Avaliação da Predição do Tempo de Vida do Enlace no Processo de *Offloading* Computacional em VANETs

Paulo H. G. Rocha[1], Alisson B. de Souza[1], José G. R. Maia[2], César L. C. Mattos[1], Francisco A. Silva[3] e Paulo A. L. Rego[1]

[1]Departamento de Computação – Universidade Federal do Ceará (UFC)
Fortaleza – CE – Brasil

[2]Instituto UFC Virtual – Fortaleza – CE – Brasil

[3]Universidade Federal de Piauí (UFPI) – Picos – PI – Brasil

paulorocha@great.ufc.br,{alisson,gilvanm}@ufc.br,

{cesarlincoln,paulo}@dc.ufc.br,faps@ufpi.edu.br

Abstract. *Vehicular networks (VANETs) enable intelligent applications in urban mobility scenarios. However, the communication time (link lifetime — LLT) between nodes is generally short due to the dynamism of vehicular mobile scenarios, which can affect applications and processes in VANETs, such as computational offloading. Thus, it is critical to obtain a reasonable estimate of the TVE between vehicles to improve the decision of when and to which vehicle to offload. Our work investigates different machine learning (ML) algorithms to evaluate the feasibility of predicting TVE in Road and Urban scenarios. We trained several ML models, and the results show that ML techniques based on SVR (Support Vector Regression) effectively reduce the task loss rate by up to 5% in the computational offloading process.*

Resumo. *As redes veiculares (VANETs) possibilitam aplicações inteligentes em cenários de mobilidade urbana. No entanto, o tempo de comunicação (tempo de vida do enlace — TVE) entre os nós é geralmente curto devido ao dinamismo dos cenários móveis veiculares, o que pode afetar aplicações e processos em VANETs, como o offloading computacional. Assim, é fundamental obter uma boa estimativa do TVE entre os veículos para melhorar a decisão de quando e para qual dispositivo fazer offloading. Este trabalho investiga diferentes algoritmos de aprendizado de máquina (do inglês, Machine Learning - ML) para avaliar a viabilidade de prever o TVE em cenários Rodoviários e Urbanos. Vários modelos de ML foram treinados e os resultados mostram que as técnicas de ML baseadas em SVR (Support Vector Regression) são efetivas, chegando a reduzir a taxa de perda de tarefas em 5% no processo de offloading computacional.*

1. Introdução

As redes veiculares (VANETs) desempenham um papel essencial em ITS (do inglês, *Intelligent Transportation Systems*), oferecendo diversos serviços e aplicativos que possibilitam sua implementação [Salem et al. 2022]. Graças ao constante desenvolvimento do campo das VANETs, houve o surgimento de novas aplicações complexas, como direção autônoma, realidade aumentada, *streaming* de vídeo em veículos, navegação

automática, monitoramento em tempo real, entre outras [Alabbasi and Aggarwal 2020]. Tais aplicações representam um desafio à limitada quantidade de recursos de um veículo, dado que várias delas demandam poder computacional e são sensíveis à latência, o que tornam o cenário desafiador [He et al. 2021]. Uma solução para melhorar o processamento das aplicações em VANETs é utilizar a técnica de *offloading* computacional para aproveitar os veículos próximos e seus recursos ociosos, e transferir tarefas ou aplicativos inteiros para serem executados nesses dispositivos [Li et al. 2018].

Nesse contexto, dois fatores essenciais no *offloading* computacional, *atraso* e *custo*, dificilmente são bem tratados devido às frequentes falhas de comunicação causadas pela alta mobilidade dos veículos [Lee and Atkison 2021]. A literatura recente considera a predição do tempo de vida do enlace (TVE) entre dois veículos uma tecnologia habilitadora para melhorar a qualidade da comunicação em VANETs [Khatri et al. 2021]. As soluções existentes não preveem o TVE de forma eficiente ou não são flexíveis o suficiente para lidar com possíveis alterações de atributos no processo de predição - o que pode ocorrer ao lidar com novos cenários [Deng et al. 2020][de Souza and Villas 2020].

Este trabalho investiga o uso de técnicas de predição baseadas em regressão para estimar o TVE entre nós em VANETs. A técnica de simulação foi utilizada para gerar *datasets* com diferentes parâmetros que podem impactar o TVE entre veículos em cenários Urbano e de Rodovia. Os *datasets* gerados foram então utilizados para treinar e avaliar diferentes algoritmos de ML da literatura na tarefa de estimar o TVE. Posteriormente, o algoritmo de ML com os melhores resultados foi utilizado como função de predição no processo de decisão de um algoritmo de *offloading*, sendo realizada uma avaliação de desempenho dessa abordagem com outros algoritmos de *offloading* da literatura.

As principais contribuições do presente trabalho são: (i) um método baseado em simulação de cenários Urbanos e de Rodovias para gerar *datasets* voltado para a pesquisa da predição do TVE e a proposição de um conjunto de atributos (*features*) de interesse. Os *datasets* estão disponíveis para a comunidade[1]; (ii) a comparação de algoritmos de ML baseados em regressão para avaliar a solução mais eficiente na predição do TVE em VANETs; e (iii) a extensão de um algoritmo de decisão de *offloading* proposto em um trabalho anterior [de Souza et al. 2020] para usufruir da capacidade de predição do TVE e aprimorar as decisões de *offloading* de tarefas.

O restante deste trabalho está organizado da seguinte forma. A Seção 2 aborda os trabalhos relacionados. A Seção 3 apresenta a proposta e introduz as ferramentas e métodos utilizados. Na Seção 4, são discutidos os resultados dos experimentos. A Seção 5 apresenta as principais conclusões e direcionamentos para trabalhos futuros.

2. Trabalhos Relacionados

Esta seção apresenta trabalhos que têm utilizado diferentes algoritmos para prever a localização de objetos móveis (e.g., veículos), bem como estimar o TVE entre dois nós. A Tabela 1 apresenta um resumo das contribuições dos trabalhos relacionados e os compara sob quatro aspectos: atributos utilizados na predição, cenário de experimentação, uso de ML na predição e se o trabalha lida com *offloading* computacional.

