APLICAÇÃO DE REDES NEURAIS CONVOLUCIONAIS PARA A DETECÇÃO DE FACES DE BUGIOS-RUIVO

Maik Henrique Carminati, Andreza Sartori – Orientadora, **Julio Cesar de Souza Junior** – Co-Orientador

Curso de Bacharel em Ciência da Computação
Departamento de Sistemas e Computação
Universidade Regional de Blumenau (FURB) – Blumenau, SC – Brasil

mcarminati@furb.br, asartori@furb.br, juliosouza@furb.br

Resumo: *O Bugio-Ruivo é uma espécie de primatas nativos da mata atlântica ameaçada de extinção. Embora muitos estudos sobre ecologia e comportamento da espécie venham sendo realizados no Brasil, pouco é produzido de novas tecnologias para a marcação e identificação de indivíduos. Com isso, este trabalho busca realizar o reconhecimento facial de bugios-ruivo em cativeiro e em seu habitat natural, auxiliando assim os veterinários e pesquisadores do Projeto Bugio-FURB. Para isso foi utilizado a Rede Neural Convolucional YOLOv4, que possui uma alta capacidade de reconhecimento de objetos comparado a versões anteriores dele. Foi estruturado uma base de dados contendo 6017 imagens de primatas separados em cinco classes: bugio-ruivo, bugio-manto, macaco-pata, humano e chimpanzé. Os testes foram realizados de forma manual carregando imagens distintas e comparando seus resultados. O modelo treinado alcançou uma precisão média de 98,56% para a identificação de indivíduos bugios-ruivo. Uma maior precisão foi obtida em imagens de boa qualidade, com um ambiente controlado e com a face do primata mais próxima e totalmente visível.*

Palavras-chave*: Bugio-ruivo. Projeto bugio-FURB. Rede Neural Convolucional. YOLOv4.*

1 INTRODUÇÃO

O *Alouatta Guariba Clamitans*, também conhecido como bugio-ruivo, guariba ou barbado, pertence à família dos primatas, e podem viver em média de 15 a 20 anos. Quando atingem seu estado adulto podem pesar uma média de 6kg variando de macho para fêmea. O comprimento do seu corpo pode alcançar 1,18 metros e sua cauda aproximadamente 45 centímetros. Ele é encontrado ao longo da Mata Atlântica em alguns estados brasileiros como Santa Catarina, Espírito Santo e Rio Grande do Sul, como também em algumas regiões da Argentina. A população atual dos bugios-ruivo não é conhecida, mas estima-se que esteja próxima dos 10.000, que vivam em populações com uma média de 500 indivíduos, e que sua tendencia populacional esteja em declínio (BICCA-MARQUES, 2015).

As principais ameaças identificadas para o bugio-ruivo são a agricultura, pecuária, expansão urbana, epidemias, desmatamento, redução de habitat e caça, pois interferem diretamente em seu habitat natural. Animais domésticos como os cachorros também podem apresentar motivos de preocupação quando os bugios se encontram perto de pequenas reservas florestais próximas de áreas urbanas, bem como redes de energia elétrica próximas às florestas (BICCA-MARQUES, 2015). Buss (2012) aponta que a febre amarela também foi um fator que afetou drasticamente algumas populações de bugios em certos estados brasileiros e em determinadas regiões, causando a extinção local da espécie dos bugios-ruivo e Bicca-Marques (2015) complementa afirmando que esses primatas têm um nível de extinção global definido como quase ameaçada, enquanto no Brasil é caracterizada como vulnerável (BICCA-MARQUES, 2015).

A produção de estudos sobre a ecologia e comportamento como também o monitoramento de populações de bugios têm grande importância para a criar estratégias de conservação para a espécie. Crouse *et al.* (2017) aponta que grande parte das pesquisas sobre populações de animais requerem a captura e marcação desses indivíduos utilizando colares coloridos ou suas próprias características naturais. Esses métodos podem ou não permitir a identificação dos bugios em datas futuras e que, para ter um melhor resultado em estudos da ecologia desses animais, é necessária a coleta de dados por um longo período.

O Projeto Bugio, criado em 1991 pela FURB, tem como objetivo principal estudar o comportamento e a ecologia do bugio-ruivo em seu habitat natural, como também a conservação dessa subespécie. O projeto conta com atividades de acompanhamento de bugios em vida livre e sob cuidados humanos. Quando capturados, eles são levados para a sede do projeto e são microchipados, permanecendo no estabelecimento até que tenham se recuperado e que suas condições sejam apropriadas para a sobrevivência na floresta. Em alguns casos, quando capturados e sedados para a implantação do microchip, esses bugios acabam ficando com sequelas permanentes como cicatrizes e feridas, impactando assim diretamente na sua reabilitação (FURB, 2001).

Conforme relato dos médicos veterinários responsáveis pelo Projeto Bugio-FURB, uma forma de resolver este problema seria realizando o reconhecimento facial dos bugios, pois estes animais apresentam características faciais singulares. Segundo Abreu (2021), para resolver os problemas de reconhecimento facial, diversas técnicas foram desenvolvidas ao longo dos anos, inicialmente baseando-se em modelos de Aprendizado de Máquina, até as mais atuais,

relacionadas à Redes Neurais Artificias (RNA). Algumas das técnicas mais atuais utilizam Redes Neurais Convolucionais para permitir a segmentação de instância, na qual possibilita separar cada objeto de seu fundo e formar segmentos a nível de pixel, como é o caso do Mask-R CNN (MEIRA, 2020). Outra técnica atual, conhecida como Yolo (You Only Look Once), é um método de detecção de objetos de passada única que utiliza uma Rede Neural Convolucional para extrair as características das imagens (ALVES, 2020). Krause (2019) que desenvolveu um protótipo para reconhecimento facial de bugios-ruivo por meio de Redes Neurais Convolucionais, executou testes com três RNAs diferentes, sendo elas, a Ineception-ResNet v2, ResNet 50 e Xception.

A fim de dar continuidade ao trabalho de Krause (2019), que criou uma base de dados e realizou o reconhecimento facial de bugios-ruivo para o Projeto Bugio-FURB, este protótipo visa aprimorar a identificação facial de bugios-ruivo, sem a necessidade de realizar uma marcação manual nas imagens para executar os testes. Além disso, a base de dados foi ampliada para 6017 imagens, aumentando o número de bugios-ruivo e inserindo outros tipos de primatas como exemplos negativos, que também foram utilizados para os testes. Com isso, o trabalho desenvolvido melhorou a velocidade que a face do bugio é marcada e que pode ser enviada para o protótipo de reconhecimento.

2 FUNDAMENTAÇÃO TEÓRICA

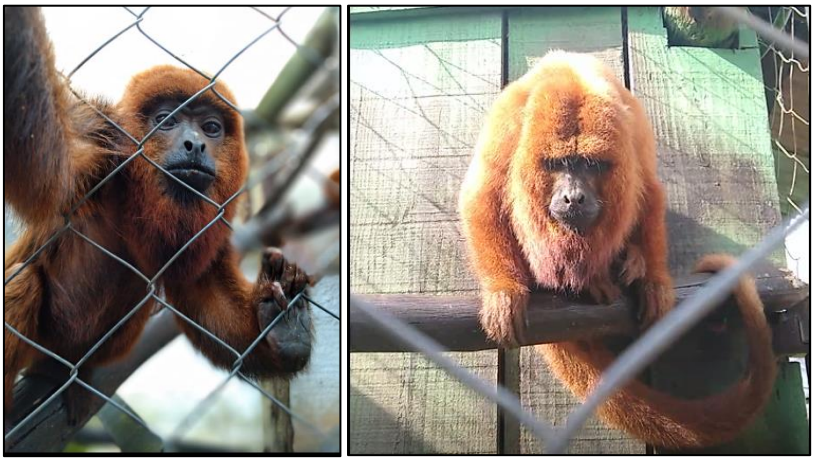
Neste capítulo são apresentados os principais assuntos relacionados as características físicas dos Bugios-Ruivo, bem como as Redes Neurais Artificiais (RNA). Na seção 2.3 é apresentado o protótipo atual desenvolvido por Krause (2019) de reconhecimento facial dos bugios-ruivo. Ao final, são descritos os trabalhos correlatos a este protótipo.

2.1 BUGIO-RUIVO E PROJETO BUGIO - FURB

Os bugios, conhecidos popularmente como guaribas, barbudos e roncadores, são primatas da família *Alouatta* e possuem a maior distribuição geográfica na região neotropical, ocorrendo desde o México até a Argentina. Uma característica marcante do grupo é a cor intensa da pelagem, que varia entre o marrom e o ruivo. Outras características que se destacam são o formato dos dentes, desenvolvidos para dietas ricas em vegetais como também uma cauda capaz de suportar o peso do próprio corpo. Dentre os bugios o Alouatta guariba Clamitans, popularmente conhecido como o bugio-ruivo é um dos ameaçados de extinção e está marcado como vulnerável (AMDA, 2019*).*

O bugio-ruivo possui um peso médio entre 5 e 12kg, atinge a maturidade sexual entre 3 e 5 anos, variando de macho para fêmea e a longevidade da espécie é de aproximadamente 15 a 20 anos. Eles apresentam dimorfismo sexual, sendo os machos adultos maiores que as fêmeas em todas as características corporais e dimorfismo por dicromatismo, sendo os machos adultos com pelagem ruiva e as fêmeas e filhotes com pelagem castanho escuro. A Figura 1 apresenta ao lado esquerdo a imagem de um bugio fêmea e um macho ao lado direito. A característica mais marcante desse animal é a presença do osso hioide bastante desenvolvido com o corpo central oco, formando uma câmara de ressonância de som, produzindo o ronco ou rugido característico desta subespécie (FURB, 2001).

