Sistema de Monitoramento de Aduanas: Deep Learning para Contagem Automatizada de Veículos, Pedestres e Fiscalizações Veiculares em Áreas de Controle Fronteiriço

Rafhael Pereira Zrenner
Centro de Tecnologias Aplicadas Itaipu Parquetec
Paraná, Brasil
rafhael.zrenner@itaipuparquetec.org.br

Layon Luciano Lucena Alves
Centro de Tecnologias Aplicadas Itaipu Parquetec
Paraná, Brasil
layon.alves@itaipuparquetec.org.br

Érick Oliveira Rodrigues
Universidade Tecnológica Federal do Paraná UTFPR
Paraná, Brasil
erickrodrigues@utfpr.edu.br

***Resumo*—Este artigo apresenta uma solução que utiliza visão computacional para detectar, classificar e contabilizar veículos e pedestres em vídeos capturados por câmeras convencionais em uma unidade aduaneira. O sistema proposto integra processamento de imagens, contagem automática e publicação dos dados em uma plataforma web para visualização em tempo real. Implantado na Aduana da Ponte Internacional da Amizade (PIA), em Foz do Iguaçu (PR), em parceria com a Receita Federal do Brasil, o sistema alcançou um F1-Score médio superior a 93% na contagem de carros, motos, caminhões e pedestres, e 90,1% para veículos fiscalizados.**

***Palavras-chave*—Visão Computacional, Fronteiras, Aduanas, Contagem, Veículos, Pedestres**

I. Introdução

O projeto "Muralha Inteligente" é uma iniciativa conjunta da Itaipu Binacional, da Fundação Parque Tecnológico Itaipu - Brasil[1] e da Receita Federal do Brasil (RFB), com o objetivo de desenvolver e implementar inovações tecnológicas voltadas ao aumento da segurança em regiões de fronteira no Brasil [1]. Com foco específico em Foz do Iguaçu - Paraná, o projeto busca utilizar tecnologias inteligentes no combate ao contrabando e ao descaminho, promovendo ações de fiscalização e controle mais eficazes [2].

Este trabalho está alinhado ao escopo técnico do projeto Muralha Inteligente, com foco no eixo de Integração de Aplicações, por meio da aplicação de técnicas de Inteligência Artificial (IA). Especificamente, propõe-se uma solução que emprega visão computacional para a contabilização automatizada de alvos em área aduaneira, com distinção entre carros, motos, ônibus, caminhões e pedestres.

Nos últimos anos, diversos estudos têm explorado o uso de visão computacional para tarefas como detecção, contagem e rastreamento de veículos e pedestres em ambientes urbanos. Entretanto, grande parte dessas abordagens considera cenários genéricos, com iluminação controlada e infraestrutura idealizada, o que limita sua aplicabilidade em situações reais mais adversas.

Diante desse cenário, este artigo apresenta o desenvolvimento e a validação de uma solução leve e modular para detecção e contagem automática de veículos, pedestres e fiscalizações em áreas aduaneiras, utilizando visão computacional. A proposta busca atender a requisitos operacionais específicos, como limitações de hardware, alta variabilidade de iluminação e ausência de controle sobre o fluxo. Diferentemente de abordagens comuns na literatura, a solução aqui descrita foi implantada em um ambiente real de fronteira, com validação prática em parceria com a RFB.

II. Trabalhos Relacionados

Diversas abordagens têm explorado a aplicação de visão computacional para a contagem e classificação de veículos e pedestres em ambientes urbanos. A maioria das soluções utiliza algoritmos da família YOLO combinados com bibliotecas como OpenCV [3], além de integrações com sistemas de rastreamento ou ferramentas de visualização de dados.

Entre essas iniciativas, destaca-se uma aplicação web construída com a ferramenta OpenDataCam, utilizando YOLOv3 para a detecção de veículos em vídeos aéreos capturados por drones [4]. A solução fornece relatórios e dashboards com dados de tráfego urbano, mas depende fortemente de hardware especializado e câmeras com perspectiva ortogonal, diferentemente da proposta deste trabalho, que utiliza câmeras fixas em cenários reais de fronteira.

Outro estudo apresenta um sistema para contagem de veículos em uma rotatória, utilizando vídeos de câmeras públicas e algoritmos em Python com YOLO [5]. Embora tecnicamente semelhante, a aplicação está limitada a um ponto urbano específico, sem integração com plataformas de análise de dados nem validação em ambientes de alto fluxo.

Uma solução voltada à contagem de pedestres faz uso de detecção combinada com rastreamento, utilizando YOLOv3 e

[1]A instituição anteriormente conhecida como Fundação Parque Tecnológico Itaipu - Brasil passou a se chamar Itaipu Parquetec.

algoritmos de correlação discriminativa [6]. Essa abordagem, embora robusta, depende da reidentificação de alvos ao longo do tempo, o que aumenta a complexidade computacional. Em contraste, a proposta deste trabalho adota uma estrutura mais simples, sem rastreamento, priorizando eficiência e leveza. O sistema realiza a contagem verificando se o centro de cada detecção passa por regiões específicas da imagem, conhecidas como regiões de interesse. Para identificar entradas e saídas, ele apenas compara as detecções do quadro atual com as do quadro anterior. Isso elimina a necessidade de acompanhar individualmente cada objeto, reduzindo o custo computacional e tornando o processamento mais adequado a aplicações em tempo real.

Em outro estudo, é apresentada a quantificação do fluxo de veículos em áreas urbanas utilizando YOLOv5 para detecção e o algoritmo DeepSORT para rastreamento [7]. Embora empregue um modelo mais recente, que oferece maior precisão, essa abordagem demanda maior capacidade computacional, especialmente para manter o rastreamento em tempo real, devido à complexidade do DeepSORT. Em contraste, a proposta aqui descrita adota YOLOv4-tiny, uma arquitetura mais leve, priorizando velocidade de inferência e eficiência em ambientes com recursos computacionais limitados.

