FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING
DEGREE PROGRAMME IN WIRELESS COMMUNICATIONS ENGINEERING

MASTER'S THESIS

COMMUNICATION-EFFICIENT SCHEDULING POLICY FOR FEDERATED LEARNING UNDER CHANNEL UNCERTAINTY

Author	Madhusanka Dinesh Weeraratne Manimel Wadu
Supervisor	Prof. Mehdi Bennis
Second Examiner	Dr. Sumudu Samarakoon

December 2019

Manimel Wadu M. (2019) Communication-Efficient Scheduling Policy for Federated Learning Under Channel Uncertainty. University of Oulu, Degree Programme in Wireless Communications Engineering, 43 p.

ABSTRACT

Federated learning (FL) is a promising decentralized training method for on-device machine learning. Yet achieving a performance close to a centralized training via FL is hindered by the client-server communication. In this work, a novel joint client scheduling and resource block (RB) allocation policy is proposed to minimize the loss of accuracy in FL over a wireless system with imperfect channel state information (CSI) compared to a centralized training-based solution. First, the accuracy loss minimization problem is cast as a stochastic optimization problem over a predefined training duration. In order to learn and track the wireless channel under imperfect CSI, a Gaussian process regression (GPR)-based channel prediction method is leveraged and incorporated into the scheduling decision. Next, the client scheduling and RB allocation policy is derived by solving the aforementioned stochastic optimization problem using the Lyapunov optimization framework. Then, the aforementioned solution is extended for scenarios with perfect CSI. Finally, the proposed scheduling policies for both perfect and imperfect CSI are evaluated via numerical simulations. Results show that the proposed method reduces the accuracy loss up to $25.8\,\%$ compared to FL client scheduling and RB allocation policies in the existing literature.

Keywords: Federated learning, channel prediction, client scheduling, Gaussian process regression.

TABLE OF CONTENTS

FOREWORD

This thesis introduces a novel joint client-scheduling and resourceblock allocation policy derived via Lyapunov optimization framework for federated learning under imperfect channel state information. I express my sincere gratitude to my supervisor, Dr. Sumudu Samarakoon for his support and guidance throughout my masters studies and Prof. Mehdi Bennis for his excellent directions and suggestions. I must mention, Prof. Markku Juntti and Prof. Nandana Rajathewa for giving me the opportunity to work as research assistant, in university of Oulu which was actually one of my turning point of my life and learnt a lot by working with ICON group. Further, I should thank Mrs. Inosha Sugathapala for letting me know about learning and researching opportunities in university of Oulu. Finally, I thank my mother and my wife for the solid support in my ups and downs throughout my life.

Oulu, 6th December, 2019

Madhusanka Dinesh Weeraratne Manimel Wadu

LIST OF ABBREVIATIONS AND SYMBOLS

AI	Artificial Intelligence
AdaGrad	Adaptive Gradient
BS	Base Station
BER	Bit Error Rate
CC	Central Controller
CCDF	Complementary Cumulative Distribution Function
CEN	Centralized training
CoCoA	Communication-efficient distributed dual Coordinate Ascent
CSI	Channel State Information
CSI-R	Channel State Information at the receiver
CSI-T	Channel State Information at the transmitter
DPP	Drift Plus Penalty
FEDPAQ	Federated Learning with Periodic Averaging and Quantization
Full-CSI	Channel State Information at both transmitter and receiver
GD	Gradient Descent
GP	Gaussian Process
GPR	Gaussian Process Regression
IDEAL	Scheduling without communication constraints
IID	Independently Identically Distributed
KSGD	Kalman-based Stochastic Gradient Descent
LoS	Line of Sight
MAE	Mean Absolute Error
MFG	Mean Field Game
ML	Machine Learning
MSE	Mean Sqaure Error
NN	Neural Network
QAW	Quantity Aware Scheduling
QAW-GPR	Quantity Aware Scheduling with GPR
QUNAW	Quantity Unaware Scheduling
RAND	Random
RB	Resource Block
ReLU	Rectified Linear Unit
RMSE	Root Mean Squared Error
RMSLE	Root Mean Squared Logarithmic Error
RMSProp	Root Mean Square Propagation
RR	Round Robin
RV	Random Variable
PF	Propotional Fair
SGD	Stochastic Gradient Descent
SINR	Signal to Interference plus Noise Ratio
SVM	Support Vector Machine
UDN	Ultra Dense Networks

Symbols

K	Total number of users
k	Signle user
$\mathcal{K}$	Set of users
D	Total dataset size
$\mathcal{D}$	Dataset
s	Scheduling variable
$\boldsymbol{s}$	Scheduling vector
$\boldsymbol{S}$	Scheduling matrix
B	Number of resource blocks
b	Single resource block
X	Inputs
$\boldsymbol{x}$	Vector of inputs
y	Value label
h	Channel
$\hat{h}$	Channel estimation
λ	Resource allocation
$\boldsymbol{\lambda}$	Resource allocation vector
$\boldsymbol{\Lambda}$	Resource allocation matrix
γ	SINR
$\hat{\gamma}$	SINR estimate
γ_0	SINR threshold
ε	Error
ν	Auxilary variable for Lyapunov framework - upper bound
l	Auxilary variable for Lyapunov framework - knowledge
q	Virtual queue for tracking upper bound of the error
g	Virtual queue for tracking knowledge
ϕ	Trade off parameter
f_m	Doppler frequency
φ	Coefficient of exploration
ν^{avg}	Running average
f	Loss function
r	Rate
p	Power
N_0	Noise
I	Interference
ψ	Dual function
ς	Variance
ϱ	Regualizer
θ	Dual variable
argmax	Argument maximum
argmin	Argument minimum
w	Model parameter (weight)
$\boldsymbol{w}$	Model parameter (weight) vector
j	Knowledge
$\boldsymbol{j}$	Knowledge vector

J	Knowlegde matrix
c	Covariance
$\boldsymbol{C}$	Covariance matrix
ζ_1	Sin-squared kernal length parameter
ζ_2	Sin-squared kernal period parameter
T	Training duration

Functions

$\sum\sum$	Double summation
$\lvert\boldsymbol{x}\rvert$	Euclidean norm of vector $\boldsymbol{x}$
$\mathbb{I}(x)$	Indicator function, i.e. returns 1 if $x \geq 0$, 0 otherwise
$L(\cdot)$	Lyapunov function
$\mathbb{P}(\cdot)$	Probability of the event
$\mathbb{E}(\cdot)$	Expected value of a random variable or variable
$(\cdot)^\star$	Solution of an optimization problem
$\mathbf{X}^\dagger$	Transpose of $\mathbf{X}$
$\mathbf{0}$	Vector of 0s, i.e., $(0, \ldots, 0)$; size of the vector is implicit
$\mathbf{1}$	Vector of 1s, i.e., $(1, \ldots, 1)$; size of the vector is implicit

1 INTRODUCTION

Artificial intelligence (AI) is defined as "the science and engineering of making intelligent machines" by John McCarthy, recognized as one of the fathers of AI [1]. After defeating the top human "Go" players by "AlphaGo", the huge potential in AI is proven to the world without a doubt [2]. AI can be explicitly programmed by using a pile of if-then statements or a complex statistical data mapping to a model which is referred as machine learning (ML). Thus, machine learning is a branch of AI and it allows systems to learn from data without being explicitly programmed by human.

ML model becomes more accurate in predicting or making decisions with the increasing size of the dataset it has been trained on [3]. Thus for ML, data plays a crucial role. In cloud ML, also denoted as "centralized" method, it is required all training data to be available on one single place, where the ML model is trained. On the other hand, proliferation of internet of things (IoT) such as cameras, wearable devices, and autonomous vehicles generating wealth of data every second. Coupling with ML applications, this data opens up doors for plenty of possible meaningful applications [4]. Applications dedicated for medical purposes and for vehicular networks are such instances. However, users' unwillingness to reveal privacy-sensitive data (i.e., medical treatment details) for such applications prevents it from its growth. Typically, these data is unevenly distributed over a large number of devices, and every device has a tiny fraction of the data. Thus, these devices are required to offload the raw data to a cloud for training ML model, since quality of the ML model will be low if the model is trained on an edge device alone with its tiny portion of data [3]. Further, most of these devices are inter connected with one another and servers over wireless networks. In addition to the infrastructure and power costs in wireless networks, since bandwidth is rare, over-the-air communication is a costly option. Thus, in "centralized" approach to train a ML model, communication cost plays a significant figure in addition to aforementioned privacy concerns [5]. However, a promising solution for the aforementioned issues, is to adhere with distributed ML techniques where model is trained locally with local datasets and aggregated at one central device mostly at the edge of network.

Among the distributed ML techniques, *federated learning* (FL) is one of the most commonly used technique. In FL, the goal is to train a high quality ML model whereby training data reside distributed over a set of clients [6]. Therein, clients perform local computing to train unique neural network (NN) model with their own data to minimize a predefined empirical loss function. Thereafter, each client share its locally trained model parameters (trained weights of local NN model) with a server for model averaging. Then the server aggregates those models and produce an improved global ML model. Next, aggregated model is broadcast back to clients, completing a "global iteration". Afterwards, clients adopt the received global model and conduct further local trainings. This training process iteratively continues, until a high quality ML model is trained. Example applications of federated learning are natural language processing [7], face and voice recognition [8] at hand-held devices, pedestrian behavior analysis in autonomous vehicles [9] and diagnosing diseases from wearable devices [10].

ML with FL, enhances users' hyper-personalized experience in web surfing, online shopping, etc, while preserving data privacy [6]. The reason is devices communicate and exchange only their locally trained models instead of their private data. Moreover in FL, communication overhead is reduced significantly compared to the "centralized" ML

approach. However, with deep NN models with millions of parameters, sharing model parameters or dataset is going to be more or less the same.

Recent decade, the computational power of handheld devices have significantly overwhelmed [11]. Thus, an added advantage in distributed ML techniques, is computation offloading from cloud to devices. FL with these key features have brought up being a promising edge ML algorithm ensuring "code to data", instead of "data to code".

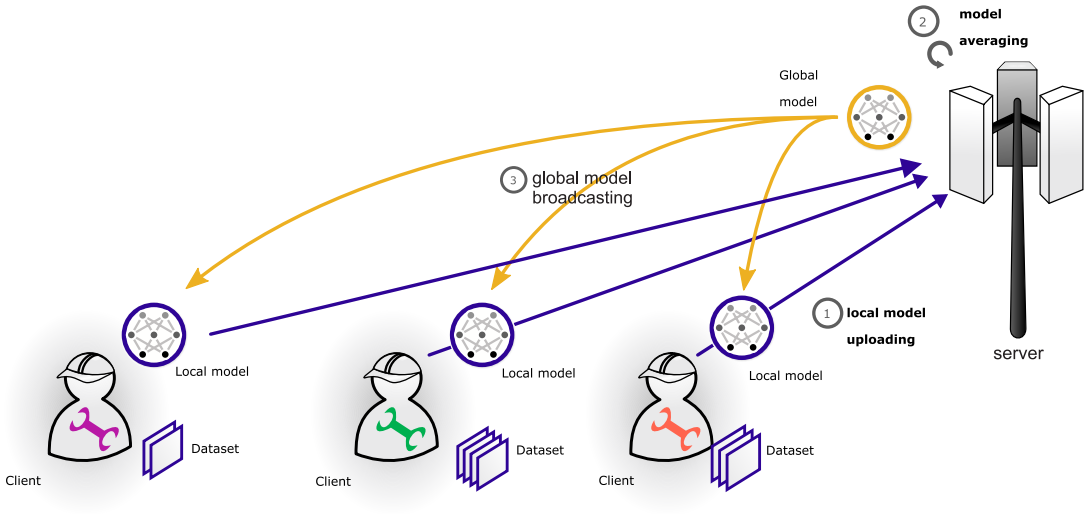

Figure 1. Federated learning model

Further, in FL, it is assumed that data distribution among clients are independently identically distributed (IID) for the convergence of FL to centralized approach [6]. It is numerically shown that, convergence is achieved when the number of global model exchanging communication rounds are approaching infinity. On the other hand, variance in the number of data per client is a measure of heterogeneity of dataset distribution. With the increase of heterogeneity of dataset distribution, affects further increasing the trained model accuracy gap between centralized approach and FL [12]. Moreover, as mentioned, most of the handheld devices are connected over wireless networks. Hence, the reliability of wireless networks affects heavily on the accuracy of the model trained over wireless network with FL.