Figura 1 – Dimorfismo sexual e dimorfismo por dicromatismo



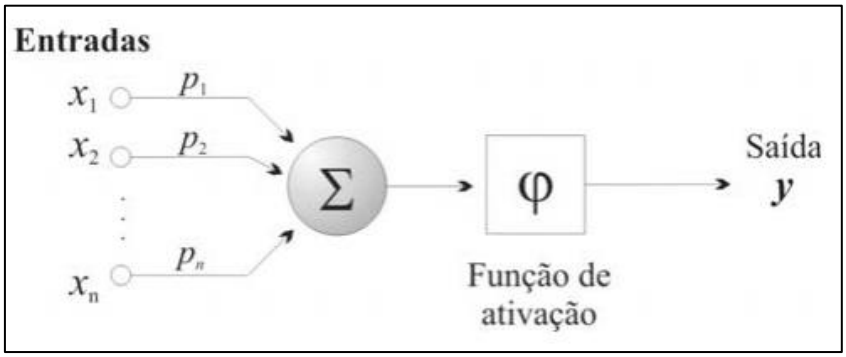
Fonte: Krause (2019).

O Projeto Bugio foi criado em 1991 pela FURB, localiza-se na cidade de Indaial/SC e tem como missão treinar estudantes e profissionais para desenvolverem atividades na área de primatologia e gerar conhecimento sobre os bugios-ruivo que auxiliem em ações de conservação da subespécie e de seu habitat. O projeto também realiza todo o tratamento e cuidados com os bugios, sendo que muitas vezes recebem esses animais de ocorrências atendidas pela Polícia Militar Ambiental e os mantém sob cuidados humanos até que suas condições para a sobrevivência na floresta sejam apropriadas (FURB, 2001).

2.2 REDES NEURAIS ARTIFICIAIS (RNA)

Uma RNA é um sistema desenvolvido para modelar por meios de modelos matemáticos a maneira de como o cérebro realiza tarefas (FLECK *et al.* 2016). A Figura 2 representa uma RNA com um conjunto de n conexões de entrada, seguidos por seus pesos que podem ser negativos ou positivos. Em seguida, apresenta um somador para acumular os sinais de entrada multiplicados pelo respectivo peso e por fim uma função de ativação que limita o intervalo permissível de amplitude do sinal de saída a um valor fixo (FERNEDA, 2006). Em suma, a RNA é composta por nós ou unidades conectadas por ligações direcionadas, essas ligações servem para propagar a ativação de um nó para outro sendo que cada nó irá calcular uma soma ponderada de suas entradas. As RNAs são capazes de realizar muitas tarefas complexas de aprendizagem, embora necessite de grande esforço para obter a estrutura da rede correta e alcançar uma convergência para algo próxima ao global. Isso significa que, a convergência será o menor ou o maior valor possível para a função, na qual o valor foi atribuído e que não viole nenhuma restrição (RUSSELL; NORVING, 2013).

Figura 2 – Modelo matemático de um neurônio artificial



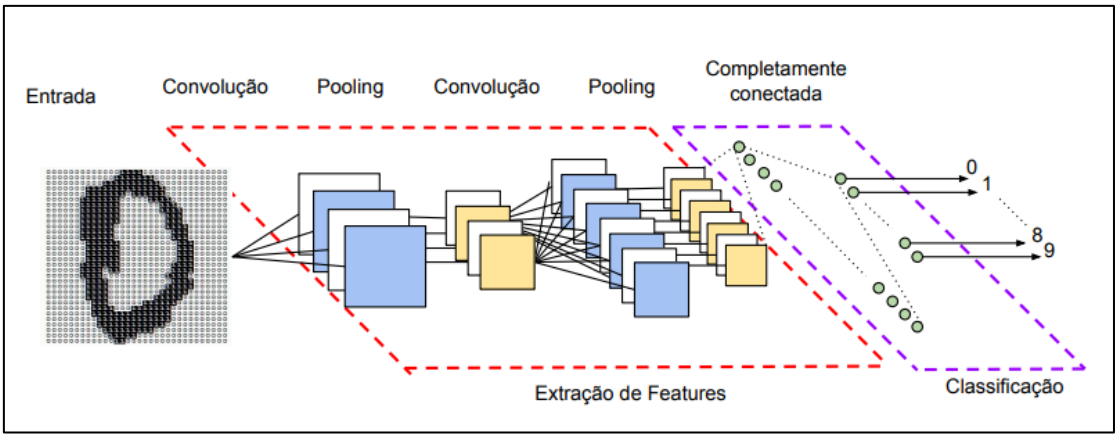
Fonte: Ferneda (2006).

2.2.1 Rede Neural Convolucional

As Redes Neurais Convolucionais (Convolutional Neural Networks - CNN), são um tipo de RNA inspirada no processo biológico. Além de ser conhecida por gerar modelos mais precisos, as CNNs tendem a ser computacionalmente mais eficientes que modelos totalmente conectados, pois exigem menos parâmetros e são mais fáceis de paralelizar com uma unidade de processamento gráfico (Graphics Processing Units - GPU). Uma CNN além de poder ser utilizada para detecção de imagens, serve também para estruturas de sequência unidimensionais, como áudio, texto e análise de series temporais que é um conjunto de observações feitas em sequência ao longo do tempo. As denominadas camadas convolucionais podem também ser chamadas de correlações cruzadas (ZHANG *et al.* 2021).

Uma Rede Neural Convolucional, como apresentado na Figura 3, consiste em várias camadas com funções distintas. Para realizar a extração de características, inicialmente é comum aplicar o dado de entrada sobre camadas de convolução, que são compostas por diversos neurônios. Outra camada importante utilizada após as camadas de convolução é a camada de agrupamento (*pooling*), que possui a função de reduzir a dimensionalidade dos dados da rede. O resultado da repetição das camadas de convolução e agrupamento é um conjunto de características especializado na tarefa em que ela foi treinada. Quando se deseja realizar uma classificação, é adicionado uma camada totalmente conectada após a última camada de convolução e agrupamento. Essa camada é responsável por definir um caminho de decisão a partir dos resultados obtidos dos filtros das camadas anteriores. A última camada é a função de classificação, que é fundamental no treinamento, pois influencia no aprendizado dos filtros e no resultado da rede. (VARGAS; CARVALHO; VASCONCELOS, 2016).

Figura 3 – Arquitetura da Rede Neural Convolucional



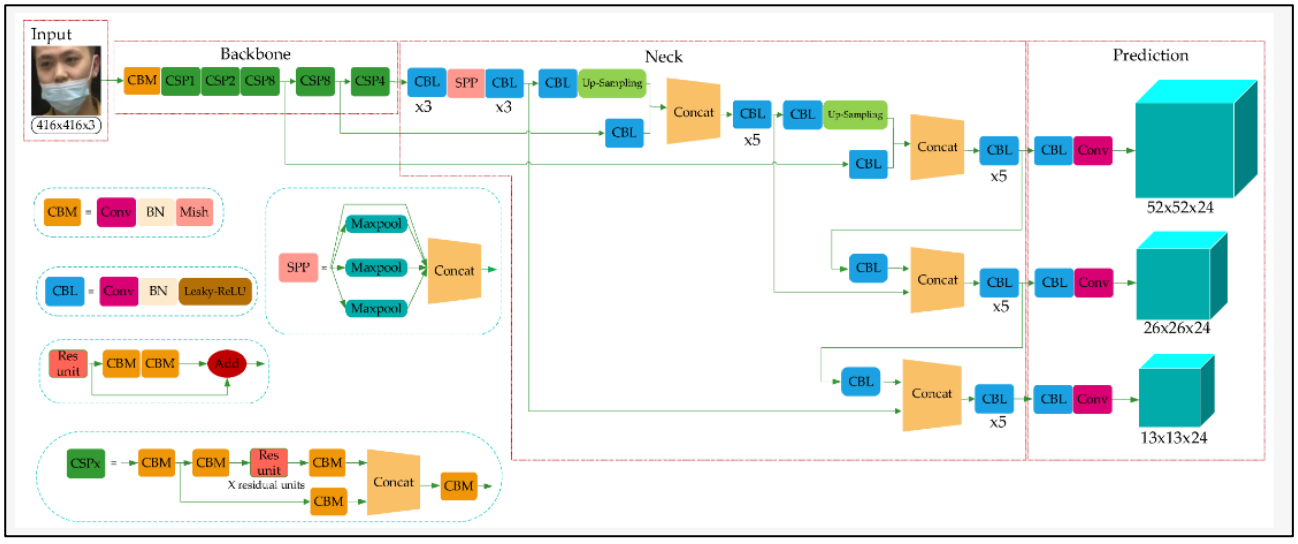
Fonte: Vargas, Carvalho, Vasconcelos (2016).

