A HYBRID PLATFORM FOR CONTEXT-AWARE V2X COMMUNICATIONS

FINAL PROJECT REPORT

by

Mohamed Hefeida, West Virginia University
Umair Mohammad, Florida International University
and
Sameh Sorour, Queen's University

Sponsorship
PacTrans, and University of Idaho

for
Pacific Northwest Transportation Consortium (PacTrans)
USDOT University Transportation Center for Federal Region 10
University of Washington
More Hall 112, Box 352700
Seattle, WA 98195-2700

In cooperation with US Department of Transportation-Research and Innovative Technology Administration (RITA)

DISCLAIMER

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated under the sponsorship of the U.S. Department of Transportation's University Transportation Centers Program, in the interest of information exchange. The Pacific Northwest Transportation Consortium, the U.S. Government and matching sponsor assume no liability for the contents or use thereof.

Technical Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
	01745556	

4. Title and Subtitle	5. Report Date
A Hybrid Platform for Context-Aware V2X Communication	
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Mohamed Hefeida, 0000-0003-4738-9415; Umair Mohamed, and Sameh Sorour	2019-S-UI-2

9. Performing Organization Name and Address	10. Work Unit No. (TRAIS)
PacTrans Pacific Northwest Transportation Consortium University Transportation Center for Region 10 University of Washington More Hall 112 Seattle, WA 98195-2700	11. Contract or Grant No. 69A355174110

12. Sponsoring Organization Name and Address	13. Type of Report and Period Covered
United States of America Department of Transportation Research and Innovative Technology Administration	14. Sponsoring Agency Code

15. Supplementary Notes

Report uploaded at www.pacTrans.org

16. Abstract

This report presents a new paradigm for Mobile Edge Learning ("MEL") that enables the implementation of realistic distributed machine learning (DML) tasks on wireless edge nodes while taking into consideration heterogeneous computing and networking environments. A heterogeneity aware (HA) scheme was designed to solve the problem of dynamic task allocation for MEL in a way that maximizes the DML accuracy over wireless heterogeneous nodes or "learners" while respecting time constraints. This will enable context aware vehicle to everything (V2X) communication. The problem was first formulated as a quadratically constrained integer linear program (QCILP). Being non-deterministic polynomial-time (NP)-hard, it was relaxed into a non-convex problem over real variables that could be solved using commercially available numerical solvers. The relaxation also allowed us to propose a solution based on deriving the analytical upper bounds of the optimal solution using Lagrangian analysis and Karush-Kuhn-Tucker (KKT) conditions. The merits of the proposed analytical solution were demonstrated by comparing its performance to numerical approaches and comparing the validation accuracy of the proposed HA scheme to a baseline heterogeneity unaware (HU) equal task allocation approach. Simulation results showed that the HA schemes decreased convergence time up to 56 percent and increased the final validation accuracy up to 8 percent.

17. Key Words	18. Distribution Statement
V2X, Mobile Edge Computing, Internet-of-Things (IoT)	No restrictions.

19. Security Classification (of this report)	20. Security Classification (of this page)	21. No. of Pages	22. Price
Unclassified.	Unclassified.		NA

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

CV2X	Cellular vehicle to everything
DD	Distributed datasets
DML	Distributed machine learning
ETA	Equal task allocation
FL	Federated learning
GD	Gradient descent
H-MEC	Hierarchicl-mobile edge computing
HA	Heterogeneity aware
HU	Heterogeneity unaware
IoT	Internet-of-things
IP	Interior-point
KKT	Karush-Kuhn-Tucker
MEC	Mobile edge computing
MEL	Mobile Edge Learning
ML	Machine learning
NP	Non-deterministic polynomial-time
QCILP	Quadratically-constrained integer linear program
QCLP	Quadratically constrained linear program
QCQP	Quadratically constrained quadratic program
SGD	Stochastic gradient descent
TP	Task parallelization
V2X	Vehicle to everything

EXECUTIVE SUMMARY

This report presents a new paradigm for Mobile Edge Learning ("MEL") that enables the implementation of realistic distributed machine learning (DML) tasks on wireless edge nodes while taking into consideration heterogeneous computing and networking environments. A heterogeneity aware (HA) scheme was designed to solve the problem of dynamic task allocation for MEL in a way that maximizes the DML accuracy over wireless heterogeneous nodes or "learners" while respecting time constraints, enabling context aware vehicle to everything (V2X) communication. The problem was first formulated as a quadratically constrained integer linear program (QCILP). Being non-deterministic polynomial-time (NP)-hard, it was relaxed into a non-convex problem over real variables that could be solved by using commercially available numerical solvers. The relaxation also allowed us to propose a solution based on deriving the analytical upper bounds of the optimal solution using Lagrangian analysis and Karush-Kuhn-Tucker (KKT) conditions. The merits of the proposed analytical solution were demonstrated by comparing its performance to numerical approaches and comparing the validation accuracy of the proposed HA scheme to a baseline heterogeneity unaware (HU) equal task allocation approach. Simulation results showed that the HA schemes decreased convergence time up to 56 percent and increased the final validation accuracy up to 8 percent.

.

CHAPTER 1 INTRODUCTION

Accelerated migration toward the deployment of smart infrastructure has resulted in the accumulation of many Internet-of-things (IoT) devices that generate huge amounts of data at the network edge. The transfer of such huge volumes of data distributed across multiple heterogeneous wireless networks to cloud servers for centralized processing faces many challenges in terms of latency, bandwidth, security, and privacy [2]. It is anticipated that edge servers or even processors/controllers attached to edge devices such as smart phones, drones, and closed-circuit cameras will perform 90 percent of the analytics [3]. Paradigms such as mobile edge computing (MEC) and hierarchical-MEC (H-MEC) [4], [5] can optimize the offloading and distribution of computational tasks. Although these approaches consider computing and communication heterogeneity, they do not consider the specifics of machine learning (ML)-based analytics across heterogeneous edge nodes. Hence, they are not suited to perform ML tasks in a distributed manner, a.k.a. distributed ML (DML) at the edge [1], [6], [7].

DML has recently attracted significant attention within the machine learning realm, motivated by two distinct yet related practical scenarios: distributed datasets (DD), which are usually referred to as federated learning (FL) in the literature [8], and task parallelization (TP). In DD, (big) datasets are separately generated/collected by multiple devices but cannot be transferred to a central hub because of some constraints (e.g., bandwidth, privacy) [6]. In this case, the learning process runs in sequential phases: nodes (a.k.a. learners) perform local training/learning on their individual datasets; then a central orchestrator collects locally updated models, performs global processing, and returns the globally updated model to the learners for the next phase. On the other hand, TP usually involves a node that parallelizes the learning process by distributing its locally generated dataset to multiple cores/nodes for local training followed by global updates [1]. Therefore, this main node acts as the orchestrator of the learning process.

This work introduces the novel “Mobile Edge Learning (MEL)” paradigm, which can accommodate both DD and TP. A unified model is important because each approach can be used to achieve different objectives for edge DML. DD is mainly used for preserving privacy and has been applied in areas such as healthcare [9] and vehicular networks [10]. In contrast, TP is not widely studied but can be applicable in cases where communication capabilities outweigh the computational resources. Moreover, it can offer some level of privacy in comparison to cloud-

based approaches by keeping data within the confines of a trusted edge environment. One applicable scenario would be an orchestrator such as an edge server that utilizes the resources of idle end devices and fog nodes [11]. Another example application is a set of drones flying together to capturing aerial scans from different angles. If an image from a particular direction needs to be analyzed on-board, the relevant drone may distribute the data to its peers for learning. Privacy-preserving TP can be applied for medical diagnostics with ML at the edge or for continuous monitoring of vulnerable adults in a care home setting where multiple sensors collecting data can collaborate with a trusted edge server. The design of recent incentive mechanisms for FL may facilitate the adoption of TP [12], [13].

Although DML was initially studied for wired/non-heterogeneous environments [14], [15], extending these DML studies to resource-constrained edge environments has gained significant importance. However, early works such as Wang et al. [7], Tuor et al. [16], Wang et al, [17], and Conway-Jones et al. [18] were heterogeneity unaware (HU); they did not consider the inherent heterogeneities in the computing and communication capacities of different edge learners and links, respectively. The implications of such heterogeneities for optimal task allocation, model selection, accuracy enhancement, and promoting time and energy efficient techniques are clearly game-changing but have only been recently investigated [19], [20], [21], [22]. However, these works have focused only on FL and optimizing the number of local updates without any task size allocation.

To the best of our knowledge, this work is the first attempt toward establishing an optimization framework that can enable efficient execution of DML tasks on a set of neighboring heterogeneous wireless edge nodes. Specifically, the report provides the following contributions:

- Inaugurates the MEL research by considering both distinct scenarios of DD (FL) and TP, as opposed to previous works that have only considered DD. This separates MEL from FL.
- Designs a dynamic task allocation mechanism, referred to as the heterogeneity aware (HA) scheme, which maximizes the learning accuracy in a heterogeneous environment while satisfying time constraints.
- Jointly considers the impacts of dataset size allocation and number of local updates, in contrast to previous works, which have considered only local updates.
- Derives a low-cost approach to solve the main optimization problem in the proposed HA scheme, which significantly reduces execution time.

- Presents a quantitative heterogeneity analysis and clearly highlights the gains achieved by the HA schemes with increasing heterogeneity and enabling context-aware cellular vehicle to everything (CV2X) communication.
- Demonstrates the superiority of the proposed HA approach in terms of convergence time reduction and final validation accuracy improvement.

As demonstrated by Wang et al. [7], the learning accuracy in edge environments can be maximized by maximizing the number of local updates in between every global update. To this end, the problem for both the TP and DD scenarios is first formulated as a quadratically-constrained integer linear program (QCILP), which is a non-deterministic polynomial-time (NP)-hard problem. The problem is then relaxed to its real-valued counterpart, which is still non-convex. A solution is then proposed for the original problem based on the analytical upper bounds derived from the Karush-Kuhn-Tucker (KKT) conditions. The merits of this proposed solution for the HA schemes are shown by comparing its performances against solutions from both the numerical optimizer and the heterogeneity-unaware (HU) equal task allocation approach [7], [16]. This comparison was done for TP and DD in two types of environments (indoor - 802.11 type and outdoor - cellular type) with two different datasets. The results demonstrated that the proposed HA schemes perform better in terms of achievable updates and learning accuracy than the HU approaches.

The remainder of this report is organized in the following way. Chapter 2 introduces preliminaries on DML models and their transition to MEL. The problem of interest in this report is formulated in Chapter 3, and our proposed solution for this problem is detailed in Chapter 4. Chapter 5 illustrates the testing results, and Chapter 6 concludes the report.

