

Estacionamento inteligente na Unifesp

Davi Seiji Kawai Santos
Ciência da Computação
Universidade Federal de
São Paulo – UNIFESP
São José dos Campos, Brasil
davi.seiji@unifesp.br

Pedro Henrique Masteguin
Ciência da Computação
Universidade Federal de
São Paulo – UNIFESP
São José dos Campos, Brasil
p.masteguin@unifesp.br

RESUMO

Com objetivo de otimizar a eficiência e a economia de tempo na tarefa do estacionamento, utilizando o modelo de inteligência artificial YOLO, o projeto consiste na detecção e identificação de automóveis nas vagas do estacionamento do campus da UNIFESP. Essa abordagem a vançada proporcionará a contagem e monitoramento em tempo real da quantidade de vagas vazias e preenchidas provendo aos usuários uma gestão mais eficiente e inteligente, em uma experiência de estacionamento simplificada.

I. INTRODUÇÃO E MOTIVAÇÃO

Estacionar é algo frequente na vida da maioria das pessoas todos os dias, para qualquer lugar que é preciso o uso de veículos para locomoção, majoritariamente carros e motos, é necessário estacionar. A procura de vagas e também o controle de veículos dentro de um estacionamento são temas muito importantes para a melhora do dia-a-dia.

Um artigo muito interessante sobre estacionamento, Seco. et. Al. [1], explica bem sobre o que é um estacionamento, assim como diversos tipos de estacionamentos. Na introdução diz que “Antes e no fim de cada viagem é necessário dispor de um local próprio para estacionar o automóvel, o que, particularmente em áreas urbanas, nem sempre é fácil de encontrar.”. Isso é muito verdade na vida de muitos, e buscar facilitar essa busca por vagas pode ser muito útil na vida dos motoristas.

Aprimorar o uso de estacionamentos é tema de diversas pesquisas e projetos presentes na literatura, uma vez que se faz necessário esse tipo de pesquisa tendo em vista o crescimento de veículos automotivos enquanto que o número de estacionamentos e vagas não acompanha tal crescimento. Um estudo realizado pelo INRIX em 2019, mostrou que em Nova Iorque motoristas gastam em média 107 horas por ano procurando por vagas para estacionar [2] o que implica em um grande volume de gastos com combustíveis e tempo, sem contar a grande emissão de poluentes e a inevitável densidade de veículos que gera congestionamentos.

Primeiramente, com essas informações podem ser resolvidos diversos problemas dentro de um estacionamento, como por exemplo, fazer uma melhor distribuição dos carros e motos no

estacionamento, se adequando a necessidade do estacionamento e das pessoas.

Alguns desses problemas podem ser vistos em: Barata. et.al [3]. Neste artigo é dito que problemas como: congestionamento e ruído (que geram ainda mais problemas), além de ser comum na vida real, também é comum em universidades. Na Unifesp, principalmente em horários de picos que seriam antes de início de aula pode ser observado esses problemas que podem ser evitados.

Porém, os problemas a serem resolvidos e os problemas ficarão a critério do estacionamento, neste caso da Unifesp. Por meio deste trabalho vamos fornecer dados importantes ao estacionamento para a solução, que pensamos que os principais seriam:

- Vagas estacionadas (onde e quantidade)
- Vagas livres (onde e quantidade)
- Total de carros presentes no estacionamento (estacionados e em locomoção)

Para que esses dados sejam obtidos, terá que ser feito um mapeamento do estacionamento requerido, que neste trabalho será da Universidade Federal de São Paulo - UNIFESP, no Campus São José dos Campos, mais especificamente na unidade do Parque Tecnológico.

Este mapeamento será feito a partir do uso de um drone, que irá capturar imagens, primeiro com o estacionamento vazio, para que seja possível a identificação de todas as vagas do estacionamento, e depois em um dia normal, para conseguir imagens para treinos e testes.

Após esse mapeamento, será usado uma biblioteca da linguagem Python, que usa o deep learning para identificar carros em uma imagem, contando e fazendo a classificação das vagas (ocupadas e desocupadas), e ao fazer isso, teremos nossas informações buscadas nesse projeto. O modo que será feito vai ser explicado mais na frente deste relatório.

II. CONCEITOS FUNDAMENTAIS

Aqui vamos falar alguns dos principais conceitos para o melhor entendimento do nosso trabalho.

- *Vaga no estacionamento*

Serão aqueles locais do estacionamento que são designados para estacionar veículos.

- *Dados do trabalho*

Os dados deste trabalho serão as fotos tiradas no estacionamento da Unifesp, que serão coletadas com um drone. O drone utilizado neste trabalho será o Phantom 3 Advanced da marca DJI, as suas especificações podem ser vistas no site [4].

- *Estacionamento da Unifesp*

O estacionamento trabalhado em questão está localizado na Av. Cesare Monsueto Giulio Lattes, 1201, São José dos Campos, que vai ser mostrado abaixo:



Fig 1. Estacionamento da Unifesp [5]

- *Inteligência Artificial*

É a busca automação do pensamento e comportamento inteligente. Neste trabalho seria treinar uma máquina para que ela fique inteligente o bastante de modo que identifique os objetivos a partir de uma imagem de estacionamento.

- *Aprendizagem de máquina*

No capítulo 18 do livro “Inteligência Artificial” [6], fala que: “Um agente estará aprendendo se melhorar o seu desempenho nas tarefas futuras de aprendizagem após fazer observações sobre o mundo.”. Então, aprendizagem de máquina é basicamente ensinar uma máquina para que ela faça o trabalho pedido com melhor eficácia.

- *Deep learning*

No livro “Deep learning with python” [7] temos a seguinte definição no capítulo 1: “Deep learning is a specific subfield of

machine learning: a new take on learning representations from data that puts an emphasis on learning successive layers of increasingly meaningful representations. [...] In deep learning, these layered representations are (almost always) learned via models called neural networks, structured in literal layers stacked on top of each other.”. Ou seja, é uma subárea do aprendizado de máquina baseada em camadas de neurônios interconectados (redes neurais) que imitam o funcionamento do cérebro humano, visando aprender e realizar tarefas complexas de forma automatizada.

- *Neurônio*

Unidade básica de processamento de uma rede neural. Recebe informações de entrada, realiza cálculos com tais informações e produz uma saída.

- *Bounding Boxes*

As bounding boxes são caixas delimitadoras que mostram regiões de interesse em uma imagem. Neste trabalho em específico, as bounding boxes representarão as vagas e os carros. Elas são definidas por coordenadas que indicam o centro da imagem (x e y), altura e largura. É importante lembrar que será utilizado o formato YOLO, no qual as coordenadas são normalizadas e o primeiro número representa o objeto associado àquela bounding box.