2.2.2 YOLOv4: You Only Look Once

O YOLOv4 (You Only Look Once), é um método de detecção de objetos de passada única que utiliza uma CNN para extrair características. Essa CNN conforme apresentado na Figura 4, consiste em um `Backbone`, `Neck` e um `Head (Prediction)` que atuam em tempo real e realiza o treinamento em uma única Unidade de Processamento Gráfico (GPU) convencional (BOCHKOVSKIY; WANG; LIAO, 2020). O `Backbone` utiliza a CNN CSPDarknet53 como extrator de características. O Spatial Pyramid Pooling (SPP) e o método Path Aggregation Network (PAN) são as camadas mais relevantes que compõem o `Neck`. O bloco SPP sobre o CSPDarknet53 aumenta significativamente o campo receptivo e separa as características de contexto mais significativas, sem causar redução significativa da velocidade. Já o PAN é o método de segmentação e agregação de parâmetros do `backbone` responsável criar atalhos entre as camadas para diferentes níveis do detector. A `Head` é composta pela PAN e versão 3 do YOLO, baseado em âncoras, que utiliza classificadores lógicos independentes, gerando a arquitetura do YOLOv4 (LEOCÁDIO *et al.* 2021).

Atualmente, algoritmos de detecção de objetos são divididos em duas categorias. A primeira é baseada no R-CNN. A segunda é o algoritmo firmado em One-Stage baseado na arquitetura *Single Shot Detector* (SSD) e YOLO, que possui alto desempenho na detecção de objetos multi-escala em tempo real. Combinado com a estrutura da ResNet, o YOLOv3 integrou o módulo residual em si mesmo e obteve o Darknet53. Com base nisso, o YOLOv4 construiu o CSPDarkNet53 aproveitando as características de aprendizado da Cross-Stage Partial Network (CSPNet) (YU; ZHANG, 2021).

Figura 4 – Arquitetura da Rede Neural Artificial YOLOv4



Fonte: Yu; Zhang (2021).

2.3 RECONHECIMENTO FACIAL DE BUGIOS-RUIVO ATRAVÉS DE REDES NEURAIS CONVOLUCIONAIS

O trabalho de Krause (2019), desenvolveu um protótipo para reconhecimento facial de bugios-ruivo por meio de Redes Neurais Convolucionais. Para isso foram feitos testes com três tipos de Redes Neurais Convolucionais, sendo elas a Ineception-ResNet v2, ResNet 50 e Xception. A técnica Triplet Loss foi aplicada nos modelos utilizados de RNA para

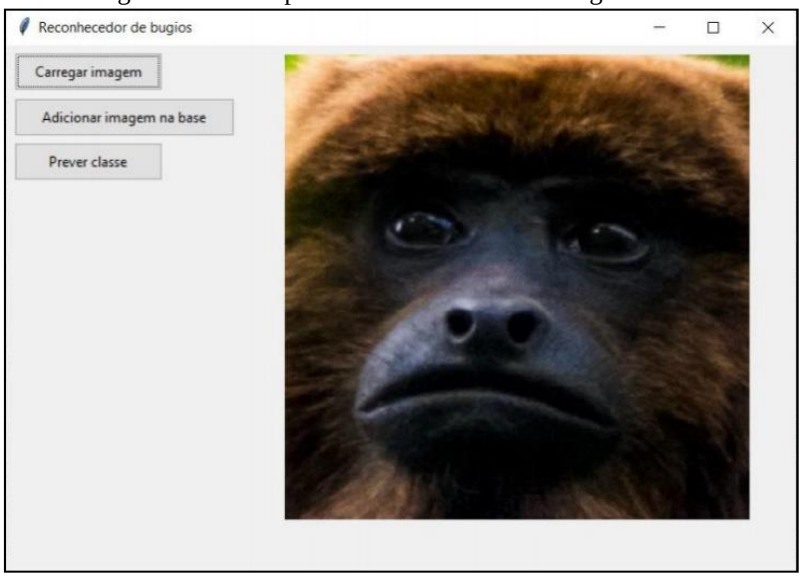
treiná-las nas extrações de características, enquanto para a classificação foram utilizados os algoritmos de Aprendizagem de Máquina K-Nearest Neighbors (KNN), Recurrent Neural Network (rNN), Support Vector Machine (SVM).

Para a execução do protótipo, foram selecionadas algumas imagens que foram redimensionadas para que pudessem ser utilizadas para o treinamento da RNA. Então, foi construída a Rede Neural Convolucional, na qual foram utilizadas as redes Inception ResNet v2, ResNet50 e Xception, que depois de treinadas, foram extraídas as características relevantes para realizar o treinamento de um classificador. Com o classificador treinado, foi possível começar a realizar as previsões sobre as imagens. Para aumentar a eficiência do protótipo, foi utilizado a estratégia de Data Augmentation, que utiliza a translação, rotação, modificação de perspectiva e outras combinações de operação a fim de aumentar a quantidade das imagens (MELO, 2019). Com isso, a partir de cada imagem foram geradas outras nove, totalizando uma base de dados com 1790 imagens (KRAUSE, 2019).

Para realizar o treinamento da rede, a base de dados foi dividida em duas partes, na qual 85% das imagens foram utilizadas para o treinamento e 15% para a validação. As imagens foram divididas em 40 lotes e cada lote possuí 64 imagens. Para resolver alguns problemas que surgiram durante o treinamento, foi criado um gerador de lotes, para que as classes dentro de um lote ficassem organizadas (KRAUSE, 2019).

Para extrair os melhores resultados dentre as redes treinadas, as redes Xception, Inception-ResNet v2 e a ResNet50 receberam as mesmas configurações. A ResNet50, obteve os melhores resultados. A rede Xception apresentou melhor performance no treinamento e poderia ter sido treinada com uma quantidade menor de imagens que as outras redes utilizadas, para alcançar o mesmo resultado. O trabalho apresentou um protótipo para o reconhecimento facial de bugios-ruivo e mostrou que seu objetivo de fazer o reconhecimento facial foi atingido com uma acurácia de 99,94%. A função Triplet Loss desempenhou um papel importante no treinamento, pelo fato de comparar a âncora com um exemplo positivo e outro negativo, fazendo a rede aprender com menos iterações. Por fim, foi desenvolvida uma aplicação desktop para exibir o protótipo e demonstrar suas funcionalidades. A aplicação apresentada na Figura 5 possui três funcionalidades principais: carregar a imagem, prever a classe e adicionar uma nova imagem à base (KRAUSE, 2019).

Figura 5 – Protótipo de reconhecimento de Bugios-Ruivo



Fonte: Krause (2019).

2.4 TRABALHOS CORRELATOS

Neste capítulo serão apresentados três trabalhos correlatos, que possuem características semelhantes à proposta deste projeto. O Quadro 1 descreve o trabalho de Schofield *et al*. (2019), que consiste no reconhecimento facial de chimpanzés utilizando aprendizagem profunda a partir de vídeos. O segundo trabalho de Sinha *et al*. (2018) apresentado no Quadro 2, explora as tendências da detecção facial e reconhecimento de primatas. Por fim, o Quadro 3 apresenta o trabalho de Chen *et al*. (2020), que aborda um estudo sobre o reconhecimento facial do panda gigante com base em imagens.

Quadro 1 – Chimpanzee face recognition from videos in the wild using deep learning

Referência	Shofield *et al*. (2019)
Objetivos	Reconhecimento Facial de Chimpanzés
Principais funcionalidades	Foram utilizadas técnicas de Aprendizagem Profunda para detectar, rastrear e reconhecer chimpanzés. A base de dados é composta por aproximadamente 50 horas de vídeos contendo 10 milhões de detecções faciais.

Ferramentas de desenvolvimento	Para a detecção facial foi utilizado um detector profundo de disparo único com uma arquitetura VGG-16. Para o rastreamento foi utilizada a Kanade-Lucas-Tomasi (KLT) que agrupa os rostos pertencentes ao mesmo indivíduo e para o reconhecimento utilizou-se uma variante da VGG-M com 25 classes de reconhecimento de identidade.
Resultados e conclusões	Detecção de faces alcançou 81% de precisão média, o modelo de reconhecimento obteve uma precisão geral de 92,27% e para o sexo do chimpanzé 96,16%.

Fonte: elaborado pelo autor.

O trabalho de Shofield *et al.* (2019) se relaciona com o objetivo deste trabalho pois apresenta uma abordagem automatizada para a coleta de dados da espécie de primatas *Pan Troglodytes Verus* popularmente conhecidos como chimpanzés. Se difere também do trabalho atual, pois tem o objetivo de reconhecer os chimpanzés e definir seu gênero. O trabalho utiliza técnicas de Aprendizado Profundo para detectar, rastrear e reconhecer os chimpanzés. Para treinar o detector de chimpanzés, 3707 imagens foram marcadas manualmente com uma caixa delimitadora ao redor da face. Para o reconhecimento facial, as imagens foram rotuladas com a identidade do chimpanzé e utilizadas para treinar o identificador de identidade e gênero.

Quadro 2 – Exploring Bias in Primate Face Detection and Recognition

Referência	Sinha *et al*. (2018)
Objetivos	Detectar e reconhecer primatas de várias espécies
Principais funcionalidades	Foi utilizado Aprendizagem Profunda para detectar e reconhecer os tipos de primatas. A base de dados foi criada a partir de três conjuntos de dados que variavam entre 1400 e 3000 imagens cada.
Ferramentas de desenvolvimento	Para a detecção facial foi utilizado o Tiny Faces junto com a ResNet10. Para o reconhecimento foram utilizados a Análise de Componentes Principais (PCA), a Análise Discriminante Linear (LDA), o VGG-Face e o VGG-Face em conjunto com o Finetuning.
Resultados e conclusões	Os resultados da detecção facial ficaram com uma média de 85,58% de precisão. O reconhecimento obteve um maior percentual de assertividade, na qual em alguns casos chegou a 100%.

Fonte: elaborado pelo autor.

