ON DEVICE LEARNING FOR HUMAN ACTIVITY RECOGNITION ON LOW POWER MICROCONTROLLERS

by

Muhammed Talha Karagül

Ensar Muhammet Yozgat

Feyzullah Asıllıoğlu

CSE4197 Engineering Project 1

Project Specification Document

Supervised by:

Asst. Prof. Sanem Arslan Yılmaz

Marmara University, Faculty of Engineering

Computer Engineering Department

15.11.2024

Table of Contents

1. PROBLEM STATEMENT

Protecting personal data and ensuring privacy have become fundamental requirements in the digital age. Securely processing sensitive information and safeguarding personal data from external platforms can be achieved through on-device learning techniques, particularly on low-power microcontrollers. In this project, we use Tiny Machine Learning algorithms for human activity recognition to do data processing on low-cost, low-power devices. We aim to design an effective solution that preserves data privacy while training models on local data.

2. PROBLEM DESCRİPTİON AND MOTİVATİON

As digital devices become an inseparable part of our daily lives, personal data protection and security are increasingly important. Our smartphones, wearable devices, and other daily technologies collect extensive data, from movements to habits. While this data is processed to provide better user experiences, it poses significant security risks. This situation has made individual privacy protection an undeniable necessity in the digital world.

Human Activity Recognition (HAR) systems provide valuable information by monitoring users' daily activities. However, processing this data on central servers means sharing personal information without full control and creates significant privacy risks. Even seemingly simple information like sitting or walking can become a security threat when it falls into the wrong hands. Therefore, processing personal data directly on the device without transferring it to external platforms holds great potential for ensuring user privacy.

The main objective of this project is to ensure the highest level of privacy while training machine learning models on low-power devices. Our developed model will recognize human activities by processing them directly on the device, thus securing user information without transferring it to external platforms. Our project aims not only to ensure privacy but also to achieve these goals with low power consumption. By developing machine learning models that operate with low power on the device, we aim to increase energy efficiency and reduce costs.

On-device learning techniques emerge as a new field where research is still limited. With this project, we aim to both make an innovative contribution to the literature and provide a cost-effective and sustainable solution that considers energy efficiency while protecting user privacy.

3. MAIN GOAL AND OBJECTIVES

3.1. Main Goal:

Our main goal is to implement on-device learning for Human Activity Recognition (HAR) on low-power microcontrollers like STM32L476. This includes model personalization for HAR while maintaining privacy and improving performance through federated learning.

3.2. Objectives:

The necessary objectives to achieve this main goal are listed below.

- To deploy a cloud-based model on the STM32L476 board to perform on-device inference with 1-dimensional convolutional neural network (1D CNN) algorithms.
- To re-train only the dense layer of a cloud-based model using local data on the STM32L476 board.
- To re-train all layers of the cloud-based model using local data on the STM32L476 board.
- To implement federated learning through training the 1D CNN model with local data using multiple devices, sending only the model parameters to a server, creating a global model by aggregating weights on the server, and sending back global model parameters to the devices.

4. RELATED WORK

Craighero et al. [1] focus on Human Activity Recognition for microcontrollers and try to tackle the challenge of allowing personalized model adaptation directly on low-power devices such as the STM32. Traditional HAR models have been formally

trained on large datasets at servers, but their study argues that such a setting inherently limits personalization because data sharing will be restricted due to privacy issues. To address this challenge, they propose an on-device learning solution that enables microcontrollers to fine-tune a pre-trained deep learning model using local data on the device and hence support privacy-sensitive and personalized HAR.

This setup enables the model to support the retraining of all layers through backpropagation, while simultaneously providing tools for estimating memory and computational resource requirements, which assist in managing these constraints. Their study evaluates the model on the WISDM and ST datasets, demonstrating that personalized training significantly enhances model accuracy, thereby reducing inter-subject variability in activity patterns.

Their study focuses on realizing on-device personalization for HAR on low-power microcontrollers, hence being close to our project goals. In their study, they do not perform entirely on-device training but rather compare a global model sequentially by applying no personalization, retraining only the dense layer (partial personalization), and retraining all layers (full personalization). In addition to their study, our project focuses on federated learning, a methodology enabling entirely model train across multiple devices utilizing local data, thereby enhancing both privacy and adaptability. The training process exclusively employs pre-trained models to optimize resource efficiency further and ensure robust initial performance.

Lin et al. [2] demonstrate that while machine learning models can generally be trained on resource-constrained edge devices, when this limitation goes down to 256KB memory in IoT microcontrollers, the challenge increases. The traditional frameworks such as PyTorch and TensorFlow require more memory than these devices can support. This paper introduces a framework with both algorithmic and system-level optimizations, featuring two main techniques: Quantization-Aware Scaling (QAS) and Sparse Update.

