Arquitetura Embarcada com TinyML e Modelos Linguísticos para Monitoramento Veicular Inteligente

Rejanio Moraes, Morsinaldo Medeiros, Fellipe Milomem, Marianne Silva e Ivanovitch Silva

***Resumo*— A Internet das Coisas (IoT) no contexto automotivo impulsiona o desenvolvimento de soluções embarcadas que realizam inferência local e comunicação em tempo real. Nesse cenário, técnicas como TinyML, que executa modelos de aprendizado de máquina em dispositivos com recursos limitados, e Small Language Models (SLMs), que geram descrições interpretáveis localmente, ganham destaque. Este trabalho propõe uma abordagem embarcada em Raspberry Pi, integrando dados via protocolo OBD-II, GPS e acelerômetro a um pipeline de inferência contextual. A solução estima estilo de direção, emissão de CO_2, tipo de via e anomalias, usando um agente linguístico para relatar eventos detectados com baixa latência.**

***Palavras-Chave*— IoT Automotiva, TinyML, Agente linguístico.**

***Abstract*— The Internet of Things (IoT) in the automotive context drives the development of embedded solutions capable of performing local inference and real-time communication. In this scenario, techniques such as TinyML, which runs machine learning models on resource-constrained devices, and Small Language Models (SLMs), which locally generate interpretable descriptions, have gained prominence. This work proposes an embedded approach using a Raspberry Pi platform, integrating data via the OBD-II protocol, GPS, and accelerometer into a contextual inference pipeline. The solution estimates driving style, CO_2 emissions, road type, and anomalies, using a language agent to report detected events with low latency.**

***Keywords*— Automotive IoT, TinyML, Language Agent.**

I. Introdução

A integração entre veículos conectados, Internet das Coisas (IoT) e inteligência artificial embarcada tem impulsionado o surgimento de sistemas veiculares inteligentes, ampliando as capacidades de automação e monitoramento em tempo real [1]. Com a evolução das tecnologias de comunicação sem fio, como Wi-Fi, LTE e 5G, dispositivos embarcados tornaram-se elementos-chave na coleta, processamento e transmissão de dados em tempo real [2]. Essa infraestrutura tem permitido aplicações voltadas à segurança viária, ao monitoramento do condutor e à manutenção preventiva, que dependem de decisões rápidas e descentralizadas.

Nesse cenário, surge o paradigma de TinyML, que visa a execução de modelos de aprendizado de máquina em dispositivos com restrições de energia, memória e processamento [3]. A combinação de TinyML com plataformas embarcadas de baixo custo, como o Raspberry Pi, viabiliza a incorporação de inteligência local em soluções de IoT, reduzindo a dependência da nuvem e os custos com comunicação de dados [4]. Além disso, técnicas de compressão, quantização e destilação têm viabilizado a execução de *Small Language Models* (SLMs) em dispositivos de borda [5]. Isso tem expandido as capacidades desses sistemas no tratamento de dados semiestruturados ou contextuais, como descrições interpretáveis de eventos veiculares [6].

Apesar desses avanços, ainda é limitada a adoção de arquiteturas embarcadas capazes de unir aquisição de dados, inferência contextual com modelos de linguagem e comunicação eficiente em um mesmo pipeline [7]. A maioria das soluções limita-se à coleta e transmissão de dados brutos ou à inferência remota em nuvem, o que compromete a resposta em tempo real, especialmente em ambientes com conectividade instável ou intermitente [8].

Diante deste contexto, este trabalho propõe uma abordagem embarcada para inferência contextual em veículos, composta por módulos de aquisição sensorial, inferência embarcada, agentes e SLM e interface gráfica local, todos executados em uma plataforma Raspberry Pi. A solução integra dados veiculares adquiridos via protocolo de comunicação OBD-II com sensores auxiliares (GPS e acelerômetro), sendo capaz de realizar inferências em tempo quase real por meio de modelos de aprendizado de máquina e *softsensors*. Os algoritmos implementados estimam variáveis como estilo de direção, emissão de CO_2, tipo de combustível, classificação da via trafegada, exigência dos sistemas veiculares e detecção de anomalias.

Além disso, a abordagem inclui um *agente linguístico*, responsável por converter os eventos inferidos em descrições textuais interpretáveis, utilizando um SLM embarcado. Entre os exemplos de uso, estão alertas do tipo "Condução agressiva detectada em zona urbana com histórico de acidentes" ou "Velocidade acima do permitido em trecho com alta incidência de infrações", gerados com base na inferência local e no cruzamento com dados abertos da Polícia Rodoviária Federal (PRF).