CHAPTER 2 SYSTEM MODEL FOR MOBILE EDGE LEARNING

2.1 Distributed Machine Learning Preliminaries

Distributed machine learning (DML) involves the operation of orchestrating one machine learning task over a system of K learners belonging to the set $\mathrm{K} = \{1, 2, \ldots, K\}$. As clarified in Chapter 1, this approach was motivated by one of two scenarios, namely distributed datasets (DD) and task parallelization (TP). An illustration of the learning cycles for these two scenarios is given in Figure 2-1. The latter scenario fully encompasses the former but also adds the batch transfer component from the orchestrator to the learners. Although we mainly present the system model, formulation, and solution for the latter scenario, we show the variations in all these components when the former scenario is considered.

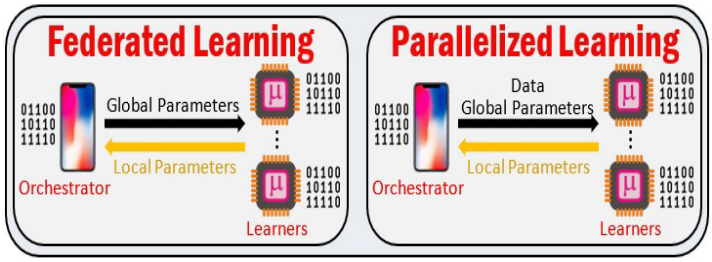


Figure 2-1. Illustration of the two distinct approaches of distributed datasets and task parallelization

Most DML approaches employ the stochastic gradient descent (SGD) method for updating the model because of large datasets and constraints on training time. This choice is also justified by the generality of this approach for centralized ML and its proven superior learning accuracy in comparison to deterministic gradient descent (GD) methods [23]. The ML model is updated by using a randomly picked subset or "batch" of the total dataset in one iteration of the SGD. Given the randomly picked batch of size d_k data samples allocated to learner k (l_k), $k \in \mathrm{K}$, this learner updates the local model such that the local loss is minimized on the assigned batch [7]. The total size of all batches is denoted by $d = \sum_{k=1}^{K} d_k$, which is usually preset by the orchestrator 0, given learner k^0s computational capabilities, the desired accuracy, and the time-constraints of the training process.

Each learner forwards the local model $\mathbf{w}_k \forall\, k \in \mathrm{K}$ to the orchestrator after performing a certain number of local updates. The size of learner k's model in bits will depend upon the ML model used, storage precision and compression strategy, and possibly the batch size. The orchestrator obtains the global model by aggregating all received local models. It then transmits

the global model to all learners, and the process cycles between local and global updates until the orchestrator ends the training either because of the exhaustion of resources or achievement of target performance. Readers interested in the local/global loss function minimization and local parameter aggregations are referred to Wang et al. [7], [17] for further details.

In this study, we considered the setting in which all K learners perform a synchronized equal number of local updates per global update [1]. This setting is in contrast to the asynchronous setting, in which the number of local cycles per global update cycle is different or global aggregation is not synchronized [15]. The asynchronous setting is clearly more general and will be the subject of future work. We denote this fixed number by τ. In this setting, the global update only occurs periodically, and we refer to each interval between global updates as the global update period. For time-critical learning tasks (which are of more interest in edge environments), this period is typically preset by the orchestrator based on the time constraint on either finishing the process or reaching a certain performance measure. The orchestrator performs the aggregation of the parameters only after all learners have sent back their result within that global update period. Consequently, the global update period includes the time required for the transmission of the global model and/or data batches to the learners, execution of local updates, receipt of local models, and the global update.

To this end, we define the time T as the duration needed to perform the first three steps of the process (i.e., excluding the global update process). We refer to the time T as the global cycle clock to differentiate it from the total global update period consisting of the global cycle clock and the global update time. The latter is omitted from our calculations, as it is not affected by the learners' channel and computational heterogeneity, unlike the times of the other three processes. Furthermore, efficient over-the-air aggregation approaches [24], [25] can allow for the exclusion of this time. We care about this time T, as it includes processes fully dependent upon the batch sizes and number of local updates; most importantly it captures the heterogeneity of the learners' computational capabilities and the underlying wireless communication links. Whereas these components may not be of significant influence when DML is executed over controlled wired and infrastructural servers, their high heterogeneity can tremendously impact the performance of DML when it is applied in wireless and mobile edge environments. This is where the MEL paradigm comes into play, as detailed in the next subchapter.

2.2 Transition to MEL

Let us introduce the MEL system model by transitioning the aforementioned DML model to a heterogeneous wireless environment. We define the parameters that relate the number of local updates and dataset size to the heterogeneous computing and communication capacities of wireless edge learners and their impact on the global update clock duration. Figure 2-2 illustrates the considered MEL setting. Note again that the description focuses on the more general TP scenario, but all variations for the DD scenario are clarified whenever needed.

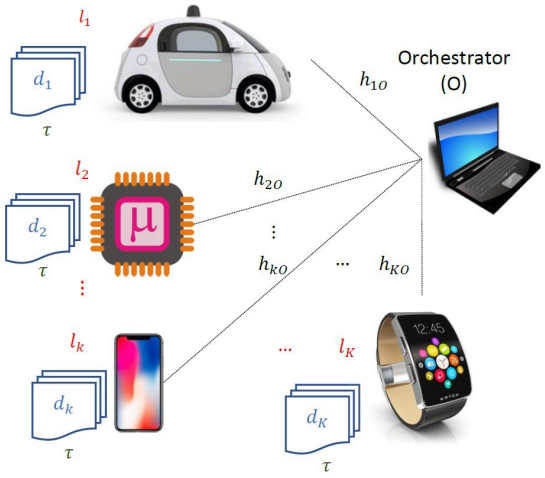


Figure 2-2 System model of a MEL setting

Let us define B_k^{data} as the batch size allocated to learner k in bits, which can be expressed as follows:

$$B_k^{data} = d_k \mathcal{F} \mathcal{P}_d \qquad (1)$$

where F is the number of features in the dataset, and P_d is the data bit precision. On the other hand, the size of learner k's local model in bits (denoted by B_k^{model}) is given by:

$$B_k^{model} = \mathcal{P}_m \left(d_k S_d + S_m \right) \qquad (2)$$

where P_m is the model bit precision. The variables P_d and P_m can represent a simple fixed-type, such as floating point, or may also include a compression ratio in case efficient storage strategies are used. As shown by Chiang and Zhang (2), the local parameter matrix size consists of two parts, one depending upon the batch size (represented by the term $d_k S_d$, where S_d is the number of

model coefficients related to each sample of the batch) and the other related to the constant size of the employed ML model (denoted by S_m).

The orchestrator sends the bit concatenation of the data batch and initial global model to learner k with transmission power P_{ko} using a wireless channel of bandwidth W and gain h_{ko}. (In the DD scenario, only the globally initialized model is sent.) Each learner randomly selects dk samples from its own stored dataset for each local update per global cycle. Once learner k receives this information, it re-sets the local model $\tilde{\mathbf{w}}_k$ to the global model $\mathbf{w}$ provided by the orchestrator. It then performs τ training/learning updates of the local model by using its allocated batch. Typically in ML, the algorithm sequentially goes over all features of each data sample once in one iteration (or epoch). Consequently, the number of computations required per iteration X_k is equal to:

$$X_k = d_k C_m \qquad (3)$$

which clearly depends upon the number of data samples d_k assigned to each learner and the computational complexity C_m of the model. Once finished, learners send their local model $\tilde{\mathbf{w}}_k \forall k \in \mathrm{K}$ to the orchestrator. The latter updates the global model $\mathbf{w}$ as described by Wang et al. [7] and sends it back, along with a new random batch of $d_k \forall k \in \mathrm{K}$ samples from the dataset to each learner. (In the DD setting, each learner selects a new random set of dk samples for the new cycle.) Then the process repeats.

Given the above description, the times of each learner k, $\forall k \in \mathrm{K}$, whose sum must be bounded by the global update clock T, can be detailed as follows. The first time $t^S_k \forall k \in \mathrm{K}$ comprises the time needed to send the allocated batch and global model $\mathbf{w}$ to learner k. t^S_k can thus be expressed as (where N_0 is the noise power spectral density)[5]:

$$t_k^S = \frac{d_k \mathcal{F} \mathcal{P}_d + \mathcal{P}_m \left(d_k \mathcal{S}_d + \mathcal{S}_m\right)}{W \log_2 \left(1 + \frac{P_{ko} h_{ko}}{N_0}\right)}, k \in \mathcal{K} \qquad (4)$$

The second time is τ times the duration $t_k^C \forall k \in \mathcal{K}$ needed by learner k to perform one local update. Defining f_k as learner k's local processor clock speed dedicated to the DML task, t^C_k can be expressed as:

$$t_k^C = \frac{X_k}{f_k} = \frac{d_k C_m}{f_k}, k \in \mathcal{K} \quad (5)$$

The third and final time t^R_k is required for learner k to send its updated local model $\tilde{\mathbf{w}}_k$ to the orchestrator. Assuming that the channels are reciprocal and do not change during the duration of one global update cycle, and assuming that learner k uses the same power P_{ko} on its assigned channel, t^R_k can be computed as:

$$t_k^R = \frac{\mathcal{P}_m \left(d_k \mathcal{S}_d + \mathcal{S}_m\right)}{W \log_2 \left(1 + \frac{P_{ko} h_{ko}}{N_0}\right)}, k \in \mathcal{K} \quad (6)$$

Thus, the total time t_k taken by learner k to complete the above three processes is equal to:

$$\begin{aligned} t_k &= t_k^S + \tau t_k^C + t_k^R \\ &= \frac{d_k \mathcal{F} \mathcal{P}_d + 2\mathcal{P}_m \left(d_k \mathcal{S}_d + \mathcal{S}_m\right)}{W \log_2 \left(1 + \frac{P_{ko} h_{ko}}{N_0}\right)} + \tau \frac{d_k \mathcal{C}_m}{f_k}, k \in \mathcal{K} \end{aligned} \quad (7)$$

As described above, $t_k \leq T$ must hold $\forall\, k \in \mathrm{K}$ for the orchestrator to successfully update the global model within the time constraint.

CHAPTER 3 PROBLEM FORMULATION

As mentioned in Chapter 1, the objective of the work presented in this report was to optimize the task allocation (i.e., distributed batch sizes d_k for each learner k and local model updates τ) such that it maximizes the accuracy of the DML process in each global cycle (and thus eventually the accuracy of the entire learning process) within a preset global cycle clock T. It has been well established in the literature that the loss function in the general SGD-based ML is minimized (and this typically maximizes the learning accuracy) by increasing the number of model updates [14]. When edge DML settings allow only a fixed number of global updates to be performed, increasing the number of local updates per global cycle can increase the accuracy [17]. The convergence proof for distributed SGD in the context of a wireless edge setting has already been provided in detail by Wang et al. [7], [17]. We used these results to show that our proposed MEL model will also converge.