- *YOLOv8*

O YOLO (You Only Look Once) é um modelo de detecção de objetos que usa redes neurais convolucionais (CNN) para identificar objetos em imagens. O YOLOv8 é uma versão aprimorada do YOLO, que oferece melhor precisão e desempenho. O seu funcionamento consiste em dividir a imagem em uma grade de células e cada célula é responsável por prever caixas delimitadoras e classes de objetos, em uma arquitetura de camadas convolucionais e totalmente conectadas, para gerar as previsões das caixas e assim classificar o objeto.

Diferente de outras abordagens, o YOLOv8 realiza a detecção em uma única etapa ao examinar a imagem inteira ao invés de realizar previsões para cada grade separadamente, tornando o processo mais rápido. Outro ponto que influencia na velocidade de funcionamento do modelo é o fato de utilizar-se de uma rede pré-treinada como ponto de partida, mesmo quando utilizado para detecção em um dataset customizado, dessa forma o conhecimento prévio é aproveitado e cada nova rede gerada dentro do treinamento é salva de forma automática pela biblioteca, a fim de encontrar a melhor rede com base no dataset e nos parâmetros passados.

Após treinando, a rede produzida pelo modelo pode ser utilizada no modo de predição para detectar os objetos desejados, retornando ao usuário as posições das caixas delimitadoras bem como as probabilidade da classe do objeto.

- *Cross-validation*

O cross-validation é um protocolo de avaliação que será usado neste projeto. Ele consiste em dividir o conjunto de dados em k folds, permitindo a avaliação repetida do modelo em diferentes subconjuntos de treinamento e teste.

III. TRABALHOS RELACIONADOS

Aprimorar o uso de estacionamentos é tema de diversas pesquisas e projetos presentes na literatura, uma vez que se faz necessário esse tipo de pesquisa tendo em vista o que foi apresentado na introdução e motivação. Visando a melhoria no fluxo de veículos e reduzir o congestionamento de vias, o uso de sistemas de estacionamento inteligentes é indispensável, como é apresentado no artigo [8] “Smart parking in IoT-enabled cities: A survey” que faz uma análise sobre a implementação de sistemas de estacionamento inteligente e suas tecnologias: “[...] One of the key components in ITS is the Smart Parking System (SPS), which relies significantly on analyzing and processing the real-time data gathered from vehicle detection sensors and the radio frequency identification (RFID) systems that are placed in parking lots to report the absence and/or presence of a vehicle [...].” Os sistemas de estacionamento inteligentes são peças importantes em sistemas de transporte inteligente e informam a presença ou não de automóveis em uma vaga de estacionamento a partir de dados processados em tempo real.

O artigo [9], tem uma relação com o nosso trabalho, tendo em vista que também busca implementar o estacionamento inteligente em um campus de uma universidade. Porém, por mais que o objetivo geral acaba sendo o mesmo, o meio para chegar nele é diferente do nosso, já que neste trabalho mencionado, além de câmeras foi usado sensores para auxiliar na detecção das vagas, enquanto o nosso será feito o processamento somente de imagens.

Para tal processamento, existem diversos meios e tecnologias para detecção de veículos no âmbito desse projeto utilizaremos processamento de imagens de vídeo, que segundo o artigo Carpark system: A review of smart parking system and its technology [10], o processamento de imagens consiste basicamente em uma ou mais câmeras e a partir de um software processa e analisa os frames para detecção e diferenciação de veículos e vagas.

A partir disso, um dos meios de obtenção dessas imagens pode ser utilizando-se de VANT (Veículo aéreo não tripulado), um drone com uma câmera acoplada para monitoramento e inspeção visual de disponibilidade de vagas. Isso pode ser muito bem observado em outro trabalho [11] relacionado ao nosso. Nele tem o uso de um drone para pegar imagens sobre o estacionamento, e a partir dessa imagem, é possível identificar as vagas livres e as ocupadas. Os testes são semelhantes ao qu

A biblioteca utilizada para a criação de um modelo de aprendizado de máquina é o Tensor Flow, uma biblioteca de código aberto para aprendizado de máquina e deep learning da Google Brain, biblioteca escolhida devido ao grande suporte e uso em aprendizado profundo de máquina e principalmente a aplicação em trabalhos relacionados ao nosso. Um projeto de

sistema de gerenciamento de estacionamento [12] mostra a aplicação do Tensorflow na detecção de veículos e vagas:

- Imagem de vídeo antes do processo de classificação do detector de objeto Tensorflow:



Fig 2 Retirada de [12].

- Imagem de vídeo após o processo de classificação do detector de objeto Tensorflow

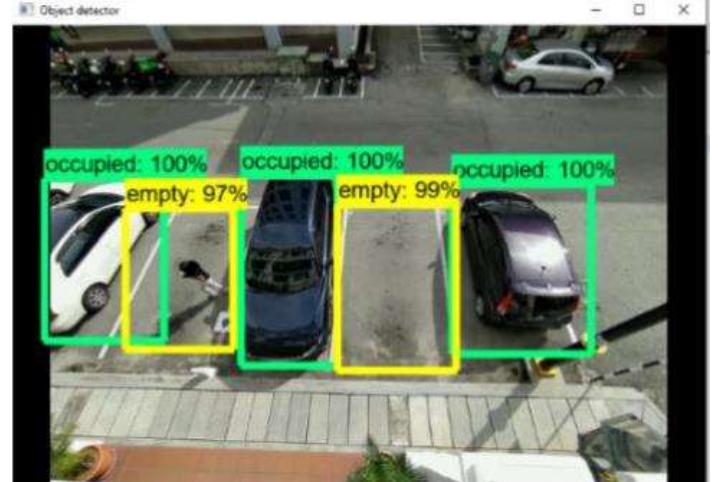


Fig 3. Retirada de [12].

Isso nos dá uma base sólida para trabalharmos com tal biblioteca na aplicação do sistema de estacionamento inteligente no campus da UNIFESP em São José dos Campos.

IV. OBJETIVOS

O principal objetivo deste trabalho é que por meio de uma imagem do estacionamento, que será obtido em tempo real, conseguimos responder os seguintes tópicos:

- Quantidade de vagas disponíveis
- Quantidade de vagas preenchidas
- Quantidade de carros no estacionamento

V. METODOLOGIA EXPERIMENTAL

O projeto será divido em partes, mas especificamente em 3:

- *Capturar as imagens (dados)*

Este passo será a construção do database que será usado, em alguns outros projetos observados muitas vezes são usados dados já capturados, como um exemplo o artigo [13], onde ele diz que foi usado um dados já existente de uma cidade chamada Santander da Espanha, que busca ser inteligente, e em cima deste database foi feito o trabalho deles.