Lin et al. show how the proposed framework scales on-device training to microcontrollers with full-precision accuracy comparable to models trained in the cloud but with much less memory. This allows for a range of applications whereby

IoT devices can learn continuously and adapt locally, rather than just performing inference, to support privacy-sensitive personalized applications. While our study focuses on personalization for HAR on microcontrollers, their study focuses on general training stability and memory reduction, neither of which targets HAR or real-time applications.

Khoua et al. [3] review some benefits and challenges of training on resource-limited devices such as smartphones and IoT systems in this emerging field of edge learning where models are being trained on decentralized edge devices rather than on the cloud. It categorizes the edge learning methods that mainly focus on distributed approaches, including Federated Learning and Split Learning. The models in Federated Learning are usually trained locally at each device and sometimes aggregated into a global model. Split Learning splits the models between devices and servers, sharing the load of training while preserving data privacy. Other on-device training techniques discussed are transfer learning, model compression, and adaptive inference. Although this paper provides a comprehensive overview of edge learning strategies, it does not specifically address Human Activity Recognition (HAR) or on-device personalization for microcontrollers, which is the aim of our project.

Daghero et al. [4] describe a lightweight deep-learning architecture for efficient Human Activity Recognition on low-power wearables. Traditional solutions of HAR on wearables adopt simpler models because of the limited computing and memory capabilities. However, their study introduces a deep learning approach fitted with wearables requiring minimal memory and energy consumption.

They utilized 1D CNNs with sub-byte quantization and adaptive inference for microcontroller constraint adaptation. Their approach yields performance superior to shallow machine learning models on four benchmark datasets, allowing for up to 60% reduction in computational load without sacrificing accuracy. It shows the deep learning model optimization for ultra-low-power, real-time activity recognition, which enables always on-device monitoring without reliance on clouds. Though closely related to our work, the study focuses more on inference optimization than model training on devices for HAR.

5. SCOPE

This project is focused on processing and analyzing movement data exclusively on the STM32 microcontroller. Our work is strictly limited to the capabilities and constraints of the STM32 board, and we do not aim to ensure compatibility or functionality with other hardware or microcontroller platforms. The primary objective is to develop and test data processing algorithms without creating a real-time application. For this purpose, a prerecorded WISDM dataset with six classes (walking, jogging, ascending stairs, descending stairs, sitting, and standing) will be used, which will simulate sensor data without involving any live streaming [5]. Below are the specific constraints and assumptions that define the boundaries of our project.

5.1. Constraints

- The project is designed to work strictly within the STM32 microcontroller, and compatibility with other hardware or microcontrollers is not within the scope.
- The project will utilize the Wireless Sensor Data Mining (WISDM) dataset, which contains 1,098,207 raw time series examples across six specific activities: walking (38.6%), jogging (31.2%), ascending stairs (11.2%), descending stairs (9.1%), sitting (5.5%), standing (4.4%) and have 36 users. The transformed version of this dataset includes 5,424 examples with 46 attributes. Complex activities or additional datasets are beyond the project's scope.
- The STM32 microcontroller has a memory limit of 128 KB, requiring careful model optimization to fit within this restriction.
- Data transfer will rely solely on the UART protocol, as the STM32 board lacks Ethernet or wireless connectivity, necessitating all data processing to occur locally on the device.
- The device will not contain any in-built sensors or external data collection components; all testing and processing will rely on pre-recorded data.
- The scope will remain focused on Human Activity Recognition (HAR), with no solutions extending to other application domains.

5.2. Assumptions

- We assume that we can easily access the WISDM dataset and that the dataset's six classes will be sufficient for our model to achieve meaningful activity recognition within the project's goals.
- The data for one individual from the WISDM dataset can be stored in the STM32's memory without exceeding capacity limits.
- We assume the 1D CNN model will be successfully implemented in C and can be executed without issues on the STM32 board, given the available computational and memory resources.
- UART-based communication on the STM32 board is expected to function reliably for data transfer during testing.

6. METHODOLOGY AND TECHNICAL APPROACH

Our approach combines three powerful methodologies: Tiny Machine Learning (TinyML), federated learning, and the STM32L476RGx microcontroller to provide a human activity recognition system directly on embedded devices. Each of them brings unique capabilities that help to meet the challenges of on-device data processing, limited power, and privacy constraints. Federated learning provides data privacy in that models can be trained locally on each device, thus not centralizing user data, while TinyML guarantees efficient machine learning on resource-constrained hardware. The STM32L476RGx board with a low-power ARM Cortex-M4 processor is an ideal platform to deploy these methodologies since it allows us to maximize energy efficiency and computational performance, and is commonly used in wearable technologies. Together, these techniques lay the foundation for a high-performance, privacy-centric HAR system, driving embedded AI capabilities in a direction that embraces security and adaptability.

