UNIVERSIDADE PRESBITERIANA MACKENZIE
FACULDADE DE COMPUTAÇÃO E INFORMÁTICA
TECNOLOGIA EM CIÊNCIAS DE DADOS

PROJETO APLICADO II

Projeto Sentinela: Automatização do Controle de Acesso Veicular em Condomínios.

PROFESSOR: FELIPE ALBINO DOS SANTOS

GRUPO:

ALINE CORRÊA – 10414773 – 10414773@mackenzista.com.br
ISAQUE PIMENTEL – 10415608 – 10415608@mackenzista.com.br
MAIKI SOARES – 10415481 – 10415481@mackenzista.com.br
VANESSA CORDEIRO – 10415118 – 10415118@mackenzista.com.br

São Paulo
2024

SUMÁRIO

1. APRESENTAÇÃO DO GRUPO DE TRABALHO 4

2. PREMISSAS DO PROJETO 4

2.1. ESCOLHA DA EMPRESA 4

2.2. AREA DE ATUAÇÃO 4

3. APRESENTAçÃO DOS DADOS 5

3.1. DADOS DE TREINAMENTO E TESTE DO MODELO 5

3.2. DADOS DO SISTEMA DE SEGURANÇA DO CONDOMÍNIO 6

3.3. DADOS DE SAÍDA 6

4. OBJETIVO E METAS 7

5. CRONOGRAMA DE ATIVIDADES 7

6. DEFINIÇÃO DE BIBLIOTECAS PYTHON 8

7. DEFINIÇÃO DA BASE DE DADOS 10

8. ANÁLISE EXPLORATÓRIA 12

9. TRATAMENTO DA BASE DE DADOS 14

10. METODOS ANALÍTICOS 14

11. DESCRIÇÃO DA METODOLOGIA DA ACURÁCIA 16

12. CONSOLIDAÇÃO DOS RESULTADOS DO MÉTODO ANALÍTICO 18

12.1. PROCESSAMENTO DOS DADOS 18

12.2. TREINAMENTO DO MODELO 18

13. REVISÃO DO DESEMPENHO DO MÉTODO ANALÍTICO 20

14. DESCRIÇÃO DE RESULTADOS PRELIMINARES 22

14.1. DETECÇÃO DE PLACAS VEICULARES. 22

14.2. RECONHECIMENTO DE PLACAS VEICULARES. 23

14.3. MODELO DE NEGÓGIO.....24

15. ELABORAÇÃO DO ESBOÇO DO STORYTELLING 25

1. APRESENTAÇÃO DO GRUPO DE TRABALHO

Somos um grupo de alunos de Ciências de Dados desenvolvendo um projeto de Aprendizagem de Máquina com o objetivo de criar uma solução para um problema de dados com aplicação comercial. Com esse objetivo, criamos o Sentinela, um sistema de detecção e conversão de placas a partir de fotos da dianteira de veículos. Após a divulgação do nosso produto, uma administradora de condomínios nos contactou solicitando implementação desse sistema.

2. PREMISSAS DO PROJETO

O sistema de reconhecimento de placas veiculares será implementado em um dos condomínios de casas do cliente para automatizar o controle de entrada e saída de veículos. Com o intuito de melhorar a segurança do condomínio, será registrado todas as placas dos veículos que entram e saem, além de identificar veículos não autorizados.

Um dos pré-requisito do cliente é de que o serviço deve ser capaz de integrar-se com os sistemas de segurança existentes no condomínio, como câmeras de vigilância e sistemas de controle de acesso.

2.1. ESCOLHA DA EMPRESA

A VAIM Administradora de Condomínios é uma empresa líder no setor de administração e gestão condominial na cidade de São José dos Campos - SP, dedicada a proporcionar conforto e segurança aos seus moradores. Fundada em 1988 com a visão de oferecer serviços de excelência, a VAIM faz a gestão de mais de 10 condomínios de casas e se destaca pela expertise em proporcionar ambientes residenciais seguros, por isso o interesse em adquirir sistema de reconhecimento de placa veicular.

2.2. AREA DE ATUAÇÃO

A VAIM atua na administração de condomínios residenciais de alto padrão na região do Vale do Paraíba, onde a qualidade de vida e a segurança dos moradores são prioridades absolutas. Os serviços da administradora abrangem desde a gestão

financeira e administrativa até a implementação de soluções tecnológicas inovadoras, com o objetivo a experiência dos residentes e otimizar a operação do condomínio.

Com um aumento do fluxo de entrada e saída de veículos e por consequência do tempo de espera, os moradores do condomínio se reuniram com a administradora para buscar uma solução eficiente para gestão de veículos mantendo o padrão de segurança da empresa.

Como é uma empresa que investe em tecnologia e inovação, já implementou um sistema de reconhecimento facial na portaria. Buscando superar as expectativas dos moradores, tomou a decisão de contratar agora a implementação do reconhecimento de placas veiculares.

3. APRESENTAÇÃO DOS DADOS

Eis os dados a serem utilizados ao longo do processo de implementação do Sentinela, sistema de reconhecimento de placas veiculares.

3.1. DADOS DE TREINAMENTO E TESTE DO MODELO

Os dados abaixo são aqueles mais relevantes no processo de treinamento do modelo:

- Imagens de placas veiculares: Conjunto de imagens contendo placas veiculares capturadas pelas câmeras do condomínio do cliente. Elas devem conter uma variedade de condições: diferentes ângulos, iluminações e condições climáticas.
- Informações dos veículos autorizados: Conjunto de informações sobre veículos autorizados a entrar no condomínio como modelo, cor e placa.
- Imagens de veículos não autorizados: Conjunto de imagens de veículos não autorizados que tentaram acessar o condomínio. É importante treinar o modelo a identificar e a sinalizar veículos suspeitos.

