AutoMHS-GPT: Automação de Seleção de Modelos e Hiperparâmetros por meio de Modelo Generativo para Detecção de Anomalias em Redes Veiculares

Lucas Airam C. de Souza[1,2], **Matteo Sammarco**[3], **Nadjib Achir**[2], **Miguel Elias M. Campista**[1], **Luís Henrique M. K. Costa**[1]

[1]Grupo de Teleinformática e Automação (GTA)
Universidade Federal do Rio de Janeiro (UFRJ)
[2]École Polytechnique, INRIA Saclay, França
[3] Independent Researcher

Resumo. *O Aprendizado de Máquina Automatizado surge como alternativa para reduzir o tempo de instanciação dos sistemas ao acelerar o processo de busca por modelos e hiperparâmetros. Essas técnicas, porém, ainda demandam alto tempo de execução. Em aplicações críticas, como a detecção de intrusão em redes veiculares, o atraso para a aplicação de contramedidas pode ocasionar catástrofes. Assim, é essencial garantir modelos acurados no menor tempo possível para detectar as ameaças de forma eficaz. Este trabalho propõe o AutoMHS-GPT, um sistema que utiliza a inteligência artificial generativa para reduzir o tempo de definição de hiperparâmetros e modelos na implantação do aprendizado de máquina para detecção de ameaças em redes veiculares. A partir de uma descrição do problema, o modelo generativo retorna um texto contendo o modelo adequado com os seus hiperparâmetros para o treinamento. Os resultados mostram que o AutoMSH-GPT produz modelos com maior desempenho de classificação de ameaças em comparação com abordagens de aprendizado de máquina automatizado avaliadas, AutoKeras e Auto-Sklearn, apresentando uma revocação 9% maior no melhor caso. Além disso, a proposta atual reduz o processo de busca e treinamento de modelos, realizando a tarefa em cerca de 30 minutos, enquanto os demais arcabouços avaliados necessitam entre dois a três dias.*

1. Introdução

A capacidade de automação de tarefas e redução de custos torna o aprendizado de máquina uma tecnologia atrativa para diversas áreas, como redes de computadores, veículos inteligentes, medicina, entre outras. No entanto, a implantação de modelos de inteligência artificial em ambientes de produção demora dias ou até meses [Paleyes et al. 2022]. Essa demora ocorre devido ao processo de coleta de dados e treinamento de modelos. Um modelo demanda um tempo de treinamento da ordem de minutos [Kumar et al. 2021]. O tempo de execução varia conforme o tamanho do conjunto de dados, modelo utilizado e a capacidade computacional disponível. Porém, com a grande disponibilidade de modelos e hiperparâmetros existentes, é difícil determinar para uma nova tarefa de aprendizado qual conjunto de modelo-hiperparâmetro utilizar. Assim, uma abordagem comum é aplicar a otimização de hiperparâmetros.

A tarefa de otimização de modelos e hiperparâmetros é a maior responsável pelo alto tempo de execução e consumo de recursos computacionais, uma vez que os processos de busca

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001, CNPq (408255/2023-4), FAPERJ (E-26/211.144/2019 e E-26/200.892/2021), Fundação de Desenvolvimento da Pesquisa - Fundep - Rota 2030 e FAPESP (15/24485-9 e 14/50937-1.)

geralmente são em espaços enormes, com milhares de combinações possíveis. Assim, automatizar a definição de modelos e hiperparâmetros é essencial para a redução do tempo de disponibilização de um modelo de aprendizado de máquina. Além disso, automatizar as etapas do aprendizado de máquina permite que usuários com pouco conhecimento técnico possam desenvolver modelos de aprendizado de máquina com alta qualidade, preocupando-se apenas com a aquisição dos dados.

Dessa forma, o Aprendizado de Máquina Automatizado (*Automated Machine Learning* - AutoML) [Feurer et al. 2015, Jin et al. 2019, LeDell e Poirier 2020] é um tópico de pesquisa que busca automatizar o processo de aprendizado, onde a seleção combinada de algoritmo e otimização de hiperparâmetros (Combined Algorithm Selection and Hyperparameter Otimization - CASH) é um dos casos específicos abordados pelo AutoML [Thornton et al. 2013]. O AutoML utiliza técnicas de otimização bayesiana para reduzir o espaço de busca e determinar direções mais promissoras dentro do espaço de hiperparâmetros e modelos, geralmente considerando um tempo máximo de execução. Entretanto, essas técnicas ainda demandam alto tempo de execução ou produzem resultados com baixo desempenho de classificação quando o tempo de busca é curto. O cenário torna-se mais preocupante em redes veiculares, onde as aplicações são sensíveis à latência. Em especial na área de detecção de intrusão, onde um veículo vulnerável pode ser comprometido por um atacante, colocando em risco a segurança física de passageiros.

Este trabalho propõe o AutoMHS-GPT[1], um sistema que utiliza a inteligência artificial generativa para reduzir o tempo de busca por hiperparâmetros e modelos na implantação do aprendizado de máquina. O sistema considera como caso de uso a instanciação de modelos para detecção de ameaças em redes veiculares. O ChatGPT [OpenAI 2023] é utilizado como a inteligência artificial generativa (*Generativa Artificial Intelligence* - GenAI). A partir de uma descrição do problema, "determinar o melhor modelo e conjunto de hiperparâmetros para um conjunto de dados de ataques em redes veiculares", a GenAI retorna um texto contendo uma resposta adequada. Após essa etapa, inicia-se o treinamento com o resultado transformado em um código que implemente o modelo. Os resultados mostram que o AutoMSH-GPT produz modelos com maior desempenho de classificação de ameaças em comparação com abordagens de aprendizado de máquina automatizado avaliadas, AutoKeras e Auto-Sklearn, apresentando uma revocação 9% maior no melhor caso. O mesmo efeito ocorreu em outras métricas avaliadas, como a acurácia, precisão e F1 score, demonstrando que o modelo acerta a classificação de mais amostras e gera menos falsos positivos do que as alternativas avaliadas. Além disso, a proposta atual reduz o processo de busca e treinamento de modelos, realizando a tarefa em cerca de 30 minutos, enquanto os demais arcabouços avaliados necessitam entre dois a três dias para a produção do modelo.