Nesta parte do projeto, devemos montar nossos dados, tanto do treino quanto dos testes, tendo em vista que não temos em mãos as imagens do estacionamento da Unifesp. Para isso será usado um drone, que vai se posicionar em um ângulo para começar a pegar as imagens. As imagens serão tiradas de cima e serão mostradas mais abaixo.

- *Base de dados*

Com as imagens do drone em mãos, primeiro iremos trabalhar para identificar as vagas em nosso estacionamento. Para isso, usaremos o estacionamento vazio para facilitar a visualização das vagas. O primeiro passo foi dividir o estacionamento em 3 partes, e também cortar elas retirando partes que não vão ser usadas, que será mostrado abaixo:

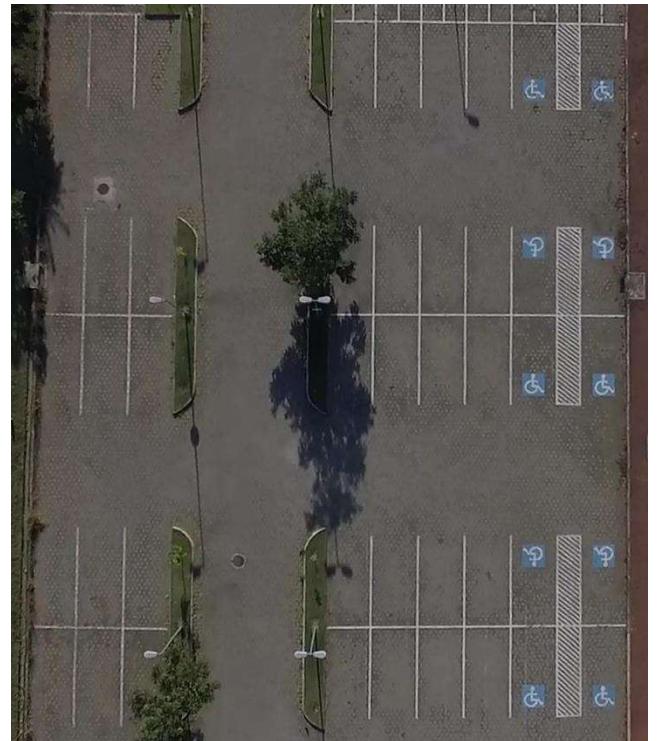


Fig 4. Estacionamento da Unifesp.

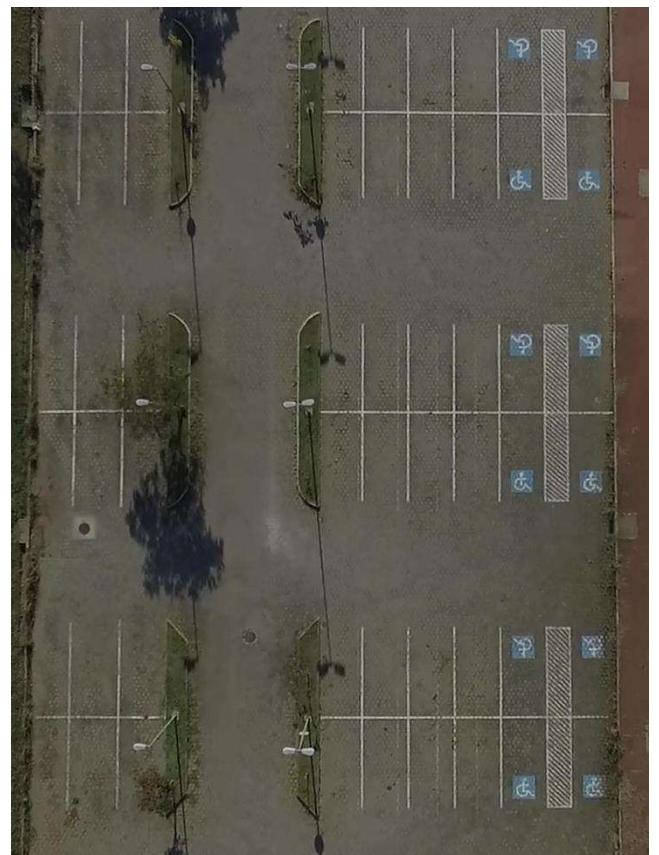


Fig 5. Estacionamento da Unifesp.



Fig 6. Estacionamento da Unifesp.

Importante ressaltar que foi usado a biblioteca cv2(OpenCV) para a manipulação das imagens. O próximo passo foi identificar as vagas, e para isso foi usado uma função da própria biblioteca cv2, chamada selectROI. Nela você seleciona as regiões de interesse, que no nosso caso serão as vagas e é dado 4 inteiros, que usando a função cv2.rectangle é desenhado essa região. Após fazer isso nossa nova imagem ficou do seguinte jeito:



Fig 7 Estacionamento da Unifesp com as vagas em azul.



Fig 8 Estacionamento da Unifesp com as vagas em azul.

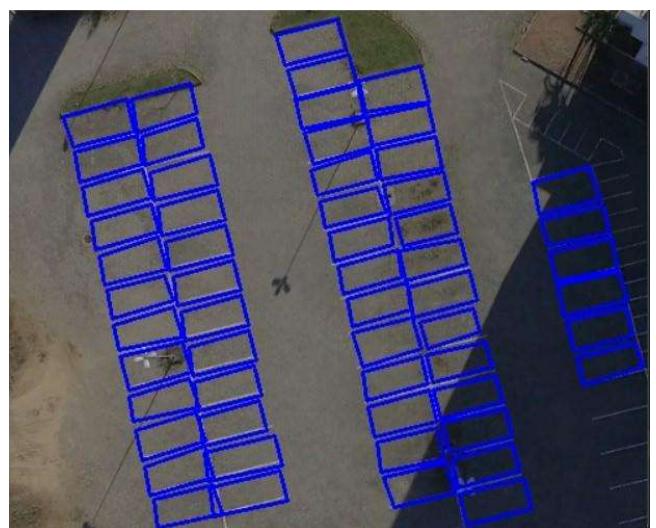


Fig 9 Estacionamento da Unifesp com as vagas em azul.

Agora com as vagas identificadas, pegamos os vídeos onde temos imagens do estacionamento com carros, e retiramos frames dos vídeos, para termos fotos com movimentação de carros. Para isso, foi usado diversas funções da biblioteca OpenCV, tanto para rotacionar os frames quanto para ter um intervalo entre os frames pegos. Importante ressaltar que para

cada vídeo as coordenadas das bounding boxes acabou variando um pouco, por conta de pequenas variações na altura e na posição do drone.