O trabalho de Sinha *et al*. (2018) também se relaciona com esse trabalho pois tem o objetivo de detectar e reconhecer diversos tipos de primatas utilizado Aprendizagem Profunda. A detecção facial foi implementada utilizando o Tiny Faces com uma abordagem de duas etapas, a primeira para detectar os olhos e a segunda para detectar a face. Utilizou também uma rede ResNet101 com modelos pré-treinados na base de dados da ImageNet. Para o reconhecimento facial dos primatas Sinha *et al.* (2018) utilizou Análise de Componentes Principais (PCA) para condensar a informação, a Análise Discriminar Linear (LDA) para separar os objetos da imagem. O VGG-Face que é uma arquitetura de reconhecimento facial já treinada com rostos humanos também foi utilizado juntamente com o VGG-Face com Finetuning para ajustar os dados já treinados.

Quadro 3 – A study on giant panda recognition based on images of a large proportion of captive pandas

Referência	Chen *et al*. (2020)
Objetivos	Reconhecer pandas gigantes de forma menos abusiva
Principais funcionalidades	Foi utilizado uma base de dados com 6441 imagens com uma CNN para realizar a tarefa de detecção e reconhecimento facial.
Ferramentas de desenvolvimento	Para a detecção facial foi utilizado o Faster R-CNN e o ResNet-50, seguido da segmentação e alinhamento que utilizaram uma ResNet-50 modificada. Para o reconhecimento foram utilizados a ResNet-101, ResNet-50 e a ResNet-18.
Resultados e conclusões	Como foram utilizadas apenas imagens frontais, a detecção de faces obteve uma precisão de 100% e os resultados dos modelos de classificação foram de 95.53% para o ResNet-101, 96.27% para o ResNet-50 sendo o mais preciso, e 95.02% para o ResNet-18.

Fonte: elaborado pelo autor.

O trabalho de Chen *et al.* (2020) relaciona-se com o trabalho atual pois utiliza de CNNs para reconhecer pandas gigantes de uma forma menos abusiva e totalmente automática. Porém neste correlato as imagens obrigatoriamente devem possuir toda a face do panda, diferente do trabalho atual que não necessita dessa característica. Foi realizado a segmentação da imagem e os pixels que não fazem parte da face do panda foram preenchidos com cor preta, após a segmentação foi feita um alinhamento da imagem e por fim foi utilizado a CNN para identificar a identidade do panda na imagem de entrada.

3 DESCRIÇÃO DO PROTÓTIPO

Neste capítulo serão descritos os aspectos mais relevantes de especificação e implementação para a compreensão do protótipo para a detecção facial de bugios-ruivo. São abordados os requisitos do protótipo, seguidos de sua especificação e desenvolvimento da CNN.

3.1 REQUISITOS

Nesta seção são apresentados os Requisitos Funcionais (RF) e o Requisitos Não Funcionais (RNF) do protótipo desenvolvido respectivamente nos quadros Quadro 4 e Quadro 5.

Quadro 4 – Requisitos Funcionais

RF01	detectar e marcar a face do primata
RF02	permitir selecionar imagem a ser detectada
RF03	exibir resultado da classificação

Fonte: elaborado pelo autor.

Quadro 5 – Requisitos Não Funcionais

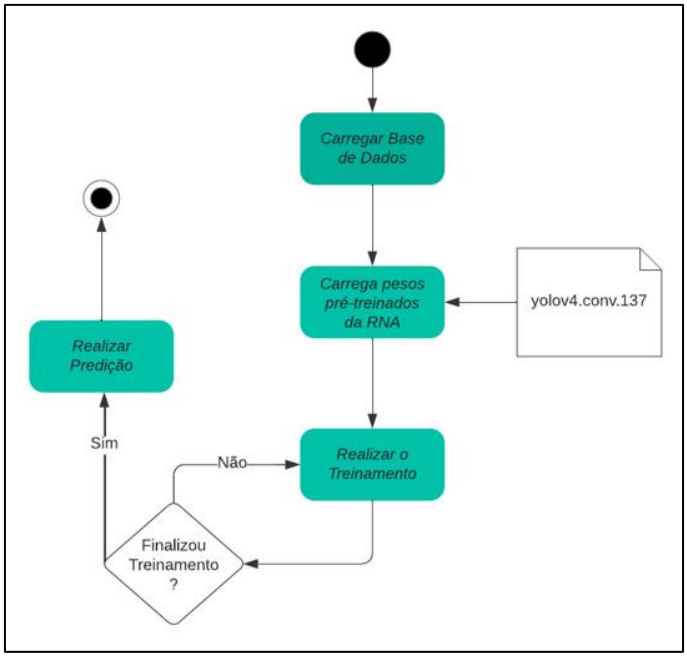
RNF01	ser implementado na linguagem Python no ambiente de desenvolvimento Google Colab para realizar o treinamento
RNF02	utilizar YOLOv4 para treinar o modelo de detecção
RNF03	utilizar C# para a implementação do protótipo
RNF04	salvar configurações do protótipo em arquivo de texto

Fonte: elaborado pelo autor.

3.2 DESENVOLVIMENTO DO PROTÓTIPO

A Figura 6 apresenta o diagrama de atividades utilizadas para o treinamento da CNN do protótipo desenvolvido. Inicialmente a base de dados de imagens utilizadas para o treinamento são carregadas. Em seguida, a rede importa o arquivo yolov4.conv.137 que possui pesos pré-treinados responsáveis por criar o ponto inicial do treinamento. Após importar o arquivo, inicia-se o treinamento e executa a quantidade de iterações definidas. Por fim, ao finalizar o treinamento a rede realiza as predições das classes treinadas.

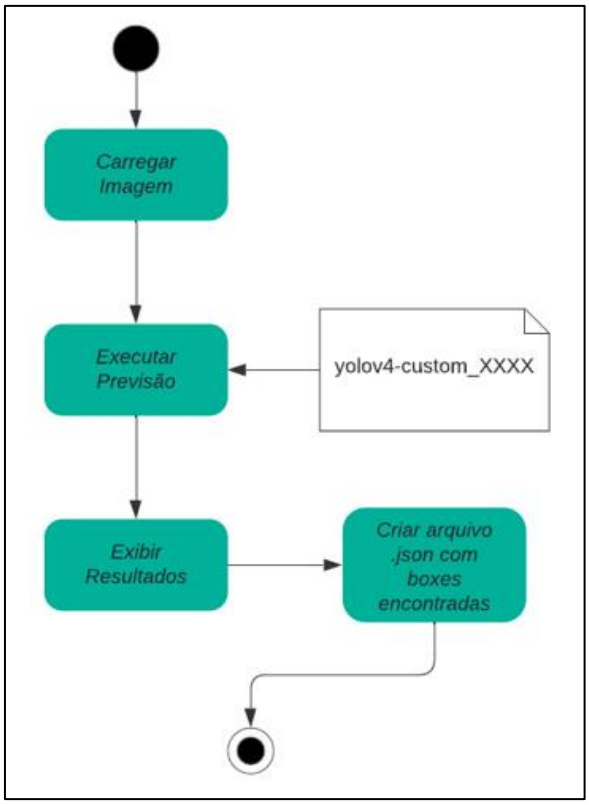
Figura 6 – Diagrama de Atividades de Treinamento



Fonte: elaborado pelo autor.

Na Figura 7 é apresentado o diagrama de atividades utilizado para realizar os testes de detecção. A primeira etapa desse processo é o carregamento da imagem a ser testada. Então, é executada a previsão com base no modelo treinado, e exibe a imagem demonstrando os resultados. Na última etapa, é criado um arquivo json (JavaScript Object Notation), que é um formato simples para a troca de dados, onde ficam armazenadas as informações da classe definida e limitadores da borda referente a detecção realizada.

Figura 7 – Diagrama de Atividades de Teste



Fonte: elaborado pelo autor.

3.2.1 Base de Dados de Bugios-Ruivo e Data Augmentation

Para a criação de uma base de dados de Bugios-Ruivo, foram utilizados dois conjuntos de dados com primatas do Projeto Bugio-FURB. As imagens dos Bugios-Ruivo foram feitas nas instalações do Projeto Bugio-FURB na cidade de Indaial-SC. O primeiro conjunto de dados, disponibilizado por Krause (2019), contém 28 imagens e 36 vídeos de bugios-ruivo. A segunda base de dados foi disponibilizada pelo bolsista de pesquisa Rafael Sperandio na qual possui 178 imagens de bugios.

A partir da base disponibilizada por Krause (2019), que continha em sua maioria vídeos, foram realizadas conversões desses vídeos para fotos e foi criado uma base com 210 imagens de bugios-ruivo. Com isso, uma única base de dados com as imagens de Krause (2019) e Rafael foi formada, contendo um total de 388 imagens. Algumas das imagens da base de dados foram removidas, pois possuíam qualidade inferior as demais e por esse motivo também impossibilitavam a detecção da face do primata.

