Avaliação do Impacto de DP-SGD em Modelos Otimizados com Tinyml

Davi Bezerra Yada da Silva[1], **Aldri Luiz dos Santos**[2], **Jeandro de M. Bezerra**[1,2]

[1]Campus de Quixadá - Universidade Federal do Ceará (UFC)

[2]Depto. de Ciência da Computação – Universidade Federal de Minas Gerais (UFMG)

davi09bezerra@alu.ufc.br,aldri@dcc.ufmg.br,jeandro@ufc.br

Resumo. *Os modelos de aprendizado profundo (MAP) são aplicados na detecção de ataques e anomalias em redes IoT. O paradigma tiny machine learning (tinyml) viabiliza a execução local desses modelos com baixo consumo de recursos e maior privacidade. No entanto, MAPs ainda podem vazar dados por ataques adversariais. Este trabalho implementa uma rede feedforward para classificação e um autoencoder para detecção de anomalias, treinados com DP-SGD no conjunto IoT-23. Os modelos foram otimizados com tinyML e implementados em um Raspberry Pi 4. O modelo feedforward manteve 87% de acurácia com privacidade alta ($\epsilon = 0.5$), enquanto a otimização reduziu em até 91% o tamanho dos modelos, 82% o uso de RAM e 80% o tempo de execução. A combinação de privacidade diferencial e tinyML mostrou-se viável para segurança em dispositivos de borda.*

Abstract. *Deep learning models (DLMs) are applied to attack and anomaly detection in IoT networks. The tinyml paradigm enables local execution of these models with low resource consumption and increased privacy. However, DLMs can still leak data through adversarial attacks. This work implements a feedforward network for classification and an autoencoder for anomaly detection, both trained with DP-SGD on the IoT-23 dataset. The models were optimized with tinyml and deployed on a Raspberry Pi 4. The feedforward model retained 87% accuracy under strong privacy ($\epsilon = 0.5$), while optimization reduced model size by up to 91%, RAM usage by 82%, and execution time by 80%. The combination of differential privacy and tinyml proved feasible for edge device security.*

1. Introdução

A Internet das Coisas (IoT) tem impulsionado áreas como automação e redes inteligentes, gerando grandes volumes de dados utilizados por modelos de aprendizado profundo em aplicações de segurança [Vidal 2020]. A necessidade de execução local favorece o uso de *tinyML*, que permite operar modelos em dispositivos de baixo custo e baixa latência, sem depender da nuvem [Zhou et al. 2025]. No entanto, esses modelos lidam com dados sensíveis e estão sujeitos a vazamentos.

Mesmo com execução local, modelos de aprendizado profundo (MAP) podem sofrer ataques adversariais, expondo hábitos dos usuários [Huckelberry et al. 2024]. Para

mitigar esses riscos, exploram-se abordagens como *federated* e *split learning*, que evitam o compartilhamento direto de dados, além de mecanismos como criptografia e privacidade diferencial (PD) [Abadi et al. 2016]. A privacidade diferencial tem-se mostrado promissora pela sua capacidade de anonimizar dados e diminuir o risco de vazamento por modelos de aprendizado de máquina [Abadi et al. 2016]. No entanto, sua aplicação em cenários restritos com *tinyml* ainda é pouco explorada.

Este trabalho propõe dois MAP — um *feedforward* e um *autoencoder*, ambos para detecção de anomalia, aplicados ao conjunto de dados IoT-23 e otimizados com *tinyml*. Utiliza-se o algoritmo *Differentially Private Stochastic Gradient Descent* (DP-SGD) [Chua et al. 2024] para evitar vazamentos de dados nos modelos e se avalia o impacto da otimização em três níveis de otimização. O modelo *feedforward* manteve 87% de acurácia sob privacidade alta ($\epsilon = 0.5$), enquanto o *autoencoder* sofreu distúrbios de funcionamento com ruído. A otimização reduziu até 91% o tamanho dos modelos, 82% o uso de RAM e 80% o tempo de execução, evidenciando a viabilidade da abordagem. O restante deste artigo está organizado da seguinte forma: a Seção 2 apresenta os trabalhos relacionados; a Seção 3, os conceitos envolvidos; a Seção 4 detalha a experimentação; a Seção 5, os resultados; e a Seção 6 conclui o trabalho.

