A Survey on Distributed Tiny ML

AUTHORS
Rok Štular
Supervisor: Mingkun Yang
Responsible Professor: Qing Wang

AFFILIATION
TU Delft

1. INTRODUCTION

The rapid growth in data from interconnected devices necessitates new processing methods, with **Distributed Tiny Learning** (DTL) offering a solution by distributing machine learning tasks across edge devices to **enhance data security and privacy**, improve **latency**, and take advantage of the distributed computing power. The advent of 6G networks can **boost DTL's efficiency and scalability,** enabling **real-time decision-making** for applications like autonomous vehicles and smart cities. This paper surveys DTL, maps the ecosystem and evaluates two DTL algorithms to provide practical insights into their effectiveness.

2. OBJECTIVE

The goal is to explore the state-of-the-art in DTL, identify its potential benefits and challenges, and analyze its practical applications for advancing 6G applications. Additionally, two DTL algorithms are implemented and benchmarked to demonstrate their effectiveness and viability.

3. METHODOLOGY

The survey follows a prescribed form [1], and is split into several stages:

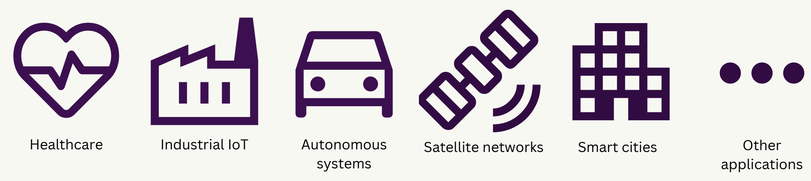
- Identification of Relevant Databases and Search Terms
- Application of Inclusion and Exclusion Criteria
- Screening and Selection Process
- Data Extraction and Synthesis
- Quality Assessment
- Analysis and Interpretation

4. SURVEY FINDINGS

- Different training approaches, like federated learning and collaborative learning, each have privacy, scalability, and decentralization advantages. [2]
- Network topologies can be centralized (Figure 1), decentralized (Figure 2), swarm, or split, impacting performance and fault tolerance. [2]
- Training data can be partitioned in various ways (data-parallel, model-parallel, or hybrid) to optimize resource usage. [2]
- Scalability and fault tolerance are crucial for real-world deployments, allowing DTL networks to incorporate additional devices and withstand failures without performance degradation. [3]
  - Scalability is achieved through techniques like hierarchical clustering, decentralized aggregation, and hardware-aware scaling, which optimize resource utilization and reduce communication overhead.
  - Fault tolerance is maintained by secure federated learning methods that ensure resilience against device failures and malicious activity, ensuring continuous operation even under adverse conditions.
- Communication efficiency can be improved by several means, ranging from network design to specific information sharing techniques. Model gossiping and protocol optimisations help reduce communication overhead, while quantisation, sparsification, and compression reduce the size of model updates. [4]

5. APPLICATIONS OF DISTRIBUTED TINYML

In **healthcare**, DTL enhances data privacy and diagnostic efficiency through federated learning, while in **industrial IoT**, it improves predictive maintenance and resource optimization. **Autonomous systems** benefit from low-latency data processing for quick decision-making, **smart cities** use it for real-time urban management, and **satellite networks** leverage it to reduce communication latency and enhance environmental monitoring. In addition, there are many other areas that benefit from application of DTL. [5]