Let the loss function of the global model be given by $F(\mathbf{w})$. Let us assume that this function is convex, ρ Lipschitz, and β-smooth (readers are referred to Wang et al. [17] and Mohammad et al. [26] for details on these assumptions and the parameters ρ and β). Assuming that a total of L updates are performed, the upper bound on the difference between the loss at update L and the loss of the optimal model $\mathbf{w}^*$ can be defined as:

$$F(\mathbf{w}[L] - F(\mathbf{w}^*) \leq \frac{1}{L\left[\eta\phi - \frac{\rho h(\tau)}{\tau\epsilon^2}\right]} \qquad (8)$$

The learning rate is given by η and $\varphi = 1 - \eta\beta_2$. The function $h(\tau) = \frac{\delta}{\beta}[(\eta\beta + 1)^\tau - 1] - \eta\delta\tau$ and ϵ is a parameter that bounds the local losses. For more details on δ and, the reader is referred to Wang et al. [17]. It is further assumed that $0 < \eta\beta < 1$, $\eta\phi - \frac{\rho h(\tau)}{\tau\epsilon^2} \geq 0$, and $(\eta\beta + 1)^\tau \geq \eta\beta\tau + 1$. For simplicity, let us assume that the same number of local updates τ are performed by each learner in every global cycle. Hence, we can define the total updates $L = G\tau$ for a fixed number of global updates G. Let us define $A = \eta\phi + \frac{\rho\delta}{\epsilon^2}$, $B = \frac{\rho\delta}{\beta\epsilon^2}$, and $C = \eta\beta + 1$. Using all of the above, we can define the upper bound on the loss as:

$$F(\mathbf{w}[L] - F(\mathbf{w}^*) \leq \frac{1}{G\tau\,[A + B(1 - C)]} \qquad (9)$$

Because $A + B(1 - C) \geq 0$ and the number of global updates G will be fixed, it is clear that the bound on the loss converges to zero as $\tau \to \infty$.

Figure 3-1 shows the progress of the global validation loss for two different datasets under various scenarios. In the figure, the loss converges for both datasets, but it is closer to zero for one of them. More details about the performance, as well as the simulation environment, datasets, and ML models used, are provided in Chapter 5.

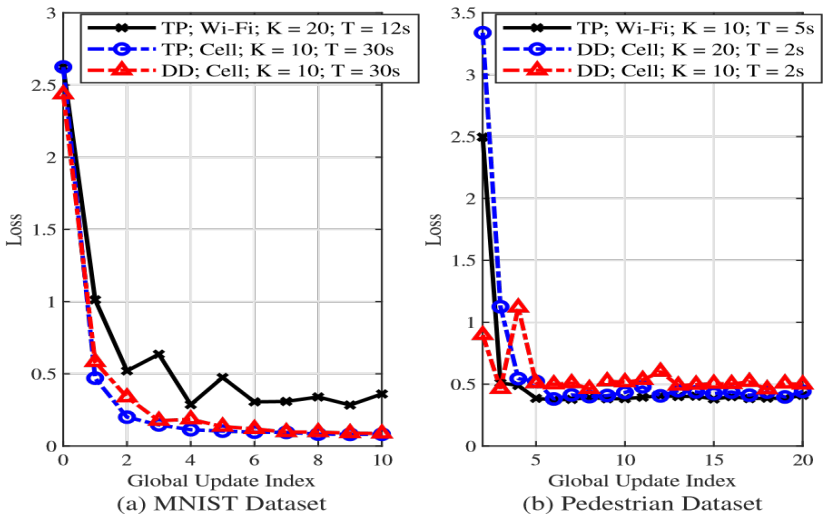


Figure 3-1 Validation loss after each global update for: (a) the MNIST dataset and (b) the Pedestrian dataset.

Given the above facts, maximizing the DML accuracy in our MEL setting can thus be re-worded as optimizing the assigned batch sizes d_k to each of the learners so as to maximize the number of local updates τ per global cycle, while bounding $t_k \forall\, k$ by the preset global cycle clock T. The optimization variables in this problem are thus τ and $d_k \forall\, k \in \mathrm{K}$. To simplify the notation in the remainder of this report, we can re-write the expression of t_k in Equation (7) as a function of the optimization variables as follows:

$$t_k = C_k^2 \tau d_k + C_k^1 d_k + C_k^0,\ k \in \mathcal{K} \qquad (10)$$

The variables C_k^2, C_k^1, and C_k^1 respectively represent the quadratic, linear, and constant coefficients of learner k in terms of the optimization variables τ and d_k expressed as:

$$C_k^2 = \frac{\mathcal{C}_m}{f_k}, k \in \mathcal{K} \tag{11}$$

$$C_k^1 = \frac{\mathcal{F}\mathcal{P}_d + 2\mathcal{P}_m\mathcal{S}_d}{W \log_2\left(1 + \frac{P_{ko}h_{ko}}{N_0}\right)}, k \in \mathcal{K} \tag{12}$$

$$C_k^0 = \frac{2\mathcal{P}_m\mathcal{S}_m}{W \log_2\left(1 + \frac{P_{ko}h_{ko}}{N_0}\right)}, k \in \mathcal{K} \tag{13}$$

Given the above information, the bound $t_k \leq T \; \forall \; k \in \mathrm{K}$ can be written as follows:

$$C_k^2 \tau d_k + C_k^1 d_k + C_k^0 \leq T, k \in \mathcal{K} \tag{14}$$

Recall that τ is the number of local model updates done by each learner in every global cycle, whereas T is the duration in which all steps of one global cycle must be completed, a.k.a global cycle time. Clearly the relationship between t_k and the optimization variables d_k and τ is quadratic. Furthermore, the optimization variables τ and $d_k \; \forall \; k \in \mathrm{K}$ are all non-negative integers. Consequently, the problem of interest in this report can be formulated as a quadratically constrained integer linear program (QCILP) as follows:[8]

$$\max_{\tau, d_k \; \forall \; k \in \mathcal{K}} \quad \tau \tag{15a}$$

$$\text{s.t.} \quad C_k^2 \tau d_k + C_k^1 d_k + C_k^0 \leq T, \quad k \in \mathcal{K} \tag{15b}$$

$$\sum_{k=1}^{K} d_k = d \tag{15c}$$

$$\tau \in \mathcal{Z}_+ \tag{15d}$$

$$d_k \in \mathcal{Z}_+, \quad k \in \mathcal{K} \tag{15e}$$

Constraint (15b) guarantees that $t_k \leq T \; \forall \; k \in \mathrm{K}$. Constraint (15c) ensures that the sum of local dataset sizes assigned to each learner is equal to the size of the total dataset required by the orchestrator to complete the learning task with the desired performance, given the time limits and its own computational and communication capabilities. Constraints (15d) and (15e) are simply non-negative integer constraints on the optimization variables. Therefore, the above problem is a QCILP, which is well-known to be non-deterministic polynomial-time (NP)-hard [27].

CHAPTER 4 PROPOSED SOLUTION

4.1 Problem Relaxation

As clarified in the previous chapter, the considered problem is a QCILP, which in general is NP-hard [27]. These problems can be solved in polynomial time by using certain heuristic techniques such as the branch-and-bound methods employed by commercially available solvers. However, these techniques are still highly complex and require a long execution time, which may not be suitable for edge environments. Therefore, we propose to simplify the problem by relaxing the integer constraints in equations (15d) and (15e), solving the relaxed problem, and finally rounding the obtained real results back into integers. The relaxed problem is given by:

$$\max_{\tau, d_k \ \forall \ k \in \mathcal{K}} \quad \tau \tag{16a}$$

$$\text{s.t.} \quad C_k^2 \tau d_k + C_k^1 d_k + C_k^0 \leq T, \quad k \in \mathcal{K} \tag{16b}$$

$$\sum_{k=1}^{K} d_k = d \tag{16c}$$

$$\tau \geq 0 \tag{16d}$$

$$d_k \geq 0, \quad k \in \mathcal{K} \tag{16e}$$

where the cases for τ and/or all d_k's being zero represent scenarios in which MEL will not be feasible, and thus the orchestrator must send the learning tasks to the edge or cloud server. The above resulting program becomes a quadratically constrained linear program (QCLP), which can be solved with techniques such as interior-point methods. There are efficient solvers that implement these approaches, such as OPTI [28].

On the other hand, the associated matrices for each of the quadratic constraints in the relaxed problems can be written in a symmetric form. Because these matrices will have two non-zero values that are positive and equal, the associated eigenvalues will sum to zero. This means that these matrices are not positive semi-definite, and hence, the relaxed problem is non-convex. Consequently, we cannot derive the optimal solution of this problem analytically. Yet we can still derive upper bounds on the optimal variables and solution by using Lagrangian relaxation and the Karush-Kuhn-Tucker (KKT) conditions, which may further reduce the complexity in comparison to heuristic solvers without loss of performance. The next subchapter describes derivation of the upper bounds of the optimal variables by applying Lagrangian analysis and KKT conditions on the relaxed non-convex problem.

4.2 Upper Bounds Using the KKT Conditions

The Langrangian function of the relaxed problem is expressed as:

$$L(\mathbf{x}, \lambda, \nu, \alpha) = -\tau + \sum_{k=1}^{K} \lambda_k \left(C_k^2 \tau d_k + C_k^1 d_k + C_k^0 - T\right) + \nu \left(\sum_{k=1}^{K} d_k - d\right) - \alpha_0 \tau - \sum_{k=1}^{K} \alpha_k d_k \quad (17)$$

where $\lambda_k \forall\, k \in \mathrm{K}$, ν, and $\alpha_0/\alpha_k \forall\, k \in \mathrm{K}$, are the Lagrangian multipliers associated with the time constraints of the K learners in equation (16b), the total batch size constraint in (16c), and the non-negative constraints of all the optimization variables in (16d) and (16e), respectively. Using the well-known KKT conditions, the following theorem introduces upper bounds on the optimal variables of the relaxed problem.