2. Trabalhos Relacionados

Diversas abordagens têm sido propostas para preservar a privacidade no contexto de *tinyml* e computação de borda. Os principais trabalhos diferem em suas aplicações, técnicas e ferramentas utilizadas. A Tabela 1 resume essas contribuições.

Tabela 1. Comparação de trabalhos relacionados

Autor	Modelos	Técnica de privacidade	Avaliação	Métricas
[Radwan et al. 2024]	*Split learning* com *TinyML*	Transmissão de ativações intermediárias	Análise de desempenho	Acurácia e eficiência energética
[Saranya et al. 2024]	*Random Forest*, *K-nearest neighbors*	Criptografia AES leve	Estudo de caso	Acurácia, precisão, *recall*, *f1-score*
[Nurmi et al. 2023]	Aprendizado federado	Privacidade diferencial local	Prova de conceito e análise	Custo computacional, precisão e nível de proteção
Proposto. 2025	*Feedforward*, *Autoencoder*	DP-SGD	Análise de *trade-off*	Erro de reconstrução, precisão, acurácia, *recall*, *f1-score*

[Radwan et al. 2024] propõem um sistema federado com *tinyML* para classificação de sentimentos em texto, com foco em eficiência energética e privacidade. A proteção é feita pela transmissão de ativações intermediárias, dificultando a reconstrução dos dados. O método foi testado em cenários sem fio com ruído, demonstrando robustez e baixo custo computacional. [Saranya et al. 2024] propõem uma arquitetura segura para IoT em saúde, utilizando *tinyml* para análise em tempo real de sinais vitais e emissão de alertas. A proteção dos dados é feita por criptografia leve (AES) e reforçada com um sistema de detecção de intrusões, sendo adequada para dispositivos com recursos limitados. [Nurmi et al. 2023] apresentam o SPHERE-DNA, que combina aprendizado federado com privacidade diferencial local para monitoramento de saúde. A proposta utiliza sensores multimodais e processamento em borda para reconhecer atividades com precisão, evitando centralização de dados e necessidade de terceiros confiáveis.

Esses trabalhos foram selecionados por abordarem a privacidade no contexto de *tinyml*, com foco em ferramentas baseadas em modelos de aprendizado profundo. Os trabalhos evidenciam aprendizado federado, criptografia e privacidade diferencial local. Em contraste, o presente trabalho aborda o uso de DP-SGD implementado em modelos *tinyml*.

3. Fundamentação Teórica

Este trabalho abrange a otimização de modelos diferencialmente privados utilizando *tinyml*. Esses conceitos são melhor explicados a seguir.

TinyML é uma abordagem da inteligência artificial que permite executar algoritmos de aprendizado de máquina localmente em dispositivos embarcados e de baixo custo, com baixo consumo energético [Dutta and Bharali 2021]. É adequada para redes IoT, sistemas industriais e aplicações em saúde, que exigem decisões rápidas com baixa latência. Ao descentralizar o processamento, reduz o tráfego com a nuvem, melhora a privacidade e elimina a dependência de servidores, viabilizando operação autônoma e em tempo real.

Essa tecnologia se baseia em otimizar modelos através de técnicas como a quantização e poda para execução em *hardware* restrito. A quantização é uma técnica de otimização que reduz o tamanho e a complexidade dos modelos ao converter operações e parâmetros de *float* 32 para formatos como *float* 16 ou *int* 8 [Hubara et al. 2021]. Essa conversão permite uma redução de até 4 vezes no tamanho de um modelo e uma inferência mais rápida, com operações realizadas com números inteiros. A quantização de 8 *bits* é mais otimizada e necessita de amostras de treino para aprender a representar os valores reais com 8 *bits*.

Privacidade diferencial é um modelo matemático que limita o quanto uma informação sensível pode ser revelada, inserindo ruído controlado nos dados para dificultar a identificação de indivíduos [Dwork 2006]. Essa proteção é regulada pelo parâmetro ϵ, que define o *trade-off* entre privacidade e utilidade: quanto menor o ϵ, maior a privacidade e menor a utilidade. No aprendizado profundo a PD é uma técnica bastante utilizada contra ataques adversariais, que visam obter dados dos modelos [Pustozerova et al. 2023]. Uma técnica de PD para MAPs é o otimizador DP-SGD, que modifica o treinamento ao cortar gradientes e adicionar ruído a cada iteração [Chua et al. 2024], reduzindo memorização e dificultando ataques adversariais, como mostra a Figura 1.

