UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
DEPARTAMENTO DE ENGENHARIA ELÉTRICA E ELETRÔNICA
CURSO DE GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Samuel Eduardo Noll

Classificação de tipos de placas veiculares presentes em imagens através de aprendizado de máquina e redes neurais

Florianópolis
2023

Samuel Eduardo Noll

Classificação de tipos de placas veiculares presentes em imagens através de aprendizado de máquina e redes neurais

Trabalho de Conclusão do Curso de Graduação em Engenharia Elétrica do Centro Tecnológico da Universidade Federal de Santa Catarina como requisito para a obtenção do título de Bacharel em Engenharia Elétrica.
Orientador: Prof. Danilo Silva, Ph.D.

Florianópolis
2023

Ficha de identificação da obra

Noll, Samuel Eduardo
Classificação de tipos de placas veiculares presentes em imagens através de aprendizado de máquina e redes neurais / Samuel Eduardo Noll ; orientador, Danilo Silva, 2023.
57 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Centro Tecnológico, Graduação em Engenharia Elétrica, Florianópolis, 2023.

Inclui referências.

1. Engenharia Elétrica. 2. Aprendizado de Máquina. 3. Placas veiculares. I. Silva, Danilo. II. Universidade Federal de Santa Catarina. Graduação em Engenharia Elétrica. III. Título.

Samuel Eduardo Noll

Classificação de tipos de placas veiculares presentes em imagens através de aprendizado de máquina e redes neurais

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

Florianópolis, 9 de fevereiro de 2023.

Documento assinado digitalmente
Miguel Moreto
Data: 13/02/2023 17:37:59-0300
CPF: ***.850.100-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Miguel Moreto, Dr.
Coordenador do Curso de Graduação em Engenharia Elétrica

Banca Examinadora:

Documento assinado digitalmente
Danilo Silva
Data: 13/02/2023 12:14:48-0300
CPF: ***.557.924-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Danilo Silva, Ph.D.
Orientador
Universidade Federal de Santa Catarina

Documento assinado digitalmente
Richard Demo Souza
Data: 13/02/2023 12:35:28-0300
CPF: ***.267.379-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Richard Demo Souza, Dr.
Universidade Federal de Santa Catarina

Documento assinado digitalmente
MARIO DE NORONHA NETO
Data: 13/02/2023 13:11:44-0300
CPF: ***.859.519-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Mario de Noronha Neto, Dr.
Instituto Federal de Santa Catarina

Este trabalho é dedicado à Deus, aos meus pais, Valdir e Mirian, e aos meus irmãos, Milena e Isaque.

AGRADECIMENTOS

Sou muito grato a Deus por me proporcionar fazer este curso, me dar forças para completá-lo e por permitir finalizar este trabalho. Agradeço aos meus pais, Valdir e Mirian, que muito me incentivaram a fazer este curso, que me suportaram durante toda a minha vida incluindo durante esta graduação. Ao meu pai Valdir agradeço por me ajudar a realizar este trabalho buscando dar todo suporte que eu precisava e me incentivou a concluí-lo. Aos meus irmãos, Milena e Isaque, também agradeço por estarem comigo desde sempre. Agradeço também aos meus colegas de classe, principalmente ao Vitor, Vinícius e Vanderson, pela amizade, o bom convívio e ajuda para realizar a graduação. Ao professor Danilo também agradeço porque me orientou muito bem neste trabalho.

Tudo o que fiz foi por meio de Jesus. Toda a vontade, força, aprendizado, resiliência, empolgação, determinação que tive veio por meio dele e sem ele eu nada conseguiria fazer. Por isso, este trabalho é para sua glória.

RESUMO

O objetivo da pesquisa é avaliar o desempenho de algumas redes neurais de classificação ao classificar tipos de placas de identificação veicular contidas em imagens. Essas imagens foram tiradas principalmente com foco na parte dianteira e traseira de caminhões que transitaram em entradas e saídas de portos. Foram escolhidas 7 classes que representam os seguintes 7 tipos de placas: brasileira do padrão antigo, brasileira do novo padrão Mercosul, argentina do padrão antigo, argentina do novo padrão Mercosul, paraguaia do padrão antigo, paraguaia do novo padrão Mercosul e chilena. O treinamento supervisionado foi feito em dois modelos de redes neurais: YOLOv3 e ResNet50; e utilizou-se de 120 imagens que foram rotuladas de cada uma das 7 classes para servir na etapa de treinamento da rede neural. O algoritmo que obteve o melhor resultado foi o que utilizou o modelo YOLOv3, resultando em uma acurácia aproximada de 80% no conjunto de teste.

Palavras-chave: Aprendizado de Máquina. YOLOv3. Placas veiculares.

ABSTRACT

The goal of the research is to evaluate the performance of some classification neural networks by classifying types of vehicular license plates contained in images. These images were taken primarily fpcusing on the front and rear of trucks were entering and leaving harbors. Seven classes representing the following 7 types of plates were chosen: old standard brazilian, new Mercosul standard brazilian, old standard argentine, new Mercosul standard argentine, old standard paraguayan, new Mercosul standard paraguayan and chilean standard. Supervised training was done on two neural networks models: YOLOv3 and ResNet50; 120 images were labeled from each one of the 7 classes for the neural network training step. The algorithm that obtained the best result was the one that used the YOLOv3 model, resulting in an approximate 80% of accuracy in the test set.

Keywords: Machine Learning. YOLOv3. License plate.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

OCR *Optical Character Recognition*

LPR *License Plate Recognition*

JPEG *Joint Photographic Experts Group*

XML *Extensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

Nas rodovias, nas entradas e nas saídas de recintos algumas empresas utilizam o reconhecimento automático de placas veiculares para identificar veículos [1]. Para isso, é realizada a tiragem de fotos da parte dianteira ou traseira do veículo para capturar uma imagem que contenha a placa, e na sequência é utilizado um algoritmo de reconhecimento de caracteres, chamado de OCR, que retorna uma sequência de caracteres relacionados à placa daquela imagem [2].

Existem diversos tipos de algoritmos de reconhecimento de caracteres de placas (*LPR – License Plate Recognition*) [3]. Alguns utilizam-se de apenas métodos de processamento de imagens[3], outros utilizam aprendizado de máquina [4], mas todos eles têm dificuldade de acertar todas as vezes os caracteres das placas analisadas. Os fatores que influenciam em falhas são vários, tais como a orientação inesperada da placa na imagem, placa com valores parcialmente apagados, imagem desfocada, caracteres da placa muito similares [5].

Na empresa GttLogistics® [1], na qual o autor trabalhou, este pôde perceber mais um fator que gera falhas. Este acontece especificamente em sistemas que precisam reconhecer vários tipos de placas, isto é, não somente de com um padrão de design, mas placas com diferentes padrões de design, alguns de países distintos. No sistema desta empresa, são utilizados diferentes algoritmos de LPR, um para cada tipo de placa. A falha acontece quando é enviado uma imagem e, por não se saber qual é o tipo de placa, o sistema a processa em todos os algoritmos LPR e mais de um retorna um valor reconhecido. Ou seja, neste caso, pelo menos um algoritmo feito para reconhecer certo tipo de placa acabou reconhecendo um valor para outro tipo de placa e, quando isso acontece, geralmente o valor reconhecido é incorreto. Como não se sabe qual é o tipo de placa presente na imagem, o sistema usa um dos valores reconhecidos pelos algoritmos como o certo, e as vezes escolhe um valor incorreto.

Existem algumas maneiras de minimizar os erros advindos desse problema, como, por exemplo, dar preferência para escolher os valores dos algoritmos cujos tipos de placa aparecem com mais frequência nas imagens. Porém se houvesse uma etapa que descobrisse qual o tipo de placa, esses sistemas poderiam utilizar-se dessa informação para escolher o algoritmo correto de reconhecimento de caracteres da placa, reduzindo-se ainda mais esse tipo de falha.

Este projeto busca encontrar um algoritmo que reconheça o tipo da placa veicular, de alguns países do Mercosul, presente em uma imagem visando reduzir a falha no reconhecimento de caracteres.

1.1 OBJETIVOS

Nas seções abaixo estão descritos o objetivo geral e os objetivos específicos deste TCC.

1.1.1 Objetivo Geral

Pesquisar, implementar, treinar e testar algoritmos que utilizem aprendizado de máquina (*machine learning*) com redes neurais para classificar o tipo de placa presente numa imagem. O uso deste algoritmo é útil para evitar falhas em sistemas que usam vários algoritmos LPR para reconhecer vários tipos de placas.

1.1.2 Objetivos Específicos

Para alcançar o objetivo geral, as seguintes etapas devem ser realizadas:

- Definir quais são as classes (os tipos de placas) que o algoritmo deverá classificar;
- Obter imagens com qualidade e tamanho suficientes para o treinamento, validação e teste dos modelos de redes neurais;
- Rotular as imagens;
- Distribuir adequadamente as imagens rotuladas em 3 conjuntos: de treinamento, te validação e de testes;
- Definir modelos de redes neurais a serem testados para a classificação de tipos de placas;
- Implementar os modelos em código de programação e treiná-los com o conjunto de treinamento;
- Avaliar o desempenho dos conjuntos de validação e de testes dos modelos treinados.