Uma aplicação voltada para segurança viária propõe um sistema de detecção de pedestres realizando travessias de risco, empregando YOLOv4-tiny com o algoritmo SORT para rastreamento [8]. Embora compartilhe a mesma arquitetura de detecção utilizada neste trabalho, seu foco está na identificação de pedestres em situações de risco, com ênfase na análise comportamental e na segurança viária. Em contraste, o presente estudo concentra-se na quantificação de pedestres, visando à contabilização para análise de fluxo, sem abordar aspectos comportamentais ou de risco.

Também foi avaliada a viabilidade da contagem automatizada de pessoas em shopping centers utilizando exclusivamente a biblioteca OpenCV. A proposta consiste na adaptação de um código open source fundamentado na detecção de centróides, com posterior rastreamento e registro em banco de dados [9]. Embora funcional, a solução apresentou limitações em cenários com ângulos desfavoráveis, obstáculos ou fluxo intenso, evidenciando a sensibilidade do método às condições de captura.

Em cenários controlados, uma proposta alternativa avalia a contagem de pessoas em ambientes internos utilizando o modelo YOLOv4-Tiny executado localmente em um Raspberry Pi 4 com lente fisheye [10]. Embora também adote uma arquitetura leve para detecção em tempo real, o contexto de aplicação difere significativamente: essa solução foi projetada para operar em cenários controlados e com baixo fluxo, como salas fechadas, enquanto a proposta deste trabalho foi desenvolvida para ambientes fronteiriços, caracterizados por alta variabilidade de iluminação, fluxo intenso e múltiplas classes de objetos. Além disso, a solução aqui descrita distingue-se por sua integração a uma plataforma web com dashboards interativas, otimizando o suporte à tomada de decisão.

Esses estudos demonstram a versatilidade das técnicas de visão computacional para o monitoramento do tráfego de veículos e pedestres. No entanto, ainda há espaço para soluções mais integradas a plataformas web com recursos de business intelligence (BI), como a apresentada neste artigo. A proposta aqui desenvolvida distingue-se por ser estruturada como um *framework*, com foco na contagem direta de veículos, veículos fiscalizados e pedestres, validação prática em ambiente fronteiriço e integração com dashboards para visualização e análise em tempo real.

III. Materiais e Métodos

A solução proposta tem como objetivo principal realizar a contagem do fluxo de veículos e pedestres que transitam diariamente pela aduana de Foz do Iguaçu, além de contabilizar os veículos submetidos à fiscalização.

O problema abordado é caracterizado como uma tarefa de detecção de objetos, focando na detecção e classificação de pedestres e diferentes categorias de veículos (carros, motos, ônibus e caminhões). A detecção é realizada por meio de delimitações do tipo bounding box, permitindo a identificação precisa dos objetos de interesse nas imagens. Para isso, são utilizadas transmissões de vídeo provenientes de câmeras convencionais, sem inteligência embarcada.

Com base nesse cenário, foram analisados os requisitos do sistema e definidas as tecnologias adequadas para o desenvolvimento de um algoritmo de visão computacional capaz de atender a essas demandas. Inicialmente, foi estruturado um fluxo de trabalho para guiar o desenvolvimento da solução, conforme ilustrado na Fig. 1.

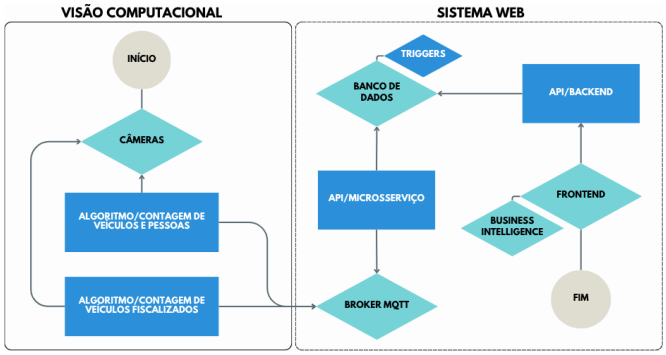

Fig. 1. Visão geral do fluxo das etapas aplicada no desenvolvimento da solução.

Conforme ilustrado na Fig. 1, o fluxo foi segmentado em duas partes principais: Visão Computacional (à esquerda) e Sistema Web (à direita). Essa separação tem como objetivo modularizar a solução, evitando a construção de um sistema monolítico e, consequentemente, facilitando a manutenção, a escalabilidade e a integração entre os componentes.

Na Fig. 1 (à esquerda), encontra-se a estrutura relacionada à visão computacional, que é ativada a partir do acesso ao streaming das câmeras. Esse fluxo é processado por dois algoritmos complementares: o primeiro é responsável pela detecção, classificação e contagem de veículos e pedestres

que circulam pela área aduaneira, tanto no sentido de entrada quanto de saída do país (Brasil–Paraguai); o segundo algoritmo atua na identificação de veículos fiscalizados que ingressam no Brasil, filtrando apenas aqueles que passaram por inspeção. Em seguida, os dados são organizados e enviados para múltiplos tópicos em um broker MQTT (Message Queuing Telemetry Transport) [11].

Na Fig. 1 (à direita), está representada a estrutura do sistema web, que se inicia pela API (Application Programming Interface) responsável por se conectar ao broker MQTT e capturar os dados publicados nos diferentes tópicos. Esses dados são então registrados em um banco de dados. Em seguida, triggers são utilizados para estruturar as informações de forma organizada, facilitando o consumo pela API no backend e sua posterior exibição no frontend, que incorpora *dashboards* e visualizações interativas com recursos de BI.