Theorem 1. The optimal value of the batch size $d_k^* \forall$ allocated to each learner $k \in \mathrm{K}$ satisfies the following bound:

$$d_k^* \leq \frac{T - C_k^0}{\tau^* C_k^2 + C_k^1} \qquad \forall\, k \in \mathcal{K} \qquad (18)$$

Moreover, the analytical upper bound on the optimization variable τ belongs to the solution set of the polynomial given by:

$$d \prod_{k=1}^{K} (\tau^* + b_k) - \sum_{k=1}^{K} a_k \prod_{\substack{l=1 \\ l \neq k}}^{K} (\tau^* + b_l) = 0 \qquad (19)$$

$$\text{where } r_k^0 = C_k^0 - T,\ a_k = -\frac{r_k^0}{C_k^2} \text{ and } b_k = \frac{C_k^1}{C_k^2},\ \forall\, k \in \mathcal{K}.$$

Proof: From the KKT optimality conditions, we have the following relations given by the following equations:

$$C_k^2\tau^* d_k^* + C_k^1 d_k^* + C_k^0 - T \leq 0, \quad k \in \mathcal{K} \tag{20}$$

$$\alpha_0^*, \alpha_k^*, \text{ and } \lambda_k^* \geq 0 \quad k \in \mathcal{K} \tag{21}$$

$$\lambda_k^* \left(C_k^2\tau^* d_k^* + C_k^1 d_k^* + C_k^0 - T\right) = 0, \quad k \in \mathcal{K} \tag{22}$$

$$-\alpha_0^* \tau^* = 0 \tag{23}$$

$$-\alpha_k^* d_k^* = 0 \quad k \in \mathcal{K} \tag{24}$$

$$\sum_{k=1}^{K} d_k^* - d = 0 \tag{25}$$

$$-\nabla\tau^* + \sum_{k=1}^{K} \lambda_k^* \nabla \left(C_k^2\tau^* d_k^* + C_k^1 d_k^* + C_k^0 - T\right) + \nu^* \nabla \left(\sum_{k=1}^{K} d_k^* - d\right) - \alpha_0^* \nabla\tau^* - \nabla \left(\sum_{k=1}^{K} \alpha_k^* d_k^*\right) = 0 \tag{26}$$

From the conditions in equation (20), we can see that the batch size at any learner k must satisfy equation (18). Moreover, it can be inferred from (22) that the bound in (18) holds with equality for any learner $k \in \mathcal{K}$ if $\lambda_k^* \geq 0$. The upper bound will be equal to the optimal solution (i.e. strong duality will hold) for some feasible τ^* when, strictly speaking, $\lambda^*_k > 0 \; \forall \; k \in \mathrm{K}$ because in that case, the second term in (22) must be equal to zero. By re-writing the bound on d_k^* in (18) as an equality and substituting it back into (25), we have the following relation:

$$d = \sum_{k=1}^{K} d_k^* = \sum_{k=1}^{K} \left[\frac{T - C_k^0}{\tau^* C_k^2 + C_k^1}\right] = \sum_{k=1}^{K} \left[\frac{a_k}{\tau^* + b_k}\right] \tag{27}$$

The expression on the right-most hand-side has the form of a partial fraction expansion of a rational polynomial function of τ^* where $a_k, b_k \in \mathrm{R}^{++}$. Therefore, we can expand equation (27) to the form shown in (28). Finally, the expanded form can be cleaned up in the form of a rational function with respect to τ^*, which is equal to the total dataset size d as shown in equation (29).

Please note that the degrees of the numerator and denominator will be $K - 1$ and K, respectively. Furthermore, the poles of the system will be $-b_k$, and since $b_k \geq 0$, the system will be stable. Furthermore, $\tau^* = -b_k$ is not a feasible solution for the problem because it is eliminated by the $\tau \geq 0$ constraint. Therefore, we can rewrite equation (29) as shown in (19). By solving this polynomial, we obtain a set of solutions for τ^*, and one of them is feasible. The problem being non-convex, this feasible solution τ^* will constitute the upper bound to the solution of the relaxed problem.

$$\frac{a_1}{\tau^* + b_1} + \frac{a_2}{\tau^* + b_2} + \cdots + \frac{a_k}{\tau^* + b_k} + \cdots + \frac{a_K}{\tau^* + b_K} = \frac{1}{(\tau^* + b_1)(\tau^* + b_2)\ldots(\tau^* + b_k)\ldots(\tau^* + b_K)} \times \Big[a_1(\tau^* + b_2)(\tau^* + b_3)\ldots(\tau^* + b_k)\ldots(\tau^* + b_K) + a_2(\tau^* + b_1)(\tau^* + b_3)\ldots(\tau^* + b_k)\ldots(\tau^* + b_K) + \ldots + a_k(\tau^* + b_1)(\tau^* + b_2)\ldots(\tau^* + b_{k-1})(\tau^* + b_{k+1})\ldots(\tau^* + b_K) + \cdots + a_K(\tau^* + b_1)(\tau^* + b_2)\ldots(\tau^* + b_k)\ldots(\tau^* + b_{K-1}) \Big] \tag{28}$$

$$d = \frac{\sum_{k=1}^{K} a_k \prod_{\substack{l=1 \\ l \neq k}}^{K} (\tau^* + b_l)}{\prod_{k=1}^{K} (\tau^* + b_k)} \tag{29}$$

As a last step, to ensure that the solution set is feasible, it must be noted that according to equations (23) and (24), α_0^* and $\alpha_{k^*} \forall\, k$ must be equal to 0. Expanding the vanishing gradient condition in (26), it can be shown that the following two relations can be derived (representing K + 1 equations):

$$\lambda_k^* C_k^2 \tau^* + \lambda_k^* C_k^1 + \nu^* = \alpha_k^*,\ k \in \mathcal{K} \tag{30}$$

$$-1 + \sum_{k=1}^{K} \lambda_k^* C_k^2 d_k^* = \alpha_0^* \tag{31}$$

By setting $\alpha_0^* = 0$ and $\alpha_{k^*} = 0$ for $k \in \mathrm{K}$, we can write λ^*_k in terms of ν^*, as shown in equation (32), and substitute the resulting expression in (31) to find ν^* by using the values of d_k^* and τ^* obtained from equations (18) and (19), respectively.

$$\lambda_k^* = -\frac{\nu^*}{C_k^2 \tau^* + C_k^1},\ k \in \mathcal{K} \tag{32}$$

$$\nu^* = -\frac{1}{\sum_{k=1}^{K} \frac{C_k^2 d_k^*}{C_k^2 \tau^* + C_k^1}} \tag{33}$$

The values of λ^*_k for $k \in \mathrm{K}$ can be obtained by back-substituting ν^* in equation (32). As one can observe, as long as there exists a τ^* greater than zero, ν^* will be negative, and hence, λ_k^* for $k \in \mathrm{K}$ will be strictly greater than zero. Hence, as long as there exists a $\tau^* > 0$ in the feasible set such that $d^*_k > 0$, there will exist a set of $\lambda^*_k > 0$ for $k \in \mathrm{K}$. This fact can be used to verify the feasibility of

the solution. This step is also helpful when multiple values may exist of τ greater than zero for choosing the optimal τ^*. Although there was a possibility that the above bounds on $d^*_k \forall\, k$ in equation (18) and solution the for τ^* from (19) would require improved steps to reach feasibility, the extensive simulations presented in Chapter 5 demonstrated a lack of optimality gap, and the optimal variables were obtained directly from the derived expressions.

CHAPTER 5 SIMULATION RESULTS

In this chapter, we describe tests of our proposed dynamic task allocation solutions for MEL in realistic learning and edge settings. More specifically, for both the TP and DD schemes, the numerical OPTI-based solution to the relaxed QCLP formulation (HA-TP/DD-Num) is compared to the analytical results (HA-TP/DD-Ana) from equations (18) and (19). We also show the merits of these two solutions in comparison to those of the heterogeneity-unaware (HU) (HU-TP/DD) equal task allocation (ETA) scheme employed by Wang et al. [7] and Tuor et al. [16] by comparing the number of local updates and the validation accuracy. We first introduce the simulation environment and then present the validation results. The validation accuracy results presented for each individual scenario are an average of 10 runs.

5.1 Simulation Environment

A typical MEC consists of a cloudlet of heterogeneous computing devices connected via wireless channels. In our simulation, the edge nodes or learners were assumed to be located in an area with a 50-m radius for an 802.11 type environment and a 500-m radius for a cellular type environment. Half of the considered learners emulated the capacity of a typical fixed/portable computing device (e.g., laptops, tablets, roadside units. etc.) and the other half emulated the capacity of commercial micro-controllers (e.g., Raspberry Pi) that can be attached to different indoor or outdoor systems (e.g., smart meters, traffic cameras, onboard units). The settings thus emulated an edge environment that could be located either indoors or outdoors. The employed channel model for both the 802.11 and cellular type links is summarized in table 5-1.

Table 5-1 List of simulation parameters

Parameter	**Value**
Wi-Fi Attenuation Model	$7 + 2.1\log(R)$ dB [29]
Cell Attenuation Model	$128 + 37.1\log(R)$ dB [5]
Learner Bandwidth (W)	5 MHz
Device proximity indoor (R_0)	50m
Device proximity outdoor (R_0)	500m
Max. Tran. Power (P_k^{max})	23 dBm
Noise Power Density (N_0)	-174 dBm/Hz
Computation Capability (f_k)	2.4 GHz and 700 MHz
Pedestrian Dataset size (d)	9,000 images
Pedestrian Dataset Features (F)	648 (18 × 36) pixels
MNIST Dataset size (d)	60,000 images
MNIST Dataset Features (F)	784 28 × 28) pixels

The main variable representing the computational capabilities of the learners is the processor clock speed $f_k \forall\, k \in \mathrm{K}$. Although several factors can influence the computational capability, including number of cores, available memory, interconnect speeds, etc., we considered clock speed because it can be related directly to time and energy consumption [20]. In addition, clock speed is still used in state-of-the-art works to model the computational capability for MEC in general [30], [31] and federated learning (FL) environments in particular [20], [32]. The communication capability can be measured by the achievable rate, $R_k \forall\, k \in \mathrm{K}$ which is defined as:

$$\rho_k = W \log_2 \left(1 + \frac{P_{ko} h_{ko}}{N_0}\right), \quad k \in \mathcal{K} \qquad (34)$$

The rate is influenced by the bandwidth W, each learner's transmission power $P_{ko} \forall\, k$, and the channel gain $h_{ko} \forall\, k$, which in turn is influenced mainly by the distance of the learner from the orchestrator $R_k \forall\, k$ because it is the inverse of the path loss defined in table 5-1.

Once again, using the rate defined in equation (34) to model the channel was supported by the latest works in edge FL [19], [20], [21], [22]. The selection of the path-loss model was supported by previous works on MEC [30], [31], and the Wi-Fi model was adopted from previous empirical results [29]. One simplification in our model was that perfect parallel communication without interference was assumed, in contrast to some works on resource allocation for edge FL [19]. We justified this by stating that all schemes would be affected equally by channel degradation due to interference because resource allocation was not the focus of this work. The selection strategy for the remaining parameters was also supported by the above referenced works on MEC and wireless edge FL.

It is assumed that at the start of each global cycle, learners send the information related to clock speed and transmission power that they can dedicate for the next cycle. This information is not incorporated into the communication time because the size of this packet in bits is negligible in comparison to the local ML model. As for the channel gain and learner distances, it is assumed that the edge servers can make use of the existing efficient channel estimation and triangulation algorithms. This task can be completed while the learners are doing their local updates just before the local model exchange.

For simulating the learning environment, two datasets were considered: namely the Pedestrian [33] and MNIST [34] datasets. The Pedestrian dataset has 8,000 training images

consisting of 684 features (18 x 36 pixels). The ML model used for this dataset is a single-layer neural network with 300 neurons in the hidden layer. The set of parameters $\mathbf{w} = [w_1, w_2]$ is the concatenation of two sub-matrices, in which w_1 is 300×648 and w_2 is 300×2, neither of which depends upon the batch size ($S_d = 0$). Thus, the size of the model is 6,240,000 bits, which is fixed for all learners. The forward and backward passes require 781,208 floating point operations [35].

