Um Algoritmo de Compressão de Dados baseado em TinyML para Internet das Coisas

Gabriel Signoretti * **Marianne Silva** * **Pedro Andrade** * e **Ivanovitch Silva** *

* *Universidade Federal do Rio Grande do Norte/PPgEEC, Natal-RN*

E-mails: {gabrielsig, marianne.silva.086, pedro.meira.055}@ufrn.edu.br e ivanovitch.silva@ufrn.br

Abstract: With the advancement and mass adoption of Internet of Things (IoT) solutions, new challenges arise such as transmission and storage of the growing volume of data. Thus, it is clear that when devices transmit potentially irrelevant or redundant data, there is greater energy and processing expenditure, in addition to the unnecessary use of the communication channel. Thus, data compression solutions in edge computing devices themselves become increasingly attractive, enabling the elimination of samples that would have little or no contribution to the application, reducing the volume of data needed to represent the information. However, such devices present on the market today have serious limitations in terms of storage and processing power. In order to circumvent these limitations, the field of TinyML appears, which seeks ways to implement machine learning models in devices with low computational power. In this context, the present work proposes the development of a new online, unsupervised, and automatically adaptable data compression algorithm for IoT applications. To validate the proposal, a case study with data captured from vehicular sensors was used and preliminary results show that it is possible to reach 90% of compression rates, with mean absolute errors of 1,1 in the analyzed scenario.

Resumo: Com o avanço e adoção em massa de soluções de Internet das Coisas (IoT), surgem novos desafios como transmissão e armazenamento do crescente volume de dados. Desta forma, percebe-se que quando dispositivos transmitem dados potencialmente irrelevantes ou redundantes, há um maior gasto de energia e processamento, além do uso desnecessário do canal de comunicação. Assim, soluções de compressão de dados nos próprios dispositivos (*edge computing*) se tornam cada vez mais atrativas, possibilitando a eliminação de amostras que teriam pouca ou nenhuma contribuição para a aplicação, reduzindo o volume de dados necessários para representar as informações. No entanto, tais dispositivos presentes hoje no mercado tem sérias limitações de armazenamento e poder de processamento. A fim de circundar tais limitações, surge o campo de *TinyML*, que busca maneiras de implementar modelos de aprendizado de máquina em dispositivos de baixo poder computacional. Nesse contexto, o presente trabalho propõe o desenvolvimento de um novo algoritmo de compressão de dados on-line, não supervisionado, e automaticamente adaptável para aplicações de IoT. Para validar a proposta um estudo de caso com dados capturados de sensores veiculares foi utilizado e resultados preliminares mostraram ser possível alcançar 90% de taxa de compressão, com um erro médio absoluto de 1.1 no cenário analisado.

Keywords: IoT; Intelligent Vehicle; Data Compression; TinyML; Evolving Algorithm.
Palavras-chaves: IoT; Veículos Inteligentes; Compressão de Dados; TinyML; Algoritmo Evolutivo.

1. INTRODUÇÃO

O conceito de Internet das Coisas (IoT), permite o monitoramento e a conectividade de dispositivos do dia a dia uns com os outros (Gubbi et al., 2013; Vashi et al., 2017). A proliferação cada vez mais ubíqua desses dispositivos conectados à rede possibilita um cenário em que sensores e atuadores se misturam perfeitamente com o ambiente ao nosso redor, e as informações são compartilhadas entre uma multitude de plataformas (Gubbi et al., 2013).

Dessa forma, esse novo paradigma abrange muitas áreas da vida moderna e, até certo ponto já vem mudando a forma como a sociedade interage com o mundo ao seu redor. Nesse contexto, essas tecnologias surgem envolvendo os mais diversos domínios, como saúde, transporte, agricultura, cidades inteligentes, indústria e entre outros. Com isso, esses cenários tornam-se facilitadores críticos para aumentar o volume de dados gerados, devido à coleta em uma ampla variedade de contextos (Azar et al., 2019). Estes fluxos de dados podem formar um grande volume muito rapidamente e, como consequência, demandar a ne-

cessidade de uma quantidade de espaço de armazenamento em disco expressiva (Signoretti et al., 2019a).