6.1. TinyML

Traditional applications of machine learning usually run on large server infrastructures, demanding high processing power, memory, and energy. Today's devices are smaller, portable, and energy-efficient, but there is certainly a need for

strong analysis to be enabled on small devices. TinyML is a means of deploying AI models onto resource-constrained embedded devices, thereby bringing the analytic capabilities of larger systems to smaller, portable devices. It includes optimization techniques for energy efficiency and processing capacity so that advanced analysis can be done even on hardware-constrained devices like microcontrollers. It provides substantial benefits for devices like IoT devices, sensors, smart home systems, and wearables that are always carried with us. Another huge advantage of TinyML is that personal data is processed directly on a device. Hence, data privacy is preserved, and real-time data analysis becomes an option.

TinyML can be a powerful tool in human activity recognition applications with such a feature set. Human Activity Recognition systems analyze data coming from sensors to identify different activities that a user performs. Finding the repetitive patterns in this type of sequential data is very important. The 1 Dimensional Convolutional Neural Network (1D CNN) [6] is an ideal solution for sequential pattern recognition. As a key component of TinyML, it excels in analyzing time series data. In 1D CNN, filters (kernels) are slid over the data to find specific patterns and classify the data based on these patterns. For example, the convolution layers in the 1D CNN can detect sequential patterns of movement that occur while walking or running, therefore analysis could be done efficiently.

The general architecture of a 1D CNN shown in Figure 1 contains several essential layers. First is the convolution layer, which provides sliding filters over the input data to extract specific features, perfect for the identification of movement patterns. The convolutional layer output is then downsampled by the pooling layer, which saves memory, speeds up computation, and generally prevents overfitting, increasing accuracy. Then, an activation layer processes the extracted features in a nonlinear way, enabling the model to learn more complicated patterns. To conclude, all the extracted features are combined by the fully connected dense layer to generate final predictions. This layer identifies the user's current activity in applications such as human activity recognition. The 1D CNN can recognize such complicated motion patterns even on a small device because of this structure, hence it is the preferable choice in energy-efficient system design [7].

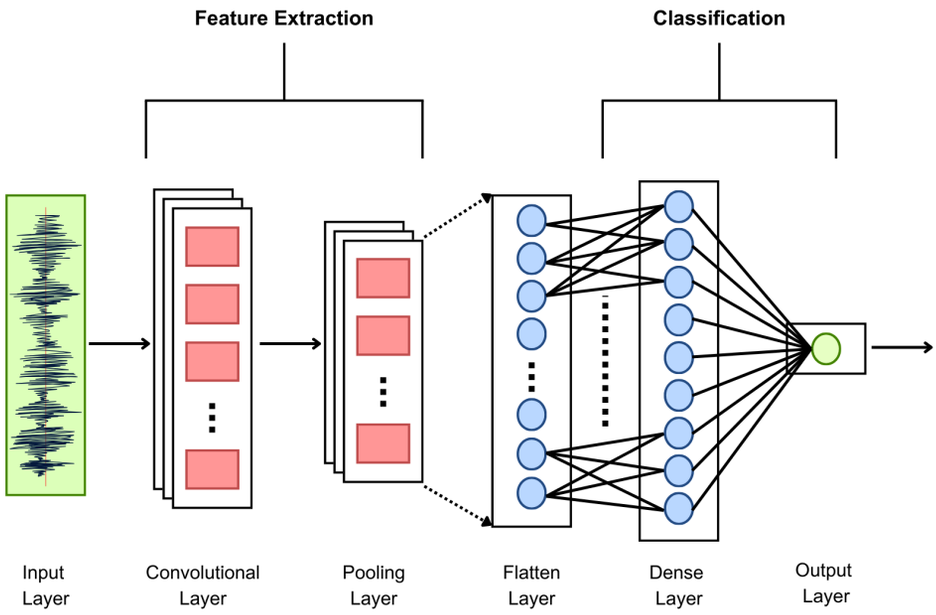

Figure 1: General architecture of a 1D CNN

The 1D CNN models are deployed on embedded systems with optimized tools such as the LiteRT. LiteRT is an optimized version of TensorFlow Lite, dedicated to devices with low hardware capabilities, where memory usage and processing power have to be at a minimum, enabling models like 1D CNN to run even on low-power microcontrollers [8]. LiteRT optimizes data privacy and energy efficiency, enabling models to operate securely on the device itself. This allows user data to remain private, where data processing is done entirely on the device. TinyML, along with LiteRT, enables small, power-constrained devices to have efficient AI capabilities while guaranteeing the output of being secure and high performance for applications like human activity recognition.