2 REVISÃO DE BIBLIOGRAFIA

Este trabalho acima de tudo é de aprendizado de máquina, ou *machine learning*. Ele envolve o treinamento de alguns modelos de redes neurais, como o YOLOv3 e o ResNet50.

2.1 APRENDIZADO DE MÁQUINA

Uma simples definição de aprendizado de máquina seria a ciência de fornecer aos computadores a habilidade de aprender sem serem explicitamente programados. No aprendizado de máquina, são fornecidos ao computador os dados de entrada e, às vezes, os dados esperados de saída e, a partir destes, o computador retorna um programa que gera essas saídas com base nos dados de entrada [6].

Portanto, o computador teria "aprendido" como fazer a predição da saída baseada neste processo de fornecer dados de entrada. Este processo é chamado de "treinamento" [6].

Esse programa retornado pelo computador é baseado num modelo de aprendizado, o qual dado os dados de entrada ele retorna uma saída e, através do processo de treinamento, é atualizado os parâmetros deste modelo a fim de ele retornar uma saída esperada [6].

Neste trabalho o tipo de problema é a classificação, pois a saída esperada para o dado de entrada (que é uma imagem) é o tipo de placa, que é um valor discreto e finito.

Para avaliar o modelo treinado no fim do processo de treinamento, é utilizado uma métrica de avaliação. Neste trabalho, foi utilizada a Acurácia[6], que é uma métrica que avalia a razão entre, dado um número D de dados de entrada, o número de vezes que o modelo acertou A e o número de dados de entrada D. Por exemplo, se o dado de entrada for uma imagem e forem dados 100 imagens para o modelo fazer a predição, se o modelo tiver acertado 85 imagens, a acurácia é de 85%.

$$Acurácia = \frac{Número\ de\ acertos}{Número\ total\ de\ entradas}$$

2.1.1 Redes Neurais

São um tipo de modelo de aprendizado, o qual é assim chamado por imitar o funcionamento dos neurônios do corpo humano. São formadas por camadas e em cada uma destas, por neurônios, que seria a unidade básica da rede neural [7].

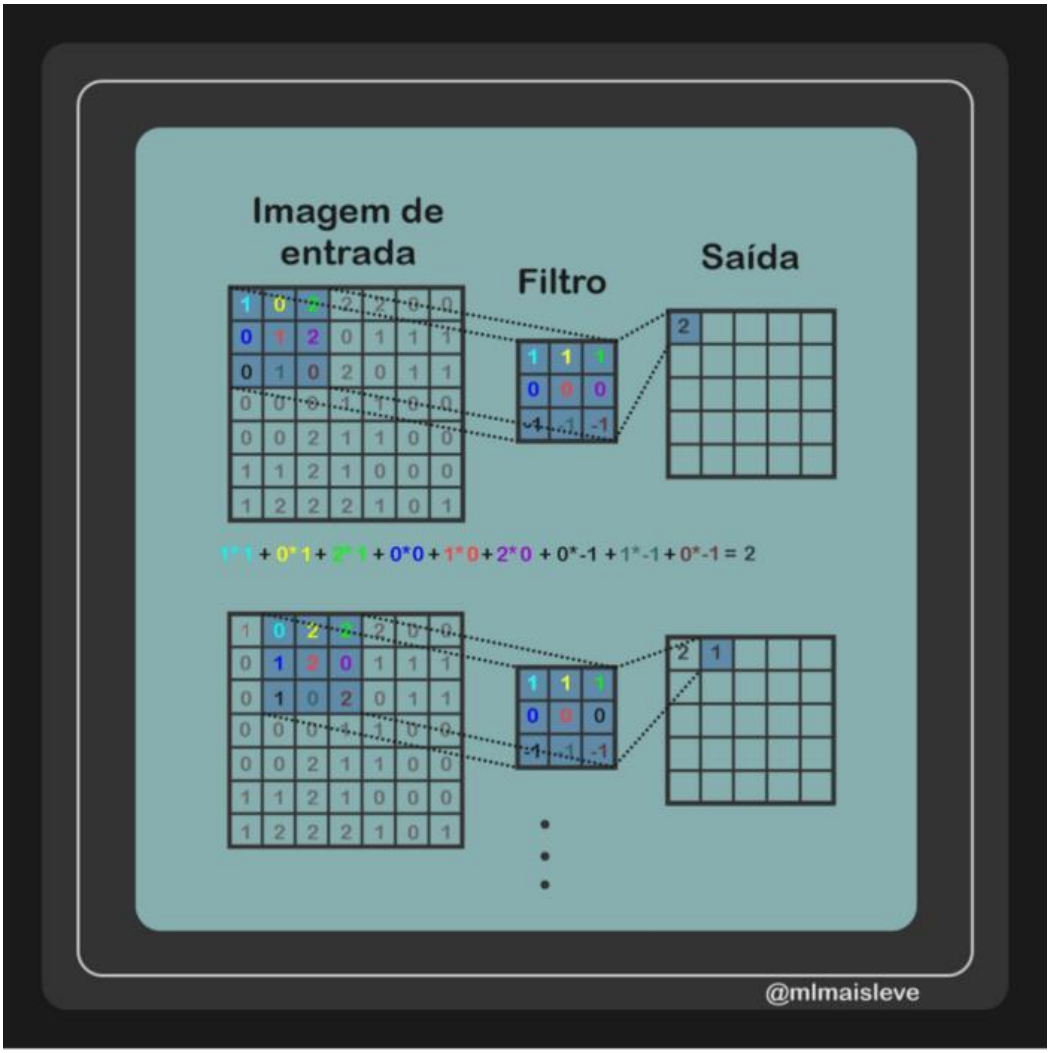
Cada neurônio recebe os valores de saída dos neurônios da camada anterior conectados a ele e calcula a saída por uma função de ativação, cujo parâmetro é chamado de “peso”. É possível colocar várias camadas num modelo de rede neural [7].

2.1.2 Camadas Convolucionais

A camada convolucional é uma importante camada de uma rede neural chamada Rede Neural Convolucional (*CNN – Concolutional Neural Network*). Essa camada é utilizada para identificar padrões e características de uma imagem, como bordas, formas, texturas ou cores [8].

Numa camada convolucional existem vários filtros que são responsáveis por detectar determinada característica de uma imagem. Esses filtros são aplicados à imagem de entrada através de convoluções – operação na qual o filtro se desloca de determinada forma na imagem a fim de cobrir toda ela, resultando numa outra imagem que contém características da imagem de entrada [8]. Uma demonstração disso pode ser vista na Figura 1.

Figura 1 – Exemplo de aplicação de um filtro em pixels através de convolução



Fonte: Vitor Borba Rodrigues, 2019.

Neste trabalho, foi utilizado dois modelos de redes neurais: o YOLOv3 e o ResNet50, os quais serão explicados nas próximas seções. Ambas são redes neurais convolucionais, assim chamadas por executarem o processo de convolução para as saídas dos neurônios de algumas de suas camadas.

2.1.3 Classificação de imagens

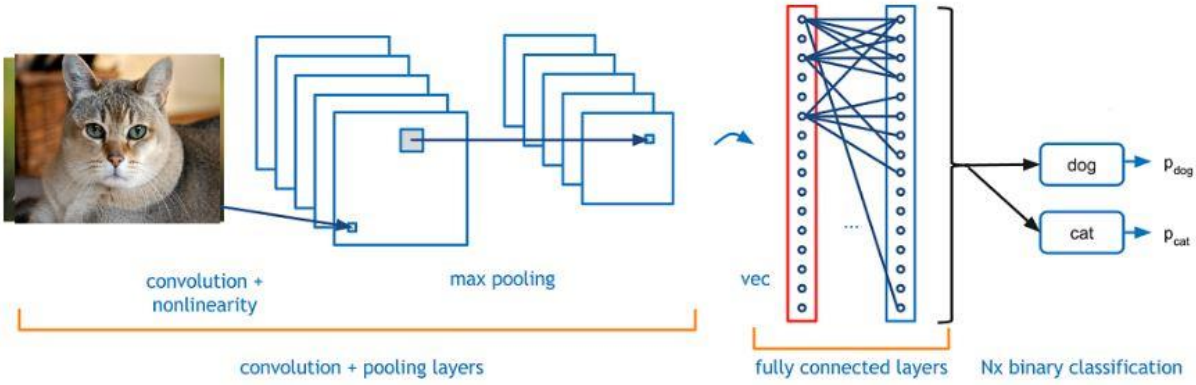
É uma vertente do Aprendizado de Máquina em que, dada uma série de classes e dada uma imagem, o modelo classificador estima qual classe está presente na imagem [9]. Por

exemplo, uma série de classes poderia ser gato e cachorro, e o modelo classificador iria buscar identificar se a imagem contém um gato ou um cachorro.

Existem diferentes técnicas para a classificação de imagens, dentre elas a CNN, a qual utiliza redes neurais e camadas convolucionais para classificar imagens. O modelo classificador tem como entrada uma imagem e como saída uma matriz que contém a probabilidade de a imagem ser de cada uma das classes [9]. O modelo busca encontrar características na imagem que descrevem a classe. Identificando e atribuindo pesos a essas características, o modelo consegue diferenciar uma classe da outra [9].

A Figura 2 ilustra uma CNN atuando na classificação de gatos e cachorros.

Figura 2 – Ilustração de uma CNN que classifica gatos e cachorros



Fonte: Yatheendra Pravan, 2021.