Redes neurais profundas são modelos compostos por múltiplas camadas interconectadas capazes de aprender representações complexas dos dados [Reis 2021]. Esses modelos são amplamente utilizados em tarefas como classificação e detecção de anomalias pela capacidade de generalização e adaptação a diferentes padrões. Durante o treinamento, os pesos das conexões são ajustados iterativamente por algoritmos como o *Stochastic Gradient Descent* (SGD), que calcula os gradientes da função de perda e atualiza os parâmetros com base no erro.

Feedforward é uma arquitetura com fluxo unidirecional de dados entre camadas, comum em tarefas de classificação. Escolheu-se essa arquitetura por sua simplicidade estrutural e bom desempenho com baixo custo computacional [Bezerra 2016].
Autoencoder é uma arquitetura usada para compressão e reconstrução de dados, composta por *encoder*, espaço latente e *decoder* [Li et al. 2023]. Esse modelo realiza treinamento

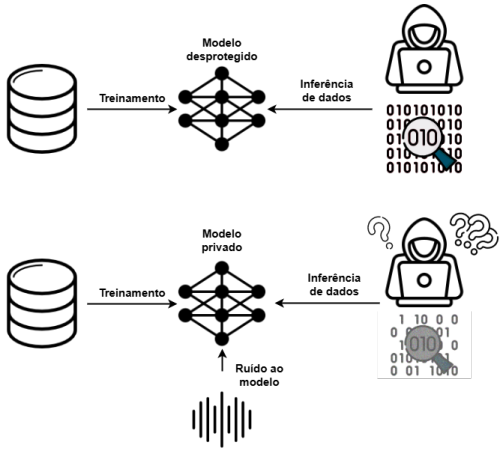

Figura 1. Representação de modelo diferencialmente privado

com dados não rotulados. Os dados aprendidos são considerados ”dados”normais, que o modelo consegue reconstruir com precisão. A diferença entre o valor de reconstrução e o valor real é chamado erro de reconstrução. Esse valor é usado para detecção de anomalias, com valores indicando dados fora da normalidade.

4. Metodologia da Experimentação

Conjunto de dados: Para avaliar os modelos propostos, utilizou-se o conjunto de dados público IoT-23, composto por 23 capturas de tráfego IoT com base em conexões IP. Foram selecionadas amostras de múltiplos cenários para formar um conjunto mais representativo, com as classes *Port Scanning*, *Okiru*, *DDoS*, *C&C*, *Attack* e tráfego benigno. Os atributos utilizados incluem portas, protocolos, estados de conexão e duração. A divisão dos dados seguiu a proporção de 50% para treino, 30% para teste e 20% para validação.

Modelos: A abordagem foi implementada com os modelos *feedforward e autoencoder*, arquiteturas que interagem com os dados de formas diferentes. A escolha dos modelos é para avaliar o DP-SGD em diferentes arquiteturas. A Tabela 2 resume os parâmetros dos modelos utilizados. O *feedforward* classifica o tráfego entre as classes e o *autoencoder* detecta anomalias. O *autoencoder* gera erros de reconstrução para os dados de entrada, um valor usado para comparar tráfego benigno e maligno. Valores mais altos de erros indicam possíveis anomalias.

Tabela 2. Arquiteturas utilizadas

Modelo	Camadas	Ativação oculta	Ativação saída	Épocas
Feedforward	5	ReLU	*Softmax*	15
Autoencoder	6	ReLU	*Sigmoid*	25

DP-SGD e Tinyml: A solução utiliza o algoritmo *Differentially Private ADAM* (DP-ADAM), uma variação do DP-SGD que incorpora médias ponderadas dos gradientes para treinar o modelo com garantias de privacidade. Implementado com a biblioteca *TensorFlow Privacy*, o otimizador aplica cortes nos gradientes e insere ruído controlado a cada iteração, regulado por três parâmetros principais: norma de corte, multiplicador de ruído e número de *micro-batch* [TensorFlow Privacy Team 2024]. Níveis menores de norma de

corte e maiores de ruído resultam em menor ϵ, ou seja, maior privacidade. Os modelos foram treinados com $\epsilon = 0.5$, 1.0 e 2.0.