6.2. Federated Learning

Federated learning is an artificial intelligence approach to learning processes on local devices while preserving the privacy of users' data. Traditional machine-learning models are usually trained on a centralized server. However, in federated learning, each device will train its model on its data and then share only the model parameters. This way, the user's data never leaves the device, and it allows one to build a global model without compromising individual data privacy. Federated learning is an extremely powerful alternative to this, especially in scenarios where data sensitivity is a big concern. In this project, we will implement

different approaches like personalization and federated learning. Each approach provides a different level of personalization: the "No Personalization" phase, with a centrally trained base model; the "Partial Personalization" phase, with partial customization allowed; and finally, the "Full Personalization" phase, whereby every device re-train the global model. However, the entire model is first-time trained on each device in the federated learning approach. All of these steps will increase the capabilities of on-device learning while ensuring data privacy and model accuracy optimization.

The objective for splitting the WISDM is to split the dataset with %80 for training and %20 for testing. This 80-20 split is a standard approach for different machine learning model applications. However, implementing this ratio depends on the STM32L476RGx board's memory constraints. If memory limitations arise, it may impact our ability to fully adhere to this division.

6.2.1. Base Implementation (No-personalization):

The first step in this project will be to deploy 1D CNN models on the STM32L476 board without any local learning. This model will be trained with a sufficiently large, 6-class WISDM dataset in the cloud. The cloud-trained model will serve as the foundation for our on-device inference as shown in Figure 2.

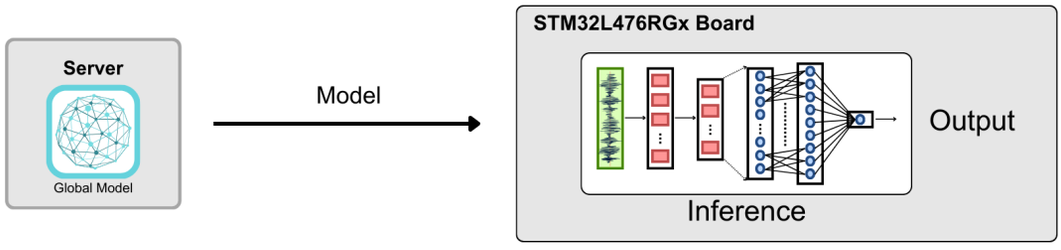

Figure 2: No personalization approach through local inference

Through this base implementation, we aim to demonstrate the feasibility of running complex machine-learning models on low-power embedded systems. This will provide a basis for the next steps of the project to ensure the performance of the model and the preservation of the data. By starting with a well-performing generic model, we can focus on on-device learning with local data, which will allow us to understand the model and determine a boundary success rate for testing.

6.2.2. Partial Personalization and Full Personalization:

In the next section, we will build upon the baseline model by enabling partial personalization on the STM32L476 board. Instead of running the pre-trained model as-is, we will leverage the LiteRT framework to fine-tune the model using local data on the device. The key idea here is to keep the pre-trained convolutional layers of the 1D CNN model frozen and only retrain the dense layers using the data that is already on the device on the individual STM32L476 board as shown in Figure 3.

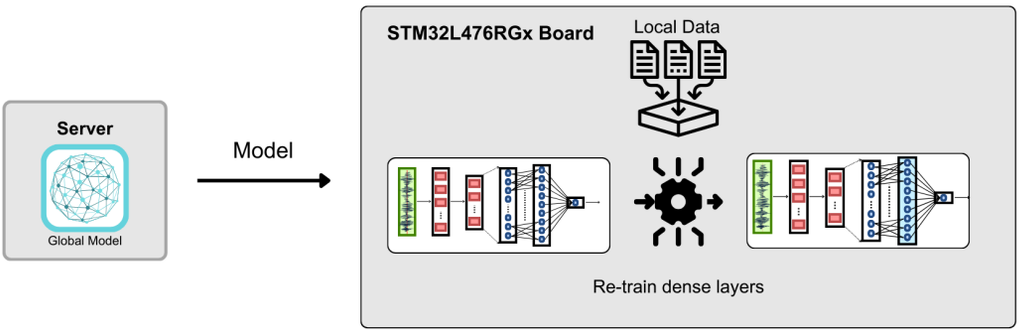

Figure 3: Partial personalization approach through local re-training

This method is expected to allow the model to make better inferences about unique user data without consuming much more computing power such as training the model completely than this approach. Compared to the initial base implementation, this partial personalization technique should demonstrate improved accuracy for the individual user, without significantly increasing the computational or memory requirements on the STM32L476 board.

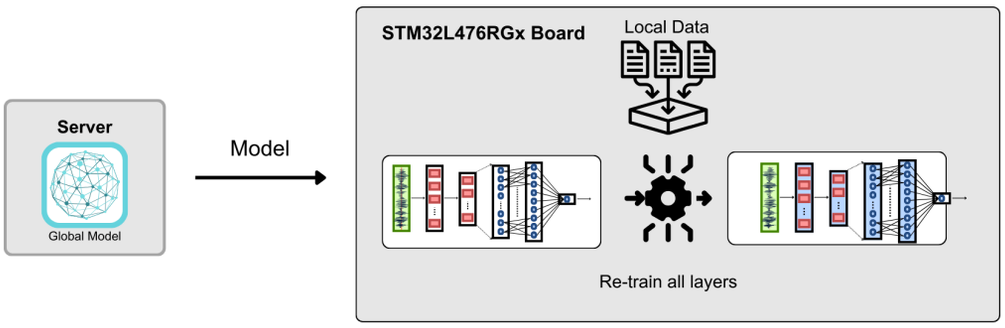

Figure 4: Full personalization approach through local re-training

On the other hand, full personalization includes the retraining of all layers from the global model on the device unlike partial as illustrated in Figure 4. Additionally,

this approach provides better accuracy because it allows more customization for users.

