ANTT – Agência Nacional de Transportes Terrestres

RDT – Recurso de Desenvolvimento Tecnológico

RELATÓRIO FINAL

PROJETO 11 - DESENVOLVIMENTO DE TECNOLOGIA E ANÁLISE DE VIABILIDADE DO USO DE VISÃO COMPUTACIONAL PARA IDENTIFICAÇÃO DE INCIDENTES E NECESSIDADES DE ATENDIMENTO OPERACIONAL NA BR-116/PR NO TRECHO SOB CONCESSÃO DA AUTOPISTA LITORAL SUL

Autopista Litoral Sul

06/02/2024

Sumário

1. DESCRIÇÃO DO PROJETO

1.1. Título

Desenvolvimento de tecnologia e análise de viabilidade do uso de visão computacional para identificação de incidentes e necessidades de atendimento operacional na BR-116/PR no trecho sob concessão da Autopista Litoral Sul.

1.2. Resumo

Este relatório descreve o planejamento, a solução desenhada, as atividades desempenhadas, os resultados e outras informações relevantes sobre a execução do projeto do uso de visão computacional e análise avançada para identificação de veículos parados em acostamento, no qual teve como seu principal objetivo a adaptação das tecnologias de inteligência artificial em imagens de câmeras CFTV de monitoramento rodoviário da Autopista Litoral Sul, a fim de analisar a viabilidade técnica dos seus usos para registros e envios de ocorrências, seja por falhas técnicas (pane mecânicas) ou outras irregularidades nos trechos da rodovia.

As equipes da Arteris e Programmer's trabalharam em conjunto entre os meses de julho de 2023 e dezembro de 2023, na execução da solução proposta com imagens de câmeras CFTV instaladas ao longo das rodovias concedidas pela Litoral Sul. As imagens foram compartilhadas e usadas para o treinamento de modelos de visão computacional, identificação e categorização de veículos, e subsequente aplicação de analítica avançada para a checagem de veículos que possam estar presentes na mesma posição em um intervalo de tempo pré-determinado. Os resultados dessas análises estão disponibilizados no final do documento juntamente com comentários sobre as simulações realizadas.

De modo geral, os resultados obtidos pela aplicação final foram muito satisfatórios, validando que o modelo desenvolvido teve boa capacidade de detecção, tempo de aferição e distinção entre classes, e persistência de ID dos veículos de forma favorável ao rastreamento.

1.3. Palavras-chave

Visão Computacional, Inteligência Artificial, Treinamento do modelo.

1.4. Justificativa

O setor de rodovias impulsionado pelas exigências e normativas da ANTT e buscando ganhos em segurança viária, qualidade dos serviços e eficiência operacional, tem feito uso cada vez maior de novas tecnologias. Não obstante, observa-se a utilização ainda modesta

de tecnologias digitais em processos mais tradicionais.

O projeto visa introduzir tecnologias digitais de “gerenciamento de tratativa de imagens” com uso de Inteligência Artificial e “Deep Learning” em processo rodoviários, neste primeiro momento mais especificamente no tocante a identificação de veículos parados. O projeto não é apenas relevante para a engenharia rodoviária nacional, mas essencial para permitir maior agilidade, flexibilidade e controle para a ANTT e concessionárias para prover um serviço de melhor qualidade aos usuários.

A Autopista Litoral Sul é responsável pela administração do trecho conhecido como Corredor do Mercosul, que compreende o Contorno Leste de Curitiba (BR-116), a BR-376 e a BR-101 e o Contorno de Florianópolis, fazendo a ligação da capital paranaense ao município de Palhoça, no estado de Santa Catarina. O trecho engloba 23 municípios em sua malha viária e tem 356,6 quilômetros de extensão. O contrato foi assinado em 14 de fevereiro de 2008 com vigência de 25 anos.

Considerando que o uso desta nova tecnologia visa interpretar imagens geradas por câmeras, sem nenhum requerimento óptico específico, permite que os pontos de monitoramento sejam mais facilmente implementados e mantidos, podendo ter sua implementação em larga escala. Futuramente, a função “pontos de monitoramento” pode ser implementada em outros locais das rodovias em onde já existam câmeras, melhorando a análise e potencializando investimentos já realizados.

Considerando integrações e futuras expansões, a solução alavancará maior segurança e benefício para os usuários através da assertividade nos atendimentos, podendo os recursos da concessionária serem direcionados exclusivamente para a resolução de problemas, reduzindo assim a necessidade de “busca” por ocorrências na via. Desta forma, esta aplicação ainda contribui significativamente para práticas ESG no quesito de redução de emissão de poluentes, em virtude da diminuição de deslocamentos desnecessários percorridos pelas viaturas das empresas concessionárias.

1.5. Objetivos

1.5.1. Objetivo Geral

Analisar a viabilidade técnica e homologar com a Agência Nacional de Transportes Terrestres (ANTT) a utilização de solução baseada em tecnologias avançadas de Inteligência Artificial, Machine Learning, Deep Learning e Visão Computacional, através do conceito de MVP (Mínimo Produto Viável) do uso de novas técnicas na operação de monitoramento de tráfego rodoviário, identificação de incidentes e necessidade de prestação de atendimento aos usuários da rodovia.

1.5.2. Objetivos Específicos

Esta pesquisa tem como objetivo:

1) Criar uma arquitetura tecnológica para utilização de modelos de inteligência artificial, tendo como finalidade a melhora do monitoramento das imagens das câmeras instaladas nas rodovias. A arquitetura deve prever possíveis inclusões de novas inteligências no futuro, para solucionar diferentes casos de uso, que serão observados durante a implementação desta ao longo do projeto.
2) Treinar e validar um modelo para identificação automática de veículos parados na rodovia ou em acostamento, que necessitem de atendimento por parte das equipes operacionais, e identificação de ocorrências que geram paralisação de tráfego ou congestionamento, de forma a ter maior assertividade e rapidez na prestação de serviço ao usuário da via, e contribuir para a segurança viária.
3) Elaboração de relatório final, com análise de eficácia do modelo gerado, recomendações de eventuais melhorias e possíveis pontos críticos para a implementação em larga escala.

1.6. Organização do trabalho

O projeto foi realizado em 12 etapas, sendo elas:

1) Criação de estrutura base de acordo com a arquitetura proposta

Nesta etapa desenhou-se a arquitetura de solução para o desenvolvimento do projeto, além da preparação de todo ambiente tecnológico para estruturação base do modelo de visão computacional.

2) Criação de base de imagens inicial para o treinamento da inteligência 'Base Line'

Foram selecionados e disponibilizados em nuvem vídeos capturados por mais de uma câmera de CFTV, com imagens de ângulos e horários diferentes (manhã, tarde e noite) do tráfego das rodovias da Litoral Sul, além de imagens genéricas do tráfego de outras concessões do grupo Arteris. Estas imagens serviram de base para o treinamento da inteligência 'Base Line'.

3) Treinamento da inteligência 'Base Line'

Nesta fase de treinamento da inteligência, primeiro definiu-se qual arquitetura de modelo seria adotada para atendimento dos objetivos do projeto e qual ambiente seriam realizados os treinamentos. Definido estes dois pontos, os treinamentos foram realizados utilizando a base de imagens criada na etapa anterior.

4) Testes da inteligência 'Base Line'

Após a etapa de treinamento, foi possível testar a inteligência aplicada em outros vídeos diferentes dos selecionados para treinamento do modelo. Este teste foi capaz de identificar previamente e capacidade de detecção e seu comportamento diante das imagens disponíveis.

5) Análise estatística dos resultados da inteligência 'Base Line'

Nesta etapa, através dos testes realizados, analisamos estatisticamente os resultados do modelo testado. Obtivemos algumas métricas como:

- Quantidade de veículos que pararam e foram detectados corretamente;
- Quantidade de veículos que pararam e não foram detectados corretamente;
- Quantidade de veículos que não pararam e foram detectados incorretamente;
- Número total de veículos detectados;
- Número total de veículos não detectados;
- Análise dos cenários que tiveram maior número de falhas;
- Análise dos tipos de veículos que tiveram maior número de falhas.

6) Aprimoramento da inteligência com nova base de imagens

Após essa primeira rodada de treinamento e teste, e dado os desafios observados, um novo conjunto de vídeos/imagens foram selecionados para tentar mitigar os problemas encontrados.

7) Treinamento da inteligência 'Aprimorada'

Nessa Etapa foi feito o treinamento da inteligência com o novo conjunto de imagens gerado para compor e aprimorar o cenário proposto.

8) Testes da inteligência 'Aprimorada'

Com o resultado não surtindo mudanças significativas no modelo, foram selecionadas e testadas alternativas de arquitetura de modelos compactos, desta vez modelos já pré-treinados, a fim de avaliar o desempenho e o que melhor se adequasse a finalidade do projeto.

9) Correção e otimização para a criação de uma inteligência 'Final'

Nesta fase foram feitas correções e otimizações utilizando técnicas de Data Augmentation para geração de mais imagens para a base de treinamento, assim como trabalhou-se em uma mitigação de casos em que as imagens se provaram difíceis ao entendimento dos modelos.

Dado os resultados do treinamento com as imagens otimizadas, definiu-se qual seria o modelo da inteligência final.

10) Treinamento da inteligência 'Final'

Com o modelo da inteligência final definido, o treinamento poderia ser realizado, aplicando a abordagem de maior sucesso.

11) Teste da Inteligência 'Final'

Assim que o treinamento do modelo foi finalizado, alguns testes foram realizados para medirmos o nível de assertividade, assim como o nível de recall na prática.

12) Homologação final e coleta de resultados

Nesta etapa foi feita a validação do modelo em tempo real, a fim de avaliar seu

comportamento e coletar os resultados com a inteligência rodando em uma imagem ao vivo de uma câmera posicionada na rodovia Arteris Litoral Sul.

1.7. Período de execução

O projeto teve duração de 6 meses, com a data de início em 17/07/2023 e data de conclusão em 17/01/2024.

1.8. Cronograma de execução

Etapa	Atividade	Período de execução - 24 semanas																							
		jul/23		ago/23				set/23				out/23				nov/23				dez/23				jan/24	
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	Criação de estrutura base de acordo com a arquitetura proposta	x	x	x	x																				
2	Criação de base de imagens inicial para o treinamento da inteligência ‘Base Line’			x	x																				
3	Treinamento da inteligência ‘Base Line’					x	x																		
4	Testes da inteligência ‘Base Line’							x	x	x	x														
5	Análise estatística dos resultados da inteligência ‘Base Line’									x	x			x	x	x	x	x	x	x	x				
6	Aprimoramento da inteligência com nova base de imagens											x	x												
7	Treinamento da inteligência ‘Aprimorada													x	x										
8	Testes da inteligência ‘Aprimorada’															x	x								
9	Correção e otimização para a criação de uma inteligência ‘Final’																	x	x						
10	Treinamento da inteligência ‘Final’																			x	x	x	x		
11	Teste da Inteligência ‘Final’																					x	x	x	x
12	Homologação final e coleta de resultados																					x	x	x	x

Tabela 1 – Cronograma físico das etapas do projeto

1.9. Local de execução

O projeto foi desenvolvido de forma remota pelas equipes responsáveis pela programação das tecnologias Programmers Informática LTDA, com supervisão e apoio da equipe da Arteris.