Por fim, a abordagem proposta foi validada por meio de um estudo de caso conduzido em ambiente real, com a instrumentação de um veículo em tráfego urbano e rodoviário. Os resultados obtidos demonstraram a viabilidade técnica da solução, a responsividade dos modelos embarcados e a estabilidade operacional do sistema mesmo sob condições variáveis, evidenciando sua capacidade de adaptação e funcionamento

Rejanio Moraes, Morsinaldo Medeiros, Fellipe Milomem, Ivanovitch Silva, Universidade Federal do Rio Grande do Norte, Natal-RN, e-mails: {rejanio.moraes.080@ufrn.edu.br morsinaldo.medeiros.075, fellipe.nogueira.702}@ufrn.edu.br, ivanovitch.silva@ufrn.br; Marianne Silva, Universidade Federal de Alagoas, Penedo-AL, e-mail: marianne.silva@penedo.ufal.br. Este trabalho foi parcialmente financiado pela FUNDEP (29271.01/2023.03-00) e CNPq (405531/2022-2).

contínuo em cenários dinâmicos e restritivos.

O artigo está organizado da seguinte forma: a Seção II apresenta os trabalhos relacionados; a Seção III descreve a abordagem proposta; a Seção IV detalha o estudo de caso para validar a abordagem; a Seção V discute os resultados obtidos; e por fim, a Seção VI apresenta as conclusões e trabalhos futuros.

II. Trabalhos Relacionados

A IoT tem viabilizado o surgimento de soluções embarcadas inteligentes voltadas ao monitoramento veicular em tempo real. Nesse contexto, o paradigma do TinyML tem se destacado por possibilitar a execução de modelos de aprendizado em dispositivos com recursos limitados. Em [9], os autores propõem um sistema de detecção de sonolência baseado em TinyML, utilizando modelos como SqueezeNet e MobileNet-V2 em microcontroladores, validando sua aplicabilidade em aplicações de segurança veicular com baixa latência e baixo consumo energético.

Outra aplicação de TinyML no contexto automotivo é apresentada em [6], onde os autores otimizam SLMs para controle embarcado de funções veiculares. O estudo mostra como arquiteturas leves podem ser adaptadas para chamadas de função (*function-calling*) dentro do veículo, ampliando a autonomia das interações sem dependência de nuvem. Apesar dos avanços, esses trabalhos concentram-se em tarefas isoladas, sem integração de múltiplos modelos ou inferência contextual abrangente.

Em paralelo, [10] explora a classificação do comportamento de condução a partir de dados coletados através do protocolo de comunicação OBD-II, empregando algoritmos como SVM, AdaBoost e Random Forest. A proposta atinge bons resultados na identificação de estilos de direção, mas mantém abordagem centrada em inferência remota ou em pós-processamento, sem incorporar elementos linguísticos interpretáveis nem operação em tempo real embarcado.

Diferente dessas abordagens, este trabalho propõe uma abordagem embarcada com agentes de inferência, incluindo um SLM local, responsável por converter eventos em alertas interpretáveis. A solução integra sensores veiculares (OBD-II, GPS, acelerômetro) em um pipeline unificado, operando de forma autônoma em tempo real e com comunicação adaptativa, o que representa um avanço prático e técnico no contexto da IoT automotiva embarcada.

III. Abordagem Proposta

A abordagem proposta neste trabalho consiste na construção de uma arquitetura embarcada e inteligente, composta por múltiplos módulos integrados de aquisição, inferência e comunicação, capazes de operar de forma autônoma mesmo sob restrições de conectividade e processamento. O sistema é implementado sobre uma plataforma baseada em Raspberry Pi 5 e utiliza sensores veiculares (OBD-II), posicionamento global (GPS) e movimento inercial (acelerômetro/giroscópio), compondo um pipeline para monitoramento contextual de veículos.

A solução combina algoritmos de TinyML, SLMs e *softsensors* para estimar variáveis como comportamento do condutor, tipo de via, emissão de CO_2 e risco de sinistros ou infrações. Os dados inferidos são visualizados em tempo real por meio de uma interface sensível ao toque e organizados em uma estrutura de agente, na qual agentes especializados operam em paralelo para aquisição, análise, geração textual e comunicação dos eventos detectados.