Então, no processo de pegar os frames, podemos selecionar o tempo entre cada foto, e o escolhido foram 2 segundos entre as imagens. Com essas imagens, desenhamos as vagas para arrumar as regiões de interesse de acordo com o vídeo e após esse processo, salvamos em uma pasta todas imagens, de acordo com o ângulo e vídeo da mesma. No total, nossa base de dados ficaram com 145 imagens.

Algumas das imagens ficaram da seguinte maneira:



Fig 10 Estacionamento da Unifesp com vagas em azul.



Fig 11 Estacionamento da Unifesp com vagas em azul.



Fig 12 Estacionamento da Unifesp com vagas em azul.

Após tudo isso, de teremos em nossa base de dados, separadas por pastas, as fotos e as respectivas coordenadas de todas as regiões de interesse de cada imagem, que no caso do trabalho serão as vagas.

- *Protocolo de validação*

O protocolo que será usado será o cross validation, que consiste em dividir nossos dados em k-folds, e realizar k-iterações onde o conjunto de teste cada vez será um fold e o de treino serão todos menos o de teste. Para dividir os nossos dados,

juntamos todos os dados em uma pasta e dividimos aleatoriamente em 5-folds.

Assim, fazendo as k-iterações, teremos um método de avaliação, que será o mAP (mean Average Precision) de modo que a partir dele, teremos a melhor rede treinada. Para isso teremos que ter a verdade de todas as vagas, que serão feitas manualmente e serão usadas na avaliação dos folds.

- *Pipeline experimental*

O pipeline experimental é uma sequência de etapas que descreve o processo da realização deste trabalho. Neste trabalho, o pipeline experimental está da seguinte maneira:

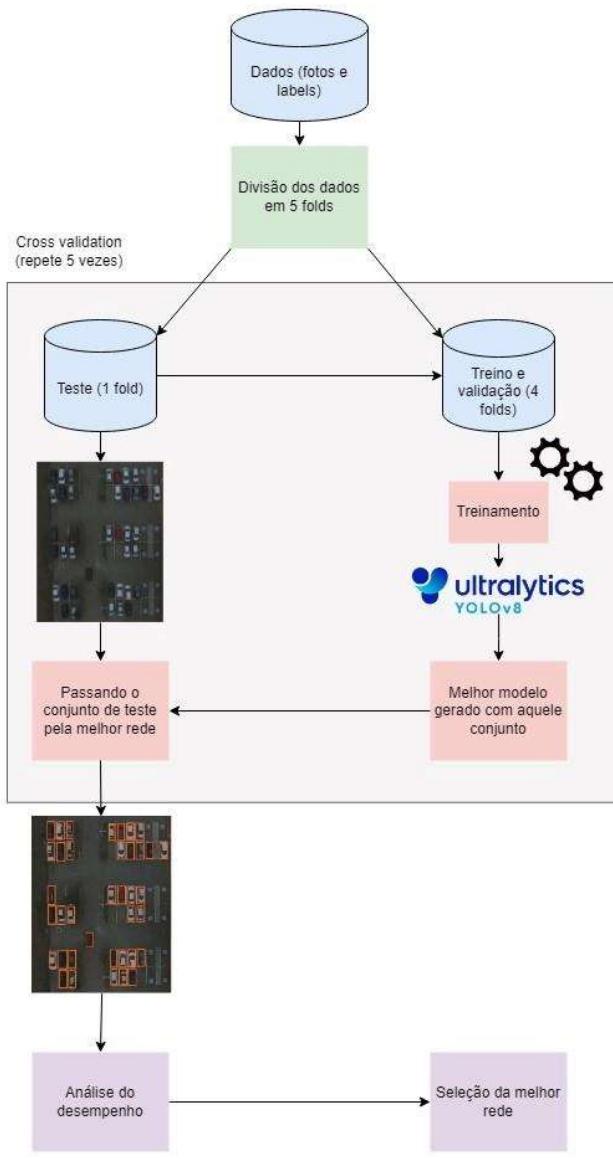


Fig 13 Pipeline do projeto

- *Treinamento da inteligência artificial*

Essa etapa consiste em se utilizar dos dados obtidos, as imagens de drone, para criação de um modelo capaz de identificar e diferenciar os veículos das vagas, bem como a diferenciação de veículos em transito. Para isso utilizaremos a biblioteca TensorFlow juntamente com o modelo de rede neural YOLO, uma rede pré-treinada utilizada na detecção de objetos, que será utilizada para classificar o objeto carro nos nossos vídeos.

Com os dados pré-processados na base, construímos a arquitetura da rede neural convolucional utilizando o TensorFlow, especificamente o modelo YOLO. O treinamento será realizado em várias iterações sobre o conjunto de dados, onde a imagem passa pelo modelo que faz as previsões da classificação da seguinte maneira: O YOLO, uma rede pré-treinada para a detecção de diversos objetos, divide a imagem em grades e prevê a presença de carros atribuindo a eles caixas delimitadoras, com isso e com as coordenadas das vagas que obtemos do mapeamento do estacionamento, iremos comparar esses dados para verificar se o objeto carro está ou não sobre um objeto vaga.

Uma das vantagens do modelo YOLO é seu amplo suporte e estrutura de código aberto. Isso nos permite ajustar as configurações da rede, caso seja necessário, para otimizar a detecção de carros em nossas imagens específicas. Podemos explorar diferentes hiperparâmetros, arquiteturas de rede ou estratégias de treinamento para melhorar a precisão e a robustez do nosso modelo.

Então, partindo para o treinamento, os dados pré-determinados são divididos em duas partes: a parte de treino e a parte de validação. Neste caso, utilizamos 80% dos dados para treinamento e o restante para validação. É importante ressaltar que cada foto possui um arquivo correspondente no formato .txt, com o mesmo nome da foto, contendo as coordenadas das bounding boxes onde há um carro.

Em seguida, utilizamos uma função da biblioteca YOLO para treinar a rede neural pré-treinada com os dados fornecidos. Definimos também o número de épocas para o treinamento, sendo estabelecido o valor de 25 épocas. Ao final de cada época, é calculado o MAP, uma métrica comumente usada na detecção de objetos, para avaliar a melhoria ou piora do desempenho do modelo após as atualizações nos pesos. O objetivo é garantir a obtenção da melhor configuração da rede ao final do treinamento.

- *Testes*

Com a melhor rede de cada etapa do cross-validation, podemos prosseguir para os testes. Nessa fase, selecionamos o único conjunto de dados que não foi utilizado no treinamento e aplicamos a rede a todas as imagens desse conjunto. Como também temos os rótulos verdadeiros correspondentes, podemos analisar a porcentagem de acertos obtida por essa rede.