1.10. Equipe executora

Identificação da Equipe Executora da Programmer's Informática LTDA

Nome	Área de Atuação	Função	ID Empresa
Ariel Giacomini Bueno	Operacional	Scrum Master	Programmer's
Douglas Fernandes Sammur	Operacional	Cientista de Dados	Programmer's
Bianca Bertoldo	Operacional	Cientista de Dados	Programmer's
Rafael Cruz Fereira	Operacional	Desenvolvedor	Programmer's

Tabela 2 – Equipe Executora da Programmer's Informática LTDA

Identificação da equipe responsável pela obtenção de Vídeos Gravados de Monitoramento de Tráfego

Nome	Área de Atuação	Função	ID Empresa
Fernando César da Silva	Operacional	Coordenador de CCO	Autopista Litoral Sul
Eduardo Loewen	Operacional	Gerente de Gestão Operacional	Arteris

Tabela 3 – Equipe obtenção das imagens

Identificação da equipe de supervisão e apoio da Autopista Litoral Sul e Arteris S.A

Nome	Área de Atuação	Função	ID Empresa
Fernando César da Silva	Operacional	Coordenador de CCO	Autopista Litoral Sul
Eduardo Loewen	Operacional	Gerente de Gestão Operacional	Arteris
Marcio Lima	Tecnologia	Gerente de TI	Arteris

Tabela 4 – Equipe de supervisão Autopista Litoral Sul e Arteris

2. MÉTODOS E TÉCNICAS UTILIZADAS

Os principais métodos para o projeto serão:

- Visão Computacional e Rede Neural Convolucional: Para a computação, uma imagem é um conjunto de pixel, que são entendidas pelos algoritmos como uma informação conforme a disposição de cada pixel.

  Um algoritmo de visão computacional é treinado por meio de diferentes dimensões através da representação de cor no pixel. Após ser treinada, a máquina "aprende" a identificar uma imagem. São feitas várias camadas de processamento até que seja possível fazer a classificação da imagem (output category). Neste momento, a máquina é capaz de dizer o que está na imagem.

- Aprendizado de máquina e Aprendizagem profunda: Para a construção de algoritmos de inteligência artificial por meio de aprendizado de máquina, é preciso executa três grandes fases: preparação de dados, treinamento e teste.

  Em um projeto de visão computacional, os dados se consistem em imagens, então todas as imagens que serão utilizadas precisam passar por um processo de qualidade. Os algoritmos são treinados reconhecendo padrões que se repetem nos dados, para que isso seja possível, é necessário um grande volume de dados mapeados.

  Para criação de uma base para utilização em aprendizado de imagens, é necessário que se rotulem os objetivos dentro de uma imagem,

- Scrum: Todo desenvolvimento foi gerenciado por meio de Frameworks ágeis baseado em Scrum, com entregas parciais e incrementais a cada Sprint, com duração de 2 semanas cada. Cada Etapa de um projeto é equivalente a um Sprint.

As técnicas que envolveram os métodos foram:

- Câmeras de monitoramento do tráfego ao longo das rodovias da Autopista Litoral Sul, além de outras imagens genéricas do tráfego de outras concessões do grupo Arteris foram utilizadas para captura de imagens utilizadas para treinamento do modelo de visão computacional;

- Câmera IP (RTSP) de monitoramento do tráfego localizada na BR116, KM 81 – Sentido Norte, utilizada para testes ao vivo (real-time) do modelo de visão computacional;

- Instância do OneDrive para compartilhamento de arquivos entre Arteris e Programmer's, incluindo trechos de gravações de monitoramento do tráfego;
- Pré-processamento de vídeos em frames, seleção de imagens, comunicações com Azure Custom Vision via Portal, SDK e API;
- Azure Custom Vision utilizado para rotulagem das imagens, e treinamento de modelos compactos em arquitetura YOLOv2 Tiny;
- Azure Blob Storage na assinatura da Arteris para armazenamento dos recursos (vídeos, frames e rótulos), utilizado como ponte no envio e recebimento de dados pela Máquina Virtual durante os treinamentos dos modelos e testes da aplicação;
- Máquina Virtual no Microsoft Azure (NC8as_T4_v3) configurada com 8 vCPU, 1 GPU (T4 - 12GB), 56GB RAM e 352 GB SSD para armazenamento. A máquina conta com um ambiente Linux / Ubuntu 20.04 com um driver NVIDIA 535.104.12, SDK Deepstream 6.3, TAO Toolkit 5.0, Visual Studio Code e Anaconda;
- Técnicas de Data Augmentation para maior polaridade do conjunto de dados;
- Configuração de um Webhook no Microsoft Teams para recebimento dos alertas/notificações;

3. ETAPAS

3.1. Etapa 1 – Criação de estrutura base de acordo com a arquitetura proposta

3.1.1. Arquitetura do Projeto

A arquitetura de solução do projeto foi desenhada levando em consideração um cenário de desenvolvimento de um produto mínimo viável (MVP) cujo seu principal objetivo foi de realizar uma prova de viabilidade técnica do uso da visão computacional aplicada a um cenário real de monitoramento de tráfego para identificação de prováveis incidentes. Desta maneira, a arquitetura proposta foi simplificada contendo apenas elementos chave para a execução ágil do projeto:

- Assinatura do Microsoft Azure da própria Arteris, onde foi utilizado o serviço do Azure Custom Vision para rotulagem e treinamento de modelos compactos;
- Máquina virtual com GPU NVIDIA (T4) para realizar tanto a construção da aplicação quanto o treinamento de outros modelos;
- Blob Storage para hospedar os resultados obtidos.

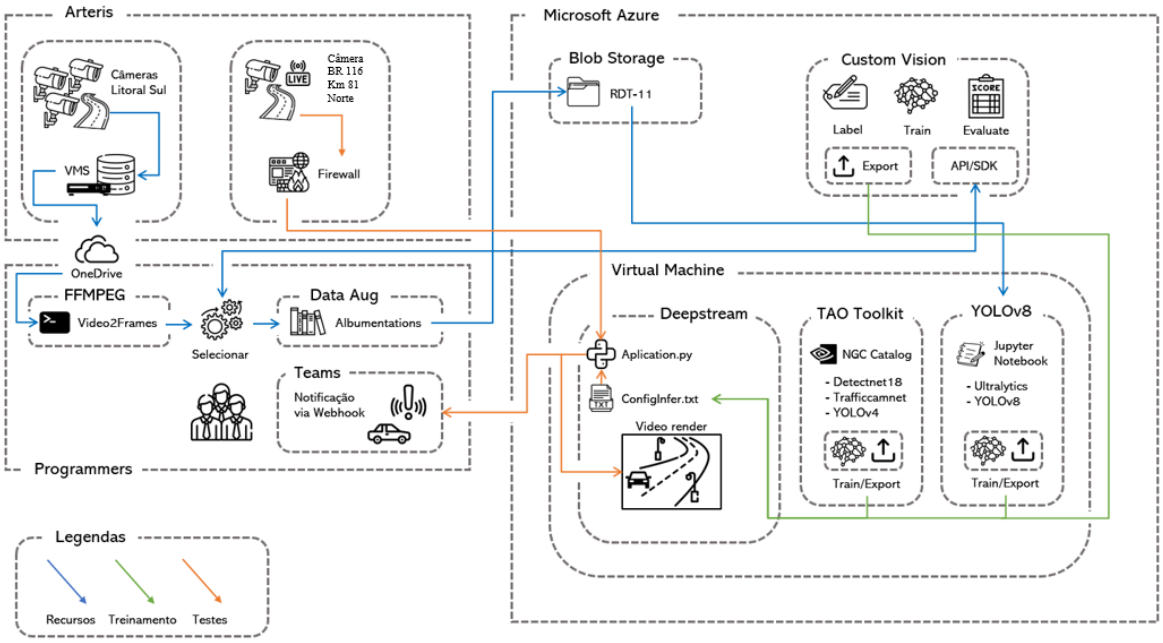


Figura 1 – Arquitetura do Projeto

3.1.2. Aplicação para processamento e análise dos dados de saída dos modelos

É de extrema importância para o desenvolvimento da inteligência a definição das ferramentas e a lógica por trás dos cálculos que receberão os dados de saída vindos dos modelos.

Como foco do projeto, é preciso uma aplicação forte o bastante para lidar com processamento de vídeos ao vivo, seja em questão de reproduzir o conteúdo recebido, e até mesmo na velocidade para conseguir fazer as imagens passarem pela inferência dos modelos detectores de objetos.

3.1.2.1. Aplicação SDK Deepstream NVIDIA

A aplicação do projeto foi desenvolvida utilizando algumas tecnologias de ponta da NVIDIA, em específico o SDK Deepstream, uma solução de software avançada projetada para otimizar o processamento de vídeo e análise de dados em tempo real. Este SDK é parte integrante da estratégia da NVIDIA para fornecer ferramentas de ponta para processamento de vídeo, inteligência artificial (machine learning) e meta análises.

O SDK Deepstream opera com base em um modelo de pipeline de processamento de dados flexível e eficiente. Ele permite que os desenvolvedores construam pipelines personalizados

para captura, processamento e análise de vídeo, aproveitando a potência das GPUs (unidade de processamento gráfico) da NVIDIA. Esses pipelines são compostos por componentes modulares, que incluem desde decodificadores de vídeo até algoritmos de inteligência artificial desenvolvidos para tarefas específicas como detecção e reconhecimento.

Destaca-se alguns dos pontos fortes para escolha desta aplicação:

1) **Alto Desempenho em Processamento de Vídeo:** Integrado com a tecnologia das GPUs NVIDIA, o SDK Deepstream oferece processamento de vídeo de alta velocidade quando utilizado seu conversor TensorRT, permitindo análise de múltiplos fluxos em tempo real.
2) **Integração Profunda com IA:** Suporta várias extensões de IA, o que facilita a implementação para uma maior variedade de aplicações.
3) **Flexibilidade e Modularidade:** A arquitetura modular do Deepstream, apoiada pela tecnologia NVIDIA, permite a personalização de pipelines de acordo com necessidades específicas.

3.1.2.2. Configurações técnicas da aplicação

Todos os passos da instalação e configuração do Deepstream em uma VM Azure Tesla T4 com sistema operacional Ubuntu 20.04 estão descritos abaixo:

1) Instalar Dependências:

Insira os seguintes comandos para instalar os pacotes necessários antes de instalar o Deepstream SDK:

```
$ sudo apt install \
libssl3 \
libssl-dev \
libgstreamer1.0-0 \
gstreamer1.0-tools \
gstreamer1.0-plugins-good \
gstreamer1.0-plugins-bad \
gstreamer1.0-plugins-ugly \
gstreamer1.0-libav \
libgstreamer-plugins-base1.0-dev \
libgstrtspserver-1.0-0 \
libjansson4 \
libyaml-cpp-dev \
libjsoncpp-dev \
protobuf-compiler \
gcc \
make \
git \
python3
```

2) Instalar CUDA Toolkit 12.2:

Executar os seguintes comandos:

```
$ sudo apt-key adv --fetch-keys https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2204/x86_64/3bf863cc.pub
$ sudo add-apt-repository "deb https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2204/x86_64/ /"
$ sudo apt-get update
$ sudo apt-get install cuda-toolkit-12-2
```

3) Instalar NVIDIA driver 535.104.12:

Faça o download do driver na página oficial e execute os seguintes comandos:

```
$chmod 755 NVIDIA-Linux-x86_64-535.104.12.run
$sudo ./NVIDIA-Linux-x86_64-535.104.12.run --no-cc-version-check
```

4) Instalar TensorRT 8.6.1.6:

Execute os seguintes comandos no terminal:

```
sudo apt-get install libnvinfer8=8.6.1.6-1+cuda12.0 libnvinfer-plugin8=8.6.1.6-1+cuda12.0 libnvparsers8=8.6.1.6-1+cuda12.0 \
libnvonnxparsers8=8.6.1.6-1+cuda12.0 libnvinfer-bin=8.6.1.6-1+cuda12.0 libnvinfer-dev=8.6.1.6-1+cuda12.0 \
libnvinfer-plugin-dev=8.6.1.6-1+cuda12.0 libnvparsers-dev=8.6.1.6-1+cuda12.0 libnvonnxparsers-dev=8.6.1.6-1+cuda12.0 \
libnvinfer-samples=8.6.1.6-1+cuda12.0 libcudnn8=8.9.4.25-1+cuda12.2 libcudnn8-dev=8.9.4.25-1+cuda12.2
```

5) Instalar Deepstream SDK:

Baixe o pacote Deepstream 6.3 dGPU Debian e digite o comando:

```
sudo apt-get install ./deepstream-6.3_6.3.0-1_amd64.deb
```

3.1.2.3. Pipeline da aplicação

Para a utilização da tecnologia, precisou-se criar uma pipeline que a consumisse. Com essa pipeline, vários plugins podem ser adicionados para realizar diversas operações, seja recebimento do fluxo de vídeo, envio de imagem para componentes de inferências do modelo, funções capazes de analisar os insights por trás de cada imagem, entre outros.