Deste modo, na Figura 1 observa-se uma visão geral da abordagem proposta, destacando a estrutura de hardware, o processo de aquisição sensorial, os modelos de inferência embarcados, as inferências técnicas com agentes (SLM) e interface gráfica local. Nas subseções seguintes, cada um desses módulos é descrito em detalhe.

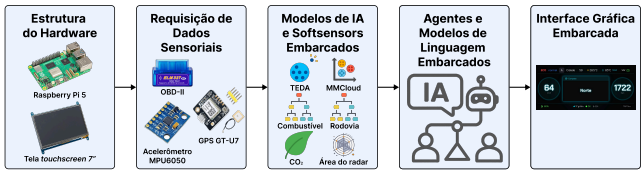


Fig. 1: Visão geral da abordagem proposta.

A. Estrutura de Hardware

A estrutura foi implementada sobre uma plataforma embarcada de baixo custo e alto desempenho, tendo como núcleo computacional o Raspberry Pi 5 B (processador ARM v8 quad-core e 8gb ram) devido à sua compatibilidade com bibliotecas de inferência local, conectividade integrada (Wi-Fi, Bluetooth, USB) e suporte a sistemas operacionais completos.

O sistema incorpora três sensores, o GPS GoolooTech GT-U7 (via UART) para posicionamento geográfico, o acelerômetro/giroscópio MPU6050 (via I2C) para detecção de padrões dinâmicos de movimento, e um adaptador OBD-II Bluetooth para aquisição de dados veiculares em tempo real. Complementa-se com uma tela sensível ao toque de 7 polegadas, utilizada para visualização local dos eventos inferidos e interação com os módulos do sistema. Todos os componentes foram integrados fisicamente via conexões GPIO, I2C, UART e USB, com alimentação por banco de baterias, garantindo operação autônoma e contínua mesmo sob condições adversas.

B. Requisição de Dados Sensoriais

A coleta de dados é realizada em tempo real a partir de três fontes: GPS, acelerômetro/giroscópio e interface OBD-II. Cada sensor é acessado por bibliotecas compatíveis com o ambiente Linux embarcado, garantindo integração ao sistema.

O módulo GPS GT-U7 fornece dados de localização e tempo por meio da interface UART, utilizando a biblioteca `PySerial`. O sensor inercial MPU6050 opera via protocolo I2C, com leitura realizada pela biblioteca `smbus2`. Já os dados da ECU veicular são acessados por meio da interface OBD-II, com uso da biblioteca `PyOBD` e comunicação via Bluetooth serial (RFCOMM) com adaptadores ELM327.

As amostras obtidas são organizadas em um *buffer* incremental local, alimentando o pipeline de inferência e permitindo posterior sincronização com a nuvem.

C. Modelos de IA e Softsensors Embarcados

Após a aquisição dos dados sensoriais, o sistema executa um pipeline embarcado composto por modelos de aprendizado de máquina e *softsensors*, que estimam variáveis não diretamente mensuráveis a partir de sinais veiculares combinados. Os principais modelos utilizados são:

- **Detecção de *Outliers* (TEDA)** – Detecta anomalias com base em medidas de tipicidade e excentricidade no sensor virtual *área do radar*, composto por variáveis como velocidade, RPM, carga e posição do acelerador [11].
- **MMCloud** – Classifica o estilo de direção em *Agressivo*, *Normal* ou *Cauteloso*, com base na área do radar e carga do motor [12].
- **Random Forest** – Utilizado para classificar tipo de combustível (gasolina ou etanol) e outro para o tipo de via (urbana ou rodoviária), a partir de variáveis como RPM, velocidade, carga, aceleração e avanço da ignição [13].
- **Modelo de** CO_2 – Estima a emissão de CO_2 com base no combustível utilizado, consumo e distância percorrida [14].
- ***Softsensor* de área do radar** – Calcula a demanda de exigência do veículo com base em velocidade, RPM, carga e posição do acelerador [12].

Todos os modelos são executados localmente utilizando bibliotecas `Python` como `NumPy/scikit-learn`. Os resultados inferidos são armazenados com marcação temporal e geográfica, alimentando os módulos de visualização e comunicação.

