A Survey on Distributed Tiny Machine Learning

Exploring Techniques, Applications, Challenges, and Future Directions in Distributed Tiny Machine Learning

Rok Štular[1]

Supervisor(s): Qing Wang[1]**, Mingkun Yang**[1]

[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 23, 2024

Name of the student: Rok Štular
Final project course: CSE3000 Research Project
Thesis committee: Qing Wang, Mingkun Yang, Johan Pouwelse

Abstract

The explosive growth in data collection driven by the proliferation of interconnected devices necessitates novel approaches to data processing. Traditional centralised data processing methods are increasingly inadequate due to the sheer volume of data generated. Distributed Tiny Learning (DTL) offers a compelling solution by distributing machine learning tasks across multiple edge devices and processing data locally, thus minimising the need for data transmission to central servers. This approach is particularly beneficial in scenarios with limited network bandwidth and stringent privacy requirements, enhancing data security and compliance with privacy regulations. The advent of 6G networks, with their promise of unprecedented speed, capacity, and reliability, can further amplify the power of DTL. By providing higher bandwidth and lower latency, 6G enables more efficient data processing and communication among edge devices, thereby enhancing the overall performance and scalability of DTL systems. This integration supports real-time decision-making for applications such as autonomous vehicles, smart cities, and healthcare monitoring.

This paper conducts a comprehensive survey of the state-of-the-art in DTL, categorising scientific literature, mapping the ecosystem and tools, and addressing performance, efficiency, and scalability challenges in ultra-low-power devices within a 6G context. Additionally, it implements and benchmarks two DTL algorithms, providing practical insights into their effectiveness and operational viability.

1 Introduction

In the modern technology landscape, data collection has experienced incredible growth. As the number of deployed interconnected devices increases, so does the need to process the produced and collected data.

While data processing was traditionally performed in a centralized fashion, the sheer volume of collected data points has shifted the focus to distributed data processing techniques - machine learning being no exception to that trend [1].

The concept of Distributed Tiny Learning involves distributing the learning process across multiple edge devices, which process data locally, thereby minimizing the need to transmit sensitive information to a central server. This approach is particularly vital in scenarios where network bandwidth is limited and privacy is a concern, making it highly relevant for modern IoT deployments. The ability to process data at the edge not only reduces latency but also enhances data security and compliance with privacy regulations by keeping sensitive information on local devices. This paradigm shift is essential in the era of the Internet of Things (IoT), where countless devices continuously generate and process vast amounts of data [2].

The growing interest in Distributed Tiny Learning aligns with the evolving requirements of 6G networks, which promise to deliver unprecedented speed, capacity, and reliability. [3] These networks are expected to support a vast array of IoT devices, demanding efficient, scalable, and secure data processing solutions. Edge computing and Distributed Tiny Learning are poised to play critical roles in realizing the full potential of 6G by enabling intelligent data processing directly on IoT devices [4]. This shift not only optimizes network usage but also supports real-time decision-making processes, which are crucial for applications such as autonomous vehicles, smart cities, and healthcare monitoring systems [5].

The works of Lin et al. [6] and Gonzalez-Soto et al. [7] explore frameworks for decentralized Tiny learning; however, most existing studies lack a detailed exploration of the practical integration of these technologies into real-world IoT applications for 6G environments, which would demonstrate their true viability and performance benefits.

This project aims to conduct a comprehensive survey and categorize the state-of-the-art scientific literature, map out the ecosystem and available tools, and specifically target the performance, efficiency, and scalability challenges of implementing Distributed Tiny Learning on ultra-low-power devices within a 6G context.

In addition, two algorithms for distributed TinyML will be implemented, evaluated, and benchmarked to provide practical insights into their effectiveness and operational viability. This dual approach aims to bridge the gap between theoretical frameworks and real-world applications, offering concrete evidence of the benefits and challenges associated with Distributed Tiny Learning in advanced network environments.

2 Methodology

The methodology chosen for this literature review systematically identifies, selects, and analyses relevant literature on distributed TinyML techniques, applications, challenges, and future directions. Following the guidelines established by Kitchenham and Charters [8], it consists of six distinct steps designed to ensure a comprehensive and representative collection of scientific literature for the research:

1. **Identification of Relevant Databases and Search Terms:** A set of relevant databases, including IEEE Xplore[1], ACM Digital Library[2], and Google Scholar[3] were identified for conducting literature searches. Out of those, Google Scholar was not used as a primary literature source but rather as an augmentation tool to enhance the results obtained through the first three databases.
   A combination of keywords and search terms related to distributed TinyML were used to retrieve relevant articles. The exact query is presented in subsection A.1.
2. **Inclusion and Exclusion Criteria:** Articles considered for inclusion were those published in peer-reviewed journals, conference proceedings, and relevant technical reports. The inclusion criteria include studies focusing

[1] https://ieeexplore.ieee.org/Xplore/home.jsp
[2] https://dl.acm.org/
[3] https://scholar.google.com/

on distributed machine-learning techniques specifically tailored/adapted for resource-constrained devices. Exclusion criteria involved studies unrelated to TinyML, non-English publications, and any articles lacking substantial relevance to the scope of this review.

3. **Screening and Selection Process:** Initial screening involved the assessment of titles and abstracts to identify potentially relevant articles. Full-text screening was conducted for articles passing the initial screening phase, applying the inclusion and exclusion criteria to determine their final eligibility.

4. **Data Extraction and Synthesis:** Relevant data from selected articles were extracted systematically, including information on distributed TinyML algorithms, architectures, applications, performance metrics, and challenges. Data synthesis involved organising extracted information thematically, enabling a coherent presentation of key findings, trends, and gaps in the literature.

5. **Quality Assessment:** The quality of selected articles was assessed based on factors such as research rigour, methodology clarity, experimental design, and contribution to the field of distributed TinyML. High-quality studies were accorded greater weight in the synthesis of findings and conclusions.

6. **Analysis and Interpretation:** Analytical techniques such as thematic analysis and comparative evaluation were employed to interpret the synthesised data, identify recurring patterns, and elucidate emerging trends in distributed TinyML research. Critical insights derived from the analysis were used to formulate conclusions and propose avenues for future research.

The implementation of two algorithms is based on the findings of the literature review. As a part of the research process, within the "Data Extraction and Synthesis" step, two suitable algorithms for distributed tiny machine learning were selected as implementation candidates. They were chosen based on specific criteria such as popularity, performance, practical applicability, and scalability.

One of the implementations will target the Arduino Nano 33 BLE[4] board, as it is a prevalent development platform for deployed tiny edge devices [9]. The other implementation will target the more powerful Raspberry Pi computer, an immensely popular platform for edge-deployed devices [10]. The programming language of choice will be C/C++, a popular language for embedded software development [11] and Python, a popular framework for machine learning tasks [12].

The development process will start with the initial prototype implementation on a general-purpose laptop before being transferred to the resource-constrained microcontroller. After the initial microcontroller implementation is complete, an iterative process will follow, during which the algorithm implementations will be refined to approach or surpass the benchmark results in the available literature.

[4]https://store.arduino.cc/products/arduino-nano-33-ble

3 Related work

A survey by Verbraeken et al. provides a comprehensive examination of distributed machine learning, driven by the escalating demands for artificial intelligence and the limitations of hardware acceleration in processing extensive training data [13].

González-Soto et al. introduce a novel decentralised and collaborative machine learning framework designed specifically for resource-constrained IoT devices, offering an alternative to traditional federated learning models with enhanced security features [7].

McMahan et al. describe a decentralised approach to machine learning where training data remains on mobile devices, maintaining privacy while harnessing rich, user-generated data to enhance device functionalities through improved language and image models. This method involves iterative model averaging and has been empirically validated across various architectures and datasets, showing resilience against unbalanced and non-IID (non-Independent and Identically Distributed) data distributions typical of this setup [14].

A survey performed by Peteiro-Barral and Guijarro-Berdiñas provides an overview of distributed learning methods, highlighting their advantages in scaling up learning algorithms and dealing with naturally distributed datasets encountered in real-world applications [15].

Rajapakse et al. present a comprehensive survey of reformable TinyML solutions, showcasing a new taxonomy that evaluates the adaptability of models on microcontroller units (MCUs) post-deployment [16].

A survey by Tsoukas et al. categorises prevalent optimisation techniques for Neural Network compression alongside an overview of available development boards and TinyML software [17].

Additionally, another survey by Tsoukas et al. explores the potential of TinyML technology in revolutionising healthcare applications by enabling on-device data processing, thereby eliminating the need for data transmission to external servers [18].

A paper by Lakshman and Eisty identifies key obstacles faced by TinyML developers and investigates state-of-the-art Software Engineering approaches tailored to the unique demands of TinyML engineering [19].

4 Results

The results of this literature survey are presented by examining various aspects that serve as key differentiating factors among different algorithms and frameworks. These aspects were identified through the analysis of overarching categories and themes within the surveyed literature.