3.2. DADOS DO SISTEMA DE SEGURANÇA DO CONDOMÍNIO

Os dados abaixo são aqueles encontrados no sistema de segurança do condomínio do cliente:

- Registros de acesso: Histórico de entrada e saída de veículos no condomínio, incluindo informações como horários, placas dos veículos e ações tomadas pela equipe de segurança.
- Dados sobre câmeras de vigilância: Informações sobre as câmeras de vigilância do condomínio, como localização, ângulos de visão e qualidade das imagens. Esses dados são importantes para o funcionamento eficaz do sistema de reconhecimento de placas.

3.3. DADOS DE SAÍDA

Os dados abaixo são retornados pelo sistema de identificação de placas veiculares integrado ao sistema de segurança:

- Placas reconhecidas: Saída do sistema que identifica e reconhece as placas dos veículos capturados pelas câmeras de vigilância. Essas informações podem incluir a placa identificada, a confiança da identificação e o horário da captura.
- Alertas de segurança: Notificações geradas pelo sistema em caso de detecção de veículos não autorizados ou comportamento suspeito. Esses alertas podem conter informações sobre o veículo e ações recomendadas para a equipe de segurança do condomínio.

Com esses dados de treinamento, do sistema de segurança e de saída, será possível construir um sistema robusto que atenda as especificações da VAIM Administradora de Condomínios, melhorando a segurança e eficiência operacional.

4. OBJETIVO E METAS

Como grupo de alunos de Ciência de Dados, nosso **objetivo** do Projeto Aplicado II é *Desenvolver um modelo de reconhecimento de placas veiculares com alta precisão.*

Para alcançar esse objetivo, o modelo a ser desenvolvido precisa realizar as seguintes tarefas:

1. *Detectar placas veiculares em imagens de carros*.
2. *Aplicar processamento de imagem e OCR para extrair os caracteres da placa*.

As ações a serem realizadas pelo modelo serão as **metas** do nosso projeto.

A partir desse modelo de reconhecimento de placas, as próximas etapas na implementação efetiva do sistema são:

- Integrar o sistema de reconhecimento de placas com os sistemas de segurança existentes no condomínio.
- Realizar testes para validar a eficácia do sistema.
- Treinar a segurança do condomínio no uso do novo sistema.

5. CRONOGRAMA DE ATIVIDADES

Diante das próximas etapas e do objetivo do Projeto Aplicado II, as tarefas necessárias foram executadas de acordo com o cronograma de atividades proposto abaixo, na Figura 3. Além disso, foram indicados também os responsáveis pela execução de cada tarefa.

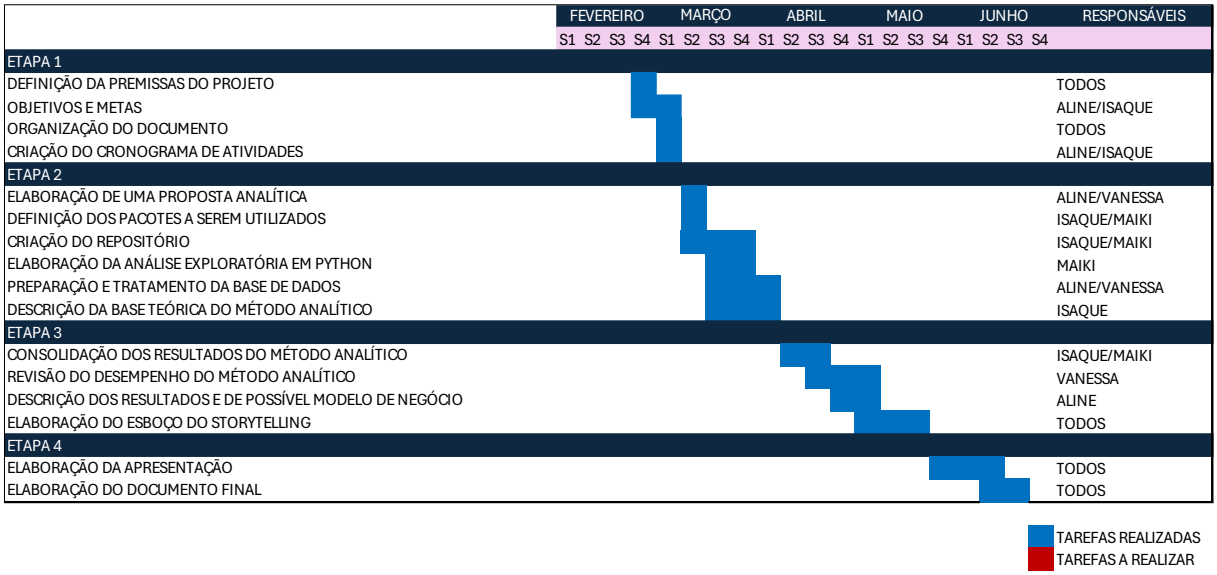


Figura 1 - Cronograma de atividades.
Fonte: Elaboração própria (2024).

6. DEFINIÇÃO DE BIBLIOTECAS PYTHON

Para desenvolver um sistema de reconhecimento de placas veiculares, existem algumas bibliotecas Python que podem ser úteis nessa tarefa:

- **OpenCV**: biblioteca de processamento de imagem, usada no pré-processamento de imagens, detecção de contorno e segmentação.
- **TensorFlow**: framework de aprendizado de máquina e deep learning usado no treinamento de modelo de reconhecimento de imagens.
- **PyTesseract**: interface para biblioteca OCR Tesseract, útil para reconhecer os caracteres das placas.
- **Matplotlib** e **Seaborn**: bibliotecas de visualização de dados e análise de resultados.
- **Numpy** e **Pandas**: bibliotecas de manipulação e análise de dados padrão em Python.