O primeiro aspecto refere-se aos atributos utilizados na predição. A alta mobi-

[1]https://github.com/PauloHGR/datasetsSBRC-2022

lidade do cenário veicular exige que fatores específicos sejam observados, dentre eles a distância, direção e velocidade, que são os atributos mais comuns. O segundo aspecto é o cenário utilizado na simulação, onde destacam-se cenários Rodoviários e Urbanos. O terceiro aspecto diz respeito ao uso de técnicas de ML para a predição, onde destacam-se [Zhang et al. 2017] e este trabalho. Por fim, o quarto aspecto identifica os trabalhos no contexto de *offloading* computacional.

Trabalho	**Atributos da Predição**	**Cenário**	**Usou ML na Predição?**	**Fez *Offloading*?**	**Contribuição**
[Ye et al. 2019]	Distância, Direção e Velocidade	Urbano	Não	Não	Novo protocolo chamado MPBRP (*Mobility Prediction Based Routing Protocol*) que busca prever a mobilidade dos nós, através das posições e ângulos para encontrar a melhor rota.
[Kulla et al. 2019]	Velocidade Relativa e Posição	Rodovia	Não	Não	Um método que prevê o TVE entre dois veículos em movimento com base em sua velocidade relativa.
[Nabil et al. 2019]	Posição, Velocidade e Aceleração	Rodovia	Não	Não	São propostos dois protocolos, um para prever a rota mais estável e outro para prever o tempo de entrega dos pacotes antes de enviar os dados. Foi desenvolvido um esquema de predição do TVE, visando garantir a eficiência dos protocolos.
[Moura et al. 2019]	Distância	Urbano	Não	Sim	É proposta uma solução distribuída para realizar o *offloading* dos dados em redes de sensores veiculares, através de um algoritmo guloso que determina um sub-conjunto de veículos transmissores.
[de Souza et al. 2020]	Posição, Velocidade e Aceleração	Rodovia	Não	Sim	Um esquema para melhorar o desempenho de *offloading* em redes multi-interface, através de um algoritmo de escolha gulosa chamado GCF.
[Zhang et al. 2017]	Tempo de Vida do Enlace, SLS, Função de Similaridade, Mobilidade Relativa, Distância e Velocidade	Rodovia	Sim	Não	Implementa um método de predição do TVE em ambientes de VANETs usando o algoritmo de Aprendizado de Máquina Adaboost. Um *benchmark* entre o Adaboost e outros métodos de ML é realizado.
Este trabalho	Distância, Ângulo, Pseudo-ângulo,Velocidade, Velocidade Relativa, Aceleração Relativa e SLS	Rodovia e Urbano	Sim	Sim	Propõe um esquema de predição do TVE através do algoritmo de ML SVR, escolhido através de uma avaliação de desempenho entre outros métodos de ML. Além disso, estende o algoritmo GCF com a nova função de predição.

Tabela 1. Comparação com os Trabalhos Relacionados.

Ao contrário dos trabalhos mencionados, este avalia diferentes técnicas de ML para a predição do TVE e a posterior inclusão dessa técnica em um algoritmo de *offloading* computacional em VANETs. São utilizados oito atributos para lidar com o problema de predição do TVE em VANETs, dois deles propostos e utilizados exclusivamente neste trabalho, como o pseudo-ângulo e SLS (*Spatial Locality Similarity*). Outra diferença relevante é o fato deste trabalho considerar cenários Urbano e de Rodovia ao avaliar a eficiência da predição e do *offloading*.

3. Metodologia

Esta seção apresenta a metodologia utilizada para aplicar técnicas de ML na predição do TVE. A Figura 1 ilustra todas as etapas realizadas neste trabalho.

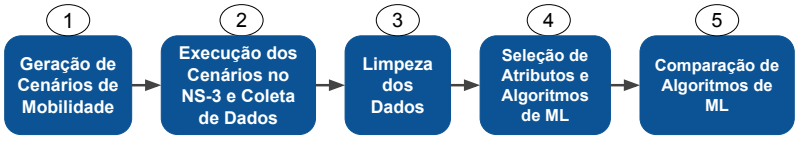


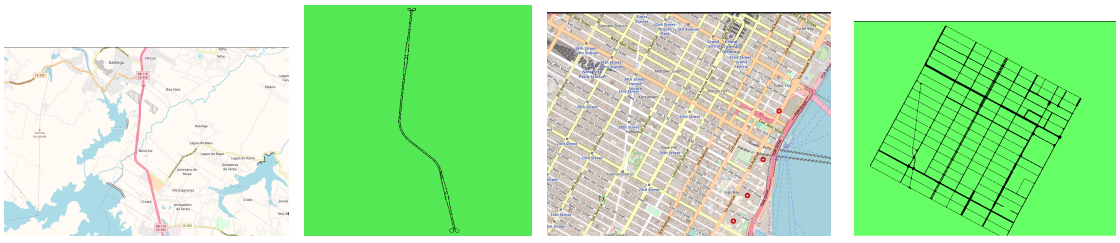
Figura 1. Fluxo de Trabalho da Metodologia Proposta.

O processo começa com a implementação de simulações com diferentes cenários de VANETs para coletar os atributos necessários para o processo de aprendizagem. Após esta etapa, é coletado e calculado um conjunto de atributos de comunicação entre os veículos. Cada cenário gera um *dataset* que passa por um processo de limpeza para torná-lo adequado ao treinamento. Técnicas de ML são então aplicadas aos *datasets* finais para prever o valor do TVE e, então, são comparados utilizando diferentes métricas.

3.1. Geração de Cenários

Uma parte importante do processo de aprendizagem é identificar os fatores que afetam uma característica que se deseja medir. Visando coletar esses fatores, também chamados atributos, foram construídos dois cenários de rede veicular: um de Rodovia e um Urbano. Ambos os cenários são amplamente utilizados em aplicações de rede veicular [Hussain et al. 2019, Sun et al. 2020].