First, different model training approaches are explored in subsection 4.1. In subsection 4.2, various network configurations that support DTL are analysed. Methods used to divide and allocate data across multiple edge devices are examined in subsection 4.3. In subsection 4.4 the scalability of DTL systems and their resilience to faults are investigated. In subsection 4.5, an overview of the practical applications of DTL across various domains is provided. Finally, in subsection 4.6

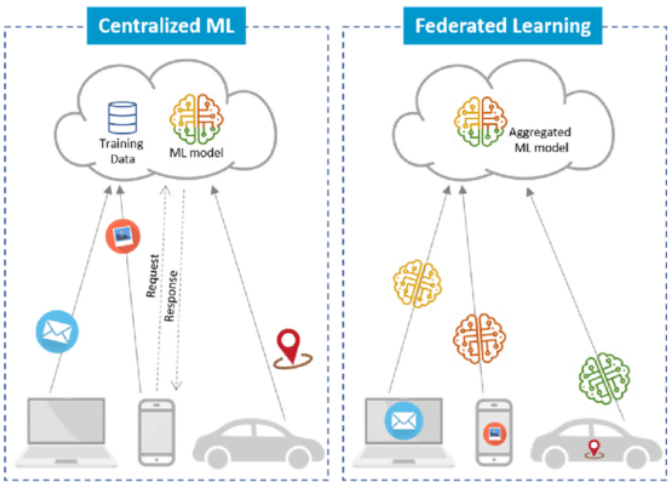

Figure 1: Collaborative learning and federated learning [32]

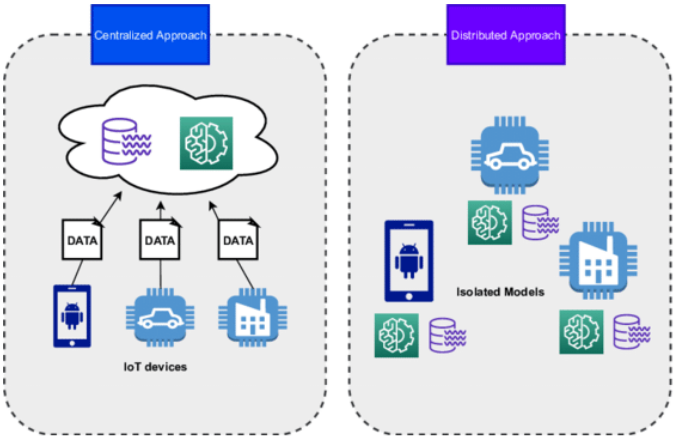

Figure 2: Centralised learning and decentralised learning [41]

the efficiency of communication protocols within DTL systems is addressed.

4.1 Model Training Approach and Architecture

Model training approaches in distributed TinyML vary and can be categorised into federated, collaborative, swarm, and split/hybrid learning. These methods determine how data is handled during training and how model weight updates are propagated within the network. Each approach has its advantages regarding privacy, scalability, and decentralisation [20; 21; 22; 23].

Federated Learning

Federated learning involves training models locally on edge devices using their data and only sending model updates to a central server, ensuring data privacy. This approach has gained popularity due to stringent privacy laws and its efficiency in reducing the load on central servers, especially in the context of 6G networks. Research indicates that federated learning is particularly advantageous in scenarios involving sensitive data, such as healthcare, where privacy is paramount [24; 25; 26; 27; 28]. Moreover, federated learning reduces latency and improves the system's scalability by minimising the reliance on a central server [21].

Collaborative Learning

Collaborative learning, as depicted in Figure 1, differs from federated learning in the means of training. In the former, training datasets are exchanged between devices, which aids in balancing the loads across the network and achieving scalability. It is particularly effective in heterogeneous networks where load distribution based on device capabilities is crucial [29; 30; 31]. This approach also facilitates faster convergence of the learning process by leveraging the computational power of multiple devices [23].

Swarm Learning

Swarm learning eliminates the need for a centralised server by using a decentralised data distribution channel for model updates and data sharing. An example of such a mechanism is blockchain, which has been demonstrated to be a viable option when implementing an edge-based machine-learning swarm [33; 34]. This peer-to-peer communication model enhances privacy and scalability [35; 20; 36]. The fully decentralised nature of swarm learning also enhances fault tolerance and reduces the risk of single points of failure [37].

Split/hybrid Learning

Split learning, or hybrid learning, involves dividing a model into parts and training them independently, often combining different distributed learning approaches. This method leverages collaborative and federated learning benefits, balancing performance and privacy [38; 39; 40]. Split learning can also be adapted to varying network conditions and device capabilities, making it highly flexible [22].

4.2 Network Topology

Network topology plays a crucial role in determining the performance and efficiency of distributed TinyML systems. The topology affects how data and model updates are propagated through the network, influencing latency, scalability, and fault tolerance.

Centralised Topology

In a centralised topology, all edge devices communicate with a central server that coordinates the training process and aggregates model updates. While this topology is straightforward and easy to implement, it suffers from scalability issues and potential single points of failure [21]. Centralised topologies are often used in small-scale networks or where a powerful central server is available [42; 43; 14].

Decentralised Topology

Decentralised topologies eliminate the need for a central server by enabling direct communication between edge devices. This approach enhances scalability and fault tolerance, as there is no single point of failure. In addition, decentralised designs reduce the communication load in the network, as there are fewer opportunities for communication bottlenecks to develop [44], which can, in turn, lower the overall latency of the network communication and information prop-

agation [45]. Peer-to-peer networks and blockchain-based systems are common decentralised topologies [46]. Decentralised topologies are particularly useful in environments where devices are mobile or intermittently connected [47; 48; 49].

Hybrid Topology

Hybrid topologies combine elements of both centralised and decentralised approaches. For example, a hierarchical structure where groups of edge devices communicate with local aggregators, which in turn communicate with a central server, can balance the benefits of both approaches [50]. Hybrid topologies are adaptable and can be optimised for specific application requirements and network conditions [39; 51].

4.3 Data Partitioning

Data partitioning strategies in distributed TinyML can be classified into data-parallel, model-parallel, and hybrid approaches. These strategies determine how data is divided and distributed among edge devices for processing.

Data Parallel Partitioning

Data parallel processing involves dividing the entire dataset into smaller, complete data samples, and distributing those samples across multiple devices or processors. Each device or processor then works on its assigned portion of the data independently. Zhao et al. discuss various methods of data-parallel machine learning, focusing on the synchronisation and communication required for model aggregation. They highlight three primary communication mechanisms: bulk synchronous parallel, asynchronous parallel, and stale synchronous parallel [52].

Fan et al. introduce a loss function weight reorder stochastic gradient descent method, which enhances accuracy and performance compared to traditional methods [53]. Additionally, Wang et al. propose incorporating device reputation into model aggregation, significantly improving performance on non-IID datasets in federated learning environments [54]. Deb et al. report a substantial reduction in CPU usage through their optimised data-parallel learning techniques [55].

Model Parallel Partitioning

Model parallel partitioning involves splitting different parts of a single model across multiple devices or processors. Each device then works on a specific portion of the model, allowing for more efficient processing of large models that may not fit entirely on a single device. The difference between the model parallel and data parallel approach is shown in Figure 3. This approach can reduce the memory and CPU consumption on individual devices, as highlighted by Jeon et al. and Zhang et al. [50; 56]. However, it introduces challenges related to synchronisation and communication overheads.

Jeon et al. proposed a privacy-sensitive model parallel framework that enhances data privacy by processing sensitive features on more secure devices and less sensitive features on less secure devices. This framework addresses privacy concerns in distributed learning environments while leveraging the benefits of model parallelism [50].

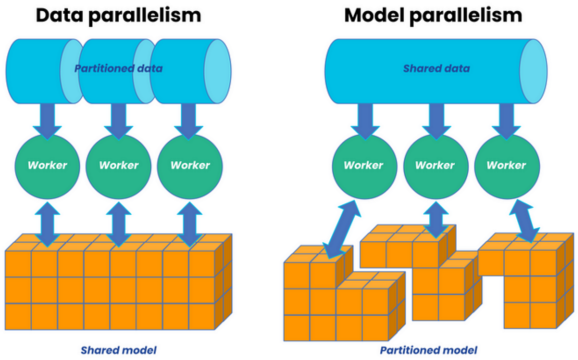

Figure 3: Data-parallel and model-parallel partitioning [57]

Zhang et al. discussed split learning, where a model is partitioned into multiple segments, each trained on a different device. This approach reduces the memory footprint on individual devices and allows for training more complex models on resource-constrained edge devices. It also facilitates a more balanced workload distribution, enhancing the efficiency of distributed training processes [56].

Advancements in model compression techniques, such as quantisation and pruning, further optimise model parallel partitioning by reducing model size without significantly impacting accuracy. These techniques lower computational and memory requirements, enabling the deployment of sophisticated models on edge devices with limited resources [58].

Hybrid Partitioning

Hybrid partitioning combines data-parallel and model-parallel techniques to maximise resource utilisation and efficiency [59]. This approach can adapt to varying workloads and device capabilities, providing a balanced solution for complex distributed learning tasks [60; 61].

4.4 Scalability and Fault tolerance

Scalability and fault tolerance are critical factors in the deployment of distributed TinyML systems. Scalability refers to the system's ability to efficiently incorporate additional devices without significant performance degradation, while fault tolerance pertains to the system's resilience against device failures and malicious nodes. Effective fault tolerance mechanisms and load-balancing strategies are essential to maintaining these attributes [62].