Após a criação da estrutura padrão do pipeline para recebimento de fluxos, foram criadas as funções customizadas capazes de perceber se um objeto (veículo) está parado, e notificá-lo se for o caso. A lógica por trás da função capaz de detectar se um objeto está parado ou não, é feita com base nos cálculos de IOU (Intersection over Union) que um objeto apresenta ao longo de X minutos, sendo X o valor definido pelo usuário. A função guarda as coordenadas do objeto detectado no frame inicial, e após passar os minutos definidos em X, ele compara os objetos com base nos Ids definidos pelo componente Tracker do Deepstream.

O Tracker é um componente cuja função é atribuir “Ids” a cada objeto detectado ao longo dos frames, uma maneira de tratarmos cada objeto individualmente. Para que o Tracker tenha

um bom desempenho, algumas métricas do nosso modelo devem estar em um nível mais elevado, especificamente o Recall. Uma vez que o recall do modelo seja alto o suficiente para manter os Ids dos objetos, conseguimos realizar um cálculo mais preciso e variado de informações sobre os objetos presentes em cena, como por exemplo o tempo de tela em que o objeto ficou parado.

Para efeito explicativo, digamos que a inteligência detecte um veículo parado no acostamento em primeira instancia, guardando o Id do veículo e suas coordenadas. 5 minutos após a detecção, ele guarda os valores das detecções presentes em tela e checa se o Id do veículo ainda está presente em cena. Se caso ainda estiver, realiza o cálculo de IOU entre os valores anteriores e o valor atual das coordenadas do veículo. O cálculo de IOU por sua vez retorna um valor entre 0 e 1, indicando o nível de proximidade que as coordenadas estão uma da outra. Quanto mais próximo de 1 maiores são as proximidades entre os objetos em questão. Limites de proximidade podem ser definidos para saber se um veículo está parado ou não, de preferência trabalharemos com +0.5 (50%) de proximidade.

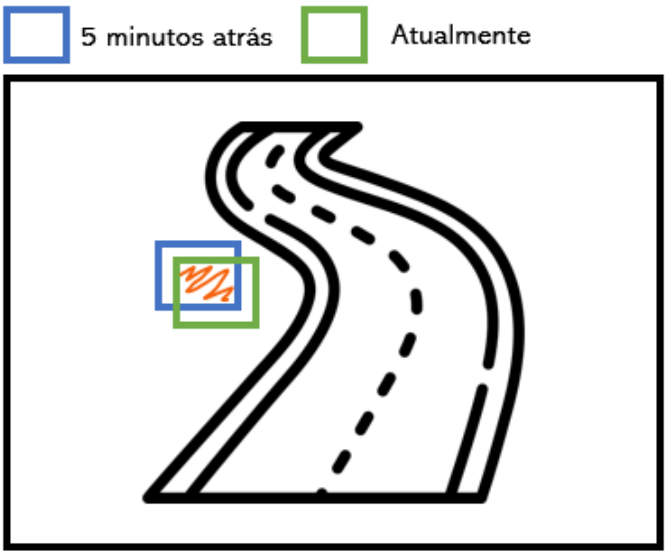


Figura 2 – Exemplo de funcionamento do algoritmo

A marcação em laranja corresponderia a área do IOU calculado, indicando a porcentagem (%) de proximidade que as coordenadas estão uma da outra. Se algum veículo tiver o IOU maior que 50% (0.5), um alerta seria enviado contendo seu Id, suas coordenadas, e o tempo que ele ficou parado em questão.

3.2. Etapa 2 – Criação de base de imagens inicial para o treinamento e teste da inteligência ‘Base Line’

Definida toda a estrutura base da inteligência, uma base de imagens inicial para o treinamento e teste do modelo foi criada.

3.2.1. Seleção das imagens

Os vídeos disponibilizados no serviço de nuvem da Microsoft OneDrive entre Arteris e Programmer’s foram divididos em dois tipos:

1) Vídeos para o treinamento do modelo com o fluxo de tráfego normal com duração de 1 hora:

Figura 3 – Imagens da câmera localizada na BR101 KM 130+070 – Sentido Sul selecionadas para treinamento

Figura 5 – Imagens da câmera localizada na BR101 KM 208+300 – Sentido Sul selecionadas para treinamento

Figura 6 – Imagens da câmera localizada na BR101 KM 208+300 – Sentido Sul selecionadas para treinamento

Além de imagens da Litoral Sul, outras imagens genéricas de demais concessões do grupo Arteris também foram selecionadas para treinamento do modelo.

2) Vídeo para teste do algoritmo com o fluxo de tráfego normal, com a presença de veículo parado em acostamento.

Figura 7 – Imagens da câmera localizada na BR116, KM 81+000 – Sentido Norte com a simulação realizada para testes.

Na ocasião, um veículo de guincho leve utilizado na operação da concessionária Arteris Litoral Sul foi utilizado para simular uma parada no acostamento da via.

3.2.2. Tratamento das imagens

Para realizar os cortes das imagens nos vídeos de treinamento utilizou-se a ferramenta “FFMPEG”, aplicativo que permite fazer diversas tarefas de codificação e decodificação de arquivos multimídias, mantendo a taxa de 1 segundo para 5 frames convertidos. Ao todo foram utilizados 3 vídeos de duração de 1 hora com ângulos e presets diferentes, totalizando no final 1.124 imagens para compor a base inicial de treinamento.

3.3. Etapa 3 – Treinamento da inteligência 'Base Line'

3.3.1. Ambiente e ferramenta de treinamento

Visando o treinamento de um modelo "base", o Azure Custom Vision é uma ferramenta capaz de conseguirmos integrar os recursos de armazenamento, rotulagens e treinamento. O Azure Custom Vision, parte dos serviços cognitivos da Microsoft Azure, é uma ferramenta poderosa e acessível de aprendizado de máquina que permite ao usuário treinar, validar e implementar modelos de visão computacional personalizados. Ele é projetado para ser simples, mas robusto, oferecendo uma interface de usuário intuitiva junto com opções de personalização detalhadas para usuários mais avançados.

O processo geralmente envolve os seguintes passos:

Carregamento de Dados: Os usuários começam carregando imagens para treinar o modelo. Essas imagens podem representar diversas categorias ou objetos que o modelo precisa identificar. Quanto ao meio para enviar as imagens, você pode fazer isso diretamente pelo portal, por SDK, ou pela API dos Serviços Cognitivos.

Rotulagem das imagens: Uma vez que as imagens já estejam presentes no seu projeto, você precisa atribuir rótulos a elas, independente se forem para classificadores ou detectores de objetos. Você pode optar por utilizar um rotulador inteligente para agilizar essa parte do processo também.

Treinamento do Modelo: Com as imagens rotuladas, o Custom Vision utiliza algoritmos de aprendizado de máquina para treinar um modelo para reconhecer padrões e características nas imagens.

Validação e Ajuste: Após o treinamento, o modelo é testado com um novo conjunto de imagens para avaliar sua precisão. Os usuários podem então ajustar o modelo, refinando sua capacidade de reconhecimento.

E caso o domínio do seu modelo seja **compacto**, você ainda pode baixar o arquivo do modelo em diferentes formatos.

3.3.2. Arquitetura do modelo

Levando em consideração que o foco do projeto é realizar detecções em Real-time (ao vivo), entende-se que os modelos compactos são as melhores opções para o projeto. Estes modelos possuem as camadas reduzidas, ou seja, o tempo de inferência/resposta é mais rápido que

outras arquiteturas mais complexas. Também é possível abordar as questões de custos e diversidade, além de precisarem de menos recursos computacionais para funcionar, o que são capazes de se adequar diretamente a quase todo dispositivo de ponta.

Os modelos treinados sob domínio compacto no ambiente do Azure Custom Vision, são gerados a partir de uma arquitetura YOLOv2 Tiny, com poucas variações em seus cálculos pós-camada de saída.

3.3.3. Rotulação das imagens

Como mencionado anteriormente, um dos passos que compõe o treinamento do modelo é a rotulagem das imagens, independente se forem para classificadores ou detectores de objetos.

Utilizou-se o ambiente do Azure Custom Vision para rotulagem das imagens, definidas da seguinte forma: “car” para os veículos leves, “truck” para veículos pesados e “motorbike” para motocicletas. Ao final do processo, obtivemos um total de 4.884 rótulos, divididos entre:

Rótulo	Quantidade
Car	3.373
Truck	1.169
Motorbike	342
Total	**4.884**

Tabela 5 – Rotulagem das imagens

3.4. Etapa 4 – Testes da inteligência ‘Base Line’

3.4.1. Testes da inteligência

Através dos testes, o modelo demonstrou-se capaz de diferenciar as classes dos objetos, pelo alto valor de 95,9% alcançado na métrica “Precisão”. Quanto ao valor de “Recall”, habilidade de detectar todos os veículos em cena, encontramos uma oportunidade de melhoria, visto que seu resultado de 70,1% se demonstrou bem abaixo do alcançado na métrica de precisão. Abaixo os resultados do primeiro treinamento:

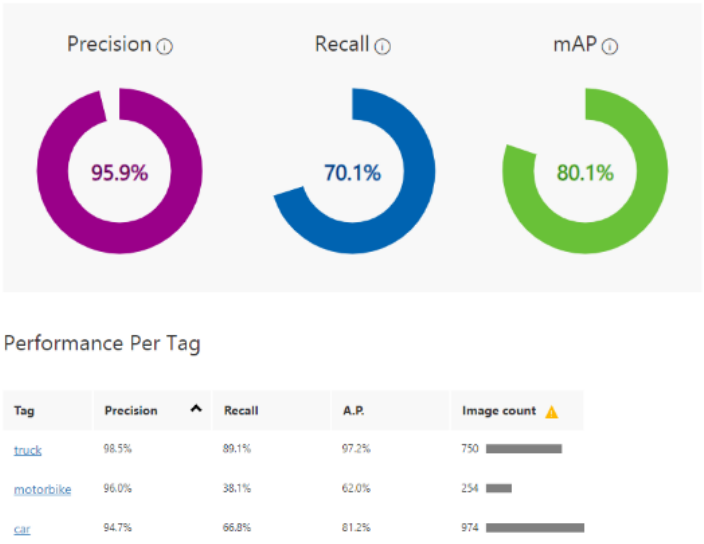


Figura 8 – Resultados do primeiro treinamento

Em consequência do resultado obtido na métrica de Recall, notou-se que o modelo apresentava bastante instabilidade para manter os Ids dos objetos detectados.

Figura 9 – Teste de recall

Como demonstrado acima, o veículo parado na primeira imagem carregava o Id de número 2, mas após poucos segundos ele alterou o Id para o número 19. Essa quebra de identidade comprometeria toda a aplicação para geração do alerta/notificação.

3.4.2. Envio de Notificação/Alerta

Uma vez em que o veículo estiver parado por X minutos, um alerta é enviado para o Microsoft Teams via Webhook. O formato do alerta é passado como um card, como o exemplo abaixo:

Figura 10 – Exemplo de alerta padrão

É informado a rodovia em questão, o Id do veículo, o tempo em que ele ficou parado, quando o alerta foi enviado, e por último as demarcações na tela indicando as posições dos veículos, sendo o em vermelho informado o veículo que está parado.