Usando a função para predizer do YOLO, podemos definir a confiança, que foi usado variações de modo que seja possível definir a melhor confiança, ou seja aquele que vai obter a melhor porcentagem de acerto. No final, analisando as redes treinadas juntamente com as variações de confiança, será possível obter a melhor rede gerada pelo treinamento.

Dessa forma, através da avaliação das redes treinadas nos diferentes conjuntos de dados e das variações de confiança, podemos identificar a rede que obteve o melhor desempenho em termos de porcentagem de acerto. Isso nos permite selecionar a melhor configuração da rede para ser utilizada na detecção de carros com base nos resultados obtidos nos testes.

VI. ANÁLISE DOS PRIMEIROS TESTES

A partir do treinamento foi possível identificar os carros nas nossas imagens, escolhemos uma das redes para realizar um teste inicial e variando a confiança de 0.2 à 0.5 identificamos que a melhor confiança foi a de 0.3 onde obtivemos os melhores resultados analisando a diferença entre a quantidade de carros no estacionamento (verdade) e a predição da rede, onde chegamos a uma taxa de acerto de 90%, podendo ser verificado pelo gráfico:

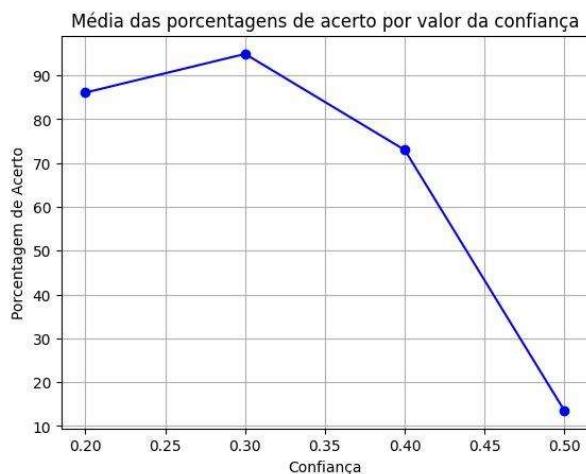


Fig 14 Porcentagem de acerto pela confiança

Onde podemos notar que a maior taxa de acerto foi quando usamos a confiança igual a 0.3. Contudo nessa rede encontramos duas situações de falha na predição.

A primeira questão tratava da sobreposição de caixas delimitadoras (bounding boxes) para o mesmo veículo. Quando ocorria mais de uma predição de caixa delimitadora para um único veículo, isso resultava em uma contagem incorreta de vagas no resultado. Para solucionar esse problema, foi utilizado o parâmetro IOU (Interseção sobre União) durante a predição.

O parâmetro IOU é uma métrica que calcula a sobreposição entre duas regiões, no caso, as caixas delimitadoras. No trabalho em questão, foi utilizado o valor de 0.2 para o IOU. Isso significa

que, se houver uma sobreposição de mais de 20% entre duas caixas delimitadoras ao identificar dois carros, a caixa delimitadora menor é excluída. Essa abordagem foi adotada para evitar a detecção duplicada de veículos na mesma imagem.

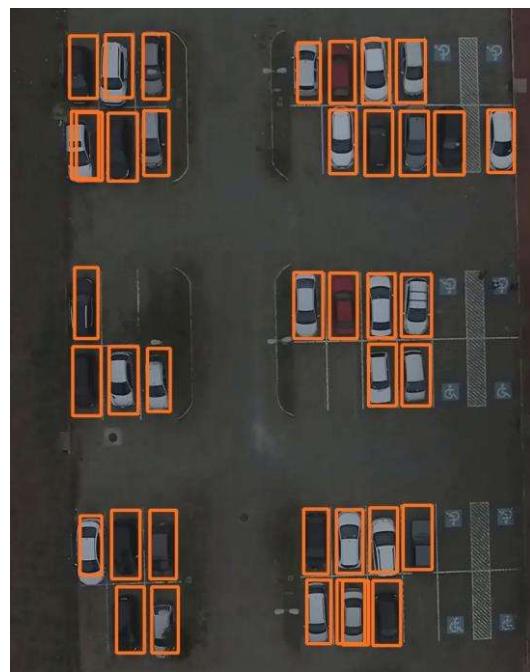


Fig 15 Sobreposição de bounding boxes

Na imagem acima podemos notar que em algumas das vagas há a sobreposição de caixas de identificação.

Outra falha que identificamos foi ao predizer os dados da área 3, pois devido ao ângulo de filmagem e altura das imagens ser diferente das outras duas áreas, a taxa de acerto foi muito baixa, o que nos mostrou a necessidade de criar e treinar uma rede separada para esse ângulo.



Fig 16 Exemplificação do ângulo 3

Podemos analisar mais alguns resultados a partir dos gráficos gerados pela rede no treinamento utilizando o fold 1 como teste:

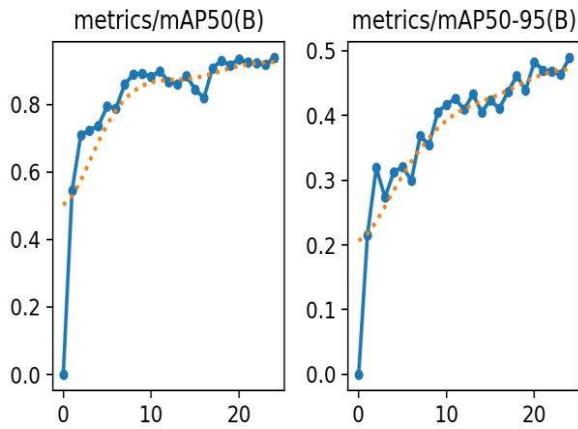


Fig 17 Gráficos de mAP pelas épocas

Na figura 17 temos a presença de duas métricas, a mAP50 e a mAP50-95, a diferença dessas duas métricas consiste em que a primeira mede a precisão média das detecções com um limiar fixo de 50% e a segunda mede a precisão média das detecções com um limiar variando de 50 a 95% e fornece uma visão mais abrangente da precisão da detecção em diferentes níveis de confiança. Ambas as medidas oferecem uma avaliação de desempenho da precisão da rede e nos dois gráficos podemos notar que a partir de 20 épocas há uma estabilização dos valores de mAP.