Scalability

Scalability is critical for integrating an increasing number of devices without performance degradation. Techniques for large-scale optimisation in distributed machine learning, especially within 6G networks, facilitate the integration of additional ultra-low-powered devices [63; 30]. Methods such as hierarchical clustering and decentralised aggregation improve scalability by reducing the communication overhead [64; 65].

Furthermore, we can optimise resource utilisation across the entire distributed TinyML network by leveraging tech-

niques like hardware-aware scaling. This approach dynamically tailors the computational demands of individual inference tasks to the specific capabilities of the underlying hardware on each device [66].

Fault Tolerance

Fault tolerance ensures the resilience and continuous operation of distributed TinyML systems despite device failures and malicious nodes. Research has shown that distributed learning frameworks can maintain performance even under adverse conditions, such as packet loss or malicious gradient submissions [67; 68; 69]. Techniques such as secure and privacy-enhanced federated learning (SPEFL) protect against malicious nodes by verifying the integrity of model updates [68].

4.5 Use Cases and Applications

The practical applications of distributed TinyML span various domains, highlighting its versatility and potential impact. Key use cases include healthcare, autonomous systems, smart cities, and industrial Internet of Things (IoT). Each area benefits from the unique advantages of distributed TinyML, such as improved data privacy, reduced latency, and enhanced real-time processing capabilities [70].

Healthcare

Healthcare is a prominent field where distributed TinyML is making significant strides. The proliferation of IoT-enabled medical devices has created a demand for efficient data processing methods that can handle the sensitive nature of medical information. Distributed TinyML offers a solution by enabling localised data processing, thereby preserving patient privacy while still leveraging the power of machine learning for diagnostic and monitoring purposes [71].

Federated learning is particularly advantageous in healthcare applications due to its privacy-preserving design. This method allows hospitals and medical devices to collaboratively train models without sharing raw data, ensuring compliance with privacy regulations such as HIPAA. Surveys by Elayan et al. have highlighted the adoption of federated learning in healthcare, showcasing its potential in applications like predictive diagnostics and remote patient monitoring [72; 73].

Industrial IoT

Industrial IoT (IIoT) encompasses various applications where distributed TinyML can enhance operational efficiency and predictive maintenance. In agricultural settings, distributed learning approaches have been tailored to monitor crop health and optimise resource usage. Devaraj et al. demonstrated a distributed learning approach specifically designed for agricultural applications, showcasing its effectiveness in rural areas where centralised data processing may not be feasible [26].

Predictive maintenance is another crucial application in IIoT, where the timely detection of equipment failures can prevent costly downtimes. Jiang et al. introduced a federated learning mechanism that utilises IoT-based data to perform high-accuracy predictive maintenance. By leveraging data from various sensors and devices across an industrial setup, this approach minimises the risk of failures and extends the lifespan of equipment [74].

Autonomous Systems

Autonomous systems, particularly in vehicular networks, benefit significantly from distributed TinyML. The ability to process data locally on edge devices such as vehicles ensures low latency and quick decision-making, critical for safe and efficient operations [75]. Research also explores the use of machine learning in vehicular networks supported by 6G infrastructure. The study compares different machine learning algorithms' ability to detect problematic driving patterns, enhancing road safety and traffic management [76].

Autonomous drones and robots also utilise distributed TinyML for navigation and obstacle avoidance. These systems require real-time data processing to adapt to dynamic environments, and distributed learning enables them to share and learn from each other's experiences without relying on a central server [77]. This collaborative approach improves the overall intelligence and responsiveness of autonomous systems [78; 5].

Smart Cities

Smart cities leverage distributed TinyML to enhance urban services, including traffic management, energy optimisation, and environmental monitoring. By deploying TinyML models on edge devices such as traffic lights, streetlights, and environmental sensors, cities can achieve real-time data processing and decision-making [79].

For instance, smart traffic management systems can use distributed learning to analyse traffic patterns and optimise signal timings, reducing congestion and improving traffic flow [80; 81; 82]. Similarly, energy optimisation systems can monitor and control energy usage in real-time, leading to more efficient and sustainable urban living [83; 84]. Research also highlights the application of distributed TinyML in smart cities, demonstrating its potential to transform urban environments [80; 82].

Satellite Networks

In satellite networks, distributed TinyML addresses the challenges posed by the high latency of long-distance communication. Zhao et al. proposed a novel distributed machine learning network structure for low Earth orbit (LEO) satellite networks. This structure involves a layered hierarchical approach to combat delays, enabling satellites to collaboratively process data and update models without relying on ground stations. This approach reduces latency and enhances the overall efficiency of satellite networks [85].

Satellite-based distributed learning can be particularly beneficial for global environmental monitoring and disaster response applications, where timely data processing is crucial. By enabling satellites to share and process data collaboratively, distributed TinyML ensures that critical information is quickly available for decision-making and response efforts [86].

4.6 Communication Efficiency

Efficient communication is crucial in Distributed TinyML, where edge devices with limited bandwidth must collaborate over constrained networks [87]. Strategies such as model

gossiping and protocol optimisations reduce the volume and frequency of data exchanges, improving overall network efficiency [88; 89; 90; 91; 92; 63].

Model Gossiping
Model gossiping is a technique where devices exchange model updates with a subset of their peers instead of all nodes. This reduces communication overhead and helps scale the network to more devices [93]. Gossiping protocols are robust to network changes and device failures, making them suitable for dynamic environments [92; 94; 95].

Protocol Optimisations
Optimising communication protocols to reduce data transfer size and frequency is essential for maintaining efficiency in distributed TinyML systems. Techniques such as quantisation, sparsification, and compression of model updates help to minimise bandwidth requirements [96]. Protocol optimisations also include efficient aggregation methods, like secure multi-party computation (SMC), to enhance privacy and reduce communication costs [97].

Quantisation Techniques
Quantisation techniques are vital in reducing the size of model updates transmitted between devices. By representing model parameters with lower precision, these techniques significantly decrease the amount of data that needs to be sent, thus conserving bandwidth and reducing latency [98]. In addition, lower communication requirements often translate to lower power consumption, which is a very important consideration for edge-deployed tiny devices [99]. Popular quantisation methods include fixed-point arithmetic, where parameters are stored as integers, and dynamic quantisation, which adapts the precision based on the value range of parameters [100; 101; 102].

Sparsification Methods
Sparsification involves reducing the number of non-zero elements in model updates, thus decreasing the volume of data that needs to be communicated. Techniques like gradient sparsification and weight pruning ensure that only the most critical information is transmitted, effectively lowering communication overhead [103]. This approach is particularly beneficial when network bandwidth is severely limited [104].

Compression Algorithms
Compression algorithms such as Huffman coding, run-length encoding, and more sophisticated methods like Deep Compression can be applied to model updates to reduce data size further. These algorithms exploit the redundancy in data to encode information more efficiently, resulting in significant bandwidth savings [105]. Combining compression with other techniques like quantisation and sparsification can lead to even more significant improvements in communication efficiency [106].

Efficient Aggregation Methods
Efficient aggregation methods are essential to combine model updates from multiple devices while minimising communication costs. Techniques such as federated averaging, where local models are averaged to update the global model, reduce the need for frequent and large-scale data transfers [14]. Advanced methods like secure multi-party computation (SMC) ensure that the aggregation process is efficient and privacy-preserving, making them suitable for sensitive applications [107].

Adaptive Communication Strategies
Adaptive communication strategies dynamically adjust the frequency and volume of data exchanges based on network conditions and resource availability. Techniques such as bandwidth-aware scheduling and adaptive compression can optimise communication, ensuring efficient use of available resources while maintaining model accuracy [108]. These strategies are particularly useful in heterogeneous networks where devices may have varying capabilities and network connections [109].

Future Directions in communication efficiency
Future research in communication efficiency for distributed TinyML could explore the integration of emerging technologies such as 5G and edge AI. These technologies promise to enhance network capabilities, enabling more efficient and reliable communication between devices [110]. Additionally, advancements in hardware design, such as specialised communication processors, could further optimise the data exchange process in TinyML systems [111].

5 Algorithm implementation details

As part of this research, two algorithms were selected to demonstrate the feasibility of using TinyML. These algorithms were chosen to exemplify different approaches to efficiently handling machine learning tasks within the limited computational and memory resources of such devices. Each algorithm highlights a unique method of optimising performance and resource usage, thereby providing insights into the practical implementation of machine learning in resource-constrained environments.

5.1 Incremental Learning Vector Quantisation

The first of the two implemented algorithms was Incremental Learning Vector Quantisation (ILVQ) [112]. ILVQ is an enhanced version of the Learning Vector Quantisation algorithm, conceptually similar to the k-nearest neighbours classifier, with the key distinction that it does not require the entire dataset to be stored in memory. Instead, 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. Classification of a given data vector is then performed by selecting the prototype that is closest to the data vector, according to some distance metric. In this experiment, euclidean distance was used.