Foi escolhido um trecho de 5 quilômetros da BR-116 na região metropolitana de Fortaleza, Brasil (Figura 2(a)) para o cenário de Rodovia. O cenário Urbano consiste em um trecho de 2 quilômetros quadrados do centro de Manhattan, Nova York, EUA (Figura 2(c))[Kim et al. 2020, Eckermann et al. 2019]. O SUMO[2] e o ns-3[3] foram utilizados para executar os cenários simulados e realizar a etapa de coleta de dados. O SUMO é uma ferramenta de geração de mapas e movimentações de veículos ou pedestres. O ns-3 é um simulador de rede consolidado que permite a utilização dos cenários gerados pelo SUMO. As Figuras 2(b) e 2(d) ilustram as representações no SUMO para os cenários considerados.



(a) Mapa da Rodovia. (b) Cenário de Rodovia no SUMO. (c) Mapa Urbano. (d) Cenário Urbano no SUMO.

Figura 2. Representação Gráfica dos Cenários.

O tempo da simulação foi definido em 85 segundos. Para mitigar interferências e colisões de pacotes, o intervalo de transmissão do pacote é definido em três valores: 0,5, 1 e 2 segundos. Esses valores são distribuídos de acordo com *id* do veículo. O alcance máximo de transmissão dos veículos foi configurado em 250 metros, através do modelo de rádio propagação *Two-Ray Ground*. Foi utilizada uma taxa de transferência de até 27 Mbps sobre uma largura de banda de canal de 10MHz permitida pelo módulo 802.11p do ns-3. O número de veículos em cada cenário varia devido às diferenças em suas topologias. Foram utilizados 100 veículos no cenário de Rodovia e 200 veículos no cenário Urbano para diminuir os efeitos da dispersão de veículos no cenário. As viagens de cada veículo dentro destes cenários são geradas automaticamente pelo SUMO, respeitando as

[2]https://www.eclipse.org/sumo/
[3]https://www.nsnam.org/

regras das vias e trajetos do mapa. Dessa forma, os parâmetros de configuração de cada viagem (como velocidades e posições dos veículos) são armazenados em um *trace* de mobilidade.

3.2. Coleta de Dados e Definição de Atributos

Com os cenários definidos e configurados no simulador, o próximo passo é a execução da simulação e coleta de dados. O simulador ns-3 permite coletar atributos relacionados à velocidade (V_x, V_y), à posição no plano (P_x, P_y) e à distância entre dois veículos. A partir das coordenadas do vetor velocidade pode-se obter a velocidade escalar como $Speed = \sqrt{V_x^2 + V_y^2}$. As coordenadas x e y dos vetores de posição e de velocidade são essenciais para obter atributos adicionais como, por exemplo, ângulos, velocidade relativa e aceleração relativa.

Os veículos fazem parte de uma rede de comunicação V2V (Veículo-para-Veículo), onde cada veículo solicita e inicia uma comunicação com veículos vizinhos por meio de *broadcast*, com pacotes de 1024 *bytes*. Quando a comunicação é estabelecida, os veículos permanecem enviando/recebendo pacotes até que sua comunicação encerre. Durante o tempo de comunicação, o carro transmissor i (cliente) envia pacotes com suas informações de velocidade e posição dentro de um determinado intervalo de tempo. Ao receber os pacotes, o carro receptor j (servidor) calcula as suas informações de velocidade e posição, além de outros atributos discutidos posteriormente. Por fim, esses dados são armazenados em um arquivo externo, que serve como um *dataset*.

Os trabalhos supracitados se concentram em um conjunto menor de atributos para prever o TVE, conforme resumido na Tabela 1. Neste trabalho, oito variáveis independentes são utilizadas no processo de predição. São atributos consolidados em diversos trabalhos na literatura e representam valores que podemos coletar em um ambiente real de trânsito. A distância D_{ij} entre veículos é obtida diretamente do ns-3: $D_{ij} = \sqrt{(P_{xi} - P_{xj})^2 + (P_{yi} - P_{yj})^2}$.

A partir do vetor velocidade, é possível calcular os ângulos Θ_i e Θ_j relativos aos veículos transmissores e receptores, através da equação : $\Theta = \arctan\left(\frac{V_y}{V_x}\right)\frac{180}{\pi}$. Também é considerado o pseudo-ângulo entre os dois veículos, utilizado quando é preciso comparar ângulos diferentes, mas sem grandes custos de desempenho [Nešović 2017]. O pseudo-ângulo é utilizado para determinar a posição do veículo i em relação à posição do veículo j. Ele fornece um número no intervalo $[0, 8)$, que indica a posição de j no plano. Conclui-se que o intervalo $[0, 4]$ indica que j está afastando-se de i, enquanto o intervalo $[4, 8)$ indica que j está aproximando-se de i. A Figura 3 ilustra como o pseudo-ângulo se comporta no cenário da Rodovia.

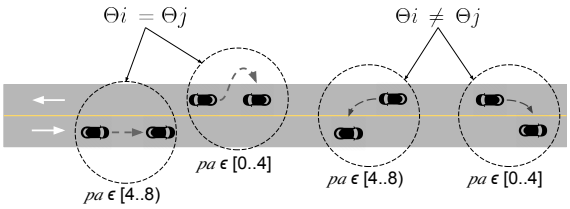


Figura 3. Representação do pseudo-ângulo para o cenário de Rodovia.

A soma do quadrado das velocidades de i e j é outro atributo considerado neste trabalho. O objetivo é dar mais peso às velocidades mais altas ou às maiores diferenças de velocidade entre os carros, já que, no ambiente de simulação, as velocidades podem ser semelhantes. O atributo é calculado por $S_v = Speed_i^2 + Speed_j^2$.