Typically, in a wireless link, the received signal power attenuates with increasing distance between the transmitter and the receiver. On top of this attenuation, obstructions like buildings, trees, adds fading to received signal power. The reason for fading is, reflections and scattering from various objects typically creates several copies of the transmitted signal and reach the receiver via multiple paths with different delays. This phenomenon is specifically denoted as "multipath fading" in wireless communication [13]. The effect to the wireless channel is that, this leads to a high error rate with lower signal-to-noise ratio. Additionally, the over-the-air link between a client and server, is subject to interference from other clients and network users in the system. Wireless channels are unreliable, because of those aforementioned interferences, multipath fading, etc. Further, wireless networks are resources limited (i.e., limited spectrum). Hence, the amount of resource allocation for wireless clients are generally constrained in an wireless environment. The quality of the trained model in FL, will heavily be affected when the

training process continues over unreliable wireless links with resource constraints. As a result, this leads to more emphasis on allocating available spectrum to each client in an efficient manner, exploiting the unreliability of the wireless channel behavior. As a solution, it is possible to schedule most effective model parameters to update or most effective model updating clients per global iteration to share their models or to quantize the model updates to reduce the communication overhead.

In order to adapt wireless transmissions to unreliable channel conditions, it is required to measure the channel propagation characteristics between transmitters and receivers beforehand. This information on channel state is technically denoted as channel state information (CSI). Measuring CSI beforehand, plays an essential role in wireless system performance in bit error rate (BER) in decoding phase [13]. CSI represents the combined effect of multipath fading, scattering and signal power decay with distance. However, the acquisition of accurate CSI timely involve considerable resource overhead in limited resource environment. As an alternative approach to estimate the channel is to predict the channel with an efficient online prediction scheme. With this it is possible to save the resource overhead involve in channel estimation. However, predicting channel with online learning tool is more or less erroneous. Thus, there is a trade off with accuracy of CSI and the expenditure of resource overhead in limited spectrum.

1.1 Overview

The main contribution of this work is **a novel joint client-scheduling and resource block (RB) allocation policy for FL under imperfect CSI.** I consider a set of clients that communicates with a server over wireless links to train a NN model within a predefined training duration. First, derive an analytical expression for the loss of accuracy in FL with scheduling compared to a centralized training method. Here, the CSI knowledge is obtained via CSI prediction and knowledge exploration-based sampling. Then, cast the client scheduling and resource block (RB) allocation problem to minimize both the loss of FL accuracy and the CSI prediction uncertainty under communication constraints. Therein, the objective of the scheduling problem is to optimize a tradeoff between the accuracy loss minimization and maximizing the network-wide knowledge on CSI. Due to the stochastic nature of the aforementioned problem, resort to the drift-plus-penalty (DPP) technique from the Lyapunov optimization framework to decouple the problem into a per time solvable sub problems [14]. Finally, the proposed solution under both perfect and imperfect CSI are evaluated via numerical simulations. Simulation results show that the proposed solution achieve up to 25.8 % reduction in loss of accuracy compared to state-of-the-art client scheduling and RB allocation methods.

The presenting of the thesis is organized as follows. After this introduction in this section 1, presents state-of-the-art. Then in section 2, presents the system model and formulates the NN model training over wireless links, under imperfect CSI as an optimization problem. In Section 3, formulated problem is recast in terms of loss of accuracy with scheduling compared to a centralized training method. Then, GPR-based CSI prediction technique is introduced and Lyapunov optimization is used to solve the problem and derive client scheduling and RB allocation policy under both perfect and imperfect CSI. Section 4 evaluates the proposed scheduling policies. Finally, conclusions are drawn in Section 5.

1.2 State-of-the-art

1.2.1 Neural network model based machine learning : A brief overview

A neural networks performs similar functionality with neurons in human nervous system. Consider an example neural network model in Fig. 2 consisting of three layers: hidden layer, input layer and output layer.

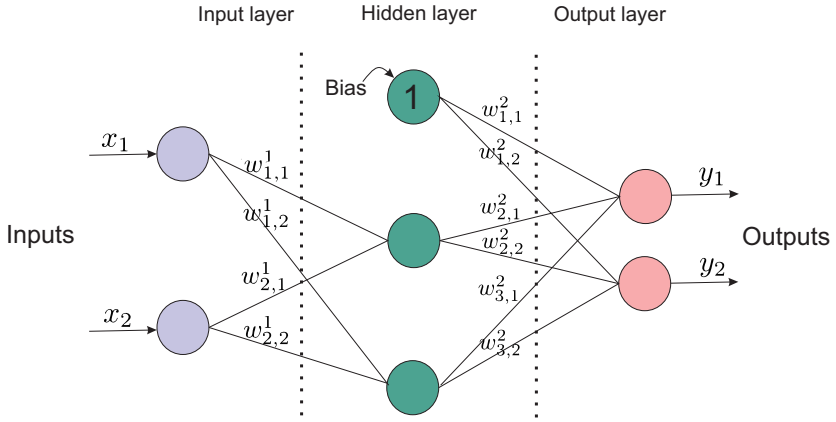

Figure 2. Neural network model

Inputs to a NN model can be texts, audio files, image pixels, etc. The inputs ($\boldsymbol{x}$) that are fed into the input layer are then multiplied with their corresponding weights ($\mathbf{w}^1$). These weights indicates how important that input is to the outcome. Then, weighted values are summed up, passed through an activation function (i.e., tanh, rectified linear unit (ReLU), sigmoid) and forwarded to the next layer. This continues up to the output layer. In this view, the input-output relation is cast as a functional approximation $y = g(\mathbf{x}, \mathbf{w})$ that depends on the choice of weight parameters. Here, the values of the weights are known as *the model*. The node called *bias*, allows to adjust the activation function curve to meet desired output. The goal is to tune the weights that infer outputs for given inputs. This procedure is known as the model training. Within the scope of this thesis, we only focus on NN model training with the supervised learning. Under supervised learning, a set of inputs and their corresponding labelled outputs known as *training data* is used for the model training. At the beginning of model training a randomly chosen values are assigned as the weights. Then the weight parameters are optimized using a loss function quantifies the performance of the trained NN model.

Loss function and regularization:

The loss functions measures and quantifies the performance of the trained model. Based on the application requirement: to derive a functional approximation (*regression*) or to classify data (*classification*), the choice of the loss function varies. Table 1 lists down a set of widely used loss functions. Here, for i-th input training data, y_i is the actual output, $\hat{y}_i$ is the prediction from NN model, $y_{i,j} = 1$ if i-th element belongs to class j and 0 otherwise, and $p_{i,j}$ is the probability that i-th element in class j. Moreover, a training dataset $\mathcal{D}$ consisting D samples assumed.

Table 1. Commonly used loss functions for NN model training [15].

Loss function	Definition	Application
Mean absolute error (MAE) *or* ℓ-1 loss	$\frac{1}{D}\sum_{i\in\mathcal{D}}\lvert y_i - \hat{y}_i\rvert$	Regression
Mean square error (MSE) *or* ℓ-2 loss	$\frac{1}{D}\sum_{i\in\mathcal{D}}(y_i - \hat{y}_i)^2$	Regression
Multi-class cross entropy *or* negative loglikelihood	$-\sum_{i\in\mathcal{D}} y_{i,j}\log_2(p_{i,j})$	Classification
Multi-class support vector machine (SVM) *or* Hinge loss	$\max(0, 1 - y_{i,j}\hat{y}_{i,j})$	Classification

In addition to a loss function, a regularizer $\varrho(\cdot)$ function is used, to penalize unnecessary over-fitting of $\boldsymbol{w}$ to the dataset in training process [16]. Overfitting means that the output of the trained model perfectly (almost) align with the labelled outputs in the training dataset while exhibits poor performance with new data, i.e. high training accuracy and low inference accuracy. The reason for this is few weight parameters in the model are heavily dominating the prediction. The term over-fitting is clearly convincing from the following Fig. 3.

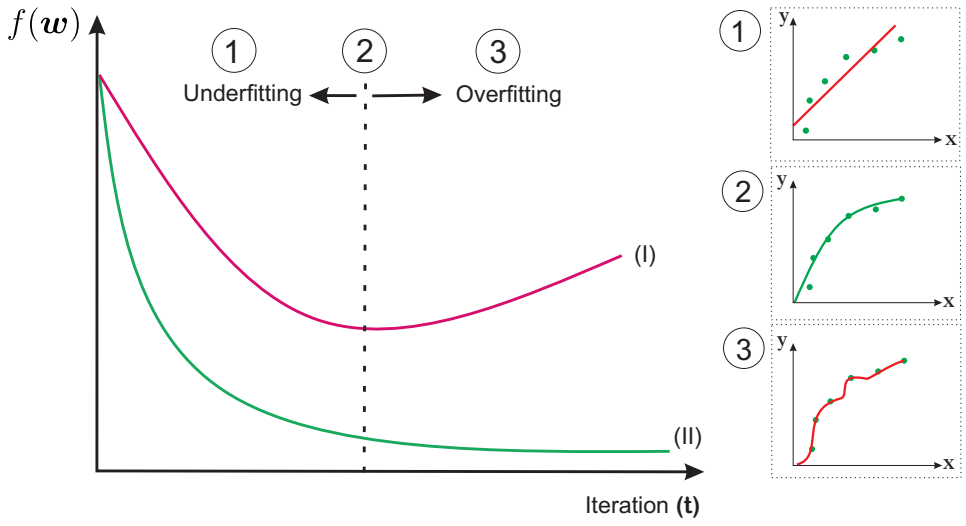

Figure 3. (Left) Changes of the loss function $f()$ with model update iterations over (i) training data and (ii) validation data, and (Right) regressed function.

Generally, two datasets are used in NN model training as shown in Fig. 3. Those are used for the purpose of training the NN model and for the validation separately. The validation dataset, which is not used for training of NN, evaluatesthe performance of the trained model against unseen new data. This is called test accuracy. From Fig. 3, it can be observed that the test loss function value decreases with training iterations. Excess training further improves the training accuracy resulting the outputs to be biased towards only the training labels. Hence, the model no longer will be suitable to infer the unseen data. As a result, loss over the validation data increases and test accuracy regrades.

Hence, thereafter the value of $f(\boldsymbol{w})$ increases for the validation dataset. Thus, that point is the good choice that depends on the choices of training data. To overcome this kind of over-fittings a regularization function is added to optimization problem with respect to weights. Introducing a regularizer penalize the weights which are growing too much. Such most commonly used regularization functions and their features are described as follows:

- L1 norm (Lasso) - $||\boldsymbol{w}||_1$: This has the property that force $\boldsymbol{w}$ to become sparse during optimization/ training, since shrink the parameters to zero. In the solution for a L1 regularized training, majority of features have zero weights and very few will have non zero values. Thus, L1 norm in regularizer produces a simple model, that is interpretable and contains only a subset of inputs for decision making.
- L2 norm (Ridge/ Tikhonov) - $||\boldsymbol{w}||_2$: Here, the regularization term is the sum of squares of weights ($\boldsymbol{w}$). L2 regularization forces weights to be small but not make them zero as in L1 norm. Thus, L1 norm does non sparse solution.

1.2.2 Distributed optimization for machine learning

Distributed optimization offers the promising scalability in ML systems and computational offloading capability. The challenge for that goal is, developing an communication efficient distributed optimization algorithm for the distributed clients.

CoCoA method:

In [17], a framework for distributed optimization is presented, denoted as communication-efficient distributed dual coordinate ascent (CoCoA). Authors of this work have developed a primal-dual framework that can be applied for variety of convex optimization problems. Next, authors have derived the convergence rate of the proposed CoCoA method for minimizing the objectives with convex regularized loss functions. The significance in CoCoA distributed optimization is, that the scalability does not degrade the convergence speed. In this work, decoupled subproblems which are independently solved subproblems in each client, are allowed to be solved to any accuracy level towards its optimal solution depending on client's computational capabilities. Thus, the proof that presents in this work, is helpful in analysis on computation vs. communication trade-off.