O alerta também foi customizado para diferenciar quando um ou mais veículos param na rodovia, posteriormente podendo-se tratar de diferentes incidentes.

Quanto as caixas de marcação desenhadas, todas podem ser formatadas, seja tonalidades de cores, formatos, ou tamanhos diferentes.

E por último, um alerta capaz de informar se algum veículo deixou de estar parado em cena, adicionado como recurso para informar os operadores que não há mais a necessidade de enviarem suporte para o trecho.

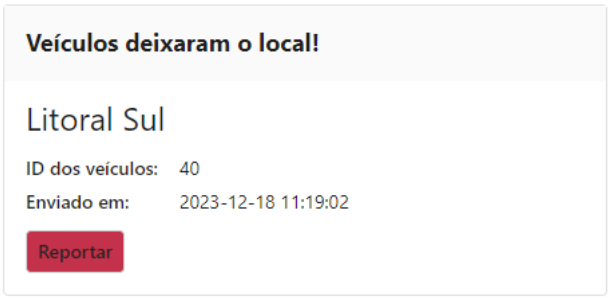


Figura 12 – Exemplo de alerta de saída

3.5. Etapa 5 – Análise estatística dos resultados da inteligência 'Base Line'

Com a mesma base de imagens, 6 vídeos das 3 câmeras com variação de horário entre manhã e à tarde foram selecionados para realizarmos uma contagem de objetos visual durante 5 minutos, e posteriormente comparamos com a capacidade de detecção do modelo. Obtivemos as seguintes métricas e observações:

CONTAGEM REAL

MANHÃ			
RÓTULO	ESQUERDA	DIREITA	TOTAL
CAR	31	50	81
MOTORBIKE	3	0	3
TRUCK	33	40	73
TOTAL	67	90	157

TARDE			
RÓTULO	ESQUERDA	DIREITA	TOTAL
CAR	60	38	98
MOTORBIKE	0	0	0
TRUCK	45	30	75
TOTAL	105	68	173

DETECTADOS MODELO

MANHÃ						
RÓTULO	ESQUERDA		DIREITA		TOTAL	
CAR	28	90%	43	86%	71	88%
MOTORBIKE	2	67%	0	-	2	67%
TRUCK	31	94%	39	98%	70	96%
TOTAL	61	91%	82	91%	143	91%

TARDE						
RÓTULO	ESQUERDA		DIREITA		TOTAL	
CAR	55	92%	31	82%	86	88%
MOTORBIKE	0	-	0	-	0	-
TRUCK	43	96%	29	97%	72	96%
TOTAL	98	93%	60	88%	158	91%

NÃO DETECTADOS MODELO

MANHÃ						
RÓTULO	ESQUERDA		DIREITA		TOTAL	
CAR	3	10%	7	14%	10	12%
MOTORBIKE	1	33%	0	-	1	33%
TRUCK	2	6%	1	3%	3	4%
TOTAL	6	9%	8	9%	14	9%

TARDE						
RÓTULO	ESQUERDA		DIREITA		TOTAL	
CAR	5	8%	7	18%	12	12%
MOTORBIKE	0	-	0	-	0	-
TRUCK	2	4%	1	3%	3	4%
TOTAL	7	7%	8	12%	15	9%

Figura 13 – Análise 1

CONTAGEM REAL

MANHÃ			
RÓTULO	ESQUERDA	DIREITA	TOTAL
CAR	67	41	108
MOTORBIKE	7	2	9
TRUCK	16	27	43
TOTAL	90	70	160

TARDE			
RÓTULO	ESQUERDA	DIREITA	TOTAL
CAR	68	51	119
MOTORBIKE	10	2	12
TRUCK	41	39	80
TOTAL	119	92	211

DETECTADOS MODELO

MANHÃ						
RÓTULO	ESQUERDA		DIREITA		TOTAL	
CAR	54	81%	36	88%	90	83%
MOTORBIKE	4	57%	0	0%	4	44%
TRUCK	14	88%	23	85%	37	86%
TOTAL	72	80%	59	84%	131	82%

TARDE						
RÓTULO	ESQUERDA		DIREITA		TOTAL	
CAR	62	91%	45	88%	107	90%
MOTORBIKE	6	60%	0	0%	6	50%
TRUCK	40	98%	36	92%	76	95%
TOTAL	108	91%	81	88%	189	90%

NÃO DETECTADOS MODELO

MANHÃ						
RÓTULO	ESQUERDA		DIREITA		TOTAL	
CAR	13	19%	5	12%	18	17%
MOTORBIKE	3	43%	2	100%	5	56%
TRUCK	2	13%	4	15%	6	14%
TOTAL	18	20%	11	16%	29	18%

TARDE						
RÓTULO	ESQUERDA		DIREITA		TOTAL	
CAR	6	9%	6	12%	12	10%
MOTORBIKE	4	40%	2	100%	6	50%
TRUCK	1	2%	3	8%	4	5%
TOTAL	11	9%	11	12%	22	10%

Figura 14 – Análise 2

CONTAGEM REAL

MANHÃ			
RÓTULO	ESQUERDA	DIREITA	TOTAL
CAR	187	148	335
MOTORBIKE	5	24	29
TRUCK	46	53	99
TOTAL	238	225	463

TARDE			
RÓTULO	ESQUERDA	DIREITA	TOTAL
CAR	173	218	391
MOTORBIKE	24	30	54
TRUCK	53	44	97
TOTAL	250	292	542

DETECTADOS MODELO

MANHÃ						
RÓTULO	ESQUERDA		DIREITA		TOTAL	
CAR	184	98%	138	93%	322	96%
MOTORBIKE	4	80%	16	67%	20	69%
TRUCK	45	98%	48	91%	93	94%
TOTAL	233	98%	202	90%	435	94%

TARDE						
RÓTULO	ESQUERDA		DIREITA		TOTAL	
CAR	171	99%	205	94%	376	96%
MOTORBIKE	19	79%	21	70%	40	74%
TRUCK	51	96%	39	89%	90	93%
TOTAL	241	96%	265	91%	506	93%

NÃO DETECTADOS MODELO

MANHÃ						
RÓTULO	ESQUERDA		DIREITA		TOTAL	
CAR	3	2%	10	7%	13	4%
MOTORBIKE	1	20%	8	33%	9	31%
TRUCK	1	2%	5	9%	6	6%
TOTAL	5	2%	23	10%	28	6%

TARDE						
RÓTULO	ESQUERDA		DIREITA		TOTAL	
CAR	2	1%	13	6%	15	4%
MOTORBIKE	5	21%	9	30%	14	26%
TRUCK	2	4%	5	11%	7	7%
TOTAL	9	4%	27	9%	36	7%

Figura 15 – Análise 3

Além dessas métricas, conseguimos observar alguns comportamentos tendenciosos, como:

1) Observou-se que o modelo apresentava dificuldades em identificar objetos separados quando estão muito próximos um do outro, o tratando como se fosse apenas um objeto em questão.
2) Veículos de médio porte (carros com carroceria grande ou outros veículos poucos convencionais), o modelo teve dificuldade em diferenciar qual classe manter entre "car" ou "truck" mostrando variações ao longo das detecções.
3) Dado a silhueta comum de uma moto com outros objetos (pessoas, bicicletas etc.) o modelo foi capaz de criar detecções errôneas fora da rodovia (tratando pedestres como motociclistas).

O que reforçou novamente o ponto de que embora o modelo tenha conseguido uma boa assertividade em identificar o objeto em algum ponto da trajetória, ele ainda não havia demonstrado eficácia em conseguir manter detecções subsequentes nos frames seguintes.

3.6. Etapa 6 – Aprimoramento da inteligência com nova base de imagens

3.6.1. Seleção da nova base de imagens

Após essa primeira rodada de treinamento, e dado os desafios observados, um novo conjunto de vídeos/imagens foi selecionado para tentar mitigar os problemas encontrados. Não apenas

isso, como também preparar melhor a inteligência para os cenários que serão utilizados nos testes. No total, foram selecionadas 1.192 novas imagens.

Figura 16 – Novas imagens selecionadas

3.6.2. Rotulagem das imagens

Assim como antes, utilizamos o ambiente do Azure Custom Vision para rotulagem das novas imagens, conseguindo um total de 6.865 novos rótulos, divididos em:

Rótulo	Quantidade
Car	3.422
Truck	3.278
Motorbike	165
Total	**6.865**

Tabela 6 – Rotulagem das novas imagens

3.7. Etapa 7 – Treinamento da inteligência “Aprimorada”

3.7.1. Treinamento do modelo com nova base de imagens

Após novas imagens para implementação do modelo, a inteligência não demonstrou mudanças de suas métricas anteriores. Sua precisão, recall, ou até mesmo o mAP obtiveram variações entre 1-2% no treinamento realizado. Resultados que nos fizeram questionar a

arquitetura “YOLOv2 Tiny” utilizada pelo Azure Custom Vision, uma vez que quantidade e variedade de imagens não melhoraram o desempenho do modelo.

3.7.2. Migração de ambiente para treinamento

Como alternativa para os problemas encontrados com arquitetura de modelo que vinha sendo utilizada, os dados foram convertidos e migrados para os ambientes NVIDIA, uma vez que possuem uma grande biblioteca com diversas outras arquiteturas para modelos compactos. O foco em si seria encontrar a melhor opção que se adequasse a nossa base de imagens.

3.7.2.1. TAO Toolkit

Desenvolvido pela NVIDIA, o TAO (Train, Adapt, and Optimize) Toolkit é uma ferramenta avançada para treinamento e otimização de modelos de IA. Este toolkit permite que desenvolvedores e cientistas de dados personalizem e otimizem modelos de IA pré-treinados, tornando o processo mais eficiente e acessível. A NVIDIA projetou o TAO Toolkit para ser flexível e fácil de usar, abrindo as portas para um treinamento de modelos de IA mais rápido e personalizado.
A ferramenta oferece uma abordagem modular e intuitiva para o treinamento e a adaptação de modelos de IA, com foco na otimização e eficiência:

1) **Modelos Pré-Treinados da NVIDIA:** Acesso a uma extensa biblioteca de modelos de IA de alta qualidade, desenvolvidos pela NVIDIA.
2) **Personalização e Adaptação:** Permite a personalização desses modelos, utilizando um conjunto mínimo de dados etiquetados para adaptá-los a necessidades específicas.
3) **Otimização de Desempenho:** Inclui ferramentas para otimizar modelos para execução em uma variedade de dispositivos e plataformas, aproveitando a tecnologia da NVIDIA.

O ponto forte em questão do TAO Toolkit é que sua ampla gama de modelos pré-treinados possui diferentes tipos de arquiteturas (redes neurais) para conseguirmos testar os mais diversos cenários e desafios que a aplicação final poderá ter**.**

3.7.3. Seleção de arquiteturas e modelos pré-treinados

Após configurar o ambiente NVIDIA dentro da máquina virtual, com todas as instalações Deepstream e TAO Toolkit prontas para uso, poderíamos realizar o download de alguns

modelos pré-treinados que a NVIDIA tem na sua biblioteca. Utilizamos o NVIDIA NGC, uma plataforma abrangente que oferece um repositório de software para GPU, facilitando o acesso a uma ampla gama de aplicações, frameworks de deep learning e SDKs. NGC disponibiliza containers pré-construídos, modelos de AI, ferramentas de software e SDKs que são otimizados para rodar em diversas plataformas NVIDIA.

Foram selecionadas três opções de arquitetura para realização dos testes: Detectnet_v2, YOLOv4 e TrafficCamNet.

3.8. Etapa 8 – Testes da inteligência “Aprimorada”

Nessa fase, foram realizados os testes das três arquiteturas selecionadas para avaliar o desempenho de cada uma, a fim de encontrar a melhor solução para o projeto.

