modelos. Para tanto, imagens com maior qualidade que apresentem indivíduos de uma única espécie em destaque serão extraídas e anotadas partir das plataformas WikiAves e eBirds. Nesse ponto, é levantado como hipótese que modelos pré-treinados com essas imagens possam extrair características mais ricas de cada espécie do que se forem treinados diretamente com imagens obtidas por *webcams* dentro dos comedouros.

Por fim, como proposta de melhoria de longo prazo, será desenvolvido o protótipo de um sistema capaz de coletar dados em outras residências de forma autônoma. Para tanto, imagina-se confeccionar um comedouro de proporções menores, além de empregar o computador de placa única Raspberry PI 4 com uma câmera conectada a fim de registrar os pássaros. A estratégia de coleta e de armazenamento adotará uma estrutura híbrida de computação de borda e em nuvem, na qual os dados serão tradados localmente aproveitando os recursos computacionais do Raspberry PI 4, porém a agregação dos dados gerados em diferentes localidades ocorrerá por meio da nuvem. Com isso, espera-se coletar imagens em cenários diferentes ou até mesmo de novas espécies, o que contribuirá com a variabilidade no conjunto de dados.

Um dos potenciais desdobramentos da proposta de longo prazo é empregar o modelo construído para anotar os dados diretamente no Raspberry PI 4, deslocando assim a inferência da nuvem para a borda. Atualmente, a adoção da IA sobre a borda é discutida de modo a tornar as aplicações mais autônomas, além de reduzir a carga de trabalho na nuvem. Dessa forma, a perspectiva de IA sobre a borda poderá ser explorada também a longo prazo.

REFERÊNCIAS

ALEXANDRINO, Eduardo *et al*. Which birds are Brazilians seeing on urban and non-urban feeders? An analysis based on a collective online birding. **Ornithology Research**, 2022.

ALQAYSI, Hiba *et al*. A Temporal Boosted YOLO-Based Model for Birds Detection around Wind Farms. **Journal of Imaging**, v. 7, n. 11. 227 p, 2021.

BARBOSA, Karlla *et al*. The contribution of citizen science to research on migratory and urban birds in Brazil. **Ornithology Research**, 2021.

BENGIO, Yoshua; LECUN, Yann; HINTON, Geoffrey. Deep Learning for AI. **Communications of the ACM**, v. 64, n. 7, p. 58-65, 2021.

BOCHKOVSKIY, Alexey; WANG, Chien-Yao; LIAO, Hong-Yuan . YOLOv4: Optimal Speed and Accuracy of Object Detection. **ArXiv**, 2020.

BONTER, David; GREIG, Emma. Over 30 Years of Standardized Bird Counts at Supplementary Feeding Stations in North America: A Citizen Science Data Report for Project FeederWatch. **Frontiers in Ecology and Evolution**, 2021.

CHEN, Kai *et al*. MMDetection: Open MMLab Detection Toolbox and Benchmark. **ArXiv**, 2019.

DAYER, Ashley *et al*. Observations at backyard bird feeders influence the emotions and actions of people that feed birds. **People and Nature**, 2019.

D'AFFONSECA, Anselmo; MACEDO, Ingrid; COHN-HAFT, Mario. **Aves da Região de Manaus**. INPA, 2012.

EBIRD. **eBird**: Discover a new world of birding. 2023. Disponível em: https://ebird.org/home. Acesso em: 8 mai. 2023.

ELGENDY, Mohamed. **Deep Learning for Vision Systems**. Manning Publications, f. 239, 2020. 478 p.

FELZENSZWALB, Pedro; MCALLESTER, David; RAMANAN, Deva. A discriminatively trained, multiscale, deformable part model. *In:* IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION. 2008.

GIRSHICK, Ross *et al*. Rich feature hierarchies for accurate object detection and semantic segmentation. *In:* IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION. 2014, p. 580-587.

GIRSHICK, Ross. Fast R-CNN. *In:* IEEE INTERNATIONAL CONFERENCE ON COMPUTER VISION. 2015, p. 1440-1448.

GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. **Deep Learning**. MIT Press, v. 1, f. 401, 2016. 801 p.

GU, Jiuxiang *et al*. Recent Advances in Convolutional Neural Networks. **ArXiv**, 2015. Disponível em: https://arxiv.org/abs/1512.07108. Acesso em: 24 mar. 2023.

HE, Kaiming *et al*. Deep Residual Learning for Image Recognition. **ArXiv**, 2015.

HOWARD, Jeremy; GUGGER, Sylvain. **Deep Learning for Coders with fastai and PyTorch**. 1 ed. O'Reilly Media, 2020. 621 p.

HUANG, YO-PING; BASANTA, HAOBIJAM. Recognition of Endemic Bird Species Using Deep Learning Models. **IEEE Access**, v. 9, 2021.

KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey. ImageNet Classification with Deep Convolutional Neural Networks*. In:* ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS. 2012.

LECUN, Yann *et al*. Gradient-based learning applied to document recognition. **IEEE**, 1998.

LECUN, Yann; BENGIO, Yoshua; HINTON, Geoffrey. Deep Learning. **Nature**, v. 521, p. 436-444, 2015.

LIN, Tsung-Yi *et al*. Feature Pyramid Networks for Object Detection. **ArXiv**, 2016.

LIN, Tsung-Yi *et al*. Microsoft coco: Common objects in context*. In:* COMPUTER VISION–ECCV 2014. 2014.

MAO, Xin *et al*. Domain randomization-enhanced deep learning models for bird detection. **Scientific reports**, v. 11, n. 1. 639 p, 2021.

MIRUGWE, Alex; NYIRENDA, Juwa; DUFOURQ, Emmanuel. Automating Bird Detection Based on Webcam Captured Images using Deep Learning*. In:* CONFERENCE OF THE SOUTH AFRICAN INSTITUTE OF COMPUTER SCIENTISTS AND INFORMATION TECHNOLOGISTS. 2022.