A. Anotação de Dados

Para dar início ao desenvolvimento da solução, foi utilizada a ferramenta de anotação Computer Vision Annotation Tool (CVAT) [12]. As imagens foram obtidas a partir de câmeras da RFB localizadas na Aduana da PIA, abrangendo tanto o fluxo de entrada quanto de saída do país. Foram realizados registros em vídeo com duração entre 15 e 30 minutos, para garantir a captura de um número suficiente de veículos e pedestres, contemplando diferentes níveis de movimentação ao longo do dia. As gravações foram feitas em dias aleatórios nos períodos da manhã, tarde e noite, com o objetivo de abranger uma ampla variedade de condições de iluminação (luz natural e iluminação artificial) e clima (tempo ensolarado, chuvoso, nublado e parcialmente ensolarado), garantindo maior adaptabilidade ao modelo frente às variações do ambiente.

A partir dos vídeos obtidos, foram extraídas 1.906 imagens utilizando um script desenvolvido em Python com o auxílio da biblioteca OpenCV. O script percorre cada vídeo quadro a quadro e realiza a extração de frames com base em um intervalo fixo de tempo, definido em segundos. Essa abordagem permite reduzir a redundância causada por frames visualmente semelhantes, garantindo uma amostragem mais representativa do conteúdo dos vídeos. Além disso, os frames extraídos são redimensionados para uma resolução padronizada de 1280×720 pixels, otimizando o processo posterior de anotação e treinamento do modelo de detecção.

As imagens selecionadas foram anotadas manualmente utilizando a ferramenta CVAT. A Fig. 2 apresenta dois exemplos de imagens durante o processo de anotação: à esquerda, uma cena capturada por uma câmera posicionada em uma área de fluxo exclusivo de pedestres, na qual todos os indivíduos são manualmente marcados; à direita, uma imagem de uma câmera dedicada ao monitoramento prioritário do tráfego de veículos, onde todas as classes-alvo presentes na cena também são rotuladas.

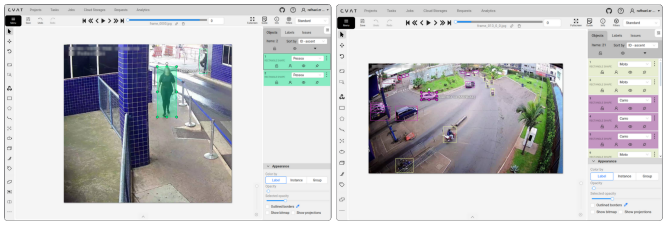

Fig. 2. Exemplo de anotação de imagens com a ferramenta CVAT.

B. Arquitetura de Detecção, Classificação de Objetos e Treinamento

Para atender aos requisitos de detecção de objetos, foi adotada a arquitetura de rede neural convolucional YOLOv4-tiny, uma versão otimizada da família YOLO (You Only Look Once), projetada para detecção de objetos em tempo real. A adoção da arquitetura justifica-se, principalmente, por sua leveza, alta taxa de processamento e eficiência computacional. Com apenas 23,1MB, o modelo é significativamente mais compacto que a versão completa da YOLOv4, que possui 245MB, o que facilita sua implantação em sistemas com recursos de hardware limitados. Além disso, a YOLOv4-tiny atinge até 371 FPS em uma GPU 1080Ti, sendo consideravelmente mais rápida que a versão padrão, que atinge no máximo 63 FPS em resoluções reduzidas. Essa alta taxa de quadros por segundo torna a YOLOv4-tiny adequada para aplicações em tempo real [13].

Diferentemente de abordagens tradicionais, a YOLO realiza a detecção a partir de uma única varredura na imagem, o que permite maior eficiência computacional. A imagem de entrada é dividida em múltiplas sub-regiões, em que a rede estima células e atribui a cada uma a probabilidade de conter um objeto. Com base nessas estimativas, apenas as células com probabilidade acima de um limiar pré-definido são mantidas, descartando as demais, o que otimiza o processo de detecção e classificação de objetos relevantes [14] [8].

A arquitetura YOLO, baseada em técnicas de deep learning e redes neurais convolucionais para detecção de objetos em imagens, apresenta compatibilidade com a tecnologia CUDA (Compute Unified Device Architecture), desenvolvida pela NVIDIA, a qual é voltada à computação massivamente paralela. Essa compatibilidade permite que o treinamento e a inferência da rede neural sejam acelerados por meio do uso de Unidades de Processamento Gráfico (GPUs), resultando em uma redução significativa no tempo de treinamento e processamento [15] [16] [6].

Para o treinamento, utilizou-se o *framework* Darknet, uma rede neural de código aberto implementada em C e CUDA, com suporte à execução em CPU e GPU. O sistema oferece suporte opcional ao OpenCV, para ampliar a compatibilidade com diversos formatos de imagem, e ao CUDA, para aceleração do processamento em GPU. O Darknet é amplamente reconhecido por ser a base da arquitetura de detecção de objetos em tempo real YOLO [8]. Os dados anotados foram convertidos para o formato nativo do Darknet

(arquivos .txt no estilo YOLO, um por imagem), e o modelo treinado foi exportado nos formatos .cfg e .weights, utilizados posteriormente na etapa de inferência.

C. Inferência

Para realizar a inferência, entendida neste contexto como o processo de detecção e classificação de objetos em imagens com base nos padrões aprendidos durante o treinamento, foi utilizado o OpenCV, uma biblioteca de código aberto voltada para aplicações de visão computacional. Entre seus diversos módulos, destaca-se o módulo DNN (Deep Neural Network), responsável pelas funcionalidades relacionadas ao aprendizado profundo. Esse módulo oferece suporte à inferência com modelos pré-treinados provenientes de diferentes *frameworks*, como Caffe, TensorFlow, Torch e Darknet. Isso permite que a execução dos modelos treinados ocorra diretamente no OpenCV, sem necessidade de ferramentas externas. [8].