3.8.1. DetectNet_v2

DetectNet_v2 é baseado em redes neurais convolucionais (*CNNs*) e é otimizado para trabalhar com eficácia em GPUs, porém a arquitetura utilizada por ele teve dificuldade em reconhecer objetos “pequenos” em comparação com tamanho das imagens de entrada (1280x720 pixels), fazendo com que as demarcações dos objetos (bouding boxes) não fossem perfeitamente ajustáveis. Impossibilitando sua utilização no projeto.

Figura 17 – Teste com DetectNet_v2

3.8.2. YOLOv4

Embora a arquitetura do YOLOv4 seja especial para lidar com detecções de objetos pequenos, ele apresentou certa dificuldade para reconhecer e manter os veículos com vista traseira. As detecções se penduraram por pouco mais tempo que YOLOv2 Tiny do Custom Vision, mas com essas dificuldades para identificar e manter veículos traseiros nos impossibilitou de seguir com o modelo.

Figura 18 – Teste com YOLOv4.

3.8.3. TrafficCamNet

O TrafficCamNet sendo um modelo especializado na detecção de veículos em vídeos de tráfego, apresentou bons resultados. Porém, em determinados vídeos/ângulos ele apresentou uma dificuldade semelhante ao *DetectNet_v2* para lidar com objetos menores. Por não possuir em sua base imagens focadas em tráficos de rodovias, lidar com objetos pequenos e tópicos assim, o problema para manter o recall dos objetos enquanto parados ainda predominou.

Figura 19 – Teste com TrafficCamNet.

3.9. Etapa 9 – Correção e otimização para a criação de uma inteligência 'Final'

Após os testes realizados com outras arquiteturas e modelos para a inteligência, e em virtude de alguns resultados inconsistentes, uma das alternativas para mitigação de casos em que imagens se provaram difíceis ao entendimento dos modelos foi a utilização de técnicas de Data Augmentation, cujo objetivo é a melhoria na quantidade, qualidade e diversidade das imagens, dessa forma, melhorando o desempenho dos modelos de aprendizado.

3.9.1. Técnicas de Data Augumentation

Utilizou-se a biblioteca de código aberto “Albumentations” por ser rápida para geração de imagens, além de ser amplamente conhecida e utilizada em projetos de visão computacional. Seu objetivo principal é ajudar na melhoria do desempenho dos modelos, transformando a diversidade e a qualidade dos dados de treinamento.

Estas transformações incluem rotações, distorções, cortes, ajustes de cor e muitas outras técnicas para melhoria das imagens. Ao aplicar estas modificações à base de imagens de treinamento, os modelos de aprendizado de máquina podem aprender a reconhecer padrões e características em uma variedade maior de cenários, aumentando sua robustez e precisão.

Levando em conta o projeto em questão, uma das dificuldades foi o caso de alguns veículos com tonalidades de cores mais claras (brancos por exemplo) apresentarem-se como pontos em dificuldade para detecção do modelo.

Para mitigar esses problemas, e até mesmo gerar mais dados para consumo do modelo, foram utilizadas técnicas de redimensionamento que simulam um efeito espelho (flip-horizontal) e técnicas para destacar o contraste de objetos por blocos pré-definidos nas imagens chamada CLAHE.

O flip horizontal foi usado para aumentar o conjunto de dados durante o treinamento de algoritmos de aprendizado de máquina, permitindo que o modelo aprenda a reconhecer objetos em diferentes orientações.

Figura 20 – Flip-Horizontal

Comumente utilizado em análises de imagens de raio X, o método CLAHE é particularmente eficaz para aplicar um contraste onde objetos de diferentes tonalidades - brancos, pretos e cinzas - são facilmente destacados. Isso se deve à sua capacidade de adaptar a equalização do histograma de forma localizada em diferentes partes da imagem.

Figura 21 – CLAHE

3.9.2. Rotulagem das imagens com Data augumentation

Anteriormente, a base tinha **2.316** imagens com um total de rótulos **11.749**, divididos em:

Rótulo	Quantidade
Car	6.795
Truck	4.447
Motorbike	507
Total	**11.749**

Tabela 7 – Rotulagem das imagens pré Data Augumentation

Posteriormente a aplicação do Data Augmentation, a base aumentou sua quantidade para **9.264** imagens com um total de rótulos **47.380**, divididos em:

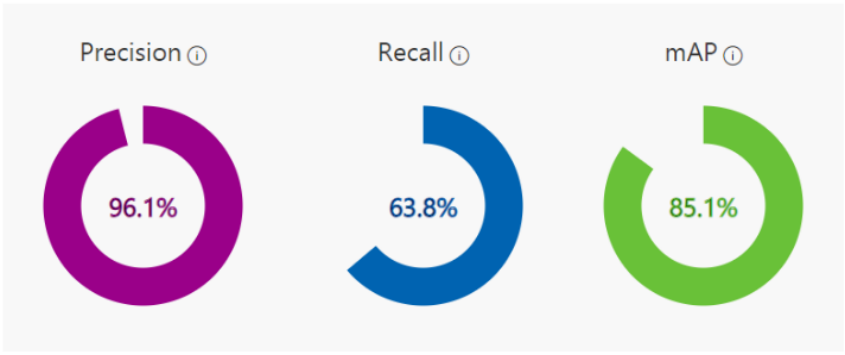
Rótulo	Quantidade
Car	27.600
Truck	17.856
Motorbike	1.924
Total	**47.380**

Tabela 8 – Rotulagem das imagens pós Data Augumentation

3.9.3. Teste do modelo com base de imagens otimizadas

A fim de avaliar a mitigação das dificuldades em comum, observadas nos modelos pré-treinados, com a aplicação das técnicas de Data Augumentation, realizou-se o teste do modelo no ambiente do Azure Custom Vision com a nova base de imagens.

A fim de solucionar um problema detectado de má distribuição de classes que o Data Augmentation não conseguiu resolver, os rótulos definidos em "*car*", "*truck*" e "*motorbike*" foram alterados para "*light_vehicle*" (carros e motos) e "*heavy_vehicle*" (caminhões).



Performance Per Tag

Tag	Precision	Recall	A.P.	Image count
light_vehicle	96.3%	55.3%	75.7%	8705
heavy_vehicle	95.9%	79.0%	94.5%	7818

Figura 22 – Teste do modelo com aplicação de técnicas de Data Augumentation

Após a realização do teste, o desempenho se mostrou inferior com relação ao primeiro teste realizado. Os resultados obtidos foram de 96,1% de Precisão, 63,8% de Recall e 85,1% de maP.

3.9.4. Considerações sobre a correção, otimização e definição do modelo da inteligência 'Final'

Analisando os resultados obtidos com os modelos treinados e testados, tanto no ambiente do Azure Custom Vision, quanto no TAO Toolkit, concluiu-se que os resultados não satisfatórios estavam relacionados a arquitetura da rede neural utilizada para treinamento do modelo, no caso os modelos compactos. Sendo assim, se fez necessário a utilização de outras arquiteturas de modelos mais complexos, que não possuam um tempo grande de inferência que possam comprometer os cálculos em Real-time.

3.9.5. Escolha do modelo da inteligência ‘Final’

Levando em consideração as características citadas acima, o modelo de arquitetura complexa escolhido para a sequência do desenvolvimento do projeto foi a do YOLOv8, justamente por ter uma boa relação entre “arquitetura complexa versus tempo de inferência “.

3.9.5.1. YOLOv8

O YOLOv8 é a mais recente versão do algoritmo "You Only Look Once" (YOLO), desenvolvido pela Ultralytics. Esse algoritmo representa um avanço significativo na detecção de objetos em tempo real, oferecendo melhorias notáveis em precisão e velocidade em relação às suas versões anteriores. YOLOv8 continua a tradição da série YOLO de fornecer uma solução eficiente e poderosa para a detecção de objetos em imagens e vídeos, tornando-o uma escolha popular entre desenvolvedores e pesquisadores em visão computacional.

Assim como seus predecessores, utiliza uma abordagem única de passagem (single-pass) para detecção de objetos, o que significa que ele analisa a imagem inteira durante a detecção, em contraste com abordagens que analisam partes da imagem separadamente. Isso permite ao YOLOv8 alcançar uma velocidade de processamento notavelmente rápida sem sacrificar a precisão. O algoritmo aplica redes neurais profundas para identificar e classificar objetos em várias categorias, fazendo-o eficiente em uma variedade de cenários de detecção de objetos. O YOLOv8 é projetado para oferecer uma detecção de objetos rápida e precisa, ideal para aplicações em tempo real, além de ser facilmente integrável com outras tecnologias e frameworks de IA.

3.10. Etapa 10 – Treinamento da inteligência ‘Final’

Para permitir que o modelo de aprendizagem produza previsões corretas, os dados devem ser inseridos nele e seus parâmetros ajustados. O modo de treinamento YOLOv8 do Ultralytics foi projetado para utilizar totalmente os recursos para treinar modelos de detecção de objetos de maneira eficaz e eficiente.

A base utilizada para treinamento é a mesma gerada com as otimizações de Data augumentation, composta de 9.264 imagens, onde 6.485 imagens foram utilizadas para treino, 1.853 para teste e 926 de validação.

Usando o conjunto de dados já rotulado, o modelo foi treinado no Jupyter Notebook, ferramenta para desenvolver softwares de código aberto, padrões abertos e serviços para computação interativa em dezenas de linguagens de programação, na máquina virtual após ser baixado suas dependências. O modelo passou por treinamento em 100 épocas, com taxa de aprendizado (hiper parâmetro que controla o quanto deve-se ajustar o modelo em resposta ao erro) de 0,01, um tamanho de lote de treinamento de 8 e tamanho de entrada das imagens de 600x600 pixels. Um bom ajuste de hiper parâmetros é crucial para garantir que o modelo tenha um bom desempenho quando aplicado a dados novos.

O treinamento envolveu a conferência no modelo da capacidade de distinguir objetos das 3 classes: carro, moto e caminhão. O tempo para treinamento necessário foi de 1,68 horas. Na camada de saída do YOLOv8, são obtidas caixas delimitadoras da posição do veículo e pontuações representando a probabilidade de que a caixa contenha um objeto de determinada classe.

O desempenho do modelo treinado é avaliado usando uma série de métricas, incluindo pontuação F1, precisão, recall e loss. Estas medidas oferecem um indicador do desempenho do modelo na distinção entre classes. Abaixo as principais delas:

1) Precisão: A precisão é uma métrica que quantifica o número de previsões positivas corretas feitas. É calculado como o número de verdadeiros positivos dividido pelo número total de verdadeiros positivos e falsos positivos. O resultado é um valor entre 0,0 para nenhuma precisão e 1,0 para precisão total ou perfeita.
2) Recall: O Recall é uma métrica que quantifica o número de previsões positivas corretas feitas de todas as previsões positivas que poderiam ter sido feitas. É calculado como o número de verdadeiros positivos dividido pelo número total de verdadeiros positivos e falsos negativos (por exemplo, é a taxa de verdadeiros positivos). O resultado é um valor entre 0,0 para nenhum recall e 1,0 para recall completo ou perfeito.
3) Perda: A perda é calculada em treinamento e validação e sua interpretação é o quão bem o modelo está se saindo para esses dois conjuntos. É um somatório dos erros cometidos para cada exemplo em conjuntos de treinamento ou validação. Quanto menor a perda, melhor um modelo.

O desempenho do YOLOv8 por 100 épocas é mostrado na figura abaixo com gráficos de perda, precisão e recall.

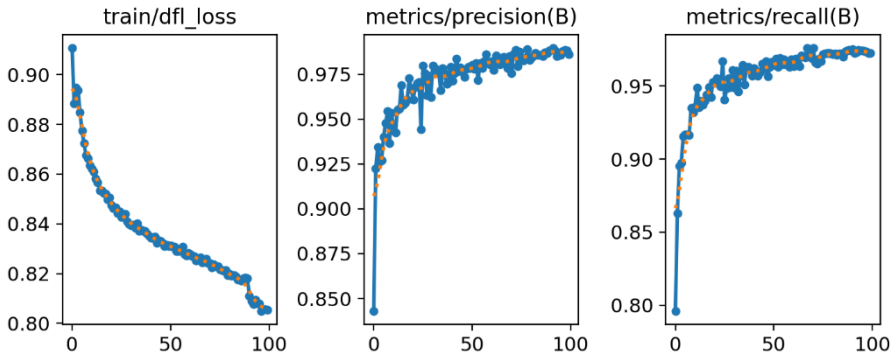


Figura 23 – Gráficos de perda, precisão e recall obtidos com o treinamento.