Utilizou-se o *tensorflow lite* para a otimização dos modelos. Eles foram convertidos em dois formatos: otimização padrão *tflite*, que apenas otimiza o modelo; e quantização *int 8*, em que converteu-se a representação de dados de valores decimais de 32 *bits* para inteiros de 8 *bits*. A otimização padrão é uma conversão de modelo que permite a implementação de outros métodos de quantização. Para otimizar modelos diferencialmente privados, criou-se uma função personalizada para carregar o otimizador e a função de ativação em um arquivo *keras*.

Cenário de inferência: Os modelos diferencialmente privados são implementados no dispositivo *Raspberry Pi* 4. O experimento consiste no *Raspberry Pi* recebendo o tráfego dos dispositivos IP do *dataset* IoT-23 e realizando o papel de *gateway*, com os modelos implementados. A Figura 2 ilustra o ambiente do experimento, com os modelos recebendo tráfego dos dispositivos.

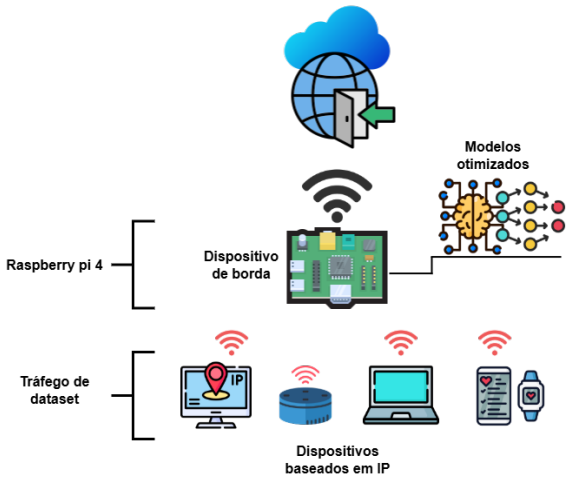

Figura 2. Cenário do experimento

Cada modelo foi implantado individualmente no dispositivo e executado separadamente em diferentes experimentos. O *Raspberry Pi 4* foi configurado para realizar inferências com amostras pré-processadas do conjunto de teste, simulando o tráfego de dispositivos IoT na rede. Implementou-se um *pipeline* com a biblioteca *tflite runtime* que importa os modelos no dispositivo, preparando o ambiente para execução. Durante os testes, avaliou-se as métricas tempo médio de inferência, uso de memória RAM e tamanho do modelo. Para cada arquitetura, foram testadas uma versão *baseline* (sem privacidade) e três versões com privacidade diferencial, utilizando $\epsilon = 0.5$, 1.0 e 2.0.

5. Análise dos Resultados

No modelo *autoencoder*, as três versões privadas apresentaram resultados semelhantes em termos de previsão, indicando que o uso de *DP-SGD* impactou pouco na qualidade da reconstrução. Por isso, as análises comparativas seguintes utilizaram apenas $\epsilon = 2.0$. A Tabela 3 apresenta as métricas do modelo *feedforward* para os diferentes níveis de ϵ. Os resultados revelam a relação direta entre o nível de privacidade e a utilidade do modelo, evidenciando o *trade-off* entre privacidade e desempenho.

Tabela 3. Métricas do *feedforward* e suas versões

ϵ	acurácia	precisão	*recall*	*f1-Score*
baseline	0.93	0.92	0.93	0.92
2.0	0.89	0.89	0.89	0.87
1.0	0.88	0.87	0.88	0.86
0.5	0.87	0.87	0.87	0.85

A Figura 3 apresenta a saída do *autoencoder* para tráfegos benigno (esquerda) e malicioso (direita). As barras representam os erros de reconstrução, e alturas acima do limiar de 0.16 indicam possíveis anomalias. Esse valor foi definido empiricamente com base no comportamento do conjunto de dados malicioso.