A classificação pode ser de dois tipos: binária ou multi-classe [10]. A binária é quando só há duas classes, como é o caso do exemplo citado acima de gatos e cachorros. A multi-classe é quando existem mais que duas classes, que é o caso deste trabalho.

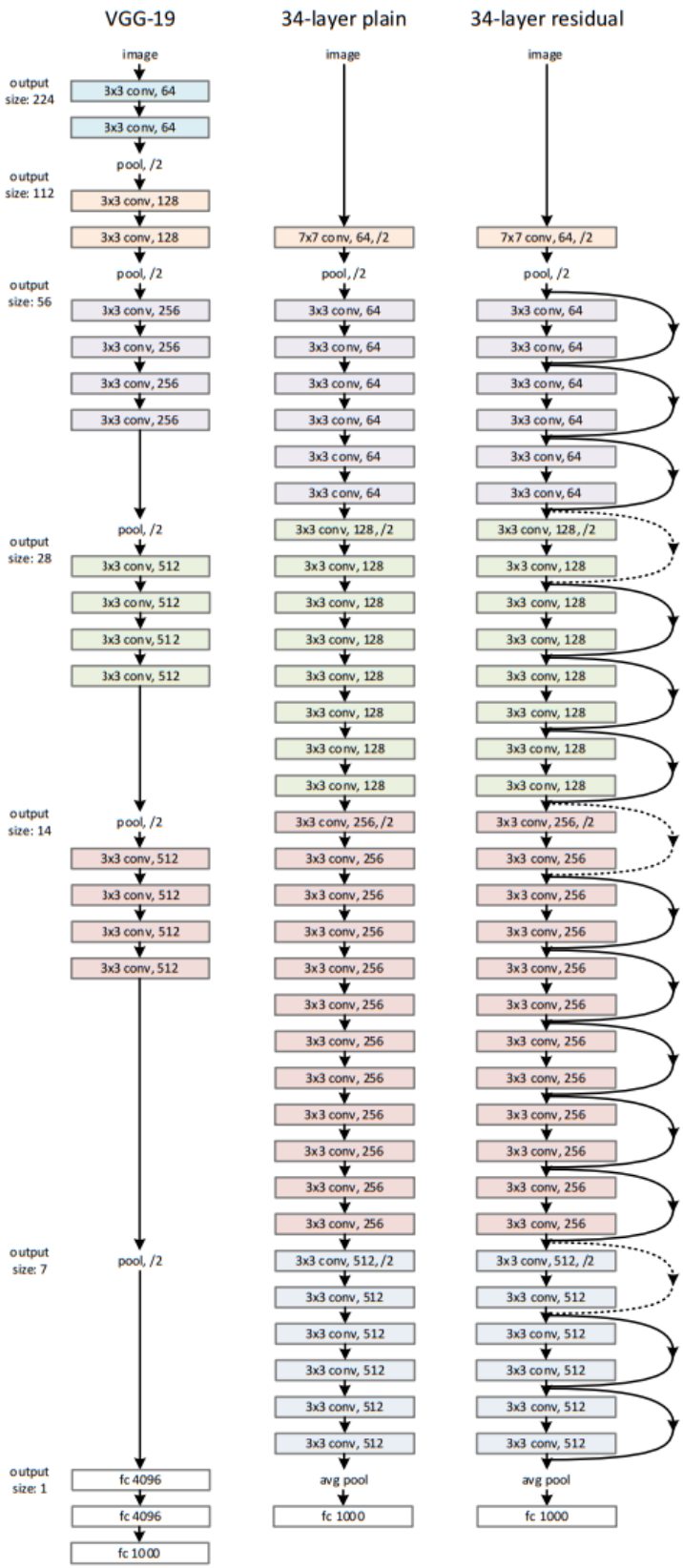
2.1.3.1 ResNet

ResNet é o acrônimo de *Residual Neural Network*, o qual é uma rede neural que atua na classificação de imagens com algumas camadas de convolução e com camadas residuais dentre as camadas modelo. Esta camada é chamada assim por estar conectada à camada anterior

da rede neural e à uma camada anterior a aquela a fim de calcular a diferença entre as saídas destas camadas para aumentar o desempenho da atualização de pesos nas etapas de treinamento [11].

Uma ilustração de como as camadas estão relacionadas pode ser vista na Figura 3. Nela, podemos ver à esquerda uma arquitetura, a VGG-19, utilizada de base para formar uma ResNet de 34 camadas residuais. Podemos nesta figura perceber como as camadas residuais foram postas a fim de periodicamente atualizar a saída de uma camada com o valor da saída de camadas anteriores [11].

Figura 3 – Arquitetura da ResNet34 baseado na VGG-19



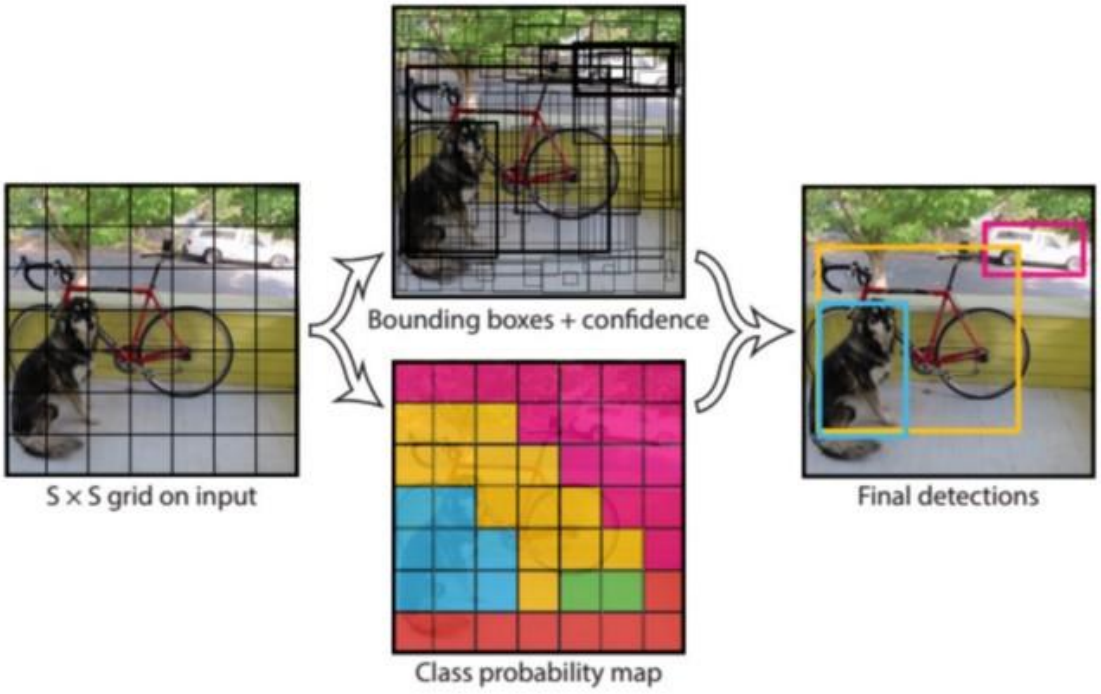
Fonte: GeeksforGeeks, 2023.

Neste trabalho foi utilizado o modelo ResNet com 50 camadas residuais, por isso o nome do modelo é ResNet50.

2.1.4 Detecção de objetos

A detecção de objetos é feita para classificar e indicar o posicionamento, altura e largura de objetos em imagens. Uma maneira simples de explicar o funcionamento dela é que ela divide a imagem em uma grade de células, nas quais pode conter ou não um objeto. Depois busca encontrar o centro desses objetos e criar caixas delimitadoras, que são caixas que contém certo objeto, ele descobre quais caixas delimitadores representam o mesmo objeto e juntam em uma única caixa delimitadora por objeto. Assim, esse modelo tem como saída a classe, o ponto central, a altura, a largura e, adicionalmente, a probabilidade de a classe deste objeto estar correta [12]. A Figura 4 ilustra com um exemplo este processo de detecção de objetos.

Figura 4 – Ilustração do processo de detecção de objetos em imagens



Fonte: Manish Chablani, 2017.

2.1.4.1 YOLO

YOLO é acrônimo de *You Only Look Once*, que significa “você apenas vê uma vez”, uma vez que este modelo foi feito para a identificação e localização de objetos em *frames* de vídeos em tempo real [12]. Neste trabalho, foi utilizado a versão 3 deste modelo, que contém alguns aprimoramentos em relação a versão 1 e 2 do YOLO, por isso o nome YOLOv3.

Uma das diferenças do YOLO para os outros modelos de detecção de objetos é que ele busca detectar os objetos em uma imagem apenas, e não em muitas imagens, de forma incremental, como os outros modelos. Isso o torna um pouco mais rápido do que os outros, sendo utilizado em vídeos com maior taxa de atualização de *frames* [13].

2.2 TREINAMENTO

Existem alguns tipos de treinamento: supervisionado, não-supervisionado e por reforço. Cada deles é utilizado para certos tipos específicos de problemas computacionais [14]. O treinamento supervisionado, por exemplo, é utilizado para problemas de regressão, quando a saída é uma variável contínua, ou de classificação, quando a saída é uma variável discreta [14].