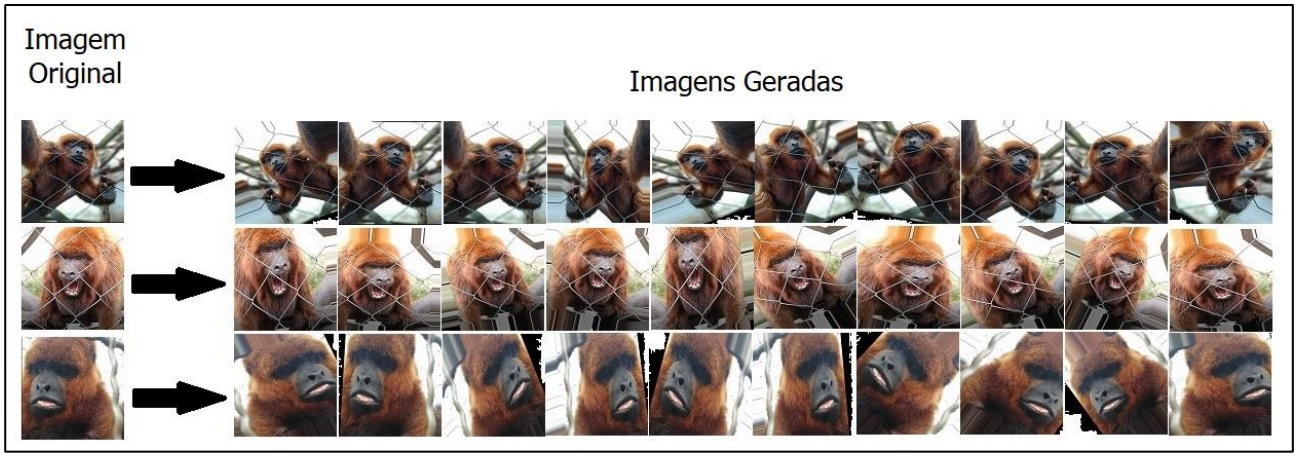
Então, a base de dados foi estruturada com um total de 360 imagens de bugios-ruivo. Essas imagens contém a face desses primatas com variações de pose, rotação e iluminação como mostrado na Figura 8. Com essas variações na imagem, algumas não possuem toda a face do bugio visível. As imagens da base de dados foram utilizadas em seu tamanho original, não havendo a necessidade de recortá-las, pois foram realizadas marcações nas imagens ao redor da região da face do primata.

Figura 8 – Imagens utilizadas na base de dados

Fonte: elaborado pelo autor.

Devido à pequena quantidade de imagens para realizar o treinamento da RNA, foi utilizado a técnica de *data augmentation* para aumentar o número de imagens da base de dados. Essa técnica gera variações da imagem original gerando novas imagens. Para isso, utilizou-se o pacote `ImageDataGenerator` da biblioteca `Keras`, criando imagens a partir de parâmetros como espelhamento, rotação, escala, deslocamento e preenchimento definidos na função. Para ampliar a base de dados atual foram utilizados os parâmetros descritos e, a partir de cada imagem da base atual, foram geradas mais nove imagens com alterações do *data augmentation*, como apresentado na Figura 9. Por fim, a base de imagens dos bugios-ruivo ficou composta por 3609 imagens.

Figura 9 – *Data augmentation* aplicado a base de dados inicial de bugios-ruivo



Fonte: elaborado pelo autor.

3.2.2 Base de Dados de Outras Espécies e Data Augmentation

Além da base de bugios-ruivo, foram selecionadas manualmente imagens de outras espécies de primatas para complementar a base de dados com exemplos negativos, isto é, exemplos de imagens que não são de bugios-ruivo, conforme apresentado na Figura 10. Para isso, foram selecionadas três bases de dados, que geraram quatro novas classes no treinamento da RNA. A primeira, disponibilizada pela plataforma do Kaggle (2019) possui duas classes, uma de bugios-manto e outra de macaco-pata. A segunda base de dados possui imagens de chimpanzés e foi disponibilizada por IMAGES.CV (2022?). Por fim, a terceira base de dados foi disponibilizada na plataforma do Kaggle (2021) e possui imagens de humanos.

Essas bases de dados de outras espécies de primatas utilizados como exemplos negativos, passaram também pelo processo de *data augmentation*, com o intuito de expandir a base atual. Para cada imagem de exemplo negativo, foram geradas 5 imagens, complementando a base com mais 2408 imagens. A classe de bugios-manto ficou composta por 840 imagens, a classe de chimpanzés possui 512 imagens, os macacos-pata 888 imagens e os humanos 336 imagens. A Figura 10 apresenta exemplo das imagens selecionadas para a base de dados.

Figura 10 – Imagens consideradas exemplos negativos para o treinamento da rede

Fonte: elaborado pelo autor.

Por fim, a base de dados para o treinamento da RNA ficou composta por cinco classes de primatas, sendo elas: bugios-ruivo, bugios-manto, macaco-pata, chimpanzé e humanos. A base de dados final possui um total de 6017 imagens que caracterizam todas as classes informadas na base de dados.

3.2.3 Arquitetura

Neste trabalho foi utilizado a arquitetura YOLOv4 de Bochkovskiy, Wang, LIAO (2020). Esta arquitetura foi inicialmente desenvolvida para projetar uma rápida detecção de objetos em sistemas de produção e optimização de cálculos paralelos, em vez do indicador teórico de baixo volume de computação. Foi projetada também para ser possível treinar a rede utilizando apenas uma GPU convencional, bem como realizar testes e obter retornos que sejam precisos em tempo real.

As estruturas fornecidas dos arquivos padrão estão configuradas para realizar a detecção de 80 classes. Essas configurações do total de classes, foram alteradas para cinco, isto é, o número de classes selecionadas na base de dados. Outra alteração realizada foi o filtro das camadas de convolução, que seguindo a recomendação da documentação YOLO é calculado por `filtros = (classes + 5) * 3`. Portanto esse número de filtros foi alterado para um total de 30 nas camadas anteriores a uma camada YOLO.

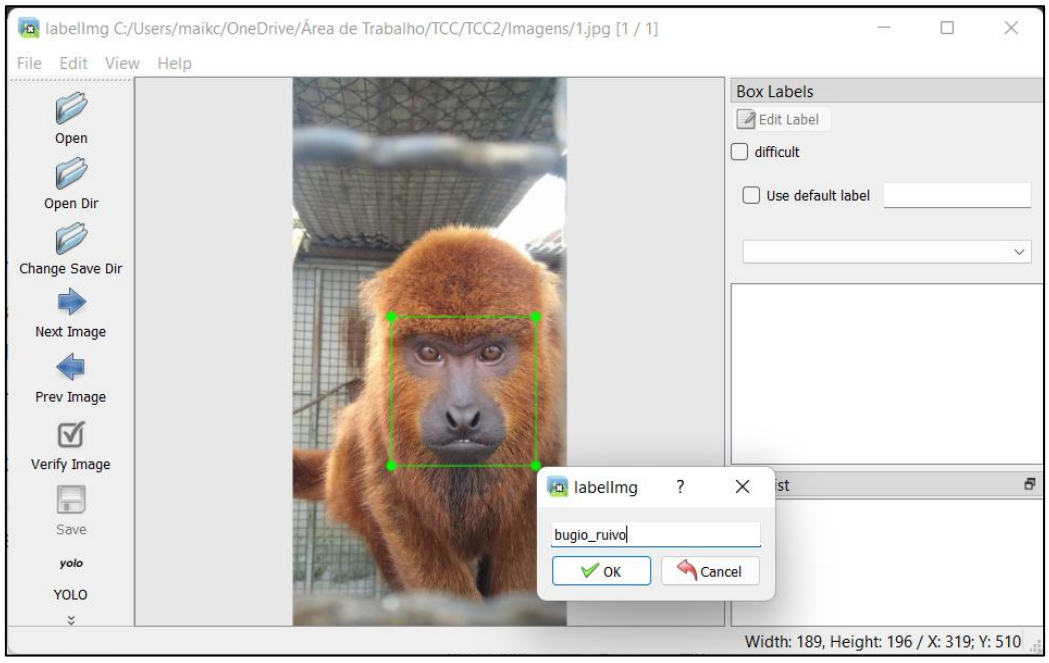
Também foram necessárias alterar as configurações do *batch e subdivisions* devido a gargalos de processamento e capacidade da máquina. O *batch* define o número de imagens que serão carregadas na memória a cada iteração de treinamento e foi configurado com `batch=64`. *Subdivisions* é a divisão das imagens no *batch*, impactando diretamente na velocidade de treinamento. Isto significa que, quanto menor o valor informado mais rápido o treinamento da rede ocorre. Esta propriedade foi definida com `subdivisions=32`, desta forma, a cada iteração de treinamento da rede, são carregadas 64 imagens em memória e são processadas duas imagens por vez.

Por fim, foi calculado o valor da quantidade de iterações que a rede deveria executar (max batches). Esse cálculo é feito utilizando a fórmula `(classes * 2000)` recomendado pela documentação do YOLO (BOCHKOVSKIY; WANG; LIAO, 2020). Os autores da arquitetura YOLOv4 informam que o número de iterações não deve ser menor que o número de imagens e que deve possuir um valor mínimo de 6000. Isso significa que, caso o número total de imagens seja maior que o número de iterações, deve-se adicionar mais 2000 iterações até que o número calculado seja maior que a quantidade de imagens. O *max batches*, que define a quantidade total de iterações que a RNA irá executar, foi informado com `max_batches = 10000` e tiveram suas *steps* com 80% e 90% desse valor respectivamente (`steps=8000,9000`).

3.2.4 Pré-Processamento e Treinamento

Para realizar o treinamento da RNA, a base de dados foi dividida em dois grupos, treino e teste. O grupo de treino é composto por 80% das imagens da base de dados, isto é, possui 4819 imagens. O grupo de teste, utilizado para realizar os testes da rede já treinada, possui 1205 imagens, que é equivalente a 20% do total da base de dados. Nenhuma das imagens da base foi utilizada nos dois grupos e todas elas foram marcadas manualmente com uma caixa delimitadora (*bounding box*) ao redor da face dos bugios-ruivo e demais primatas como apresentado na Figura 11.

Figura 11 – Interface do LabelImg e marcação de faces



Fonte: elaborado pelo autor.