Federated Learning:

Similarly to the work in [17], a distributed optimization framework is proposed in [6] without using primal-dual formulation. The goal of this work is to optimize a set of parameters $\boldsymbol{w}$ that minimizes an arbitrary cost, which is separable over the data points as follows:

$$\min_{\boldsymbol{w}} f(\boldsymbol{w}) = \frac{1}{n} \sum_{i=1}^{n} f_i(\boldsymbol{w}; x_i, y_i), \tag{1}$$

where n is the number of data points, $\boldsymbol{w}$ are the model parameters vector, and $(\boldsymbol{x}_i, y_i)$ is the i-th data inputs and its corresponding label. The function f_i is typically a loss function that numerically represents the how efficiently neural network model predict

with $\boldsymbol{w}$ with input $\boldsymbol{x}_i$ with respect to given label or output y_i. This is the first article proposing distributed machine learning optimization denoted as *federated learning*. In this work they have assumed that, data points are IID distributed across a large number clients. Further, data lies on clients' handheld devices, which may be privacy sensitive. Authors of this work have proposed an algorithm to solve (1) in distributed setting. Notably, proposed algorithm has achieved higher convergence rate compared to CoCoA method in [17]. Idea in the algorithm is that distributed clients train a neural network model with each local datasets inside global iteration. Then each client update weights to a server where models are averaged and produce $\tilde{w}$. This is broadcast back to clients and for the next communication round $\tilde{w}$ is used as reference model for all clients in solving their sub problem. However, in this proposed distributed setting, largest bottleneck is the communication overhead. This overhead scales with the number of clients and the depth of NN model (number of layers, number of nodes per layer, and connectivity among layers), in which the communication conditions need to be accounted towards the design of fast and accurate FL over-the-air.

Optimizer - Stochastic gradient descent (SGD) technique:

Optimizers are used to minimize objective functions updating parameters in them. In ML model training applications to solve (1), using GD technique performs update in (2) but using dataset $\mathcal{D}$. The capabilities of ML methods are limited computing being the limiting factor by the need of high computational power over large datasets. Since, selecting a large dataset for training on every iteration of GD is computationally exhaustive, choosing a random subset of data samples and run GD is more efficient and faster [18]. Stochastic gradient descent (SGD) is an implementation of this concept using a batch of data samples solving the aforementioned issue. As a solution for the above limitation SGD, the stochastic approximation of GD, is proposed [18]. Here, instead of deriving the gradient over the whole dataset, the gradient is calculated over a subset of data (a batch) as follows:

$$\boldsymbol{w} := \boldsymbol{w} - \eta \nabla f(\boldsymbol{w}) = \mathbf{w} - \eta \sum_{i \in \mathcal{S} \subset \mathcal{D}} \frac{\nabla f_i(\boldsymbol{w})}{|\mathcal{S}|} \tag{2}$$

where η is the learning rate, which determines how far for each step of update $\boldsymbol{w}$ should change in the direction of $\nabla f(\boldsymbol{w})$, $\mathcal{S}$ is the subset of the original dataset $\mathcal{D}$, and $|\mathcal{S}|$ is the cardinality of the subset $\mathcal{S}$. This greatly reduces the computational complexity in terms of matrix multiplication and inversion. Continuing this process iteratively will achieve $f(w)|_{w=w^\star}$ as low as possible if f is convex.

Fig. 4 shows the effect of the learning rate (η) in minimization of f during the training process.

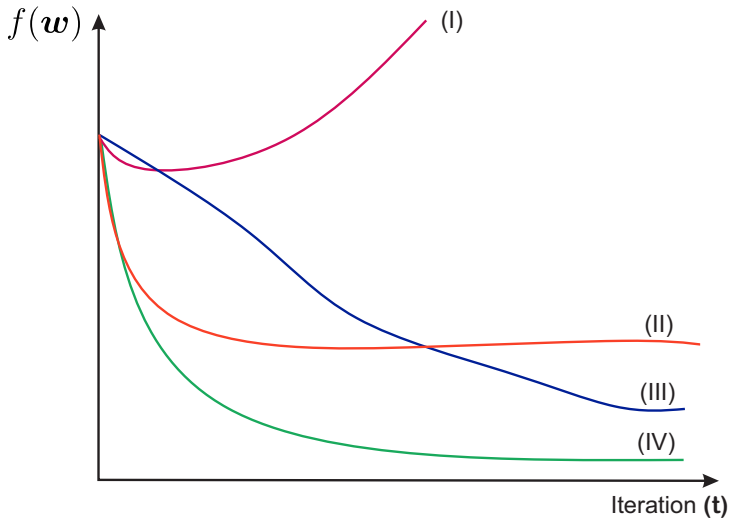

Figure 4. Effect of learning rate to change in the objective function value per iteration, with (i) too higher (ii) high (iii) low and (iv) just perfect learning rate values.

Mostly, the choice of η yields a tradeoff between convergence speed and the optimality of the solution. If loss function f is convex, then there should exist a best choice for learning rate that will provide the fastest convergence as it is guaranteed to achieve the optimality.

The main advantage of using SGD as optimizer for NN model training is that the reduction in computational burden and faster iterations. SGD further have the advantage of escaping local minima due to the randomness in parameter updates. The reason for this is a gradient calculated over stochastically chosen subset of dataset may direct towards a locally nonoptimal solution, allowing the iterative process to escape a local minimum. However, aforementioned advantages are achieved in expense of slightly less convergence rate with respect to GD. Due to the randomness occurred by selecting a portion of dataset, undesirable fluctuations can be observed in weight updates. Another limitation is learning rate is not adaptively change since the learning rate is fixed throughout the learning process. To resolve above limitations in SGD, several variants have been proposed in the existing literature. The following are few of such variants of SGD in literature.

- **Momentum method** [19]: SGD with momentum method remembers the update of the weight in each previous iteration, and computes the next weight change with contribution of previous update. This is mathematically represented as $\boldsymbol{w} := \boldsymbol{w} + \alpha\Delta\boldsymbol{w} - \eta\nabla f(\boldsymbol{w})$, where α is the contribution factor. This SGD variant tends to keep the same direction more likely, preventing from oscillations unlike in original form of SGD. Thus, this method has been used in the training of NN.

- **Averaged SGD** [20]: This SGD variant records the average of its parameter vector over time. The mathematical representation of the update is, $\boldsymbol{w} := \boldsymbol{w} - \eta\nabla f(\frac{1}{t}\sum_{i=1}^{t}\boldsymbol{w})$. In this variant of SGD, reduces the fluctuations of weight updates compared to the original form of SGD.

- **RMSProp** (Root Mean Square Propagation) [21]: In RMSProp, the key idea is to divide the learning rate by running average of the magnitudes of gradients in terms of mean square. This method keeps the effect of previous gradient changes while admitting new values. Towards this, a new function $v(\boldsymbol{w}, t) := \gamma v(w, t-1) + (1-\gamma)(\nabla f(\boldsymbol{w}))^2$, is introduced where γ is the forgetting factor. Then the weights at each iteration is updated as, $\boldsymbol{w} := \boldsymbol{w} - \frac{1}{\sqrt{v(\boldsymbol{w},t)}}\eta \nabla f(\frac{1}{t}\sum_{i=1}^{t}\boldsymbol{w})$

There are several other SGD optimization variants such as AdaGrad (adaptive gradient), Kalman-based Stochastic Gradient Descent (kSGD), etc. SGD in any above variants will have specific features towards some application requirements in distributed ML like adaptive learning rates, smooth weight updates, etc.

1.2.3 Distributed machine learning under communication constraints

As mentioned in section 1.2.2, communication overhead is the main bottleneck in FL and model parameter updating utilize communication resources depending on the model depth and number of clients. However, if the training with FL communicates over a resource constrained wireless, then model parameter updating might be impossible for some clients. Furthermore, due to channel uncertainty, model updating may not be smooth, even though resources are available. Except few works considering FL over wireless networks, the vast majority of the existing literature in FL assumes ideal client-server communication conditions, which means no resource limitations or channel uncertainties.

LAG - reusing outdated gradient concept:

In [22], a gradient updating technique named *lazily aggregate gradients (LAG)* is proposed to reduce communication overhead by reusing outdated gradient updates. Therein, a set of simple rules are imposed to detect slowly varying gradients and to reuse the outdated gradients. This prevents transmitting unnecessary (less effective) updates and saves bandwidth. The work generally can be applied for distributed machine learning as well as multi-agent optimization, and distributed signal processing [22].

With contrast to GD, in LAG use either the current gradient update from a client or an outdated gradient computed using previous gradient update. The rule to update the gradient or not depends on the difference between current aggregated gradient and current calculated gradient. Thus, server first broadcast aggregated model update to all clients, then if clients with aforementioned difference greater than some threshold then update the difference only. Here this rule is decided by worker and the other method is server determines which client should send their gradient difference. Then those who are in that set calculates gradient difference and update server. It has been shown that a significant communication reduction with numerical simulations compared to gradient descent.

Client scheduling based communication overhead reduction:

In [23], authors analyze another approach which is possible in order train a NN with FL under a resource limited and unreliable environment. Therein, due to limited resources,

only a portion of clients can be scheduled to update the model parameters in each communication round. In their work authors study the impact of conventional scheduling policies such as random (RAND), round robin (RR), and proportional fair (PF) on the accuracy of FL over wireless networks.

For the comparison, first authors of the paper develop an analytical model to characterize the performance of FL with scheduling in wireless networks assuming to have a homogeneous data distribution among the clients. Particularly, using that analytical model tractable expressions are derived which are applicable to analyze scheduling schemes and interferences. The assumption of this work is if the channel is having a low signal-to-interference-plus-noise ratio (SINR) then the update is dropped even if the client is scheduled for that communication round. Then aforementioned derived analytical model is used analyze the aforementioned scheduling policies. Authors of this work have shown that FL with PF outperforms RAND and RR with a high SINR threshold while under low SINR RR is more preferable. However, though they compare three conventional scheduling policies, an optimal scheduling policy is not derived in this work and further in this work a probabilistic distributions are used, particularly Rayleigh distribution instead of estimation or predictions.

Similar approach of scheduling in FL is drawn in [24] to overcome limitations in wireless resources. Therein, the training loss of FL is minimized at each model exchanging iteration. For this work aforementioned, clients-server conventional system model is considered. Since with transmitting all training parameters of FL over wireless links with resource constraints and with typical unreliability, the quality of the ML model trained will be affected. This effect due to the limitation of wireless resources, is minimized by the server per global communication round, by selecting an appropriate subset of clients for execution of FL model averaging. In this work this objective is achieved by formulating this as an optimization problem subjected to aforementioned resource limitations in wireless networks. The goal of the optimization problem is to minimize loss function as in [6] which is the representation of the quality of the model trained. Using this, a closed-form expression for the expected value of the convergence rate is derived. Then using that to analyze the impact of wireless resource conditions on FL, derived optimal transmit power allocation for clients under given resource block allocation for uplink and scheduling policy. Thereafter, scheduling of clients and uplink resource block allocation is optimized. However, in this work assumed to have a perfect CSI at the clients and server. Moreover, an optimal user scheduling policy for entire training period is not derived.

Quantization based communication overhead reduction:

A contrast framework to aforementioned methods is proposed in [25] by adopting idling, selecting, and quantizing based approach to increase communication efficiency in FL. Therein, they have reduced the communication overhead to tackle with communication bottleneck in a wireless environment. This proposed method is denoted as "FEDPAQ" in this work which means "Federated Learning method with Periodic Averaging and Quantization". In FedPAQ authors has proposed three main features in order to reduce communication overhead:

- Periodic averaging: In contrast with the conventional FL, where all clients synchronize their models through the server in each communication round, in this

work authors propose to store updates at clients and execute averaging at the server periodically. This approach is significant to reduce the communication cost but of course with the trade-off with the accuracy of the trained model. Consequently, this approach reduce the overall communication overhead in training model with FL.

- Selecting a sub set of clients for model update per communication round: Ideally as aforementioned, a wireless communication system is with resource limitations. Thus, only a portion of clients are able to simultaneously upload their model parameters in FL. This results in a dramatically slow training since aggregation process can only be initialized once after all parameters are updated. Therefore, in this work authors propose further to select only a portion of clients to update model parameters in FL in each communication round. Further, having all clients participating in the training process even introduces significant overhead eventhough given unconditional channel state. Thus this, intuitively reduce the overall communication overhead as well in FL model training. This selection criteria can be affected by several factors as proposed by authors. Accordingly, a selected client should be able to reach server without being thorny, should be plugged in, idle and connected to a free wireless network.

- Quantized model update: Most importantly, this is the contrast novel idea this work propose to reduce the communication overhead. That is to deploy quantization operator to model update of clients before uploading it to the server for aggregation in each communication round. This ideally reduce the communication overhead depending on the quantizer used with expense of the accuracy of the trained model accuracy. Furthermore, this approach reduces the time taken to upload model parameters in FL of clients.

Then authors of this work has proved the convergence guarantees of the proposed method, FedPAQ considering strongly-convex and non-convex losses. Finally, authours presents empirical results which demonstrates that the communication and computation tradeoff with proposed FedPAQ method.

Finally, it can be noted that the communication aspects in FL are neglected in the aforementioned works such as channel uncertainties due to issues discussed earlier. For instant, optimal client scheduling and resource allocation throughout the entire training duration, even under the absence of the perfect channel state information (CSI) is not addressed in the existing literature.