NAVNEET, Dalal; TRIGGS, Bill. Histograms of oriented gradients for human detection. **IEEE computer society conference on computer vision and pattern recognition**, 2005.

PADILLA, Rafael ; NETTO, Sergio ; SILVA, Eduardo . A Survey on Performance Metrics for Object-Detection Algorithms*. In:* 2020 INTERNATIONAL CONFERENCE ON SYSTEMS, SIGNALS AND IMAGE PROCESSING (IWSSIP). 2020.

PADILLA, Rafael *et al*. A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit. **Electronics**, v. 10, n. 3, 2021.

PINHEIRO, Brenda; SOARES, Lucas. Sistema de Reconhecimento Automático de Pássaros da Fauna de Linhares - ES Utilizando Redes Neurais Convolucionais*. In:* SIMPÓSIO BRASILEIRO DE AUTOMAÇÃO INTELIGENTE - SBAI. 2021.

QIU, Zhibin *et al*. Detection of bird species related to transmission line faults based on lightweight convolutional neural network. **IET Generation, Transmission &**

Distribution, v. 16, n. 5, p. 869-881, 2021.

REDMON, Joseph *et al*. You Only Look Once: Unified, Real-Time Object Detection. **ArXiv**, 2015.

REN, Shaoqing *et al*. Faster R-CNN: Towards real-time object detection with region proposal networks. **Advances in Neural Information Processing Systems**, v. 28, 2015.

RUSSELL, Stuart; NORVIG, Peter. **Artificial Intelligence: A Modern Approach, Global Edition**. Pearson Higher Ed, f. 584, 2021. 1167 p.

SHI, Xiaohang *et al*. Detection of Flying Birds in Airport Monitoring Based on Improved YOLOv5*. In:* IEEE 6TH INTERNATIONAL CONFERENCE ON INTELLIGENT COMPUTING AND SIGNAL PROCESSING (ICSP 2021). 2021.

SIMONYAN, Karen; ZISSERMAN, Andrew. Very Deep Convolutional Networks for Large-Scale Image Recognition*. In:* ICLR. 2015.

Srijan; Samriddhi; GUPTA, Deepak . Mobile Application for Bird Species Identification Using Transfer Learning*. In:* IEEE INTERNATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE IN ENGINEERING AND TECHNOLOGY (IICAIET). 2021.

TAN, Mingxing *et al*. Efficientdet : Scalable and efficient object detection*. In:* IEEE/CVF CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION. 2020, p. 10781-10790.

VIOLA, Paul; JONES, Michael. Rapid object detection using a boosted cascade of simple features*. In:* IEEE COMPUTER SOCIETY CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION. 2001.

WHITE, Maria *et al*. The Joy of birds: the effect of rating for joy or counting garden bird species on wellbeing, anxiety, and nature connection. **Urban Ecosystems** , 2023.

WIKIAVES. **WikiAves**: A Enciclopédia das Aves do Brasil. 2023. Disponível em: https://www.wikiaves.com.br. Acesso em: 8 mai. 2023.

XIANG, Wenbin *et al*. Birds Detection in Natural Scenes Based on Improved Faster RCNN. **Applied Sciences**, v. 12, n. 12. 6094 p, 2022.

XIE, Saining *et al*. Aggregated Residual Transformations for Deep Neural Networks. **ArXiv**, 2017.

ZAIDI, Syed *et al*. A Survey of Modern Deep Learning based Object Detection Models. **ArXiv**, 2021. Disponível em: https://arxiv.org/abs/2104.11892. Acesso em: 17 mar. 2023.

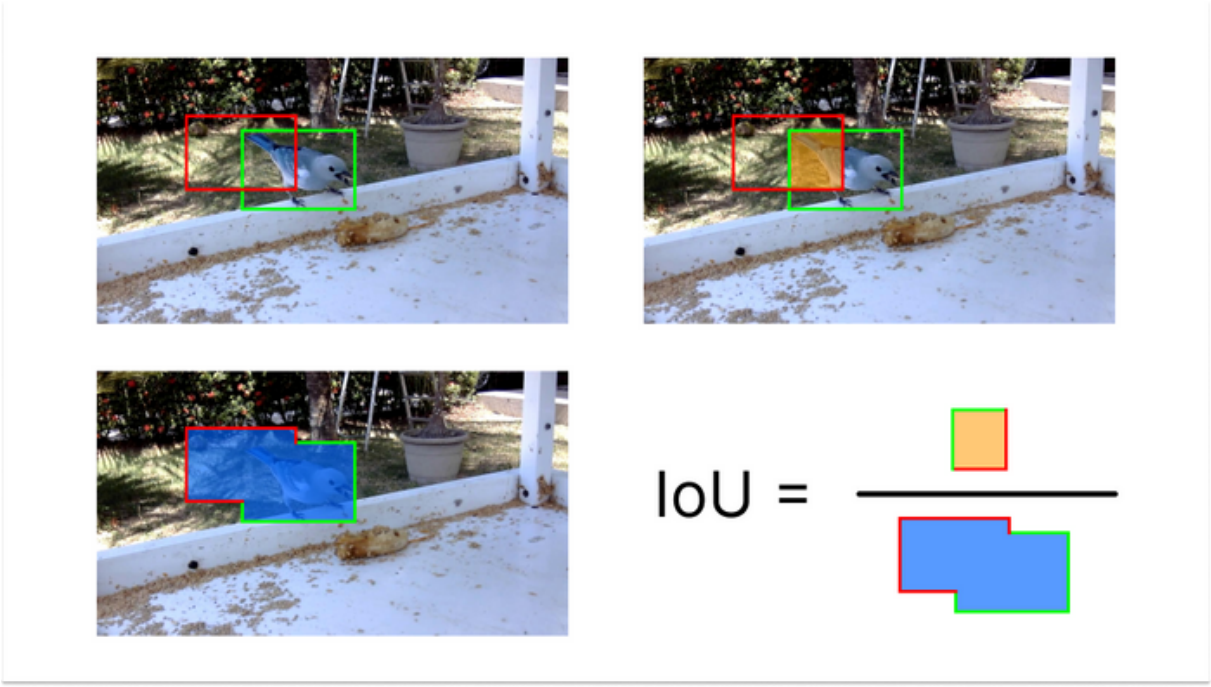
ZHANG, Aston *et al*. **Dive Into Deep Learning**: Interactive Deep Learning Book with Code, Math and Discussions ; Implemented with PyTorch, NumPy/MXNet, and TensorFlow. 2021.