Os scripts Python que foram utilizados no projeto podem ser encontrados no repositório Github abaixo:

https://github.com/isaque-pimentel/projeto-aplicado-II

Segue por exemplo uma extração do README do Projeto Aplicado II:


README

Projeto Aplicado II

Projeto Sentinela: Automatização do Controle de Acesso Veicular em Condomínios

Grupo:

- ALINE CORRÊA – 10414773 – 10414773@mackenzista.com.br
- ISAQUE PIMENTEL – 10415608 – 10415608@mackenzista.com.br
- MAIKI SOARES – 10415481 – 10415481@mackenzista.com.br
- VANESSA CORDEIRO – 10415118 – 10415118@mackenzista.com.br

Apresentação do Grupo

Somos alunos de Ciências de Dados trabalhando em um projeto de Aprendizado de Máquina para resolver problemas de dados com aplicação comercial. Desenvolvemos o **Sentinela**, um sistema para detectar e converter placas de veículos a partir de fotos da frente dos veículos. Após a divulgação do produto, uma administradora de condomínios (empresa fictícia) nos contatou interessada em implementar o sistema.

Figura 2 – Captura de tela do README do Projeto Aplicado II no Github.
Fonte: Elaboração própria (2024).

7. DEFINIÇÃO DA BASE DE DADOS

Para implementar esse sistema, foi feita uma busca na internet por uma base de dados de imagens completa. A base de dados escolhida foi um *dataset* fornecido pelo Kaggle, intitulado *Car License Plate*, que pode ser encontrado no endereço abaixo:

`https://www.kaggle.com/datasets/andrewmvd/car-plate-detection/data`

Esse *dataset* é uma coleção abrangente de 433 imagens contendo veículos automotivos de marcas variadas com placas veiculares de identificação de diversos países. Este conjunto de dados é uma fonte valiosa para projetos de reconhecimento de placas veiculares sobre o qual criaremos nosso modelo de detecção para localizar onde se encontram as placas dentro das imagens.

Descrição da Base de Dados

A base de dados contém imagens de veículos automotivos capturadas de diversas perspectivas e condições de iluminação. Cada imagem é um arquivo PNG que está acompanhada de anotações, no formato XML, indicando a localização das placas de identificação dos veículos.

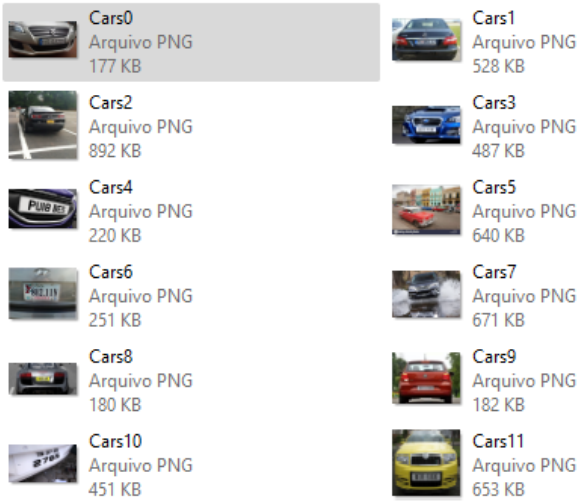


Figura 3 – Amostra das imagens de veículos automotivos no formato PNG.
Fonte: Elaboração própria (2024).

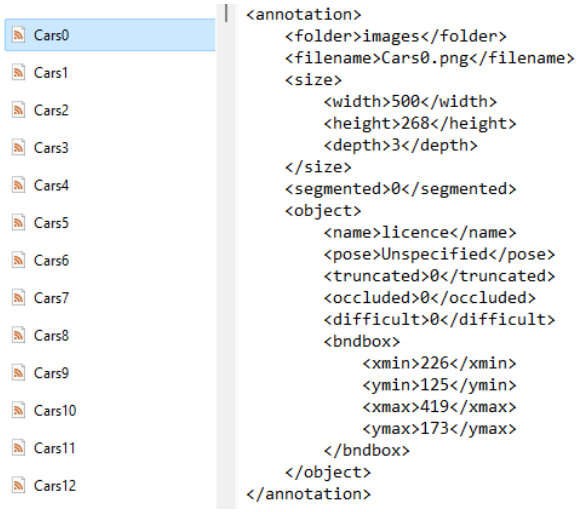


Figura 4 – Amostra das anotações da localização das placas no formato XML.
Fonte: Elaboração própria (2024).

8. ANÁLISE EXPLORATÓRIA

Proposta da Análise Exploratória

Para esse *dataset*, a proposta de análise exploratória é a seguinte:

- Exploração das imagens: iniciar a análise exploratória examinando de maneira aleatória algumas imagens do *dataset* para entender a diversidade dos veículos e condições de captura.
- Distribuição das classes de veículos: verificar a distribuição dos tipos de veículos presentes no *dataset* para entender se há um desequilíbrio entre as classes.
- Tamanho das imagens: analisar o tamanho médio das imagens para determinar a resolução mais comum e os potenciais desafios de pré-processamento.
- Visualização das anotações: visualizar algumas imagens com suas respectivas anotações para verificar a qualidade e precisão das marcações das placas de identificação.
- Análise de iluminação: avaliar a distribuição das condições de iluminação nas imagens para entender como isso pode afetar a detecção das placas.
- Pré-processamento das imagens: Explorar técnicas de pré-processamento, como ajuste de contraste e equalização de histograma, para melhorar a qualidade das imagens e facilitar a detecção das placas.

Com essa análise, seremos capazes de obter insights para o desenvolvimento do sistema de reconhecimento de placas veiculares a partir dessa base *Car License Plate*.