1.2.4 Stochastic optimization - Lyapunov optimization framework

If randomness is presented in the objective function or in constraints or in both, the problem becomes a stochastic optimization problem. The formal representation of a stochastic optimization problem is as follows:

$$\begin{aligned} \text{minimize} \quad & \bar{y_0} && \text{(3a)} \\ \text{subject to} \quad & \bar{y_i} \leq 0, \quad i \in \{1, 2, ..., K\} && \text{(3b)} \\ & \alpha(t) \in \mathcal{A}, && \text{(3c)} \end{aligned}$$

where $\alpha(t)$ is the optimal decision chosen from the feasible set of the decision variables $\mathcal{A}$, $y_0(t)$ is the time dependent objective function, $\bar{y_i}$ are continuous convex constraints.

Some of the state-of-the-art methods to solve such stochastic optimization problems are direct search methods, for instant Nelder-Mead method, but those methods are computationally exhaustive. The other methods in literature are namely stochastic approximation, stochastic programming. Lyanunov optimization framework proposed in [14] is a method to decouple and solve such stochastic optimization problem in the form (3). First, to track the dynamics of the time average constraints over the time in (3), auxiliary variables $\boldsymbol{\gamma}(t)$ are introduced to achieve $\bar{\boldsymbol{\gamma}} = \bar{\boldsymbol{y}}$. To track those time average conditions are met, a set of virtual queues are introduced and evolve as follows:

$$Q_i(t+1) = \max(Q_i(t) + \gamma_i(t) - y_i(t), 0) \quad \forall i. \tag{4}$$

Then, Lyapunov function is defined as,

$$L(t) = \frac{1}{2} \sum_i Q_i(t)^2, \tag{5}$$

where Q_i s are introduced queues. With the definition of (5), one slot Lyapunov drift $\Delta(t) = L(t+1) - L(t)$ is computed. Then, upper bound of the $\Delta(t)$ is derived. Let V be a parameter which controls the tradeoff between queue stability and optimality of the solution of (3). In [14], it has been proved that by minimizing the upper bound of drift-plus-penalty (DPP: $\Delta(t) + V y_0^{avg}$) expression on each slot t, yields $\bar{\boldsymbol{\gamma}} = \bar{\boldsymbol{y}}$ when $t \to \infty$, where $y_0^{avg} = \frac{1}{t} \sum_{\tau=1}^{t} y_0(t)$.

Applications of Lyapunov optimization:

In [26], authors have cast joint power allocation and user scheduling problem in ultra dense small networks (UDNs) to a dynamic stochastic game using mean-field game (MFG) theory. Then it is solved using the DPP approach in the Lyapunov optimization framework. Therein, the Lyapunov optimization framework allows to decouple master problem into several subproblems defined per base station (BS).

Authors of the paper [27], have used Lyapunov stochastic framework to derive an online control algorithm for data center devices to reduce time averaged electric utility consumption in a data center. The proposed algorithm operates without any knowledge in statistics of the workload or utility consumption for processes in datacenter.

1.2.5 Communication channel uncertainty:

Channel is a medium which is used to transmit signals from one place to another. There exist three key propagation phenomenon due to reflections, diffraction and scattering nature of the signal through a wireless over the air channel [13]. Namely path loss, shadow fading and multipath fading. Total attenuation is the overall effect from these three phenomenon. Attenuation due to path loss can be modelled using free space path loss model. Moreover, the effect of the attenuation due to aforementioned multipath phenomenon is approximated for simulations with models. Such models are,

- Rayleigh fading model using Rayleigh probability distribution: This distribution is achieved by taking the magnitude of a complex number with real and imaginary parts from two IID zero mean gaussian random variables (RV). Mathematically this probability distribution is given by, $\mathbb{P}(r) = \frac{2r}{\mathbb{E}(R^2)} \exp\left(\frac{-r^2}{\mathbb{E}(R^2)}\right)$, where R is the random variable. This distribution following envelope of channel response is ideal for the case where without line of sight (LoS) component present in transmission. Rayleigh fading model is a reasonable model when there are many objects causing scattering before signal reaches the receiver, in the channel path. Thus, now R is the absolute value of the channel response $|h(t)|$. There power is exponentially distributed in the distribution while phase is uniformly distributed.

- Clarke-Gans fading model: In this model main assumption is an isotropic scattering in a Rayleigh distributed fading model [28]. Gans have developed a model including Doppler effect by a Doppler filter [29]. Using this model, correlated Rayleigh distributed channels can be simulated with appropriate Doppler frequency values. Clarke and Gans's fading modelling is represented in following Fig. 5,

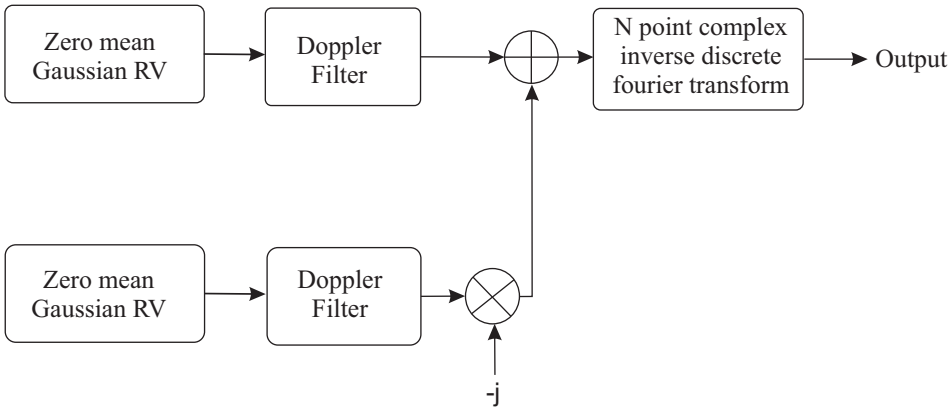

Figure 5. Generating Clarks Gans faded model

Another popular method of fading model distribution called Rician distribution. This model is ideal for modelling the envelope of the channel response of a channel having LoS conditions. Here LoS component is the dominant. Nakagami model is a generalized version of Rayleigh distribution [30]. There are more other statistical and empirical models for modelling fading effect in different environments in literature.

1.2.6 Gaussian process regression (GPR) for predictions

Regression means estimating an unknown function with given a set of inputs and observations. GPR got the attention of machine learning applications since this is very light weight learning method which allows to learn with covariance also denoted as kernel machines using probabilistic approach. Suppose there is a unknown, untractable function g, with output $y(t) = g(t, x(t))$ [31]. If g is known to be gaussian process (GP) it's output at time t $(y(t))$ can be estimated via regression. In general, Gaussian process regression is a method to interpolate data points generated from an unknown function. Therein, this predicted values are modelled by a Gaussian process with a prior covariances.

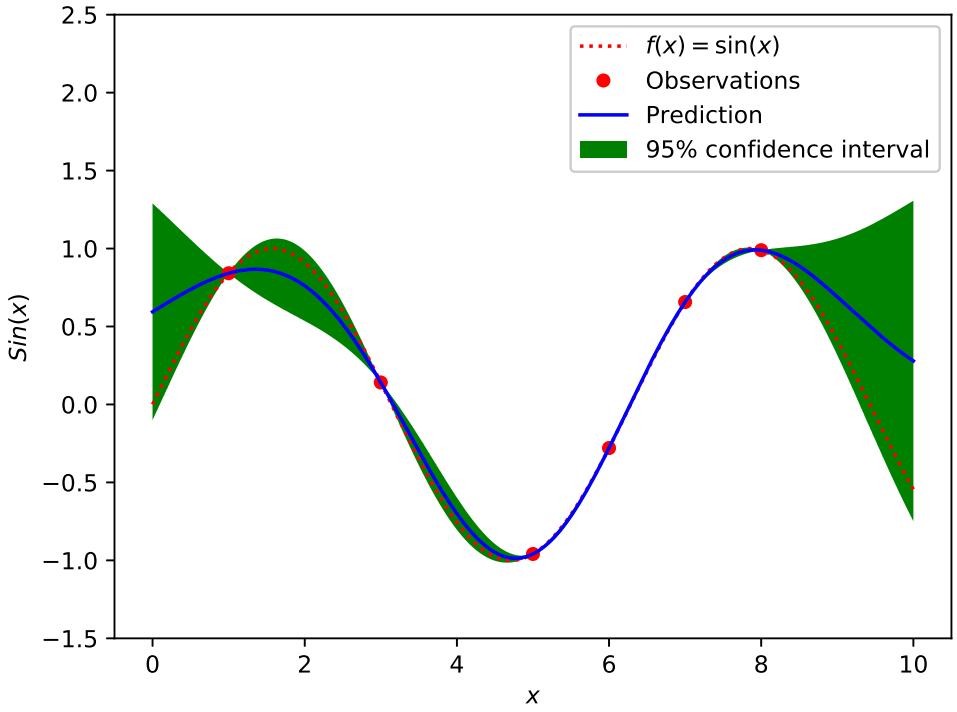

Figure 6. Prediction of $\sin x$ function using GPR.

In Fig. 6, shows how GPR used to predicting the process of $\sin x$ with 6 observation points. There, it is clearly visible that confidence interval is less in intermediate observation points compared to beyond of observation points. However, the GP model has captured the pattern of function $\sin(x)$ with observation points. Further, the data points "anchor" the predicted function at locations.

The work of [32] provides comprehensive theoretical aspects of GPs in machine learning. Further, a wide variety of kernel functions are explained with their properties. Typically, an unknown deterministic or stochastic function guessing is uncertain due to lack of data points and noise in measurements.

In the context of modelling dynamics of an unknown function basing on [33] book, provides a comprehensive theoretical aspects. Simply, with the data available in GPR, build a model that match to the unknown function from which the data used for training or fitting are generated. However, the model structure is not specifically defined a priori as in neural networks, but is developed from the data. After learning or fitting, using that model make predictions with inference for any input and outputs are real-valued. In GP, the Bayesian framework allows to handle aforementioned aspect, with probability distribution with a collection of random variables, which are from a joint Gaussian distribution as below,

$$g(\boldsymbol{x}) = \mathcal{GP}(m(\boldsymbol{x}), K(\boldsymbol{x}, \boldsymbol{x}')) \tag{6}$$

where $m(\boldsymbol{x})$ is the mean function, and $K(\boldsymbol{x}, \boldsymbol{x}')$ is the covariance function. With aforementioned two parameters a GP can be fully specified. Covariance function defines correlations between observed data points in the process

$$K(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}\Big(g(\boldsymbol{x})g(\boldsymbol{x}')\Big) \tag{7}$$

This covariance function must produce a positive semi definite covariance matrix, since in the prediction its necessary to take the inverse of this covariance matrix. As mentioned before, there exists several such defined kernel functions.

- Squared exponential (SE) kernel: An element in K is calculated within this kernel function as below,

$$k(x_i, x_j) = \alpha^2 \exp\left(-\frac{1}{2}\left(\frac{(x_i - x_j)}{\lambda}\right)^2\right) \tag{8}$$

  where λ and α are hyper parameters which determines length scale and amplitude respectively. This kernel is very smooth. Thus, infinitely differentiable. Significance of this kernel function is that in this function, the variables close to input space are highly correlated, while far away are uncorrelated.

- Periodic or sin-squared kernel: An element in K is calculated within this kernel function as below,

$$k(x_i, x_j) = \exp\left(-\frac{2\sin^2\left(\frac{x_i - x_j}{2}\right)}{\lambda^2}\right) \tag{9}$$

  where λ is a hyper parameter which determines periodicity. A scaling factor is added to the beginning if required.

There are several other kernel functions such as Matern class of kernel, linear kernel, etc. All kernel functions depicts different properties capturing the features of the data points.

After fitting the model, then for single point (x_*) predictions, a posteriori is a simple Gaussian.

$$p(g(x_*)|y) = \mathcal{N}(m, \sigma^2) \tag{10}$$

where, $m = k_*^T K^{-1} y$ and $\sigma^2 = k(x_*, x_*) - \boldsymbol{k_*}^T K^{-1} \boldsymbol{k_*}$. There $(k_*)_i = k(x_*, x_i)$. The same procedure for multiple point predictions but the posterior defined over functions is a Gaussian process. However, with number of posterior, the GPR complexity is in $\mathcal{O}(n^2)$.