Tthe MNIST dataset consists of 60,000 28x28 pixel images (784 features). The employed ML model for these data is a three-layer neural network with the following configuration: [784,300,124,60,10]. The model parameters include weight matrices with sizes of 784×300, 784×300, 784×300, 300×124, 124×60, and 60×10, and bias vectors with lengths of 300, 124, 60, and 10, respectively. The model is 8,974,080 bits, and it requires 67,424,160 floating point operations [35].

5.2 Heterogeneity Analysis

The previous chapters explained that the proposed MEL approach is heterogeneity aware (HA) because it allocates the number of local updates and batch sizes while considering the heterogeneous communication and computation environments. The most impactful variables on the rate heterogeneity are the transmission power and the distance because the bandwidth is constant (resource allocation has been studied in other works [19], [21], [22], [36]). Furthermore, other variables in the path loss model shown in table 5-1 are either constant or do not contribute as significantly. We assume that the maximum transmission power limit P_k^{max} is 23 dBm or 0.1995W in linear scale. We further assume that each learner-orchestrator pair uses a lower power level per global cycle instead of the maximum allowed. To simulate a heterogeneous environment, the transmission power is drawn from the following distribution:

$$P_k \sim P_k^{max} - P_\sigma U, \quad k \in \mathcal{K} \qquad (35)$$

The variable P_σ represents the maximum value by which the power can be reduced, and $U \sim \mathrm{U}\{0,1\}$ represents the uniformly distributed random variable on $[0,1]$. The focus of the proposed HA scheme is on optimal execution of the MEL task and not the optimization of the networking parameters. Hence, we used this approach of random power allocation. Other works have explicitly studied power and resource allocation for MEL. To simulate the heterogeneous co-location of learners, the distances are simply drawn from $R_k \sim R_\sigma U$. R_σ limits the learners to within

a smaller radius than the defined radius R_0 for different environments, such as 50 m for indoors. In general, $0 \leq R_\sigma \leq R_0$.

A similar strategy is used to set the computational capability of each device. Modern multicore processors usually offer a speedup over the advertised clock speeds, and new micro-controllers have clock speeds of more than 1GHz as opposed to older versions that were in the range of a few hundred MHz. Therefore, the strategy used to model the processor clock involves choosing a reference speed followed by an additional amount to represent some speedup, as shown below:

$$f_k \sim f_{ref} + f_\sigma U, \quad k \in \mathcal{K} \tag{36}$$

The reference clock is given by f_{ref}, whereas f_σ represents the maximum additional clock speed. In the subsequent simulations, two sets of reference speeds are used: $f_{ref,1} = 2400$MHz and $f_{ref,2} = 700$MHz. The associated additional maximum amounts are $f_{\sigma,1}$ and $f_{\sigma,2}$, respectively.

It is very difficult to exactly quantify the heterogeneity of the environment because of the complex relationships among all variables and the random nature of the important parameters such as the rate ρ_k, which are further dependent upon multiple random variables. In general, all variables are drawn from the uniform distribution in which the σ sub-scripted quantities represent the possible magnitude of change. To quantify the expected level of heterogeneity, we define a new metric called the heterogeneity factor H_{fac} as follows:

$$\mathcal{H}_{fac} = \frac{P_\sigma}{P_k^{max}} + \frac{R_\sigma}{R_0} + \frac{f_{\sigma,1}}{f_{ref,1}} + \frac{f_{\sigma,2}}{f_{ref,2}} \tag{37}$$

H_{fac} is defined as the sum of the ratios of the 'σ' variables to the reference values. A ratio must be defined because all parameters have different physical units, but their effect must be captured as each contributes to the heterogeneity level. The heterogeneity factor is not calibrated to have a value in the range [0,1], and a higher H_{fac} simply implies a more heterogeneous environment, which is expected to tolerate the seamless transitions between devices (i.e., context awareness).

5.3 Achievable Updates versus Heterogeneity

To perform the heterogeneity analysis, we vary P_σ from 0.01 to 0.10 in steps of 0.01 and R_σ from 5 m to 50 m in steps of 5 m. The reference radius R_0 is set to 50 m because we are modeling an indoor, 802.11 type environment. The clock speed heterogeneity is emulated by

varying $f_{\sigma,1}$ from 240 MHz to 2400 MHz in steps of 200 MHz and $f_{\sigma,1}$ from 70 MHz to 700 MHz in steps of 70 MHz. The reference values and channel parameters are defined in table 5-1.

We plot the achievable number of local updates τ versus the heterogeneity factor H_{fac} and compare the performance of the HA and HU schemes. The HA-TP and HA-DD both offer a gain on the number of achievable local updates per global cycle as H_{fac} increases. This gain can be observed specifically for values of $\mathrm{H}_{fac} \geq 1$. On the other hand, the achievable τ remains constant for the HU schemes with increasing H_{fac} implying that these are truly HU. Our HA schemes can perform flexible local dataset size allocations per global cycle to maximize local updates. In the following subchapters, we show that for fixed numbers of global cycles, this also translates to higher final validation accuracies and lower convergence times.

For the remaining simulations, R_σ is equal to R_0 (i.e., 50 m for 802.11 type and 500 m for cellular type environments), whereas P_σ is set to 0.05 W. The processor speedups are limited by setting $f_{\sigma,1} = 240$ MHz and $f_{\sigma,2} = 100$ MHz. These parameters are selected to have a $\mathrm{H}_{fac} = 1.50$, which represents a high heterogeneity level in which the HA schemes provide noticeable gains, and enabling seamless context-aware communications is key. Moreover, this selection strategy truly tests the capacity of the HA schemes to offer a higher performance in comparison to the HU schemes because it supports lower speedups as well as lower transmission powers and channel gains.

5.4 Simulation Results for Task-Parallelization

Figure 5-1 shows the results from training with the MNIST dataset with the aforementioned deep neural network model. Figures 5-2a and 5-2b depict the number of local updates τ achieved by all tested approaches versus the number of learners (for $T = 30$s and $T = 60$s) and against the global cycle time (for $K = 10$ and $K = 20$), respectively. We can first make the trivial observation that τ increases with the number of learners K as the local dataset size sent/allocated to each node is smaller. We can also see that the performance of the OPTI-based numerical (HA-TP-Num) and UB-Analytical (HA-TPAna) solutions are identical for all simulated numbers of learners K and global cycle times T. We can finally observe from both sub-figures that the dynamic heterogeneity aware (HA) task allocation scheme for MEL achieves a significantly larger number of local updates than the heterogeneity unaware (HU) scheme. For instance, the HA scheme makes it possible to perform six updates (as opposed to one) for 20 learners, each with a global cycle time of 30s, a gain of 600 percent. In general, gains in the

region of 200 percent to 600 percent can be achieved by the HA-TP in comparison to the HU-TP. Another interesting result is that the performance of the HU scheme for $T = 60$s or $K = 20$ is actually much lower than the performance of our proposed solutions for $T = 30$s or $K = 10$, respectively. This implies that the HA schemes may achieve a higher learning accuracy than the HU scheme in half the time or with half the number of learners.

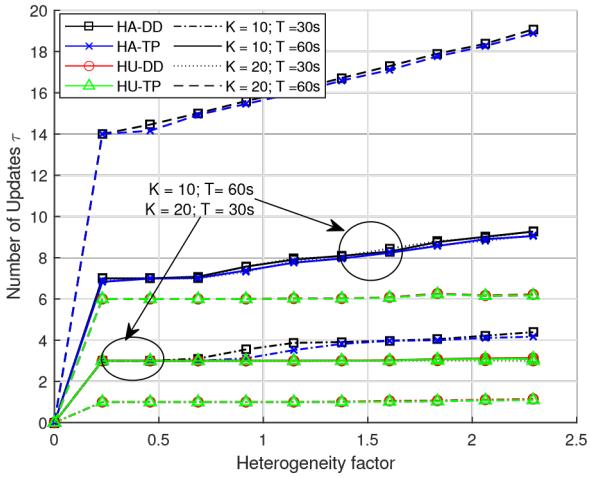


Figure 5-1 Achievable local updates τ for the MNIST dataset versus the heterogeneity factor for different sets of learners and global cycle times.

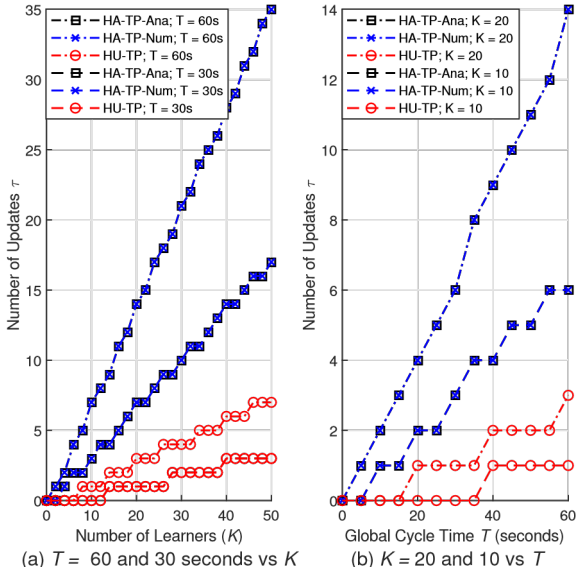


Figure 5-2 Achievable local updates τ for the MNIST dataset. (a) Performance comparison of all schemes for T =30 and 60 seconds vs K (b) Performance comparison of all schemes for K =10 and 20 vs T

Figure 5-3 illustrates the achievable number of local updates for the Pedestrian dataset. Figure 5-3a shows the number of local updates τ achieved by all tested approaches versus the number of learners for $T = 15$ and $T = 5$ seconds. Similar to the results of the MNIST dataset, the numerical solution (HA-TP-Num) matches the analytical solution (HA-TP-Ana) for all simulated scenarios. For both $T =$ 5s and 15s, the HA schemes provide a significantly higher number of local updates. Another interesting result is that the performance of the HU-TP scheme for $T = 15$ seconds is actually worse than that of the HA scheme with $T = 5s$, which implies that a higher accuracy can be achieved in half the time. Figure 5-3b illustrates the number of local updates τ versus the global cycle time T, for $K = 10$ and 20 learners. A system with $K = 20$ and $T = 10$s can give a 500 percent gain with the HA scheme. Further, the HA schemes can perform more local updates at lower T, even with fewer learners K than the HU scheme.