The primary contribution of ILVQ is its ability to handle incremental learning in two distinct ways: within-class and between-class. Within-class incremental learning allows the model to gradually assimilate new information within the same class. Conversely, between-class incremental learning enables the model to progressively learn new information from new classes as they emerge. In essence, ILVQ can

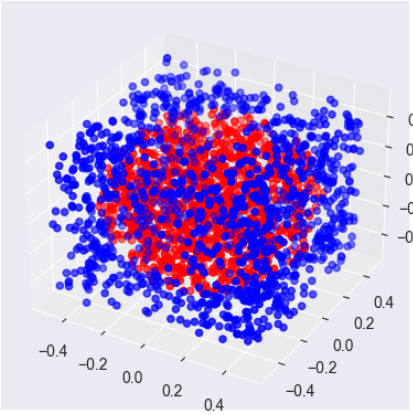

Figure 4: Visualisation of the synthetic dataset (n = 2500)

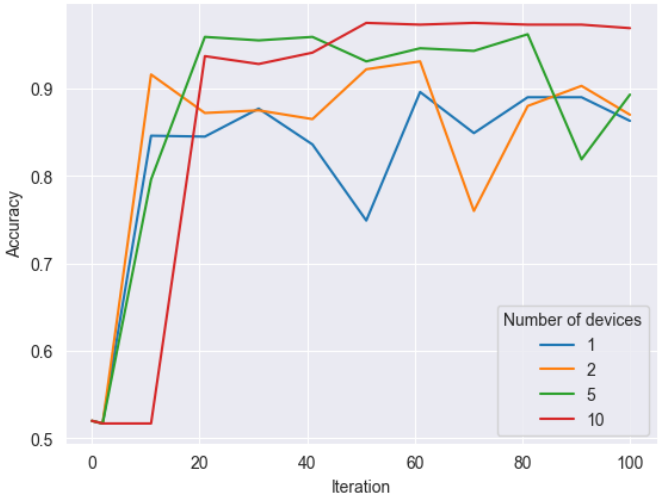

Figure 5: Accuracy of different number of devices

update its knowledge base within existing classes and recognise and learn new classes during the training phase [113; 114]. This capability makes it particularly suitable for dynamic environments where new data categories may continually appear.

ILVQ was selected for this study due to its simplicity and low-performance constraints, which make it an excellent candidate for implementation on low-powered microcontrollers [115]. Its efficient use of memory and computational resources aligns well with the limitations typically associated with TinyML applications, ensuring practical feasibility in resource-constrained settings.

Experiment setup

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

The dataset was generated by uniformly sampling vectors, with every component within the $[-\frac{1}{2}, \frac{1}{2}]$ range. Effectively, all sampled data points were in a 1x1x1 cube, centred at the $(0, 0, 0)$ point.

The generated dataset was then split into two groups: the group of points within the sphere and those outside the sphere. Figure 4 depicts a sample of the generated synthetic dataset, with points within the sphere coloured red and points outside of the sphere coloured blue.

The implementation of the algorithm also had to be adapted for the distributed topology of the machine learning network. To achieve this, a federated learning approach was adopted, with a simple aggregation algorithm which propagated the best-performing model to all devices in the learning network.

Distributed training networks with 1, 2, 5, and 10 devices were simulated, and the training accuracy of each configuration was measured. The data was uniformly distributed among all participating devices, resulting in an IID dataset and training setup.

The algorithm was written in C++ and deployed to an Arduino 33 BLE board. As only a single board was available for performing the experiment, additional devices were simulated by swapping the active model prototypes - meaning that to simulate ten different devices, there were ten sets of prototypes, which were trained in a round-robin fashion.

Benchmark results

The performance of the Incremental Learning Vector Quantisation algorithm was evaluated based on its classification accuracy over multiple training iterations. The experiment measured the algorithm's ability to adapt and improve over time, particularly in a resource-constrained environment.

As illustrated in Figure 5, the trained models achieved a classification accuracy of at least 85% across all simulated configurations by the conclusion of the training process. This performance significantly surpasses the baseline accuracy of 52% associated with a "random guessing" strategy, thereby demonstrating the efficacy of the Incremental Learning Vector Quantisation algorithm in effectively learning and classifying data points.

However, it is important to note that the accuracy of the trained model could potentially decrease significantly if it were applied to non-IID data distributions. Non-IID data, where the distribution of training samples is not uniform across devices, presents additional challenges for model generalisation and accuracy. This concern is supported by findings in related research, such as the work by Chiu et al. [116], which highlights the potential impact of non-IID data on model performance. Consequently, further investigation into the algorithm's robustness in such scenarios is necessary to fully understand its practical applicability in diverse real-world settings.

5.2 MNIST classification with TensorFlow Federated learning framework

The second implemented algorithm was a simple neural network classifier, tasked with classifying the digits from the MNIST handwritten digit dataset[5] [117].

[5]https://en.wikipedia.org/wiki/MNIST_database

Layer type	Output shape	Number of params
Dense	(None, 10)	7850
Softmax	(None, 10)	0
Total parameters	7850 (30.66 KB)	

Table 1: TensorFlow model summary

The implementation was written using the TensorFlow federated machine learning framework[6].

Experiment setup

The model selected for this experiment comprises two layers: a dense layer and a softmax output layer. This architecture was chosen for its simplicity and efficiency [118].

One of the primary reasons for selecting this particular model was its minimal memory footprint. The entire model requires less than 32KB of memory, as detailed in Table 1. This characteristic makes it exceptionally well-suited for deployment on devices with limited memory resources, which is a common constraint in many real-world applications of federated learning.

In the experimental setup, each device was provided with a portion of the input dataset in a non-IID (non-Independent and Identically Distributed) manner. This means that the distribution of digit samples across devices was not uniform; for example, some devices might receive more samples of digit 1 and fewer of digit 7. This approach evaluated the model's performance under more realistic and challenging conditions, reflecting the variability and heterogeneity often encountered in practical federated learning scenarios. By doing so, the robustness and adaptability of the model in handling diverse data distributions were assessed.

The experiment was conducted by simulating various federated learning configurations to evaluate the model's performance. Specifically, simulations were conducted with 5, 10, 50, and 100 devices. By varying the number of devices, we aimed to understand the impact of federation size on model accuracy, convergence rate, and stability.

Benchmark results

As shown in Figure 6, the accuracy of the model was found to be consistent across all setups and sufficiently high (around 85% across all setups), indicating the robustness of the algorithm in distributed learning environments. However, it was observed that a lower number of devices resulted in a more significant variance in the accuracy metrics, suggesting that larger federations provide more stable and reliable performance.

6 Responsible Research

In conducting this literature survey, ethical aspects and the reproducibility of methods were considered, as ensuring responsible research practices helps maintain the integrity and credibility of the scientific process.

[6]https://www.tensorflow.org/federated

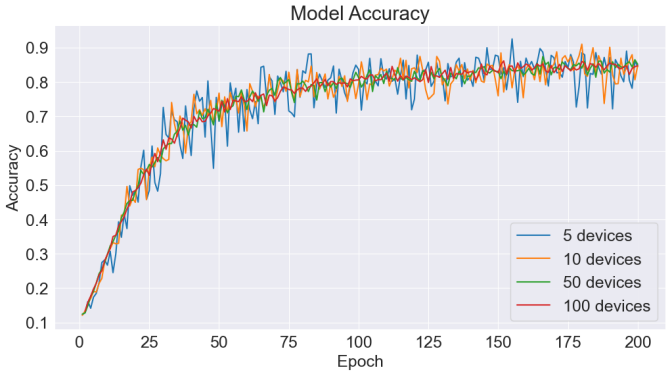

Figure 6: Accuracy of the model with a different number of devices

6.1 Ethical Considerations

Ethical considerations in research involve several key aspects, including:

- **Transparency and Honesty**: All sources of information used in this survey are accurately cited to give proper credit to original authors and allow readers to verify the sources independently. There is no fabrication or falsification of data or findings.
- **Respect for Intellectual Property**: Proper citations are made to respect the intellectual property rights of authors whose work has been referenced. This includes adhering to copyright laws and licensing agreements.
- **Bias Avoidance**: Efforts have been made to include a wide range of studies to provide a balanced view of the current state of TinyML research. This includes avoiding the selective reporting of results that support a particular viewpoint and considering diverse perspectives and methodologies.

6.2 Reproducibility of Methods

To ensure reproducibility in this literature survey, a systematic methodology was adopted, guided by established protocols for systematic reviews.

By adhering to it, the survey ensures that the research process is transparent, verifiable, and reproducible. This approach aligns with the guidelines established by Kitchenham and Charters [8], emphasising the importance of systematic reviews in providing reliable and comprehensive insights into research topics.

The study also follows the FAIR principles[119] by citing all analysed articles, only using publicly available repositories, and publishing the source code replication package[7].

7 Conclusions and Future Work

Distributed TinyML offers a spectrum of training approaches, including federated, swarm, collaborative, and split learning. Each approach caters to specific requirements concerning

[7]https://github.com/rstular/BSc-Thesis-Tiny-Machine-Learning-Survey

data privacy, scalability, and resource constraints. Data partitioning strategies, such as model-parallel and data-parallel partitioning, can optimise resource utilisation and training efficiency for distributed models. Scalability and robustness are crucial for real-world deployments, and advancements in fault tolerance mechanisms and load-balancing techniques are essential for ensuring reliable operation. Distributed TinyML demonstrates significant potential in various use cases, including healthcare, autonomous systems, smart cities, and satellite networks. Its ability to process data locally while facilitating collaborative learning opens doors for innovative applications. Communication efficiency is paramount in resource-constrained environments. Strategies like model gossiping, compression, protocol optimisations, and efficient aggregation methods are crucial for minimising network overhead. Network topology plays a vital role in determining performance. Centralised, decentralised, and hybrid topologies have advantages and disadvantages, and the optimal choice depends on the specific application and network characteristics.