A relação entre velocidade e distância é feita pelo cálculo da Semelhança de Localidade Espacial (SLS), que usa a distância e a diferença de velocidade entre os veículos para indicar o comprimento do enlace. O SLS foi considerado um atributo importante no trabalho de [Zhang et al. 2017]. Ao calcular o SLS, deve-se definir a velocidade máxima e o alcance máximo de transmissão. Esses valores são definidos respectivamente por $S_{max} = 17$ m/s e $r = 250$ metros. O atributo é então calculado por:

$$SLS = \left(1 + \frac{D_{ij}^2}{r^2} + \left(Speed_i - \frac{Speed_j}{2} S_{max}\right)^2\right)^{-1/2}. \quad (1)$$

Também são utilizados outros dois atributos mencionados em [Nabil et al. 2019] e [Härri et al. 2008], a velocidade relativa (RS) e a aceleração relativa (RA) do carro cliente em relação ao carro vizinho. Essas variáveis são dadas por:

$$RS = (V_{xi} - V_{xj})^2 + (V_{yi} - V_{yj})^2, \quad (2)$$

$$RA = 2[(P_{xi} - P_{xj})(V_{xi} - V_{xj}) + (P_{yi} - P_{yj})(V_{yi} - V_{yj})]. \quad (3)$$

A variável alvo será o tempo de comunicação Tc entre o veículo i e o veículo j. A variável alvo, ou variável dependente, é o valor que deverá ser predito usando técnicas de ML. O valor de Tc para dois veículos durante a simulação é determinado ao contar os registros armazenados pelo carro j de cada pacote entregue pelo carro i. Cada veículo possui um identificador $IdCar$, armazenado com os demais atributos. Outra informação capturada da simulação é o instante t em que ocorre a comunicação entre o veículo i e j. Com os valores de $IdCar$ dos carros i e j, é percorrido todo o *dataset* até chegar ao último registro de ambos os carros. Nesse ponto é identificado o instante t. O Algoritmo 1 resume e formaliza esse procedimento.

O instante t é quando o carro vizinho (j) recebe o pacote do carro transmissor da simulação (i). O valor t é armazenado de forma redundante em duas variáveis chamadas t_i e t_f. O algoritmo busca linha por linha (l) em cada *dataset* os registros de comunicação de carros com $IdCar_i$ e $IdCar_j$. O primeiro registro é armazenado em r pelo algoritmo, e a variável c recebe o valor 1, que indica para as iterações subsequentes que o primeiro registro está em r. A variável $TransTime$ refere-se ao tempo de transmissão do pacote. É comparada a diferença entre o registro atual e o anterior com o tempo de transmissão do pacote. Se essa diferença de registro for menor que o tempo de transmissão, os carros estão em comunicação. Então, o t_f do primeiro registro é atualizado para receber o t_i do registro atual. Se a diferença de tempo nos registros for maior que o tempo de transmissão do pacote, houve queda de enlace e o valor de t_f é atualizado no *dataset*. Por fim, quando não forem encontrados mais carros com os identificadores especificados, as comparações terminam e $Tc = r[t_f]$ é executado.

3.3. Processo de Limpeza de Dados

Antes de treinar os modelos de ML, foi realizado um pré-processamento dos *datasets* para eliminar dados ruidosos e discrepantes. Quando perdas de comunicação ocorrem entre

Algorithm 1: Processamento do Tempo de Comunicação.

Input: $IdCar_i$, $IdCar_j$, {dataset}, $TransTime$
Output: {dataset} + Tc
$c \leftarrow 0$;
$r \leftarrow []$;
foreach *l in* {*dataset*} **do**
  if $\{IdCar_i, IdCar_j\} \subset l$ **then**
    if c = *0* **then**
      $r \leftarrow l$;
      $c \leftarrow 1$;
    else
      if $(l[t_i] - r[t_i]) < TransTime$ **then**
        $r[t_f] \leftarrow l[t_i]$;
      else
        $\{dataset\} \leftarrow r[t_f]$;
        $r \leftarrow l$;
      end
    end
  end
end
if $r \neq \emptyset$ **then**
  $\{dataset\} \leftarrow r[t_f]$;
end

o cliente e os veículos vizinhos, pode haver um restabelecimento da comunicação em seguida. Assim, é considerada uma janela mínima de 1 segundo durante a comunicação ociosa entre os veículos para armazenar o registro no *dataset*. Tal estratégia elimina quebras de conexão curtas, que podem comprometer o treinamento ao gerar *datasets* com muitas conexões breves.

Durante a simulação, é comum que veículos parados estabeleçam comunicação entre si. Nesse caso, amostras desses veículos foram eliminadas, já que veículos parados geram grandes tempos de comunicação, o que poderia ocasionar algum viés ou dados ruidosos. Os veículos parados acabam também gerando muitas amostras, desbalanceando os dados. Por fim as amostras dos veículos mais lentos são semelhantes a dos veículos parados, o que generaliza a predição para os veículos parados, não atrapalhando o processo de predição. Por consistência, também foram removidas amostras coletadas com distâncias superiores a 250 metros, que é o alcance máximo configurado.

3.4. Seleção dos Algoritmos de ML

Após construir e limpar os conjuntos de dados, os algoritmos de ML a serem avaliados na solução são selecionados. O objetivo é identificar a melhor abordagem de aprendizado para prever o TVE para cenários variados. São considerados métodos de regressão para prever o tempo de comunicação com base nas variáveis independentes previamente detalhadas. As seguintes técnicas são avaliadas: ϵ-SVR, KNN (*k-Nearest Neighbors*), *Random Forest*, XGBoost e Rede Neural MLP (*Multilayer Perceptron*). A Tabela 2 mostra os parâmetros usados na configuração de cada técnica de ML.

O método *Grid Search* é utilizado para selecionar a combinação de hiperparâmetros com a menor taxa de erro de predição. Na rede neural, é utilizada a biblioteca *Keras*[4] e algumas combinações de hiperparâmetros fornecidos pelo *Keras* são testadas, com o melhor resultado mostrado na Tabela 2. Para realizar a medição das métricas, foi

[4]https://keras.io/

utilizada uma máquina virtual com uma *CPU Intel(R) Xeon(R) E5-2650 v3 @2.30GHz*, 1 núcleo de *CPU*, 1 GB de *RAM* e sistema operacional *Linux Ubuntu 20.04.2 LTS*, escolhida para simular um dispositivo com restrição de recursos, i.e., simulando um veículo.