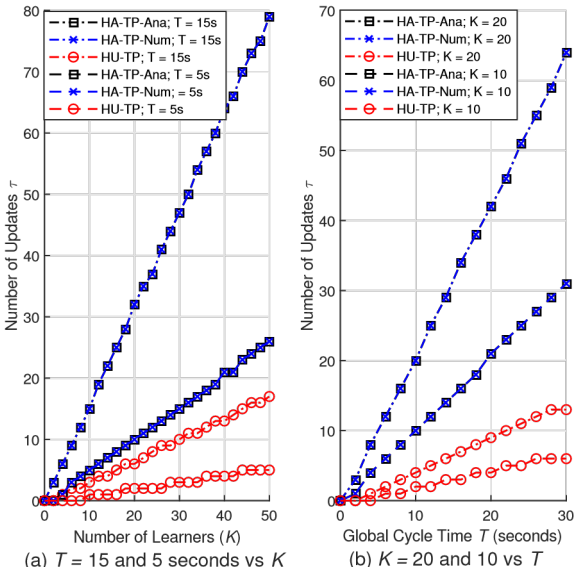


Figure 5-3 Achievable local updates τ for the Pedestrian dataset. (a) Performance comparison of all schemes for $T = 15$ and 5 seconds vs K (b) Performance comparison of all schemes for $K =$10 and 20 vs T

5.5 Learning Accuracy

Figures 5-4 and 5-5 depict the progression of learning accuracy achieved by both our dynamic HA scheme and the HU scheme for the MNIST and Pedestrian datasets, respectively. Results showed that the HA scheme clearly outperforms the HU method by reaching a higher final accuracy of up to 0.6 percent for the MNIST dataset and 8 percent for the Pedestrian dataset. Furthermore, the HA-TP can reduce the convergence times up to 56 percent to reach

certain accuracy thresholds for the MNIST dataset and up to 50 percent for the Pedestrian dataset in comparison to the HU-TP. For example, for the MNIST dataset with $K = 10$ and $T = 30$s, an accuracy of 96.50 percent can be achieved in five global cycles by the HA-TP as opposed to the nine updates required by the HU-TP, which represents time savings of 2 minutes or 56 percent. This clearly exhibits the merits of the dynamic scheme in attaining learning goals in far less time.

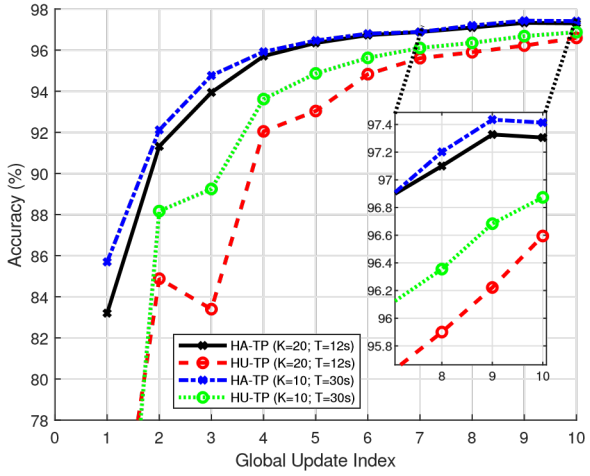


Figure 5-4 Validation accuracy progression for the MNIST dataset up to 10 global cycles for (a) K =20 and T =12s, (b) K =10 and T =30s

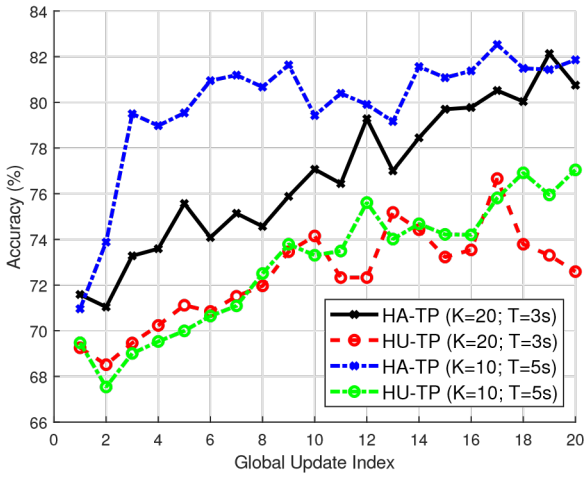


Figure 5-5 Validation accuracy progression for the Pedestrian dataset up to 20 global cycles for (a) K =20 and T =5s, (b) K =20 and T =3s

5.6 Simulation Results for Distributed Datasets

The next phase of simulations emulated the outdoor setting with cellular type links and examined each of the four different possible scenarios: DD with the proposed HA scheme (DD-HA), DD with the HU scheme (DD-HU), TP with the proposed HA scheme (HA-TP), and TP with the HU (HU-TP) method. For the proposed HA schemes, it is observable that the analytical solutions (HA-DD-Ana and HA-TP-Ana) match the OPTI-based numerical solutions (HA-TP-Num and HA-DD-Num). Although the TP and DD approaches are different and apply to different applications, we directly compare results for HA-DD and HA-TP to gain insights into our MEL model performance for the two distinct yet practical approaches. The parameters for the outdoor environment are also provided in table 5-1.

Figure 5-6 and figure 5-7 illustrate the achievable number of local updates for all schemes for the MNIST and Pedestrian datasets, respectively; figures 5-6a and 5-7a plot the number of updates against the number of learners K, whereas figures 5-6b and 5-7b plot τ versus the global cycle time. In general, the DD approach achieves the highest number of local updates τ because it requires only the exchange of model parameters without any data transmission. As larger global update cycles are allowed, which may not always be the case with MEL, the performances of HA-DD and HA-TP become similar. The gain of HA-DD is more noticeable for the Pedestrian dataset because a less complex model is trained on a smaller dataset, as opposed to the MNIST dataset with the larger model. It can be inferred that the DD approach provides a higher gain over the TP approach when the size of the model is comparable to the assigned local dataset sizes, as will be the case when smaller models are used. In all cases, the results of HA-DD/TP-Ana match the numerical solution, and HA-DD/TP-Num and HA-DD/TP both provide a higher set of local updates τ than HU-DD/TP, with gains ranging from 100 percent to 600 percent.

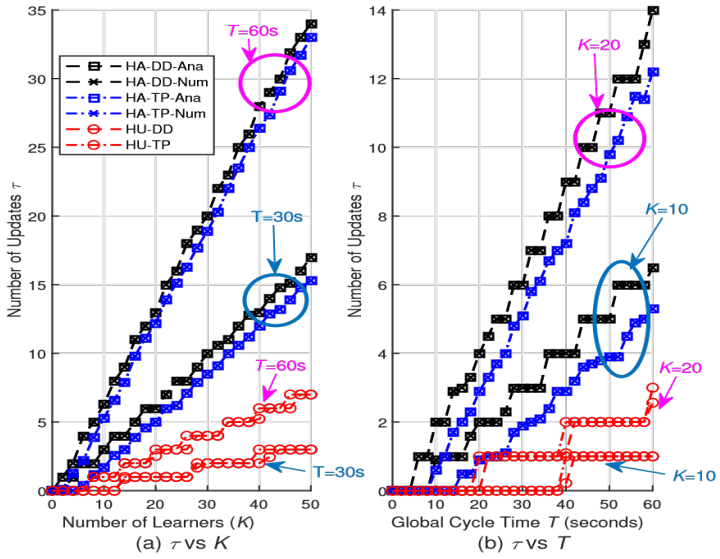


Figure 5-6 Achievable local updates τ for the MNIST dataset (a) versus K for T =30s and T =60s, and (b) versus T for K = 10 and 20.

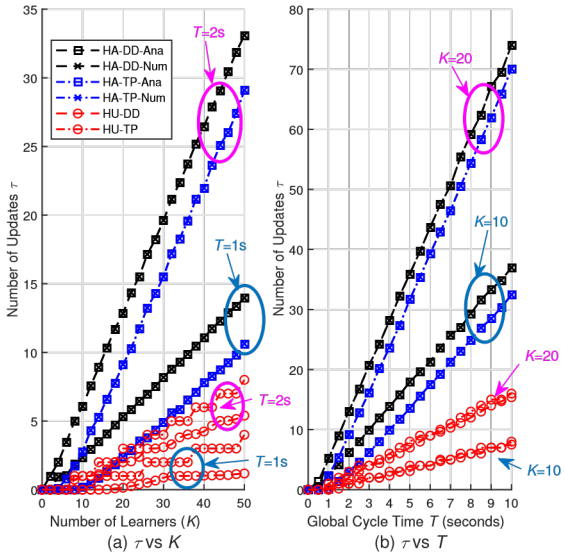


Figure 5-7 Achievable local updates τ for the Pedestrian dataset versus K for (a) T =1s and T =2s and (b) versus T for K = 10 and 20.

5.7 Learning Accuracy

This section presents learning accuracy performance comparisons for all schemes. Figure 5-8 compares the validation accuracy of HA-DD/TP and HU-DD/TP for the MNIST dataset when the global cycle time is set to 60 s and the system consists of ten learners. In general, the dynamic HA schemes are far superior to the HU approach, providing a final validation accuracy

of more than 5 percent. For example, the HA schemes reach an accuracy of 96 percent in four global cycles whereas the HU approaches require seven cycles, a reduction in time of about 43 percent. We also notice that TP seems to work slightly better than the DD approach despite the possibility of more local updates (larger τ) in the latter scenario. One possible explanation is that the former approach resembles SGD more accurately because each learner uses a different random subset of data after every global update. In contrast, learners use all or part of the same subset of data for local updates in each global cycle in the DD scenario.

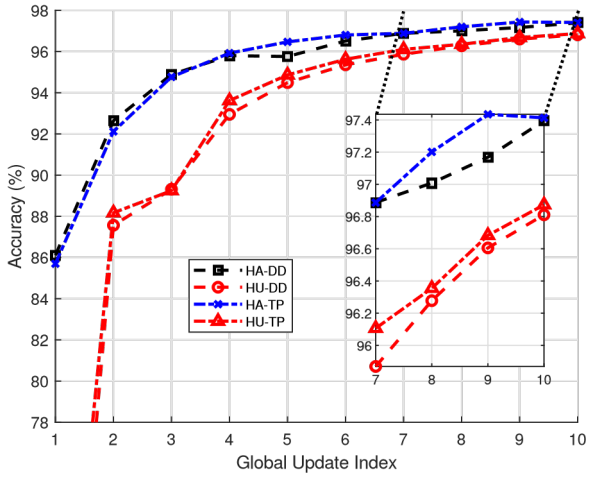


Figure 5-8 MNIST validation accuracy results with $K = 10$ and $T = 60$s for DD and TP using both the dynamic HA and the HU schemes.

Figure 5-9 compares the learning accuracy of all schemes for the Pedestrian dataset when the global cycle time is set to 5 s and the system consists of ten learners. Once again, the dynamic HA schemes are far superior to the HU approach and provide a higher final validation accuracy of up to 8 percent after 20 global updates. Furthermore, the HA schemes require less time to converge to certain accuracy thresholds. For example, the HA schemes can reach an accuracy of 76 percent in six global updates, which reduces the convergence time by 40 percent. We also notice that the DD approach reaches many accuracy milestones faster than TP, such as requiring 50 percent less time to reach 77 percent accuracy. One possible interpretation is that the difference in the average number of local updates per learner (more than five) makes up for the deterministic nature of the DD scenario. On the other hand, the TP approach seems to converge more smoothly toward the latter global update cycles, which may be due to the more "stochastic" nature of the data distribution process.