O gráfico acima mostra que à medida que o modelo é treinado para um número maior de épocas, as perdas do modelo diminuem. Também podemos verificar como a precisão e o recall aumentam com o passar das épocas, atingindo valores excelentes de ≈ 98% para precisão e ≈ 97% para recall.

Uma curva PR (Precisão-Recall) é um gráfico da precisão (eixo y) e do recall (eixo x) para diferentes limiares de probabilidade. Um modelo perfeito é representado como uma “curva” em coordenada de (1,1). Quanto melhor o modelo, mais sua curva PR será representada se curvando em direção a uma coordenada de (1,1), com maior área sob a curva.

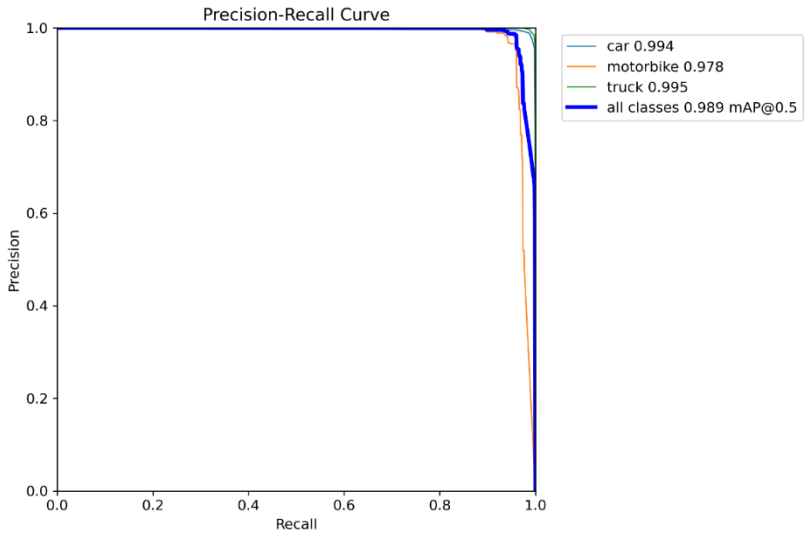


Figura 24 – Curva Precisão-Recall

Uma forma bastante simples de visualizar a performance de um modelo de classificação é através de uma matriz de confusão. Esta matriz indica quantos exemplos existem em cada grupo: falso positivo (FP), falso negativo (FN), verdadeiro positivo (TP) e verdadeiro negativo (TN). No eixo x da matriz estão as classes reais, e no eixo y as classes previstas pelo modelo. É interessante visualizar a contagem destes grupos tanto em números absolutos quanto em porcentagens da classe real, já que o número de exemplos em cada classe pode variar. O resultado esperado é que os valores estejam concentrados na diagonal principal.

Avaliando a matriz confusão gerada, é possível verificar que 99,49% dos carros, 95,77% das motos e 99,71% dos caminhões foram detectados corretamente.

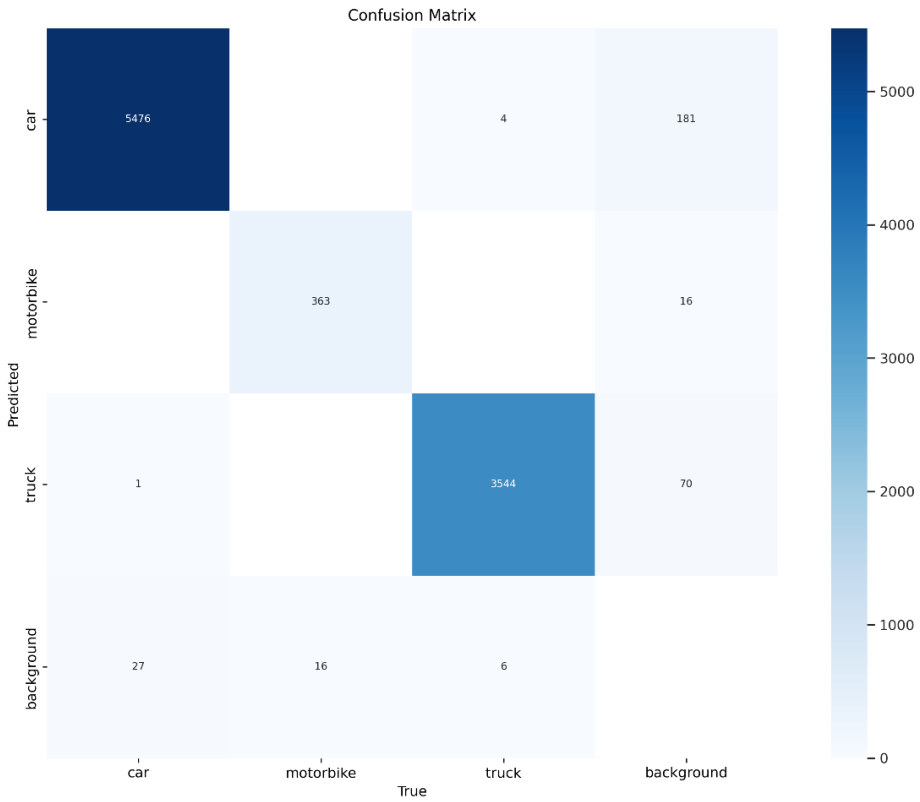


Figura 25 – Matriz confusão em números absolutos

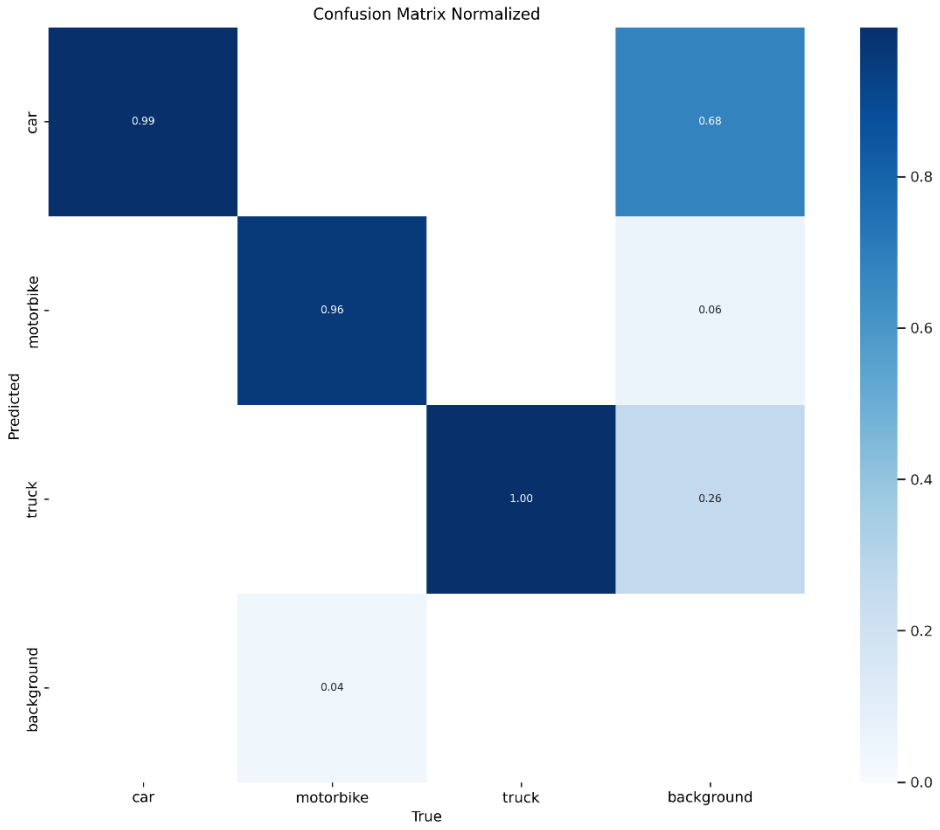


Figura 26 – Matriz confusão normalizada por classe

O método de modelagem desenvolvido com YOLOv8 demonstrou sua superioridade na detecção de veículos com alta precisão, apesar da diversidade do conjunto de dados. Além disso, possui altíssima precisão na generalização e validação de novas imagens. O modelo teve um bom desempenho de detecção e persistência em testes realizados com as imagens testadas.

3.11. Etapa 11 – Teste da inteligência ‘Final’

Assim que o treinamento do modelo foi finalizado, alguns testes foram realizados com Python Nativo para medir o nível de assertividade, assim como o nível de recall na prática. Os resultados obtidos foram muito satisfatórios, conseguindo identificar praticamente todo veículo em cena, assim como manter os Ids por tempos superiores a 9 minutos.

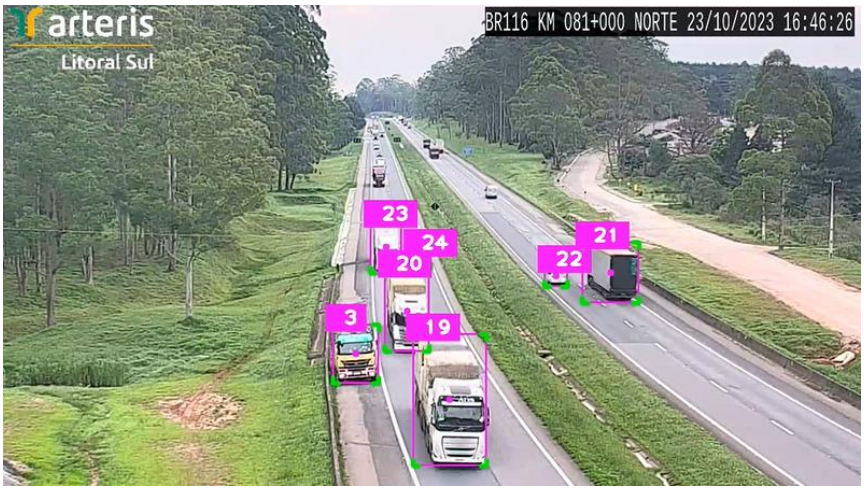


Figura 27 – Teste em Python Nativo

Posteriormente foi testado com o Deepstream já integrado ao envio de notificações e os resultados também se demonstraram eficazes.

Figura 28 – Teste em Deepstream

Após a resolução do ponto de melhoria identificado, referente ao recall, o modelo e o algoritmo estavam prontos para a realização de testes em tempo real.

3.12. Etapa 12 – Homologação final e coleta de resultados

3.12.1. Teste do modelo em tempo real

Para validação do modelo em tempo real, foi realizado um teste em tempo real na máquina virtual com a equipe da Arteris e da Programmer's.

O teste foi realizado por volta das 10 horas da manhã, no dia 21 de dezembro de 2023, com o auxílio da câmera de monitoramento localizada na BR 116 KM 81 Sentido Norte, litoral Sul. O teste contou com o apoio da equipe do Centro de controle operacional (CCO) da concessionária, que coordenou a operação do veículo para as simulações realizadas em pista. O tempo de aferição dos veículos foi dentro do esperado, e o modelo conseguiu distinguir com precisão as diferenças entre carros, caminhões e motos.

Definiu-se o tempo de 3 minutos para que o veículo parado enviasse um alerta. Primeiramente foi realizado o teste com o carro parado em uma distância média do limite capturado pela câmera, e atingido esse limite o alerta foi enviado corretamente para o canal de testes no Microsoft Teams, contendo informações de código identificador (ID) do veículo parado, o tempo de duração da parada, horário do alerta e identificação da rodovia, juntamente com uma imagem da câmera, com uma demarcação em vermelho ao redor do veículo notificado.