Future research directions in distributed TinyML can explore several promising avenues:

- Integration with emerging technologies: The synergy between distributed TinyML and technologies like 6G and edge AI holds immense potential for enhancing network capabilities and communication efficiency.
- Hardware advancements: Specialised communication processors and energy-efficient hardware designs can further optimise data exchange and processing within TinyML systems.
- Security and privacy enhancements: Robust security mechanisms are essential for protecting sensitive data and ensuring the integrity of model updates in distributed environments.
- Federated learning for non-IID data: Addressing the challenges associated with non-IID data distributions is crucial for ensuring the generalisability and robustness of federated learning models in real-world scenarios.
- Explainability and interpretability: Developing techniques to understand how distributed TinyML models arrive at their decisions can enhance trust and facilitate their adoption in safety-critical applications.

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.

A Appendix

A.1 Article database search query

```
(("federated" NEAR/5 "learning" NEAR/5 "iot") OR ("
    distributed" NEAR/5 "learning" NEAR/5 "iot") OR
     ("distributed" NEAR/5 "learning" NEAR/5 "
    microcontroller") OR ("federated" NEAR/5 "
    learning" NEAR/5 "microcontroller") OR ("
    federated" NEAR/5 "learning" NEAR/5 "embedded
    system") OR ("distributed" NEAR/5 "learning"
    NEAR/5 "embedded system") OR ("federated" NEAR
    /5 "learning" NEAR/5 "sensor") OR ("distributed
    " NEAR/5 "learning" NEAR/5 "sensor") OR ("
    federated" NEAR/5 "machine learning" NEAR/5 "
    iot") OR ("distributed" NEAR/5 "machine
    learning" NEAR/5 "iot") OR ("distributed" NEAR
    /5 "machine learning" NEAR/5 "microcontroller")
     OR ("federated" NEAR/5 "machine learning" NEAR
    /5 "microcontroller") OR ("federated" NEAR/5 "
    machine learning" NEAR/5 "embedded system") OR
    ("distributed" NEAR/5 "machine learning" NEAR/5
     "embedded system") OR ("federated" NEAR/5 "
    machine learning" NEAR/5 "sensor") OR ("
    distributed" NEAR/5 "machine learning" NEAR/5 "
    sensor"))
```

References

[1] 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):30:1–30:33, March 2020.

[2] Project Forum - Projects - Tiny Machine Learning for 6G Networks - Group.

[3] Li-Hsiang Shen, Kai-Ten Feng, and Lajos Hanzo. Five Facets of 6G: Research Challenges and Opportunities. *ACM Computing Surveys*, 55(11):235:1–235:39, February 2023.

[4] Albert Banchs, Marco Fiore, Andres Garcia-Saavedra, and Marco Gramaglia. Network intelligence in 6G: challenges and opportunities. In *Proceedings of the 16th ACM Workshop on Mobility in the Evolving Internet Architecture*, MobiArch '21, pages 7–12, New York, NY, USA, October 2021. Association for Computing Machinery.

[5] Mu-Yen Chen, Min-Hsuan Fan, and Li-Xiang Huang. AI-Based Vehicular Network toward 6G and IoT: Deep Learning Approaches. *ACM Transactions on Management Information Systems*, 13(1):6:1–6:12, October 2021.

[6] Ji Lin, Ligeng Zhu, Wei-Ming Chen, Wei-Chen Wang, and Song Han. Tiny Machine Learning: Progress and Futures [Feature]. *IEEE Circuits and Systems Magazine*, 23(3):8–34, 2023. Conference Name: IEEE Circuits and Systems Magazine.

[7] Martín González-Soto, Rebeca P. Díaz-Redondo, Manuel Fernández-Veiga, Bruno Fernández-Castro, and Ana Fernández-Vilas. Decentralized and collaborative machine learning framework for IoT. *Computer Networks*, 239:110137, February 2024.

[8] Barbara Kitchenham and Stuart Charters. Guidelines for performing Systematic Literature Reviews in Software Engineering. 2, January 2007.

[9] Sriram S, Hariharathmajan Rk, Barathi Babu M, Amal Pradeep, and Karthi R. Federated learning on low-power Arduino Nano33 BLE Sense to predict the

length of stay using a linear regression model. *Procedia Computer Science*, 235:671–682, January 2024.

[10] Zheng Lu and Xing Liu. IoT Application Development Based on Java and Raspberry Pi. In *2021 IEEE 12th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pages 0600–0606, October 2021. ISSN: 2644-3163.

[11] Markus Voelter, Arie van Deursen, Bernd Kolb, and Stephan Eberle. Using C language extensions for developing embedded software: a case study. In *Proceedings of the 2015 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications*, OOPSLA 2015, pages 655–674, New York, NY, USA, October 2015. Association for Computing Machinery.

[12] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12(85):2825–2830, 2011.

[13] 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].

[14] 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].

[15] Diego Peteiro-Barral and Bertha Guijarro-Berdiñas. A survey of methods for distributed machine learning. *Progress in Artificial Intelligence*, 2(1):1–11, March 2013.

[16] Visal Rajapakse, Ishan Karunanayake, and Nadeem Ahmed. Intelligence at the Extreme Edge: A Survey on Reformable TinyML. *ACM Computing Surveys*, 55(13s):282:1–282:30, July 2023.

[17] Vasileios Tsoukas, Anargyros Gkogkidis, Eleni Boumpa, and Athanasios Kakarountas. A Review on the emerging technology of TinyML. *ACM Computing Surveys*, April 2024. Just Accepted.

[18] Vasileios Tsoukas, Eleni Boumpa, Georgios Giannakas, and Athanasios Kakarountas. A Review of Machine Learning and TinyML in Healthcare. In *Proceedings of the 25th Pan-Hellenic Conference on Informatics*, PCI '21, pages 69–73, New York, NY, USA, February 2022. Association for Computing Machinery.

[19] Shashank Bangalore Lakshman and Nasir U. Eisty. Software engineering approaches for TinyML based IoT embedded vision: a systematic literature review. In *Proceedings of the 4th International Workshop on Software Engineering Research and Practice for the IoT*, SERP4IoT '22, pages 33–40, New York, NY, USA, February 2023. Association for Computing Machinery.

[20] Stefanie Warnat-Herresthal, Hartmut Schultze, Krishnaprasad Lingadahalli Shastry, Sathyanarayanan Manamohan, Saikat Mukherjee, Vishesh Garg, Ravi Sarveswara, Kristian Händler, Peter Pickkers, N. Ahmad Aziz, Sofia Ktena, Florian Tran, Michael Bitzer, Stephan Ossowski, Nicolas Casadei, Christian Herr, Daniel Petersheim, Uta Behrends, Fabian Kern, Tobias Fehlmann, Philipp Schommers, Clara Lehmann, Max Augustin, Jan Rybniker, Janine Altmüller, Neha Mishra, Joana P. Bernardes, Benjamin Krämer, Lorenzo Bonaguro, Jonas Schulte-Schrepping, Elena De Domenico, Christian Siever, Michael Kraut, Milind Desai, Bruno Monnet, Maria Saridaki, Charles Martin Siegel, Anna Drews, Melanie Nuesch-Germano, Heidi Theis, Jan Heyckendorf, Stefan Schreiber, Sarah Kim-Hellmuth, Jacob Nattermann, Dirk Skowasch, Ingo Kurth, Andreas Keller, Robert Bals, Peter Nürnberg, Olaf Rieß, Philip Rosenstiel, Mihai G. Netea, Fabian Theis, Sach Mukherjee, Michael Backes, Anna C. Aschenbrenner, Thomas Ulas, Monique M. B. Breteler, Evangelos J. Giamarellos-Bourboulis, Matthijs Kox, Matthias Becker, Sorin Cheran, Michael S. Woodacre, Eng Lim Goh, and Joachim L. Schultze. Swarm Learning for decentralized and confidential clinical machine learning. *Nature*, 594(7862):265–270, June 2021. Publisher: Nature Publishing Group.

[21] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Hubert Eichner, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konečný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Mariana Raykova, Hang Qi, Daniel Ramage, Ramesh Raskar, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and Open Problems in Federated Learning, March 2021. arXiv:1912.04977 [cs, stat].

[22] Praneeth Vepakomma, Otkrist Gupta, Tristan Swedish, and Ramesh Raskar. Split learning for health: Distributed deep learning without sharing raw patient data, December 2018. arXiv:1812.00564 [cs, stat].

[23] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated Learning: Challenges, Methods, and Future Directions. *IEEE Signal Processing Magazine*, 37(3):50–60, May 2020. arXiv:1908.07873 [cs, stat].

[24] Lee Andrew Bygrave. *Data Privacy Law: An International Perspective*. Oxford University Press, January 2014.

[25] Chen Zhang, Yu Xie, Hang Bai, Bin Yu, Weihong Li, and Yuan Gao. A survey on federated learning. *Knowledge-Based Systems*, 216:106775, March 2021.