D. Integração Web

Em complemento à solução de visão computacional, foi planejado o desenvolvimento de uma plataforma web para facilitar a visualização e análise dos dados gerados na contabilização dos alvos. A plataforma foi segmentada em componentes distintos, buscando garantir modularidade, escalabilidade e clareza na manutenção do sistema, integrando serviços de mensageria, armazenamento de dados, API e interface com o usuário.

Para o serviço de mensageria, foi adotado o protocolo MQTT, que segue o modelo publicação/assinatura. O broker responsável por essa comunicação é o Mosquitto, escolhido por sua leveza e aderência ao padrão MQTT [11]. Para facilitar essa comunicação, utilizou-se a biblioteca Paho MQTT, que fornece uma interface cliente para conexão com o broker, publicação de mensagens e assinatura de tópicos [17].

O armazenamento dos dados coletados foi realizado em um banco de dados relacional MySQL, solução consolidada, de código aberto, já compatível com a infraestrutura existente da RFB e conhecida pela equipe técnica [18].

No backend, empregou-se o ambiente Node.js, com o *framework* Express.js, amplamente utilizado na criação de APIs e servidores web leves e performáticos [19] [20].

A interface com o usuário foi construída com a biblioteca React, que possibilita o desenvolvimento de componentes reutilizáveis e interfaces responsivas [21]. Para a visualização gráfica e camada de BI, integrou-se a biblioteca ApexCharts, permitindo a criação de gráficos interativos e dinâmicos diretamente na interface web [22].

Essa integração completa resulta em uma plataforma compatível com as necessidades da RFB, capaz de operar em tempo real e fornecer informações para monitoramento eficiente do tráfego fronteiriço e apoio à tomada de decisões.

E. Pré-processamento de Dados e Configuração do Treinamento

Para a extração das imagens utilizadas no treinamento, foi desenvolvido um script em Python com a biblioteca OpenCV. O script permite a seleção de um ou mais vídeos armazenados localmente e realiza a extração de quadros com base em um intervalo de tempo configurável, convertido a partir da taxa de quadros por segundo (FPS) de cada vídeo. Os frames extraídos são redimensionados para resolução de 1280×720 pixels e salvos no formato JPEG, organizados em subpastas nomeadas conforme o vídeo de origem. Para otimizar o desempenho em grandes volumes de dados, a aplicação foi implementada com suporte a multithreading, permitindo o processamento simultâneo de múltiplos arquivos de vídeos.

A rotulagem das imagens seguiu as cinco classes de interesse pré-definidas: carro, moto, ônibus, caminhão e pedestre. A divisão do conjunto de dados foi realizada por meio da técnica Holdout, com 80% das imagens destinadas ao treinamento (1.526) e 20% para teste e validação (380), totalizando 1.906 amostras.

Para garantir um treinamento eficiente e com tempos reduzidos, foi utilizada uma máquina de alto desempenho equipada com sistema Ubuntu 24.04, 126 GB de RAM, GPU NVIDIA RTX 4500 Ada (24 GB) e processador AMD Ryzen Threadripper 3970X (32 núcleos).

Para otimizar a detecção de objetos com a arquitetura YOLOv4-tiny, foram ajustados hiperparâmetros específicos no arquivo de configuração `.cfg`, conforme detalhado na Tabela I, que descreve os parâmetros aplicados durante o treinamento da rede. Com o uso da máquina de alto desempenho e esse conjunto de ajustes, o processo de treinamento foi concluído em aproximadamente 80 minutos.

TABELA I
HIPERPARÂMETROS UTILIZADOS NA CONFIGURAÇÃO DA YOLOV4-TINY

Hiperparâmetro	**Valor Utilizado**
`batch`	64
`subdivisions`	1
`width` x `height`	416 × 416
`max_batches`	12000
`steps`	9600, 10800
`random`	1
`learning_rate`	0.0005
`iou_normalizer`	0.1
`anchors`	8, 20, 17, 29, 21, 62, 36, 50, 50, 92, 123, 203

Para a validação do treinamento da rede, foi utilizada a flag `-map` durante a execução no *framework* Darknet, permitindo o cálculo automático da mean average precision (mAP) a cada iteração. A mAP é uma medida amplamente empregada na avaliação de modelos de detecção de objetos, por considerar a média das precisões em diferentes níveis de recall e limiares de confiança. A ativação dessa flag possibilitou o monitoramento contínuo do desempenho das classes-alvo ao longo do processo de treinamento, facilitando a identificação do ponto de convergência ideal.

F. Integração com o Sistema Web

A integração entre os algoritmos de contagem de veículos e pedestres e a plataforma web é realizada via protocolo MQTT, utilizando o *broker* Mosquitto como intermediário. Cada tipo

de objeto detectado (carro, moto, ônibus, caminhão e pedestre) é associado a um tópico específico. As mensagens publicadas seguem dois formatos principais: (i) `ID_CAMERA DIREÇÃO`, indicando a contagem de um novo alvo, com o identificador da câmera e a direção do movimento (entrada ou saída); e (ii) `ID_CAMERA DATA/HORA_INÍCIO DATA/HORA_FIM TEMPO_FISCALIZAÇÃO_SEGUNDOS`, representando o tempo de fiscalização de um veículo, incluindo o momento de início, fim e a duração total em segundos. Para assegurar o funcionamento contínuo do sistema, mesmo na ausência de detecções, os algoritmos enviam periodicamente (a cada 60 segundos) mensagens com valor zero, sinalizando que a câmera e o algoritmo seguem operando normalmente.

IV. Resultados e Discussões

Com a metodologia estabelecida, foram conduzidos experimentos para avaliar a eficácia da solução proposta na detecção e contagem de veículos e pedestres, bem como na integração e transmissão dos dados em tempo real. Os testes foram realizados utilizando o conjunto de dados anotado e a arquitetura YOLOv4-tiny treinada conforme os parâmetros previamente descritos. O desempenho do modelo foi mensurado por meio da medida mAP e o F1-Score, aplicados às diferentes classes-alvo. Além disso, foram analisadas situações específicas como o impacto da oclusão, variações de iluminação e sobreposição de objetos, fatores comuns em ambientes fronteiriços e que podem comprometer a acurácia da detecção.