6.2.3. Federated Learning

In the final phase, we will implement a complete on-device training approach. Unlike the previous methods, we start without a cloud-trained model, instead building and training the model directly on the STM32L476 boards using local WISDM data.

Each device independently trains its 1D CNN model using the LiteRT framework, adapting all layers to the user's specific data. This represents a significant development over personalization approaches as the entire model architecture, including all CNN layers, is trained for the first time using local data. Federated learning implementation represents the most sophisticated level of on-device learning in our project, balancing individual user adaptation with collective model improvement while maintaining strict privacy requirements.

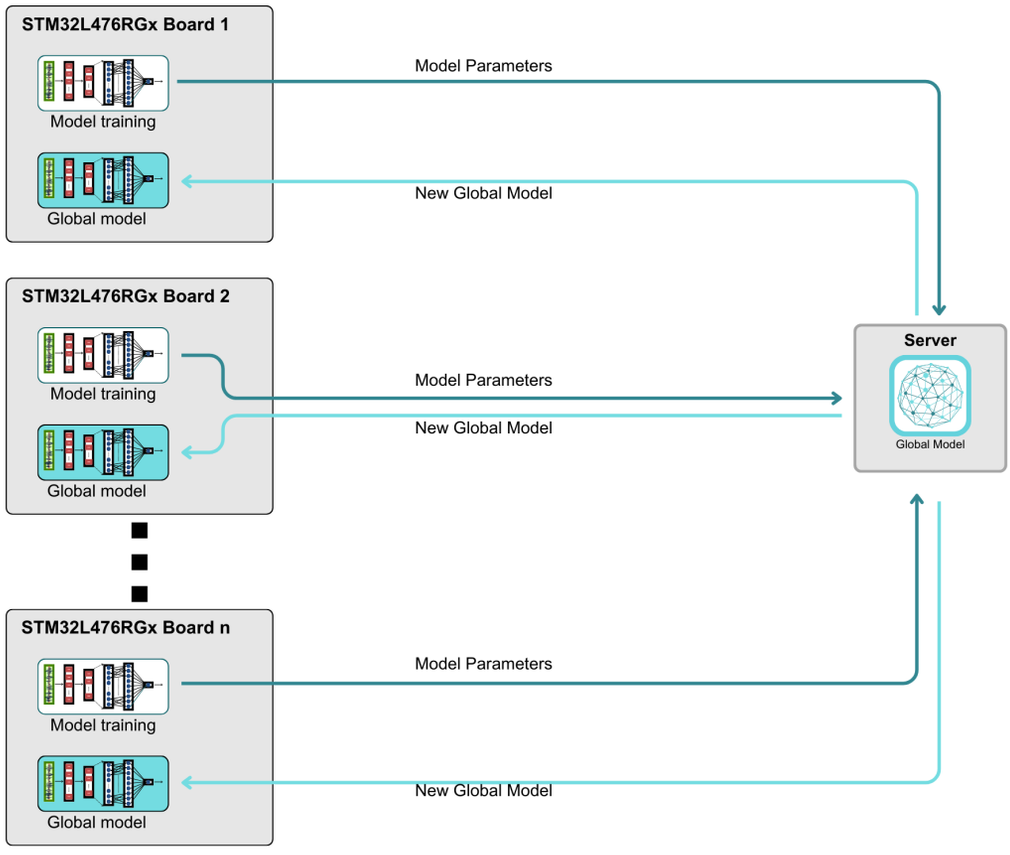

Figure 5: Federated learning process for full personalization

For the federated learning method, the main communication protocol will be a UART-based system. Each board only transmits the model parameters, which are trained with local data to the server with this UART-based system. To generate a global model that includes the various experiences of every user, the cloud server aggregates these parameters collected from different boards and creates a new global model. As shown in Figure 5 above, this global model will be placed back on the boards with the same UART system and will work as a new model on each device.

6.3. STM32L476RGx Board

Our main development card will be STM32L476RGx as shown in Figure 6 for implementing machine learning algorithms. The STM32L476xx devices are ultra-low-power microcontrollers based on the high-performance Arm® Cortex®-M4 32-bit RISC core operating at a frequency of up to 80 MHz [9].