Portanto, pode-se perceber que a geração dos dados pelos sensores ocorre em formato de fluxo contínuo que possui características variadas de acordo com cada aplicação e que constantemente evolui com o tempo. A taxa de amostragem, por exemplo, pode variar de uma vez por dia a cada milissegundo dependendo do contexto (Yamaoka et al., 2019), um exemplo desses é a captura de dados veiculares (Signoretti et al., 2019b). Esta variação na coleta ocorre porque existem várias demandas de aplicação distintas, modelos de dispositivos, diferentes necessidades de configuração ou limitações no canal de transmissão de dados (Shukla and Simmhan, 2016).

Nesse cenário, a transmissão e armazenamento do crescente volume de dados precisam ser otimizados. Evitando que dispositivos transmitam dados potencialmente irrelevantes ou redundantes, o que pode gerar uma economia de energia e processamento, além de evitar usar o canal de comunicação desnecessariamente (Correa et al., 2019). Sabe-se, que nesses casos, a maior parte do consumo de energia de um dispositivo que se comunica utilizando o meio sem fio está na transmissão de dados e não no processamento de informação. Dessa maneira, a utilização de técnicas de compressão de dados visa eliminar informações redundantes para minimizar o espaço necessário para armazenamento(Azar et al., 2019). Assim, ao possibilitar a eliminação de amostras que teriam pouca ou nenhuma contribuição para a aplicação, o volume de dados necessários para representar as informações é reduzido.

Para isso, a nova tendência é adicionar "inteligência" aos dispositivos IoT para que eles processem dados e tomem decisões sem passar todas as informações brutas para a nuvem (Azar et al., 2019). Porém, a maioria dos dispositivos de IoT presentes hoje no mercado tem sérias limitações de armazenamento e poder de processamento, o que faz com que os algoritmos, em sua maioria, não sejam estruturados com técnicas de Aprendizado de Máquina (Mahdavinejad et al., 2018).

Deste modo, a motivação desse trabalho vem da necessidade do desenvolvimento de algoritmos especializados que possam se adequar a esse cenário e suas limitações. Assim, sabe-se que o TinyML permite a implantação de modelos de aprendizado de máquina em dispositivos IoT e que apresentam baixo consumo de energia, poder de processamento e latência (Banbury et al., 2021; Gorospe et al., 2021). Além disso, técnicas de TinyML permitem que algumas análises e interpretações sejam realizadas localmente e em tempo real no ponto de coleta. Esse cenário tem o potencial para redução de custos e melhorar a proteção e privacidade dos dados (Banbury et al., 2021). Outrossim, para cenários de IoT, pode-se argumentar que preferencialmente os algoritmos devem funcionar sem o conhecimento prévio dos dados, ou seja, de forma não supervisionada. Estes são uma classe de modelos que não são explicitamente instruídos para prever algo específico, mas os padrões nos dados são identificados pelo próprio algoritmo e é deixado para o usuário interpretar a saída e chegar a uma conclusão (Zoppi et al., 2020).

Desta forma, este trabalho tem como objetivo desenvolver um novo algoritmo de compressão de dados para séries temporais. No entanto, por ser aplicado em um contexto com diversas restrições, o algoritmo deve ser capaz de atender uma série de características específicas ao mesmo tempo que se mantém efetivo. Dentre as principais características que o algoritmo deve atender, é possível citar: (a) baixo custo de processamento, possibilitando ser executado em dispositivos de baixo poder computacional; (b) eficiência de armazenamento; (c) capacidade de ser executado *on-line* e de maneira não-supervisionada e autônoma, já que não se tem acesso prévio aos dados de interesse; (d) ser evolutivo por natureza, detectando e se adaptando a mudanças no comportamento dos dados em tempo real; (e) possuir número reduzido de parâmetros dependentes do usuário ou do problema, de forma a ser possível executá-lo em uma variedade de cenários de maneira *plug-and-play*.

Por fim, o restante desse artigo encontra-se organizado da seguinte forma: na Seção 2, são apresentados os trabalhos relacionados enquanto que na Seção 3 são elucidados detalhes do algoritmo proposto; a Seção 4 descreve o estudo de caso realizado; a Seção 5 discute os principais resultados obtidos; por fim, a Seção 6 apresenta as considerações finais e indica caminhos promissores para trabalhos futuros.