Para realizar a marcação das imagens dos primatas, foi utilizado a ferramenta LabelImg desenvolvida por Tzutalin (2005). A LabelImg é uma ferramenta de anotação de imagem gráfica, desenvolvida em python e utiliza Qt para

a criação de sua interface, permitindo que o software seja compatível com plataformas como Linux, Windows, macOS e Android. A ferramenta foi escolhida pois salva as anotações feitas nas imagens como arquivos .txt compatíveis com o YOLO. Esse arquivo gerado, possui um sequencial de números separados por espaços em branco, como apresentado na Quadro 6. Cada linha do arquivo representa uma caixa delimitadora e sua classe de marcação.

Quadro 6 – Dados gerados na marcação da imagem da Figura 11

```
0 0.619444 0.577778 0.550000 0.263889
```

Fonte: elaborado pelo autor.

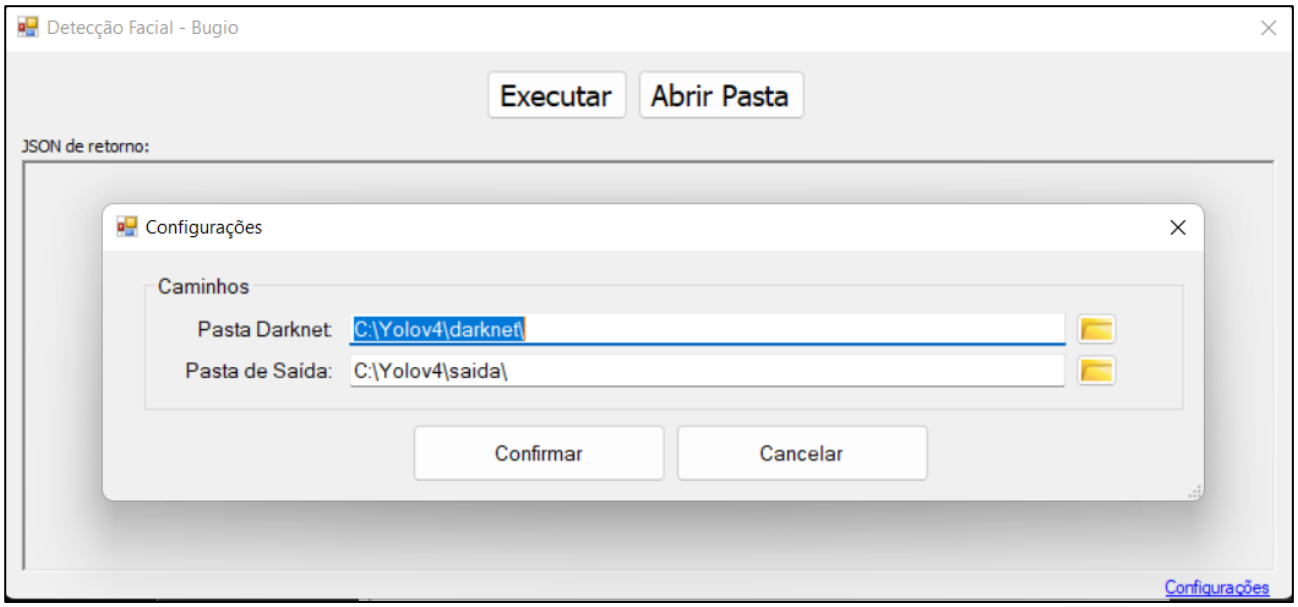
O primeiro número de cada linha do arquivo, representa a classe na qual foi realizada a marcação da face do primata, no Quadro 6 o número "0" representa a primeira classe do arquivo obj.names utilizada no treinamento. As demais anotações, representam a caixa delimitadora na sequência Xmin, XMax, YMin, YMax. XMin está em [0,1], onde 0 é o pixel mais à esquerda e 1 é o pixel mais à direita na imagem. As coordenadas Y vão do pixel superior (0) ao pixel inferior (1) (VITTORIO, 2018).

O treinamento da RNA YOLOv4, foi realizado na plataforma do Google Colab e em uma máquina com 16GB de RAM e uma GPU NVIDIA GeForce GTX 1650 Ti, essa máquina também foi utiliza para realizar os testes. Para a conclusão do treinamento foram utilizados 10000 lotes (*batches*) e 156 épocas (*epochs*) com aproximadamente 18 horas de treinamento.

3.2.5 Aplicação

A fim de demonstrar a finalidade do protótipo gerado, uma aplicação desktop foi desenvolvida utilizando a linguagem C#. Esta aplicação, apresentada na Figura 12, possui o objetivo de realizar a predição de uma imagem de entrada. A fim de deixar o protótipo dinâmico, ele deve ser parametrizado com o caminho da pasta `darknet`, que contêm os arquivos necessários para realizar a predição do YOLOv4. A aplicação deve também parametrizar o caminho da pasta de saída, que é o caminho onde a predição que foi realizada será armazenada, juntamente com um arquivo `json` contendo os resultados definidos em forma de texto. Caso as informações da configuração não sejam preenchidas, o protótipo informará dois caminhos pré-definidos para tentar executar as predições, esses caminhos são apresentados na Figura 12.

Figura 12 – Interface e configuração do protótipo desenvolvido

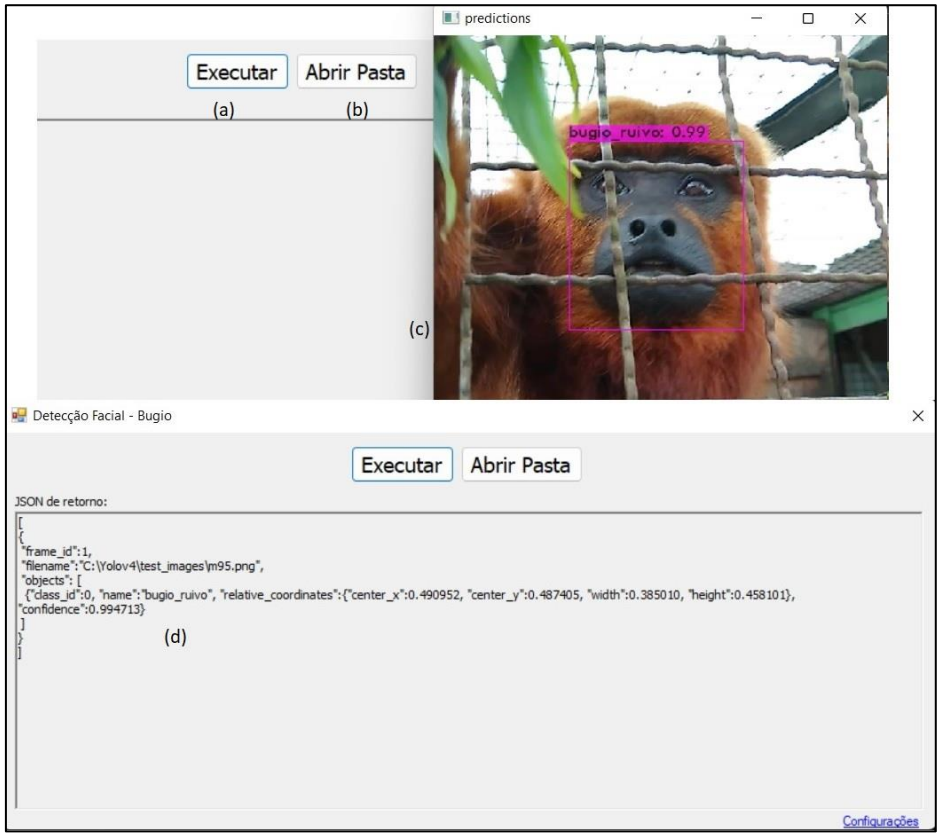


Fonte: elaborado pelo autor.

Para realizar a detecção facial no protótipo, deve-se efetuar a ação do botão Executar, exibido na Figura 13 (a), essa ação irá abrir uma nova janela onde será possível escolher a imagem a ser detectada. Ao selecionar a imagem, o protótipo irá realizar a predição, gerando a integração do protótipo desenvolvido com a rede YOLOv4 através de um arquivo `.bat`, demorando em média 20 segundos. Ao finalizar a predição da imagem selecionada, uma imagem com as marcações da face detectada será exibida, Figura 13 (c). Ao fechar o arquivo da imagem, o conteúdo gerado no formato `json` será exibido na área de texto da interface, conforme apresentado na Figura 13 (d). Esses arquivos gerados podem

ser encontrados na pasta de saída informado no painel de configurações, esta pasta pode ser acessada diretamente pressionando o botão de Abrir Pasta, Figura 13 (b).