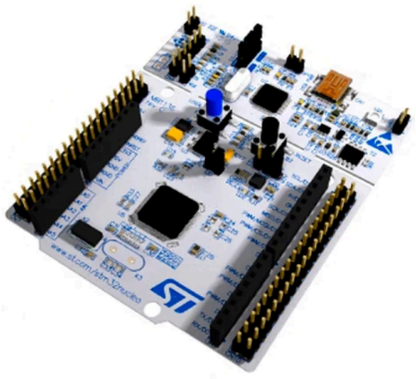

Figure 6: STM32L476RGx development Board [10]

Since the STM32L476RG lacks an Ethernet connection, alternative methods like UART-based communication [11] will allow each microcontroller to exchange model parameters or gradients across the connected system. While Ethernet would offer higher speed, UART can provide a reliable solution for low-bandwidth applications where power efficiency and simplicity are priorities.

STM32CubeIDE, provided free of charge by STMicroelectronics, will be our main development environment. STM32CubeIDE offers coding, advanced debugging, and tools for deploying firmware into the device in a single application, which is a great advantage [12].

7. PROFESSIONAL CONSIDERATIONS

7.1. Methodological considerations/engineering standards

- We will use GitHub in the implementation of the models. GitHub provides a version control environment and collaboration across team members.
- We will implement our project with C/C++.
- We will use the WISDM dataset which has 6 classes. These classes are “Walking”, “Jogging”, “Upstairs”, “Downstairs”, “Sitting”, and “Standing”.
- STM32CubeIDE will be our main IDE. And we will use this program to upload our code and models.
- We will use a universal asynchronous receiver transmitter (UART) for communication across boards and our main server.
- We will use the LiteRT framework for implementing 1D CNN.

7.2. Realistic Constraints

7.2.1. Economical

We will utilize our existing STM32L476 development boards and leverage open-source software solutions for development, eliminating any need for additional hardware purchases or software licensing costs. This efficient use of available resources allowed us to focus on the technical implementation aspects of on-device learning.

7.2.2. Environmental

In our project, the impact on the environment will be minimal, making it even more environmentally friendly, as the machine learning model training will be implemented on a low-power microcontroller.

7.2.3. Sustainability

Our project is designed to support sustainability in many ways. The STM32L476 microcontroller used contributes to the goal of sustainable technology development by minimizing energy consumption due to its low power consumption. The main goal of the project is to realize machine learning applications such as human activity recognition (HAR) without transferring data to the cloud and processing it only on the device. This approach reduces the need for data transfer, which in turn reduces the energy consumption of the network infrastructure.

7.2.4. Ethical

In our project, we will use open-source resources and avoid any unauthorized use of patented designs or concepts. Additionally, by demonstrating that data can remain private, we strengthen our commitment to ethical standards.

7.2.5. Health and Safety

There are no risks to users or the public's health or safety from our project. It is safe to use and creates no risks to people because it only requires code and a microcontroller. Furthermore, by focusing on Human Activity Recognition, this project has the potential to positively impact public health.

7.2.6. Social

As stated in previous sections, our project does not involve any discrimination based on social values, and it is not harmful to individuals.

7.3. Legal Considerations

The programs that will be used in our project are open source and we do not need any licenses for developing our project.

8. MANAGEMENT PLAN

8.1. Description of Task Phases

Phase 1: Literature review and understanding of 1D CNN for Human activity recognition.

- Research the fundamentals of 1D CNN structures for Human Activity Recognition (HAR) applications.
- Analyze previous studies to assess how this project can contribute to the current literature.

Phase 2: Preparation of PSD.

Phase 3: Implement a 1D CNN algorithm onto the STM32L476 board.

- Implement the 1D CNN model using C/C++ on the STM32L476 board and train it on cloud servers with the WISDM dataset.
- Deploy the trained model onto the device to test inference performance and evaluate its accuracy.

Phase 4: Implement a partial learning for personalization.

- Freeze the convolutional layers of the model and retrain only the dense layers with local data for personalization.
- Observe whether partial personalization improves accuracy for individual users.

Phase 5: Implement a full-on device re-training global model with local data.

- Enable full model re-training of the global model on the device using local data, achieving complete personalization.

Phase 6: Implement a UART-based communication system for federated learning methods.

- Set up a UART-based communication protocol to securely transmit model parameters from devices to a central server.

Phase 7: Implement Federated learning methods with on-device models with on-device learned model with local data.

- Aggregate locally trained model parameters from each device to create a centralized global model.
- Update each device with the improved global model, ensuring privacy while enhancing overall accuracy.

Phase 8: Preparing Final Project Documents.

8.2. Division of Responsibilities and Duties Among Team Members

All the tasks will be shared equally between all the team members. As represented in Figure 7.