A. Avaliação de Desempenho em Detecção e Classificação

O treinamento do modelo utilizando os hiperparâmetros mencionados na Tabela I resultou em um mAP geral de aproximadamente 72%, considerando as cinco classes de interesse: carros, motos, ônibus, caminhões e pedestres. Essa medida reflete a média das precisões individuais ponderadas pela taxa de recuperação (recall) em diferentes limiares de confiança. A Tabela II apresenta o desempenho por classe, com os valores de precision, recall e F1-score obtidos sobre o conjunto de teste. A Fig. 3 apresenta o gráfico com o mAP geral e a função de perda (loss) ao longo das 12.000 iterações de treinamento. No eixo da esquerda (em azul), observa-se a redução progressiva da loss, indicando que o modelo está aprendendo e ajustando seus pesos corretamente. Já no eixo da direita (em vermelho), vemos a evolução do mAP, que alcança aproximadamente 71% no final do treinamento, valor considerado satisfatório para aplicações práticas.

TABELA II
Medidas de desempenho por classe nos testes

Classe	Precision (%)	Recall (%)	F1-Score (%)
Carro	94,29	76,29	84,34
Moto	94,41	32,20	48,02
Caminhão	86,61	87,75	87,17
Ônibus	84,62	74,58	79,28
Pedestre	84,67	47,70	61,02

Os resultados apresentados na Tabela II indicam que as classes caminhão, carro e ônibus obtiveram um equilíbrio entre precisão e `recall`, com valores de F1-score acima de 71,17%, sendo que o caminhão alcançou o melhor resultado, com 87,17%. Por outro lado, as classes moto e pedestre apresentaram desempenho significativamente inferior, com destaque para o baixo `recall` da classe moto (32,20%), o que compromete sua capacidade de detecção. Esses resultados sugerem a necessidade de ajustes no balanceamento da base de dados.

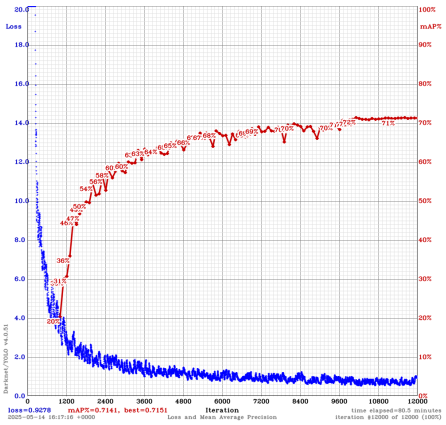

Fig. 3. Gráfico de treinamento - mAP (vermelho) e função de perda (azul).

B. Avaliação de Desempenho em Contabilização

Com a aplicação em operação, foram conduzidos testes em cenários reais utilizando múltiplas câmeras instaladas nas áreas de entrada e saída da aduana, tanto para veículos quanto para pedestres, em diferentes períodos do dia (matutino, vespertino e noturno).

As medidas de desempenho da solução foram avaliadas por meio do F1-Score, o qual considera o equilíbrio entre precisão e recall. Para isso, realizou-se a contagem manual da quantidade real de veículos e pedestres que transitaram pela área monitorada, a fim de compará-la com os resultados fornecidos automaticamente pela solução. Essa abordagem permitiu validar a acurácia do sistema na contagem dos veículos e pedestres.

A contagem automatizada é realizada por meio de um algoritmo que detecta os veículos e os pedestres com base na arquitetura YOLOv4-tiny e aplica a lógica de interseção entre polígonos definidos em regiões de interesse (áreas de contagem). A cada frame processado, o algoritmo calcula o centroide da *bounding box* de cada objeto detectado e classificado e constrói um quadrado ao redor desse ponto. Esses controides são então comparados com as áreas de contagem (definidas como polígonos fixos), utilizando operações geométricas de interseção da biblioteca Shapely. Sempre que um objeto previamente detectado deixa de ser identificado em uma região de passagem, o sistema interpreta como uma nova passagem e registra essa ocorrência.

Uma possível melhoria futura seria a inclusão de mecanismos de rastreamento entre frames, para lidar com situações em que um objeto identificado em um quadro deixe momentanea-

mente de ser detectado e volte a aparecer em quadros subsequentes. Esse aprimoramento permitiria evitar duplicações na contabilização e tornaria o método mais robusto. Por outro lado, essa abordagem evita o uso de bibliotecas de rastreamento mais complexos, o que reduz o custo computacional.

A lógica utilizada na contabilização de carros fiscalizados compartilha princípios semelhantes ao algoritmo de contagem de veículos e pedestres, especialmente na forma como as regiões de interesse são definidas e monitoradas. Ambas as abordagens utilizam polígonos predefinidos (áreas de fiscalização) para identificar a presença de carros, além de operar sobre fluxos contínuos de vídeo. No entanto, a contagem de carros fiscalizados introduz mecanismos adicionais de persistência temporal, como o controle de tempo de permanência em uma vaga e a associação de eventos de entrada e saída. A detecção é validada apenas quando o veículo permanece na área monitorada por um período superior a um limiar mínimo, o que permite distinguir fiscalizações reais de passagens rápidas ou falhas momentâneas na detecção. Como vantagem, essa abordagem reduz significativamente o número de falsos positivos e permite o cálculo do tempo de fiscalização.

Os resultados para a entrada de veículos no Brasil (Tabela III) mostra bom desempenho para carro e moto, enquanto ônibus apresentou valores mais baixos, provavelmente pelo baixo número de amostras. À noite, o desempenho do sistema foi levemente afetado pela baixa iluminação, resultando em uma queda de até 7% no F1-Score de carros e motos, e de 5% para ônibus; não houve amostras de caminhão.