ZOU, Zhengxia *et al*. Object Detection in 20 Years: A Survey. **ArXiv**, 2019. Disponível em: https://arxiv.org/abs/1905.00505. Acesso em: 17 mar. 2023.

APÊNDICE A — EXEMPLIFICAÇÃO DE VP, FP E FN

Neste apêndice, será ilustrado como o cálculo da IoU pode ser realizado, além de apresentar alguns exemplos de detecções que podem ser classificadas como VPs, FPs, e FNs. O emprego da matriz de confusão também será exemplificado.

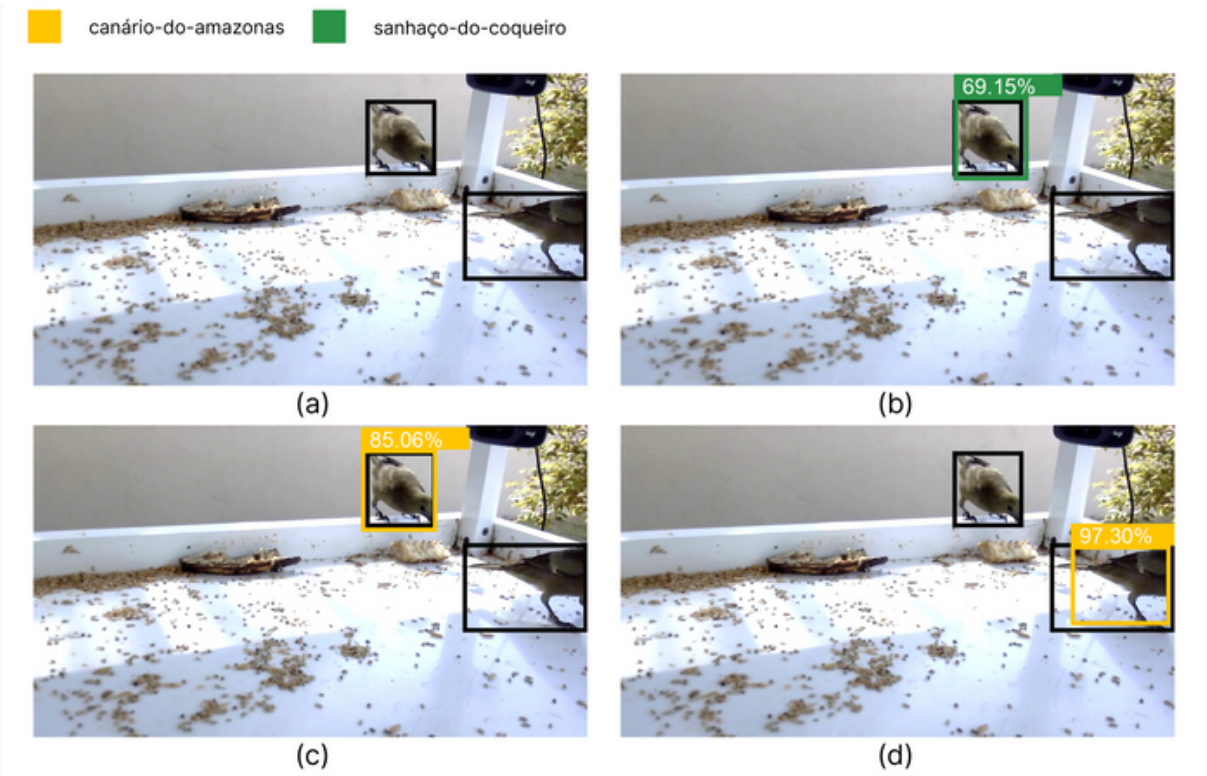
Na imagem abaixo, é possível observar uma caixa delimitadora em verde que representa a anotação de um pássaro, bem como outra em vermelha que ilustra uma detecção. Nesse contexto, a IoU é encontrada por meio da razão entre a área de intersecção, destacada em laranja, pela área de união, realçada em azul. Vale destacar que, exceto na análise da matriz de confusão, a IoU só é calculada se a classe prevista pela detecção for a mesma do objeto em questão.



Com base nisso, uma detecção pode ser definida como um VP caso a caixa delimitadora estimada se sobreponha a de um objeto anotado da mesma classe prevista com um valor de IoU superior ao limiar definido. Se pelo menos uma dessas condições não for satisfeita, então a detecção é classificada como um FP, sendo considerado que o plano de fundo da imagem foi detectado. Por fim, um FN ocorre quando sobre um objeto anotado não houver qualquer detecção VP, isto é, quando ele deixar de ser detectado corretamente.