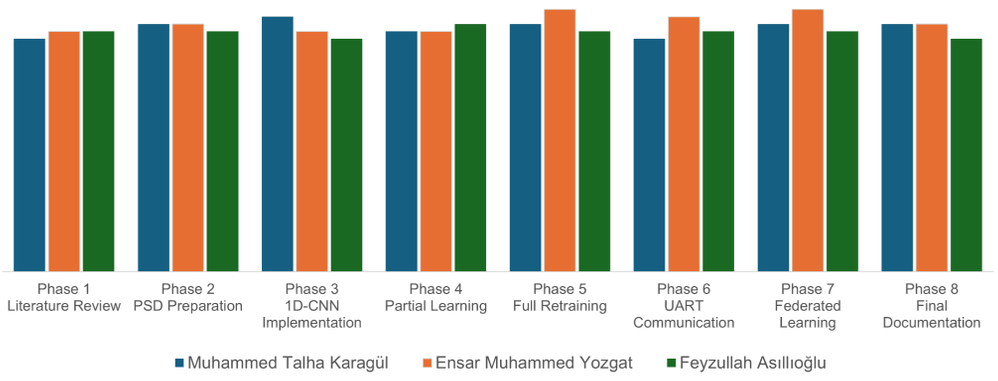

Figure 7: Division of Responsibilities Among Group Members

There are 3 milestones for our project:

Milestone 1: Deploying no personalization model on the board.

Milestone 2: Implementing partial personalization and full personalization of the global model with local data on the device.

Milestone 3: Implementing federated learning with entire model training on the device methods across the boards.

A schedule for phases and milestones is shown in Figure 8 below.

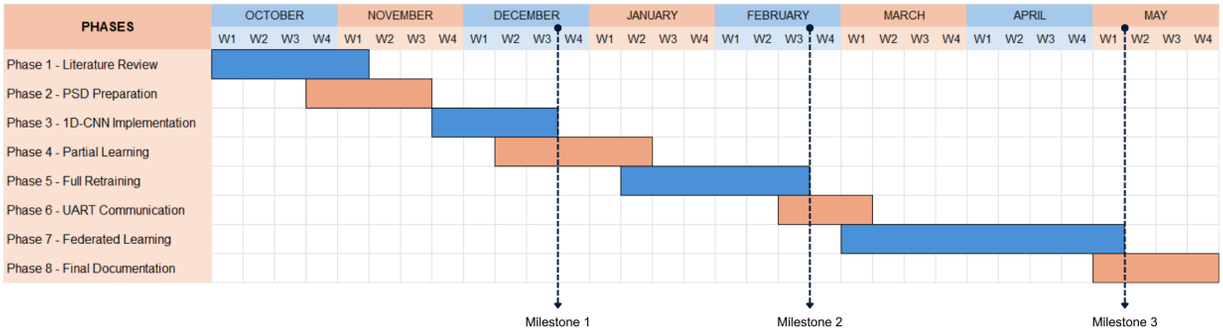

Figure 8: Project Timeline with Milestones

9. SUCCESS FACTORS AND RISK MANAGEMENT

9.1. Success Factors

Success Factor for Objective 1: In our initial implementation without personalization, we aim to achieve an F1 score of 0.8, as a benchmark similar to that in On-device Personalization for HAR on STM32 [1].

Success Factor for Objective 2: With partial personalization, our goal is to reach an F1 score of around 0.95, aligning with results achieved through partial personalization in the literature [1].

Success Factor for Objective 3: Since all layers are re-trained with local data, we expect that full personalization will give a better result than either of these previous objectivists. At this stage, we expect the F1 score to be 0.97 similar to the values obtained in the literature [1].

Success Factor for Objective 4: With the use of federated learning across devices, we aim for improved accuracy and more consistent performance, expecting a minimum F1 score of 0.8 since in this approach we will not use cloud-based trained models and will not use any personalization.

9.2. Risk Management

Risk 1: Uart protocol may not be enough to implement federated learning methods.

Resolution: Since the STM32L476 does not include ethernet, we may need to switch to a board with ethernet. If necessary, we can consider alternative

development boards that offer built-in ethernet capabilities. However, before making this transition, we will thoroughly evaluate the UART performance limits in our initial implementation and assess whether the data transfer rates are sufficient for our federated learning requirements. We may also explore optimizing our communication protocol and data compression techniques to maximize UART efficiency before deciding on hardware changes.

Risk 2: STM32L476 may not have enough memory space when training all layers on the device.

Resolution: We may need to add additional memory units. To address this limitation, we will first optimize our model architecture and implement memory-efficient training techniques such as gradient quantization and pruning. If these optimizations prove insufficient, we can explore external memory solutions like adding SRAM or switching to a board with a larger memory as a more radical option.

Risk 3: The Wireless Sensor Data Mining (WISDM) dataset may not be sufficiently large or suitable for our 1D CNN model requirements.

Resolution: We may prefer to use the ST dataset [13] in addition to the WISDM dataset, or we may consider other datasets as needed.