D. Agentes e Modelos de Linguagem Embarcados

Esse módulo é estruturado com o Agente Linguístico, responsável por transformar inferências técnicas em mensagens interpretáveis utilizando um SLM embarcado. O modelo empregado é o Qwen2.5-0.5B, um modelo do tipo GPT-like, executado localmente via `GGML`, quantizado no formato `Q4_K`, com aproximadamente 494 milhões de parâmetros. Além disso, o agente Linguístico foi implementado com duas *tools*: uma para interpretar a saída do modelo de comportamento do motorista e a outra para informações de sinistros e infrações. Esse SLM foi configurado para gerar descrições curtas e contextuais sobre essas *tools*, como no seguinte exemplo de interação:

Usuário: *Como estou dirigindo?*

E o sistema responde, com base nas inferências:

Sistema: *Você está dirigindo de forma cautelosa. Ótimo para segurança e economia!*

Vale destacar que os alertas da *tool* de sinistros e infrações utilizam dados abertos da PRF[1], referentes ao período de 2019 a 2024. Os arquivos, em formato CSV e organizados por bimestre ou mês, foram unificados com a biblioteca `pandas`. As tabelas de sinistros já incluem coordenadas geográficas (latitude e longitude). Já os dados de infrações foram associados a coordenadas por meio da correspondência entre os campos BR e KM com registros de acidentes. Quando não foi possível estabelecer essa correspondência, os campos de localização foram mantidos vazios. As coordenadas obtidas foram então convertidas com `geopandas` para facilitar o cálculo da proximidade entre o ponto atual e os locais registrados.

Esse alerta permite indicar zonas de maior risco de sinistros e trechos com histórico elevado de infrações, contribuindo para inferências mais contextualizadas sobre o ambiente de condução.

[1] https://www.gov.br/prf/pt-br/acesso-a-informacao/dados-abertos/dados-abertos-da-prf

E. Interface Gráfica Embarcada

A interface local do sistema é composta por uma tela sensível ao toque de 7 polegadas, conectada ao Raspberry Pi via HDMI e USB. Desenvolvida com `Electron Vite React`, essa interface possibilita a visualização em tempo real dos dados inferidos, alertas contextuais e histórico local de eventos. O layout da aplicação contempla o Painel Veicular que exibe variáveis obtidas via OBD-II; e Indicadores Ambientais que demonstra estimativas embarcadas de emissão de CO_2, consumo e tipo de combustível, além da classificação do estilo de condução (ver Figura 2).

Fig. 2: Interface embarcada exibindo painel veicular.

IV. Estudo de Caso

Nesta seção apresenta-se o estudo de caso que tem como objetivo avaliar a abordagem proposta em ambiente real, observando sua capacidade de aquisição, inferência e comunicação de eventos.

Com base nisso, definem-se as seguintes questões de pesquisa: **QP1:** A abordagem embarcada permite inferência contextual em tempo real, com latência e tempo de resposta adequados às exigências de aplicações veiculares? **QP2:** O sistema consegue identificar eventos relevantes com cobertura satisfatória ao longo do trajeto monitorado? **QP3:** A utilização dos recursos computacionais é compatível com os limites da plataforma embarcada utilizada?

A. Instrumentação

O estudo de caso foi realizado em um veículo Volkswagen Polo 1.0 TSI, ano 2019, instrumentado com um adaptador OBD-II ELM327 (versão 1.5) conectado via Bluetooth à ECU do veículo. Esse dispositivo foi utilizado como coletor de dados veiculares durante o trajeto, fornecendo informações como velocidade, rotação do motor, temperatura, entre outros.

B. Preparação e Execução

O condutor foi instruído a seguir uma rota pré-estabelecida de aproximadamente 6 km na cidade de Natal/RN, abrangendo trechos urbanos e rodoviários (BR-101). A rota inclui zonas com limite de velocidade variável e regiões com diferentes características operacionais. O condutor não foi informado previamente sobre os critérios avaliados, de modo a garantir comportamento de condução natural e não induzido.

A execução foi realizada no período da manhã, em horário de tráfego moderado, com duração total de cerca de 11 minutos. Durante o percurso, o sistema embarcado operou de forma autônoma, realizando aquisição sensorial, inferência local e registro dos dados em tempo real. Desse modo, totalizou-se uma amostra de 593 registros, composta por dados veiculares, geoespaciais e inferências geradas ao longo da rota.