2. TRABALHOS RELACIONADOS

A compressão tem como intuito eliminar informações redundantes, a fim de minimizar o espaço necessário para o armazenamento. Consequentemente, reduzir o número de bits a serem transmitidos periodicamente ao nó de extremidade, além de melhorar a eficiência na análise das informações (Azar et al., 2019; Uthayakumar et al., 2018). Deste modo, após uma revisão sistemática verificou-se que algumas soluções de compressão de dados para IoT estão disponíveis na literatura, dentro elas destacam: *Swing Door Trending* (SDT) tem sido amplamente usado para compactação de dados em cenários de IoT. No entanto, a principal desvantagem é o fato de que o algoritmo SDT depende de um parâmetro principal definido pelo usuário, o Desvio de Compressão (DC) (Bristol, 1990). A definição de um valor ideal para este parâmetro depende do aplicativo e exigirá teste ou conhecimento prévio dos comportamentos dos dados. Muitos autores propuseram melhorias e novas etapas de autoconfiguração para definir automaticamente o parâmetro do DC. Em (Souza and Guedes, 2014) os autores propõem o uso da *Exponential Moving Average* (EMA) como meio de definir o DC no que chamaram de *Adaptive Swing Door Trending* (ASDT).

Em contraste, a solução SDT melhorada proposta em (Correa et al., 2019), chamada de SDT de Autodefinição (SSDT), não requer a configuração de parâmetros anteriores. No entanto, para fazer isso, o algoritmo precisa ser treinado em um subconjunto de amostras no início dos aplicativos. Seus experimentos têm mostrado bons resultados, mas, para outras aplicações onde os dados do sensor podem passar por constantes mudanças de conceito, pode não ser tão eficaz ou precisa ser periodicamente retreinado.

Em outra frente, os autores em (Moon et al., 2018), fazem uso de algoritmos de compressão com perdas baseados em transformadas para comprimir os dados de estações meteorológicas de IoT. Esses algoritmos funcionam transformando os dados do domínio do tempo para outro domínio onde podem ser potencialmente mais fáceis de compactar.

Os 3 algoritmos escolhidos foram o *Discrete Cosine Transform* (DCT), *Fast Walsh-Hadamard Transform* (FWHT) e *Discrete Wavelet Transform* (DWT), a partir do qual o DCT mostrou ter o melhor desempenho. A principal desvantagem dessa abordagem decorre do fato de que as transformações de dados não podem ser feitas on-line e ponto a ponto, uma vez que precisam que o conjunto completo de amostras seja calculado.

Assim, fica claro na literatura que estudos que desenvolveram algoritmos de compressão utilizando aprendizado de máquina ainda são considerados poucos. Além disso, nenhum deles foi considerado uma solução leve e on-line. Portanto, a partir da discussão apresentada acima, fica claro que ainda existem lacunas a serem exploradas nesta área, o que favorece o desenvolvimento de novas soluções, para compressão de dados para IoT, principalmente para o conceito emergente de TinyML.

3. *TINY ANOMALY COMPRESS*

O algoritmo *Tiny Anomaly Compress* (TAC), como o nome sugere, faz uso da detecção de anomalias para realizar a compressão da série temporal. Ele é baseado no *Typicality and Eccentricity Data Analytics* (TEDA), que se baseia nos conceitos de excentricidade e tipicidade. Essas duas grandezas representam, respectivamente, a densidade e a proximidade das amostras no espaço dos dados (Angelov, 2014b). Além disso, é uma abordagem estatística, mas difere da teoria de probabilidade tradicional (Angelov, 2014b).

De forma complementar, o TEDA surge como uma metodologia sistemática alternativa que não requer, dentre outras coisas: suposições anteriores sobre a distribuição dos dados; pré-especificação de parâmetros dependentes do problema ou do usuário; independência das amostras de dados individuais (observações); uma quantidade substancial de observações, podendo trabalhar com apenas 3 amostras de dados (Angelov, 2014b,a). Portanto, por ser inteiramente baseado nos dados e sua distribuição mútua no espaço, a abordagem pode ser categorizada como *empírica* em sua natureza.

Outrossim, o TEDA é baseado em várias novas quantidades que são calculadas a partir da proximidade/similaridade das amostras no espaço de dados. No entanto, essas medidas não são exatamente iguais à densidade usada na estatística e outras áreas (Angelov, 2014a).