O trabalho está organizado da seguinte forma. A Seção 2 introduz o problema de detectar ameaças em redes veiculares e caracteriza as mensagens transmitidas. A Seção 3 apresenta o estado da arte em detecção de intrusão em redes veiculares e aplicações do aprendizado generativo em redes de computadores. A Seção 4 descreve o sistema proposto, as etapas e dados utilizados para seleção de modelos, hiperparâmetros e posteriormente o treinamento. A Seção 5 avalia o sistema em comparação com outras propostas de AutoML do estado da arte, o AutoKeras e o Auto-Sklearn. Por fim, a Seção 6 conclui este trabalho e apresenta as direções futuras de pesquisa.

[1]Acrônimo do inglês Automated Model and Hiperparameter Selection with Generative Pre-Trained model

2. Conceitos e Caracterização do Problema

Os veículos conectados necessitam de um protocolo de comunicação entre unidades de controle (*Eletronic Control Unit* - ECU). Apesar da existência de diversas alternativas como MOST, FlexRay e *Automotive Ethernet*, o protocolo CAN (*Controller Area Network*) é o mais utilizado para a comunicação entre os componentes internos de um veículo. A ampla adoção do protocolo CAN é justificada por sua baixa complexidade, escalabilidade e robustez contra interferências eletromagnéticas [Farsi et al. 1999].

Um quadro do protocolo CAN possui o formato com 8 campos divididos em um tamanho variável de no máximo 111 bits, exibido na Figura 1. O primeiro campo, início do quadro (*Start of Frame* - SOF) indica no barramento o início da transmissão de um quadro. O campo de arbitragem (*Arbitration Field*) contém o identificador e o bit de solicitação de transmissão remota (*Remote Transmission Request* - RTR). O RTR é responsável pela identificação do tipo de mensagem, podendo ser um quadro CAN ou uma mensagem remota. O identificador presente no campo de arbitragem pode variar conforme a versão. O campo de controle indica o tamanho da mensagem, pois o campo de dados, que contém o dado a ser transmitido, é variável. O campo CRC (*Cyclical Recovery Checking*) serve para a verificação da integridade da mensagem transmitida e para a identificação da presença de erros. O fim do quadro (*End of Frame* - EOF) determina o fim da transmissão, seguido por 3 bits de intervalo entre quadros (*Interframe Space* - IFS).

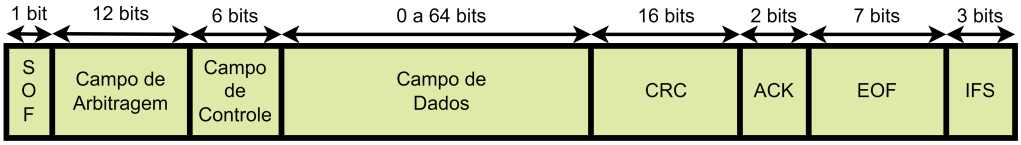

Figura 1: Formato do quadro CAN.

Além do protocolo CAN, os veículos conectados enviam mensagens de conscientização cooperativa (*Cooperative Awareness Messages* - CAM) para informar outros veículos sobre suas condições de direção. Entre as informações transmitidas, essas mensagens contêm, posição do veículo, velocidade, identificador da mensagem, estampa de tempo e aceleração. Apesar dessas mensagens serem essenciais para a construção de sistemas inteligentes de transporte cooperativo (*Cooperative Intelligent Transport Systems* - C-ITS) e melhorar as condições de direção, elas podem ser falsificadas para criar distúrbios na rede veicular. Além disso, as técnicas de criptografia existentes, como os códigos de autenticação de mensagem (*Message Authentication Code* - MAC) para garantia de integridade e autenticação são insuficientes para solucionar as vulnerabilidades relacionadas. Isto porque os veículos que executam ataques por meio de mensagens geralmente encontram-se autenticados no sistema. Assim, o desafio é verificar a veracidade das mensagens transmitidas.

Os ataques de redes veiculares abordados neste trabalho são relacionados com a veracidade das CAMs transmitidas. Entre os ataques abordados estão ataques de negação de serviço (*Denial of Service* - DoS), falsificação de posição e velocidade, difusão de localização de veículos inexistentes e repetição de mensagens. Por fim, o sistema proposto analisa as CAMs por meio de modelos de aprendizado de máquina com a finalidade de detectar comportamentos anômalos. Para agilizar o processo de treinamento, o sistema aplica a inteligência artificial generativa para a definição de modelo de aprendizado e seus hiperparâmetros. A seguir são discutidos os principais trabalhos relacionados à pesquisa desenvolvida.

3. Trabalhos Relacionados

Esta seção apresenta soluções do estado da arte para detecção de ameaças em redes veiculares, o caso de uso adotado pelo sistema proposto. Além disso, são discutidas alternativas para otimização de hiperparâmetros e seleção de modelos em sistemas de aprendizado de máquina. Por fim, são apresentadas aplicações dos modelos generativos em redes de computadores. A discussão demonstra que as principais aplicações estão restritas à detecção de anomalias ou geração de novas amostras para o treinamento de modelos.

3.1. Detecção de Ameaças em Redes Veiculares Inteligentes

A detecção de intrusão em ambientes veiculares é vital para garantir a segurança de condutores e passageiros. Diferentemente de outros sistemas computacionais, um veículo comprometido por um atacante pode causar acidentes, incluindo fatalidades. Assim, Yakan *et al.* propõem um sistema de detecção de intrusão para redes veiculares com foco na comunicação Veículo-para-Rede (*Vehicular-to-Network* - V2N) [Yakan et al. 2023]. Os autores usam modelos *Long Short-Term Memory* (LSTM) para capturar relações temporais em ataques e Aprendizado Federado (*Federated Learning* - FL) para aumentar a privacidade e a eficiência da comunicação durante o treinamento. Vinita e Vetriselvi [Vinita e Vetriselvi 2023] propõem um sistema para identificar a veracidade de mensagens de emergência transmitidas em redes veiculares. Por um lado, a comunicação Veículo-para-Tudo (Vehicle-to-Everything - V2X) traz oportunidades para aumentar a eficiência da condução, enviando uma atualização sobre as condições do tráfego local, como informar sobre acidentes. Por outro lado, os invasores podem difundir mensagens falsas para degradar as condições do tráfego. Assim, os autores propõem a utilização de modelos de aprendizado de máquina para detectar este comportamento malicioso, que executa um ataque Sybil informando falsos acidentes. Além disso, a proposta utiliza o paradigma de aprendizado federado para preservar a privacidade dos usuários. Bousalem *et al.* [Bousalem et al. 2023] propõem o uso de aprendizado por reforço para mitigar ataques DDoS em redes veiculares. A rede 5G-V2X permite instanciar fatias com poucos recursos de comunicação para isolar invasores ou clientes com comportamento suspeito. No entanto, esses nós isolados devem recuperar recursos depois que a ameaça cessar. Assim, os autores propõem um algoritmo de aprendizado por reforço para determinar quando reduzir os recursos dos usuários, atribuindo-os a uma fatia com recursos limitados, ou quando aumentar novamente seus recursos. Apesar das propostas serem promissoras no combate de ameaças em redes veiculares, os procedimentos desenvolvidos para selecionar modelos e ajustar seus hiperparâmetros representam um desafio ainda em aberto.