C. Métricas de Avaliação

Para validar o desempenho da solução embarcada, foram consideradas as seguintes métricas:

- **Tempo de inferência e resposta** (L_i, R_{SLM}): calculado como a média do tempo de execução local de um modelo i ou da geração de resposta do SLM:

$$L_i = \frac{1}{n}\sum_{j=1}^{n}(t_j^{\text{fim}} - t_j^{\text{início}}), \quad R_{\text{SLM}} = \frac{1}{m}\sum_{k=1}^{m}(t_k^{\text{resposta}} - t_k^{\text{disparo}})$$

- **Cobertura de eventos**: proporção de segmentos do trajeto com eventos;
- **Uso de recursos computacionais**: média de uso de CPU e RAM durante o trajeto:

$$U_{\text{CPU}} = \frac{1}{T}\sum_{t=1}^{T}\text{CPU}(t), \quad U_{\text{RAM}} = \frac{1}{T}\sum_{t=1}^{T}\text{RAM}(t)$$

Por fim, com o objetivo de garantir reprodutibilidade, todos os códigos-fonte, modelos embarcados, *scripts* de instrumentação e arquivos de configuração utilizados neste estudo de caso estão disponíveis publicamente em repositório no GitHub[2].

V. Resultados

Esta seção descreve os resultados obtidos a partir da avaliação do estudo conduzido com foco na análise de desempenho dos modelos embarcados, cobertura de eventos e uso de recursos computacionais. Deste modo, avaliou-se em relação à latência, o tempo médio de resposta do modelo linguístico SLM ao longo do trajeto foi de aproximadamente 1,99 segundos, com os quartis variando entre 1,67 (Q1) e 2,39 (Q3) segundos. Apesar de picos iniciais superiores a 3 segundos, os tempos se mantiveram abaixo de 2,5 segundos, demonstrando resposta adequada para aplicações embarcadas em tempo real (Figura 3).

Complementar, a Tabela I detalha os tempos de inferência dos algoritmos embarcados. Os modelos TEDA e MMCloud apresentaram desempenho muito rápido, com médias inferiores a 1 ms, o que é ideal para aplicações em tempo real. Já os modelos de classificação de via (cidade/rodovia) e de combustível (gasolina/etanol) registraram médias de 296 ms e 124 ms, respectivamente. Esses tempos são considerados compatíveis com o contexto veicular, onde respostas de até meio segundo são geralmente aceitáveis [11], [12]. Os resultados demonstram que todos os algoritmos são executados com eficiência embarcada e dentro dos limites operacionais da aplicação.

[2]https://github.com/conect2ai/SBRT2025-MMCloud-Agent

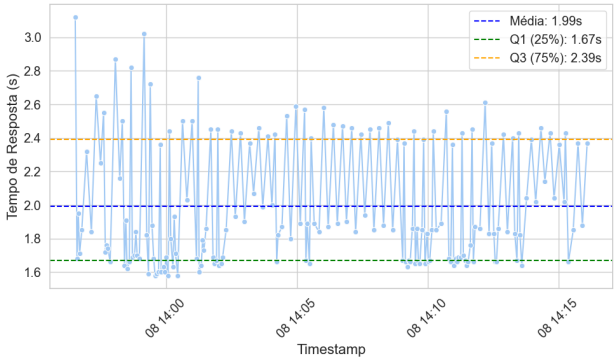


Fig. 3: Tempo de resposta do modelo SLM ao longo do trajeto.

TABELA I: Tempos de inferência dos algoritmos embarcados.

Algoritmo	Média ± DP (ms)	Mediana (ms)	Mínimo (ms)	Máximo (ms)
TEDA	0.21 ± 0.07	0.21	0.11	0.40
MMCloud	0.84 ± 0.45	0.72	0.34	2.20
Cidade/Rodovia	296.18 ± 59.12	278.90	250.03	541.70
Gasolina/Etanol	123.62 ± 99.14	81.14	50.02	479.91

Além disso, avaliou-se a cobertura espacial dos eventos por meio de um mapa georreferenciado (Figura 4), que demonstra a correlação entre registros oficiais de sinistros/multas e os eventos inferidos pela abordagem proposta.