Inicia-se por definir o espaço de dados $\mathbf{x} \in \mathbb{R}^n$, onde $\mathbf{x}$ representa o fluxo de dados em um espaço *n-dimensional*, ou seja, pode-se representar a sequência ordenada de dados, como:

$$\{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_k}, ...\},\ \mathbf{x_k} \in \mathbb{R}^n,\ k \in \mathbb{N} \tag{1}$$

Dessa forma, tem-se que cada amostra x_k é um vetor de n dimensões que representa o sistema no instante discreto k. A metodologia pode ser estendida para amostras de qualquer dimensionalidade n. Porém, a critério de exemplo, pretende-se observar um fenômeno, cujo as amostras são descritas em um espaço de uma dimensão.

Os valores de **tipicidade** (τ), e **excentricidade** (ξ) passam a ser observados a partir do momento em que 3 ou mais amostras distintas tenham sido obtidas ($k \geq 3$) (Angelov, 2014b). Assim, a **excentricidade** de uma amostra x em um instante k é definida pela razão entre a sua proximidade acumulada e soma das proximidades acumuladas de todas as demais amostras. A **tipicidade** (τ), por sua vez, é definida como o complemento da excentricidade (Angelov, 2014b).

Esses cálculos podem ser realizados de forma recursiva, o que os torna particularmente úteis. Com este método, não é necessário manter as amostras anteriores armazenadas localmente; em vez disso, o cálculo das métricas pode ser executado usando apenas a última amostra recebida e valores agregados que representam o estado do sistema no instante anterior $(k-1)$.

Isto pode ser obtido calculando a **média** (μ) e a **variância** (σ^2) também de forma recursiva para cada nova amostra recebida. Esse cálculo é mostrado nas Equações 2 e 3 Angelov (2014a). Dessa forma, é possível utilizar o formato recursivo das equações para calcular a **excentricidade** e **tipicidade**, apresentadas nas equações 4 e 5 respectivamente Angelov (2014a).

$$\mu_k(x) = \frac{k-1}{k}\mu_{k-1} + \frac{1}{k}x_k,\ \ \mu_1 = x_1 \tag{2}$$

$$\sigma_k^2(x) = \frac{k-1}{k}\sigma_{k-1}^2 + \frac{1}{k-1}\left\|x_k - \mu_k\right\|^2,\ \ \sigma_1^2 = 0 \tag{3}$$

$$\xi_k(x) = \frac{1}{k} + \frac{(\mu_k - x_k)^T(\mu_k - x_k)}{k\sigma_k^2} \tag{4}$$

$$\tau_k(x) = 1 - \xi_k(x) = \frac{k-1}{k} - \frac{(\mu_k - x_k)^T(\mu_k - x_k)}{k\sigma_k^2} \tag{5}$$

Este método recursivo usado para atualizar os valores de **excentricidade** e **tipicidade** torna a execução do algoritmo muito eficiente computacionalmente. Além disso, É importante ressaltar que ambas as funções são limitadas e, portanto, podem ser normalizadas (Kangin and Angelov, 2015), aqui apresentadas nas Equações 6 e 7 respectivamente.

$$\zeta_k(x) = \frac{\xi_k(x)}{2} \tag{6}$$

$$t_k(x) = \frac{\tau_k(x)}{k-2} \tag{7}$$

Essas duas funções são bastante importantes, pois tem um comportamento semelhante ao da *Função Densidade de Probabilidade* (FDP) da teoria de probabilidade tradicional. Porém, diferentemente da função FDP, elas não requerem nenhum conhecimento prévio sobre a distribuição das amostras e representam tanto a distribuição espacial das amostras quanto suas frequências de ocorrência (Angelov, 2014b).

Por fim, o TEDA pode utilizar a desigualdade de Chebyshev, que é um principio estatístico amplamente usado para detecção de anomalias (Angelov, 2014a). Supondo um conjunto suficientemente grande de amostras de qualquer

distribuição de dados, a desigualdade de Chebyshev garante que não mais do que $1/m^2$ amostras se encontrarão a mais de $m\sigma$ de distância da média (Saw et al., 1984).

Em (Angelov, 2014a), os autores demonstram que é possível derivar uma condição que fornece exatamente o mesmo resultado (mas sem fazer suposições sobre a quantidade de dados, sua independência e assim por diante) que a desigualdade de Chebyshev mas representada em termos do TEDA. Essa nova desigualdade pode ser vista na Equação 8.

$$\zeta_k(x) > \frac{m^2+1}{2k} \tag{8}$$

Essa fórmula, chamada de **desigualdade de excentricidade** (do inglês *excentricity inequality*) (Angelov, 2015), permite a realização de analises locais e individuais para cada nova amostra obtida em um fluxo de dados on-line. Para isso, basta que seja calculada sua excentricidade normalizada e, caso a mesma ultrapasse o limite, a amostra é classificada como anomalia.