3.2. Seleção de Modelos e Otimização de Hiperparâmetros

A seleção de modelos e otimização de hiperparâmetros são dois dos principais desafios que existem no desenvolvimento de sistemas de aprendizado de máquina. Assim, um tópico de pesquisa relevante é a otimização de hiperparâmetros [Neto et al. 2022, Horváth et al. 2023]. O AutoML [Feurer et al. 2015, Jin et al. 2019, LeDell e Poirier 2020] é uma área de pesquisa cujo foco é a automatização do processo de aprendizado, definindo pré-processamento de dados, modelos e hiperparâmetros. O CASH é um dos casos específicos abordados pelo AutoML no qual o objetivo é selecionar modelos e hiperparâmetros [Thornton et al. 2013]. Existem diversas implementações de AutoML disponíveis, como Auto-Sklearn [Feurer et al. 2015], H2O [LeDell e Poirier 2020], Auto-Keras [Jin et al. 2019] e Google AutoML [Bisong e Bisong 2019]. Além disso, existem outras propostas como o Auto-CASH [Mu et al. 2022] e a proposta de Horváth *et al.* [Horváth et al. 2023]. O Auto-CASH é uma ferramenta que utiliza aprendizado por reforço

para automatizar o processo de seleção de hiperparâmetros e modelos e com isso reduzir a necessidade de intervenção humana. Por outro lado, Horváth *et al.* utilizam a Análise de Componentes Principais (*Principal Component Analysis* - PCA) em conjunto com uma medida de similaridade como uma forma de criar meta-características que auxiliem o processo de tomada de decisão na mesma tarefa de otimização. Entretanto, essas técnicas ainda demandam alto tempo de execução para atingir resultados satisfatórios. Por outro lado, o crescente desenvolvimento de aplicações que utilizam modelos generativos traz a questão sobre a capacidade dessas redes neurais em determinar em um curto tempo uma resposta que possua alto desempenho de classificação.

3.3. Modelos Generativos em Redes de Computadores

A utilização de inteligência artificial generativa (*Generative AI* - GenAI) e redes de computadores é um tema de pesquisa explorado atualmente. Jacobs *et al.* [Jacobs et al. 2021] propõem o uso de um prompt para a configuração automática da rede, analisando a intenção de usuários por meio de uma descrição de alto nível. Zhang *et al.* [Zhang et al. 2023] estudam os casos de uso de IA generativa em redes veiculares. Os autores identificam três aplicações de rede veicular que podem ser aprimoradas com o uso de IA generativa: simulação de tráfego, aumento de dados e avaliação de risco. Além disso, os autores propõem um método para redução da comunicação com recriação de alta fidelidade no envio de informações sobre acidentes. A proposta é enviar as características mais relevantes da imagem com um texto descritivo para reconstruir o original nos demais veículos.

Devido à capacidade desses modelos em gerar novos dados a partir de observações anteriores, a IA generativa é utilizada para aumentar conjuntos de dados a fim de treinar outros modelos em um maior conjunto de amostras. TrajGAIL (*Trajectory Generative Adversarial Imitation Learning*) [Choi et al. 2021] é um arcabouço que aplica IA generativa para a geração de trajetórias de veículos urbanos. Os autores propõem combinar um Processo de Decisão de Markov Parcialmente Observável (*Partially Observable Markov Decision Process* - POMDP) com o modelo generativo para gerar dados mais realistas. A obtenção de dados sofre desafios como escassez e questões de privacidade para obter conjuntos de dados de trajetórias urbanas, bem como alto custo computacional de abordagens como Aprendizado por Reforço Inverso (*Inverse Reinforcement Learning* - IRL). De forma similar, SCAN-GAN (*Synthetic Controller Area Network-Generative Adversarial Network*) [Chougule et al. 2023] é um método para gerar dados sintéticos, porém para detecção de intrusão em redes veiculares.

Outra possibilidade de aplicação de redes generativas é a transformação das características de entrada para identificação de anomalias. Cobilean *et al.* propõem um sistema de detecção de anomalias baseado em transformação para CANs [Cobilean et al. 2023]. Como as redes de transformação detectam dependências em sequências, os autores usam esse recurso para prever o valor provável em uma sequência de mensagens CAN. Quando o valor recebido difere muito da previsão, o sistema detecta uma anomalia. Zhao *et al.* propõem e avaliam quatro estruturas de um sistema de detecção de intrusão no barramento CAN [Zhao et al. 2022]. Os autores combinam uma Rede Adversarial Generativa (*Generative Adversarial Networks*- GAN) com detecção fora de distribuição (*Out-of-Distribution - OOD*) para classificar dados como normais, ataques conhecidos ou ataques desconhecidos. O artigo treina o discriminador da GAN para distinguir sequências de mensagens benignas e maliciosas de forma diferente das propostas anteriores que utilizam a GAN como gerador de dados. Além disso, a proposta aplica um modelo de floresta de isolamento para detectar ataques desconhecidos. Du *et al.* [Du et al. 2023] apresentam aplicações de IA generativa em redes de computadores e discutem como essa tecnologia pode impactar a segurança. A IA generativa pode ser uma fonte de código ou dados

maliciosos para envenenar modelos discriminativos, como sistemas de detecção de intrusões baseados em aprendizado de máquina. No entanto, essa tecnologia também é capaz de extrair características importantes de ataques ou mesmo criar novos padrões de ataque para treinar outros modelos.

Gupta *et al.* [Gupta et al. 2023] apresentam as aplicabilidades para a segurança em redes de computadores a partir do ChatGPT (Modelo Generativo Pré-treinado - *Generative Pretrained Model*). Os autores discutem a capacidade dessa ferramenta na geração automática de códigos para realização de ataques e contramedidas. Apesar da existência dos problemas de alucinação, o modelo é capaz de gerar códigos que realizam as tarefas demandadas em diversos cenários testados pelos autores. Juttner *et al.* [Jüttner et al. 2023] utilizam o ChatGPT como um mecanismo para explicar aos usuários não-especialistas o resultado da classificação realizada por Sistemas de Detecção de Intrusão (*Intrusion Detection Systems* - IDS). Além disso, os autores propõem utilizar o modelo generativo para indicar contramedidas necessárias.