Figura 29 – Alerta do primeiro teste de parada

Em seguida, o carro deslocou-se mais a frente, mantendo seu númedo de ID, e parou novamente. Após o tempo definido, foi emitido um novo alerta de veículo parado.

Figura 30 – Alerta do segundo teste de parada

Como observado no segundo alerta, é retornado que o veículo ficou parado por 3 minutos ao invés de informar 6 minutos. Isso ocorreu porque o veículo se movimentou da posição onde estava parado anteriormente, para uma nova posição. Com isso, o cronometro é resetado para uma nova contagem de tempo real conforme a última paragem dos veículos.
Logo após, o veículo sai de cena para realizar um retorno e continuar os testes no outro sentido da rodovia, e após algum tempo, é enviado um alerta de que o veículo deixou o local.

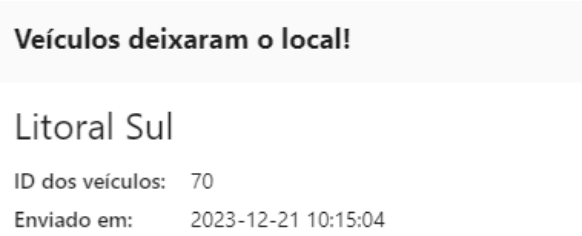


Figura 31 – Alerta de saída do veículo

Para última etapa do teste, agora no sentido sul da via, o veículo começa com ID de número 692, porém, após a passagem de caminhões em um curto espaço de tempo na frente do veículo fizeram com que o ID trocasse para 829. Mesmo assim, com a passagem de mais alguns caminhões, o rastreador manteve o ID por mais tempo e emitiu um alerta quando atingiu 3 minutos.

Figura 32 – Alerta do terceiro teste de parada

3.12.2. Estimativa de custo para escalar abordagem para mais câmeras

3.12.2.1. Arquitetura de referência para ambientes produtivos

Para detalhar a arquitetura e referência para um ambiente produtivo, os componentes e suas funções dentro da arquitetura proposta foram divididos em duas partes principais: Computação na Borda e Computação na Nuvem.

3.12.2.1.1. Computação na borda

1) Conexão com Câmeras: Haverá uma conexão direta e em tempo real com as câmeras instaladas nos trechos das rodovias para captura contínua de imagens ou vídeos, que serão analisados pelo modelo preditivo.
2) Azure Stack HCI (Azure Stack HCI – Infraestrutura Hiperconvergente | Microsoft Azure): Esta é a infraestrutura híbrida na borda (edge), que hospeda os recursos computacionais físicos necessários para processamento local. O Azure Stack HCI oferece uma plataforma robusta para executar aplicações virtualizadas em um ambiente on-premises, com integração direta ao Azure para serviços de nuvem.
    - Kubernetes no Azure Stack HCI: O Kubernetes será configurado sobre a infraestrutura do Azure Stack HCI para orquestrar os containers. Este

ambiente orquestrado permitirá a execução eficiente de aplicações em containers, neste caso, o modelo preditivo.

- Modelo Preditivo em Container: Dentro do Kubernetes, um container específico será responsável por executar o modelo preditivo de detecção de veículos parados nas rodovias. Este modelo analisará as transmissões em tempo real das câmeras instaladas ao longo dos trechos da rodovia.

3) Azure Arc: Facilitará a conexão segura e gerenciamento do ambiente de edge com os serviços de nuvem Azure, permitindo que os eventos detectados sejam enviados para a nuvem para processamento e análise adicionais.

3.12.2.1.2. Computação na nuvem

1) Azure Event Hubs (Azure Stack HCI – Infraestrutura Hiperconvergente | Microsoft Azure): Atuará como o ponto central de ingestão de eventos na nuvem, recebendo os dados dos veículos detectados pelo modelo preditivo no edge. O Event Hubs pode processar milhões de eventos por segundo, o que o torna ideal para esse cenário.
2) Azure Stream Analytics (Azure Stream Analytics | Microsoft Azure): Processará os eventos em tempo real à medida que chegam ao Event Hubs. É capaz de filtrar, classificar e agregar grandes fluxos de dados, encaminhando os resultados para armazenamento ou outras aplicações para análise.
3) Azure Data Lake (Data Lake Analytics | Microsoft Azure): Será utilizado para armazenar os dados processados pelo Azure Stream Analytics, oferecendo um vasto repositório para dados de grande volume, ideal para análises avançadas e machine learning.
4) Azure Databricks (Azure Databricks | Microsoft Azure): Integrado ao Data Lake, o Databricks será usado para processar e analisar os dados armazenados, gerando insights e indicadores que serão utilizados nos dashboards de analytics.
5) Microsoft Teams via Logic Apps (Logic App Service – IPaaS | Microsoft Azure): Os eventos relevantes, como a detecção de um veículo parado, serão comunicados aos funcionários da concessionária através de notificações enviadas ao Microsoft Teams. O Azure Logic Apps automatizará esse fluxo de trabalho, garantindo que as notificações sejam entregues em tempo real.

6) Dashboards de Analytics: Visualizações interativas e dashboards serão criados, possivelmente utilizando o Power BI (Power BI no Azure | Microsoft Azure) integrado ao Azure Databricks e Data Lake, para fornecer uma visão abrangente dos indicadores e análises geradas a partir dos dados coletados.

Esta arquitetura combina o poder do processamento de edge para análise em tempo real e baixa latência com a flexibilidade e escalabilidade da nuvem Azure para armazenamento, processamento adicional e análise de dados. Assim, permite uma resposta rápida a eventos críticos e fornece insights profundos para tomada de decisão baseada em dados.

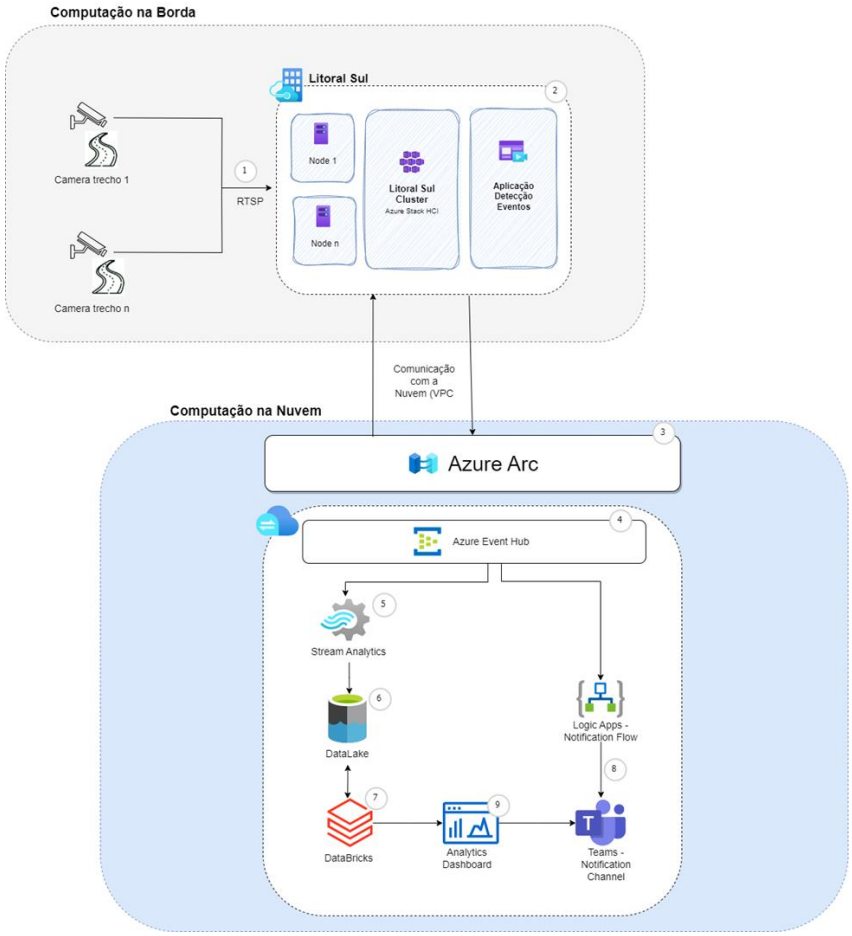


Figura 33 – Arquitetura para ambientes produtivos

3.12.2.2. Estimativa Investimento da Implementação

Para o cenário estimado, foi considerado o processamento paralelo de 10 câmeras, onde destaca-se alguns pontos de atenção:

- Qualidade da imagem da câmera;
- Quantidade de frames;
- Tráfego de rede e possível latência;
- Curva de aprendizado da solução;
- Atualização das tecnologias aplicadas na solução;
- Definição da zona de processamento na cloud;

Vale ressaltar que a estimativa apresentada como referência contém os valores aproximados e sugeridos ao fim da realização do projeto, em janeiro de 2024, e não refletem e nem representam uma proposta oficial.

A estimativa realizada contempla um valor único de investimento inicial e outro valor a ser consumido mensalmente.

3.12.2.2.1. Valor único

Foram considerados os valores de proposta comercial (jan/2024) da empresa DELL para fornecimento dos equipamentos e servidores, valores estes sujeitos a alterações.

Abaixo a descrição técnica de cada equipamento físico para a implementação:

Servidor Dell Technologies AX -750
AX-750 Quantity: 1
Chassis with up to 24x2.5" Drives
NVMe Backplane
No Rear Storage
GPU Enablement
Dell EMC AX-750
Azure Stack HCI Operating System
NVMe Node, Azure Stack HCI
All NVMe Config
Trusted Platform Module 2.0 V3
2.5" Chassis with up to 24 NMVe Switched Drives, 2 CPU

Intel® Xeon® Gold 6346 3.1G, 16C/32T, 11.2GT/s, 36M Cache, Turbo, HT (205W) DDR4-3200
Intel® Xeon® Gold 6346 3.1G, 16C/32T, 11.2GT/s, 36M Cache, Turbo, HT (205W) DDR4-3200
Heatsink for 2 CPU with GPU configuration
Performance Optimized
3200MT/s RDIMMs
16 x 8GB RDIMM, 3200MT/s, Single Rank
C30, No RAID for NVME chassis
No Controller
No Hard Drive
4 x 1.6TB Enterprise NVMe Mixed Use AG Drive U.2 Gen4 with carrier
UEFI BIOS Boot Mode with GPT Partition
Very High Performance Fan x6
Dual, Hot-Plug, Fully Redundant Power Supply (1+1), 2400W, Mixed Mode
2 x C19 to C20, PDU Style, 2.5M Power Cord, North America
Riser Config 2, Half Length, 4x16, 2x8 slots, SW GPU Capable
R750 Motherboard with Broadcom 5720 Dual Port 1Gb On-Board LOM
iDRAC9 Datacenter 15G
OpenManage Integration with MS Windows Admin Center Prem License for MSFT HCI Solutions, Perpetual
· Broadcom 57416 Dual Port 10GbE BASE-T Adapter, OCP NIC 3.0
· Mellanox ConnectX-5 Dual Port 10/25GbE SFP28 Adapter, PCIe Low Profile
· 2 x NVIDIA Ampere A2, PCIe, 60W, 16GB Passive, Single Wide, Full Height GPU,V2, SRV
· Azure Stack HCI, 2U Standard Bezel
· BOSS-S2 controller card + with 2 M.2 240GB (RAID 1)
· No Quick Sync
· iDRAC,Legacy Password
· iDRAC Group Manager, Disabled
· Microsoft Azure Stack HCI OS 22H2

· No Media Required
· ReadyRails Sliding Rails With Cable Management Arm
· Fan Foam, HDD 2U
· PowerEdge R750 CE and BIS Marking, No CCC Marking on 2.5" Chassis
· 5 Years ProSupport Plus Mission Critical 4-Hour Onsite Service-BZ
· ProDeploy Plus for AX 1U-2U-BZ Legacy