Compreensão da Base de Dados

Segue abaixo alguns exemplos das imagens de veículos encontrados na nossa base de dados:

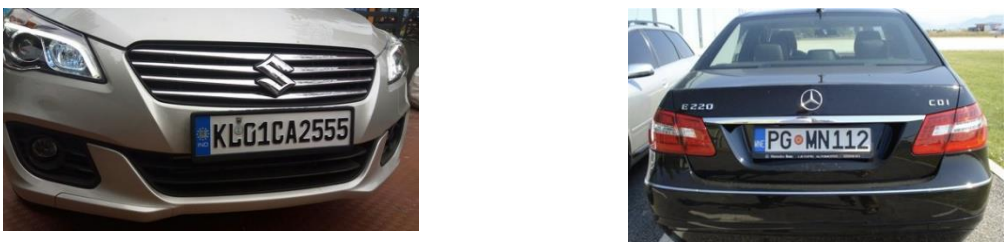


Figura 5 – Amostra Cars0 e Cars1 dentro do base de dados.
Fonte: *Car License Plate Dataset*

A identificação da localização das placas foi um processo manual realizado previamente e a posição das placas foi salva em arquivos de anotação no formato XML. É necessário transformar o formato da posição das placas (aqui chamada de *labels*) do formado XML para um formato que possa ser lido pelo modelo de detecção (no caso, algo do tipo np.array). Foi realizado um *parsing* dos dados no XML e uma conversão para um único arquivo csv. Esse procedimento foi salvo no arquivo *parse_annotations.py*.

Antes de realizar a Análise Exploratória, podemos realizar uma verificação se o quadro da etiqueta foi corretamente assinalado para uma determinada imagem. O usuário poderia escolher uma imagem para adicionar o quadro da etiqueta em amarelo. Esse procedimento foi salvo no arquivo *verify_image.py*. Por exemplo, no arquivo Car108.png temos um veículo com uma etiqueta corretamente assinalada sobre a placa, enquanto no arquivo Car126.png temos um veículo com uma etiqueta bem maior que a própria placa.

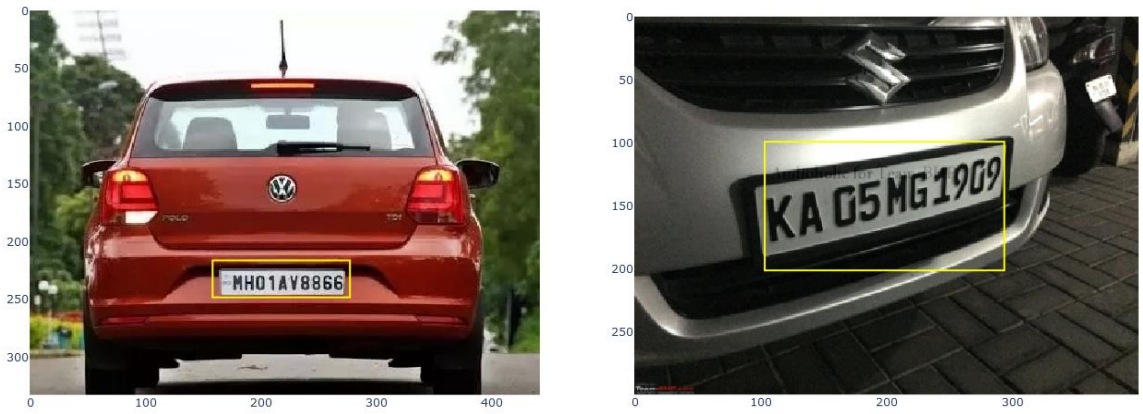


Figura 6 – Amostra Cars108.png e Cars126.png com o quadro da etiqueta em amarelo.
Fonte: *Car License Plate Dataset*

9. TRATAMENTO DA BASE DE DADOS

Para o tratamento da base de dados em um projeto de Controle de Acesso Veicular em Condomínios, é importante considerar a segurança e organização das informações contidas na base de dados.

- Coleta de dados: garantir que os dados dos veículos sejam coletados de forma correta, incluindo informações como placa do veículo, nome do proprietário, número do documento de identificação, entre outros.
- Armazenamento seguro: implementar medidas de segurança para proteger a base de dados, como criptografia e acesso restrito apenas a pessoas autorizadas.
- Atualização constante: manter a base de dados sempre atualizada, incluindo informações sobre novos veículos e alteração de proprietários.
- Controle de acesso: estabelecer políticas de controle de acesso à base de dados, garantindo que apenas pessoas autorizadas tenham permissão para acessar ou modificar as informações contidas nela.

As diretrizes acima para o tratamento da base de dados garantem a eficácia e segurança do projeto de Controle de Acesso Veicular em Condomínios a ser desenvolvido, protegendo a privacidade e integridade das informações dos veículos.

10. METODOS ANALÍTICOS

O problema de reconhecimento de placas veiculares é um problema bastante conhecido e desafiador na área de *computer vision*. Para alcançar esse objetivo, é preciso empregar uma variedade de métodos e algoritmos. Vamos apenas descrevemos algumas das bases teóricas desses métodos:

Processamento de imagem

O processamento de imagens é essencial para a detecção e pré-processamento das imagens de veículos. No campo de *computer vision*, existem diferentes técnicas como filtragem, segmentação e transformações de imagem, que são aplicadas para simplificar ou aumentar a qualidade das imagens ou destacar as características relevantes, como as placas dos veículos.

Como exemplo de transformação, temos a equalização de histogramas. Primeiramente, um histograma de uma imagem preto e branco mostra a intensidade do cinza para cada pixel da imagem. Assim, essa operação visa a uniformizar o nível de cinza numa imagem visando melhorar o contraste. Um segundo exemplo de transformação é a binarização, cujo processo visa transformar a intensidade do pixel em 1 (branco) ou em 0 (preto).

Aprendizado de Máquina

Certamente, o *Machine Learning* desempenha um papel ímpar no reconhecimento de placas veiculares. Por exemplo, os algoritmos de aprendizado supervisionado (que é o problema de deteção de imagem), como as redes neurais convolucionais (CNNs), são muito utilizados para treinar modelos capazes de extrair informações e reconhecer padrões em imagens em geral.

Como exemplo de algoritmo de ML, temos a regressão linear, onde a partir de preditores x_i é feita a predição de uma variável de saída y. Uma melhoria considerável no modelo anterior, é o perceptron, um classificar linear binário. A partir dos dados de entradas x , de pesos w e de uma função de ativação f, podemos produzir uma variável de saída y binária. É sabido que o Perceptron só converge se os dados forem linearmente separáveis, ou seja, se existir um hiperplano que separe os dados. Uma melhoria desse modelo, é o Multilayer Perceptron (MLP), que é composta por uma camada de neurônios de entrada, uma camada de neurônios de saída e pelo menos uma camada escondida. Esse modelo pode ser visto como a ideia de base das redes neurais convolucionais (CNN), que são aplicadas para imagens.