Com base nestes conceitos, o TAC surge com a introdução do conceito de uma "janela de anomalia" deslizante e dinâmica. Para fazer isso, introduziu-se uma variável de contador que investiga quantas anomalias foram encontradas desde a última amostra salva. Para controlar o comportamento dessas janelas, um novo hiperparâmetro precisa ser introduzido: o *window_limit*. Com ele, o usuário pode controlar o número de anomalias que devem ser detectadas antes que a janela seja considerada "*full*". O fluxo do algoritmo pode ser visto na Figura 1

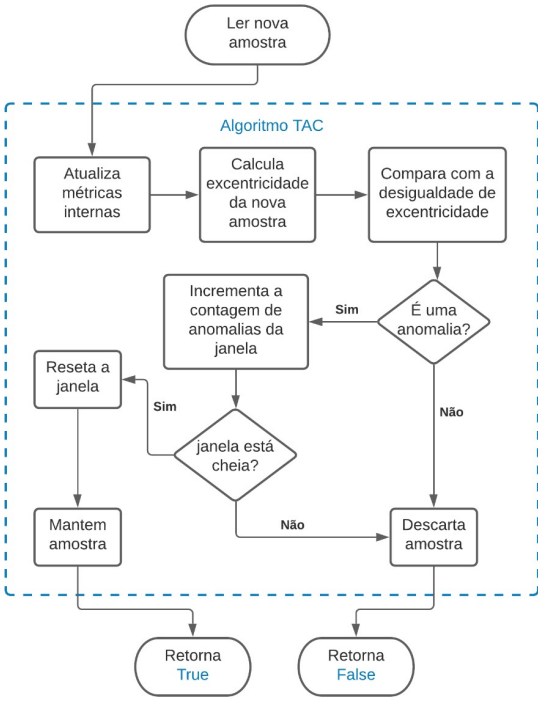


Figura 1. Fluxo do Algoritmo.

A grande diferença na execução do algoritmo surge como resultado deste conceito de "janela de anomalias". Isso ocorre porque, sempre que a janela ainda não estiver cheia, nenhuma nova amostra deve ser salva. No entanto, assim que a janela for preenchida e uma nova anomalia for detectada, esta amostra deve ser salva. Nesta etapa, diz-se que o modelo está detectando um possível desvio de conceito na série. Neste ponto, a janela deve ser zerada e os parâmetros internos do modelo (K, média e variância) retornados aos seus valores iniciais para iniciar uma nova janela.

A janela é dita dinâmica porque não tem tamanho fixo em relação ao número total de amostras de dados incluídas nelas. A janela é controlada apenas pelo número de anomalias detectadas. Portanto, o número total de amostras em cada um deles é determinado pela variabilidade intrínseca do conjunto de dados.

O hiper-parâmetro *window_limit* pode ser interpretado como o *atraso de resposta* do modelo para um desvio de conceito no sinal. Para valores maiores de *window_limit*, por exemplo, o modelo deve detectar uma quantidade maior de pontos de dados anômalos para acionar um *reset* de parâmetro interno, detectando assim um desvio de conceito e salvando uma nova amostra.

4. ESTUDO DE CASO

O estudo de caso visa avaliar o impacto do TAC e a fidelidade dos dados recuperados após a restauração/descompressão. Assim, o seu objetivo principal é conduzir um teste para analisar o desempenho do algoritmo de compressão de dados TAC recém-desenvolvido para fluxos de dados de sensores veiculares. Para isso, as próximas subseções descrevem a base de dados utilizada, métricas de avaliação e processo de execução.

4.1 Base de Dados

A base de dados utilizada neste trabalho foi obtida, por meio de um experimento realizado pelo Grupo de Veículos Inteligentes da Universidade Federal do Rio Grande do Norte, Brasil. O Grupo tem como objetivo capturar e analisar dados veiculares com a utilização de aprendizado de máquina.