Algoritmo	**Hiperparâmetros**
Rede Neural	loss_function: mae; optimizer: adam, metrics: mse,mae; epochs: 1500
SVR	C:8, epsilon:0.05, gamma:1.0, kernel:rbf
KNN	n_neighbors:13; weights:distance; n_estimators:500
Random Forest	max_depth:100; max_features:3; min_samples_leaf:2; min_samples_split:8; n_estimators:500
XGBoost	colsample_bytree: 0.8; learning_rate: 0.1; max_depth: 8; min_child_weight: 3; n_estimators: 150, nthread: 4, subsample: 0.8

Tabela 2. Hiperparâmetros dos modelos de ML.

3.5. Avaliação das Abordagens de ML

Foi realizada uma comparação das abordagens de ML propostas com um algoritmo tradicional para prever o TVE entre nós em redes veiculares. O algoritmo denominado LLT (*Link Lifetime*) é utilizado em diversos trabalhos na literatura em aplicações VANET e encontra-se bem descrito em [Nabil et al. 2019, Härri et al. 2008, Souza et al. 2013].

As seguintes métricas foram utilizadas neste trabalho para medir a eficiência das abordagens de predição: MAE (*Mean Absolute Error*, ou Erro Médio Absoluto), MAPE (*Mean Absolute Percentage Error*, ou Erro Percentual Médio Absoluto), também conhecido como MRE (*Mean Relative Error*, ou Erro Médio Relativo) e RMSE (*Root Mean Square Error*, ou Raiz do Erro Quadrado Médio). Essas métricas são calculadas por:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|Tc - Yp|; \quad MAPE = \frac{100}{n}\sum_{i=1}^{n}\frac{|Tc - Yp|}{Tc}; \quad RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}Tc - Yp^2}, \quad (4)$$

em que Tc é o valor real do tempo de duração da comunicação e Yp é o valor predito pelas soluções.

Para cada cenário foi utilizado um *dataset*, onde cada *dataset* dos cenários de Rodovia e Urbano foi dividido em outros dois conjuntos: um conjunto de treinamento (80% das amostras) e um conjunto de teste (20% restante). As previsões realizadas pelos modelos treinados nos *datasets* de teste são avaliadas usando as métricas definidas anteriormente. Durante a simulação, são calculados os valores a serem preditos, referentes ao TVE, pelo algoritmo LLT. Esses mesmos valores são usados no conjunto de teste para calcular os erros do LLT.

A Tabela 3 mostra as métricas obtidas. O MAE demonstra a distância entre os valores reais de tempo de comunicação mensurados e os valores de tempo preditos. O MAPE é o percentual médio da diferença entre o valor predito e o valor real. O RMSE dá um peso maior para valores que sejam muito diferentes entre o previsto e o real, ideal para analisar o impacto de *outliers*. Os resultados indicam que a abordagem tradicional (LLT) apresenta previsões mais extremas com base nas altas taxas de erro relatadas pelas métricas. As soluções de ML generalizam bem e apresentam tempos de comunicação preditos mais próximos da realidade.

Algoritmo	Cenário	MAE (s)	MAPE (%)	RMSE (s)	Tempo de Predição (s)
LLT	Rodovia	24,14	208,86	39,59	**2,25e-05**
Rede Neural	Rodovia	1,89	**34,55**	3,13	1,03e-01
SVR	Rodovia	**1,82**	35,53	**3,00**	5,20e-04
KNN	Rodovia	1,99	37,29	3,12	3,66e-01
Random Forest	Rodovia	2,01	38,03	3,15	4,00e-02
XGBoost	Rodovia	2,07	37,04	3,24	2,70e-03
LLT	Urbano	25,68	190,44	40,08	**2,25e-05**
Rede Neural	Urbano	**1,65**	**19,31**	2,96	1,05e-01
SVR	Urbano	1,85	21,84	2,92	5,20e-04
KNN	Urbano	1,93	22,64	2,90	3,61e-01
Random Forest	Urbano	1,84	20,43	**2,79**	4,20e-02
XGBoost	Urbano	1,96	20,94	2,95	2,60e-03

Tabela 3. Sumário dos resultados dos experimentos.

Por fim, foi medido o tempo que cada abordagem leva para prever o TVE, conforme também mostrado na Tabela 3. Supomos que cada modelo de ML esteja disponível e carregado em um servidor. Então, medimos o tempo para ler os atributos e retornar o valor predito. O algoritmo LLT foi implementado em um *script* fora do simulador. O processo de medição foi realizado fora do ambiente de simulação, pois o ns-3 utiliza tempo virtual, que é diferente do tempo real de execução do *hardware*.

Tais métricas são essenciais para decidir qual abordagem de predição escolher para *offloading* computacional em ambientes de tráfego denso. Esse tipo de cenário seria o pior caso para tomada de decisão, pois o número de predições depende do número de veículos que respondem ao processo de descoberta. Logo, muitos veículos gerariam uma sobrecarga de predições.

4. Impacto da Predição do TVE no Desempenho do *Offloading*

Esta seção apresenta os experimentos realizados para avaliar o impacto causado pelo uso de técnicas de predição no *offloading* computacional. A Subseção 4.1 aborda como os diferentes algoritmos de ML afetam o tempo necessário para tomar uma decisão de *offloading*. A Subseção 4.2 apresenta a avaliação de desempenho de uma solução de *offloading* computacional ao usar a solução de predição proposta.

4.1. Tempo de Decisão do *Offloading* Computacional

Com base nos resultados de tempo de predição mostrados na Tabela 3, foi realizado um experimento para inferir o tempo total médio que cada abordagem de ML requer para executar as predições de TVE. O objetivo é verificar quanto tempo um algoritmo de *offloading* aguarda antes de tomar a decisão de descarregar tarefas. Esse ponto é bastante importante, pois antes de decidir, o algoritmo de *offloading* deve analisar o tempo previsto de cada veículo vizinho.