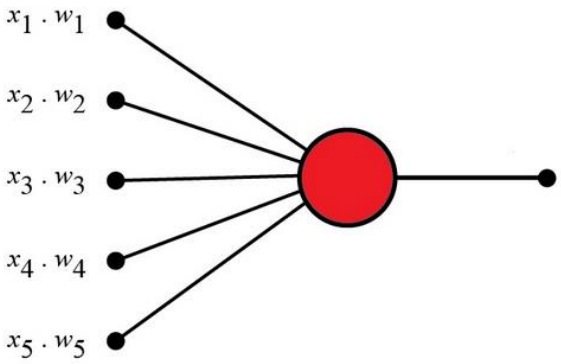


Figura 7 – Perpetron com 5 entradas, é um classificar linear binário.
Fonte: w3schools.com *(Internet)*

Detecção de Objetos

A detecção de objetos é uma técnica que permite identificar a presença e a localização de determinados objetos em imagens. Por exemplo, podemos detectar a presença de uma placa ou da face de um ser humano. Existem métodos como o algoritmo de Viola-Jones, e algoritmos baseados em redes neurais, como YOLO (You Only Look Once), que são frequentemente empregados para a detecção eficiente de objetos, inclusive o de placas de veículos.

Por exemplo, o YOLO se destacar por rápido e eficiente. A seguir, explicamos o motivo. Esse método divide uma imagem em uma grade de células, onde cada célula representa a presença de objetos dentro dela. Depois, são feitas previsões sobre cada uma das caixas e a confiança de cada uma das caixas conter um objeto. Finalmente, usa-se uma técnica chamada *non-max suppression* para combinar várias caixas que representam um mesmo objeto, removendo as caixas de baixa confiança.

Reconhecimento de Caracteres

O reconhecimento óptico de caracteres (OCR) é crucial para extrair os caracteres alfanuméricos das placas de identificação dos veículos. Existem por exemplo, técnicas como segmentação de caracteres (que é a divisão em pequenos retângulos contendo possivelmente um caractere), extração de características (é a determinação se é uma letra com traços retos ou número com traços curvos) e classificação de caracteres são aplicadas para identificar os caracteres presentes nas placas.

Enfim, essas são algumas das ferramentas possíves de serem utilizadas no desenvolvimento do nosso sistema de reconhecimento de placas veiculares.

11. DESCRIÇÃO DA METODOLOGIA DA ACURÁCIA

O reconhecimento de placas veiculares é um problema que envolve diferentes tarefas: (1) a detecção precisa da placa e (2) a extração de caracteres das placas presentes nas imagens de veículos. Para avaliar os modelos desenvolvidos, é importante utilizar uma métrica de desempenho adequada, com destaque para a acurácia.

O que é a acurácia?

É uma métrica comumente usada para medir a precisão de um modelo de classificação (por exemplo, classificar veículos entre diferentes modelos). Essa métrica representa a proporção de exemplos classificados corretamente em relação ao total de exemplos.

$$Acurácia = \frac{Numero\ de\ previsões\ corretas}{Numero\ total\ de\ previsões}$$

onde podemos expandir em previsões totalmente corretas ou parcialmente corretas com 1 caractere errado por exemplo.

No entanto, o problema de reconhecimento de placas veiculares não é um problema de classificação clássico, onde alocamos as imagens em diferentes classes. Além disso, temos 3 grandes tarefas: a deteção, segmentação e a extração de caracteres.

Detecção de placas

O objetivo dessa tarefa é localizar a região da placa. A acurácia da detecção será a proporção de placas veiculares detectadas corretamente em relação ao número total de placas. Para se dizer que uma placa foi classificada corretamente é necessário que a razão entre interseção entre a região verdadeira da placa A_{true} e a estimação da região da placa A_{est} pela união entre elas seja por exemplo maior que 70%:

$$\frac{A_{true} \cap A_{est}}{A_{true} \cup A_{est}} > 0.7$$

Segmentação

Nessa tarefa, o objetivo é a segmentação da região de cada um dos caracteres numa imagem de placa veicular. Assim, a acurácia da segmentação será a proporção de placas com todos os caracteres segmentados em relação ao número total de placas.

$$Acurácia = \frac{Numero\ de\ placas\ corretamente\ segmentadas}{Numero\ total\ de\ placas}$$

Classificação de caracteres

Enquanto na etapa da extração de caracteres, podemos utilizar métodos conhecidos de OCR para classificar as letras e o dígitos das placas. usar a classificação de cada caractere. Mais uma vez, a acurácia da classificação de

caracteres será a proporção de caracteres corretamente classificados em relação ao número total de caracteres:

$$Acurácia = \frac{Numero\ de\ carecteres\ \ corretas}{Numero\ total\ de\ caracteres}$$

onde podemos é possível expandir os resultados por cada um dos dígitos e por cada uma das letras.

12. CONSOLIDAÇÃO DOS RESULTADOS DO MÉTODO ANALÍTICO

Nessa etapa, foi realizada o processamento dos dados; separação entre dados de treinamento e de teste; treinamento do modelo.

12.1. PROCESSAMENTO DOS DADOS

A leitura dos dados é uma etapa essencial. Durante este processo, todas as imagens das placas veiculares são processadas e convertidas em arrays numpy utilizando a biblioteca *OpenCV.* As imagens são redimensionadas para um tamanho fixo 224 x 224, que é o tamanho padrão compatível com modelo pré-treinado que será utilizado.

Além disso, as imagens será renormalizadas para o formato de 8 bits, dividindo-se o array numpy por 255. As etiquetas das imagens também são renormalizadas, uma vez que a saída do modelo pré-treinado varia entre 0 e 1. Esse processo é implementado no arquivo *read_data.py*.