Switch Del Technologies | S5212F-ON
BZ - PowerSwitch S5212-ON Quantity: 2
Dell EMC S5212F-ON Switch, 12x 25GbE SFP28, 3x 100GbE QSFP28 ports, IO to PSU air, 2x PSU
OS10 Enterprise, S5212F-ON
Power Cord, 250V, 12A, 2 Meters, C13/C14
Dell NW Dual Tray, 4-post, S5212F-ON
S52XX User Manual
Dell Networking Cable, 100GbE, QSFP28 to QSFP28, Passive Copper Direct Attach, 1 Meter
3 x Dell Networking, Cable, SFP28 to SFP28, 25GbE, Passive Copper Twinax Direct Attach Cable,3 Meter
5 Years ProSupport Plus OS10 Enterprise Software Support-Maintenance-B
5 Years ProSupport Plus Mission Critical 4Hr Onsite Service, BZ
ProDeploy Plus Dell Networking S Series 5XXX Switch BZ Legacy

NVIDIA AI Enterprise - Essentials Edition Perpetual Bundle (Per GPU)		
NVIDIA Part Number	**Quantidade**	**Description**
731-AI7007+P3CMI60	2	NVIDIA AI Enterprise 24x7 Support Services for NVIDIA AI ls per GPU 5 Years

Architecture Diagram

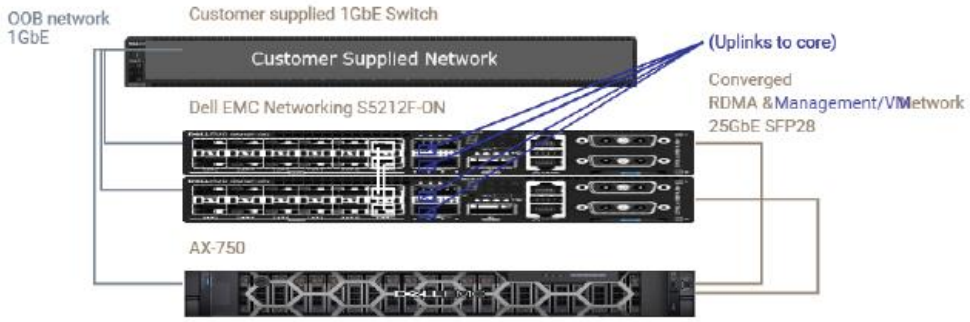


Figura 34 – Arquitetura Servidor DELL

Os valores para investimento estão distribuídos na tabela abaixo:

Produto	Unid.	Valor unitário	Valor total
Dell EMC S5212F-ON Switch (2x Unidades)	2	R$ 61.555,99	R$ 123.111,98
Dell EMC AX-750	1	R$ 215.851,52	R$ 215.851,52
NVIDIA AI Enterprise 24x7 Support Services for NVIDIA AI Enterprise Essentials per GPU 5 Years (modelo de licenciamento "Perpétuo" com suporte válido por 60 meses)	2	R$ 113.647,94	R$ 227.295,88
Total	**5**	**-**	**R$ 566.259,38**

3.12.2.2.2. Valor mensal

Além dos custos do Hardware, para esse tipo de solução será cobrado o "consumo" quando conectado no portal da Azure. O custo da assinatura do ASHCI OS seria de US$ 10/Core/Mês (Aprox. R$ 50/Core/Mês na cotação de jan/24 do dólar) e sua conta mensal seria de US$ 320 (Aprox. R$ 1600,00 na cotação de jan/24 do dólar), além dos custos dos demais serviços Azure descritos abaixo:

Tipo do Serviço do Azure	Descrição	Custo Estimado Mensal
Azure Kubernetes Service on Azure Stack HCI	2 núcleos virtuais,	R$237,45
Azure Arc	Azure Arc Kubernetes habilitado: Configurações do Kubernetes on, 32 vCPUs no cluster do Kubernetes; Instância Gerenciada de SQL habilitada para Azure Arc, Instância Única, Uso	R$286,92

	Geral: 0 vCore(s) no local, Instância Gerenciada de SQL habilitada para Azure Arc, Instância Única, Uso Geral: 0 vCore(s) no local X 200Horas, Licença incluída; SQL Server Habilitado para Azure Arc, Edição Standard, 0 X 200 Horas; Configuração de convidado do Azure Policy, 1 servidores	
Event Hubs	Camada Padrão: 4 unidades(s) de produtividade x 730 Horas, 100 milhão(ões) de eventos de entrada	R$447,20
Azure Stream Analytics	Padrão tipo, 1 unidade(s) de streaming x 730 Horas;	R$456,35
Azure Databricks	Carga de trabalho Computação para Todas as Finalidades, camada Padrão, 1 D4ASV5 (4 vCPU(s), 16 GB de RAM) x 730 Horas, PAGO CONFORME O USO, 1 DBU x 730 Horas	R$2.441,20
Storage Accounts	Redundância de Data Lake Store Gen2, Padrão, LRS, Quente Camada de Acesso, Namespace hierárquico Estrutura de arquivos, Capacidade de 1.000 GB - PAGO CONFORME O USO, operações de gravação: 4 MB x 10 operações, operações de leitura: 4 MB x 10 operações, 10 operações de leitura iterativa, 100.000 leitura de prioridade alta de arquivos, 10 operações de gravação iterativa, 10 Outras operações1.000 GB Recuperação de Dados, 1.000 GB Recuperação de dados de prioridade alta, 1.000GB Gravação de Dados, 1.000 GB Armazenamento de metadados	R$406,02
Logic Apps	Cargas de trabalho: Plano de consumo, 10.000 execuções de ação por dia x 1 dia , 100 GB de retenção de dados, 10.000 execuções do conector padrão por dia x 1 dia , 10.000 execuções do conector empresarial por dia x 1 dia ; Ambiente de Serviço de Integração: nível Premium, 0 unidades base x 730 Horas, 0 unidades de escala x 730 Horas; Contas de integração: 0 contas de integração Standard x 730 Horas, 0 contas de integração Básicas x 730 Horas.	R$116,46
Total Mensal		**R$4.391,60**

3.12.2.2.3. Resumo dos valores

Considerando os valores acima de investimento inicial (valor único) e valores a serem disponibilizados mensalmente, a estimativa para o processamento paralelo para cada 10 câmeras ao vivo é de:

1) Investimento inicial para aquisição dos equipamentos: R$ 566.259,38;
2) Valor mensal para utilização dos serviços Azure: R$ 5.991,60 + R$ 50/Core (Cotação de jan/24 do dólar) utilizado ASHCI OS.

4. CONSIDERAÇÕES E PRODUTOS GERADOS

4.1. Considerações Finais

O modelo de visão computacional desenvolvido neste projeto, mesmo que em um curto prazo, evidencia uma notável aplicabilidade de seu uso, abrindo oportunidades para a implementação de uma variedade de soluções no âmbito da operação e gestão rodoviária.

Por meio da abrangente execução do projeto, confirmou-se que o modelo de visão computacional desenvolvido demonstrou uma capacidade eficiente de detecção, um tempo de aferição otimizado, uma clara distinção entre classes e uma persistência adequada dos IDs dos veículos, proporcionando condições favoráveis para o alcance dos objetivos propostos.

Ao focar na detecção de veículos parados em acostamento associados a incidentes, entende-se que a aplicação do modelo desenvolvido não apenas fortalece a resposta a emergências, mas também a eficiência na prestação de serviços aos usuários. A identificação de incidentes, como panes mecânicas, acidentes ou outros contratempos, possibilita uma resposta imediata por parte das equipes de apoio e de socorro, minimizando o tempo de intervenção e reduzindo potenciais riscos para os usuários da rodovia.

A flexibilidade do modelo permite sua aplicação em contratos atuais e futuros, demonstrando sua versatilidade e adaptabilidade às necessidades em constante evolução do setor rodoviário. Em conjunto, essas vantagens reforçam a relevância e o impacto positivo da aplicação do modelo de visão computacional, o que tende a impulsionar a eficiência, a segurança e a qualidade dos serviços prestados pela concessionária.

4.2. Pontos de melhoria

- **Arquitetura da solução**

A arquitetura desenhada para a solução deste projeto, apesar de simplificada, demonstrou excelentes resultados em um curto espaço de tempo. Em um cenário futuro em que fatores como capacidade de rede e processamento são chaves para o sucesso da implementação da tecnologia de visão computacional, sugere-se avaliar a utilização de uma arquitetura hibrida onde a execução dos modelos e analítica avançada sejam realizadas na borda (Edge), minimizando assim o consumo de tráfego pesado de imagens pela rede e disrupção do processamento em caso de falha da rede, com o MLOPs dos modelos sendo realizado em Cloud, para fins de comparação com a ferramenta atual.

- **Perda de ID do veículo**

Embora a inteligência para identificação de veículos nas rodovias tenha se mostrado altamente eficaz, é importante reconhecer um desafio específico encontrado no decorrer do projeto que pode surgir em determinadas situações operacionais. Um ponto a ser considerado é a possibilidade de o veículo temporariamente perder seu Id inicial devido à passagem rápida de outros veículos maiores, como caminhões, em sua frente. Esta circunstância pode ocorrer em cenários muito específicos, como em um posicionamento particular da câmera, em situação de tráfego intenso, onde a visibilidade do veículo menor pode ser comprometida por pequenos períodos.

No entanto, essa limitação identificada não compromete o funcionamento do modelo. Este ponto específico pode ser contornado por meio de outras adaptações, como a implantação de presets com tempo programado de reposicionamento das câmeras, a opção de realizar uma busca no frame pelo veículo antes de removê-lo da lista de carros parados ou diminuir o tempo percorrido para identificação do veículo parado e posterior envio do alerta.

4.3. Sugestões para projetos futuros

- Ampliar a detecção para incluir objetos na pista que possam representar riscos, como animais, objetos, pedestres ou ciclistas. Isso contribuiria para a prevenção de acidentes e garantiria maior segurança dos usuários;
- Capacidade de avaliar a direção dos veículos no sentido contrário da pista ou outros comportamentos perigosos de direção dos veículos;

- Integrar sensores meteorológicos para monitorar as condições climáticas na rodovia. Isso pode auxiliar na antecipação de problemas causados por chuva intensa, neblina, ou outras condições adversas;
- Identificar através da visão computacional a necessidade de intervenções no pavimento e reparos na sinalização.

4.4. Reutilização do modelo

Várias ferramentas foram necessárias para fazer o projeto avançar, e por mais que algumas não sejam mais utilizáveis no final, muitas conseguem ser reaproveitadas para projetos e atividades futuras.

1) Os modelos compactos treinados no Azure Custom Vision podem ser reutilizados para cenários que não necessitam especificamente de recall, onde o ponto forte em si seja a assertividade que o modelo pode ter daquilo que detectou (precisão). Seja as versões de 3 ou 2 classes, ambas estão prontas e a disposição.
2) Quanto aos scripts, vários foram criados para diferentes situações, seja modificando os rótulos ou as imagens. Sobre as técnicas de Data Augmentation utilizando a biblioteca Albumentations, já possuímos scripts prontos e otimizados esperando apenas entrada de imagens já rotuladas para multiplicação delas, sendo necessário especificar apenas caminhos e parâmetros que quer nas variações.
3) E por último temos scripts utilizados para fazer comunicações mais rápidas e práticas com os serviços do Azure Custom Vision, seja utilizando a própria API dos serviços cognitivos ou o SDK Deepstream mesmo.

4.5. Produtos Gerados

- Modelo final com melhor desempenho treinado e testado;
- Relatório final com os resultados da execução da tecnologia desenvolvida e a viabilidade de seu uso na concessão Autopista Litoral Sul S/A;
- Todos os modelos treinados e algoritmos desenvolvidos durante o projeto de MVP;
- Ambiente tecnológico para aplicação de projetos de visão computacional.