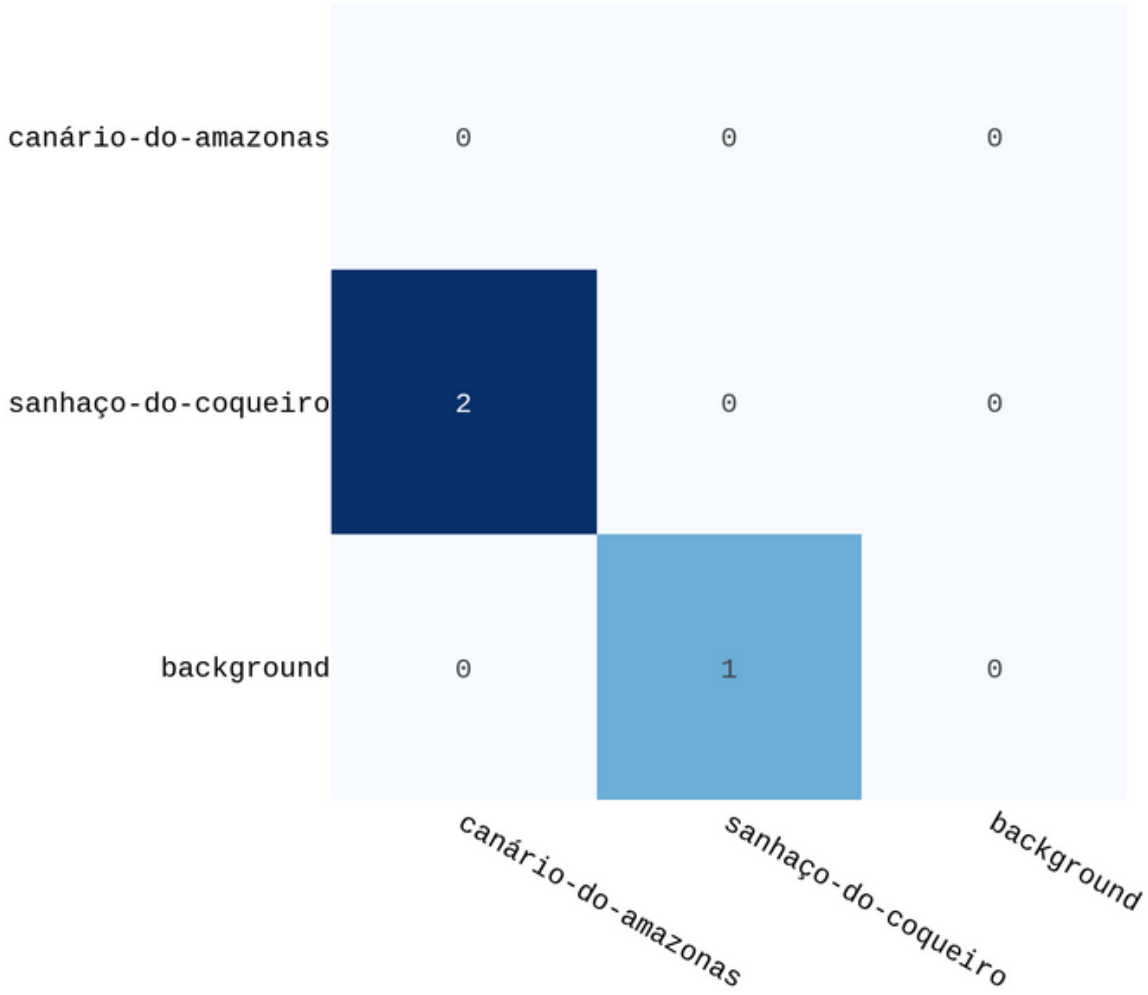
Por exemplo, na imagem a seguir, em (a) dois pássaros da espécie sanhaço-do-coqueiro são anotados com caixas delimitadoras representadas por preto. Em (b), é possível visualizar uma detecção que classifica o objeto corretamente com 69,15% de confiança. Caso a IoU seja maior que o limiar definido, por exemplo de 50%, então ela será considerada um VP.

Por outro lado, em (c), observa-se uma detecção que classifica o objeto incorretamente como canário-do-amazonas. Nesse sentido, a IoU não será calculada e a detecção será considerado um FP. Em (d), ocorre algo semelhante, porém também há a ocorrência de um FN, uma vez que um dos objetos deixou de ser detectado. Assim, ao todo é possível registrar 1 FP, 2 FPs e 1 FN. Em termos de métrica, a precisão nesse caso seria de 33,3%, enquanto a revocação seria de 50%.



Vale ressaltar que a análise é diferente no caso da matriz de confusão, uma vez que é necessário permitir que a IoU seja calculada para detecções de classes distintas também. Diante disso, caso haja um empate, isto é, caso sobre o mesmo objeto ocorram mais de uma detecção, então será considerada aquela com maior nível de confiança.

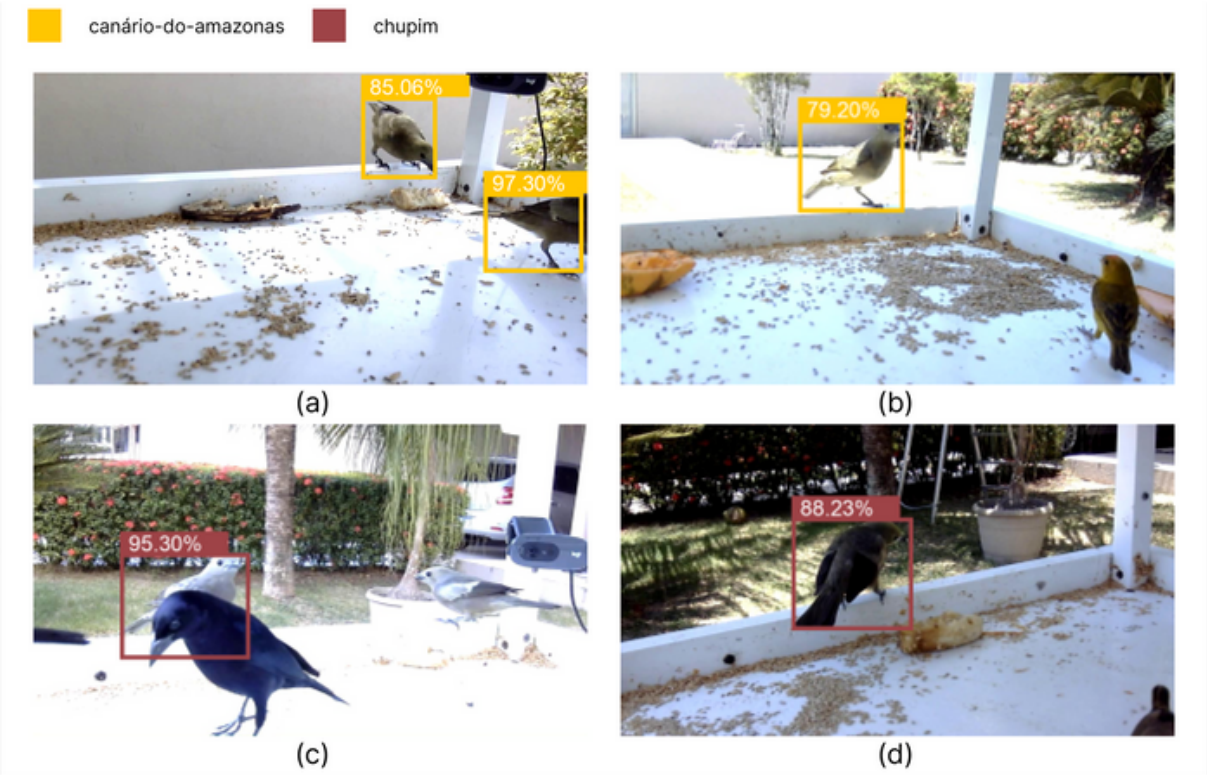
Nesse contexto, nota-se que a detecção em (c) possui um nível de confiança de 85,06%, o que é superior ao daquela apresentada em (b). Logo, para a matriz de confusão, é considerado que a espécie sanhaço-do-coqueiro foi confundida com a canário-do-amazonas em (c), assim como em (d). Além disso, a detecção antes considerada como um VP passa a detectar o plano de fundo da imagem. Dessa forma, a matriz de confusão para esses exemplo é equivalente a da imagem abaixo.