Vale ressaltar que cada imagem é representada por um array numpy tridimensional de tamanho 224 x 224 x 3, enquanto cada etiqueta é representada por um array numpy unidimensional de tamanho 4. A coleção de imagens e etiquetas é salva no formato .npy (a saber, *X.npy* e *y.npy*) para serem reutilizadas em outras etapas do processo de aprendizagem.

12.2. TREINAMENTO DO MODELO

Após repartir as imagens e etiquetas (representadas pelas variáveis X e y) em conjuntos de treinamento e teste utilizando a biblioteca *sklearn*, com uma proporção de 4:1 de dados de treinamento para dados de teste, obtemos:

```
Formato de X_train: (346, 224, 224, 3)
Formato de X_test: (87, 224, 224, 3)
Formato de y_train: (346, 4)
Formato de y_test: (87, 4)
```

Figura 8 – Tamanho dos arrays numpy representado os conjuntos de treinamento e de teste.
Fonte: Elaboração própria (2024).

O modelo pré-treinado de Deep Learning escolhido é o **Inception-ResNet-v2**. Esse é um modelo de rede neutral convolucional (CNN) que já foi treinado com milhares de imagens da base de dados *ImageNet*. Essa rede possui um total de 164 camadas sendo classificar imagens em mais de 1000 categorias de objetivos, como, caneta, teclado e outros. Portanto, essa rede possui uma boa representação de uma multitude de objetos.

Segue a arquitetura da rede neural abaixo:

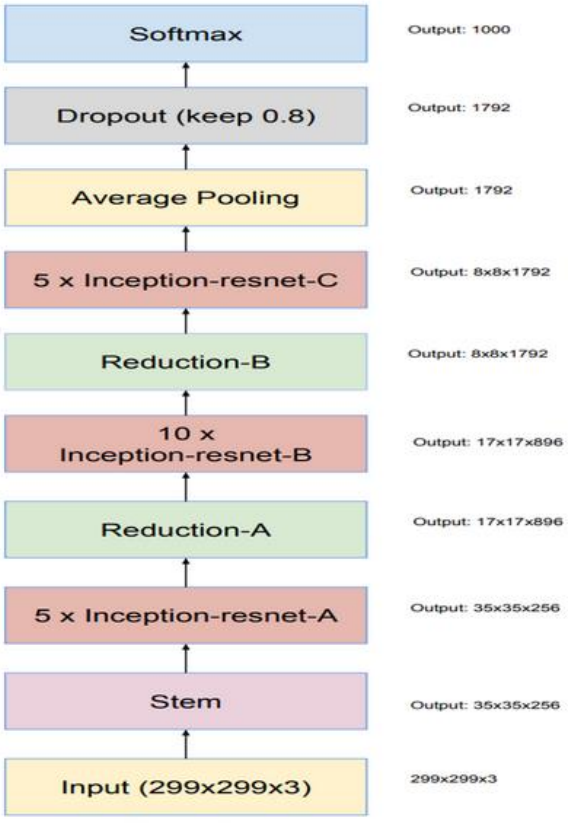
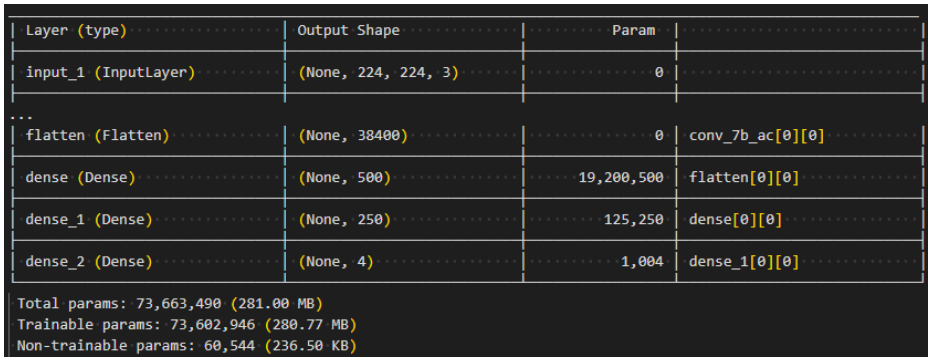


Figura 9 – Arquitetura do modelo Inception-ResNet-v2 com aproximadamente 164 camadas internas.
Fonte: Inception-ResNet-v2 by Szegedy et al..

Assim, usaremos o modelo **Inception-ResNet-v2** com peso pré-treinados para treiná-los ao nosso banco de dados. Para isso, importaremos e ajustaremos as

dimensões das entradas e saídas utilizando a biblioteca *TensorFlow*. É importante mencionar que a saída original do modelo foi redimensionada para comportar a saída das etiquetas (um array numpy unidimensional de tamanho 4). Posteriormente, o modelo foi compilado usando uma função de perda é **MSE** (i.e., *Mean Squared Error*) e o otimizador Adam com uma taxa de aprendizagem de **0.0001**.



Layer (type)	Output Shape	Param	
input_1 (InputLayer)	(None, 224, 224, 3)	0	
...			
flatten (Flatten)	(None, 38400)	0	conv_7b_ac[0][0]
dense (Dense)	(None, 500)	19,200,500	flatten[0][0]
dense_1 (Dense)	(None, 250)	125,250	dense[0][0]
dense_2 (Dense)	(None, 4)	1,004	dense_1[0][0]

Total params: 73,663,490 (281.00 MB)
Trainable params: 73,602,946 (280.77 MB)
Non-trainable params: 60,544 (236.50 KB)

Figura 10 – Resumo do modelo **Inception-ResNet-v2** modificado. Output original: 38400. Output modificado: 4.
Fonte: Elaboração própria (2024).