Diferentemente das propostas anteriores, o trabalho atual utiliza a inteligência artificial generativa para, a partir de informações sobre o conjunto de dados e problema de aprendizado, produzir um modelo de aprendizado de máquina com hiperparâmetros ajustados a tarefa de aprendizado. A proposta permite reduzir o tempo de geração de modelos, buscando opções adequadas para o treinamento e dispensando tarefas complexas como a otimização por meio de buscas exaustivas de hiperparâmetros e modelos. Por outro lado, a proposta é dependente do modelo generativo, sem garantir que o modelo gerado é ótimo. Além disso, o trabalho aborda o cenário de detecção de ameaças em redes veiculares como aplicação principal, apesar de ser possível a adaptação para outros problemas de aprendizado.

4. O Sistema AutoMHS-GPT

O AutoMHS-GPT consiste em um sistema para identificação de ameaças em redes veiculares baseado em aprendizado de máquina. O sistema possui um serviço de inteligência, que utiliza inteligência artificial generativa para estimar os melhores hiperparâmetros, incluindo o modelo de aprendizado, baseado em informações sobre o conjunto de dados. As informações consideradas são os tipos das características do conjunto de dados e a tarefa de aprendizado. Assim, o modelo generativo recebe como entrada uma sentença em linguagem natural que contém as informações previstas e gera como resposta um modelo com os hiperparâmetros adequados para o problema de classificação. Portanto, a proposta aplica o modelo generativo pré-treinado como uma forma de AutoML. O tempo de resposta, porém, é menor que o necessário para obter um modelo de classificação em comparação com ferramentas como AutoKeras e Auto-Sklearn.

A execução do sistema inicia a partir da requisição de um usuário ao serviço de inteligência por meio do módulo de segurança veicular. O AutoMHS-GPT utiliza o ChatGPT como mecanismo de busca para a atividade de busca por hiperparâmetros e modelos. Logo, o sistema recebe como entrada uma sentença em linguagem natural “Eu tenho um problema de classificação que utiliza o conjunto de dados VeReMi e quero saber qual é o melhor modelo com os hiperparâmetros definidos para esta tarefa. O conjunto de dados é tabular e quero receber apenas um modelo e seus hiperparâmetros.”. Após o processamento desta frase, o sistema retorna um modelo de aprendizado de máquina com os hiperparâmetros configurados. Os hiperparâmetros ausentes após essa etapa são configurados com o valor padrão da biblioteca de aprendizado de máquina utilizada. Pelo fato do modelo generativo ser pré-treinado, a resposta é gerada em poucos segundos. A principal vantagem da proposta é a definição do modelo de aprendizado e seus hiperparâmetros antes do treinamento. Além disso, o modelo generativo é adaptável a outras tarefas de classificação ou conjunto de dados a partir da mudança da sentença

de entrada. Por outro lado, alternativas para a otimização de hiperparâmetros e modelos executam o processo de treinamento diversas vezes para determinar o melhor modelo. Apesar da busca possuir heurísticas para a redução do tempo de execução, uma mudança no problema de aprendizado requer a execução novamente de diversos modelos.

Uma limitação da implementação atual do AutoMHS-GPT é a dependência do ChatGPT. Apesar do texto utilizado como entrada e o *prompt* serem ajustáveis, o modelo generativo é utilizado como uma interface de programação de aplicações (*Application Programming Interface* - API) externa. Assim, o sistema não controla os dados utilizados para o treinamento do modelo generativo a princípio. Por outro lado, as abordagens de AutoML despendem mais tempo para procurar o modelo com os hiperparâmetros adequados a tarefa de classificação. No caso de aplicações com maior tolerância ao alto tempo de busca por modelos e uso de capacidade computacional, o AutoML é capaz de avaliar mais possibilidades sobre o conjunto de dados e encontrar um modelo melhor. Portanto, existe um compromisso em tempo de resposta e desempenho. Dependendo da aplicação, o tempo de resposta é predominante, como o caso de um ambiente dinâmico como o veicular abordado pela proposta atual.

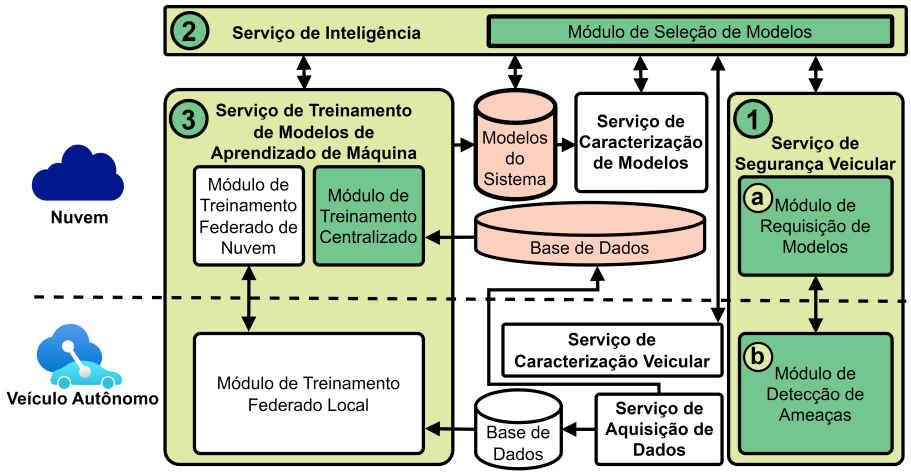

Figura 2: Arquitetura do AutoMHS-GPT. Este trabalho implementa o serviço de inteligência, serviço de segurança e o treinamento centralizado de modelos.