Applications of GPR:

Applications of GPR is spreaded over vast field such as machine learning, networking, wireless communication, control, etc since GPR is a competitive light weight tool for active learning. In wireless communication channel is uncertain as mentioned in section 1.2.5. Entropy based approach for wireless scheduling using GPR to predict the channel is presented in [34]. Here in this paper also authors have used GPR for active learning and used it's aforementioned advantage of explicit quantifying ability of information content. Exploration and exploitation based objective is optimized for wireless scheduling using the information content of GPR.

In [35], the author has used GPR for control applications. There, he has proposed a method quantifying the information acquired using entropy measure from information theory for each controlling step. Thereafter, aforementioned information collection is combined with the control objectives to cast a multi objective optimization problem. The proposed method is a dual approach for active learning and control scheme. Gaussian processes (GP) are used for regression of the state space equations of discrete non-linear systems. More importantly, the GPR framework gives added advantage to explicitly quantifying of information content.

2 SYSTEM MODEL AND PROBLEM FORMULATION

Consider a system consisting a set $\mathcal{K}$ of K clients that communicate with a server over wireless environment. Therein, the k-th client has a private dataset $\mathcal{D}_k$ of size D_k, which is a partition of the global dataset $\mathcal{D}$ of size $D = \sum_k D_k$. A set $\mathcal{B}$ of $B(\leq K)$ resource blocks are shared among the clients when communicating with the server.

Let $s_k(t) \in \{0, 1\}$ be an indicator where $s_k(t) = 1$ indicates that the client k is scheduled by the server for uplink communication at time t and $s_k(t) = 0$ otherwise. To schedule several clients simultaneously, one RB is allocated to each scheduled client. Hence, we define the RB allocation vector $\boldsymbol{\lambda}_k(t) = [\lambda_{k,b}(t)]_{\forall b \in \mathcal{B}}$ for client k with $\lambda_{k,b}(t) = 1$ when RB b is allocated to client k at time t, and $\lambda_{k,b}(t) = 0$ otherwise. The client scheduling and RB allocation are constrained as follows:

$$s_k(t) \leq \mathbf{1}^\dagger \boldsymbol{\lambda}_k(t) \leq 1 \quad \forall k, t. \tag{11}$$

The rate at which the k-th client communicates with the server at time t is given by,

$$r_k(t) = \sum_{b \in \mathcal{B}} \lambda_{k,b}(t) \log_2 \Big(1 + \frac{p|h_{k,b}(t)|^2}{I_{k,b}(t) + N_0}\Big), \tag{12}$$

where p is a fixed transmit power of client k, $h_{k,b}(t)$ is the channel between client k and the server over RB b at time t, $I_{k,b}(t)$ represents the uplink interference on client k from other client over RB b, and N_0 is the noise power spectral density.

Under imperfect CSI, the channels need to be estimated via sampling prior to transmission. The channel sampling data at time t is collected per RB allocation over the transmissions throughout $\{1, \ldots, t-1\}$, then the future channel is inferred using the past observations as $\hat{h}(t) = J\Big(t, \{t_n, h(t_n)\}_{n \in \mathcal{N}(t)}\Big)$. Here, t_n is a sampling time instant and the set $\mathcal{N}(t)$ consists of sampling indices until time t, i.e., $n \in \mathcal{N}(t)$ is held only if $s(t_n) = 1$ and $t_n < t$. With the estimated channels, a successful communication between a scheduled client and the server is defined by satisfying a target minimum rate. Therefore, according to (11), the rate constraint can be imposed per RB allocation in terms of a target signal to interference plus noise ratio (SINR) γ_0 as follows:

$$\lambda_{k,b}(t) \leq \mathbb{I}\Big(\hat{\gamma}_{k,b}(t) \geq \gamma_0\Big) \quad \forall k, b, t, \tag{13}$$

where $\hat{\gamma}_{k,b}(t) = \frac{p|\hat{h}_{k,b}(t)|^2}{I_{k,b}(t)+N_0}$ and the indicator $\mathbb{I}(\hat{\gamma} \geq \gamma_0) = 1$ if $\hat{\gamma} \geq \gamma_0$, $\mathbb{I}(\hat{\gamma} \geq \gamma_0) = 0$ otherwise.

The aim of model training is to minimize a regularized loss function,

$$F(\boldsymbol{w}, \mathcal{D}) = \frac{1}{D} \sum_{\boldsymbol{x}_i \in \mathcal{D}} f(\boldsymbol{x}_i^\dagger \boldsymbol{w}) + \xi \varrho(\boldsymbol{w}) \tag{14}$$

by fitting a weight vector $\boldsymbol{w}$ that is known as the *model* over the global dataset $\mathcal{D}$, within a predefined communication duration T. Here, $f(\cdot)$, $\varrho(\cdot)$, and ξ are the loss function, the regularization function, and the regularization coefficient, respectively. Due to the limitations of communication and privacy, we adopt FL as model training technique to derive the optimal weights that minimize $F(\boldsymbol{w}, \mathcal{D})$. As said, in FL each client computes a local model over its local dataset and shares the local model with the server. Upon

receiving the local models from all clients, the server does model averaging, calculates the global model, which is broadcasted to all clients.

Under imperfect CSI, channels between clients and the server over each RB are predicted using their past observations prior to the communication. With $\lambda_{k,b}(t) = 1$, the channel $h_{k,b}(t)$ is sampled and used as an observation in the future. It means that the RB allocation and channel sampling are carried out simultaneously. In this regard, we define the information on the channel between client k and server at time t that can be obtained by RB allocation as $\boldsymbol{j}_k(t) = [j_{k,b}(t)]_{k\in\mathcal{K}}$. For accurate CSI predictions, it is essential to acquire as much information about the CSI over the network [36]. In this view, we maximize $\sum_k \boldsymbol{j}_k^{\dagger}(t)\boldsymbol{\lambda}_k(t)$ at each t while minimizing the loss $F(\boldsymbol{w}, \mathcal{D})$. This iterative process is carried out over a training duration of T as illustrated in Fig. 7.

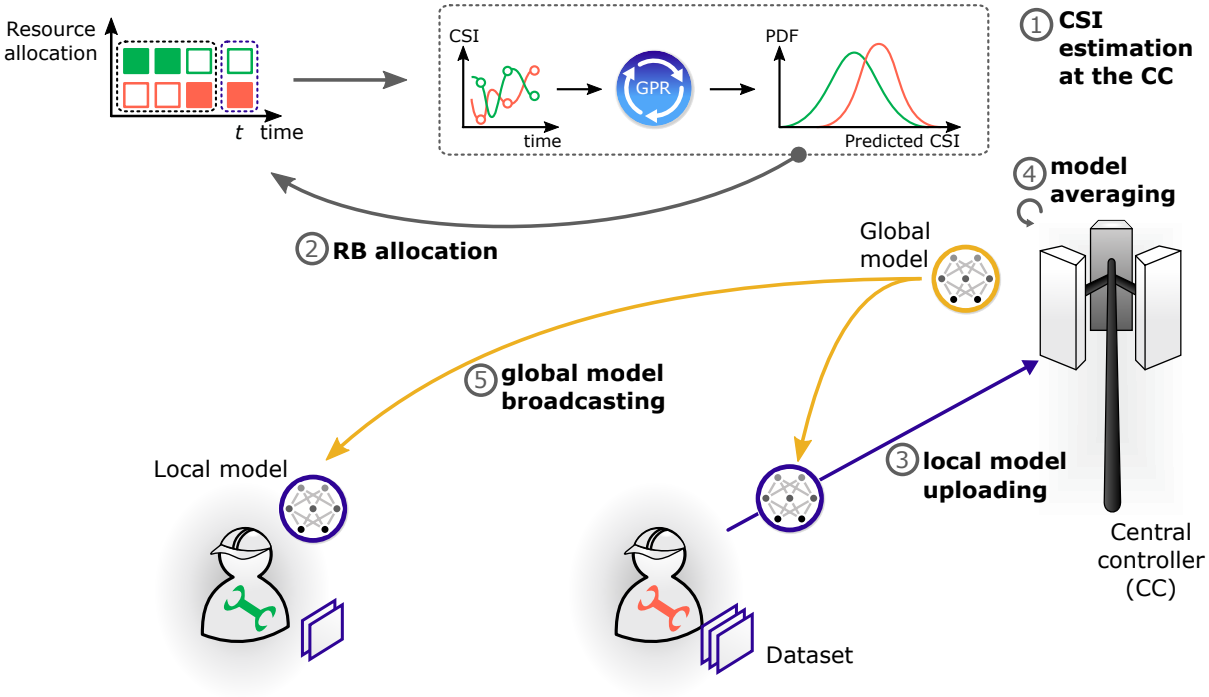

Figure 7. FL with client scheduling under limited wireless resources and imperfect CSI.

The empirical loss minimization problem for all $t \in \{1, \ldots, T\}$ is formally defined as follows:

$$\underset{\boldsymbol{w}(t),\boldsymbol{s}(t),\boldsymbol{\Lambda}(t),\forall t}{\text{minimize}} \; F\Big(\boldsymbol{w}(T), \mathcal{D}\Big) - \tfrac{\varphi}{T} \textstyle\sum_{k,t} \boldsymbol{j}_k^{\dagger}(t)\boldsymbol{\lambda}_k(t) \tag{15a}$$

$$\text{subject to} \quad (11)\text{-}(13), \tag{15b}$$

$$\boldsymbol{A}\boldsymbol{\Lambda}^{\dagger}(t) \preceq \mathbf{1}, \tag{15c}$$

$$\mathbf{1}^{\dagger}\boldsymbol{s}(t) \leq B, \tag{15d}$$

$$\boldsymbol{s}(t) \in \{0,1\}^K, \boldsymbol{\lambda}_k(t) \in \{0,1\}^b, \tag{15e}$$

$$\boldsymbol{w}_k(t) = \underset{\boldsymbol{w}'}{\text{argmin}}\, F(\boldsymbol{w}', \boldsymbol{w}(t-1), \mathcal{D}_k), \tag{15f}$$

$$\boldsymbol{w}(t) = \textstyle\sum_k \frac{D_k}{D} s_k(t)\boldsymbol{w}_k(t), \tag{15g}$$

where $\mathbf{\Lambda}^{\dagger}(t) = [\boldsymbol{\lambda}_k^{\dagger}(t)]_{k\in\mathcal{K}}$, $\varphi(> 0)$ is the parameter which controls the impact of the information exploration, and $\boldsymbol{A}$ is a $B \times K$ all-one matrix. The orthogonal channel allocation in (15c) ensures collision-free client uplink transmission with $I_{k,b}(t) = 0$ and constraint (15d) defines the maximum allowable clients to be scheduled due to the limitation in the RB availability. The stochastic gradient decent (SGD) based local model calculation at client k is defined in (15f). The global model update carried out according to (15g) depending on the dataset size of each scheduled client.

3 OPTIMAL CLIENT-SCHEDULING AND RB ALLOCATION POLICY VIA LYAPUNOV OPTIMIZATION

It can be noted that the optimization problem (15) is coupled over all clients. Hence, in what follows, the discussion of decoupling (15) over clients and the server, and then deriving the optimal client scheduling and RB allocation policy.

3.1 Decoupling of original problem via dual updates

Let us consider an ideal unconstrained scenario where the server gathers the entire data samples and trains the global model in a *centralized* manner. Let $F_0 = \min_{\boldsymbol{w}} F(\boldsymbol{w}, \mathcal{D})$ be the minimum loss under centralized training. By the end of training duration T, we define the gap between the studied FL under communication constraints and centralized training as $\varepsilon(T) = F\Big(\boldsymbol{w}(T), \mathcal{D}\Big) - F_0$. Here, $\varepsilon(T)$ is the loss of FL with scheduling compared to centralized training. Note that minimizing (15a) remains unchanged by minimizing the gap $\varepsilon(T)$ under the same set of constraints.