TABELA III
CÂMERA DE ENTRADA DE VEÍCULOS AO PAÍS

Rótulo	TP	FP	FN	Precision	Recall	F1-Score	Período
Carro	86	0	3	1	0,96	0,98	Vespetino
Moto	55	3	0	0,94	1	0,97	Vespetino
Carro	347	19	0	0,94	1	0,97	Vespetino
Moto	338	21	0	0,94	1	0,96	Vespetino
Ônibus	3	0	1	1	0,75	0,85	Vespetino
Carro	50	0	9	1	0,84	0,91	Noturno
Moto	18	4	0	0,81	1	0,90	Noturno
Ônibus	2	0	1	1	0,66	0,80	Noturno

Para a entrada de pedestres (Tabela IV), o modelo obteve F1-Score bastante elevado no período vespertino. No período noturno não houve detecções, pois a pista de pedestres é fechada, impossibilitando o cálculo das medidas.

TABELA IV
CÂMERA DE ENTRADA DE PEDESTRES AO PAÍS

Rótulo	TP	FP	FN	Precision	Recall	F1-Score	Período
Pessoa	82	0	0	1	1	1	Vespertino
Pessoa	105	1	0	0,99	1	0,99	Vespertino

Na saída de veículos do Brasil (Tabela V), o modelo manteve o F1-Score elevado para carro, moto e caminhão (até 1), embora ônibus tenha registrado desempenho inferior (0,80). À noite, os índices permaneceram altos, com destaque para carro (1,0), moto (0,98) e caminhão (0,90); não houve amostras de ônibus.

TABELA V
CÂMERA DE SAÍDA DE VEÍCULOS DO PAÍS

Rótulo	TP	FP	FN	Precision	Recall	F1-Score	Período
Carro	69	0	3	1	0,95	0,97	Vespertino
Moto	44	0	1	1	0,97	0,98	Vespertino
Caminhão	8	0	0	1	1	1	Vespertino
Carro	157	0	6	1	0,96	0,98	Vespertino
Moto	165	16	0	0,91	1	0,95	Vespertino
Caminhão	13	3	0	0,81	1	0,89	Vespertino
Ônibus	2	1	0	0,66	1	0,80	Vespertino
Carro	68	0	0	1	1	1	Noturno
Moto	29	1	0	0,96	1	0,98	Noturno
Caminhão	5	1	0	0,83	1	0,90	Noturno

Na saída de pedestres do Brasil, os resultados vespertinos (Tabela VI) foram muito positivos (F1-Score de 0,90 e 0,96), refletindo alta taxa de recall. No período noturno, o desempenho também foi satisfatório (F1-Score de 0,91).

TABELA VI
CÂMERA DE SAÍDA DE PEDESTRES AO PAÍS

Rótulo	TP	FP	FN	Precision	Recall	F1-Score	Período
Pedestre	15	3	0	0,83	1	0,90	Vespertino
Pedestre	81	0	6	1	0,93	0,96	Vespertino
Pedestre	11	0	2	1	0,84	0,91	Noturno

Além da contagem geral de veículos e pedestres, foi avaliado o desempenho do modelo na detecção de veículos que passaram por fiscalização na entrada do país. Essa análise tem importância prática, pois contribui diretamente para o monitoramento da atuação fiscal na área aduaneira.

Por fim, na detecção de veículos fiscalizados (Tabela VII), o modelo alcançou F1-Score de 0,90, com recall alto (1), embora com alguns falsos positivos (Precision de 0,82).

TABELA VII
CÂMERAS DE ENTRADA DE VEÍCULOS (FISCALIZADOS) AO PAÍS

Rótulo	TP	FP	FN	Precision	Recall	F1-Score
Carro (Fiscalizado)	32	7	0	0,82	1	0,90

A Tabela VIII apresenta os valores médios de F1-Score obtidos por classe durante os testes de contabilização realizados em ambiente de produção. Observa-se que as classes carro, moto, pedestre e caminhão demonstraram excelente desempenho, com F1-Scores superiores a 93%, indicando elevada precisão e robustez na detecção desses objetos. A classe ônibus apresentou desempenho inferior, com F1-Score de 81,9%, possivelmente em função da menor representatividade dessa categoria nos vídeos analisados, dado o fluxo mais reduzido em comparação aos demais veículos. Por fim, a subcategoria

carros fiscalizados, que representa apenas os veículos efetivamente abordados pela fiscalização, atingiu um F1-Score de 90,1%, valor que se mostra competitivo em relação à literatura disponível, considerando que muitos trabalhos similares não reportam medidas formais ou obtêm resultados em condições menos desafiadoras que o ambiente fronteiriço avaliado neste estudo.

TABELA VIII
F1-SCORE MÉDIO POR CLASSE COM BASE NOS TESTES DE CONTABILIZAÇÃO EM PRODUÇÃO

Classe	F1-Score Médio (%)
Carro	97,1
Moto	96,1
Caminhão	93,5
Ônibus	81,9
Pedestre	95,6
Carros (Fiscalizados)	90,1

Na Fig. 4, são apresentadas as telas que ilustram o funcionamento do módulo de visão computacional na aduana, com destaque para os algoritmos em operação voltados à contagem de veículos, veículos fiscalizados e pedestres. Em seguida, a Fig. 5 exibe a interface da plataforma web voltada ao monitoramento de veículos. Esse painel reúne dados operacionais, como o total de veículos, a distribuição por tipo (carros, motos, caminhões e ônibus), a porcentagem de entradas e saídas do país, além do total diário ao longo do período analisado. A plataforma também oferece uma funcionalidade de acompanhamento em tempo real. Há ainda telas específicas para o monitoramento de pedestres e veículos fiscalizados, com funcionalidades semelhantes.