canário-do-amazonas
sanhaço-do-coqueiro
background
0
0
0
2
0
0
0
1
0
canário-do-amazonas
sanhaço-do-coqueiro
background

.

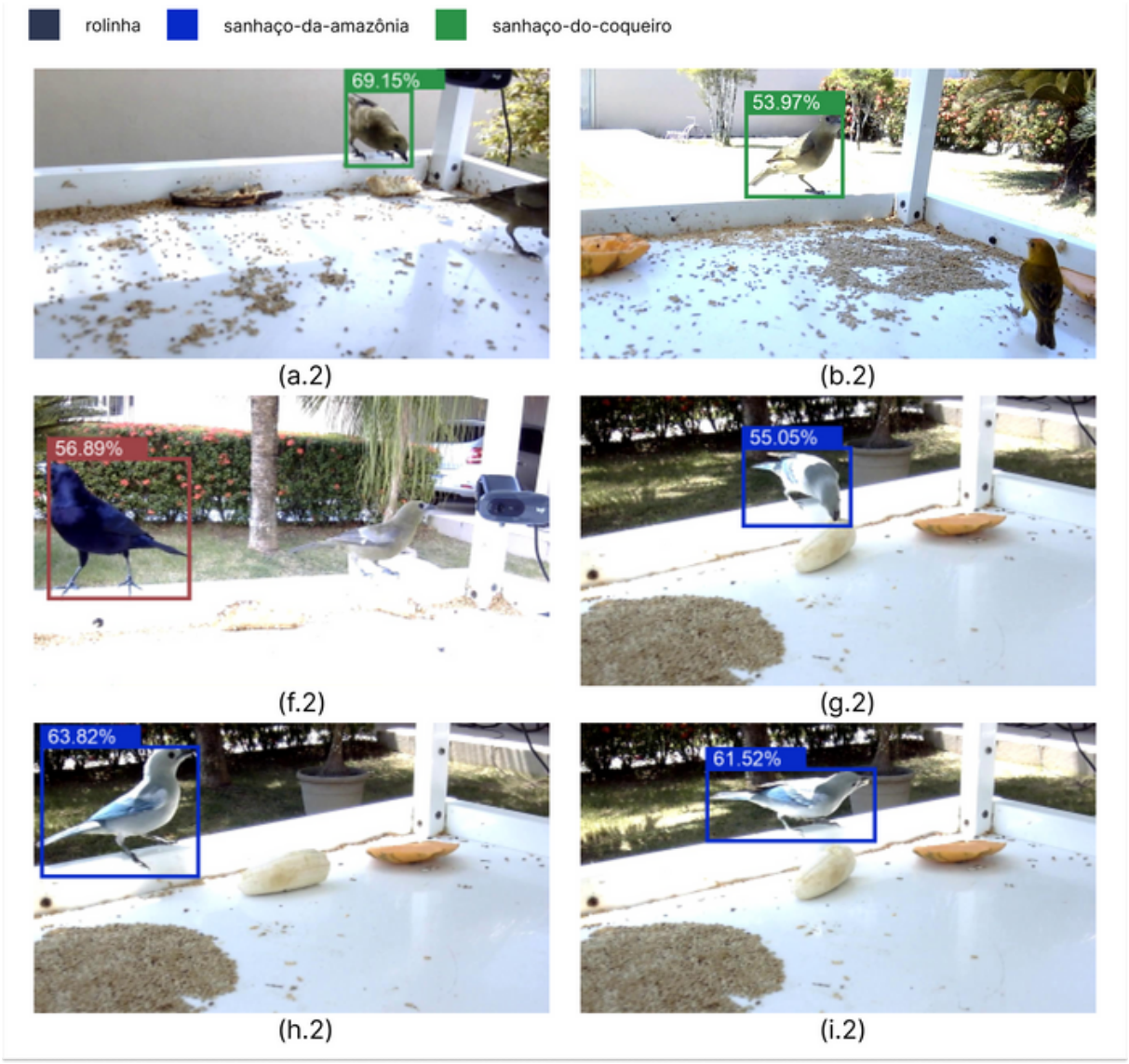
APÊNDICE B — CONFUSÕES DO MODELO DEFINIDO NA FASE DE SELEÇÃO DA BACKBONE

Neste apêndice, serão apresentadas as detecções com maior dificuldade de distinguir as espécies que foram destacadas na matriz de confusão do modelo escolhido na fase de seleção da *backbone*. Nesse sentido, foram contadas 11 detecções ao todo. Por exemplo, na imagem abaixo é possível verificar amostras nas quais integrantes da espécie sanhaço-do-coqueiro foram classificados como canário-do-amazonas e chupim respectivamente.