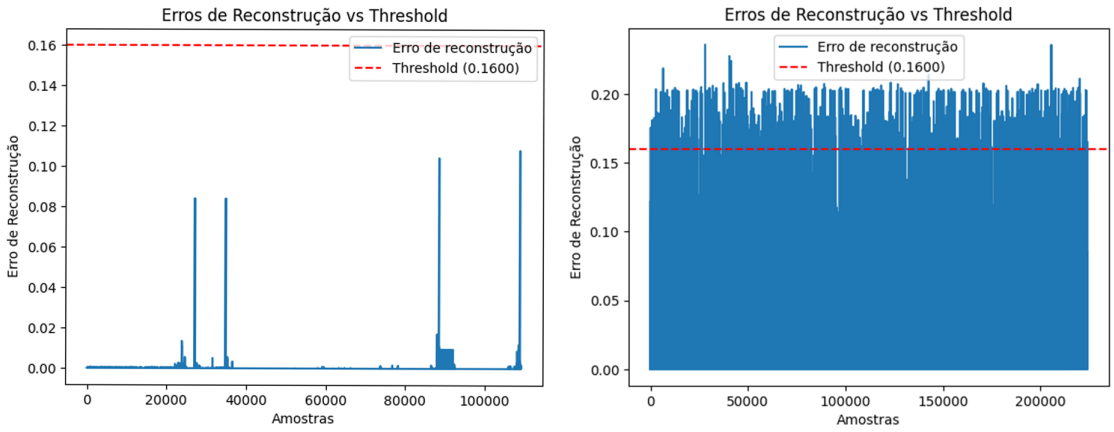

Figura 3. Análise de tráfego benigno e tráfego maligno

A Figura 4 exibe a saída do *autoencoder* com privacidade diferencial. Observa-se um padrão anômalo nas reconstruções, indicando que o ruído do DP-SGD prejudicou a capacidade do modelo de generalizar os dados. Esse comportamento, consistente nos três níveis de ϵ, dificulta a distinção entre tráfego benigno e malicioso com base no erro de reconstrução.

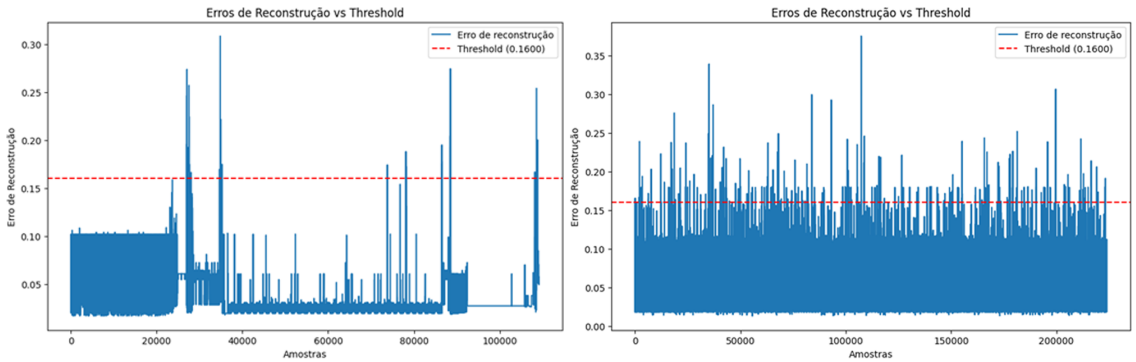

Figura 4. Análise de tráfego benigno e tráfego maligno por modelo privado

A Tabela 4 compara o desempenho dos modelos nos formatos *keras* (normal), *tflite* e quantização int8, inferindo 7000 amostras. O *autoencoder* consumiu mais recursos e tempo, especialmente no formato *keras*. Os formatos otimizados (*tflite* e quantizado) reduziram consumo e tempo, embora o modelo quantizado tenha levado mais tempo devido ao carregamento e conversão das amostras.

Tabela 4. Desempenho dos modelos otimizados (*Feedforward* / *Autoencoder*)

Formato	Tempo (seg)	RAM (MB)	Tamanho (MB)
Keras	35.22 / 36.39	612.21 / 630.41	0.18 / 0.24
tflite	06.60 / 07.09	112.18 / 112.29	0.05 / 0.06
Quantizado	12.86 / 13.66	112.28 / 112.13	0.02 / 0.02

Discussão: Os resultados mostram que técnicas de privacidade diferencial são viáveis para modelos de classificação otimizados com *tinyml* , como o *feedforward*. No entanto, deve haver uma análise do quanto a privacidade desejada pode prejudicar a utilidade. Um modelo que atinge acima de 80% de acurácia pode ser considerado aceitável, havendo alto nível de privacidade. O *trade-off* entre privacidade e utilidade é evidente na queda de performance conforme o ϵ diminui. O *autoencoder* apresentou comportamento irregular, indicando a inviabilidade com DP-SGD. Além disso, a otimização para dispositivos de borda permite desempenho semelhante a modelos robustos com menor consumo de recursos, reduzindo complexidade e acelerando a inferência.