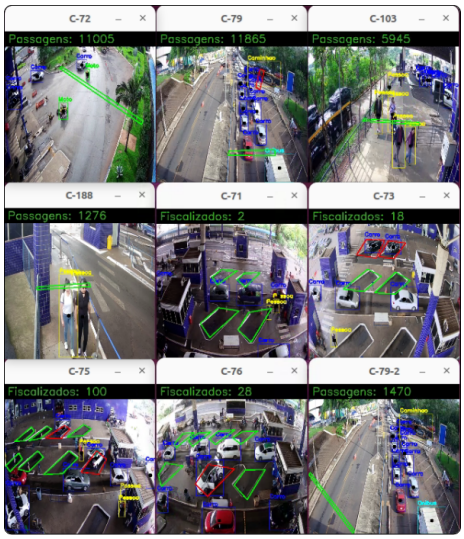

Fig. 4. Principais câmeras sendo utilizadas pelos algoritmos de contagem de veículos, veículos fiscalizados e pedestres.

C. Limitações Técnicas

Apesar dos resultados satisfatórios, o sistema apresenta algumas limitações técnicas, especialmente em cenários mais

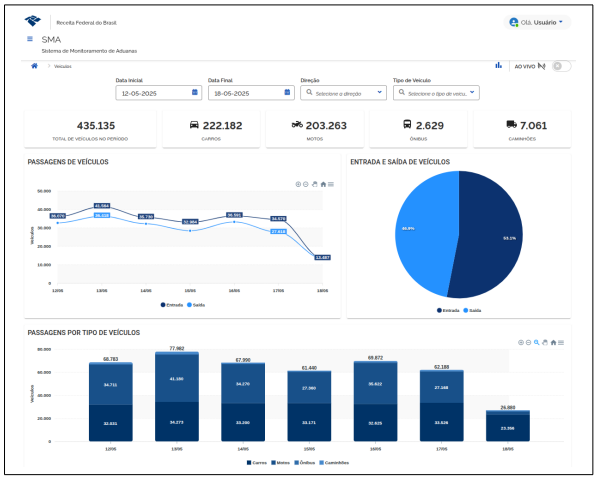

Fig. 5. Tela de veículos da plataforma web.

desafiadores. Conforme ilustrado na Fig. 6, observou-se que a detecção e classificação de motocicletas torna-se significativamente menos precisa quando estas estão distantes da câmera ou posicionadas próximas ao limite superior do enquadramento, como indicado pelo círculo vermelho. Nesses casos, a resolução aparente é reduzida, e o tamanho do objeto na imagem torna-se pequeno, dificultando a extração de características relevantes pelo modelo. Essa limitação é típica da arquitetura YOLOv4-tiny, que, embora apresente vantagens em termos de velocidade de inferência e baixo consumo de recursos computacionais, compromete parte da acurácia em alvos pequenos ou com baixa definição. Trata-se de um trade-off comum em modelos leves, que privilegiam desempenho em tempo real às custas da capacidade de análise detalhada.

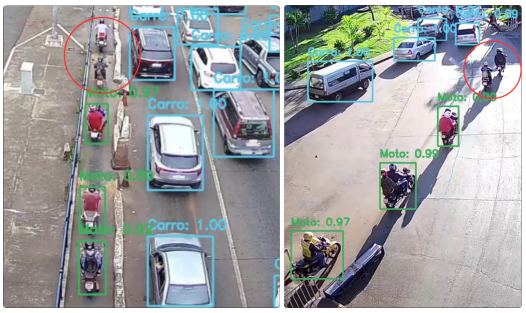

Fig. 6. Exemplo de limitações na detecção e classificação de motos.

Além disso, outras limitações observadas incluem:

- **Iluminação adversa**: condições de baixa luminosidade, sombras ou reflexo solar podem impactar a qualidade da imagem e, consequentemente, a capacidade de detecção do modelo.
- **Dependência de câmeras fixas**: o sistema foi projetado para operar com câmeras fixas, qualquer movimentação da câmera pode demandar reconfiguração na linha utilizada para contagem.
- **Arquitetura leve**: embora vantajosa para aplicações embarcadas ou em tempo real, a escolha por uma arquitetura

compacta limita a capacidade de análise aprofundada das imagens.

V. Conclusão

A metodologia deste trabalho consistiu na integração de uma solução de visão computacional com uma plataforma web capaz de operar em tempo real, permitindo o monitoramento contínuo do tráfego fronteiriço. Essa abordagem oferece suporte direto à tomada de decisão pela RFB, destacando-se pela simplicidade, eficiência e aplicação prática em ambiente de alta densidade e variabilidade operacional.

Com base nessa abordagem, os resultados obtidos validaram a efetividade da solução: F1-Score para carro: 97,1%; para motos: 96,1%; para caminhões: 93,5%; para pedestres: 95,6%; para ônibus: 81,9%; e para carros fiscalizados: 90,1%. Esses resultados demonstram a confiabilidade do sistema para a contabilização automatizada, mesmo em um cenário desafiador como a fronteira Brasil–Paraguai, em Foz do Iguaçu – Paraná. Além disso, a solução representa um avanço nos processos de monitoramento e fiscalização, fornecendo à RFB dados operacionais em tempo real, fortalecendo a capacidade institucional de análise, planejamento e tomada de decisão orientada por evidências.

Em relação a trabalhos futuros, considera-se a expansão da solução para outras unidades aduaneiras, ampliando sua cobertura geográfica e impacto institucional. Planeja-se também a adoção de arquiteturas de detecção mais recentes e a integração de técnicas de rastreamento para aumentar a robustez da contagem. No âmbito da plataforma web, vislumbra-se a utilização de bancos de dados temporais, visando maior velocidade e flexibilidade no armazenamento e consulta de dados, além da adoção de protocolos de mensageria escaláveis, que suportem maiores cargas e garantam confiabilidade na comunicação entre os componentes do sistema. Deve-se explorar ainda a comparação entre diferentes arquiteturas de detecção, o uso de dados sintéticos para balanceamento de classes minoritárias e estratégias de inferência em dispositivos de borda (*edge computing*).