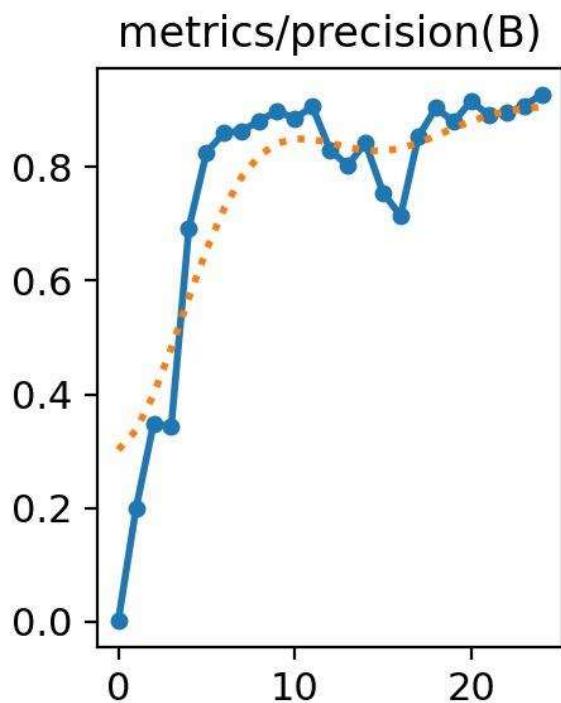


Fig 18 Gráfico de precision pelas épocas

Neste gráfico, temos as medidas de precisão mostradas ao longo das épocas. A precisão nada mais é que a proporção de resultados corretos em relação ao total de resultados. É possível

ver no gráfico que no começo tem um aumento muito grande, até que ali pela época 10, ela se estabiliza, chega até a cair um pouco, porém depois sobe e tem uma estabilização.

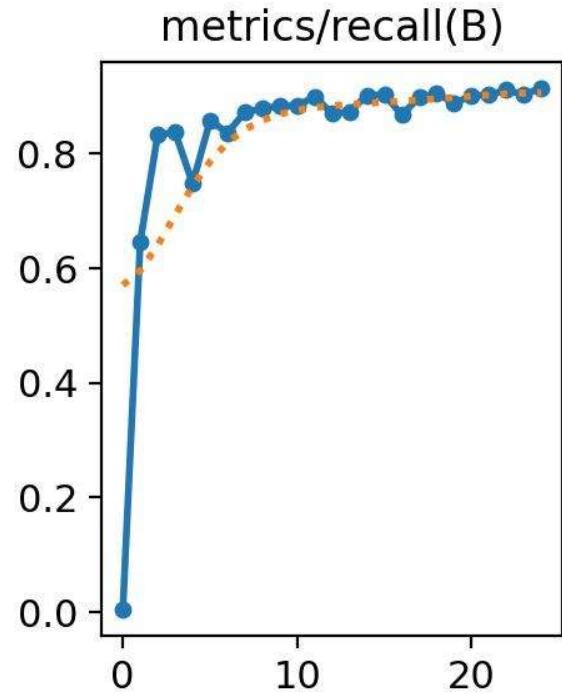


Fig 19 Gráfico de recall pelas épocas

Agora, neste outro gráfico, temos novamente as épocas no eixo X, porém agora no eixo Y se tem o recall, que é uma proporção de resultados corretos positivos em relação ao total de resultados positivos reais. Novamente temos um aumento muito grande no começo e depois ele se estabiliza a partir do 0.9, onde se mantém constante até o final.

Podemos fazer um estudo a partir desses dados que talvez não seja necessário chegar a 25 épocas, tendo em vista que depois das 20 épocas teria um resultado muito parecido, mas com um custo computacional de tempo bem menor do que o anterior.

VII. RESULTADOS E DISCUSSÕES

Então, após essas mudanças no treinamento e na predição, por conta da análise inicial, finalmente podemos realizar o cross-validation para analisar melhor as confiâncias. Para isso, como foi explicado mais em cima, foi feito a predição 20 vezes, onde foi variada a rede treinada, assim como o teste, além da confiança que variou de 0.2 até 0.5, subindo 0.1 em cada vez.

Para fazer essa análise foi usado diversas métricas de avaliação, que serão mostradas agora. A primeira foi a taxa de erro que pode ser representada da seguinte maneira:

$$\text{Taxa de Erro (\%)} = \frac{\text{Predições Erradas}}{\text{Total de carros}} \times 100$$

Fig 20 Fórmula taxa de erro feita no látex.

O gráfico ficou do seguinte jeito:

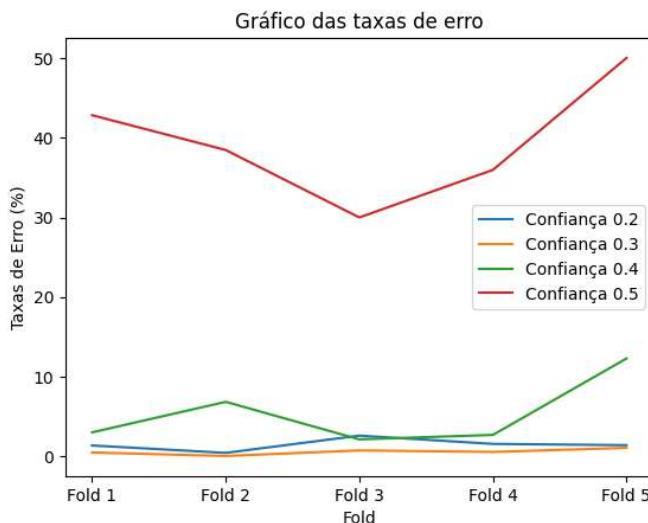


Fig 21 Gráfico das taxas de erro, variando a confiança e o fold.

É possível ver que os testes com a confiança de 0.5 foram muito pior que as outras, tendo uma taxa de erro extremamente alta, e para uma análise melhor dos outros valores de confiança foi feito outro gráfico sem essa confiança maior.

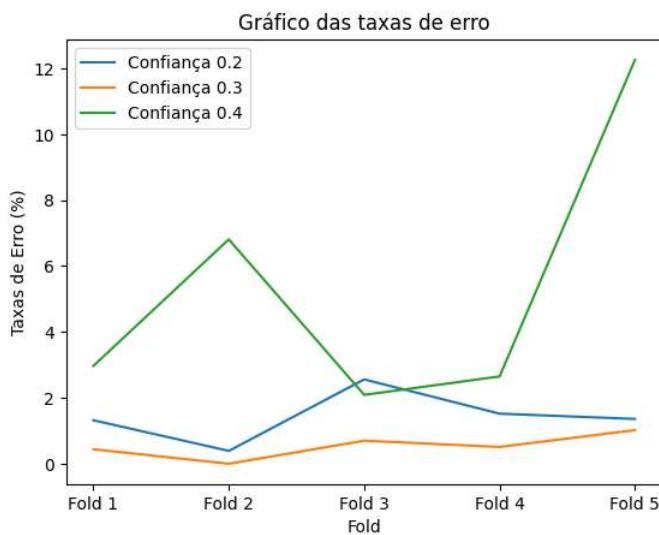


Fig 22 Gráfico das taxas de erro, variando a confiança e o fold, sem a confiança de 0.5.