Neste trabalho, será usado o treinamento supervisionado uma vez que o problema que temos nesse trabalho é o de classificação. Como os modelos selecionados para este trabalho são redes neurais, aqui será explicado apenas como funciona o treinamento supervisionado para redes neurais.

2.2.1 Treinamento supervisionado de redes neurais

O treinamento supervisionado de redes neurais é um processo complexo, mas pode ser resumidamente explicado seguindo o seguinte fluxo [15]:

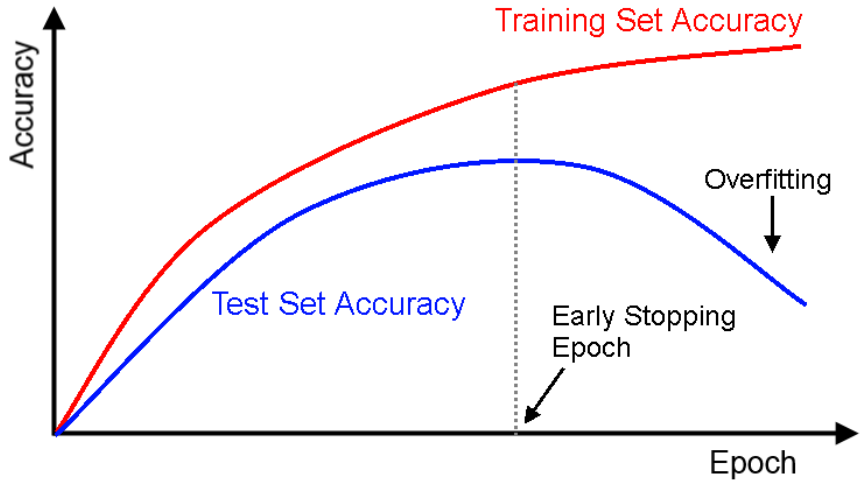
1. Dividimos o conjunto de dados em 3 conjuntos: o de treinamento, o de validação e o de teste.

2. Dividimos o conjunto de dados de treinamento em grupos menores de dados chamados *batches;*
3. O seguinte processo, chamado de Época, é repetido algumas vezes:
    a. Para cada *batch*, o modelo calcula a saída para cada dado de entrada com base nos pesos dos nós de cada camada, da relação entre eles e do tipo da camada;
    b. Depois de processar todos os *batches*, é calculado a Perda em cada nó, que é um valor que representa a diferença entre a saída calculada e a saída esperada presente nos rótulos de cada dado;
    c. Cada nó das camadas da rede neural atualiza os valores dos seus pesos a fim de minimizar a Perda;
4. No final de cada Época é calculado a Perda também do conjunto de validação afim da posterior avaliação do treinamento e do modelo treinado.
5. São realizados os passos 2, 3 e 4 diversas vezes variando alguns parâmetros de treinamento a fim de otimizar a acurácia do conjunto de validação do modelo treinado.
6. Quando se achar que o modelo estiver suficientemente treinado, é calculado a acurácia deste no conjunto de teste para estimar sua eficácia utilizando dados totalmente novos.

A quantidade de Épocas pode variar de acordo com algumas técnicas. É possível tanto determinar previamente uma quantidade de Épocas para o treinamento quanto fazer um cálculo envolvendo algumas métricas do treinamento, como a Perda ou a acurácia no conjunto de validação, após cada Época para verificar o modelo já aprendeu o suficiente, interrompendo o treinamento. Esta última técnica é chamada de *early stopping* [16].

É comum controlar o número de Épocas para evitar o *overfitting* – quando modelo treina tantas Épocas que aprende muito bem os dados do conjunto de treinamento, mas diminui a acurácia no conjunto de teste, como mostra a Figura 5. O *early stopping* busca interromper o treinamento quando chega no ponto onde se tiverem mais Épocas a acurácia no conjunto de teste começa a diminuir. A Figura 5 mostra um gráfico das acurácias dos conjuntos de treinamento e teste com o aumento da quantidade de Épocas e mostra o ponto ideal do *early stopping* [16].

Figura 5 – Gráfico que mostra quando há *overfitting* e quando deveria ocorrer o *early stopping*



Fonte: https://www.deeplearningbook.com.br, 2022.

É também possível realizar o treinamento determinando um número fixo de Épocas, salvando o estado do modelo a cada Época, e posteriormente, através da análise das acurácias de treinamento e teste, avaliar onde começou a haver o *overfitting* e assim determinar com qual número de Épocas o modelo foi treinado suficientemente bem. Este método é conhecido como *checkpointing* [17].

3 DESENVOLVIMENTO DO PROJETO

Nesta seção será explicado a metodologia e tudo o que foi feito em cada uma de suas das etapas.

3.1 METODOLOGIA DE DESENVOLVIMENTO

A seguinte metodologia foi adotada para este projeto:

1. Escolha dos tipos de placa (classes)
2. Obtenção de imagens
3. Divisão das imagens nos conjuntos de treinamento, validação e teste
4. Rotulação das imagens

5. Escolha de modelos de redes neurais
6. Treinamento dos modelos
7. Avaliação do desempenho dos modelos no conjunto de teste

Toda a implementação, treinamento e testes foram feitos utilizando Python, através do Jupyter Notebook e bibliotecas de aprendizado de máquina Keras e Tensorflow, com o auxílio de processamento de uma placa de vídeo GPU GeForce GTX 1060 com 6 GB de VRAM.

Foram utilizados alguns códigos de terceiros como base para o treinamento [18], formação dos modelos [19] e rotulação de imagens [20].

3.2 DADOS DE ENTRADA PARA TREINAMENTO

As imagens neste trabalho são os dados de entrada para o treinamento. É com base nas imagens que o modelo de redes neurais é treinado.

Esta seção trata dos quatro primeiros passos da metodologia apresentada, que são:

- Escolha dos tipos de placa (classes)
- Obtenção de imagens
- Divisão de imagens nos conjuntos de treinamento, validação e teste
- Rotulação das imagens

3.2.1 ESCOLHA DOS TIPOS DE PLACA

Cada país regulamenta os tipos de placas de identificação veicular que podem ser usados nos seus automóveis. Só no Brasil existem mais de dez tipos de placa. Entretanto não é necessário usar todos os tipos de placa existentes no mundo, mas sim escolher apenas poucos tipos uma vez que o objetivo deste trabalho é avaliar modelos de redes neurais que diferenciam tipos de placa.

Portanto, para obter uma quantidade de imagens suficientes para o treinamento dos modelos, foram escolhidos sete tipos de placa com maior recorrência nas regiões do sul do Brasil, que é onde as imagens foram obtidas.

Os sete tipos de placas são:

- **Brasileira:** brasileiras do padrão antigo, de fundo vermelhas com caracteres brancos ou de fundo cinza com caracteres pretos;
- **Brasileira Mercosul:** brasileiras do novo padrão Mercosul, de fundo branco com caracteres pretos ou de fundo branco com caracteres vermelhos;
- **Argentina:** argentinas do padrão antigo, de fundo preto com caracteres brancos ou de fundo branco com caracteres pretos;
- **Argentina Mercosul:** argentinas do novo padrão Mercosul, de fundo branco com caracteres pretos;
- **Paraguaia:** paraguaias do padrão antigo, de fundo branco com caracteres vermelhos;
- **Paraguaia Mercosul:** paraguaias do novo padrão Mercosul, de fundo branco com caracteres pretos;
- **Chilenas:** chilenas, de fundo branco com caracteres pretos.

A Figura 6 mostra pelo menos um exemplo de cada um desses sete tipos de placa.

Figura 6 – Exemplo de cada um dos sete tipos de placas escolhidos para este projeto

Fonte: Autor, 2023.

Esses sete tipos de placa representam as sete classes que o modelo de redes neurais deve classificar. Nota-se que essas placas possuem diferenças em alguns aspectos, como suas cores, design, dimensões, fontes dos caracteres, sequências de caracteres. Através destes aspectos que os modelos aprenderão a classificar estes diferentes tipos de placa.

3.2.2 OBTENÇÃO DAS IMAGENS

As imagens foram obtidas de câmeras que fotografaram a parte dianteira e traseira de veículos entrando e saindo de recintos alfandegários de portos e portos secos do sul do Brasil. A empresa catarinense GTT Logistics possuem essas câmeras nestes locais e forneceu as imagens para a execução deste trabalho.

A imensa maioria das imagens utilizadas foram tiradas num porto seco de Foz de Iguaçu, uma vez que lá trafegam veículos que contém todos os tipos de placas que foram escolhidas para este trabalho.

As câmeras utilizadas para tirar essas fotos são da empresa Pumatronix® – câmeras profissionais e específicas para a captura de imagens a serem processadas para fazer o reconhecimento de caracteres de placas veiculares contidas nas imagens.

As imagens foram gravadas em formato de compressão de imagem JPEG, coloridas e com resolução de 1280 pixels de largura por 960 pixels de altura.

3.3 DIVISÃO DAS IMAGENS EM CONJUNTOS

Para realizar o aprendizado de máquina as imagens foram divididas em três conjuntos: treinamento, validação e teste. Os conjuntos de treinamento e validação são utilizados no processo de treinamento do modelo de redes neurais e o conjunto de teste é utilizado depois do treinamento dos modelos para avaliar o desempenho dos modelos.

No treinamento do modelo, as imagens de treino são usadas para o aprendizado do modelo e as imagens de validação para avaliar o desempenho durante o treinamento.