6. Conclusão

Este trabalho apresentou uma abordagem para detecção de ataques e anomalias em tráfego IoT usando privacidade diferencial e otimização via *tinyml*. Utilizou-se o conjunto IoT-23 para treino e validação, com testes no *Raspberry Pi 4*. O classificador *feedforward* diferencialmente privado teve desempenho aceitável, apesar da taxa considerável de falsos positivos, e mostrou viabilidade para execução em ambientes de baixo poder computacional no formato *tflite*. Já o modelo de detecção de anomalias apresentou comportamento inesperado, com erros de reconstrução em amostras benignas, indicando a necessidade de ajustes, embora seu custo computacional seja baixo. Pode-se concluir que soluções para ambientes restritos podem utilizar modelos de aprendizado profundo diferencialmente privados para classificação de tráfego, havendo apenas a necessidade de ajuste de parâmetros. Outra conclusão é a inviabilidade de *autoencoders* de reconstrução nesses ambientes.

Referências

Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. (2016). Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318.

Bezerra, E. (2016). Introdução à aprendizagem profunda. *Artigo–31º Simpósio Brasileiro de Banco de Dados–SBBD2016–Salvador*.

Chua, L., Ghazi, B., Kamath, P., Kumar, R., Manurangsi, P., Sinha, A., and Zhang, C. (2024). How private is dp-sgd? *arXiv preprint arXiv:2403.17673*.

Dutta, L. and Bharali, S. (2021). Tinyml meets iot: A comprehensive survey. *Internet of Things*, 16:100461.

Dwork, C. (2006). Differential privacy. In *International colloquium on automata, languages, and programming*, pages 1–12. Springer.

Hubara, I., Nahshan, Y., Hanani, Y., Banner, R., and Soudry, D. (2021). Accurate post training quantization with small calibration sets. In *International Conference on Machine Learning*, pages 4466–4475. PMLR.

Huckelberry, J., Zhang, Y., Sansone, A., Mickens, J., Beerel, P. A., and Reddi, V. J. (2024). Tinyml security: Exploring vulnerabilities in resource-constrained machine learning systems. *arXiv preprint arXiv:2411.07114*.

Li, P., Pei, Y., and Li, J. (2023). A comprehensive survey on design and application of autoencoder in deep learning. *Applied Soft Computing*, 138:110176.

Nurmi, J., Xu, Y., Boutellier, J., and Tan, B. (2023). Sphere-dna: Privacy-preserving federated learning for ehealth. In *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1–6. IEEE.

Pustozerova, A., Baumbach, J., and Mayer, R. (2023). Differentially private federated learning: Privacy and utility analysis of output perturbation and dp-sgd. In *2023 IEEE International Conference on Big Data (BigData)*, pages 5549–5558. IEEE.

Radwan, A. Y., Shehab, M., and Alouini, M.-S. (2024). Tinyml nlp approach for semantic wireless sentiment classification. *arXiv preprint arXiv:2411.06291*.

Reis, C. H. (2021). Otimização de hiperparâmetros em redes neurais profundas. *Minas Gerais*.

Saranya, T., Jeyamala, D., Indra Priyadharshini, S., et al. (2024). A secure framework for miot: Tinyml-powered emergency alerts and intrusion detection for secure real-time monitoring. In *2024 8th International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*, pages 13–21. IEEE.

TensorFlow Privacy Team (2024). Dpkerasadamoptimizer: Differentially private adam optimizer for keras. `https://github.com/tensorflow/privacy`. Acesso em: 10 abr. 2025.

Vidal, I. d. C. (2020). Protecting: garantindo a privacidade de dados gerados em casas inteligentes localmente na borda da rede.

Zhou, H., Zhang, X., Feng, Y., Zhang, T., and Xiong, L. (2025). Efficient human activity recognition on edge devices using deepconv lstm architectures. *Scientific Reports*, 15(1):13830.