Agora neste novo gráfico é possível analisar melhor, onde os testes com confiança de 0.3 possui um desempenho melhor do que os outros. Além disso é possível ver que os valores com a confiança de 0.4 tem uma alta variação entre os seus valores, pois no fold3 ele tem uma taxa de erro menor do que a confiança de 0.2, porém no fold5 essa taxa é maior do que 10. Então,

também foi feito uma tabela de média e desvio padrão de cada valor de confiança.

-----TAXA-DE-ERRO-----		
Confiança = 0.2	Média = 1.43 %	Desvio padrão = 0.69 %
Confiança = 0.3	Média = 0.53 %	Desvio padrão = 0.33 %
Confiança = 0.4	Média = 5.36 %	Desvio padrão = 3.83 %
Confiança = 0.5	Média = 39.48 %	Desvio padrão = 6.73 %

Fig 23 Média e desvio padrão das taxas de erro.

Analisando os dados da para ver claramente o que foi falado acima, já que na confiança de 0.4, o desvio padrão é bem maior que o quando os valores de confiança são menores. Também é possível ver que a confiança de 0.3 traz o melhor resultado com média e desvio padrão menores do que as demais. Porém, vamos utilizar outras métricas de avaliação para avaliar melhor.

As próximas métricas que foram usadas são a precisão e a revocação, que permite uma melhor visualização das predições, principalmente para ver se uma rede teve mais falsos positivos ou falsos negativos. A fórmula utilizada para identificar elas foram as seguintes:

$$\text{Precisão (\%)} = \frac{\text{Carros identificados corretamente}}{(\text{Carros identificados corretamente} + \text{Carros não identificados})} \times 100$$

$$\text{Revocação (\%)} = \frac{\text{Carros identificados corretamente}}{(\text{Carros identificados corretamente} + \text{Carros identificados errado})} \times 100$$

Fig 24 Fórmula de precisão e revocação feita no látex.

O gráfico da precisão ficou dessa forma:

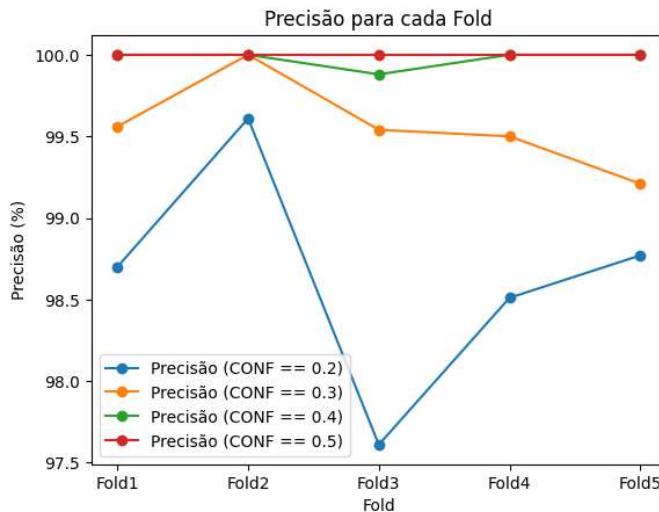


Fig 25 Gráfico das taxas de precisão, variando a confiança e o fold.

A partir disso dá para analisar que todas previsões tiveram uma precisão muito alta, sendo a pior com a confiança de 0.2, isso acontece por conta de que com menor confiança, ele tem maior problema com identificação de partes da imagens como carro, mas não são carros. Porém, não se pode analisar somente com a precisão neste trabalho, tem que levar em conta a revocação, que foi representada pelo gráfico abaixo.

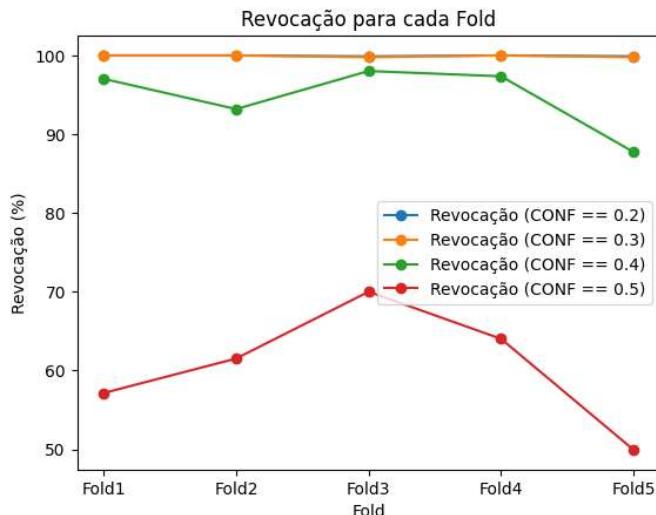


Fig 26 Gráfico das taxas de revocação, variando a confiança e o fold.

Neste gráfico, podemos ver que aqueles com confiâncias mais altas, tem a revocação bem abaixo dos outros. Ao contrário da precisão, aqueles com valor menor de revocação quer dizer que estão deixando de identificar muitos carros, o que também não é bom para o problema. Importante ressaltar que a curva azul, teve o mesmo comportamento da curva laranja, já que ambos tiveram 100% de revocação em todos os folds.

O que pode justificar o comportamento do vermelho em ambos gráficos, é que todos que ele está identificando com carro,

realmente é carro, porém, ele também deixa de identificar uma grande quantidade de carros, por isso ele acaba sendo o melhor no gráfico de precisão e o pior em revocação. Juntando o resultado dos dois gráficos também nos leva para a confiança de 0.3 como melhor resultado. Porém com esses dois valores, podemos ainda calcular outra métrica, o F1 score que é mostrado pela fórmula:

$$F1_score = 2 \times \frac{(\text{Precisão} \times \text{Revocação})}{(\text{Precisão} + \text{Revocação})}$$

Fig 27 Fórmula do F1 score, feita no latex.

O F1 score funciona como um equilíbrio entre a revocação e a precisão, como neste problema se tem confiança com maior problema na revocação e outros com a precisão, o F1 score acaba sendo interessante para avaliar a rede.