[26] Harish Devaraj, Shaleeza Sohail, Melanie Ooi, Boyang Li, Nathaniel Hudson, Matt Baughman, Kyle Chard, Ryan Chard, Enrico Casella, Ian Foster, and Omer Rana. RuralAI in Tomato Farming: Integrated Sensor System, Distributed Computing, and Hierarchical Federated Learning for Crop Health Monitoring. *IEEE Sensors Letters*, 8(5):1–4, May 2024. Conference Name: IEEE Sensors Letters.

[27] Varun Laxman Muttepawar, Arjun Mehra, Zubair Shaban, Ranjitha Prasad, and J. Harshan. Federated Learning for Wireless Applications: A Prototype. In *2024 16th International Conference on COMmunication Systems & NETworkS (COMSNETS)*, pages 300–302, January 2024. ISSN: 2155-2509.

[28] Rodolfo Stoffel Antunes, Cristiano André da Costa, Arne Küderle, Imrana Abdullahi Yari, and Björn Eskofier. Federated Learning for Healthcare: Systematic Review and Architecture Proposal. *ACM Transactions on Intelligent Systems and Technology*, 13(4):54:1–54:23, May 2022.

[29] Mohammad Arif Hossain, Abdullah Ridwan Hossain, and Nirwan Ansari. AI in 6G: Energy-Efficient Distributed Machine Learning for Multilayer Heterogeneous Networks. *IEEE Network*, 36(6):84–91, November 2022. Conference Name: IEEE Network.

[30] Hongjian Shi, Ruhui Ma, Dongmei Li, and Haibing Guan. Hierarchical Adaptive Collaborative Learning: A Distributed Learning Framework for Customized Cloud Services in 6G Mobile Systems. *IEEE Network*, 37(2):44–53, March 2023. Conference Name: IEEE Network.

[31] Xiaoyan Huang, Ke Zhang, Fan Wu, and Supeng Leng. Collaborative Machine Learning for Energy-Efficient Edge Networks in 6G. *IEEE Network*, 35(6):12–19, November 2021. Conference Name: IEEE Network.

[32] Sawsan Abdulrahman, Hanine Tout, Hakima Ould-Slimane, Azzam Mourad, Chamseddine Talhi, and Mohsen Guizani. A Survey on Federated Learning: The Journey From Centralized to Distributed On-Site Learning and Beyond. *IEEE Internet of Things Journal*, 8(7):5476–5497, April 2021. Conference Name: IEEE Internet of Things Journal.

[33] Hardeep Patel, Nishtha Chaudhari, Meet Kavathiya, Hargeet Kaur, and Kaushal Shah. An Exploration to Blockchain-based Deep LearningFramework. In *2023 10th International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 726–733, March 2023.

[34] Yong Li, Haichao Ling, Xianglin Ren, Chun Yu, and Tongtong Liu. Privacy-Preserving Swarm Learning Based on Lightweight Homomorphic Encryption and Blockchain Technology. In *2023 IEEE 5th Eurasia Conference on IOT, Communication and Engineering (ECICE)*, pages 692–697, October 2023.

[35] Panagiotis Drakatos, Erodotos Demetriou, Stavroulla Koumou, Andreas Konstantinidis, and Demetrios Zeinalipour-Yazti. Towards a Blockchain Database for Massive IoT Workloads. In *2021 IEEE 37th International Conference on Data Engineering Workshops (ICDEW)*, pages 76–79, April 2021. ISSN: 2473-3490.

[36] Oliver Lester Saldanha, Philip Quirke, Nicholas P. West, Jacqueline A. James, Maurice B. Loughrey, Heike I. Grabsch, Manuel Salto-Tellez, Elizabeth Alwers, Didem Cifci, Narmin Ghaffari Laleh, Tobias Seibel, Richard Gray, Gordon G. A. Hutchins, Hermann Brenner, Marko van Treeck, Tanwei Yuan, Titus J. Brinker, Jenny Chang-Claude, Firas Khader, Andreas Schuppert, Tom Luedde, Christian Trautwein, Hannah Sophie Muti, Sebastian Foersch, Michael Hoffmeister, Daniel Truhn, and Jakob Nikolas Kather. Swarm learning for decentralized artificial intelligence in cancer histopathology. *Nature Medicine*, 28(6):1232–1239, June 2022. Publisher: Nature Publishing Group.

[37] Zesheng Liu, Tao Zhu, Zhenyu Liu, Huansheng Ning, and Liming Chen. Reducing Communication Costs of Federated Contrastive Learning by Particle Swarm Optimization. In *2021 3rd International Academic Exchange Conference on Science and Technology Innovation (IAECST)*, pages 687–692, December 2021.

[38] Yansong Gao, Minki Kim, Sharif Abuadbba, Yeonjae Kim, Chandra Thapa, Kyuyeon Kim, Seyit A. Camtep, Hyoungshick Kim, and Surya Nepal. End-to-End Evaluation of Federated Learning and Split Learning for Internet of Things. In *2020 International Symposium on Reliable Distributed Systems (SRDS)*, pages 91–100, September 2020. ISSN: 2575-8462.

[39] Chandra Thapa, M. A. P. Chamikara, Seyit Camtepe, and Lichao Sun. SplitFed: When Federated Learning Meets Split Learning, February 2022. arXiv:2004.12088 [cs].

[40] Jianchun Liu, Yujia Huo, Pengcheng Qu, Sun Xu, Zhi Liu, Qianpiao Ma, and Jinyang Huang. FedCD: A Hybrid Federated Learning Framework for Efficient Training With IoT Devices. *IEEE Internet of Things Journal*, 11(11):20040–20050, June 2024. Conference Name: IEEE Internet of Things Journal.

[41] Enrique Mármol Campos, Pablo Saura, Aurora González Vidal, José Hernández-Ramos, Jorge Bernal Bernabe, Gianmarco Baldini, and Antonio Skarmeta. *Evaluating Federated Learning for Intrusion Detection in Internet of Things: Review and Challenges*. August 2021.

[42] Jakub Konečný, Brendan McMahan, and Daniel Ramage. Federated Optimization:Distributed Optimization Beyond the Datacenter, November 2015. arXiv:1511.03575 [cs, math].

[43] Jakub Konečný, H. Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated Optimization: Distributed Machine Learning for On-Device Intelligence, October 2016. arXiv:1610.02527 [cs].

[44] Pan Zhou, Qian Lin, Dumitrel Loghin, Beng Chin Ooi, Yuncheng Wu, and Hongfang Yu. Communication-efficient Decentralized Machine Learning over Heterogeneous Networks. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pages 384–395, April 2021. ISSN: 2375-026X.

[45] Fabrizio Marozzo, Alessio Orsino, Domenico Talia, and Paolo Trunfio. Edge Computing Solutions for Distributed Machine Learning. In *2022 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, pages 1–8, September 2022.

[46] Sonali Vyas, Mahima Gupta, and Rakesh Yadav. Converging Blockchain and Machine Learning for Healthcare. In *2019 Amity International Conference on Artificial Intelligence (AICAI)*, pages 709–711, February 2019.

[47] Aristeidis Karras, Christos Karras, Konstantinos C. Giotopoulos, Dimitrios Tsolis, Konstantinos Oikonomou, and Spyros Sioutas. Peer to Peer Federated Learning: Towards Decentralized Machine Learning on Edge Devices. In *2022 7th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference (SEEDA-CECNSM)*, pages 1–9, September 2022.

[48] Anis Elgabli, Jihong Park, Amrit S. Bedi, Mehdi Bennis, and Vaneet Aggarwal. Q-GADMM: Quantized Group ADMM for Communication Efficient Decentralized Machine Learning. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8876–8880, May 2020. ISSN: 2379-190X.

[49] Jamie McQuire, Paul Watson, Nick Wright, Hugo Hiden, and Michael Catt. Uneven and Irregular Surface Condition Prediction from Human Walking Data using both Centralized and Decentralized Machine Learning Approaches. In *2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 1449–1452, December 2021.

[50] Joohyung Jeon and Joongheon Kim. Privacy-Sensitive Parallel Split Learning. In *2020 International Conference on Information Networking (ICOIN)*, pages 7–9, January 2020. ISSN: 1976-7684.

[51] Ahmad Ayad, Melvin Renner, and Anke Schmeink. Improving the Communication and Computation Efficiency of Split Learning for IoT Applications. In *2021 IEEE Global Communications Conference (GLOBECOM)*, pages 01–06, December 2021.

[52] Xing Zhao, Aijun An, Junfeng Liu, and Bao Xin Chen. Dynamic Stale Synchronous Parallel Distributed Training for Deep Learning, August 2019. arXiv:1908.11848 [cs, stat].

[53] Yuchen Fan, Jilin Zhang, Nailiang Zhao, Yongjian Ren, Jian Wan, Li Zhou, Zhongyu Shen, Jue Wang, Juncong Zhang, and Zhenguo Wei. Model Aggregation Method for Data Parallelism in Distributed Real-Time Machine Learning of Smart Sensing Equipment. *IEEE Access*, 7:172065–172073, 2019. Conference Name: IEEE Access.

[54] Yuwei Wang and Burak Kantarci. Reputation-enabled Federated Learning Model Aggregation in Mobile Platforms. In *ICC 2021 - IEEE International Conference on Communications*, pages 1–6, June 2021. ISSN: 1938-1883.