3.3.1.1 Conjuntos de teste e validação

A escolha das imagens do conjunto de teste e validação, que são usadas para avaliar o desempenho, devem buscar representar os cenários reais, ou seja, conter imagens das placas de

diferentes ângulos, distâncias, claridades, posicionamentos no veículo, entre outros cenários, nos quais os modelos serão treinados para fazer a predição.

Portanto, observando as imagens obtidas, percebeu-se que elas se diferem nas seguintes aspectos:

1. Posicionamento da placa:
    a. Foto com uma placa posicionada no centro da dianteira ou traseira do veículo;
    b. Foto com uma placa posicionada na esquerda da dianteira ou traseira do veículo;
    c. Foto com duas placas posicionadas no centro da traseira do veículo;
    d. Foto com duas placas posicionadas em cada uma das laterais da traseira do veículo;
    e. Foto com uma placa posicionada atrás de uma das rodas da carroceria do veículo;
    f. Foto com uma placa de cor diferente da padrão posicionada no centro da dianteira ou traseira do veículo;
2. Iluminação:
    a. Natural (diurna)
    b. Artificial (noturna, por meio de iluminadores)

O aspecto 1.f - foto com uma placa de cor diferente da padrão posicionada no centro da dianteira ou traseira do veículo – existe porque alguns tipos de placas possuem uma opção com uma variação de cor.

Combinando estes dois aspectos (iluminação e posicionamento da placa), todos os cenários possíveis considerados são 12, mostrados no Quadro 1. Porém não são todas as classes, ou seja, todos tipos de placa, que possuem todos estes 12 cenários, uma vez que as normas de posicionamento das placas em cada país são diferentes.

Para cada classe, foram escolhidas 80 imagens para os conjuntos de teste e de validação com as diferentes quantidades para cada cenário. A quantidade é proporcional à frequência com que ela ocorre nas imagens obtidas. O Quadro 1 mostra a quantidade de imagens de cada cenário que foram escolhidas para cada classe.

Depois da escolha das 80 imagens, metade foi para o conjunto de validação e metade para o conjunto de teste, proporcionalmente a cada cenário de cada tipo de placa. No total, foram escolhidas 280 imagens para o conjunto de teste e 280 para o conjunto de validação.

Quadro 1 – Quantidade de imagens por cenário de cada classe

Classe/Cenário		Brasileira	Brasileira Mercosul	Argentina	Argentina Mercosul	Paraguaia	Paraguaia Mercosul	Chilena
Dia	1 no centro	8	12	12	18	8	14	20
	1 na esquerda	6	10	-	2	8	8	-
	2 no centro	6	-	10	2	8	-	-
	2 nas laterais	8	10	10	12	8	14	20
	1 atrás da roda	6	8	6	6	8	4	-
	1 no centro (outra cor)	6	-	2	-	-	-	-
Noite	1 no centro	8	12	12	32	8	14	30
	1 na esquerda	8	10	-	-	8	8	-
	2 no centro	4	-	10	4	8	-	-
	2 nas laterais	8	10	12	4	8	14	10
	1 atrás da roda	6	8	4	-	8	4	-
	1 no centro (outra cor)	6	-	2	-	-	-	-

Para ilustrar todos estes cenários, o Quadro 2 mostra um exemplo de imagem para cada cenário no caso da classe Brasileira.

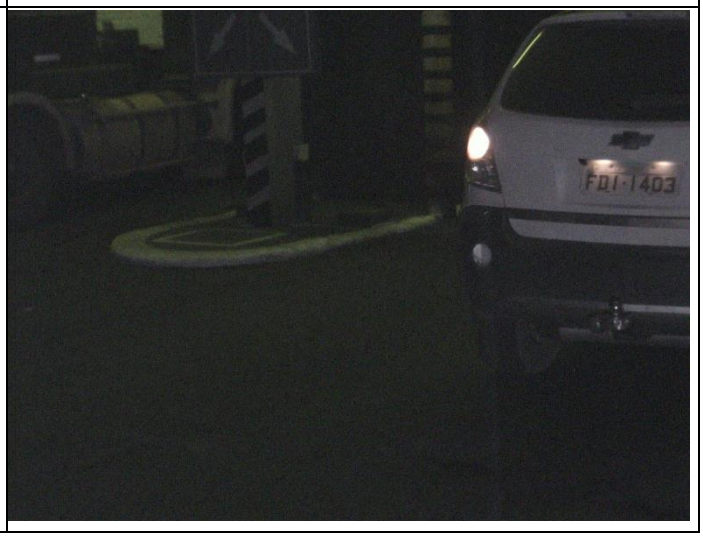
Quadro 2 - Exemplos dos cenários da classe Brasileira

Iluminação	**Posicionamento**	**Imagem**
Natural (diurna)	1 no centro	

	1 na esquerda	
	2 no centro	
	2 nas extremidades	

	1 atrás da roda	
	1 no centro (outra cor)	
Artificial (noturna)	1 no centro	

	1 na esquerda	
	2 no centro	
	2 nas extremidades	

	1 atrás da roda	
	1 no centro (outra cor)	

3.3.1.2 Conjunto de treino

Para este conjunto não houve cuidado para ter a mesma proporção de imagens para cada cenário, pois em muitos tipos de placa não sobraram quantidades suficientes de cada cenário para isso. Foram escolhidas 120 imagens de cada classe, totalizando 840 imagens no conjunto de treino.

3.3.2 ROTULAÇÃO DAS IMAGENS

Para realizar o treinamento dos modelos, uma vez que este treinamento é supervisionado, como justificado posteriormente no trabalho na seção 3.4.2, as imagens precisam ser rotuladas.

Para o treinamento do modelo YOLOv3 é necessário que o rótulo das imagens contenha a classe da placa contida na imagem e a sua localização na imagem, uma vez que esse modelo foi criado não apenas para classificar, mas também para localizar a posição da classe na imagem. Para o treinamento do modelo ResNet50 é necessário apenas a classe da placa presente na imagem como informação do rótulo da imagem.

Todas as imagens foram rotuladas manualmente através de uma ferramenta com interface visual na qual é possível visualizar a imagem e editar o posicionamento da placa a ser rotulada e escolher qual a sua classe. Essas informações foram salvas em arquivos do tipo XML, um para cada imagem. Esses arquivos foram utilizados para o treinamento tanto do modelo YOLOv3 quanto para o ResNet50.

3.4 TREINAMENTO SUPERVISIONADO

O treinamento é a etapa na qual os modelos aprendem a classificar as classes, que são, neste trabalho, os tipos de placa. É um processo no qual são feitas várias tentativas de treinamento, variando alguns parâmetros de treinamento, a fim de obter um modelo com um desempenho satisfatório.

Esta seção trata do quinto e sexto passo da metodologia apresentada, que são:

- Escolha de modelos de redes neurais
- Treinamento dos modelos

3.4.1 ESCOLHA DE MODELOS DE REDES NEURAIS

Neste trabalho foram escolhidos para o treinamento dois modelos que utilizam redes neurais, ambos conhecidos na literatura de aprendizado de máquina.

Um deles é o YOLO, acrônimo de *You Only Look Once*, o qual é amplamente utilizado para detecção de objetos em imagens, muitas vezes em tempo real. Ele não apenas classifica o objeto, mas também indica qual sua posição na imagem e o percentual de certeza que o objeto

é da classe identificada. Embora não seja necessário para este trabalho descobrir a posição da placa na imagem, este modelo tem se mostrado muito eficaz para a classificação de objetos.

A escolha desse modelo foi tomada justamente por ele aprender qual a localização da placa, pois, uma vez sabendo a localização, supôs-se que modelo teria mais facilidade em classificar a placa. Utilizou-se a versão 3 do YOLO (YOLOv3) pela facilidade de implementação em Python.

O outro modelo escolhido foi o ResNet, acrônimo de *Residual Neural Network*, que é um dos modelos de redes neurais convolucionais muito conhecidos na literatura por classificar eficazmente imagens. Utilizou-se neste trabalho a variação do modelo chamada ResNet50, que possui 50 camadas de profundidade e que teve mais facilidade de ser implementado.

3.4.2 TREINAMENTO DOS MODELOS

Nesta etapa, o treinamento supervisionado dos modelos é feito com base nos conjuntos de treino e de validação e em alguns parâmetros de treinamento. Variou-se os parâmetros de treinamento e verificou-se o aprendizado do modelo após o treinamento para encontrar os parâmetros que otimizam o aprendizado do modelo.

Neste trabalho, o treinamento é realizado por uma quantidade de Épocas pré-determinada.

O parâmetros que foram variados em ambos os modelos foram:

- Número de imagens do conjunto de treino de cada classe;
- Número de épocas de treinamento.

Existem outros parâmetros além destes e que são necessários definir para que o treinamento aconteça, mas estes não foram alterados ao longo do treinamento e não serão explicados aqui.

Neste trabalho as imagens são redimensionadas para 320 pixels de largura por 320 pixels de altura. Isto foi necessário para que fosse computacionalmente possível de realizar o treinamento.

3.4.2.1 Treinamento do modelo YOLOv3

Para o treinamento deste modelo foi feito um *fine-tuning*, isto é, foi feito a partir de um modelo treinado previamente, com os pesos dos nós da rede neural com valores já treinados. Os pesos deste modelo treinado previamente são do conjunto de dados COCO (Common Objects in Context) [21].