Na imagem a seguir, nota-se que as amostras (e) e (f) contêm detecções que classificam chupins machos como sanhaço-da-amazônia. Já nas amostras (g), (h) e (i), indivíduos da espécie sanhaço-da-amazônia são reconhecidos como sanhaço-do-coqueiro, o que pode ter se dado pelas semelhanças físicas compartilhadas entre essas espécies. Por fim, em (j), verifica-se a detecção de um canário-do-amazonas como rolinha.

É válido destacar que para alguns indivíduos em (a), (b), (f), (g), (h) e (i) ocorreram detecções simultâneas que os classificaram corretamente, no entanto com nível de confiança menor, conforme colocado na imagem abaixo.

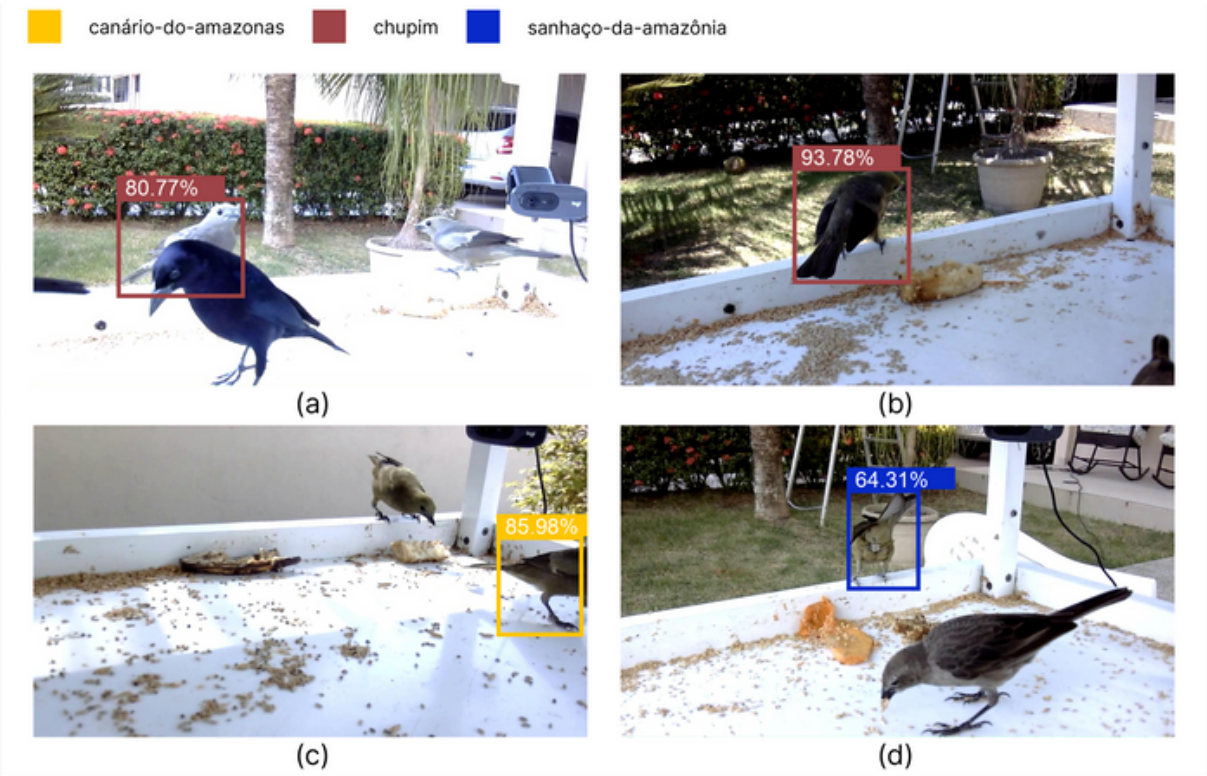

rolinha
sanhaço-da-amazônia
sanhaço-do-coqueiro
69.15%
(a.2)
53.97%
(b.2)
56.89%
(f.2)
55.05%
(g.2)
63.82%
(h.2)
61.52%
(i.2)

.