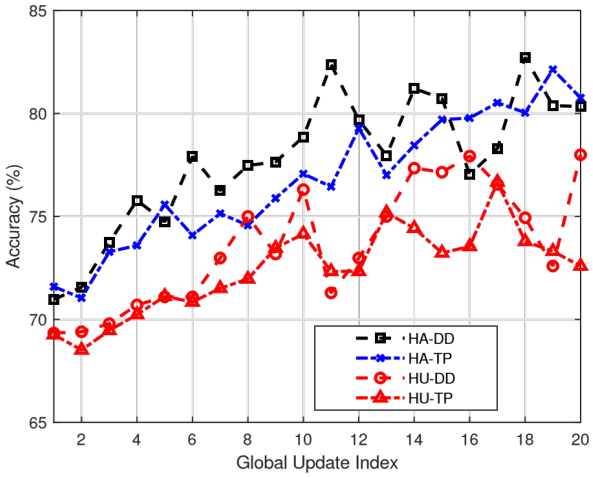
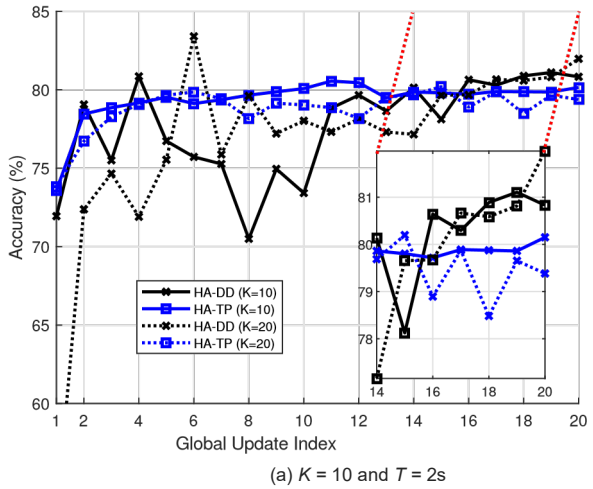


Figure 5-9 Pedestrian dataset validation accuracy results with K =10 and T =5s for DD and TP using both the dynamic HA and the HU schemes.

Figure 5-10 illustrates this behavior more clearly for the HA schemes. There we can examine the accuracy progression with K = 10 and 20 learners and T reduced to 2s; the HU schemes fail with such stringent time constraints. The HA-TP scheme initially seems to converge faster to different accuracy thresholds; but the HA-DD takes over and converges to a higher final accuracy by 1 to 2 percent in both settings. It also appears that HA-DD works better for a larger system such as with 20 learners. One possible interpretation is that the increase in the achievable local updates is large enough to significantly improve accuracy. In contrast, the gain in the number of updates achieved by HA-TP does not offset the impact of smaller dataset sizes on validation accuracy.



(a) K = 10 and T = 2s

Figure 5-10 Comparison of both dynamic HA schemes HA-DD and HA-TP for T =2s and (a) K =10 and (b) K =20.

5.8 Superiority to Centralized Approaches

This section highlights the benefits of the MEL model in comparison to those of centralized learning in terms of higher final validation accuracy achieved and describes the cost in terms of communication overhead. In the context of TP, centralized implies transmitting all data to the edge (or cloud) for analysis. In the context of DD, each learner sends the locally collected dataset to the orchestrator for training. The overall training process in both cases comprises three steps: transmission of the data from one or more learners to the orchestrator, ML model training on the complete data within the total training duration, and the return of the trained ML model to all learners. To emulate this process fairly, we set the processor clock speed to 2.4 GHz for an indoor environment (representing a peer learner or typical wireless routers) and 10 GHz for the outdoor setting, which represents the resources of a cellular base station. The channel parameters remain as described in chapters 5.1 and 5.2, whereas we employ exactly the same settings used to generate the validation accuracy results for figures 5-3 and 5-4 and figures 5-7 through 5-9. The total training time allowed for the MNIST dataset is $10T$ and for the Pedestrian dataset is $20T$. We also assess the communication overhead by providing the ratio of time spent in communication to the total time, which can be defined for each learner as $V_k = (t_k^S + t_k^R)/t_k \; \forall \, k \in \mathcal{K}$. The average communication time across all learners can be obtained by $1/K \sum_{k=1}^{K} V_k$.

Table 5-2 clearly shows that the accuracy of centralized training is sub-par in comparison to the HA schemes in all cases. In fact, the centralized approach does not work at all for the Pedestrian dataset and provides an accuracy of 50 percent, which means the model is guessing randomly. Altering the model parameter—including hidden layer sizes, learning rate, optimizer, etc.—showed no improvement. These gains do incur a high communication overhead in which the HA-TP spends between 22 to 47 percent time in communication, whereas the HA-DD approaches require a communication overhead of 1 to 5 percent. The centralized schemes spend less than 1percent time communicating. However, it is worth emphasizing that the total training time allowed for the distributed MEL model accounts for the communication time and provides a higher final validation accuracy despite these overheads. It can be inferred that dividing the total dataset needed for the learning task across multiple learners significantly enhances the achievable total model updates and final accuracy.

Table 5-2 Final accuracy and communication overhead of the HA schemes in comparison to those of centralized learning.

Environment	Dataset	K	T (s)	TP Acc	DD Acc	Cen Acc	TP Over	DD Over	Cen Over
802.11	MNIST	20	12	**97.31 %**	-	94.63 %	39.60 %	-	1.25 %
802.11	MNIST	10	30	**97.41 %**	-	95.35 %	31.46 %	-	0.38 %
802.11	Pedestrian	20	3	**81.86 %**	-	50.00 %	22.78 %	-	2.73 %
802.11	Pedestrian	10	5	**80.76 %**	-	50.00 %	33.05 %	-	6.83 %
Cellular	MNIST	10	60	**97.41 %**	97.40 %	95.27 %	14.69 %	1.47 %	0.02 %
Cellular	Pedestrian	10	5	**80.76 %**	80.34 %	50.00 %	19.62 %	2.30 %	0.01 %
Cellular	Pedestrian	10	2	80.15 %	**80.83 %**	50.00 %	42.70 %	4.52 %	0.89 %
Cellular	Pedestrian	20	2	79.38 %	**81.98 %**	50.00 %	46.03 %	4.74 %	0.79 %

5.9 Complexity Analysis

As described in previous chapters, the problem of interest is a non-convex quadratically constrained linear program (QCLP) after relaxation, which can still be written in the form of a quadratically constrained quadratic program (QCQP). The solvers of the type we used typically employ a combination of interior-point (IP) methods (e.g., IPOPT) and heuristics. For a convex QCQP, IP methods can achieve a solution in polynomial time but are generally NP-hard for non-convex problems. In contrast, our analytical solution requires the solving of a K^{th} degree polynomial. Many solvers, including MATLAB and LAPACK, find the roots using the Eigen-values of the companion matrix, which can typically be achieved in polynomial time.

The complexity of the polynomial solvers arises mainly from the QR factorization needed for diagonalization, and with fast DQR, algorithms can be achieved in $\mathbf{O}\left(4/3n^3+n^2\right)$ [37]. The complexity for the K^{th} degree-polynomial in our solution can be given by $\mathbf{O}\left(4/3K^3+K^2\right)$. The complexity of the IP method depends upon the number of the quadratic matrices, denoted by n, and the number of quadratic constraints in the problem, denoted by m. The complexity is given by $\mathbf{O}\left(n^{1/2}\left[m+n\right]n^2\right)$ [38], and for our problem, it can be expressed as $\mathbf{O}\left(\left[K+1\right]^{1/2}\left[2K+1\right]\left[K+1\right]^2\right)$. The final expression can be expanded to the following form: $\mathbf{O}\left(\left[K+1\right]^{1/2}\left[2K^3+5K^2+4K+1\right]\right)$; this is for convex problems. The complexity is actually non-polynomial because the relaxed problem is still non-convex.

Figure 5-11 better illustrates the complexity of the solution in terms of the execution times for each method. The results are presented for the HA-TP schemes. The simulations were

run on a laptop with an 8-core Intel i7 processor, and the OPTI toolbox [28] was used for the numerical optimization. For each value of K, the execution time was reported for an average of 100 runs, with different communication and computation capacities across each run. The complexity of the HA analytical schemes is on the order of $\mathbf{O}(K^3)$, whereas it is constant for the HU schemes because the solution is a linear program. On the other hand, the complexity is much higher for the case in which a numerical solution is pursued. It is demonstrable that the complexity of our proposed analytical solution for the HA schemes is comparable to that of the HU scheme. Furthermore, it seems that increasing the global cycle time does not have any impact on the execution time requirements of the HU and HA-TP-Ana schemes. In contrast, a larger T reduces the execution time of the HATP-Num method because it allows the optimizer to reach the solution faster.

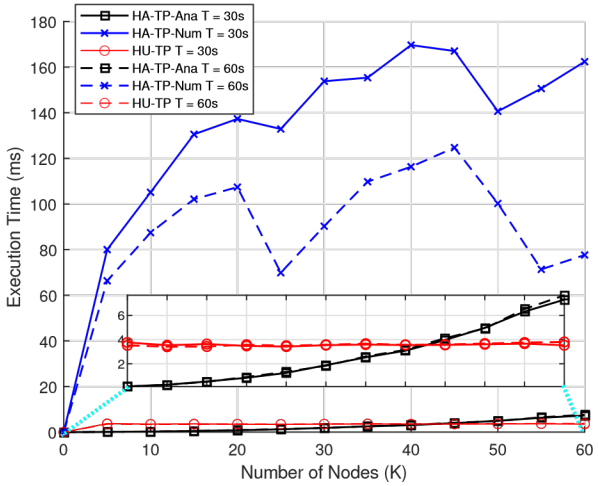


Figure 5-11 HA optimization algorithm execution time comparisons using the MNIST dataset for the TP scheme versus K for T =30 and 60 seconds.

The box plot better illustrates the behavior of the HU and HA-TP-Ana schemes. Interestingly, our proposed solution provides a lower execution time than the ETA approach up to certain values of K ($K = 40$) while simultaneously providing gains in accuracy and reductions in convergence times. Therefore, our proposed method improves the accuracy with a lower execution time for systems of a decent size. Furthermore, in the region where the execution time of our proposed HA method exceeds the HU scheme, the increase in processing time is much less than the reductions in convergence times. The execution time increases on the order of milliseconds, whereas the convergence time saved is on the order of seconds.