Por fim, este trabalho apresenta contribuições técnicas e científicas relevantes ao demonstrar a viabilidade de uma solução leve, eficaz e aplicada em um contexto operacional fronteiriço. A validação em ambiente prático evidencia não apenas a maturidade da tecnologia empregada, mas também seu potencial de impacto social, ao fortalecer as ações de controle, fiscalização e segurança em regiões estratégicas. Além disso, os resultados obtidos e as diretrizes propostas podem servir como base para futuras iniciativas de pesquisa e desenvolvimento, promovendo a inovação tecnológica no setor público e incentivando a adoção de soluções que utilizam IA.

Agradecimentos

Agradecemos à Receita Federal do Brasil pelo acesso às imagens das câmeras, fundamentais para o desenvolvimento da solução e para a elaboração deste trabalho. Estendemos nosso agradecimento, em especial, aos servidores Caio Emannuel Lira de Santana e Guilherme de Andrade Palmieri, pelo apoio técnico e institucional prestado durante o projeto.

Referências

[1] G. Queiróz and A. Hachisuca. "Desenvolvimento de Aplicativo para Cadastro e Divulgação de Veículos Suspeitos," in Anais do XIX Congresso Latino-Americano de Software Livre e Tecnologias Abertas, Evento Híbrido, 2022, pp. 141-144, doi: https://doi.org/10.5753/latinoware.2022.228042.

[2] "Muralha inteligente: Itaipu vai investir cerca de R$ 19 milhões em tecnologia para segurança da fronteira," https://www.itaipu.gov.br/noticias/tecnologia/muralha-inteligente-itaipu-vai-investir-cerca-de-r-19-milhoes-em-tecnologia-para-seguranca-da-fronteira (acessado em: 8 de maio de 2025).

[3] "OpenCV," https://opencv.org/ (acessado em: 8 de maio de 2025)

[4] L. Maus, M. Gribler, and R. M. Benetti, "Contagem e Classificação de Veículos por Visão Computacional," in Anais do VIII Workshop de Computação Aplicada, São Miguel do Oeste, 2022, pp. 49–56, doi: https://doi.org/10.14210/cotb.v12.p049-056.

[5] R. M. Benetti and F. C. Dávalos, "Desenvolvimento de um sistema automatizado para contagem de veículos aplicado em uma rotatória," in Anais da XIX Mostra de Iniciação Científica e Tecnológica da UTFPR, Medianeira, 2021.

[6] A. Cordeiro, P. P. Filho, C. Ojeda, and G. Valiati. "Rastreamento e contagem de pedestre em tempo real por meio de imagens digitais," in Anais do XVI Congresso Latino-Americano de Software Livre e Tecnologias Abertas, Foz do Iguaçu, 2019, pp. 146-149, doi: https://doi.org/10.5753/latinoware.2019.10350.

[7] L. Neitzke, F. Sano, and L. Tampelini. "Visão computacional aplicada a quantificação do fluxo de veículos," in Anais do XX Congresso Latino-Americano de Software Livre e Tecnologias Abertas, Foz do Iguaçu/PR, 2023, pp. 158-161, doi: https://doi.org/10.5753/latinoware.2023.236524.

[8] M. Santos, C. Mauricio, V. Santos, and F. Peres. "Rede Neural Convolucional para Detecção de Pedestres Realizando Travessias de Risco," in Anais Estendidos da XXXV Conference on Graphics, Patterns and Images, Natal/RN, 2022, pp. 129-133, doi: https://doi.org/10.5753/sibgrapi.est.2022.23276.

[9] B. C. de Almeida, D. M. da Silva Junior, and R. M. Haddad, "Viabilidade da visão computacional com OpenCV para a coleta assertiva e automática do fluxo de pessoas nos corredores do Viashopping," Pontifícia Universidade Católica de Minas Gerais, Belo Horizonte, 2022.

[10] M. -H. Yen, B. -S. Lin, Y. -L. Kuo, I. -J. Lee and B. -S. Lin, "Adaptive Indoor People-Counting System Based on Edge AI Computing," in IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 8, no. 1, pp. 255-263, Feb. 2024, doi: 10.1109/TETCI.2023.3300172.

[11] R. A. Light, "Mosquitto: server and client implementation of the MQTT protocol," The Journal of Open Source Software, vol. 2, no. 13, May 2017, DOI: 10.21105/joss.00265

[12] "Leading Data Annotation Platform," https://www.cvat.ai/ (acessado em: 8 de maio de 2025)

[13] "Yolo v4, v3 and v2 for Windows and Linux," https://github.com/AlexeyAB/darknet (acessado em: 21 de maio de 2025)

[14] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," arXiv preprint arXiv:2004.10934, 2020.

[15] "CUDA Zone," https://developer.nvidia.com/cuda-zone/ (acessado em: 8 de maio de 2025)

[16] "NVIDIA," https://www.nvidia.com/ (acessado em: 8 de maio de 2025)

[17] "Paho-MQTT," https://pypi.org/project/paho-mqtt/ (acessado em: 8 de maio de 2025)

[18] "O que é o MySQL?," https://cloud.google.com/mysql?hl=pt-BR (acessado em: 9 de maio de 2025)

[19] "Executar a JavaScript em Toda Parte," https://nodejs.org/ (acessado em: 9 de maio de 2025)

[20] "Introdução Express/Node," https://developer.mozilla.org/pt-BR/docs/Learn_web_development/Extensions/Server-side/Express_Nodejs/Introduction (acessado em: 9 de maio de 2025)

[21] "React. The library for web and native user interfaces," https://react.dev/ (acessado em: 9 de maio de 2025)

[22] "ApexCharts: Modern & Interactive Open-source Charts," https://apexcharts.com/ (acessado em 25 de maio de 2025)