São realizadas simulações em Rodovias e ambientes Urbanos com alto tráfego, utilizando os maiores valores mostrados na Tabela 4. O tráfego mais denso é utilizado para analisar o desempenho dos modelos de ML e a abordagem LLT no pior tipo de cenário, onde há uma demanda maior nos modelos de predição. Primeiro, foi verificada a taxa média de vizinhos que um veículo encontra durante a simulação. Os valores médios

que um determinado veículo encontrou de veículos vizinhos foram 29 para o cenário Rodoviário e 23 para o cenário Urbano. Em seguida, foi mensurado o produto desse resultado com o tempo de predição de cada abordagem, conforme mostrado na Figura 4.

Uma abordagem de *offloading* usando o modelo SVR resulta em um tempo de decisão menor do que as outras abordagens de ML. O LLT tem um tempo de predição próximo de zero, pois é um algoritmo aritmético simples. As abordagens KNN, *Random Forest* e Rede Neural têm tempo total de decisão alto, o que significa que um algoritmo de *offloading* teria que esperar muito tempo até obter todos os tempos previstos antes de decidir.

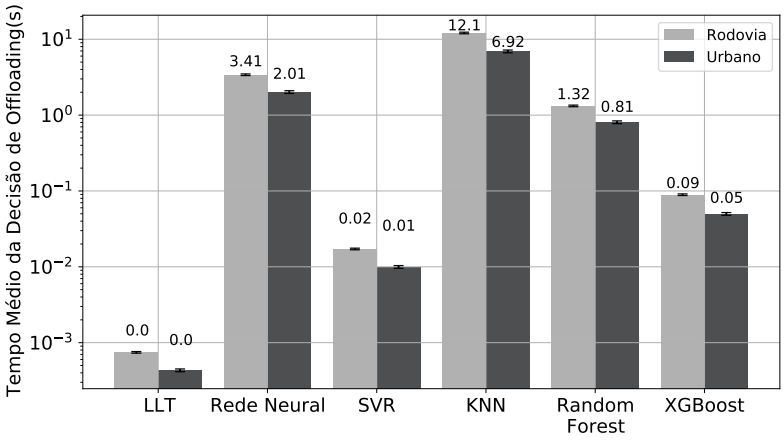


Figura 4. Tempo de decisão para o *offloading* nos cenários avaliados.

Esses resultados mostram que abordagens específicas de ML apresentam muita sobrecarga em cenários de alto tráfego. A sobrecarga é caracterizada pelo tempo que leva para tomar a decisão. Muita sobrecarga afeta o processo de decisão do algoritmo de *offloading*, pois ele deve esperar muito tempo antes de decidir. Em um ambiente de rede veicular dinâmico, longas esperas podem resultar em quebras de comunicação, perda de contato com veículos vizinhos e informações incompletas. Portanto, embora as abordagens de Rede Neural, KNN e *Random Forest* apresentem previsões com baixos erros, seu uso é inviável no *offloading* devido à sobrecarga que podem causar, pelo menos quando se considera um veículo com as capacidades computacionais descritas na última seção.

4.2. Desempenho Geral do *Offloading* Computacional

A eficiência da abordagem proposta baseada em ML na predição de TVE é avaliada em uma aplicação VANET real. Os seguintes experimentos de *offloading* computacional consideraram apenas o modelo de aprendizado SVR, devido ao seu bom desempenho preditivo (baixas taxas de erro) e menor sobrecarga no processo de decisão de *offloading* em ambientes de alto tráfego, conforme a Figura 4.

A comparação foi feita com três algoritmos de *offloading* computacional: um algoritmo de escolha aleatória, o HVC (*Hybrid Vehicular edge Cloud*) e o GCF (*Greedy for CPU Free*), de acordo com [de Souza et al. 2020]. Tais algoritmos de *offloading* usam o algoritmo LLT como função para predizer o TVE entre os veículos. De acordo com os resultados dos experimentos apresentados em [de Souza et al. 2020], o GCF apresentou as melhores taxas de sucesso no processo de decisão de *offloading*. Assim, o GCF foi

escolhido e sua função de cálculo de TVE foi substituída pela abordagem baseada em ML. A solução resultante é chamada GCFML (GCF com *Machine Learning*).

Foi executada uma simulação de *offloading* computacional nos cenários de Rodovia e Urbano usando o simulador ns-3. Os cenários considerados nos testes foram os mesmos da etapa anterior de coleta de dados. Um veículo é escolhido aleatoriamente para ser o transmissor da tarefa. Nos experimentos, foram utilizados dois fatores: número de Veículos e Tamanho da Tarefa. O mesmo Número de Veículos de [de Souza et al. 2020, Monteiro et al. 2012] foi utilizado nos ambientes de Rodovia e Urbano, conforme apresentado na Tabela 4.

Foram utilizados valores de Tamanho da Tarefa maiores que em [de Souza et al. 2020]. O objetivo de colocar cargas de trabalho maiores foi testar os algoritmos de *offloading* em situações mais críticas, que exigem maior eficiência na predição do TVE. Os fatores com seus níveis individuais para cada cenário também são mostrados na Tabela 4.

Número de Veículos no Ambiente Urbano	**Número de Veículos na Rodovia**	**Tamanho da Tarefa (MB)**	**Nome**
50 (esparso)	55 (esparso)	1,5	E1
50 (esparso)	55 (esparso)	3,0	E2
275 (médio)	275 (médio)	1,5	M1
275 (médio)	275 (médio)	3,0	M2
509 (denso)	605 (denso)	1,5	D1
509 (denso)	605 (denso)	3,0	D2

Tabela 4. Fatores e níveis considerados nos experimentos.

Os testes são executados em uma máquina com as seguintes configurações: *Processador Intel(R) Xeon(R) E5-2650 v3 CPU @2.30GHz* com 8 núcleos de *CPU* e 16 GB de *RAM*, rodando o sistema operacional *Linux Ubuntu 18.04.2 LTS*. Os modelos são treinados usando os *datasets* de cada cenário (Rodovia e Urbano). Quando um veículo cliente encontra um veículo vizinho, ele realiza uma consulta para prever a duração do TVE entre eles. Os atributos são coletados e passados para o modelo que retorna o tempo predito.