O modelo é treinado não só a descobrir quais classes estão presentes na imagem, mas também seu posicionamento. Uma vez que a saída do modelo é uma lista de objetos, cada um contendo sua posição, sua classe e seu percentual de certeza da classe, a função de Perda se baseia não apenas nas classes, mas também em seus posicionamentos.

Como para este trabalho só há interesse em descobrir qual é a classe da imagem, depois do treinamento, a avaliação deste se dá na comparação entre o objeto da lista cujo percentual de certeza da classe foi maior com a classe rotulada.

A métrica utilizada neste trabalho para avaliar o desempenho do modelo é a acurácia, que é a razão da quantidade de imagens que o modelo acertou a classe pela quantidade de imagens.

Inicialmente foram feitos alguns treinamentos apenas alterando o número de imagens e mantendo fixo o número de Épocas para avaliar o impacto do aumento da quantidade de imagens no modelo. O número de Épocas escolhido foi 8. O Quadro 3 contém as informações dos treinamentos realizados nesta etapa.

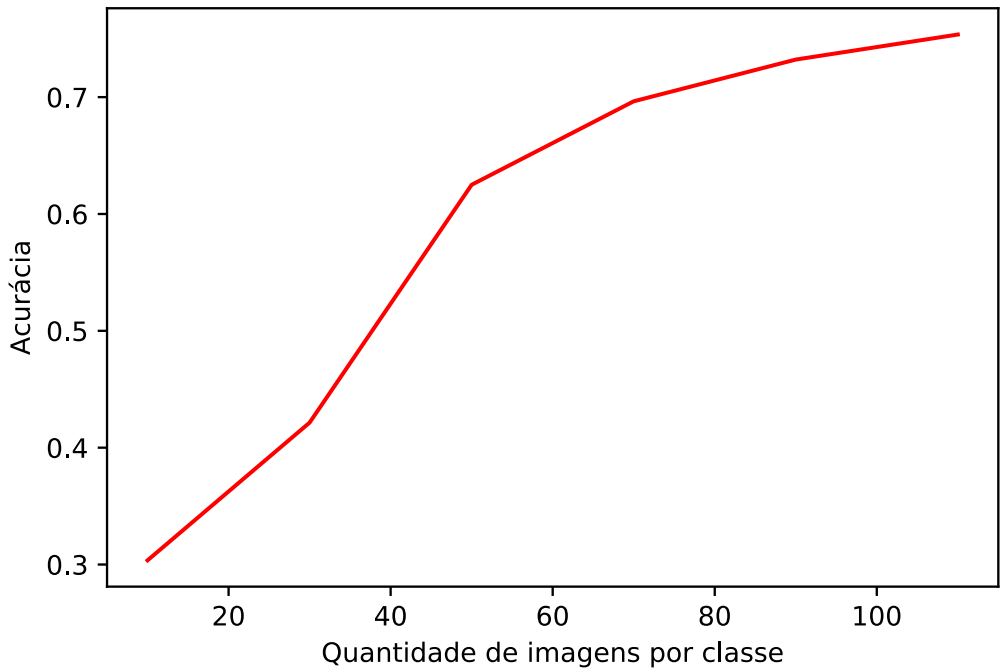
Quadro 3 – Acurácia dos treinamentos variando a quantidade de imagens por classe

Quantidade de imagens por classe	**Número de épocas**	**Acurácia do conjunto de validação**

10	8	0,303571
30	8	0,421428
50	8	0,625000
70	8	0,696428
90	8	0,732142
110	8	0,753571

O crescimento da acurácia no conjunto de validação pelo aumento da quantidade de imagens esperado é um crescimento logarítmico. Portando, haverá uma certa quantidade de imagens que não aumentará um valor significativo na acurácia do conjunto de validação. O crescimento pode ser observado na Figura 7.

Figura 7 – Acurácia do conjunto de validação pela quantidade de imagens por classe



Fonte: Autor, 2023.

A partir desta etapa, pode-se perceber que 110 imagens parecem estar próximas de chegar num ponto bom, onde o aumento da quantidade não irá aumentar significativamente a acurácia.

A próxima etapa foi treinar o modelo com todas as 120 imagens de cada classe, totalizando 840 imagens, e observar a cada época a acurácia do conjunto de treino e do conjunto de validação, a fim de encontrar a quantidade de épocas ideal para o treinamento do modelo, sem acontecer *overfitting*.

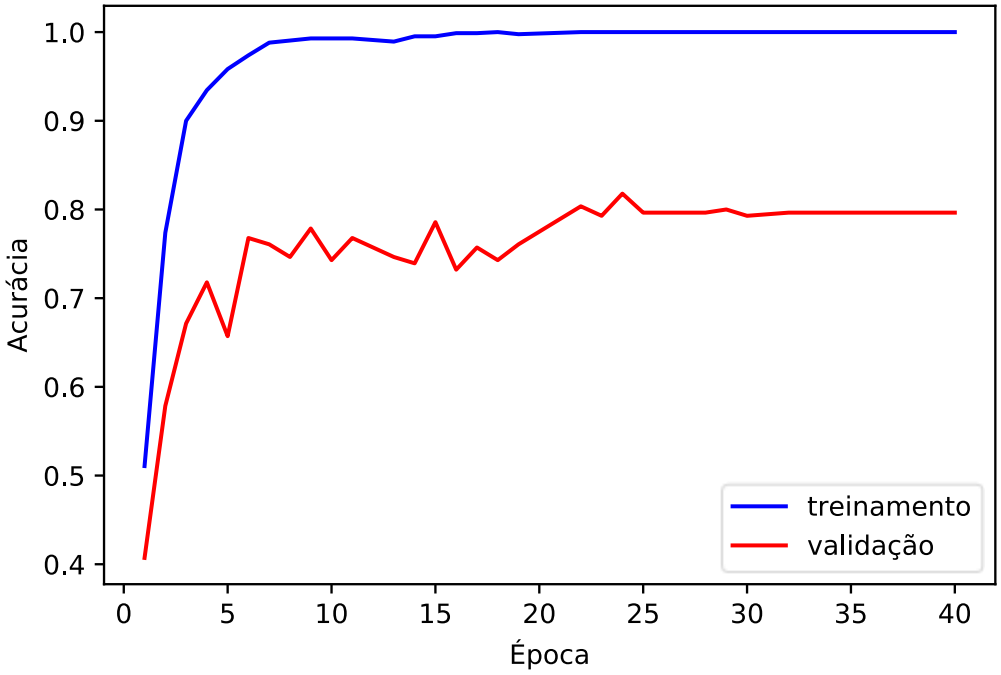
O valor escolhido de épocas foi 40 e o resultado deste treinamento pode ser observado no Quadro 4 e na Figura 8.

Quadro 4 – Acurácias dos conjuntos de treino e de validação no treinamento de 840 imagens

Época	**Acurácia do conjunto de treinamento**	**Acurácia do conjunto de validação**
1	0,510714	0,407143
2	0,773810	0,578571
3	0,900000	0,671429
4	0,934524	0,717857
5	0,958333	0,657143
6	0,973810	0,767857
7	0,988095	0,760714
8	0,990476	0,746429
9	0,992857	0,778571
10	0,992857	0,742857
11	0,992857	0,767857
12	0,992857	0,767857
13	0,989286	0,746429
14	0,995238	0,739286
15	0,995238	0,785714
16	0,998810	0,732143
17	0,998810	0,757143
18	1,000000	0,742857
19	0,997619	0,760714
20	0,997619	0,760714

21	0,997619	0,760714
22	1,000000	0,803571
23	1,000000	0,792857
24	1,000000	0,817857
25	1,000000	0,796429
26	1,000000	0,796429
27	1,000000	0,796429
28	1,000000	0,796429
29	1,000000	0,800000
30	1,000000	0,792857
31	1,000000	0,792857
32	1,000000	0,796429
33	1,000000	0,796429
34	1,000000	0,796429
35	1,000000	0,796429
36	1,000000	0,796429
37	1,000000	0,796429
38	1,000000	0,796429
39	1,000000	0,796429
40	1,000000	0,796429

Figura 8 – Gráfico da acurácia dos conjuntos de treinamento e validação pela quantidade de épocas para o modelo YOLOv3 treinado com 840 imagens



Fonte: Autor, 2023.

Pode-se observar que a acurácia estabiliza após a 22ª Época por volta de 80%, quando o número de imagens é 840.

Mais um treinamento foi feito, desta vez com 190 imagens para as classes Brasileira, Brasileira Mercosul e Paraguaia e 120 imagens para as outras 4 classes, totalizando 1050 imagens. A quebra do balanceamento das imagens por classe foi feita por conta de não haver mais imagens destas outras 4 classes. Foi repetido o mesmo processo de que foi feito no treinamento do modelo utilizando 120 imagens de cada classe por 40 Épocas. O Quadro 5 e a Figura 9 mostram o desempenho deste treinamento.