A etapa de treinamento foi realizada em um computador com *Windows 11 Home Single Language*, equipado com um processador *12th Gen Intel(R) Core(TM) i5-12500T 2.00 GHz* e 7,70 GB de RAM utilizável. Devido às limitações de recursos computacionais, treinamos o modelo com os dados X_train e y_train usando um tamanho de lote (*batch size)* de 16 em apenas 8 *epochs*.

Após o treinamento, o modelo foi salvo no arquivo *object_detection.keras*. Esse processo de pré-treinamento foi implementado no arquivo *train_model.py*.

13. REVISÃO DO DESEMPENHO DO MÉTODO ANALÍTICO

Nessa etapa, foi realizada a avaliação do desempenho do modelo sobre os dados de treinamento e validação.

Conforme explicado na etapa de treinamento, o critério de otimização do modelo **Inception-ResNet-v2** foi o erro quadrático médio (MSE em inglês), em vez da acurácia. Na verdade, o algoritmo de optimização visa minimizar uma função de erro entre as etiquetas de teste/treinamento e as etiquetas previstas pelo modelo. Como as etiquetas são representa por arrays numpy unidimensionais de tamanho 4, elas podem ser visualizadas em um espaço euclidiano de dimensão 4, onde o erro é

simplesmente que a distância euclidiana entre esses dois pontos: observado e o previsto.

Utilizando a ferramenta TensorBoard, executamos o comando `tensorboard --logdir="./object_detection` para obter o histórico de treinamento. Abaixo, podemos observar os valores da função de perda (*epoch_loss*) para cada época:

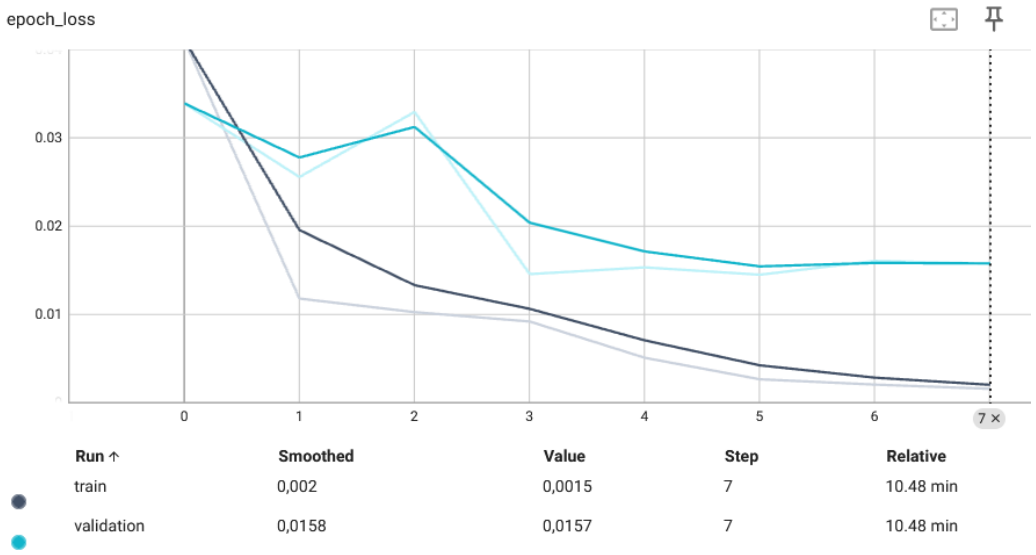


Figura 11 – Curva de aprendizagem (i.e., valor da *loss function*) para cada *epoch* nos dados de treinamento e de validação**.**
Fonte: Elaboração própria (2024).

De modo geral, o valor da *loss* diminui com o passar épocas, indicando que o modelo está aprendendo a identificar as placas veiculares cada vez melhor no *dataset* de treinamento. Certamente, aumentar o número de *epochs* resultaria em um melhor score para o modelo ao final do treinamento.

De mesma forma, o valor da *loss* decresce com a *epoch* para o *dataset* de validação. Observamos que a *loss* para os dados de validação é maior, justamente porque eles não são usados para treinar o modelo, mas sim para avaliar sua capacidade de generalização para dados não vistos anteriormente. Como o valor da *loss* dos dados de treinamento ainda não se estabilizou, o processo de aprendizagem não deveria ter sido encerrado em 8 *epochs*.

Agora, com o modelo treinado e salvo, avançaremos para a detecção de placas e reconhecimento de caracteres propriamente ditos em uma imagem de teste.

14. DESCRIÇÃO DE RESULTADOS PRELIMINARES

Nessa etapa, foi realizada a predição do modelo, ou seja, a deteção das placas veiculares juntamente com o reconhecimento de caracteres das placas.

14.1. DETECÇÃO DE PLACAS VEICULARES.

Carregando o modelo previamente salvo no arquivo *object_detection.keras*, somos capazes de utilizá-lo para fazer previsões sobre uma nova imagem de carro, identificando a localização da placa veicular. Para isso, obtivemos novas imagens de testes que não foram utilizadas no processo de treinamento e validação.

Nosso projeto de detecção automática de placas veiculares foi originalmente concebido para detecção de placas brasileiras. Após uma busca por imagens na internet, encontramos as seguintes imagens que servirão como conjunto de teste:

Figura 12 – Amostra das imagens de veículos com placas brasileiras incluindo o modelo do Mercosul. Fonte: Busca por imagens “Foto de frente de carros com placas brasileiras”.

As imagens de teste foram carregadas da pasta “dataset/TEST” e processadas da mesma maneira que as imagens de treinamento/validação. Usando o modelo, foram previstas as quatro coordenadas (x_min, x_max, y_min, y_max) identificando a localização das placas. Em seguida, as coordenadas foram transformadas de valores entre 0 e 1 para valores correspondentes ao tamanho das imagens originais. Esse processo de deteção de placas foi implementado no arquivo *detect_plate.py*.

Figura 13 – Amostra das imagens de veículos com placas identificadas por uma caixa amarela.
Fonte: Elaboração própria (2024).