No cenário de *offloading* configurado, um veículo ou cliente transmissor envia quatro tarefas para veículos vizinhos que podem processar essas tarefas, conforme [de Souza et al. 2020]. As tarefas são processadas em veículos vizinhos, que retornam o resultado ao cliente. Em seguida três métricas são avaliadas: a taxa de sucesso de *offloading*, a taxa de recuperação e a taxa de execução local. Se o processamento das quatro tarefas for bem-sucedido, significa que o *offloading* foi bem-sucedido. Caso algum veículo vizinho não retorne o resultado, houve uma falha de comunicação durante a etapa de processamento, sendo necessário realizar uma recuperação. Assim, o veículo cliente realiza o processamento de uma cópia dessa tarefa localmente. Quando não há veículos com capacidade suficiente para descarregar todas as tarefas, elas são processadas localmente no veículo cliente.

Os resultados para os cenários de Rodovia (Figura 5) e Urbano (Figura 6) indicam que o uso do modelo SVR possui potencial para substituir a abordagem tradicional de predição. O GCFML mostrou-se mais eficiente no cenário Rodoviário, obtendo melhores taxas de sucesso de *offloading* e menor taxa de recuperação nos cenários de média e alta

densidade do que suas contrapartes.

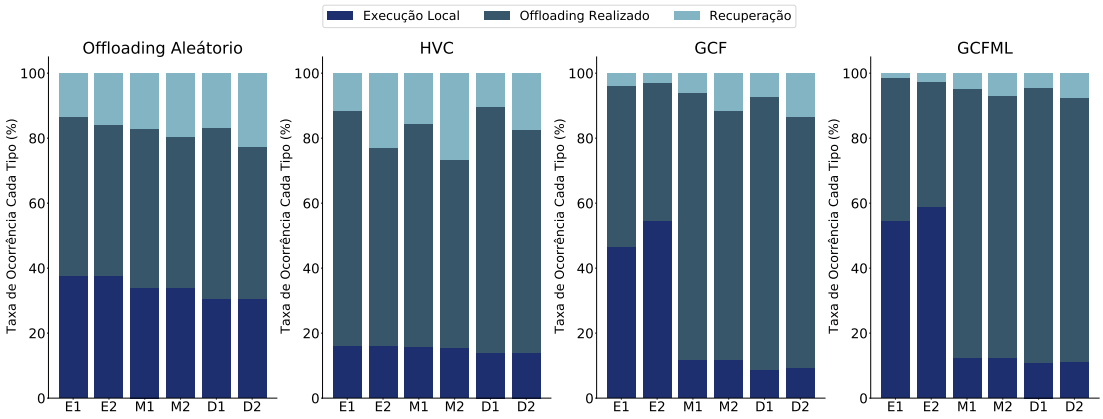


Figura 5. Resultados do *offloading* para o cenário de Rodovia.

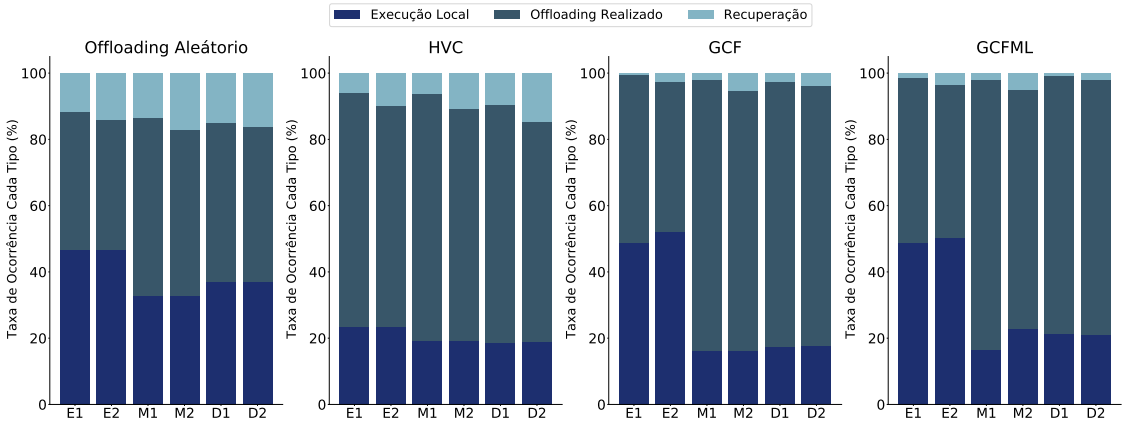


Figura 6. Resultados do *offloading* para o cenário Urbano.

A abordagem proposta diminuiu as taxas de recuperação nas seguintes situações: cenários com menos tarefas e cenários de tráfego com muitos veículos no ambiente Urbano. O GCFML toma melhores decisões de *offloading* ao optar por executar localmente em vez de enviar uma tarefa que será perdida e recuperada posteriormente. Portanto, o GCFML tem uma taxa de execução local mais alta. Outro motivo é o maior dinamismo e imprevisibilidade do cenário Urbano, o que leva a previsões de menores tempos de comunicação. O algoritmo de *offloading* tem menos opções para enviar tarefas com predições de TVE mais curtas, resultando em mais execuções locais.

No geral, os resultados demonstram que o modelo SVR reduz a taxa de recuperação no processo de *offloading* computacional em redes veiculares. Uma taxa de recuperação mais baixa fornece melhor qualidade de serviço para aplicações e usuários, pois há menos retransmissões e menos processos de correção de erros. Como a Figura 5 indica, o cenário de Rodovia é o mais desafiador para as predições, com mais recuperações em todos os níveis testados. Os melhores resultados no cenário de Rodovia foram obtidos com a abordagem GCFML, que resultou em menores taxas de recuperação e melhores taxas de sucesso. Apesar de o cenário Urbano possuir uma topologia mais complexa, os resultados de *offloading* usando o GCF e o GCFML se destacaram, gerando as menores

taxas de recuperação e as maiores taxas de sucesso nos cenários de média e alta densidade veicular.