[55] Pallav Kumar Deb, Anandarup Mukherjee, Digvijay Singh, and Sudip Misra. Loop-the-Loops: Fragmented Learning Over Networks for Constrained IoT Devices. *IEEE Transactions on Parallel and Distributed Systems*, 34(1):316–327, January 2023. Conference Name: IEEE Transactions on Parallel and Distributed Systems.

[56] Songge Zhang, Wen Wu, Penghui Hu, Shaofeng Li, and Ning Zhang. Split Federated Learning: Speed up Model Training in Resource-Limited Wireless Networks. In *2023 IEEE 43rd International Conference on Distributed Computing Systems (ICDCS)*, pages 985–986, July 2023. ISSN: 2575-8411.

[57] Rohit Kewalramani. Basics of Distributed Training — PyTorch edition, January 2024.

[58] Ayah Abo El Rejal, Andreas Pester, and Khaled Nagaty. Tiny Machine Learning for Underwater Image Enhancement: Pruning and Quantization Approach. In *2023 International Conference on Computer and Applications (ICCA)*, pages 1–6, November 2023.

[59] Simone Scardapane and Paolo Di Lorenzo. A framework for parallel and distributed training of neural networks. *Neural Networks*, 91:42–54, July 2017.

[60] Praneeth Vepakomma and Ramesh Raskar. Split Learning: A Resource Efficient Model and Data Parallel Approach for Distributed Deep Learning. In Heiko Ludwig and Nathalie Baracaldo, editors, *Federated Learning: A Comprehensive Overview of Methods and Applications*, pages 439–451. Springer International Publishing, Cham, 2022.

[61] Joana Tirana, Spyros Lalis, and Dimitris Chatzopoulos. MP-SL: Multihop Parallel Split Learning, January 2024. arXiv:2402.00208 [cs].

[62] Jiashi Feng, Huan Xu, and Shie Mannor. Distributed Robust Learning, February 2015. arXiv:1409.5937 [cs, stat].

[63] Yandong Shi, Lixiang Lian, Yuanming Shi, Zixin Wang, Yong Zhou, Liqun Fu, Lin Bai, Jun Zhang, and

Wei Zhang. Machine Learning for Large-Scale Optimization in 6G Wireless Networks. *IEEE Communications Surveys & Tutorials*, 25(4):2088–2132, 2023. Conference Name: IEEE Communications Surveys & Tutorials.

[64] Koshi Eguchi, Hideya Ochiai, and Hiroshi Esaki. MemWAFL: Efficient Model Aggregation for Wireless Ad Hoc Federated Learning in Sparse Dynamic Networks. In *2023 IEEE Future Networks World Forum (FNWF)*, pages 1–5, November 2023. ISSN: 2770-7679.

[65] Dragos Lia and Mihai Togan. Privacy-Preserving Machine Learning Using Federated Learning and Secure Aggregation. In *2020 12th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, pages 1–6, June 2020.

[66] Alberto Ancilotto, Francesco Paissan, and Elisabetta Farella. XiNet: Efficient Neural Networks for tinyML. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16922–16931, October 2023. ISSN: 2380-7504.

[67] Angelo Rodio, Giovanni Neglia, Fabio Busacca, Stefano Mangione, Sergio Palazzo, Francesco Restuccia, and Ilenia Tinnirello. Federated Learning with Packet Losses. In *2023 26th International Symposium on Wireless Personal Multimedia Communications (WPMC)*, pages 1–6, November 2023. ISSN: 1882-5621.

[68] Liyan Shen, Zhenhan Ke, Jinqiao Shi, Xi Zhang, Yanwei Sun, Jiapeng Zhao, Xuebin Wang, and Xiaojie Zhao. SPEFL: Efficient Security and Privacy-Enhanced Federated Learning Against Poisoning Attacks. *IEEE Internet of Things Journal*, 11(8):13437–13451, April 2024. Conference Name: IEEE Internet of Things Journal.

[69] Noora Al-Maslamani, Mohamed Abdallah, and Bekir Sait Ciftler. Secure Federated Learning for IoT using DRL-based Trust Mechanism. In *2022 International Wireless Communications and Mobile Computing (IWCMC)*, pages 1101–1106, May 2022. ISSN: 2376-6506.

[70] Rakhee Kallimani, Krishna Pai, Prasoon Raghuwanshi, Sridhar Iyer, and Onel L. A. López. TinyML: Tools, applications, challenges, and future research directions. *Multimedia Tools and Applications*, 83(10):29015–29045, March 2024.

[71] Yan Hu and Ahmad Chaddad. Potential of Federated Learning in Healthcare. In *2023 IEEE International Conference on E-health Networking, Application & Services (Healthcom)*, pages 1–2, December 2023.

[72] Haya Elayan, Moayad Aloqaily, and Mohsen Guizani. Deep Federated Learning for IoT-based Decentralized Healthcare Systems. In *2021 International Wireless Communications and Mobile Computing (IWCMC)*, pages 105–109, June 2021. ISSN: 2376-6506.

[73] Haya Elayan, Moayad Aloqaily, and Mohsen Guizani. Deep Federated Learning for IoT to Improve Healthcare Operations. In *2023 Fourth International Conference on Intelligent Data Science Technologies and Applications (IDSTA)*, pages 85–90, October 2023.

[74] Yuchen Jiang and Chang Ji. FedGPS: Personalized Cross-Silo Federated Learning for Internet of Things-enabled Predictive Maintenance. In *2022 IEEE Smartworld, Ubiquitous Intelligence & Computing, Scalable Computing & Communications, Digital Twin, Privacy Computing, Metaverse, Autonomous & Trusted Vehicles (SmartWorld/UIC/ScalCom/DigitalTwin/PriComp/Meta)*, pages 912–920, December 2022.

[75] Lei Liu, Chen Chen, Qingqi Pei, Sabita Maharjan, and Yan Zhang. Vehicular Edge Computing and Networking: A Survey. *Mobile Networks and Applications*, 26(3):1145–1168, June 2021.

[76] Abdelkader Mekrache, Abbas Bradai, Emmanuel Moulay, and Samir Dawaliby. Deep reinforcement learning techniques for vehicular networks: Recent advances and future trends towards 6G. *Vehicular Communications*, 33:100398, January 2022.

[77] Miguel de Prado, Manuele Rusci, Alessandro Capotondi, Romain Donze, Luca Benini, and Nuria Pazos. Robustifying the Deployment of tinyML Models for Autonomous Mini-Vehicles. *Sensors*, 21(4):1339, January 2021. Number: 4 Publisher: Multidisciplinary Digital Publishing Institute.

[78] Yassine El Hafid, Abdessamad El Rharras, Abdellah Chehri, Rachid Saadane, and Mohammed Wahbi. Real-Time Data Processing in Autonomous Vehicles Based on Distributed Architecture: A Case Study. In Xiaobo Qu, Lu Zhen, Robert J. Howlett, and Lakhmi C. Jain, editors, *Smart Transportation Systems 2020*, pages 143–154, Singapore, 2020. Springer.

[79] Dina Hussein, Dina Ibrahim, and Norah Alajlan. Original Research Article TinyML: Adopting tiny machine learning in smart cities. *Journal of Autonomous Intelligence*, 7:1–14, January 2024.

[80] Bo Liu and Zhengtao Ding. A distributed deep reinforcement learning method for traffic light control. *Neurocomputing*, 490:390–399, June 2022.

[81] Arindam Chaudhuri. Smart traffic management of vehicles using faster R-CNN based deep learning method. *Scientific Reports*, 14(1):10357, May 2024. Publisher: Nature Publishing Group.

[82] Ying Liu, Lei Liu, and Wei-Peng Chen. Intelligent traffic light control using distributed multi-agent Q learning. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–8, October 2017. ISSN: 2153-0017.

[83] Fotios Kalioras, Gabriel Filios, Stylianos Karatzas, and Sotiris Nikoletseas. (POSTER) A Holistic IoT-Enabled Approach for Indoor Air Quality Control. In *2023 19th International Conference on Distributed*

Computing in Smart Systems and the Internet of Things (DCOSS-IoT), pages 74–76, June 2023. ISSN: 2325-2944.

[84] Wamiq Raza, Anas Osman, Francesco Ferrini, and Francesco De Natale. Energy-Efficient Inference on the Edge Exploiting TinyML Capabilities for UAVs. *Drones*, 5(4):127, December 2021. Number: 4 Publisher: Multidisciplinary Digital Publishing Institute.

[85] Ming Zhao, Chen Chen, Lei Liu, DaPeng Lan, and Shaohua Wan. Orbital collaborative learning in 6G space-air-ground integrated networks. *Neurocomputing*, 497:94–109, August 2022.

[86] Zijing Wu, Ce Zhang, Xiaowei Gu, Isla Duporge, Lacey F. Hughey, Jared A. Stabach, Andrew K. Skidmore, J. Grant C. Hopcraft, Stephen J. Lee, Peter M. Atkinson, Douglas J. McCauley, Richard Lamprey, Shadrack Ngene, and Tiejun Wang. Deep learning enables satellite-based monitoring of large populations of terrestrial mammals across heterogeneous landscape. *Nature Communications*, 14(1):3072, May 2023. Publisher: Nature Publishing Group.