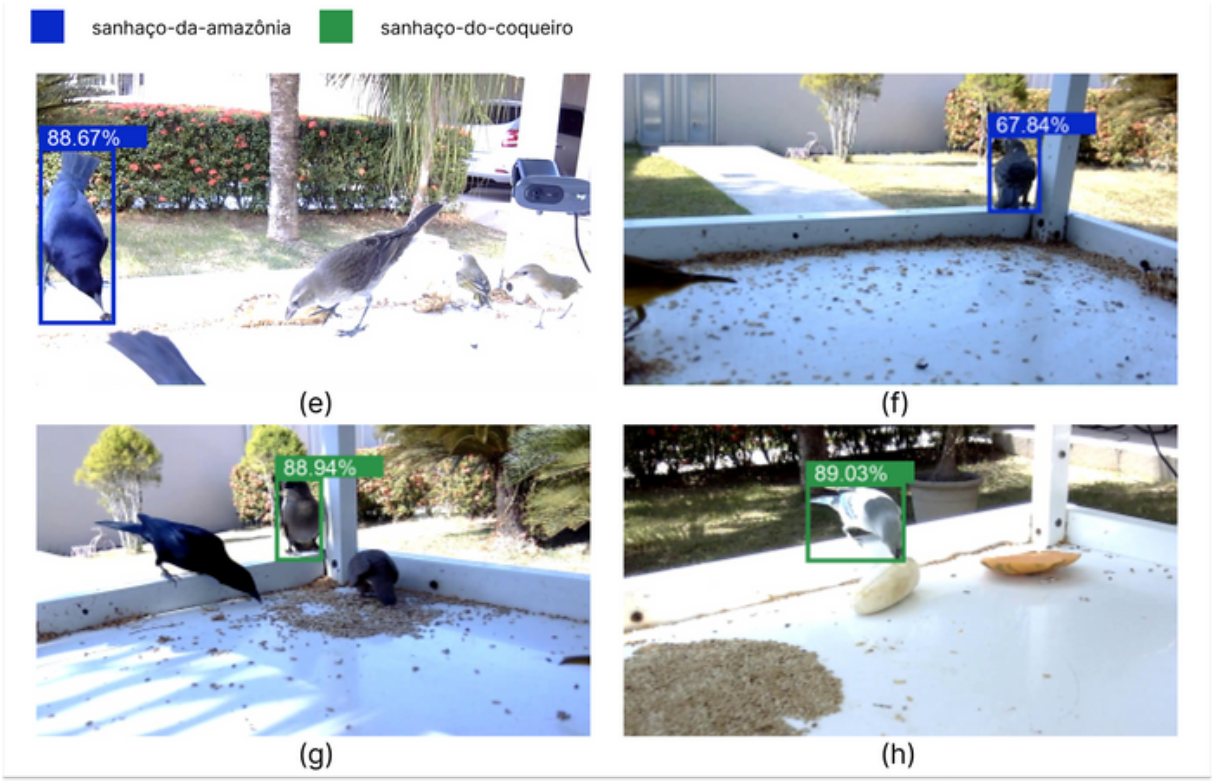
APÊNDICE C — CONFUSÕES DO MODELO DEFINIDO NA FASE DE AJUSTE DE HIPERPARÂMETROS

Neste apêndice, as detecções ressaltadas na matriz de confusão do modelo selecionado na fase de ajuste de hiperparâmetros serão expostas. Nesse sentido, foram totalizadas 8 detecções, 3 a menos do que no modelo anterior, sendo que metade delas persistiram em alguns dos indivíduos das espécies de sanhaço apresentados anteriormente.

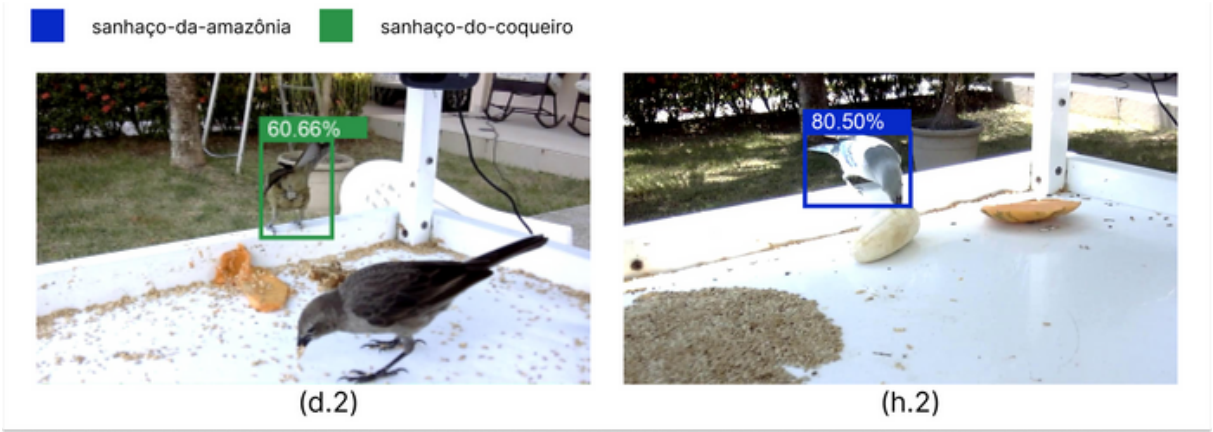
Por exemplo, na imagem abaixo, os mesmos integrantes da espécie sanhaço-do-coqueiro continuam sendo detectados como chupim e canário-do-amazonas em (a), (b) e (c). Por outro lado, em (d), observa-se uma nova detecção na qual ocorre a confusão com sanhaço-da-amazônia.



Já na imagem a seguir, dois chupins são reconhecidos como sanhaço-da-amazônia em (e) e em (f) e outro como sanhaço-do-coqueiro em (g). Por fim, em (h), um pássaro sanhaço-da-amazônia, já apresentado anteriormente, continua sendo detectado como sanhaço-do-coqueiro.



Vale destacar que, conforme colocado na imagem abaixo, para os indivíduos das amostras (d) e (h), ocorreram detecções que os classificaram corretamente, todavia com nível de confiança menor do que as incorretas visualizadas acima.