Figura 13 – Predição realizada e dados de classificação resultantes



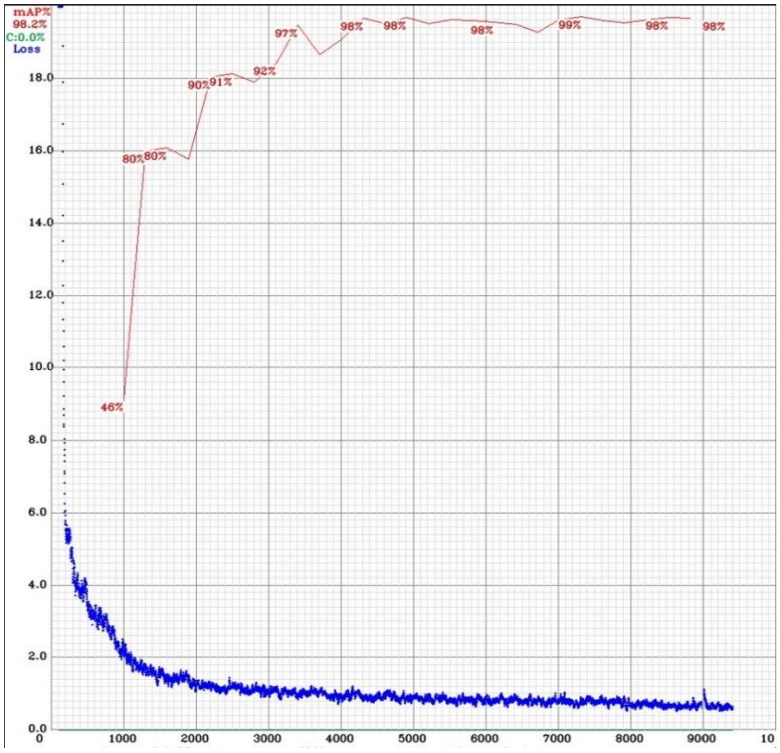
Fonte: elaborado pelo autor.

4 RESULTADOS

Neste capítulo são apresentados os resultados obtidos nos testes da RNA para realizar a detecção das faces dos primatas e determinar a classe referente à face marcada. A Figura 14 apresenta um gráfico contendo as 10.000 iterações do treinamento gerado através do YOLOv4. Para avaliar o modelo do YOLO, é usado uma predição média (Mean Avarage Precision - mAP). O mAP compara a caixa delimitadora (bounding box) das imagens de treinamento com a caixa detectada no teste e retorna uma pontuação. Quanto maior a pontuação, mais preciso é o modelo na detecção do objeto.

O gráfico demonstra a curva de aprendizado com a precisão média na linha vermelha. A precisão média (Mean Avarage Precision) no decorrer do treinamento varia de 46% nas primeiras 1000 iterações até uma mAP próxima dos 98% nas iterações terminais. Demonstra também na linha azul, a taxa de perda do treinamento ou erros no conjunto de dados de treinamento (Loss), que tendem a reduzir conforme as iterações estão sendo processadas.

Figura 14 – Gráfico de perda e mAP



Fonte: elaborado pelo autor.

Os testes de detecção foram realizados de duas formas, uma de forma manual e outra comparando os resultados dos arquivos de pesos gerados pelo treinamento da RNA. Para realizar os testes com os arquivos gerados no treinamento de 10.000 lotes, oito arquivos foram gerados com quantidades de iterações distintas. O primeiro arquivo apresenta a precisão média do treinamento após 1000 iterações. O arquivo com 9000 iterações teve uma precisão média maior, atingindo 98,56%. O arquivo com a segunda melhor precisão executou todas as 10000 iterações previstas pelo treinamento e alcançou 96,53% de precisão média. Para comparar os resultados do treinamento realizados pelo YOLOv4, foram geradas duas tabelas. A Tabela 1 retrata o percentual de treinamento dos bugios-ruivo nos 8 arquivos criados. Na tabela é possível observar a quantidade de iterações feitas no treinamento da rede e sua respectiva quantidade de épocas. Apresenta também a quantidade de imagens que obtiveram como resultado Verdadeiro Positivo (TP) e a quantidade de Falsos Positivos (FP). A última coluna da tabela representa a precisão média obtida na detecção das faces dos bugios-ruivo.

Tabela 1 – Percentuais de treinamento definidos pela rede para a classe dos bugios-ruivo

Iterações	Épocas	TP	FP	AP
1000	16	493	450	62,53%
2000	31	655	53	93,99%
3300	52	696	17	97,67%
4000	62	699	16	97,96%
7400	115	688	11	97,41%
8000	125	691	12	97,13%
9000	**140**	**707**	**9**	**98,56%**
10000	156	693	13	96,53%

Fonte: elaborado pelo autor.

A Tabela 2 representa os dados gerados e percentuais de treinamento para as 5 classes utilizadas no treinamento, que são: bugio-ruivo, bugio-manto, macaco-pata, humano e chimpanzé. A tabela possui as iterações realizadas seguida da quantidade de épocas, a quantidade de Verdadeiro Positivo (TP) e a quantidade de Falsos Positivos (FP). Na última coluna é apresentado a Intersecção sobre a União (Average IoU), que é um número entre 0 e 1 e específica quão sobreposta está a caixa delimitadora prevista à caixa delimitadora verdadeira. A tabela 2 utiliza os mesmos oito arquivos da Tabela 1 para realizar as predições.

O conjunto de treinamento que obteve a melhor precisão, foi o que utilizou 9000 iterações e 140 épocas, no caso, o mesmo conjunto que obteve a melhor precisão para os bugios-ruivo. O segundo melhor conjunto de treino obteve 73,23% de intersecção sobre a união com 10.000 iterações, obtendo assim uma precisão menor de 4,06% que o melhor

resultado mesmo que tenha sido treinado com 1000 iterações a mais. Este resultado ocorreu, pois as novas iterações treinadas resultaram em uma maior quantidade de falsos positivos, gerando uma maior perda da IoU.

Tabela 2 – Percentuais de treinamento para as 5 classes

Iterações	Épocas	TP	FP	Average IoU
1000	16	731	977	28,79%
2000	31	1020	189	61,23%
3300	52	1089	115	68,94%
4000	62	1122	88	71,53%
7400	115	1097	55	72,05%
8000	125	1118	69	72,30%
9000	**140**	**1155**	**41**	**77,29%**
10000	156	1118	46	73,23%

Fonte: elaborado pelo autor.

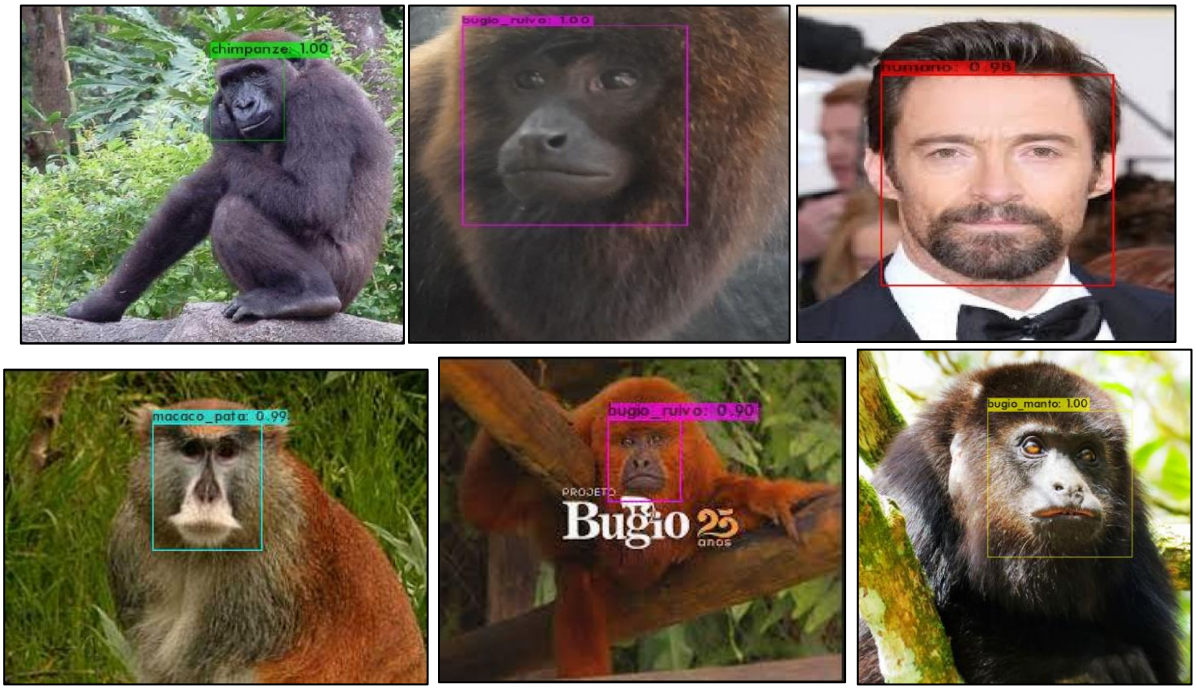
O arquivo de pesos utilizado para o teste manual foi selecionado conforme os resultados a intersecção sobre a união (*Average IoU*) apresentada na Tabela 2. Foram comparadas imagens da base de dados disponibilizados por Krause (2019) e por Rafael, como também outras imagens de páginas da internet que não pertencem a base de dados. As imagens dos primatas foram selecionadas e carregadas no detector YOLO. Os testes com imagens foram realizados a partir de um script exibido no Quadro 7. Também foram realizados testes para cada uma das classes treinadas com os arquivos de pesos gerados. Após a detecção da face do primata ser executada um arquivo JSON é criado com as predições da classe detectada, com as coordenadas relativas da caixa delimitadora e com o percentual atingido na operação. Alguns resultados obtidos do arquivo de 9000 iterações, que possui a melhor precisão média, são apresentados nas Figura 15.

Quadro 7 – Comando utilizado para executar as predições

```
darknet detector test <arquivo.data> <yolov4-custom.cfg> <yolov4-custom_xxxx.weights> <imagem> -thresh 0.6 -out result.json
```

Fonte: elaborado pelo autor.

Figura 15 – Teste com 9000 iterações das 5 classes treinadas



Fonte: elaborado pelo autor.