CHAPTER 6 SUMMARY AND CONCLUSION

The work presented in this report extends research efforts toward establishing the novel MEL paradigm. This will enable the design of performance-guaranteed distributed learning solutions for wireless edge nodes with heterogeneous computing and communication capacities, allowing their implementation in environments where context aware CV2X communications is crucial. Two different but complementary scenarios for edge learning were proposed, distributed datasets (DD) and task parallelization (TP). This report focuses on exploring the dynamic task allocation solutions that would maximize the number of local learning iterations on distributed learners in either setting, while abiding by the global cycle clock of the orchestrator. The problem was formulated as an NP-hard QCILP problem, which was then relaxed into a non-convex problem over real variables. Analytical upper bounds on the relaxed problem's optimal solution were then derived and were found to provide solutions with a zero optimality-gap in multiple scenarios. Through extensive simulations using well-known datasets, the proposed HA solutions were shown both to achieve the same performance as the numerical solvers of the QCILP problem and to significantly outperform the HU approach.

REFERENCES

[1] U. Mohammad and S. Sorour, "Adaptive Task Allocation for Mobile Edge Learning," in *2019 IEEE Wireless Communications and Networking Conference Workshop (WCNCW)*. IEEE, Apr 2019, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/8902527/

[2] M. Chiang and T. Zhang, "Fog and IoT: An Overview of Research Opportunities," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 854–864, Dec 2016. [Online]. Available: http://ieeexplore.ieee.org/document/7498684/

[3] Rhea Kelly, "Internet of Things Data To Top 1.6 Zettabytes by 2020," 2015. [Online]. Available: https://campustechnology.com /articles/2015/04/15/internet-of-things-data-to-top-1-6-zettab ytes-by-2020.aspx

[4] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A Survey on Mobile Edge Computing: The Communication Perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322– 2358, 2017. [Online]. Available: http://arxiv.org/abs/1701.01090

[5] U. Y. Mohammad and S. Sorour, "Multi-Objective Resource Optimization for Hierarchical Mobile Edge Computing," in *2018 IEEE Global Communications Conference: Mobile and Wireless Networks (Globecom2018 MWN)*, Abu Dhabi, United Arab Emirates, dec 2018, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/8648109

[6] S. Teerapittayanon, B. McDanel, and H. T. Kung, "Distributed Deep Neural Networks over the Cloud, the Edge and End Devices," *Proceedings - International Conference on Distributed Computing Systems*, pp. 328–339, 2017.

[7] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "When Edge Meets Learning : Adaptive Control for Resource-Constrained Distributed Machine Learning," in *INFOCOM*, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8486403

[8] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. PMLR, apr 2017, pp. 1273–1282. [Online]. Available: http://proceedings.mlr.press/v54/mcmahan17a.html

[9] M. J. Sheller, B. Edwards, G. A. Reina, J. Martin, S. Pati, A. Kotrotsou, M. Milchenko, W. Xu, D. Marcus, R. R. Colen, and S. Bakas, "Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data," *Scientific Reports*, vol. 10, no. 1, p. 12598, Dec 2020. [Online]. Available: https://doi.org/10.1038/s41598-020-69250-1

[10] Z. Du, C. Wu, S. Member, and T. Y. Member, "Federated Learning for Vehicular Internet of Things : Recent Advances and Open Issues," *IEEE Open Journal of the Computer Society*, vol. 1, pp. 45–61, 2020. [Online]. Available: https: //ieeexplore.ieee.org/document/9086790

[11] J. Qian, S. P. Gochhayat, and L. K. Hansen, "Distributed Active Learning Strategies on Edge Computing," in *Proceedings - 6th IEEE International Conference on Cyber Security and Cloud Computing, CSCloud 2019 and 5th IEEE International Conference on Edge Computing and Scalable Cloud, EdgeCom 2019*. Paris, France: Institute of Electrical and Electronics Engineers Inc., Jun 2019, pp. 221–226. [Online]. Available: https://ieeexplore.ieee.org/document/8854053

[12] C.-F. Liu, M. Bennis, and H. V. Poor, "Latency and ReliabilityAware Task Offloading and Resource Allocation for Mobile Edge Computing," in *2017 IEEE Globecom Workshops (GC Wkshps)*, 2017, pp. 1–7. [Online]. Available: http://arxiv.org/abs/1710.00590http: //ieeexplore.ieee.org/stamp/stamp.jsp?tp={\&}arnumber=826 9175{\&}isnumber=8269020

[13] H. Yu, Z. Liu, Y. Liu, T. Chen, M. Cong, X. Weng, D. Niyato, and Q. Yang, "A Sustainable Incentive Scheme for Federated Learning," *IEEE Intelligent Systems*, no. (Early Access), pp. 1–9, 2020. [Online]. Available: https://ieeexplore.ieee.org/document /9069185

[14] J. Dean, G. S. Corrado, R. Monga, K. Chen, M. Devin, Q. V. Le, M. Z. Mao, M. A. Ranzato, A. Senior, P. Tucker, K. Yang, and A. Y. Ng, "Large Scale Distributed Deep Networks," in *Advances in Neural Information Processing Systems 25*, 2012, pp. 1223–1231. [Online]. Available: https://reports.nips.cc/report/4687-large-sc ale-distributed-deep-networks

[15] Z. Wei, S. Gupta, X. Lian, and J. Liu, "Staleness-Aware Async-SGD for distributed deep learning," *IJCAI International Joint Conference on Artificial Intelligence*, vol. 2016-Janua, pp. 2350–2356, 2016.

[16] T. Tuor, S. Wang, T. Salonidis, B. J. Ko, and K. K. Leung, "Demo abstract: Distributed machine learning at resource-limited edge nodes," *INFOCOM 2018 - IEEE Conference on Computer Communications Workshops*, pp. 1–2, 2018.

[17] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive Federated Learning in Resource Constrained Edge Computing Systems," *IEEE Journal on Selected Areas in Communications*, no. Early Access, pp. 1–1, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8664630/

[18] D. Conway-Jones, T. Tuor, S. Wang, and K. K. Leung, "Demonstration of Federated Learning in a Resource-Constrained Networked Environment," in *2019 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2019. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8784064

[19] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A Joint Learning and Communications Framework for Federated Learning over Wireless Networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 1, pp. 269 – 283, 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9210812https: //arxiv.org/abs/1909.07972

[20] Z. Yang, M. Chen, W. Saad, C. S. Hong, and M. Shikh- Bahaei, "Energy Efficient Federated Learning Over Wireless Communication Networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 3, pp. 1935 – 1949, Nov 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9264742

[21] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Convergence Time Minimization of Federated Learning over Wireless Networks," in *IEEE International Conference on Communications*. Institute of Electrical and Electronics Engineers Inc., Jun 2020, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/91488 15

[22] ——, "Convergence Time Optimization for Federated Learning over Wireless Networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 4, pp. 2457–2471, apr 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9148815

[23] L. Bottou and O. Bousquet, "The Tradeoffs of Large Scale Learning," in *Advances in Neural Information Processing Systems*, J. C. Platt, D. Koller, Y. Singer, and S. Roweis, Eds. NIPS Foundation (http://books.nips.cc), 2008, vol. 20, pp. 161–168. [Online]. Available: http://leon.bottou.org/reports/bottou-bous quet-2008

[24] Y. Hu, M. Chen, M. Chen, Z. Yang, M. Shikh-Bahaei, H. V. Poor, and S. Cui, "Energy Minimization for Federated Learning with IRS-Assisted Over-the-Air Computation," in *ICASSP 2021 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, Jun 2021, pp. 3105–3109. [Online]. Available: https://ieeexplore.ieee.org/document/9414785/

[25] M. M. Amiri, S. Member, D. Gund¨ uz,¨ and S. Member, "Machine Learning at the Wireless Edge : Distributed Stochastic Gradient Descent Over-the-Air," *IEEE Transactions on Signal Processing*, vol. 68, pp. 2155–2169, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9042352

[26] U. Mohammad, S. Sorour, and M. Hefeida, "Task Allocation for Asynchronous Mobile Edge Learning with Delay and Energy Constraints," *arXiv preprint*, vol. 14, no. 8, pp. 1–12, 2020. [Online]. Available: https://arxiv.org/abs/2012.00143

[27] A. D. Pia, S. S. Dey, and M. Molinaro, "Mixed-integer Quadratic Programming is in NP," *Mathematical Programming*, vol. 162, no. 1, pp. 225–240, 2017.

[28] J. Currie and D. I. Wilson, "OPTI: Lowering the Barrier Between Open Source Optimizers and the Industrial MATLAB User," in *Foundations of Computer-Aided Process Operations*, N. Sahinidis and J. Pinto, Eds., Savannah, Georgia, USA, 2012.

[29] S. Cebula, A. Ahmad, J. M. Graham, C. V. Hinds, L. A. Wahsheh, A. T. Williams, and S. J. DeLoatch, "Empirical channel model for 2.4 GHz ieee 802.11 wlan," *Proceedings of the 2011 International Conference on Wireless Networks*, 2011.

[30] X. Lyu, H. Tian, C. Sengul, and P. Zhang, "Multiuser joint task offloading and resource optimization in proximate clouds," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 4, pp. 3435–3447, 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7517217

[31] X. Zhang, Y. Mao, J. Zhang, and K. B. Letaief, "MultiObjective Resource Allocation for Mobile Edge Computing Systems," in *IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), 2017*, vol. 19, no. 4, Montreal, 2017, pp. 1–5. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8292379

[32] Z. Ji, L. Chen, N. Zhao, Y. Chen, G. Wei, and R. Yu, "Computation Offloading for Edge-Assisted Federated Learning," *IEEE Transactions on Vehicular Technology*, no. Early Access, pp. 1–1, 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9492062/

[33] S. Munder and D. M. Gavrila, "An experimental study on pedestrian classification." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 11, pp. 1863–1868, 2006. [Online]. Available: https://ieeexplore.ieee.org/document/1704841

[34] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based Learning Applied to Document Recognition," *Proceedings of IEEE*, vol. 86, no. 11, pp. 2278 – 2324, 1998. [Online]. Available: https://ieeexplore.ieee.org/document/726791

[35] Brendan Shillingford, "What is the time complexity of backpropagation algorithm for training artificial neural networks? - Quora," 2016. [Online]. Available: https://www.quora.com/What-is-the-time-complexity-of-backpropagation-algorithm-for-training-artificial-neural-networks

[36] A. Abutuleb, S. Sorour, and H. S. Hassanein, "Joint Task and Resource Allocation for Mobile Edge Learning," in *2020 IEEE Global Communications Conference, GLOBECOM 2020 - Proceedings*. Institute of Electrical and Electronics Engineers Inc., dec 2020, pp. 1–6. [Online]. Available: https: //ieeexplore.ieee.org/document/9322399

[37] F. Uhlig, "The DQR algorithm, basic theory, convergence, and conditional stability," *Numerische Mathematik*, vol. 76, no. 4, pp. 515–553, 1997.

[38] A. Nemirovski, "Interior point polynomial time methods in convex programming," *Lecture Notes*, vol. 42, no. 16, pp. 3215– 3224, 2004. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.160.6909{\&}rep=rep1{\&}type=pdf