To analyse the loss of FL with scheduling, we consider the dual function of (15a) with the dual variable $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_D]$ and $\boldsymbol{X} = [\boldsymbol{X}_k]_{k\in\mathcal{K}}$ with $\boldsymbol{X}_k = [\boldsymbol{x}_i]_{i=1}^{D_k}$ as follows:

$$\begin{aligned}\psi(\boldsymbol{\theta}) &= \min_{\boldsymbol{w},\boldsymbol{z}} \Big(\sum_{\boldsymbol{x}_i\in\mathcal{D}} \frac{1}{D} f_i(\boldsymbol{x}_i^T\boldsymbol{w}) + \xi\varrho(\boldsymbol{w}) + \frac{\boldsymbol{\theta}^T(\boldsymbol{\theta}-\boldsymbol{z})}{D}\Big) \\ &= -\xi\varrho^*(\frac{\boldsymbol{X}\boldsymbol{\theta}}{\xi D}) - \sum_{i=1}^{D} \frac{f_i^*(-\theta_i)}{D} \\ &= -\sum_{k=1}^{K}\sum_{i=1}^{D_k} \frac{1}{D} f_i^*(-\theta_i) - \xi\varrho^*(\boldsymbol{v}), \end{aligned} \tag{16}$$

where $\boldsymbol{v} = \boldsymbol{X}\boldsymbol{\theta}/\xi D$, $\boldsymbol{z} = \boldsymbol{X}^T\boldsymbol{w}$ is a newly introduced variable, and $f^*(.)$, $\varrho^*(.)$ are the conjugate functions of $f(.)$ and $\varrho(.)$, respectively. With the dual formulation the relation between the primal and dual variables is $\boldsymbol{w} = \nabla\varrho^*(\boldsymbol{v})$ [23]. Based on the dual formulation, the loss of FL with scheduling is $\varepsilon(T) = \psi_0 - \psi\Big(\boldsymbol{\theta}(T)\Big)$ where ψ_0 is the maximum dual function value obtained from the centralized method.

Note that the first term of (16) decouples per client and thus, can be computed locally. In contrast, the second term in (16) cannot be decoupled per client. To compute $\varrho^*(\boldsymbol{v})$, first, each client k locally computes $\Delta\boldsymbol{v}_k(t) = \frac{1}{\xi D}\boldsymbol{X}_k\Delta\boldsymbol{\theta}_k(t)$ at time t. Here, $\Delta\boldsymbol{\theta}_k(t)$ is the change in dual variable $\boldsymbol{\theta}_k(t)$ for client k in the time t given as below,

$$\begin{aligned}\Delta\boldsymbol{\theta}_k(t) \approx \underset{\boldsymbol{\delta}\in\mathbb{R}^{D_k}}{\operatorname{argmax}} \Big(&-\frac{1}{D}\mathbf{1}^\dagger[f_i^*(-\boldsymbol{\theta}_k(t)-\boldsymbol{\delta})]_{i=1}^{D_k} - \frac{\xi}{K}\varrho^*\Big(\boldsymbol{v}(t)\Big) - \frac{1}{D}\boldsymbol{\delta}^\dagger\boldsymbol{X}_k\varrho^*\Big(\boldsymbol{v}(t)\Big) \\ &- \frac{\eta/\xi}{2D^2}\|\boldsymbol{X}_k\boldsymbol{\delta}\|^2\Big), \end{aligned} \tag{17}$$

where η depends on the partitioning of the $\mathcal{D}$ [37]. It is worth noting that $\Delta\boldsymbol{\theta}_k(t)$ in (17) is computed based on the previous global value $\boldsymbol{v}(t)$ received by the server. Then, the

scheduled clients upload $(\Delta \boldsymbol{v}_k(t), \Delta \boldsymbol{\theta}_k(t))$ to the server. Following the dual formulation, the model aggregation and update in (15g) at the server is modified as follows:

$$\boldsymbol{v}(t+1) := \boldsymbol{v}(t) + \sum_{k \in \mathcal{K}} s_k(t) \Delta \boldsymbol{v}_k(t), \tag{18a}$$

$$\boldsymbol{\theta}_k(t+1) := \boldsymbol{\theta}_k(t) + \sum_{k \in \mathcal{K}} \frac{1}{K} s_k(t) \Delta \boldsymbol{\theta}_k(t). \tag{18b}$$

Using (18a), the server computes the coupled term $\varrho^*\Big(\boldsymbol{v}(t+1)\Big)$ in (16). The decoupling of the original problem (15) using dual formulation is presented in the following figure from a schematic diagram,

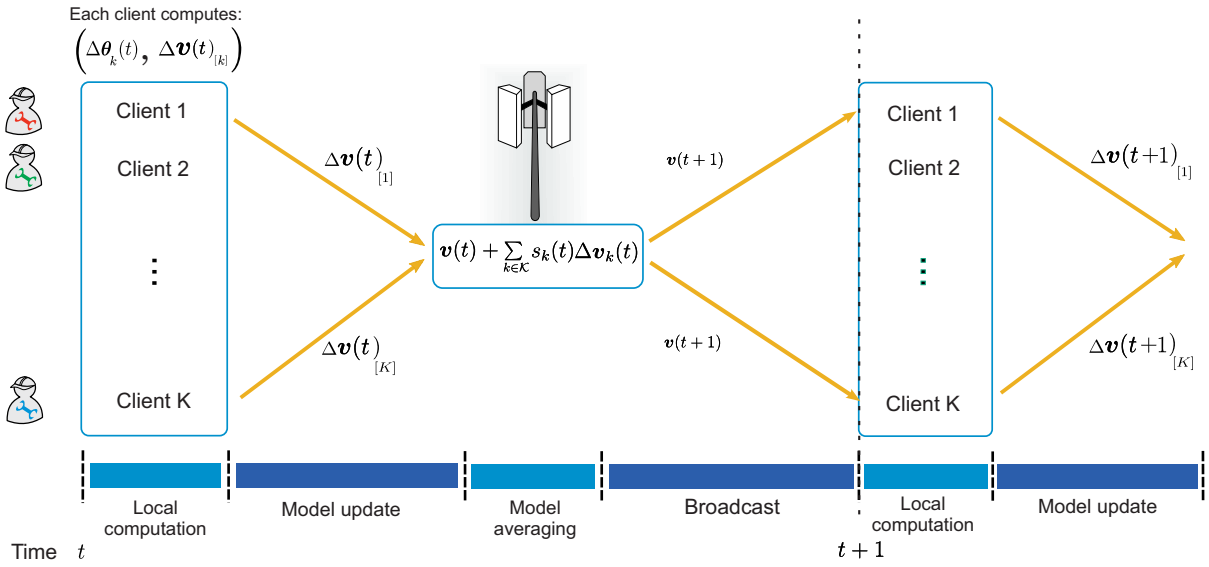

Figure 8. Model update representation using dual formulation.

Note that from the t-th update, $\Delta \boldsymbol{\theta}_k(t)$ in (17) maximizes $\Delta\psi\Big(\boldsymbol{\theta}_k(t)\Big)$, which is the change in dual function $\psi\Big(\boldsymbol{\theta}(t)\Big)$ corresponding to the of client k. Let $\boldsymbol{\theta}_k^\star(t)$ be the local optimal dual variable at time t, in which $\Delta\psi\Big(\boldsymbol{\theta}_k^\star(t)\Big) \geq \Delta\psi\Big(\boldsymbol{\theta}_k(t)\Big)$ is held. Then for a given accuracy $\beta_k(t) \in (0,1)$ of local SGD updates, the following condition is satisfied:

$$\frac{\Delta\psi_k\Big(\Delta\boldsymbol{\theta}_k^\star(t)\Big) - \Delta\psi_k\Big(\Delta\boldsymbol{\theta}_k(t)\Big)}{\Delta\psi_k\Big(\Delta\boldsymbol{\theta}_k(t)\Big) - \Delta\psi_k(0)} \leq \beta_k(t), \tag{19}$$

where $\Delta\psi_k(0)$ is the change in ψ with a null update from k-th client. For simplicity, we assume that $\beta_{k,t} = \beta$ for all $k \in \mathcal{K}$ and t, hereinafter. With (19), the gap between FL with scheduling and the centralized method after T number of communication rounds is $\varepsilon(T)$.

Theorem 1 (Upper bound of $\varepsilon(T)$)**.** *The upper bound of the $\varepsilon(T)$ after T number of communication rounds is given as,*

$$\varepsilon(T) \leq D\Big(1 - (1-\beta) \sum \sum_{t \leq T}^{k \leq K} \frac{D_k}{TD} s_k(t)\Big)^T.$$

For the proof of 1 see [Appendix 7.1]

This yields that the minimization of $\varepsilon(T)$ can be achieved by minimizing its upper bound defined in (1). Henceforth, the equivalent form of (15) is given as follows:

$$\underset{[\Delta\boldsymbol{\theta}_k(t)]_k, \boldsymbol{s}(t), \boldsymbol{\Lambda}(t), \forall t}{\text{minimize}} \; D\Big(1-(1-\beta)\sum_{t,k}\frac{D_k}{TD}s_k(t)\Big)^T - \frac{\varphi}{T}\sum_{k,t}\boldsymbol{j}_k^\dagger(t)\boldsymbol{\lambda}_k(t) \tag{20a}$$

$$\text{subject to} \quad \text{(15b)-(15e)}, (17), (18). \tag{20b}$$

3.2 GPR-based metric for information on unexplored CSI

For CSI predictions, we use GPR with a Gaussian kernel function to estimate the nonlinear relation of $J(\cdot)$ by assuming that it follows a Gaussian process (GP) as a prior.

In this view, for a finite data set $\{t_n, h(t_n)\}_{n\in\mathcal{N}}$, the aforementioned GP becomes a multi-dimensional Gaussian distribution, with a zero mean and covariance $\boldsymbol{C} = [c(t_m, t_n)]_{m,n\in\mathcal{N}}$ given by,

$$c(t_m, t_n) = \exp\Big(-\tfrac{1}{\zeta_1}\sin^2\Big(\tfrac{\pi}{\zeta_2}(t_m - t_n)\Big)\Big), \tag{21}$$

where ζ_1 and ζ_2 are the length and period hyper-parameters, respectively [38]. Henceforth, the CSI prediction at time t and its uncertainty/variance is given by [32],

$$\hat{h}(t) = c^\dagger(t)\boldsymbol{C}^{-1}[h(t_n)]_{n\in\mathcal{N}}, \tag{22}$$

$$j(t) = c(t,t) - c^\dagger(t)\boldsymbol{C}^{-1}c(t), \tag{23}$$

where $c(t) = [c(t, t_n)]_{n\in\mathcal{N}}$. Note that the client and RB dependence is omitted in the discussion above for notation simplicity. Here, the channel estimation is given in (22).

Theorem 2 (Information content of GPR)**.** *The CSI uncertainty in GPR prediction is used as the information content $j(t)$*

For the proof, see [Appendix 7.2]

With Theorem 2, CSI uncertainty is used as the information $j(t)$, in which exploring highly uncertain channels provides more insight. It is worth noting that under *perfect CSI* $\hat{h}(t) = h(t)$ and $j(t) = 0$ since there is no information content with a zero probable quantity, according to information theory.

3.2.1 Joint client scheduling and RB allocation

Due to the time average objective in (20a), the problem (20) becomes a stochastic optimization problem defined over $t = \{1, \ldots, T\}$. Therefore, we resort to the *drift plus penalty* (DPP) technique in Lyapunov optimization framework to derive the optimal scheduling policy [14]. Therein, Lyapunov framework allows us to transform the original stochastic optimization problem into a series of optimizations problems that are solved at each time t, as discussed next.