Neste trabalho, é considerado o caso de uso de uma rota experimental realizada na cidade de Natal-RN, com cerca de 30 Km, para o veículo Chevrolet Onix, que possuí cerca de 22 sensores (Signoretti et al., 2019b). Esses sensores foram capturados com a utilização de dispositivo Edge OBD-II, sistema de autodiagnóstico disponível na maioria dos veículos que circulam atualmente, além disso não precisam de *smartphone* para comunicação com a *cloud computing*, ou seja, opera de forma autônoma com acesso a unidade de controle do motor (Signoretti et al., 2019b).

Em particular, optou-se por usar os sensores de velocidade e a voltagem da bateria, coletados em uma taxa de amostragem de 500ms. Desta forma, o conjunto de dados para o veículo Chevrolet Onix resultou em 10230 amostras.

4.2 Métricas

Para avaliar os resultados de compressão dos testes, várias métricas foram escolhidas. Isso foi necessário porque o processo de compressão com perdas precisa ser avaliado em relação ao Erro de Compressão (CE) resultante , bem

como à Taxa de Compressão (TC) alcançada. Como o desempenho nessas duas frentes tende a ser inversamente proporcional, ambas devem ser consideradas na avaliação.

Erro de Compressão: o erro de compressão mede a diferença observada entre a série temporal original e os dados reconstruídos após os processos de compressão e descompressão. Para gerar a série descompactada para o TAC, pode-se utilizar métodos de interpolação quadrática ou linear simples para preencher as lacunas entre as amostras salvas.

Para medir o erro entre o sinal original e o sinal descomprimido, as métricas primárias escolhidas foram o Erro Médio Absoluto (EMA) e o Erro Quadrático Médio (EQM), que são mostrados nas Equações 9 e 10.

$$EMA = \frac{1}{n}\sum_{i=1}^{N}|x_i - \hat{x}_i| \tag{9}$$

$$EQM = \frac{1}{n}\sum_{i=1}^{N}(x_i - \hat{x}_i)^2 \tag{10}$$

Onde x_i são as amostras do sinal original, $\hat{x}_i$ são as amostras do sinal descomprimido e n é o número total de amostras.

Taxa de Compressão: como é o caso da medição do erro, existem várias abordagens para quantificar a eficácia da compressão na redução do tamanho. Para consistência entre os algoritmos, o TC foi medido como a porcentagem de redução de tamanho entre o arquivo original e o arquivo com a representação do sinal compactado. Este valor é calculado conforme mostrado na Equação 11.

$$TC = \left(1 - \frac{tamanho(arquivoCompactado)}{tamanho(arquivoOriginal)}\right) \times 100 \tag{11}$$

Onde *tamanho*() é a função que retorna o tamanho do arquivo em KB.

Por fim, o teste foi realizado de forma offline, pois todos os dados já foram previamente adquiridos. Assim, verificou-se os melhores parâmetros para a execução e ao final de cada rodada, as métricas de desempenho do algoritmo foram coletadas para o conjunto de dados avaliado.

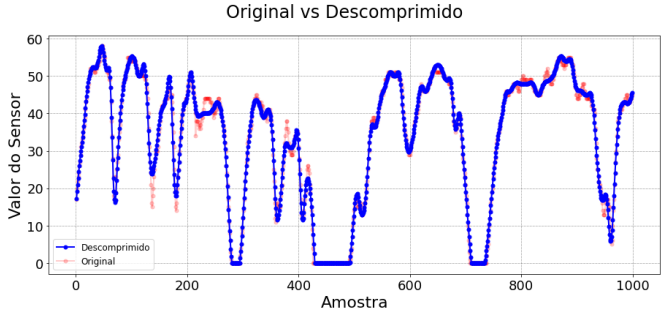
5. RESULTADOS E DISCUSSÃO

Após analisar a distribuição dos dados e realizar alguns testes, verificou-se que um bom conjunto de parâmetros a serem utilizados seria $window\_limit = 6$ e $m = 0.6$, para ambos sensores. Assim, após a definição dos parâmetros, o modelo foi executado no conjunto de dados com os dois sensores selecionados e o resultado da compressão pode ser visualizado na Tabela 1.