A Figura 2 exibe a arquitetura do sistema proposto. Os módulos e componentes brancos são funcionalidades não implementadas pelo trabalho atual. A arquitetura do sistema contém seis serviços, dos quais três são apresentados no trabalho atual. O (1) serviço de segurança veicular é responsável pela gestão dos modelos de aprendizado a serem executados no veículo. Esse serviço possui dois módulos, (a) requisição de modelos, encarregado de buscar modelos para a detecção de ameaças, e o módulo (b) responsável por executar o modelo no veículo. O módulo (a) é executado na nuvem, enquanto a detecção de ameaças executa no próprio veículo. O (2) serviço de inteligência recebe requisições por modelos e baseado nas informações recebidas busca um modelo adequado para o problema de classificação. Caso nenhum modelo seja encontrado, o módulo de seleção de modelos inicia o treinamento de um novo modelo no sistema. Este serviço aplica a inteligência artificial generativa para analisar os pedidos e retornar um modelo adequado. O (3) serviço de treinamento de modelos de aprendizado de máquina é responsável por ajustar os parâmetros dos modelos definidos pelo serviço de inteligência. Os modelos treinados são armazenados em uma base de dados de modelos para serem acessados e utilizados posteriormente. Atualmente, o sistema possui o módulo de treinamento centralizado, que pode ser utilizado para tarefa de aprendizado nas quais a informação transmitida não é sensível e pode ser armazenada na nuvem. Os demais módulos e serviços são previstos para

trabalhos futuros.

5. Desenvolvimento do Protótipo e Resultados Obtidos

Esta seção apresenta o conjunto de dados utilizado para a avaliação dos modelos criados, caracteriza o ambiente de execução e apresenta os resultados obtidos.

5.1. Descrição do Conjunto de Dados

Heijden *et al.* [Van Der Heijden et al. 2018] apresentam o *Vehicular Reference Misbehavior Dataset* (VeReMi) disponível publicamente e avaliam mecanismos de plausibilidade nos dados gerados. O conjunto de dados consiste em registros de mensagens para cada veículo na simulação e um arquivo de informações básicas que especifica o comportamento dos invasores. As mensagens são do tipo CAM. O VeReMi contém cinco ataques: posição constante, deslocamento constante, posição aleatória, deslocamento aleatório e parada eventual. Todos os ataques estão relacionados à posição do carro, enviando a mesma localização ou a posição real mais um sinal de ruído, que pode ser constante ou aleatório. O ataque de parada eventual simula um comportamento bizantino, onde o carro começa a enviar a posição real, mas de repente começa a enviar uma posição falsa constante. Kamel *et al.* [Kamel et al. 2020] estendem o conjunto de dados anterior para incluir mais dados e padrões de ataque. Assim, a extensão VeReMi contém padrões como mensagens atrasadas, DoS, repetição de dados, difusão de mensagens de veículos falsos e mau funcionamento de velocidade, além dos ataques de mau funcionamento de posição na primeira versão do conjunto de dados, descritos a seguir.

Tabela 1: Divisão de amostras conforme as classes no conjunto de dados VeReMi estendido.

Classe	Identificador	Quantidade de Amostras	Proporção de Amostras (%)
Normal	0	1900539	59,49
Posição Constante	1	43653	1,37
Deslocamento de Posição Constante	2	43567	1,36
Posição Aleatória	3	43857	1,37
Deslocamento Aleatório de Posição	4	42575	1,33
Velocidade Constante	5	41925	1,31
Alteração de Velocidade Constante	6	44359	1,39
Velocidade Aleatória	7	42258	1,32
Alteração Aleatória de Velocidade	8	42583	1,33
Parada Eventual	9	42790	1,34
Diruptivo	10	43264	1,35
Reprodução de Dados	11	44337	1,39
Mensagens Atrasadas	12	43118	1,35
Negação de Seviço (DoS)	13	131305	4,11
DoS Aleatório	14	126724	3,97
DoS Disruptivo	15	129270	4,05
Grid Sybil	16	175391	5,49
Replicação Sybil	17	44310	1,39
DoS Sybil Aleatório	18	86883	2,72
DoS Sybil Disruptivo	19	82100	2,57

O principal objetivo do atacante é causar distúrbio no ambiente veicular, seja pela indisponibilidade na comunicação ou pelo envio de dados falsos. A indisponibilidade de comunicação reduz as vantagens obtidas por sistemas de direção cooperativos, como informações das condições de trânsito em uma região. Da mesma forma, o envio de dados falsos prejudica a decisão dos veículos sobre as rotas entre origem e destino. Por exemplo, a

informação que uma via possui muitos carros, ou que a velocidade do veículo é baixa pode ser um fator decisivo para outros veículos executarem um trajeto diferente. Além disso, o atacante pode causar acidentes ao informar uma parada brusca a um veículo próximo e forçá-lo a parar desnecessariamente ou executar uma mudança de faixa.

Há 19 tipos de ataques no conjunto de dados. Posição constante (1) o veículo com comportamento malicioso reporta a mesma posição fixa apesar do seu movimento. Deslocamento de posição constante (2) o veículo adiciona um valor fixo predeterminado à sua posição atual, resultando na geração de um caminho paralelo ao caminho verdadeiro. Posição aleatória (3) o veículo envia uma posição aleatória em vez da posição real. Deslocamento aleatório de posição (4) o veículo adiciona um número aleatório à sua posição verdadeira, criando um caminho confuso. Os ataques velocidade constante (5), alteração de velocidade constante (6), velocidade aleatória (7) e alteração aleatória de velocidade (8) são respectivamente semelhantes aos quatro primeiros, a única diferença é que o atacante altera seu valor de velocidade em vez de sua localização. Parada eventual (9) o veículo envia inicialmente sua localização precisa e informações de velocidade, mas depois de um certo tempo, ele começa a reportar uma posição e velocidade zero. Disruptivo (10) o veículo reproduz informações anteriormente recebidas de vizinhos aleatórios para sobrecarregar a rede. Reprodução de dados (11) o veículo reproduz as informações recebidas de um determinado vizinho como se fossem suas.

Outros ataques afetam diretamente a comunicação, como mensagens atrasadas (12) no qual o veículo atacante envia informações precisas do seu movimento previamente registradas após um tempo pré-definido. DoS (13) o ataque consiste em inundar a rede com informações para tornar os demais veículos indisponíveis. DoS aleatório (14) o ataque é semelhante ao DoS, no entanto, todos os valores incluídos nas mensagens são aleatórios e imprecisos. DoS disruptivo (15) esse ataque combina o aumento da frequência de envio com a inundação de mensagens aleatórias recebidas anteriormente. Grid Sybil (16) a intenção do invasor é criar um congestionamento falso informando a existência de veículos fantasmas, onde pseudo-IDs de veículos são criados para veículos inexistentes em uma posição alvo específica, e o atacante mantém uma posição realista de comunicação com os veículos fantasmas. Replicação Sybil de dados (17) esse ataque é similar ao ataque 11 (Tabela 1), porém mais sofisticado. O invasor reproduz os dados de um vizinho alvo usando vários pseudo-IDs para mascarar a identidade do verdadeiro atacante. DoS Sybil aleatório (18) combina três formas de ataques em uma: aumento de frequência de mensagens enviadas, aleatoriedade de todos os valores nas mensagens e vários pseudo-IDs falsos. DoS Sybil disruptivo (19) o invasor reproduz mensagens recebidas de vizinhos anteriormente de forma aleatória com alta frequência usando muitos pseudo IDs. A Tabela 1 exibe a divisão dos dados de acordo com cada um dos ataques apresentados.