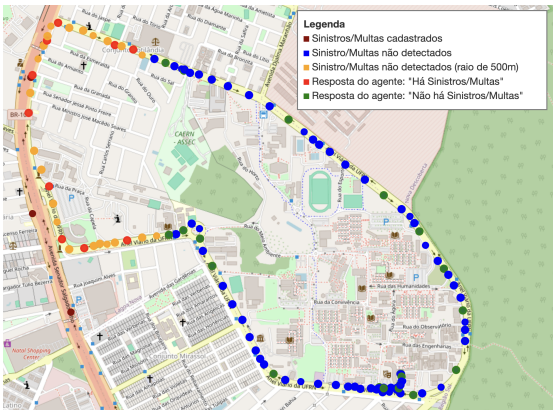
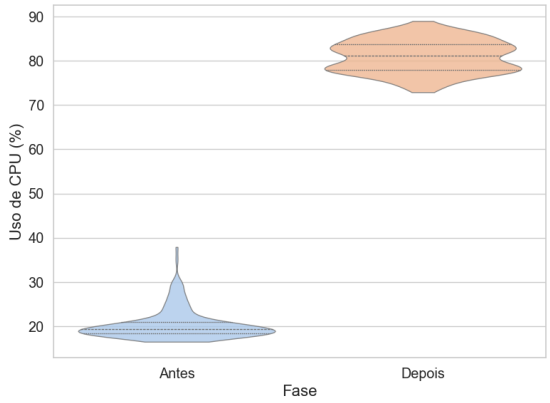


Fig. 4: Detecção de sinistros/multas e respostas do agente.

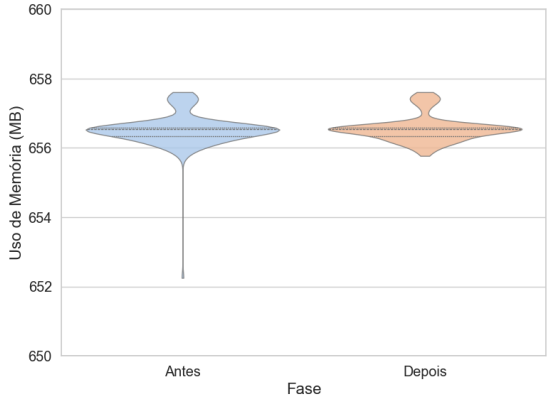
Observou-se que os pontos detectados pelo sistema concentraram-se nas proximidades dos registros oficiais, e que as respostas do agente linguístico acompanharam essas inferências. Isso indica alinhamento entre a inferência embarcada e a base externa, com boa cobertura ao longo do trajeto monitorado. O raio de 500 metros adotado para associação dos eventos demonstrou-se adequado para o padrão de segmentação dos dados geográficos utilizados.

Quanto ao uso de recursos computacionais, a Figura 5 apresenta a distribuição do uso de CPU e memória RAM. Após

a ativação dos modelos de inferência, o uso de CPU aumentou de 19% para 81%, sem ultrapassar os limites operacionais da plataforma. A memória RAM permaneceu estável em torno de 656 MB, demonstrando eficiência no gerenciamento de recursos embarcados durante a operação contínua.



(a) Distribuição do uso de CPU (%).



(b) Distribuição do uso de memória RAM (MB).

Fig. 5: Uso de recursos computacionais nas inferências.

Por fim, verifica-se que a abordagem proposta atinge os principais requisitos para aplicações embarcadas em contexto veicular. O sistema apresentou latência e tempo de resposta compatíveis com operações em tempo real, cobertura espacial alinhada com dados externos de sinistros e multas, e utilização eficiente dos recursos computacionais da plataforma. Dessa forma, considera-se que as três questões de pesquisa definidas neste trabalho — relacionadas à responsividade (QP1), cobertura de eventos (QP2) e viabilidade computacional (QP3) — foram satisfatoriamente atendidas, validando a aplicabilidade da abordagem proposta em cenários reais de IoT automotiva.

VI. Conclusão

Este trabalho propôs uma abordagem embarcada para inferência contextual em veículos, integrando modelos de aprendizado de máquina e um agente linguístico baseado em SLM, executados localmente em um Raspberry Pi com dados obtidos via OBD-II, GPS e acelerômetro.

O estudo de caso demonstrou que a solução atende aos critérios definidos pelas questões de pesquisa. Na qual, apresentou latência e tempo de resposta compatíveis com aplicações em tempo real (QP1), identificou eventos relevantes com boa cobertura espacial (QP2) e operou dentro dos limites computacionais da plataforma embarcada (QP3). Deste modo, pode-se concluir que a proposta é tecnicamente viável para aplicações em IoT automotiva inteligente.