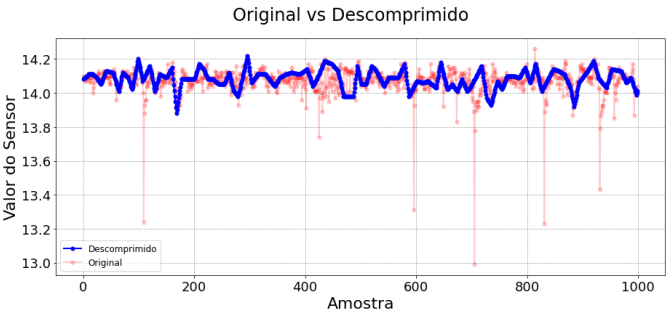
De fato percebe-se que o algoritmo atinge uma alta taxa de compressão, mesmo mantendo taxas de erro relativamente baixas. Além disso, na Figura 2 é possível verificar os sinais originais e os resultantes do processo de compressão e descompressão. Por questões de espaço, estão visíveis as primeiras 1000 amostras da serie completa que contem 10230 amostras no total. O sinal original é plotado em vermelho e o sinal descompactado resultante está sobrepondo o original, na cor azul.

Tabela 1. Resultados da Compressão

Sensor	Total de amostras	Amostras mantidas	TC (%)	EQM	EMA
Velocidade	10230	991	90.31%	4.2482	1.1326
Voltagem da Bateria	10230	1042	89.8%	0.0114	0.0581



(a) Sensor de Velocidade.



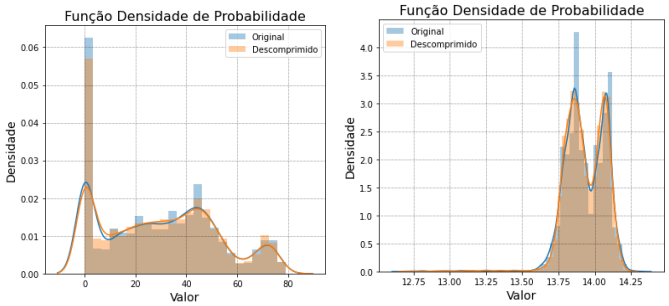
(b) Sensor da Voltagem da Bateria.

Figura 2. Primeiras 1000 amostras dos sinais Originais e Descomprimidos.

Por fim, com o intuito de aferir a similaridade dos sinais gerados pelo processo de descompactação com os sinais originais, na Figura 3 são exibidas as Funções Densidade de Probabilidade (FDP) dos mesmos. É possível ver uma grande proximidade entre as FDP originais e as dos sinais descompactados. Adicionalmente, os sinais descompactados tendem a eliminar valores muito discrepantes das distribuições, tendo efeito de suavização de ruido além da compressão.

6. CONCLUSÃO

Este artigo teve como objetivo propor um algoritmo de compressão de dados leve e dinâmico para ambientes IoT, aplicados em TinyML. É um campo de rápido crescimento, pois os aplicativos com recursos de IoT estão crescendo em número exponencialmente, trazendo novos desafios devido à grande quantidade de geração de dados causada pelo aumento de sensores conectados. Diante disso, verificou a performance do algoritmo por meio de um dataset de dados veiculares. Assim, verificou-se a taxa de compressão e o erro de compressão. Deste modo, percebe-se que o



(a) Sensor de Velocidade. (b) Sensor da Voltagem da Bateria.

Figura 3. Função Densidade de Probabilidade dos Sinais

algoritmo TAC apresenta uma contribuição interessante para a área proposta, uma vez que que o modelo TAC foi capaz de atingir um CR máximo de 90,31% com um EMA de 1,1 no cenário analisado.

Como trabalhos futuros, inclui mas não está limitado a: aumentar o número de métricas avaliadas; realizar comparação com algoritmos de *benchmark* presentes na literatura; e avaliar o impacto no desempenho do algoritmo quando embarcado em um dispositivo baseado em microcontrolador.

AGRADECIMENTOS

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001. Bem como a agência brasileira de fomento Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), Processo nº 435683/2018-7.

REFERÊNCIAS

Angelov, P. (2014a). Anomaly detection based on eccentricity analysis. In *2014 IEEE Symposium on Evolving and Autonomous Learning Systems (EALS)*, 1–8.

Angelov, P. (2014b). Outside the box: An alternative data analytics framework. *Journal of Automation, Mobile Robotics Intelligent Systems*, 8, 29–35.

Angelov, P. (2015). Typicality distribution function — a new density-based data analytics tool. In *2015 International Joint Conference on Neural Networks (IJCNN)*, 1–8.