[87] Xuanyu Cao, Tamer Başar, Suhas Diggavi, Yonina C. Eldar, Khaled B. Letaief, H. Vincent Poor, and Junshan Zhang. Communication-Efficient Distributed Learning: An Overview. *IEEE Journal on Selected Areas in Communications*, 41(4):851–873, April 2023. Conference Name: IEEE Journal on Selected Areas in Communications.

[88] Vansh Gupta, Alka Luqman, Nandish Chattopadhyay, Anupam Chattopadhyay, and Dusit Niyato. TravellingFL: Communication Efficient Peer-to-Peer Federated Learning. *IEEE Transactions on Vehicular Technology*, 73(4):5005–5019, April 2024. Conference Name: IEEE Transactions on Vehicular Technology.

[89] Jilin Zhang, Hangdi Tu, Yongjian Ren, Jian Wan, Li Zhou, Mingwei Li, Jue Wang, Lifeng Yu, Chang Zhao, and Lei Zhang. A Parameter Communication Optimization Strategy for Distributed Machine Learning in Sensors. *Sensors*, 17(10):2172, October 2017. Number: 10 Publisher: Multidisciplinary Digital Publishing Institute.

[90] Nei Kato, Bomin Mao, Fengxiao Tang, Yuichi Kawamoto, and Jiajia Liu. Ten Challenges in Advancing Machine Learning Technologies toward 6G. *IEEE Wireless Communications*, 27(3):96–103, June 2020. Conference Name: IEEE Wireless Communications.

[91] Eugenio Muscinelli, Swapnil Sadashiv Shinde, and Daniele Tarchi. Overview of Distributed Machine Learning Techniques for 6G Networks. *Algorithms*, 15(6):210, June 2022. Number: 6 Publisher: Multidisciplinary Digital Publishing Institute.

[92] J.S. Mertens, L. Galluccio, and G. Morabito. Federated learning through model gossiping in wireless sensor networks. In *2021 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, pages 1–6, May 2021.

[93] Byungjun Lee, Ho Kuen Song, Youngho Suh, Kyung Hwan Oh, and Hee Yong Youn. Energy-Efficient Gossiping Protocol of WSN with Realtime Streaming Data. In *2014 IEEE 12th International Conference on Dependable, Autonomic and Secure Computing*, pages 219–224, August 2014.

[94] Dishita Naik, Paul Grace, Nitin Naik, Paul Jenkins, Durgesh Mishra, and Shaligram Prajapat. An Introduction to Gossip Protocol Based Learning in Peer-to-Peer Federated Learning. In *2023 IEEE International Conference on ICT in Business Industry & Government (ICTBIG)*, pages 1–8, December 2023.

[95] Lina Altoaimy, Heba Kurdi, Arwa Alromih, Amirah Alomari, Entisar Alrogi, and Syed Hassan Ahmed. Enhanced Distance-Based Gossip Protocols for Wireless Sensor Networks. In *2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pages 1–4, January 2019. ISSN: 2331-9860.

[96] Jakub Konečný, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated Learning: Strategies for Improving Communication Efficiency, October 2017. arXiv:1610.05492 [cs].

[97] Runhua Xu, Nathalie Baracaldo, Yi Zhou, Ali Anwar, and Heiko Ludwig. HybridAlpha: An Efficient Approach for Privacy-Preserving Federated Learning. In *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, AISec'19, pages 13–23, New York, NY, USA, November 2019. Association for Computing Machinery.

[98] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference, December 2017. arXiv:1712.05877 [cs, stat].

[99] Alireza Danaee, Rodrigo C. de Lamare, and Vítor H. Nascimento. Energy-Efficient Distributed Learning with Adaptive Bias Compensation for Coarsely Quantized Signals. In *2021 IEEE Statistical Signal Processing Workshop (SSP)*, pages 61–65, July 2021. ISSN: 2693-3551.

[100] Song Han, Jeff Pool, John Tran, and William J. Dally. Learning both Weights and Connections for Efficient Neural Networks, October 2015. arXiv:1506.02626 [cs].

[101] Jaewoo Song and Fangzhen Lin. PocketNN: Integer-only Training and Inference of Neural Networks via Direct Feedback Alignment and Pocket Activations in Pure C++, May 2022. arXiv:2201.02863 [cs].

[102] Nil Llisterri Giménez, Junkyu Lee, Felix Freitag, and Hans Vandierendonck. The Effects of Weight Quantization on Online Federated Learning for the IoT: A Case Study. *IEEE Access*, 12:5490–5502, 2024. Conference Name: IEEE Access.

[103] Alham Fikri Aji and Kenneth Heafield. Sparse Communication for Distributed Gradient Descent. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 440–445, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[104] Yujun Lin, Song Han, Huizi Mao, Yu Wang, and William J. Dally. Deep Gradient Compression: Reducing the Communication Bandwidth for Distributed Training, June 2020. arXiv:1712.01887 [cs, stat].

[105] Song Han, Huizi Mao, and William J. Dally. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, February 2016. arXiv:1510.00149 [cs].

[106] Hongyi Wang, Scott Sievert, Zachary Charles, Shengchao Liu, Stephen Wright, and Dimitris Papailiopoulos. ATOMO: Communication-efficient Learning via Atomic Sparsification, November 2018. arXiv:1806.04090 [cs, stat].

[107] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical Secure Aggregation for Privacy-Preserving Machine Learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, pages 1175–1191, New York, NY, USA, October 2017. Association for Computing Machinery.

[108] Paul Vanhaesebrouck, Aurélien Bellet, and Marc Tommasi. Decentralized Collaborative Learning of Personalized Models over Networks, February 2017. arXiv:1610.05202 [cs, stat].

[109] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the Convergence of FedAvg on Non-IID Data, June 2020. arXiv:1907.02189 [cs, math, stat].

[110] Aymen Rayane Khouas, Mohamed Reda Bouadjenek, Hakim Hacid, and Sunil Aryal. Training Machine Learning models at the Edge: A Survey, March 2024. arXiv:2403.02619 [cs].

[111] Omid Abari, Haitham Hassanieh, Michael Rodriguez, and Dina Katabi. Millimeter Wave Communications: From Point-to-Point Links to Agile Network Connections. In *Proceedings of the 15th ACM Workshop on Hot Topics in Networks*, HotNets '16, pages 169–175, New York, NY, USA, November 2016. Association for Computing Machinery.

[112] Viktor Losing, Barbara Hammer, and Heiko Wersing. Interactive online learning for obstacle classification on a mobile robot. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, July 2015. ISSN: 2161-4407.

[113] 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.

[114] Xiao Zheng, Qiuyue Zhang, Xuan Tang, Xiujun Wang, and Chunlai Du. Efficient Online and Privacy-preserving Medical Pre-diagnosis Based on Growing Learning Vector Quantization. In *2022 Tenth International Conference on Advanced Cloud and Big Data (CBD)*, pages 85–90, November 2022.

[115] Ye Xu, Shen Furao, Osamu Hasegawa, and Jinxi Zhao. An Online Incremental Learning Vector Quantization. In Thanaruk Theeramunkong, Boonserm Kijsirikul, Nick Cercone, and Tu-Bao Ho, editors, *Advances in Knowledge Discovery and Data Mining*, pages 1046–1053, Berlin, Heidelberg, 2009. Springer.

[116] Te-Chuan Chiu, Yuan-Yao Shih, Ai-Chun Pang, Chieh-Sheng Wang, Wei Weng, and Chun-Ting Chou. Semisupervised Distributed Learning With Non-IID Data for AIoT Service Platform. *IEEE Internet of Things Journal*, 7(10):9266–9277, October 2020. Conference Name: IEEE Internet of Things Journal.

[117] Elizabeth Rani. G, Sakthimohan. M, Abhigna Reddy. G, Selvalakshmi. D, Thomalika Keerthi, and Raja Sekar. R. MNIST Handwritten Digit Recognition using Machine Learning. In *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, pages 768–772, April 2022.

[118] Jing Xie, Xiang Yin, Xiyi Zhang, Juan Chen, and Quan Wen. Personalized Federated Learning with Gradient Similarity. In *2021 18th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, pages 268–271, December 2021. ISSN: 2576-8964.

[119] Mark D. Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E. Bourne, Jildau Bouwman, Anthony J. Brookes, Tim Clark, Mercè Crosas, Ingrid Dillo, Olivier Dumon, Scott Edmunds, Chris T. Evelo, Richard Finkers, Alejandra Gonzalez-Beltran, Alasdair J. G. Gray, Paul Groth, Carole Goble, Jeffrey S. Grethe, Jaap Heringa, Peter A. C. 't Hoen, Rob Hooft, Tobias Kuhn, Ruben Kok, Joost Kok, Scott J. Lusher, Maryann E. Martone, Albert Mons, Abel L. Packer, Bengt Persson, Philippe Rocca-Serra, Marco Roos, Rene van Schaik, Susanna-Assunta Sansone, Erik Schultes, Thierry Sengstag, Ted Slater, George Strawn, Morris A. Swertz, Mark Thompson, Johan van der Lei, Erik van Mulligen, Jan Velterop, Andra Waagmeester, Peter Wittenburg, Katherine Wolstencroft, Jun Zhao, and Barend Mons. The FAIR Guiding Principles for scientific data management and stewardship. *Scientific Data*, 3(1):160018, March 2016. Publisher: Nature Publishing Group.