5.2. Ambiente de Testes

Um protótipo do sistema proposto foi implementado em Python v3.10.12 e testado em modelos criados com a biblioteca scikit-learn v1.2.1[2]. Além disso, a inteligência artificial generativa é implementada pelo ChatGPT [OpenAI 2023], sendo os arcabouços de AutoML AutoKeras [Jin et al. 2019] v1.1.0 e Auto-Sklearn [Feurer et al. 2015] v0.15.0 usados para comparação. Apesar da existência de outros arcabouços de AutoML, os dois selecionados destacam-se por serem de código aberto e facilidade de utilização. Os experimentos foram realizados em um servidor Intel Xeon E5-2650 CPU 2.00 GHz com 32 núcleos de processamento e 504 GB de RAM. Por fim, os resultados obtidos são demonstrados com intervalos de confiança

[2]O scikit-learn [Pedregosa et al. 2011] é uma biblioteca de código aberto, bem documentada, para a criação de modelos de aprendizado de máquina, que possui inúmeros desenvolvedores. Disponível em https://scikit-learn.org/

Tabela 2: Parâmetros utilizados para os arcabouços de AutoML.

Arcabouço	Parâmetro	Valor
AutoKeras	Modelo	StructuredDataClassifier
	Total de Tentativas	30
	Épocas de Treinamento	100
Auto-Sklearn	Modelo	AutoSklearnClassifier
	Tempo Limite	1725 minutos
	Tempo Limite por Tentativa	573 minutos

de 95%. O conjunto de dados foi particionado com 80% dos dados para treino e 20% para teste. O conjunto de dados é desbalanceado, sendo a classe normal majoritária. Essa configuração foi mantida nos experimentos, sendo o único pré-processamento realizado a eliminação de características altamente correlacionadas a partir da correlação de Pearson.

O AutoKeras e o Auto-Sklearn necessitam da definição de alguns parâmetros antes de iniciar o processo de busca pelos modelos e hiperparâmetros. Assim, a Tabela 2 exibe a configuração dos parâmetros utilizados para os arcabouços de AutoML. O AutoKeras necessita do número de épocas para o treinamento de cada modelo e o número total de modelos a serem avaliados. O Auto-Sklearn requer o tempo máximo para execução de cada tarefa de busca e o tempo total de busca. Além disso, ambas as propostas possuem um modelo utilizado para a otimização de hiperparâmetros. A escolha dos valores desses parâmetros do AutoML é discutida na comparação de tempo de execução de cada abordagem. Por outro lado, o AutoMHS-GPT utiliza o ChatGPT como mecanismo de busca para a mesma atividade. Após o processamento do texto em linguagem natural, o sistema retornou um modelo de floresta aleatória, cujos hiperparâmetros são 150 árvores, duas amostras como número mínimo para dividir e uma amostra no mínimo por folha. Os demais hiperparâmetros permaneceram com a configuração padrão. Como o modelo generativo é pré-treinado, a resposta é gerada em poucos segundos. A seguir são apresentados os resultados de desempenho de classificação utilizando as diferentes abordagens.

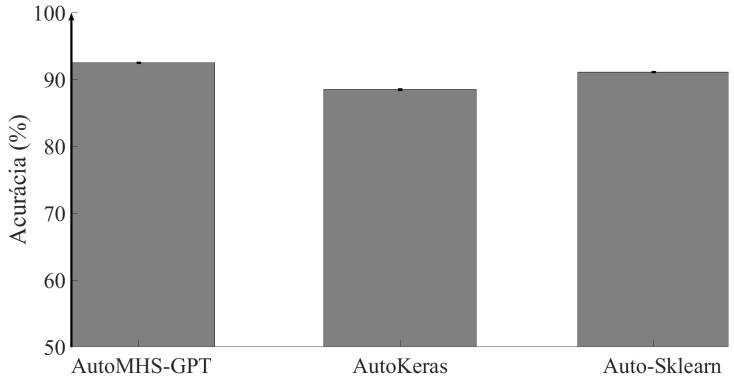

Figura 3: Avaliação de acurácia das três propostas.

5.3. Comparação entre o AutoMHS-GPT e o Estado da Arte

Essa seção apresenta os experimentos utilizados para avaliar e comparar o AutoMHS-GPT com o estado da arte. A primeira parte avalia o desempenho de classificação, enquanto a segunda parte experimental avalia o tempo necessário para execução das propostas até a instanciação dos modelos.

5.4. Avaliação do Desempenho de Classificação

O primeiro experimento realizado avalia o desempenho de classificação das diferentes propostas. Como o conjunto de dados possui 20 classes distintas, é possível utilizar duas abordagens, a classificação binária ou de múltiplas classes. A classificação binária consiste em determinar se a mensagem transmitida faz parte de um ataque ou é uma comunicação normal. Entretanto, esse tipo de classificação reduz a granularidade no processo de instanciação de contramedidas, pois o sistema de detecção apenas sabe a existência do ataque. Portanto, os resultados consideram a classificação de múltiplas classes para reação específica a cada tipo de ataque posteriormente. A Figura 3 exibe a acurácia das três abordagens avaliadas sobre o conjunto de dados de ataque em redes veiculares. O AutoMHS-GPT classifica mais amostras corretamente do que as demais propostas, apresentando o desempenho 1,35% superior em relação ao modelo gerado pelo Auto-Sklearn e 3,99% superior em relação ao AutoKeras.