First, we denote $u(t) = (1-\beta)\sum_k s_k(t)D_k/D$ and define its time average $\bar{u} = \sum_{t\leq T} u(t)/T$. Then, we introduce auxiliary variables $\nu(t)$ and $l(t)$ with time average

lower bounds $\bar{\nu} \leq \bar{u}$ and $\bar{l} \leq \frac{1}{T}\sum_{k,t} \boldsymbol{j}_k^\dagger(t)\boldsymbol{\lambda}_k(t) \leq l_0$, respectively. To track the time average lower bounds, next we introduce virtual queues $q(t)$ and $g(t)$ with the following dynamics [14]:

$$q(t+1) = \max\Big(0, q(t) + \nu(t) - u(t)\Big), \tag{24a}$$

$$g(t+1) = \max\Big(0, g(t) + l(t) - \textstyle\sum_k \boldsymbol{j}_k^\dagger(t)\boldsymbol{\lambda}_k(t)\Big). \tag{24b}$$

In this view, (20) can be recast as follows:

$$\underset{[\Delta\boldsymbol{\theta}_k(t)]_k, \boldsymbol{s}(t), \boldsymbol{\Lambda}(t), \nu(t), l(t) \forall t}{\text{minimize}} \quad D(1-\bar{\nu})^T - \varphi\bar{l} \tag{25a}$$

$$\text{subject to} \quad (20\text{b}), (24), \tag{25b}$$

$$0 \leq \nu(t) \leq 1-\beta \quad \forall t, \tag{25c}$$

$$0 \leq l(t) \leq l_0 \quad \forall t, \tag{25d}$$

$$u(t) = \sum_k \frac{(1-\beta)D_k}{D} s_k(t) \quad \forall t. \tag{25e}$$

The quadratic Lyapunov function of $q(t)$ is $L(t) = \Big(q(t)^2 + g(t)^2\Big)/2$. Given $\Big(q(t), g(t)\Big)$, the expected conditional Lyapunov one slot drift at time t is $\Delta L = \mathbb{E}[L(t+1) - L(t)|q(t), g(t)]$. Weighted by a tradeoff parameter $\phi(\geq 0)$, we add a penalty term,

$$\begin{aligned}\phi\Big(\tfrac{\partial}{\partial\nu}[(1-\nu)^T D]_{\nu=\tilde{\nu}(t)}\mathbb{E}[\nu(t)|q(t)] - \varphi\mathbb{E}[l(t)|g(t)]\Big) = \\ - \phi\Big(DT\Big(1-\tilde{\nu}(t)\Big)^{T-1}\mathbb{E}[\nu(t)|q(t)] + \varphi\mathbb{E}[l(t)|g(t)]\Big),\end{aligned} \tag{26}$$

to obtain the Lyapunov DPP. Here, $\tilde{\nu}(t) = \frac{1}{t}\sum_{\tau=1}^t \nu(\tau)$ and $\tilde{l}(t) = \frac{1}{t}\sum_{\tau=1}^t l(\tau)$ are the running time average of the auxiliary variables at time t. Using the inequality $\max(0,x)^2 \leq x^2$, the upper bound of the Lyapunov DPP is given by,

$$\begin{aligned}\Delta L - \phi\Big(DT\Big(1-\tilde{\nu}(t)\Big)^{T-1}\mathbb{E}[\nu(t)|q(t)] + \varphi\mathbb{E}[l(t)|g(t)]\Big) \leq \\ \mathbb{E}[q(t)\Big(\nu(t) - u(t)\Big) + g(t)\Big(l(t) - \sum_k \boldsymbol{j}_k^\dagger(t)\boldsymbol{\lambda}_k(t)\Big) + L_0 \\ - \phi\Big(DT\Big(1-\tilde{\nu}(t)\Big)^{T-1}\nu(t) + \varphi l(t)\Big)|q(t), g(t)],\end{aligned} \tag{27}$$

where L_0 is a uniform bound on $\Big(\nu(t) - u(t)\Big)^2/2 + \Big(l(t) - \sum_k \boldsymbol{j}_k^\dagger(t)\boldsymbol{\lambda}_k(t)\Big)^2/2$ for all t. The motivation behind deriving the Lyapunov DPP is that minimizing the upper bound of the expected conditional Lyapunov DPP at each iteration t with a predefined ϕ yields the tradeoff between the virtual queue stability and the optimality of the solution for (25) [14]. In this regard, the stochastic optimization problem of (25) is solved via minimizing the upper bound in (27) at each time t as follows:

$$\begin{aligned}\underset{\boldsymbol{s}(t), \boldsymbol{\Lambda}(t), \nu(t), l(t)}{\text{maximize}} \quad & \sum_k \Big(\frac{q(t)(1-\beta)D_k}{D} s_k(t) + g(t)\boldsymbol{j}_k^\dagger(t)\boldsymbol{\lambda}_k(t)\Big) \\ & - \alpha(t)\nu(t) - \Big(g(t) - \phi\varphi\Big)l(t)\end{aligned} \tag{28a}$$

$$\text{subject to} \quad (15\text{b})\text{-}(15\text{d}), (25\text{c}), (25\text{d}), \tag{28b}$$

$$\boldsymbol{0} \preceq \boldsymbol{s}(t), \boldsymbol{\lambda}_k(t) \preceq \boldsymbol{1}, \tag{28c}$$

where $\alpha(t) = q(t) - \phi DT\Big(1-\tilde{\nu}(t)\Big)^{T-1}$. Note that the constant κ is removed, the variable $\Delta\boldsymbol{\theta}_k(t)$ with constraints (17) and (18) are decoupled from (28), and the Boolean variables are relaxed as linear variables. Here, the objective and the constraints in (28) are affine, and the problem is a linear program (LP). Due to the independence, the optimal auxiliary variables are derived by decoupling the (28a), (25c), and (25d) as follows:

$$\nu^\star(t) = \begin{cases} 1-\beta & \text{if } \alpha(t) \geq 0, \\ 0 & \text{otherwise,} \end{cases} \quad l^\star(t) = \begin{cases} l_0 & \text{if } g(t) \geq \phi\varphi, \\ 0 & \text{otherwise.} \end{cases} \tag{29}$$

The optimal scheduling $\boldsymbol{s}^\star(t)$ and RB allocation variables $\boldsymbol{\Lambda}^\star(t)$ are found using an interior point method (IPM). It is due to the nature of LP, the optimal solution of the relaxed problem lies on a vertex of the feasible convex hull yielding the optimal solution for the problem with the Boolean variables. The joint client scheduling and RB allocation is summarized in Algorithm 1.

Algorithm 1 Joint Client Scheduling and RB Allocation

Input: $\mathcal{D}, \gamma_0, \beta, p, B, \xi$
Output: $\boldsymbol{s}^\star(t), \boldsymbol{\Lambda}^\star(t)$ for all t
$q(0) = g(0) = 0$, $\nu(0) = l(0) = 0$, $\boldsymbol{v}(0) = \mathbf{0}$
for $t = 1$ to T **do**
  Each client computes $\Delta\boldsymbol{\theta}_k(t)$ using (17)
  Channel prediction with (22)
  Calculate $\nu^\star(t)$ and $l^\star(t)$ using (29)
  Derive $\boldsymbol{s}^\star(t)$ and $\boldsymbol{\Lambda}^\star(t)$ by solving (28) using an IPM
  Local model $(\Delta\boldsymbol{v}_k(t), \Delta\boldsymbol{\theta}_k(t))$ uploading to the server
  Update $\tilde{\nu}(t)$, $q(t)$ via (24), $\boldsymbol{v}(t)$ and $\boldsymbol{\theta}(t)$ with (18)
  Global model $\boldsymbol{v}(t)$ broadcasting
  $t \to t+1$
end for

4 SIMULATION RESULTS

In this section, we evaluate the proposed client scheduling method using MNIST dataset assuming $f(\cdot)$ and $\varrho(\cdot)$ as cross entropy loss functions with a Tikhonov regularizer. For the training loss function, $\varepsilon_{\text{method}} = \text{accuracy}_{\text{centralized}} - \text{accuracy}_{\text{method}}$ is used. Here, the *centralized training* refers to the training takes place at the server with the access to the entire dataset. A dataset of 6000 samples consisting of equal sizes of ten classes for 0-9 digits are used over $K = 10$ clients. In addition, the uplink transmission power is set to $p = 1\,\text{W}$ and the channel follows a correlated Rayleigh distribution with mean to noise ratio equal to γ_0. For perfect CSI, it is assumed that a single RB is dedicated for channel measurements. The remaining parameters are presented in Table 2.

Table 2. Simulation parameters

Parameter	Value	Parameter	Value	Parameter	Value
γ_0	1.2	B	6	β	0.7
ζ_1	2	ϕ	1	ξ	1
ζ_2	5	η	0.2	$\lvert\mathcal{N}\rvert$	20
T	100	φ	1	p	1

4.1 Distribution of dataset among clients

To partition the training dataset over clients, we use the Zipf distribution to determine the local training dataset size. In other words client k owns a dataset with the following distribution of zipf.

$$D_k = \frac{Dk^{-\sigma}}{\sum_{\kappa\in\mathcal{K}} \kappa^{-\sigma}} \tag{30}$$

Here, the Zipf's parameter $\sigma = 0$ yields uniform data distribution over clients (600 samples per client), and increasing σ results in heterogeneous sample sizes among clients. It can clearly be noted how the dataset is distributed among clients with Zipf σ parameter from the following figure:

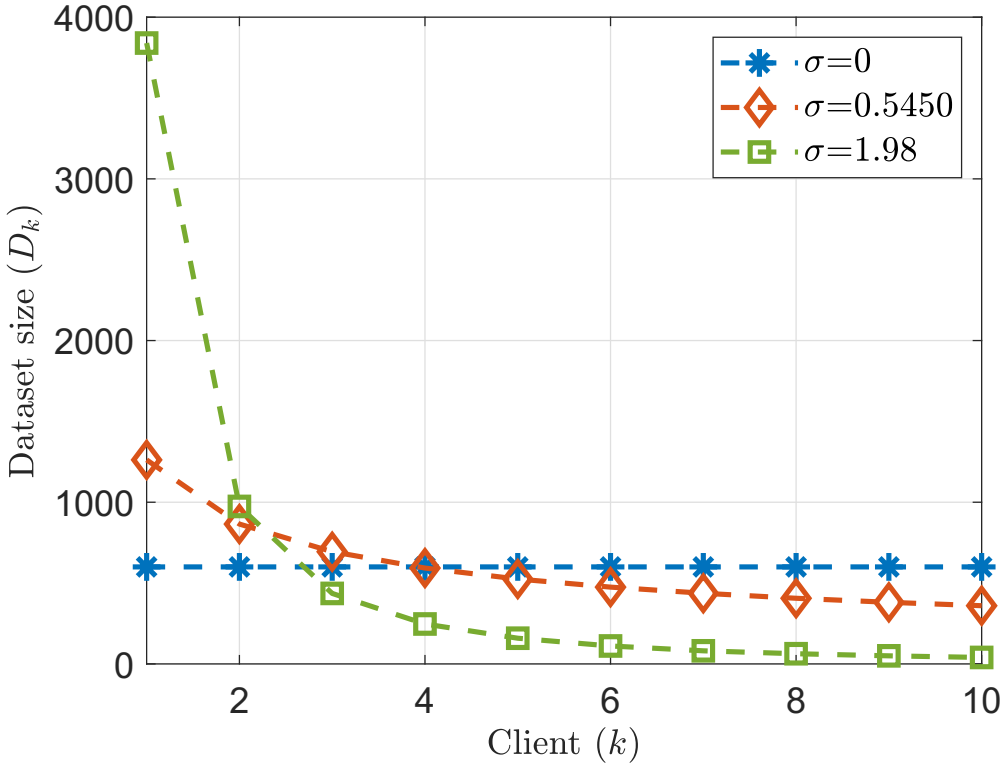

Figure 9. Zipf data distribution among clients with zipf's parameter σ, for $D = 6000$

Under perfect CSI, we denote the proposed scheduling method as data quantity aware scheduling policy "QAW". By deriving the optimal client scheduling and RB allocation based on the findings in [23], we obtain the quantity unaware baseline "QUNAW". Under imperfect CSI, the proposed GPR-based channel prediction and client scheduling method is coined as "QAW-GPR". Whereas the random scheduling baseline is denoted by "RAND". Finally, to highlight the upper bound performance, we use the *vanilla FL* method [6] without RB constraints, which is denoted as 'IDEAL" hereinafter.

4.2 Loss of accuracy comparison

Fig. 10 compares the loss of accuracy in all FL methods at each model aggregation round with respect to centralized model training.

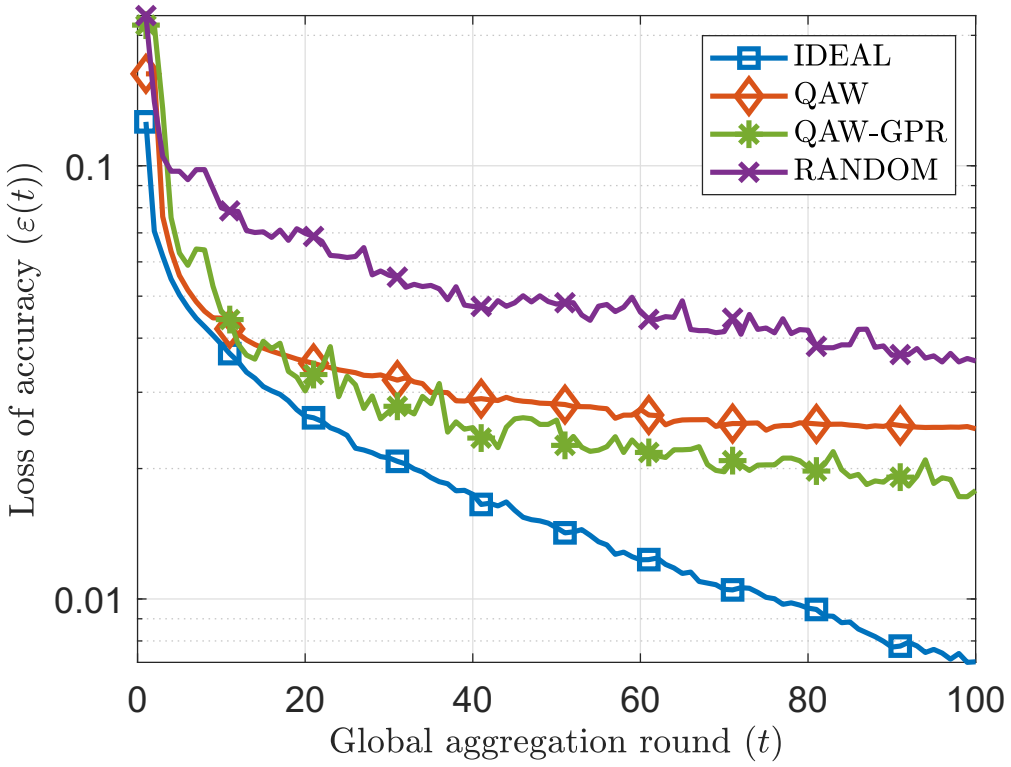

Figure 10. Comparison of the loss of accuracy in all FL methods for each model aggregation round vs. centralized training with perfect and imperfect CSI and $B = 6$, Zipf parameter $\sigma = 1.017$.