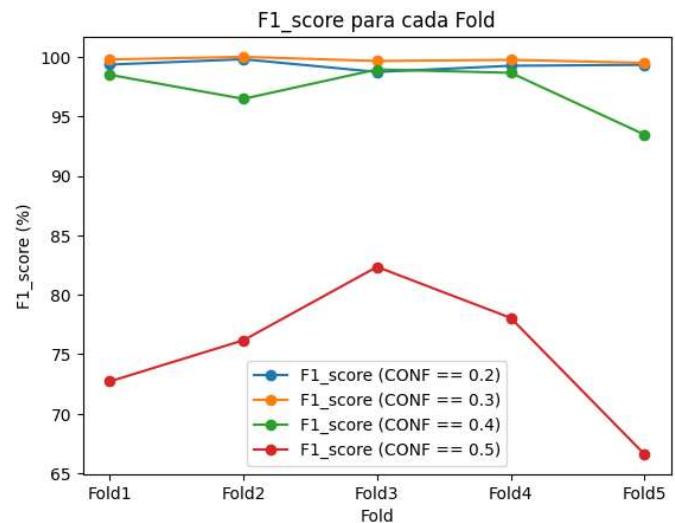


Fig 28 Gráfico do F1 score, variando a confiança e o fold.

Assim como no exemplo da taxa de erro, a confiança de 0.5 é muito pior do que as outras. Então foi feito outro gráfico para melhor visualização.

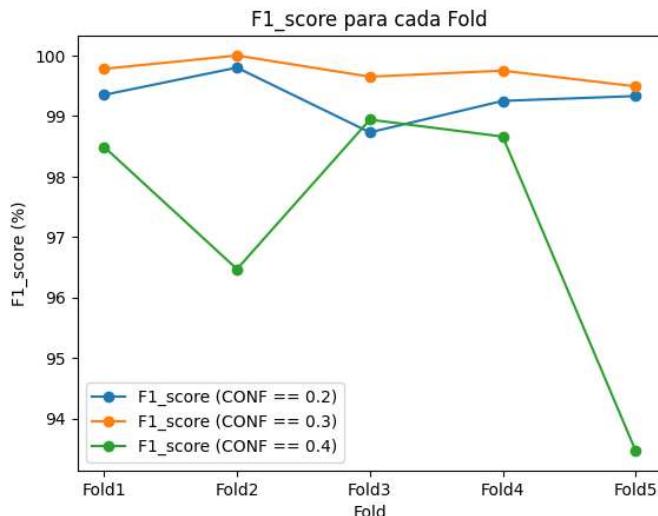


Fig 29 Gráfico das taxas de precisão, variando a confiança e o fold, sem a confiança de 0.5

Analisando este novo gráfico, chegamos novamente que o teste é melhor com aquela com confiança de 0.3, com um F1 score maior que os outros em todos os folds. Logo, chegamos na conclusão que a predição do resultado é melhor quando a confiança usada é de 0.3.

Com os resultados da melhor rede a respeito da predição dos objetos carro e o mapeamento das vagas, foi possível comparar as coordenadas das caixas delimitadoras do modelo e as coordenadas das vagas, sendo factível de analisar a quantidade de vagas preenchidas, vagas disponíveis e a quantidade de carros no estacionamento. Um exemplo do resultado de tal análise pode ser visto na imagem:



Fig 30 Frame de vídeo contendo a análise do local

Esse é o resultado final esperado, onde podemos ver que o carro em movimento não contabiliza no total de vagas ocupadas, todos os carros estão devidamente identificados bem como a quantidade de vagas preenchidas.

VIII.O QUE SERÁ ENTREGUE NO FINAL?

Ao final, será entregue um projeto utilizando redes convolucionais de uma inteligência artificial que a partir das imagens do estacionamento do campus (obtida por drone ou câmera) identifica a presença de veículos nas vagas, respondendo os tópicos apresentados na parte IV, nos objetivos.

Pretendemos responder os tópicos deixando os resultados disponibilizados em uma tabela da seguinte forma:

Opção	Quantidade
Veículos estacionados	X
Vagas Livres	Y
Veículos no estacionamento	Z

Fig 31. Tabela que será gerada no final

As variáveis X, Y e Z serão geradas pelo projeto, depois de rodar a imagem na nossa inteligência artificial.

IX. REFERÊNCIAS BIBLIOGRÁFICAS

- [1] DA MAIA SECO, Álvaro Jorge; GONÇALVES, Jorge Humberto Gaspar; DA COSTA, Américo Henrique Pires. ESTACIONAMENTO. 2008. Tese de Doutorado. Universidade de Coimbra.
- [2] Parking Logix. Is the Search for Parking Making Your Roads Congested? [online]. Disponível em: <<https://parkinglogix.com/search-for-parking-roads-congested/#:~:text=A%20study%20by%20INRIX%20found.time%2C%20gas%2C%20and%20emissions>>. Acesso em: 03/06/2023.
- [3] BARATA, Eduardo; CRUZ, Luis; FERREIRA, João-Pedro. Parking at the UC campus: Problems and solutions. Cities, v. 28, n. 5, p. 406-413, 2011.
- [4] PHANTOM 3 Advanced. Disponível em <<https://www.dji.com/br/phantom-3-adv>>. Acesso em: 03/06/2023.
- [5] Unifesp - Universidade Federal de São Paulo. Unifesp | Conheça o Campus São José dos Campos (Instituto de Ciência e Tecnologia ICT/Unifesp). Disponível em: <<https://www.youtube.com/watch?v=ZHtMwsx70s0>>. Acesso em: 28/05/2023.
- [6] RUSSELL, Stuart; NORVIG, Peter . Inteligência Artificial: tradução Regina Célia Simille. Rio de Janeiro: Elsevier, 2013
- [7] FRANCOIS, Chollet. Deep Learning with Python, Manning Publications. 2017.
- [8] AL-TURJMAN, Fadi; MALEKLOO, Arman. Smart parking in IoT-enabled cities: A survey. Sustainable Cities and Society, v. 49, p. 101608, 2019.
- [9] JABBAR, Waheb A. et al. An IoT Raspberry Pi-based parking management system for smart campus. Internet of Things, v. 14, p. 100387, 2021.
- [10] IDRIS, MY Idna et al. Car park system: A review of smart parking system and its technology. Information technology journal, v. 8, n. 2, p. 101-113, 2009.
- [11] LI, Xin; CHUAH, Mooi Choo; BHATTACHARYA, Subhrajit. Uav assisted smart parking solution. In: 2017 international conference on unmanned aircraft systems (ICUAS). IEEE, 2017. p. 1006-1013.
- [12] SATHASIVAM, Saravanaraj et al. IoT based Parking Management System using Video Processing for Open Space Parking Area. Evolution of Information, Communication and Computing System, p. 92-102, 2022.
- [13] VLAHOGIANNI, Eleni I. et al. A real-time parking prediction system for smart cities. Journal of Intelligent Transportation Systems, v. 20, n. 2, p. 192-204, 2016.
- [14] ÖNCEVARLIK, Dilan Fatma; YILDIZ, Kemal Doruk; GÖREN, Sezer. Deep learning based on-street parking spot detection for smart cities. In: 2019 4th International Conference on Computer Science and Engineering (UBMK). IEEE, 2019. p. 177-182.