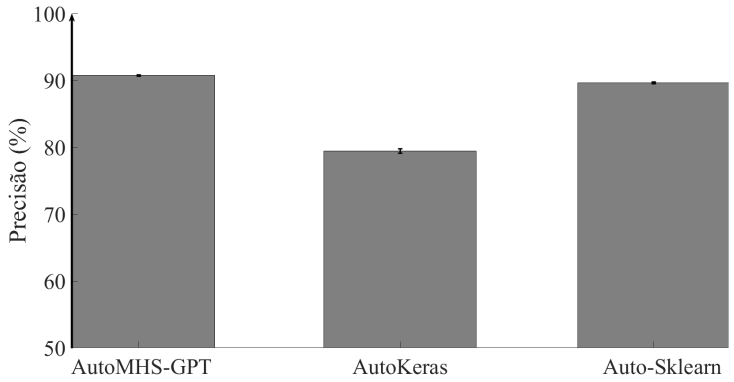

Figura 4: Avaliação de precisão das três propostas.

O mesmo comportamento pode ser observado em relação à precisão, onde o modelo gerado pela proposta atual apresenta a precisão média igual 90,74%, enquanto os demais geraram modelos com 79.44% e 89.65%, para o AutoKeras e Auto-Sklearn, respectivamente. A Figura 4 apresenta esse resultado. Isto significa que o modelo gerado produz menos falsos positivos.

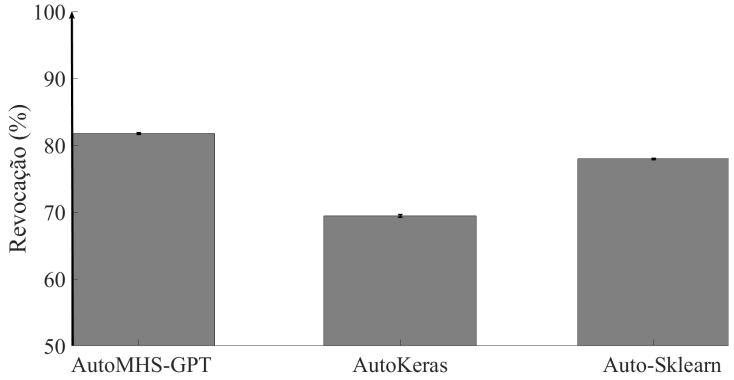

Figura 5: Avaliação de revocação das três propostas

Além disso, o AutoMHS-GPT aumenta a capacidade de detectar ataques, pois possui uma revocação maior do que as demais propostas. O resultado é exibido na Figura 5. Por fim, a F1 Score do sistema proposto também é maior do que as demais, uma vez que tanto a precisão quanto a revocação são maiores do que as dos arcabouços comparados, como exibido na Figura 6.

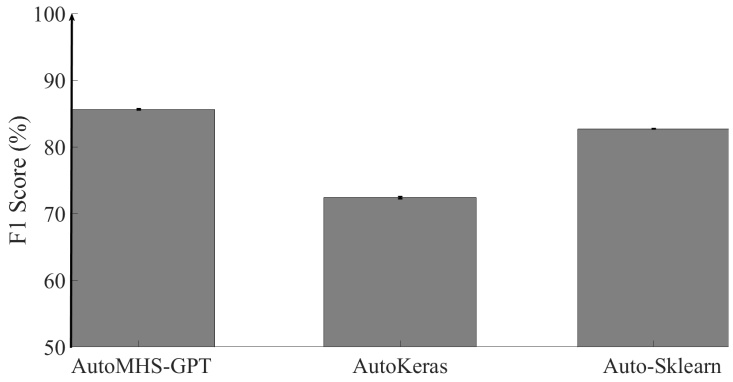

Figura 6: Avaliação de F1 score das três propostas.

A inteligência artificial generativa utilizada como mecanismo de busca por modelo e hiperparâmetro no cenário de detecção de ameaças em redes veiculares possui um desempenho superior ao AutoML. Outra métrica de avaliação importante para o problema atual é o tempo de execução. Assim, a segunda parte do experimento consiste em avaliar essa métrica para cada abordagem.

5.5. Desempenho do Treinamento do Modelo

O tempo de execução compreende o tempo de definição de modelo, hiperparâmetros e treinamento. A Figura 7 exibe o resultado desse experimento. O AutoMHS-GPT possui um tempo de execução aproximadamente duas ordens de grandeza menor do que as demais ferramentas de AutoML. Isso ocorre, pois o processo de busca por hiperparâmetros e modelo é realizado pelo modelo pré-treinado, obtendo o resultado em alguns segundos. Dessa forma, o principal fator no tempo de execução da proposta é o treinamento do modelo resultante da proposta para classificação de ameaças.

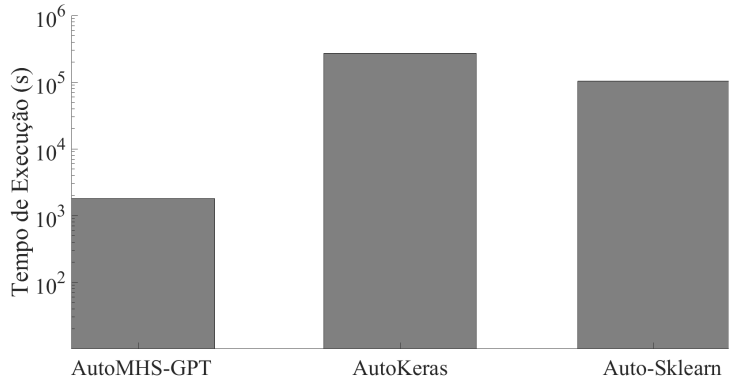

Figura 7: Tempo de execução das abordagens.

O tempo de execução das propostas de AutoML são dependentes dos parâmetros exibidos na Tabela 2. A redução do número de épocas no caso do AutoKeras gera modelos com menor desempenho de classificação. O Auto-Sklearn é mais sensível ao ajuste do tempo limite por tentativa. Um tempo limite muito restrito torna o arcabouço incapaz de treinar modelos, gerando apenas um classificador burro (*DummyClassifier*), que atribui a todas as amostras a classe majoritária no conjunto de dados. No caso do conjunto de dados avaliado, todas as amostras são classificadas como normal por este classificador. Além disso, o tamanho do conjunto de dados

influencia diretamente no tempo de execução de todas as abordagens. Portanto, para conjuntos de dados menores, a diferença pode ser reduzida. No cenário de segurança em redes veiculares, é essencial gerar um modelo no menor tempo possível. A ausência de um modelo atualizado torna o veículo vulnerável e susceptível aos ataques discutidos anteriormente, colocando a integridade física de seus passageiros em risco. Por fim, o experimento demonstra que a nossa proposta reduz o tempo de implantação do modelo para aproximadamente 30 minutos, enquanto os arcabouços de AutoML necessitam de aproximadamente 3 dias para a mesma tarefa.