Azar, J., Makhoul, A., Barhamgi, M., and Couturier, R. (2019). An energy efficient iot data compression approach for edge machine learning. *Future Generation Computer Systems*, 96, 168 – 175.

Banbury, C., Zhou, C., Fedorov, I., Matas, R., Thakker, U., Gope, D., Janapa Reddi, V., Mattina, M., and Whatmough, P. (2021). Micronets: Neural network architectures for deploying tinyml applications on commodity microcontrollers. *Proceedings of Machine Learning and Systems*, 3.

Bristol, E.H. (1990). Swinging door trending: adaptive trend recording. In *Proc. of the ISA National Conf.*, 749–753.

Correa, J.D.A., Pinto, A.S.R., Montez, C., and Leão, E. (2019). Swinging door trending compression algorithm for iot environments. In *Anais do IX Simpósio Brasileiro de Engenharia de Sistemas Computacionais*, 143–148. SBC, Porto Alegre, RS, Brasil.

Gorospe, J., Mulero, R., Arbelaitz, O., Muguerza, J., and Antón, M.Á. (2021). A generalization performance study using deep learning networks in embedded systems. *Sensors*, 21(4), 1031.

Gubbi, J., Buyya, R., Marusic, S., and Palaniswami, M. (2013). Internet of things (iot): A vision, architectural elements, and future directions. *Future generation computer systems*, 29(7), 1645–1660.

Kangin, D. and Angelov, P. (2015). In *2015 International Joint Conference on Neural Networks (IJCNN)*, 1–8.

Mahdavinejad, M., Rezvan, M., Barekatain, M., Adibi, P., Barnaghi, P., and Sheth, A. (2018). Machine learning for internet of things data analysis: a survey. *Digital Communications and Networks*, 4(3), 161–175.

Moon, A., Kim, J., Zhang, J., and Son, S.W. (2018). Evaluating fidelity of lossy compression on spatiotemporal data from an iot enabled smart farm. *Computers and Electronics in Agriculture*, 154, 304–313.

Saw, J.G., Yang, M.C.K., and Mo, T.C. (1984). Chebyshev inequality with estimated mean and variance. *The American Statistician*, 38(2), 130–132.

Shukla, A. and Simmhan, Y. (2016). Benchmarking distributed stream processing platforms for iot applications. In *Technology Conference on Performance Evaluation and Benchmarking*, 90–106. Springer.

Signoretti, G., Silva, M., Araujo, J., Silva, I., Silva, D., Ferrari, P., and Sisinni, E. (2019a). A dependability evaluation for obd-ii edge devices: An internet of intelligent vehicles perspective. In *2019 9th Latin-American Symposium on Dependable Computing (LADC)*, 1–9.

Signoretti, G., Silva, M., Dias, A., Silva, I., Silva, D., and Ferrari, P. (2019b). Performance evaluation of an edge obd-ii device for industry 4.0. In *2019 II Workshop on Metrology for Industry 4.0 and IoT (MetroInd4.0 IoT)*, 432–437.

Souza, E.J.M.N.L.A.D.C. and Guedes, L.A. (2014). Adaptive swinging door trending: Um algoritmo adaptativo para compressão de dados em tempo real. In *Anais do XX Congresso Brasileiro de Automática*.

Uthayakumar, J., Vengattaraman, T., and Dhavachelvan, P. (2018). A survey on data compression techniques: From the perspective of data quality, coding schemes, data type and applications. *Journal of King Saud University-Computer and Information Sciences*.

Vashi, S., Ram, J., Modi, J., Verma, S., and Prakash, C. (2017). Internet of things (iot): A vision, architectural elements, and security issues. 492–496.

Yamaoka, H., Itakura, K., Takahashi, E., Nakagawa, G., Michaelis, J., Kanemasa, Y., Ueki, M., Matsumoto, T., Take, R., Tanie, S., and Inoue, D. (2019). Dracena: A real-time iot service platform based on flexible composition of data streams. 596–601.

Zoppi, T., Ceccarelli, A., and Bondavalli, A. (2020). Into the unknown: Unsupervised machine learning algorithms for anomaly-based intrusion detection. In *2020 50th Annual IEEE-IFIP International Conference on Dependable Systems and Networks-Supplemental Volume (DSN-S)*, 81–81.