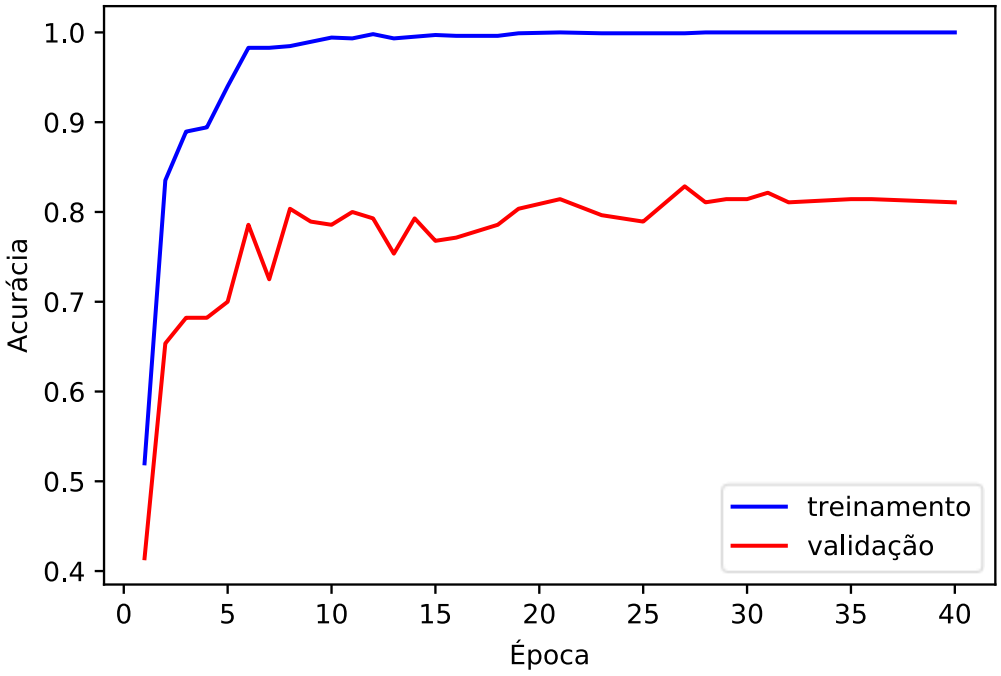
Quadro 5 - Acurácias dos conjuntos de treino e de validação no treinamento de 1050 imagens

Época	**Acurácia do conjunto de treino**	**Acurácia do conjunto de validação**
1	0,520000	0,414286

2	0,835238	0,653571
3	0,889524	0,682143
4	0,894286	0,682143
5	0,940000	0,700000
6	0,982857	0,785714
7	0,982857	0,725000
8	0,984762	0,803571
9	0,989524	0,789286
10	0,994286	0,785714
11	0,993333	0,800000
12	0,998095	0,792857
13	0,993333	0,753571
14	0,995238	0,792857
15	0,997143	0,767857
16	0,996190	0,771429
17	0,996190	0,771429
18	0,996190	0,785714
19	0,999048	0,803571
20	0,999048	0,803571
21	1,000000	0,814286
22	1,000000	0,814286
23	0,999048	0,796429
24	0,999048	0,796429
25	0,999048	0,789286
26	0,999048	0,789286
27	0,999048	0,828571
28	1,000000	0,810714
29	1,000000	0,814286
30	1,000000	0,814286
31	1,000000	0,821429
32	1,000000	0,810714
33	1,000000	0,810714

34	1,000000	0,810714
35	1,000000	0,814286
36	1,000000	0,814286
37	1,000000	0,814286
38	1,000000	0,810714
39	1,000000	0,810714
40	1,000000	0,810714

Figura 9 – Gráfico da acurácia dos conjuntos de treinamento e validação pela quantidade de épocas para o modelo YOLOv3 treinado com 1050 imagens



Fonte: Autor, 2023.

Podemos observar que a acurácia no conjunto de validação estabilizou após a 28ª Época em torno de 81%, ou seja, com uma acurácia em torno de 1% maior do que com apenas 840 imagens.

3.4.2.2 Treinamento do ResNet50

Este modelo de rede neural convolucional tem como entrada a imagem e como saída a classe calculada. Neste modelo foi realizado dois tipos de treinamentos: o treinamento não utilizando *fine-tuning* e o treinamento utilizando *fine-tuning*, através dos pesos treinados previamente com o conjunto de dados *ImageNet* [22].

Em cada um desses tipos foram variadas as quantidades de imagens e quantidade de épocas, também variados alguns parâmetros de treinamento.

O resultado do treinamento para todos os casos foi o mesmo: os modelos alcançavam 100% de acurácia no conjunto de treino e mantinham em torno de 16% a acurácia do conjunto de validação. 16% é aproximadamente a mesma acurácia que teríamos se usássemos um modelo que sorteia uma classe quando recebe uma imagem de entrada.

Esse resultado do treinamento significa que o algoritmo não aprendeu a classificar corretamente as classes, ele aprendeu apenas a classificar as placas das imagens do conjunto de treinamento, mas quando são colocadas outras imagens que contém as placas para o modelo classificar, ele tem uma acurácia de um sétimo. Como sete são os números de classe, essa acurácia indica que não houve generalização do modelo, pois ele aprendeu especificidades das imagens do conjunto de treinamento que não valem para os conjuntos de validação. Isso indica *overfitting*.

4 RESULTADOS

Uma vez que o modelo que realmente aprendeu a classificar as placas foi apenas o YOLOv3, os resultados aqui mostrados serão apenas utilizando este modelo.

Nesta etapa foram utilizados o conjunto de teste, para verificar o desempenho do modelo proposto. Foram escolhidos os modelos que tiveram um bom desempenho no treinamento para avaliar o desempenho no conjunto de teste. Estes modelos são os que foram treinados com 840 imagens e 22 Épocas e o com 1050 imagens e 29 Épocas.

O Quadro 6 mostra os modelos e suas acurácias em todos os 3 conjuntos.

Quadro 6 – Acurácia de cada modelo treinado

Modelo	**Acurácia no conjunto de treino**	**Acurácia no conjunto de validação**	**Acurácia no conjunto de teste**
YOLOv3 treinado com 840 imagens e 22 Épocas	1,000000	0,803571	0,792857
YOLOv3 treinado com 1050 imagens e 29 Épocas	1,000000	0,814286	0,857142

Utilizando o modelo com o melhor desempenho, treinado com 1050 imagens e 29 Épocas, foi possível criar a matriz de confusão no conjunto de validação, como mostrado no Quadro 7, e no conjunto de teste, como mostrado no Quadro 8. A matriz de confusão mostra para cada uma das sete classes de um conjunto quais classes foram preditas pelo modelo treinado. As linhas representam a classe verdadeira e as colunas representam a classe predita pelo modelo. A última coluna, com o título “Nenhuma”, representa quando o modelo não predisse nenhuma classe. Para cada classe de cada conjunto havia 40 imagens.

Quadro 7 – Matriz de Confusão do conjunto de Validação

	Brasileira	Brasileira Mercosul	Argentina	Argentina Mercosul	Paraguaia	Paraguaia Mercosul	Chilena	Nenhuma
Brasileira	32	3	1	0	0	1	1	2
Brasileira Mercosul	1	38	0	0	0	1	0	0
Argentina	3	0	34	0	2	0	0	1
Argentina Mercosul	0	11	0	27	0	2	0	0
Paraguaia	5	0	1	0	32	0	0	2

Paraguaia Mercosul	0	4	0	0	4	32	0	0
Chilena	0	1	4	0	0	2	33	0

Quadro 8 – Matriz de Confusão do conjunto de Teste

	Brasileira	Brasileira Mercosul	Argentina	Argentina Mercosul	Paraguaia	Paraguaia Mercosul	Chilena	Nenhuma
Brasileira	34	2	3	0	0	0	0	1
Brasileira Mercosul	0	40	0	0	0	0	0	0
Argentina	0	1	39	0	0	0	0	0
Argentina Mercosul	0	9	1	27	0	3	0	0
Paraguaia	1	0	1	0	37	0	0	1
Paraguaia Mercosul	0	4	0	0	3	33	0	0
Chilena	4	2	1	2	0	1	30	0

Nota-se que quando a classe verdadeira representa placas do novo padrão Mercosul o modelo em geral erra predizendo uma classe que também é do novo padrão Mercosul. Analogamente, o mesmo ocorre quando a classe verdadeira representa placas do padrão antigo, com o modelo errando geralmente ao predizer classes que também são do padrão antigo.

Para exemplificar os casos em que isso ocorreu, as figuras Figura 11, Figura 12, Figura 14 e Figura 15 mostram imagens que foram incorretamente preditas pelo modelo. As figuras Figura 10 e Figura 13 mostram imagens classificadas corretamente.

Figura 10 – Imagem predita corretamente como classe Brasileira Mercosul

Figura 11 – Imagem predita incorretamente como classe Brasileira Mercosul sendo a sua classe real Argentina Mercosul

Figura 12 – Outra imagem predita incorretamente como classe Brasileira Mercosul sendo a sua classe real Argentina Mercosul

Figura 13 - Imagem predita corretamente como classe Argentina

Figura 14 - Imagem predita incorretamente como classe Argentina sendo a sua classe real Paraguaia

Figura 15 - Imagem predita incorretamente como classe Brasileira sendo a sua classe real Paraguaia

5 CONCLUSÃO

Foi possível utilizar um modelo de rede neural para classificar placas presentes em imagens, mesmo estando as placas contidas em uma parte pequena da imagem. Entretanto apenas o YOLOv3 conseguiu aprender satisfatoriamente. Provavelmente isso aconteceu pelo fato deste modelo ser usado também para encontrar a posição de um objeto na imagem, o que é o caso, porque precisa-se encontrar, além da classe, a sua posição na imagem, algo que o ResNet50 não faz.