Como trabalhos futuros, propõe-se expandir os estudos de caso com múltiplos veículos, implementar uma arquitetura multiagente para execução paralela dos módulos e investigar estratégias de comunicação V2X para disseminação cooperativa de eventos em redes veiculares.

Agradecimentos

O presente trabalho foi realizado com o apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), Processo nº 405531/2022-2 e da Fundação de Desenvolvimento da Pesquisa – Fundep Rota 2030/Linha VI 29271.01.01/2023.03-00.

Referências

[1] A. Agbonyin, P. Reddy, and A. K. Jakkani, "Utilizing internet of things (iot), artificial intelligence, and vehicle telematics for sustainable growth in small, and medium firms (smes)," *International Journal of Computer Engineering and Technology*, vol. 15, no. 2, pp. 182–191, 2024.

[2] A. Kumar, P. Pattanayak, R. K. Verma, and G. Prasad, "Compact annular-shaped four-port mimo antenna for mid-band 5g/wi-fi/bluetooth/ism band/wlan and iot applications," *Wireless Personal Communications*, vol. 138, no. 1, pp. 547–574, 2024.

[3] A. Elhanashi, P. Dini, S. Saponara, and Q. Zheng, "Advancements in tinyml: Applications, limitations, and impact on iot devices," *Electronics*, vol. 13, no. 17, p. 3562, 2024.

[4] T. Jaiswal, M. Pandey, and P. Tripathi, "Real-time multiple object detection using raspberry pi and tiny-ml approach," *Recent Advances in Electrical & Electronic Engineering*, vol. 18, no. 2, pp. 244–255, 2025.

[5] Z. Örpek, B. Tural, and Z. Destan, "The language model revolution: Llm and slm analysis," in *2024 8th International Artificial Intelligence and Data Processing Symposium (IDAP)*. IEEE, 2024, pp. 1–4.

[6] Y. S. Khiabani, F. Atif, C. Hsu, S. Stahlmann, T. Michels, S. Kramer, B. Heidrich, M. S. Sarfraz, J. Merten, and F. Tafazzoli, "Optimizing small language models for in-vehicle function-calling," *arXiv preprint arXiv:2501.02342*, 2025. [Online]. Available: https://arxiv.org/abs/2501.02342

[7] H.-I. Liu, M. Galindo, H. Xie, L.-K. Wong, H.-H. Shuai, Y.-H. Li, and W.-H. Cheng, "Lightweight deep learning for resource-constrained environments: A survey," *ACM Computing Surveys*, vol. 56, no. 10, pp. 1–42, 2024.

[8] R. Zhang, H. Jiang, W. Wang, and J. Liu, "Optimization methods, challenges, and opportunities for edge inference: A comprehensive survey," *Electronics*, vol. 14, no. 7, p. 1345, 2025.

[9] M. Gupta, A. Arora, and A. Sharma, "Tinyml-based drowsiness detection system using edge ai on resource-constrained devices," *Sensors*, vol. 23, no. 12, p. 5696, 2023.

[10] K. Singh, D. Sharma, and R. Kumar, "Driving behavior classification using machine learning on obd-ii data," *The Journal of Supercomputing*, 2023.

[11] P. Andrade, M. Silva, M. Medeiros, D. G. Costa, and I. Silva, "Tedarls: A tinyml incremental learning approach for outlier detection and correction," *IEEE Sensors Journal*, 2024.

[12] M. Medeiros, T. Flores, M. Silva, and I. Silva, "A multi-layered methodology for driver behavior analysis using tinyml and edge computing," in *2024 IEEE International Conference on Evolving and Adaptive Intelligent Systems (EAIS)*. IEEE, 2024, pp. 1–8.

[13] M. Amaral, M. Medeiros, M. Andrade, T. Flores, M. Silva, and I. Silva, "Tinyml implementation and optimization for fuel type classification on obd-ii edge device," in *2024 Symposium on Internet of Things (SIoT)*, vol. 1. IEEE, 2024, pp. 1–5.

[14] M. Andrade, M. Medeiros, T. Medeiros, M. Azevedo, M. Silva, D. G. Costa, and I. Silva, "On the use of biofuels for cleaner cities: Assessing vehicular pollution through digital twins and machine learning algorithms," *Sustainability*, vol. 16, no. 2, p. 708, 2024.