Em alguns casos, os testes com imagens de bugios-ruivo exibiram erro na classificação. Os primatas tiveram suas faces marcadas corretamente, porém o rótulo com o nome da classe foi classificado de forma incorreta. A Figura 16 exibe essas classificações falso positivos para a classe de bugios-ruivo. Os falsos positivos foram mais frequentes em imagens da classe de bugios-ruivo e bugios-manto, devido a semelhança entre esses dois primatas da mesma família de primatas.

Ocorrem também em imagens que possuíam um nível mais elevado de oclusão, com qualidade baixa na imagem e em imagens que o bugio estava distante.

Figura 16 – Falsos positivos para a detecção de bugios-ruivo

Fonte: elaborado pelo autor.

5 CONCLUSÕES

Este trabalho apresentou um protótipo para realizar a detecção facial de bugios-ruivo utilizando o algoritmo de Rede Neural Convolucional YOLOv4, a fim de auxiliar os veterinários e pesquisadores do Projeto Bugio-FURB na marcação e identificação dos indivíduos. Para isso, foi construído uma base de dados contendo 6017 imagens, composta por 5 classes de primatas, sendo elas: bugios-ruivo, bugio-manto, macaco-pata, humano e chimpanzés. Os testes de detecção foram realizados de duas formas, uma de forma manual e outra comparando os resultados dos arquivos de pesos gerados pelo treinamento da RNA. O modelo treinado alcançou uma precisão média de 98,56% para bugios-ruivo. O modelo gerado alcançou o objetivo de detectar a face dos primatas com uma Intersecção sobre a União (IoU) de 77,29%, mostrando que conseguiu identificar relativamente bem a face dos primatas na maioria das imagens. Também foi gerado um arquivo `json`, responsável por permitir integrar o protótipo atual com outros protótipos ou compartilhar dados.

Com os resultados, pode-se observar que as imagens devem ter toda a face do primata visível para que seja possível realizar com maior precisão o reconhecimento de sua face. Pode-se observar também que uma maior precisão foi obtida em imagens de boa qualidade e mais próximas. O protótipo alcançou uma boa capacidade de detecção das faces, mas devido a pouca quantidade de imagens utilizadas no treinamento, o protótipo apresentou algumas limitações na classificação do rótulo do primata. Uma possível solução para aumentar a precisão da classificação da espécie do primata e de seu rótulo, seria ampliar a base de dados, com mais imagens de bugio-ruivo e ampliar também a quantidade de imagens de outras espécies de primatas a serem utilizados como exemplos negativos. Não foi possível realizar a junção do protótipo atual com o protótipo desenvolvido por Krause (2019), devido a inconsistência de informações entre os protótipos.

Como possíveis trabalhos futuros a este protótipo desenvolvido, além da ampliação da base de dados, seria desenvolver uma aplicação mobile. Está aplicação permitiria que os testes de detecção facial em conjunto com o reconhecimento do primata realizado pelos veterinários e pesquisadores possam ser realizados diretamente no habitat natural desses primatas. A aplicação mobile poderia conter também o cadastro de informações características do bugio-ruivo, como o nome, coordenadas com data e hora de onde foi localizado, e possíveis observações para que possa ser realizado uma análise dos dados.

REFERÊNCIAS

ABREU, Viviana Rubina Gonçalves. **Reconhecimento Facial-Comparação do Uso de Descritores Geométricos Heurísticos e Aprendizagem Profunda**. 2021. Dissertação (Mestrado Integrado em Engenharia Electrotécnica e de Computadores) - Departamento de Engenharia Electrotécnica e de Computadores, Faculdade de Ciências e Tecnologia, Universidade de Coimbra, Coimbra.

ALVES, Gabriel. **Detecção de Objetos com Yolo**. [2020]. Disponível em: https://iaexpert.academy/2020/10/13/deteccao-de-objetos-com-yolo-uma-abordagem-moderna/. Acesso em: 02 jun. 2021.

AMDA. Associação Mineira de Defesa do Ambiente. **Bugios são os primatas com maior distribuição dos neotrópicos**, [2019]. Disponível em: https://www.amda.org.br/index.php/comunicacao/especie-da-vez/5501-bugios-sao-os-primatas-com-maior-distribuicao-dos-neotropicos. Acessado em: 18 abr. 2021.

BICCA-MARQUES, Júlio C. *et al*., **Mamíferos – Alouatta guariba clamitans – Guariba ruivo**. Porto Alegre, [2015]. Disponível em: https://www.icmbio.gov.br/portal_antigo/component/content/article.html?id=7179:mamiferos-alouatta-guariba-clamitans-guariba-ruivo. Acesso em: 28 mar. 2021.

BOCHKOVSKIY, A., WANG, C. Y., and LIAO, H.-Y. M.. **Yolov4: Optimal speed and accuracy of object detection** 2020. Disponível em https://arxiv.org/abs/2004.10934. Acesso em: 12 jul. 2022.

BUSS, Gerson. Conservação do bugio-ruivo (Alouatta guariba clamitans) (PRIMATES, ATELIDAE) no entorno do Parque Estadual de Itapuã, Viamão, RS. 2012.

CHEN, Peng *et al.* A study on giant panda recognition based on images of a large proportion of captive pandas. **Ecology and evolution**, v. 10, n. 7, p. 3561-3573, 2020.

IMAGES.CV. **Chimpanzee image classifcation dataset**. [2022?]. Disponível em: https://images.cv/dataset/chimpanzee-image-classification-dataset. Acessado em: 5 jun 2022.

CROUSE, David *et al.* LemurFaceID: A face recognition system to facilitate individual identification of lemurs. **Bmc Zoology**, v. 2, n. 1, p. 1-14, 2017.

FERNEDA, E. Redes neurais e sua aplicação em sistemas de recuperação de informação. **Ciência da Informação**, Brasília, v. 35, n. 1, p. 25-30, jan./abr. 2006. Disponível em: http://www.scielo.br/pdf/ci/v35n1/v35n1a03.pdf. Acesso em: 14 jun 2022.

FURB. **Projeto Bugio**. [2001?]. Disponível em: http://www.furb.br/web/5579/projeto-bugio/apresentacao. Acesso em: 28 mar 2021.

KAGGLE. **Labeled Mask Dataset (YOLO_darknet)**, [2021]. Disponível em: <https://www.kaggle.com/datasets/techzizou/labeled-mask-dataset-yolo-darknet>. Acessado em: 21 maio 2022.

KAGGLE.**10 Monkey Species**, [2019]. Disponível em: <https://www.kaggle.com/datasets/slothkong/10-monkey-species>. Acessado em: 26 maio 2022.

KHAN, Asifullah *et al*. A survey of the recent architectures of deep convolutional neural networks. **Artificial intelligence review**, v. 53, n. 8, p. 5455-5516, 2020.

KRAUSE Jr, Orlando. **Reconhecimento Facial de Bugios-Ruivo Através de Redes Neurais Convolucionais**. 2019. 18 f. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

LEOCÁDIO, Rodolfo R. V. *et al*. Detecção de Abelhas Nativas em Colmeias em Campo Utilizando Visão Computacional. In: WORKSHOP DE COMPUTAÇÃO APLICADA À GESTÃO DO MEIO AMBIENTE E RECURSOS NATURAIS (WCAMA), 12., 2021, Evento Online. **Anais** [...]. Porto Alegre: Sociedade Brasileira de Computação, 2021. p. 59-68. ISSN 2595-6124. DOI: https://doi.org/10.5753/wcama.2021.15737.

LIN, Tzuta. **LabelImg**. 2015. Disponível em: https://github.com/tzutalin/labelImg. Acesso em: 12 jul. 2022.

MEIRA, Natalia. **Edge AI – MASkRCNN e Segmentação de Instâncias**. [2020]. Disponível em: http://www2.decom.ufop.br/imobilis/segmentacao-instancias/. Acesso em: 02 jun. 2021.

MELO, Carlos. **Reduzindo o Overfitting com Data Augmentation,** [2019]. Disponível em: https://sigmoidal.ai/reduzindo-overfitting-com-data-augmentation/. Acesso em: 07 jun. 2021;

RUSSELL, Stuart; NORVIG, Peter. **Inteligência Artificial**, 3ª Edição. Rio de Janeiro. Elsevier, 2013.

SCHOFIELD, Daniel *et al.* Chimpanzee face recognition from videos in the wild using deep learning. **Science advances**, v. 5, n. 9, p. eaaw0736, 2019.

SINHA, Sanchit *et al.* Exploring bias in primate face detection and recognition. In: **Proceedings of the European Conference on Computer Vision (ECCV) Workshops**. vol 11129. 2018. Lecture Notes in Computer Science(), Springer, Cham. DOI: https://doi.org/10.1007/978-3-030-11009-3_33.

VARGAS, Ana Caroline Gomes; CARVALHO, Aline; VASCONCELOS, Cristina Nader. Um estudo sobre redes neurais convolucionais e sua aplicação em detecção de pedestres. In: **Proceedings of the xxix conference on graphics, patterns and images**. 2016. Niterói: Universidade Federal Fluminense, 2016. p. 1-4.

VITTORIO, A. **Toolkit to download and visualize single or multiple classes from the huge Open Images v4 dataset**. 2018. Disponível em: https://github.com/EscVM/OIDv4_ToolKit. Acesso em: 19 jun. 2022.

YU, Jimin; ZHANG, Wei. Face mask wearing detection algorithm based on improved YOLO-v4. **Sensors**, v. 21, n. 9, p. 3263, 2021.

ZHANG, A., LIPTON, Z., Li, M., and SMOLA, A.J. **Dive into Deep Learning**. 2020. Disponível em: https://d2l.ai. Acesso em: 21 jun. 2022.