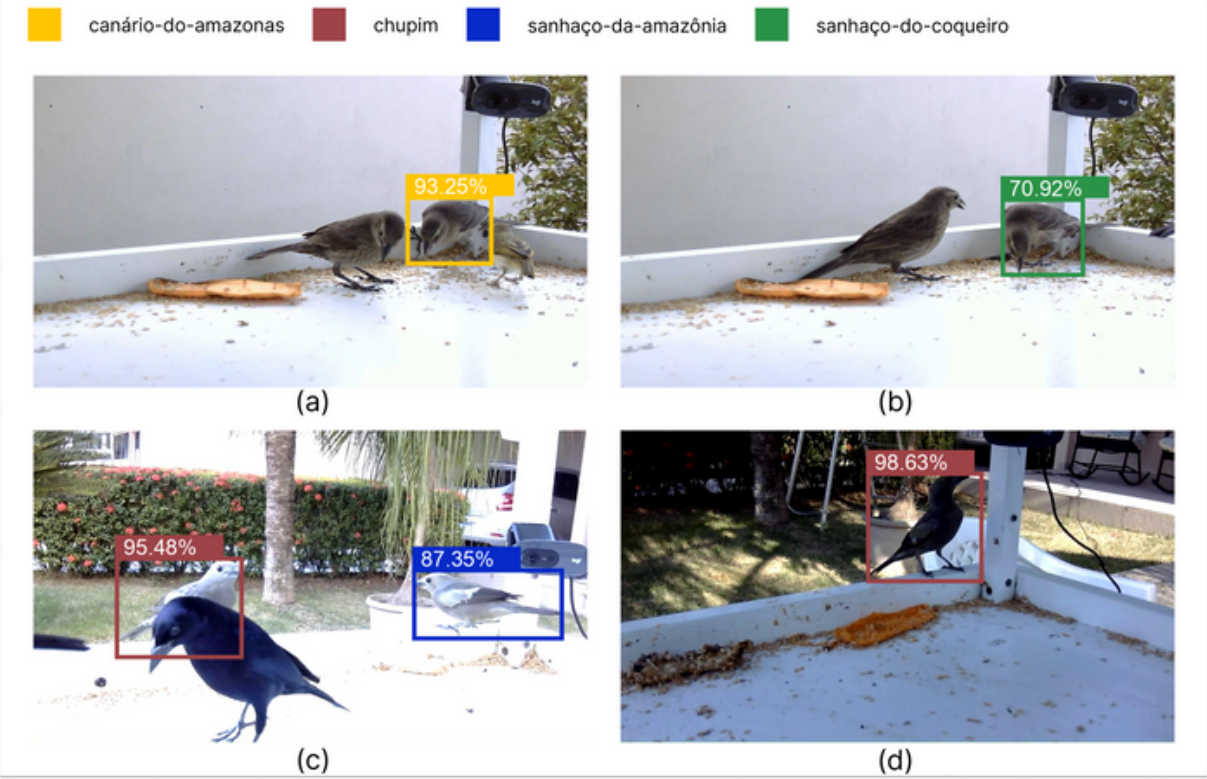


.

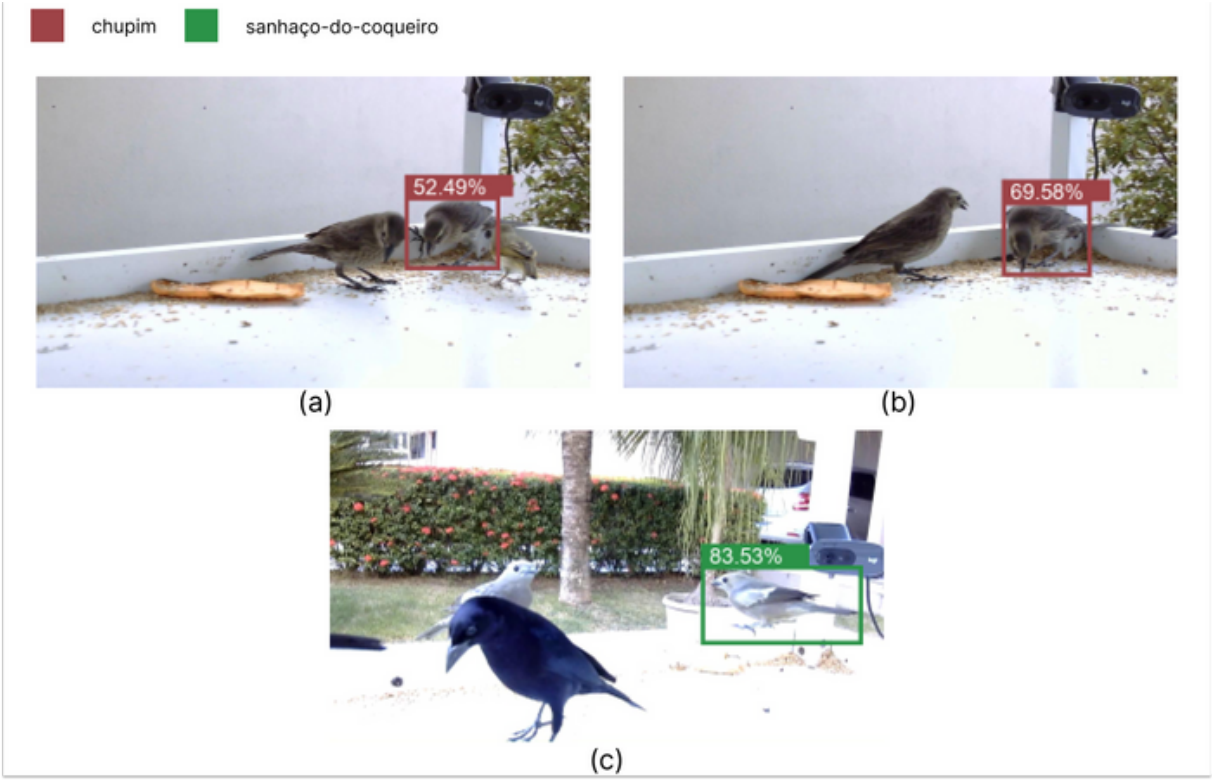
APÊNDICE D — CONFUSÕES DO MODELO DEFINITIVO

Neste apêndice, serão apresentadas as detecções salientadas na matriz de confusão do modelo definitivo treinado com o conjunto de dados total. Foram identificadas 5 confusões no geral. Além disso, vale lembrar que o conjunto de teste empregado para a avaliação desse modelo é maior, contanto com 282 imagens.

Abaixo, é possível identificar que o modelo confundiu 2 chupins com canário-do-amazonas e sanhaço-do-coqueiro nas amostras (a) e (b) respectivamente. Por outro lado, 2 sanhaços-do-coqueiro são confundidos com chupim e sanhaço-da-amazônia em (c).



Assim como nos exemplos anteriores, são observadas detecções que classificam os indivíduos corretamente, no entanto com valores inferiores do que as incorretas em relação ao nível de confiança.


chupim
sanhaço-do-coqueiro
52.49%
69.58%
(a)
(b)
83.53%
(c)

.