5. Conclusão

Os resultados experimentais alcançados em *datasets* simulados demonstram que as abordagens de ML baseadas em regressão, como SVR, são promissoras para estimar o TVE entre nós em VANETs. A estratégia melhorou a eficiência da predição. Além disso, identificar variáveis que influenciam o TVE provou ser uma importante contribuição para futuras pesquisas.

O modelo de regressão SVR, empiricamente superior aos demais, foi incorporado a um algoritmo de *offloading* e trouxe melhorias no processo de decisão de quando e para onde fazer *offloading*, resultando em taxas de recuperação menores que o LLT, reflexo das escolhas dos veículos mais apropriados para executar as tarefas.

Por fim, em trabalhos futuros pretendemos testar novas hipóteses de predição baseadas em treinamento, com cenários mais específicos. Assim, avaliaremos se modelos para determinadas situações podem superar modelos mais generalistas, como os baseados em cenários de Rodovias e Urbanos.

Referências

Alabbasi, A. and Aggarwal, V. (2020). Joint information freshness and completion time optimization for vehicular networks. *IEEE Transactions on Services Computing*.

de Souza, A. B., Rego, P. A. L., Rocha, P. H. G., Carneiro, T., and de Souza, J. N. (2020). A task offloading scheme for wave vehicular clouds and 5g mobile edge computing. In *GLOBECOM 2020-2020 IEEE Global Communications Conference*, pages 1–6. IEEE.

de Souza, A. M. and Villas, L. A. (2020). Vem tranquilo: Rotas eficientes baseado na dinâmica urbana futura com deep learning e computação de borda. In *Anais do XXXVIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 351–364. SBC.

Deng, Z., Cai, Z., and Liang, M. (2020). A multi-hop vanets-assisted offloading strategy in vehicular mobile edge computing. *IEEE Access*, 8:53062–53071.

Eckermann, F., Kahlert, M., and Wietfeld, C. (2019). Performance analysis of c-v2x mode 4 communication introducing an open-source c-v2x simulator. In *2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall)*, pages 1–5. IEEE.

Härri, J., Bonnet, C., and Filali, F. (2008). Kinetic mobility management applied to vehicular ad hoc network protocols. *Computer Communications*, 31(12):2907–2924.

He, Y., Zhai, D., Huang, F., Wang, D., Tang, X., and Zhang, R. (2021). Joint task offloading, resource allocation, and security assurance for mobile edge computing-enabled uav-assisted vanets. *Remote Sensing*, 13(8):1547.

Hussain, S., Wu, D., Memon, S., and Bux, N. K. (2019). Vehicular ad hoc network (vanet) connectivity analysis of a highway toll plaza. *Data*, 4(1):28.

Khatri, S., Vachhani, H., Shah, S., Bhatia, J., Chaturvedi, M., Tanwar, S., and Kumar, N. (2021). Machine learning models and techniques for vanet based traffic manage-

ment: Implementation issues and challenges. *Peer-to-Peer Networking and Applications*, 14(3):1778–1805.

Kim, K., Koo, S., and Choi, J.-W. (2020). Analysis on path rerouting algorithm based on v2x communication for traffic flow improvement. In *2020 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 251–254. IEEE.

Kulla, E., Morita, S., Katayama, K., and Barolli, L. (2019). Route lifetime prediction method in vanet by using aodv routing protocol (aodv-lp). In *Complex, Intelligent, and Software Intensive Systems*, pages 3–11, Cham. Springer International Publishing.

Lee, M. and Atkison, T. (2021). Vanet applications: Past, present, and future. *Vehicular Communications*, 28:100310.

Li, C., Wang, S., Huang, X., Li, X., Yu, R., and Zhao, F. (2018). Parked vehicular computing for energy-efficient internet of vehicles: A contract theoretic approach. *IEEE Internet of Things Journal*, 6(4):6079–6088.

Monteiro, R., Sargento, S., Viriyasitavat, W., and Tonguz, O. K. (2012). Improving vanet protocols via network science. In *2012 IEEE Vehicular Networking Conference (VNC)*, pages 17–24. IEEE.

Moura, D. L. L., Aquino, A. L. L., and Loureiro, A. A. F. (2019). Um mecanismo de offloading para redes de sensores veiculares. In *Anais do XXXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 280–292. SBC.

Nabil, M., Hajami, A., and Haqiq, A. (2019). Predicting the route of the longest lifetime and the data packet delivery time between two vehicles in vanet. *Mobile Information Systems*, 2019.

Nešović, E. (2017). On geometric interpretation of pseudo-angle in minkowski plane. *International Journal of Geometric Methods in Modern Physics*, 14(05):1750068.

Salem, A. H., Damaj, I. W., and Mouftah, H. T. (2022). Vehicle as a computational resource: Optimizing quality of experience for connected vehicles in a smart city. *Vehicular Communications*, 33:100432.

Souza, A. B., Celestino, J., Xavier, F. A., Oliveira, F. D., Patel, A., and Latifi, M. (2013). Stable multicast trees based on ant colony optimization for vehicular ad hoc networks. In *The International Conference on Information Networking 2013 (ICOIN)*, pages 101–106. IEEE.

Sun, P., AlJeri, N., and Boukerche, A. (2020). Dacon: A novel traffic prediction and data-highway-assisted content delivery protocol for intelligent vehicular networks. *IEEE Transactions on Sustainable Computing*, 5(4):501–513.

Ye, M., Guan, L., and Quddus, M. (2019). Mpbrp-mobility prediction based routing protocol in vanets. In *2019 International Conference on Advanced Communication Technologies and Networking (CommNet)*, pages 1–7. IEEE.

Zhang, J., Ren, M., Labiod, H., and Khoukhi, L. (2017). Link duration prediction in vanets via adaboost. In *GLOBECOM 2017-2017 IEEE Global Communications Conference*, pages 1–6. IEEE.