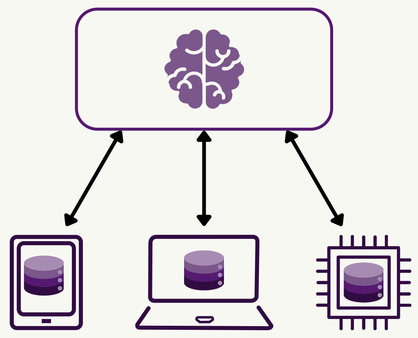


Figure 1: centralized topology

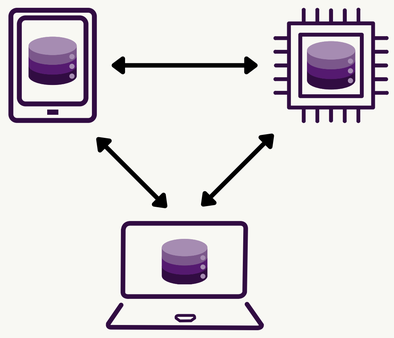


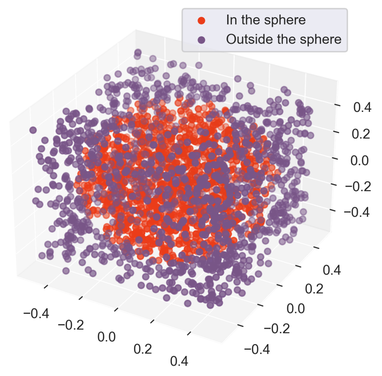
Figure 2: decentralized topology

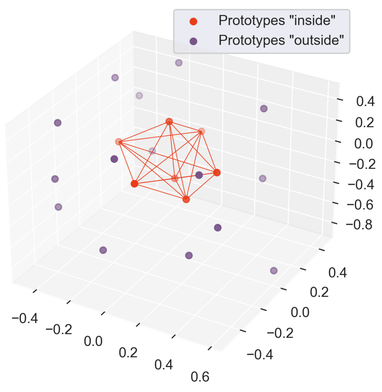
6. ALGORITHM BENCHMARK

The first implemented algorithm was **Incremental Learning Vector Quantisation** (ILVQ) [6]. It maintains **only a few prototypes** (analogous to centroids in the k-nearest neighbours algorithm) in memory, which are gradually updated throughout the training process, and later used for classification.

The model was tasked with classifying synthetically-generated points in 3-dimensional space into two categories: one for points within a sphere with the radius of 1 and another for points outside the sphere (right, top).

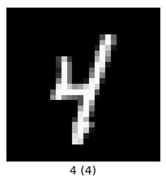
The trained model was able to classify the data points with at least 85% accuracy, by adapting the “prototypes” to the shape of the input dataset (right, bottom).

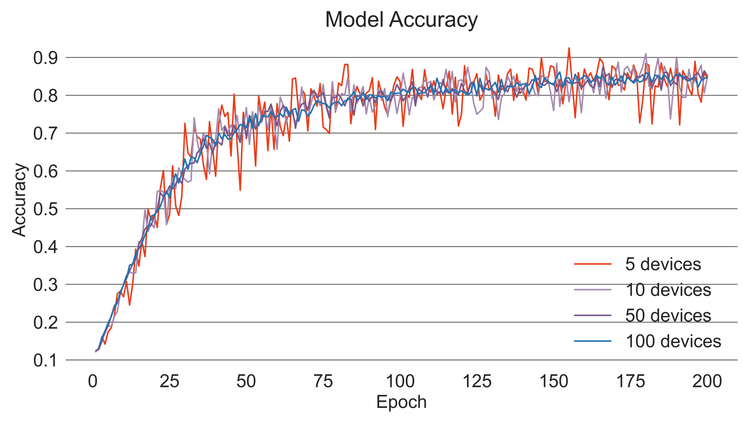




The second implemented algorithm was a simple **neural network classifier**, comprising a **dense** and a **softmax layer**, tasked with classifying the digits from the MNIST handwritten digit dataset (two examples are shown below, left). The implementation was written using the TensorFlow federated machine learning framework.

The accuracy of the model was found to be consistent across all setups and sufficiently high (**around 85%** across all setups - bottom, right), indicating the robustness of the algorithm in distributed learning environments.





7. FUTURE DIRECTIONS & CONCLUSION

Future research directions in distributed TinyML can explore several promising avenues: integration with **emerging technologies** (6G, edge AI), **hardware advancements**, **security** and **privacy** enhancements, federated learning for **non-IID data**, and **explainability** and interpretability of the models.
By addressing these research areas, distributed TinyML can evolve into a cornerstone technology for intelligent and efficient data processing at the edge, paving the way for a more interconnected and intelligent world.

RELATED LITERATURE

[1] Barbara Kitchenham and Stuart Charters. Guidelines for performing Systematic Literature Reviews in Software Engineering. 2, January 2007.
[2] Joost Verbraeken, Matthijs Wolting, Jonathan Katzy, Jeroen Kloppenburg, Tim Verbelen, and Jan S. Rellermeyer. A Survey on Distributed Machine Learning. ACM Computing Surveys, 53(2):1–33, March 2021. arXiv:1912.09789 [cs, stat]
[3] Jiashi Feng, Huan Xu, and Shie Mannor. Distributed Robust Learning, February 2015.
[4] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data, January 2023. arXiv:1602.05629 [cs].
[5] Rakhee Kallimani, Krishna Pai, Prasoon Raghuwanshi, Sridhar Iyer, and Onel L. A. Lopez. TinyML: ´ Tools, applications, challenges, and future research directions. Multimedia Tools and Applications, 83(10):29015–29045, March 2024.
[6] Ye Xu, Furao Shen, and Jinxi Zhao. An incremental learning vector quantization algorithm for pattern classification. Neural Computing and Applications, 21(6):1205–1215, September 2012.