10. BENEFITS AND IMPACT OF THE PROJECT

10.1. Scientific impact:

This project helps to progress the state-of-the-art in HAR by presenting the first proof-of-concept for practical model personalization on microcontrollers, specifically on a device like an STM32, without large-scale external resources. That work has refined the model on-device and has contributed to the development of real-time, adaptive machine-learning applications that can operate with minimal resources. This enhancement in on-device learning frameworks is helpful to the scientific community in exploring further personalized and continuous learning for HAR and other sensor-based applications where privacy, latency, and adaptability are of concern.

10.2. Economic/Commercial/Social Impact:

On-device personalization for resource-constrained devices gives very important commercial opportunities. This will provide an inexpensive alternative to cloud/server-dependent solutions, saving infrastructure and connectivity costs for companies deploying the HAR-based solution. Applications in healthcare, fitness, and wellness monitoring will have the benefits of personalized activity tracking with the assurance of user privacy, which has become more important in consumer markets. Socially, the project addresses privacy concerns by ensuring that sensitive data remains local, empowering users with control over their personal information.

10.3. Potential Impact on New Projects:

This project lays the groundwork for future applications in on-device learning and personalization of diverse fields. It allows the possibilities for new projects in IoT, edge computing, and wearable technologies, showing that such adaptive, user-specific model updates can be integrated directly on microcontrollers, thus setting a baseline for further research on memory and computational efficiency optimizations for microcontroller-based applications, possibly expanding the capabilities of other edge AI applications requiring real-time responsiveness.

10.4. Impact on National Security:

On-device personalization in HAR could also have an application in national security, specifically in surveillance, biometrics, and secure access systems, where locally stored data and processing are desired. That is consistent with security interests in supporting privacy-preserving technologies that minimize data transmission since personal or sensitive information does not need to be offloaded to remote servers. The solution can enable secure monitoring applications in the most critical environments where operational privacy and protection of data security are demanded.

REFERENCES

[1] Craighero, Quarantiello, Rossi, Carrera, Fragneto, Boracchi (2024) On-Device Personalization for Human Activity Recognition on STM32. IEEE EMBEDDED SYSTEMS LETTERS, VOL. 16, NO. 2, (JUNE 2024).

[2] Ji Lin,Ligeng Zhu,Wei-Ming Chen,Wei-Chen Wang, Chuang Gan, Song Han (2022) On-Device Training Under 256KB Memory. 36th Conference on Neural Information Processing Systems (NeurIPS 2022).

[3] Aymen Rayane Khouas, Mohamed Reda Bouadjenek, Hakim Hacid, and Sunil Aryal. (2014) Taining Machine Learning models at the Edge: A Survey

[4] Daghero, Burello, Xie, Castellano, Gandolfi, Calimera, Macii, Poncino, Pagliari (2022) Human Activity Recognition on Microcontrollers with Quantized and Adaptive Deep Neural Networks. ACM Transactions on Embedded Computing Systems, Vol. 21, No. 4, Article 46. Publication date: (August 2022)

[5] Jennifer R. Kwapisz, Gary M. Weiss and Samuel A. Moore (2010). Activity Recognition using Cell Phone Accelerometers, Proceedings of the Fourth International Workshop on Knowledge Discovery from Sensor Data (at KDD-10), Washington DC. http://www.cis.fordham.edu/wisdm/public_files/sensorKDD-2010.pdf

[6] Serkan Kiranyaz, Turker Ince , Osama Abdeljaber, Onur Avci, and Moncef Gabbouj. (2019) 1-D Convolutional Neural Networks for Signal Processing Applications. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (2019)

[7] Serkan Kiranyaz, Onur Avci, Osama Abdeljaber, Turker Ince, Moncef Gabbouj, Daniel J. Inman. (2021) 1D convolutional neural networks and applications: A survey. Mechanical Systems and Signal Processing Volume 151, April 2021, 107398

[8] Google. (2024, 11,06). LiteRT: https://ai.google.dev/edge/litert

[9] STMicroelectronics. (2024, 11 , 06). STM32L476RG Microcontroller: https://www.st.com/en/microcontrollers-microprocessors/stm32l476rg.html

[10] STMicroelectronics. (2024, 11, 06). Nucleo-L476RG Development Board: https://www.st.com/en/evaluation-tools/nucleo-l476rg.html

[11] Philips NXP. (2024, 11, 06), UART: https://www.nxp.com/docs/en/data-sheet/SCC2691.pdf

[12] STMicroelectronics. (2024, 11 , 06). STM32CubeIDE Development Tool: https://www.st.com/en/development-tools/stm32cubeide.html

[13] “Human activity recognition using CNN in Keras for sensor- tile.” (2024, 11, 8) : https://github.com/ausilianapoli/HAR-CNN-Keras-STM32/blob/master/Dataset.csv