Embora modelos de redes neurais convolucionais sejam utilizados para problemas de classificação de imagem, neste projeto o ResNet50 não conseguiu aprender bem a classificar as imagens. Um dos prováveis motivos seja porque a placa, que é o objeto que define a classe da imagem, está presente em apenas uma pequena parte da imagem. Uma possibilidade para futuros trabalhos é fazer um pré-processamento na imagem, recortando o local onde estão as placas da imagem e usar esses recortes como imagens para o treinamento. Talvez, desse modo, o ResNet50 consiga aprender corretamente. Utilizar esse modelo na prática envolveria fazer esse pré-processamento das imagens ser automático para colocar a imagem recordada como entrada no modelo. Uma possível solução seria usar as caixas delimitadoras que o YOLOv3 foi treinado a encontrar, as quais recortam a imagem apenas nas regiões das placas veiculares, e usar essa imagem recortada como entrada para a ResNet50.

Pelos resultados dos treinamentos, podemos perceber que com mais imagens de treinamento o modelo YOLOv3 tem um melhor desempenho, ou seja, a acurácia nos conjuntos de validação e teste aumentam. Portanto, com mais imagens de treino o YOLOv3 conseguirá um desempenho ainda maior, como 85%, porém não se espera muito mais que isso, pois vimos que o aumento da acurácia dos conjuntos de teste entre o modelo treinado com 840 imagens e o modelo treinando com 1050 imagens foi bem pequeno, ou seja, está chegando no ponto onde o aumento da quantidade de imagem já não influencia muito na acurácia.

A acurácia de 81% obtida no conjunto de teste é suficiente para o auxílio em alguns processos da indústria como citado na introdução. Se um software da indústria que utilizaria este modelo utilizasse da saída do YOLOv3 para fazer uma ordem decrescente dos tipos de placa que o modelo indicou na imagem, isso possivelmente ajudaria ainda mais a escolher o

algoritmo LPR adequado ao reconhecimento de caracteres. Mesmo que não acertasse qual seria o primeiro algoritmo a usar na imagem, talvez acertasse no segundo algoritmo.

Mesmo satisfeito com o resultado do treinamento, creio que haja muito espaço para otimização utilizando o YOLOv3. Nesse sentido, sugere-se alguns trabalhos futuros que possam fazer o aumento da dimensão da imagem para o treinamento pode resultar num aumento da acurácia. Como houve a necessidade de diminuir a dimensão das imagens de 1280x960 pixels para 320x320 pixels, a placa, que já era apenas uma pequena parte da imagem, ficou ainda menor e consequentemente perdeu muita informação em seus pixels que poderiam ser uteis para o modelo aprender a diferenciar melhor os tipos de placa, o que aumentaria a acurácia. Porém, para isso, necessitaria de processadores mais potentes dos que os utilizados neste trabalho.

Outra estratégia que poderia ser utilizada para aumentar a acurácia é, depois do modelo treinado, retornar qual a classe e a posição da placa calculada por ele, recortar a imagem apenas na posição da placa encontrada e utilizar este recorte de imagem num software de OCR para, através de sua saída, que seria uma sequência de caracteres, verificar se essa sequência é compatível com o padrão de caracteres do tipo da placa que o YOLOv3 teria retornado.

Os próximos passos para trabalhos futuros seria buscar resolver o problema de falta de generalização do ResNet50, utilizando um processo automático de recortar as imagens das placas anterior ao treinamento. Acredito que isto iria ajudar a controlar o *overfitting* e também à obter uma acurácia muito boa.

Outra sugestão para trabalhos futuros seria testar utilizar apenas classes de placas do padrão Mercosul ou de apenas placas que não são do padrão Mercosul. Além disso, poderia se fazer o trabalho em cima de uma classificação binária: placa do padrão Mercosul ou placa de outros padrões.

REFERÊNCIAS

[1] EasyGate. [*S. l.*], 2021. Disponível em: https://gttlogistics.com.br/easygate/. Acesso em: 21 jan. 2023.

[2] Segurança Eletrônica Tecnologia OCR traz segurança aos portos do Brasil. [*S. l.*], 2017. Disponível em: https://revistasegurancaeletronica.com.br/tecnologia-ocr-traz-seguranca-aos-portos-do-brasil/. Acesso em: 21 jan. 2023.

[3] K. T., Thomas; J, vaijayanthimala. (2014). A Review of Automatic License Plate Detection Using Edge Detection Methods. **ResearchGate**, Mahatma Gandhi University, 2014. Disponível em: https://www.researchgate.net/publication/330900287_A_Review_of_Automatic_License_Plate_Detection_Using_Edge_Detection_Methods. Acesso em: 21 jan. 2023.

[4] K. K. Kim; K. I. Kim; J. B. Kim; H. J. Kim. Learning-based approach for license plate recognition. **IEEE**. *Neural Networks for Signal Processing X. Proceedings of the 2000 IEEE Signal Processing Society Workshop*, Sydney, NSW, Australia, 2000. Disponível em: https://ieeexplore.ieee.org/abstract/document/890140/. Acesso em: 21 jan. 2023.

[5] JANNINK, Alexander. Commentary on Automatic Number Plate Recognition Claims and Applications. [*S. l.*] 15 dez. 2017. Disponível em: https://www.linkedin.com/pulse/commentary-automatic-number-plate-recognition-claims-jannink/. Acesso em: 21 jan. 2023.

[6] ALPAYDIN, Ethem. Introduction to Machine Learning, second edition. MIT Press, 2010. Disponível em: https://www.cmpe.boun.edu.tr/~ethem/i2ml2e/. Acesso em: 21 jan. 2023.

[7] GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. Deep Learning. MIT Press, 2016. Disponível em: https://books.google.com.br/books/about/Deep_Learning.html?hl=pt-BR&id=Np9SDQAAQBAJ&redir_esc=y. Acesso em: 21 jan. 2023.

[8] LECUN, Y.; BOTTOU, L.; BENGIO, Y.; HAFFNER, P.. Gradient-based learning applied to document recognition. **Proceedings Of The Ieee**, [S.L.], v. 86, n. 11, p. 2278-2324, nov. 1998. IEEE. Disponível em: http://dx.doi.org/10.1109/5.726791. Acesso em: 21 jan. 2023.

[9] SANGHVI, Kavish. Image Classification Techniques. Disponível em: https://medium.com/analytics-vidhya/image-classification-techniques-83fd87011cac. Acesso em: 21 jan. 2023.

[10] JAWALE, Anupama. Comparison of Image Classification Techniques: Binary and Multiclass using Convolutional Neural Network and Support Vector Machines. **INFOCOMP Journal of Computer Science**, [*S. l.*], 2019. Disponível em: https://infocomp.dcc.ufla.br/index.php/infocomp/article/view/618. Acesso em: 21 jan. 2023.

[11] Residual Networks (ResNet) – Deep Learning. Disponível em: https://www.geeksforgeeks.org/residual-networks-resnet-deep-learning/. Acessado em: 22 jan. 2023.

[12] CHABLANI, Manish. YOLO — You only look once, real time object detection explained. Disponível em: https://towardsdatascience.com/yolo-you-only-look-once-real-time-object-detection-explained-492dc9230006. Acessado em: 22 jan. 2023.

[13] KELTA, Zoumana. YOLO Object Detection Explained. Disponível em: https://www.datacamp.com/blog/yolo-object-detection-explained. Acessado em: 22 jan. 2023.

[14] GESSONI, Lucas. As diferentes formas de aprendizado de máquina. Disponível em: https://www.eldorado.org.br/blog/as-diferentes-formas-de-aprendizado-de-maquina/. Acessado em: 23 jan. 2023.

[15] QUEGUINER, Jean-Louis. What does training neural networks mean. Disponível em: https://blog.ovhcloud.com/what-does-training-neural-networks-mean/. Acessado em: 23 jan. 2023.

[16] Capítulo 28 – Usando Early Stopping Para Definir o Número de Épocas de Treinamento. Disponível em: https://www.deeplearningbook.com.br/usando-early-stopping-para-definir-o-numero-de-epocas-de-treinamento/. Acesso em: 23 jan. 2023.

[17] BROWNLEE, Jason. How to Checkpoint Deep Learning Models in Keras. Disponível em: https://machinelearningmastery.com/check-point-deep-learning-models-keras/. Acessado em: 23 jan. 2023.

[18] YOLO3 (Detection, Training, and Evaluation). [*S. l.*], 31 jan. 2020. Disponível em: https://github.com/experiencor/keras-yolo3/. Acesso em: 31 out. 2022.

[19] KERAS-RESNET. [*S. l.*], 5 jul. 2015. Disponível em: https://github.com/raghakot/keras-resnet/. Acesso em: 31 out. 2022.

[20] LabelImg. [*S. l.*], 15 set. 2022. Disponível em: https://github.com/heartexlabs/labelImg. Acesso em: 31 out. 2022.

[21] COCO Dataset. [*S. l.*]. Disponível em: https://cocodataset.org/. Acesso em: 11 fev. 2023.

[22] ImageNet. [*S. l.*]. Disponível em: https://www.image-net.org/. Acesso em: 11 fev. 2023.