6. Conclusão e Trabalhos Futuros

Este trabalho apresentou o AutoMSH-GPT, um sistema para automatizar a decisão de qual modelo e hiperparâmetros devem ser utilizados a partir de informações do conjunto de dados. A automatização da seleção de modelos é realizada por meio da inteligência artificial generativa a partir de dados sobre o problema de classificação. Os resultados mostram o alto desempenho de classificação do modelo gerado, além da redução no tempo de definição do modelo e seus hiperparâmetros para o treinamento, em comparação com as abordagens de AutoML. Como trabalhos futuros, pretende-se implementar uma arquitetura baseada em transformadores que seja específica para a tarefa de otimização de hiperparâmetro de modelos e adicionar novos módulos ao sistema proposto. Além disso, treinar os modelos de detecção por meio de aprendizado federado a fim de preservar a privacidade dos usuários.

Referências

Bisong, E. e Bisong, E. (2019). Google AutoML: Cloud Vision. *Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners*, páginas 581–598.

Bousalem, B. et al. (2023). DDoS Attacks Mitigation in 5G-V2X Networks: A Reinforcement Learning-Based Approach. Em *International Conference on Network and Service Management (CNSM)*, páginas 1–5. IEEE.

Choi, S., Kim, J. e Yeo, H. (2021). TrajGAIL: Generating Urban Vehicle Trajectories using Generative Adversarial Imitation Learning. *Transportation Research Part C: Emerging Technologies*, 128:103091.

Chougule, A., Agrawal, K. e Chamola, V. (2023). SCAN-GAN: Generative Adversarial Network Based Synthetic Data Generation Technique for Controller Area Network. *Internet of Things Magazine*, 6(3):126–130.

Cobilean, V. et al. (2023). Anomaly Detection for In-Vehicle Communication Using Transformers. Em *Industrial Electronics Society (IECON)*, páginas 1–6. IEEE.

Du, H. et al. (2023). Spear or Shield: Leveraging Generative AI to Tackle Security Threats of Intelligent Network Services. *arXiv preprint arXiv:2306.02384*.

Farsi, M., Ratcliff, K. e Barbosa, M. (1999). An Overview of Controller Area Network. *Computing & Control Engineering Journal*, 10(3):113–120.

Feurer, M. et al. (2015). Efficient and Robust Automated Machine Learning. *Advances in Neural Information Processing Systems (NIPS)*, 28.

Gupta, M., Akiri, C., Aryal, K., Parker, E. e Praharaj, L. (2023). From ChatGPT to ThreatGPT: Impact of Generative AI in Cybersecurity and Privacy. *arXiv preprint arXiv:2307.00691*.

Horváth, T. et al. (2023). Hyper-Parameter Initialization of Classification Algorithms using Dynamic Time Warping: A Perspective on PCA Meta-Features. *Applied Soft Computing*.

Jacobs, A. S. et al. (2021). Hey, LUMI! Using Natural Language for Intent-Based Network Management. Em *Annual Technical Conference (ATC)*, páginas 625–639. USENIX.

Jin, H., Song, Q. e Hu, X. (2019). Auto-Keras: An Efficient Neural Architecture Search System. Em *International Conference on Knowledge Discovery & Data Mining (SIGKDD)*, páginas 1946–1956. ACM.

Jüttner, V., Grimmer, M. e Buchmann, E. (2023). ChatIDS: Explainable Cybersecurity Using Generative AI. *arXiv preprint arXiv:2306.14504*.

Kamel, J., Wolf, M., Van Der Hei, R. W., Kaiser, A., Urien, P. e Kargl, F. (2020). VeReMi Extension: A Dataset for Comparable Evaluation of Misbehavior Detection in VANETs. Em *International Conference on Communications (ICC)*, páginas 1–6. IEEE.

Kumar, S. et al. (2021). Exploring the Limits of Concurrency in ML Training on Google TPUs. *Proceedings of Machine Learning and Systems*, 3:81–92.

LeDell, E. e Poirier, S. (2020). H2O AutoML: Scalable Automatic Machine Learning. Em *AutoML Workshop (ICML)*. International Machine Learning Society.

Mu, T. et al. (2022). Auto-CASH: A Meta-Learning Embedding Approach for Autonomous Classification Algorithm Selection. *Information Sciences*, 591:344–364.

Neto, H. N. C., Dusparic, I., Mattos, D. M. e Fernande, N. C. (2022). FedSA: Accelerating Intrusion Detection in Collaborative Environments with Federated Simulated Annealing. Em *International Conference on Network Softwarization (NetSoft)*, páginas 420–428. IEEE.

OpenAI (2023). ChatGPT: Optimizing Language Models for Dialogue. Available at: `https://openai.com/blog/chatgpt/`. Último acesso: 28 de janeiro de 2024.

Paleyes, A., Urma, R.-G. e Lawrence, N. D. (2022). Challenges in Deploying Machine Learning: a Survey of Case Studies. *Computing Surveys*, 55(6):1–29.

Pedregosa, F. et al. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Thornton, C. et al. (2013). Auto-WEKA: Combined Selection and Hyperparameter Optimization of Classification Algorithms. Em *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, páginas 847–855. ACM.

Van Der Heijden, R. W., Lukaseder, T. e Kargl, F. (2018). VeReMi: A Dataset for Comparable Evaluation of Misbehavior Detection in VANETs. Em *Security and Privacy in Communication Networks (SecureComm)*, páginas 318–337. Springer.

Vinita, L. J. e Vetriselvi, V. (2023). Federated Learning-based Misbehaviour Detection on an Emergency Message Dissemination Scenario for the 6G-enabled Internet of Vehicles. *Ad Hoc Networks*, 144:103153.

Yakan, H., Fajjari, I., Aitsaadi, N. e Adjih, C. (2023). Federated Learning for V2X Misbehavior Detection System in 5G Edge Networks. Em *Conference on Modeling Analysis and Simulation of Wireless and Mobile Systems*, páginas 155–163. ACM.

Zhang, R. et al. (2023). Generative AI-enabled Vehicular Networks: Fundamentals, Framework, and Case Study. *arXiv preprint arXiv:2304.11098*.

Zhao, Q., Chen, M., Gu, Z., Luan, S., Zeng, H. e Chakrabory, S. (2022). CAN Bus Intrusion Detection Based on Auxiliary Classifier GAN and Out-of-distribution Detection. *Transactions on Embedded Computing Systems (TECS)*, 21(4):1–30.