De modo geral, as placas veiculares foram identificadas nas imagens de teste. Observamos que a imagem com mais zoom na placa obteve a melhor detecção, enquanto os veículos fotografados de frente tiveram suas placas mais bem classificadas do que veículos fotografados de trás. Seria interessante testar a detecção para veículos fotografados em ângulo ou com baixa visibilidade (pouca iluminação ou qualidade ruim da imagem).

14.2. RECONHECIMENTO DE PLACAS VEICULARES.

As imagens das placas detectadas para os veículos de teste foram salvas na pasta “dataset/TEST”, depois reprocessadas para a realização do OCR (Optical Character Recognization). Utilizando o software chamado Tesseract OCR e sua API em Python (pytesseract), foi realizado o reconhecimento de caracteres nas placas detectadas.

O Tesseract é uma ferramenta *open source* que realiza as tarefas de segmentação do texto do plano de fundo quanto melhor for a qualidade da imagem. Assim, se a imagem apresentar algum ruído, iluminação ruim ou baixo contraste, a tarefa de reconhecimento será prejudicada. Apesar de ser uma ferramenta robusta, possui limitações, como não reconhecer bem imagens sob perspectiva distorcida ou fundo complexo, podendo gerar lixo no output. Esse processo de reconhecimento de caracteres a partir das placas identificadas foi implementado no arquivo *recognize_plate.py*.

String vazia

“BEESR22)|\n”

“AE\n”

String vazia

Figura 14 – Amostra das imagens de veículos com placas reconhecidas pelo Tesseract OCR.
Fonte: Elaboração própria (2024).

É claro que não reconhecemos as placas corretamente. Para a terceira placa, temos 80% das informações, com um caractere mal identificado e um tanto de lixo adicionado. Para melhorar o processo de reconhecimento, as placas precisam ser mais bem identificadas. Assim é recomendado aumento o número de *epochs* a fim que o erro de detecção seja o mínimo possível e a maior fonte de erro será no algoritmo do Tesseract sobre o qual não temos muito o que fazer.

De maneira geral, com poucas *epochs*, esse modelo pode ter um desempenho inferior especialmente se consideramos todas as etapas do processo. No entanto, essa primeira iteração permitiu entender as áreas de melhoria para nosso modelo, como, o aumento do número de *epochs* e a inclusão de mais dados de treinamento. Tentaremos implementar essas melhorias para a próxima etapa do projeto.

14.3. MODELO DE NEGÓGIO

Nosso modelo de negócio consiste em um sistema de detecção e reconhecimento de placas chamado Sentinela, solicitado por uma administradora de

condomínios que planeja implementá-lo nos condomínios de casas em São José dos Campos.

Para desenvolver o sistema de reconhecimento de placas veiculares, utilizamos diferentes bibliotecas de *machine learning* e uma base de dados contendo imagens de carros com placas estrangeiras. No entanto, nosso *dataset* era bastante limitado contendo apenas 433 imagens. Com base nos resultados preliminares, é evidente que precisamos investir em um *dataset* maior e mais abrangente.

Além disso, nosso modelo requer um treinamento mais aprofundado, que pode ser alcançado utilizando GPUs para acelerar os cálculos durante o treinamento.

15. ELABORAÇÃO DO ESBOÇO DO STORYTELLING

Com os elementos desenvolvidos no projeto, foi feito um esboço do storytelling a ser feito em uma eventual apresentação de negócio.

> *"Durante uma disciplina de Ciência de Dados, um grupo de alunos, cheios de curiosidade, paixão por novas tecnologias e muita vontade de empreender, se lançou num projeto desafiador: criar uma solução inovadora para um problema real com aplicação comercial. Assim nasceu o Sentinela, um sistema automático de detecção e reconhecimento de placas veiculares a partir de imagens de veículos.*
>
> *A jornada começou com a concepção do Sentinela, um projeto de Aprendizado de Máquina destinado a revolucionar a forma como identificamos veículos. Fomos abordados por uma administradora de condomínios, que via no Sentinela a solução ideal para garantir a segurança e o controle de acesso em propriedades administradas.*
>
> *Nosso modelo de negócio estava traçado: oferecer um sistema de detecção e reconhecimento de placas de excelência que atendesse às necessidades de condomínios em São José dos Campos - SP. Para alcançar nossa meta, mergulhamos de cabeça no vasto mundo da aprendizagem de máquina, utilizando diferentes bibliotecas em python e ferramentas para desenvolver nosso primeiro modelo.*

No entanto, logo nos deparamos com um desafio significativo: nosso conjunto de dados era limitado, contendo apenas algumas centenas de imagens de carros com placas estrangeiras. Para criar um modelo robusto e preciso, precisávamos de mais dados e poder computacional para treinar nosso algoritmo de maneira mais profunda e abrangente.

Optamos por utilizar o modelo pré-treinado Inception-ResNet-v2, uma poderosa rede neural convolucional treinada com milhares de imagens da base de dados ImageNet. Adaptamos esse modelo às nossas necessidades e o treinamos com nosso conjunto de dados, ajustando parâmetros e otimizando o processo de aprendizado.

Apesar de nossos esforços, os resultados preliminares não foram tão promissores quanto esperávamos. Reconhecemos que nossas placas ainda não eram identificadas corretamente, com erros significativos em algumas delas. Percebemos que precisávamos de mais tempo de treinamento e mais dados para melhorar a precisão do nosso modelo.

No entanto, essa primeira iteração do Sentinela nos proporcionou valiosas lições. Compreendemos a importância de aumentar o número de epochs e a quantidade de dados de treinamento para aprimorar nosso modelo. Reconhecemos os desafios que ainda temos pela frente, mas estamos determinados a superá-los e tornar o Sentinela uma solução confiável e eficaz para nossos clientes.

Assim, nossa caminhada continua, impulsionada pela paixão pela inovação e pelo desejo de criar um impacto positivo no mundo. Com cada iteração, estamos mais perto de alcançar nossa meta e transformar o Sentinela em uma ferramenta indispensável para a segurança e o controle de acesso em condomínios de todo o país".