In IDEAL within simulations there is no any communication constraints, which means any amount of clients can be scheduled regardless of any uncertainty in communication channel. Thus IDEAL is the best case scenario or the lowest accuracy loss that can be achieved. As expected, it can be noted from the figure that IDEAL has the lowest loss of $\varepsilon(100) = 0.7$ due to the absence of communication constraints. Under perfect CSI, Fig. 10 plots QAW, QUNAW and RAND accuracy losses for comparison.

Further, for $B = 6$, shows that the quantity aware scheduling (QAW) reduces the loss by 15.9 % compared to QUNAW. Under imperfect CSI, QAW-GPR and RAND are compared in Fig. 10 alongside IDEAL and QAW. While RAND shows a poor performance, QAW-GPR outperforms QAW by reducing the loss by 28 %.

However, in QAW allocates one RB for CSI measurements thus, only the rest of the RBs are allowed to be utilized for clients to upload their model and download the global model. The main reason for the improvement in QAW-GPR compared to both the others is due to the aforementioned utilization of all $B = 6$ RBs for scheduling clients compared to QAW.

4.3 Impact of per-client dataset size

Fig. 11 plots the impact of heterogeneity in the training sample size per client on the loss of accuracy.

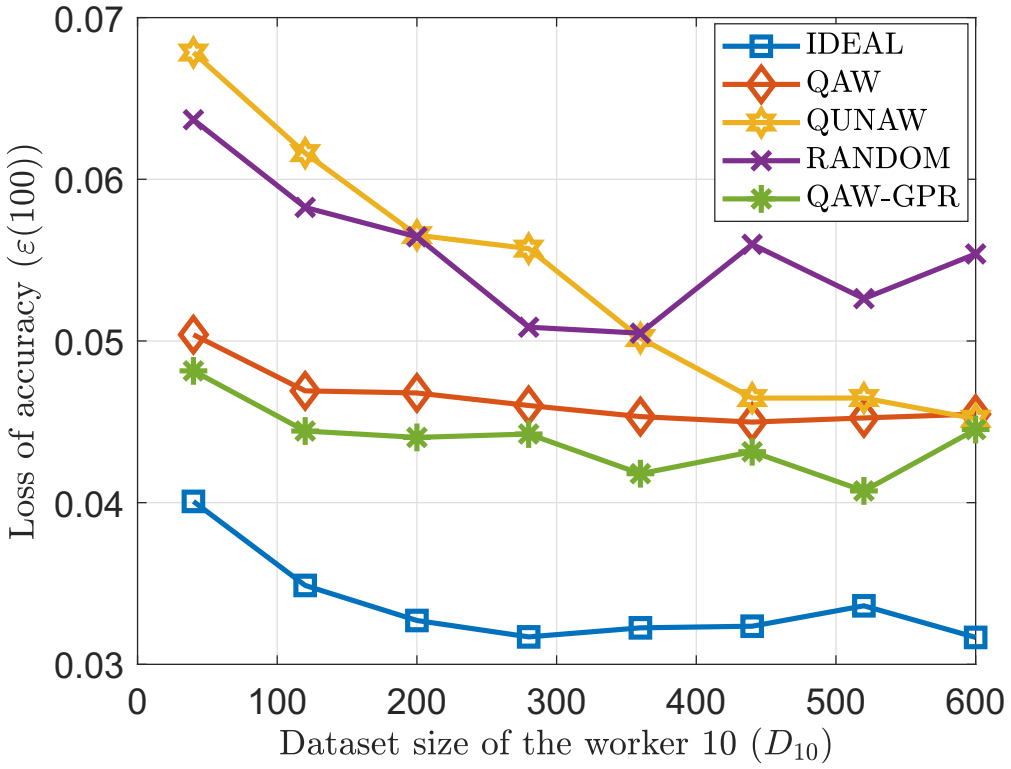

Figure 11. Comparison of the loss of accuracy per client dataset size.

Here, the x-axis represents the number of data samples per client with the minimum number of training data, i.e., the dataset size of the 10th client D_{10} as per the Zipf's distribution. All methods exhibit higher losses in accuracy when the training samples are asymmetrically distributed over clients, i.e., for the lower D_{10}. As D_{10} is increased, the losses in accuracy are reduced. It is also worth noting that the losses of accuracy in the proposed methods QAW and QAW-GPR remain almost constant for $D_{10} > 100$. The loss reductions in QAW-GPR over QAW are due to the additional RB with the absence of CSI measurement. In contrast, QUNAW yields higher losses when training data is unevenly distributed among clients. The reductions of the loss in QAW at $D_{10} = 40$ are 25.72 % and 20.87 % compared to QUNAW and RAND, respectively. The reason behind the these lower losses of the proposed methods over the baselines is that client scheduling takes into account the training dataset size. For $D_{10} = 600$, due to the equal dataset sizes per client, the accuracy loss with QAW and QUNAW identical. Therein, both QAW and QUNAW exhibit about 18.4 % reduction in accuracy loss compared to RAND.

4.4 Impact of number of the resource block availability

The following Fig. 12 draws the impact of the number of resource block availability of proposed methods both QAW and QUNAW with reference to IDEAL.

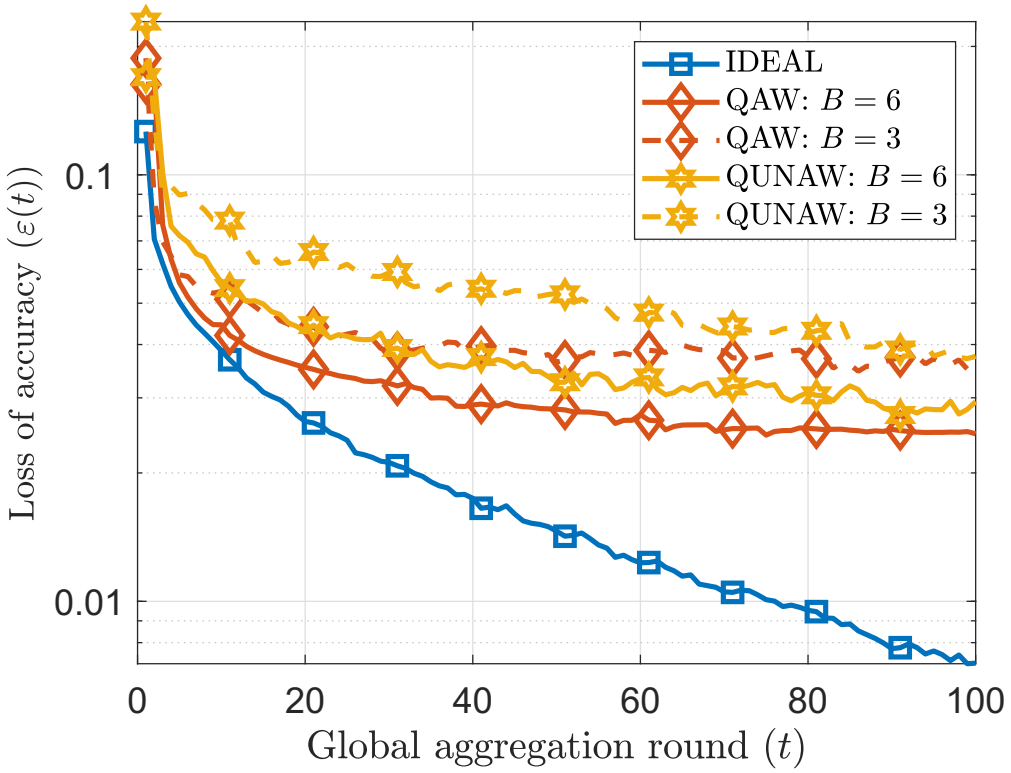

Figure 12. Comparison of the loss of accuracy in all FL methods for each model aggregation round vs. centralized training with perfect CSI and $B = \{3, 6\}$, Zipf parameter $\sigma = 1.017$.

At a glance from Fig. 12, it is noticeable that reduction in the resource block availability have caused in scarification in gains in terms of accuracy loss. For $B = 3$, the accuracy losses in both QAW and QUNAW are almost identical. The reason for that is when $B = 2$ the available RBs for scheduled clients communication is 2. Thus the impact of scheduling with the knowledge of dataset size of clients, get declined with decreasing number of RBs.

However, the increase in the accuracy loss due to doubling of the RBs to $B = 6$ value in QUNAW is 20.74 %. In QAW method, that increase in the accuracy loss is 28.63 % From these value figures notably QAW performs very well when increasing the communication feasibility. The reason behind it is that the dataset effect considered scheduling impact to accuracy is higher when there is more possibilities to schedule clients. Thus, it can be concluded that the effect of QAW shines when there is higher number of RB available.

4.5 Impact of Fairness

Finally, in terms of fairness, the accuracy per client for different FL methods are investigated in Fig. 13.

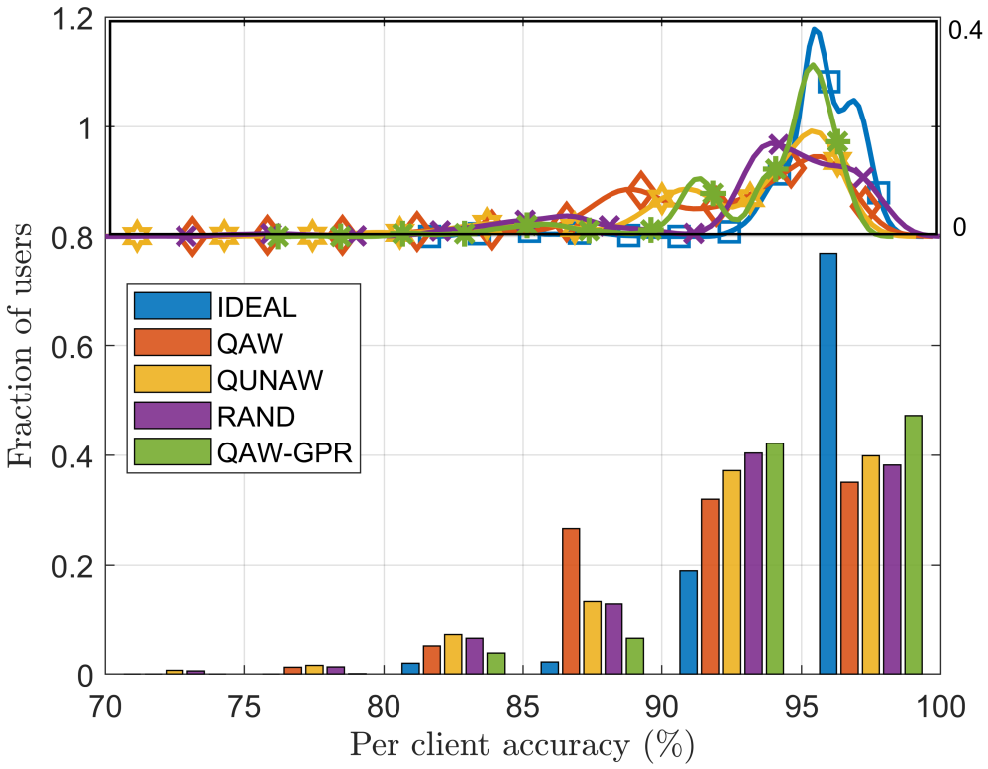

Figure 13. Fairness comparison of the training accuracy among clients, Zipf parameter $\sigma = 1.071$.

Here, IDEAL exhibits the highest average training accuracy of 95.3 % as well as the lowest variance of 5.6 over the clients compared to all other methods. This demonstrates that the most fairness in terms of training accuracy is provided by IDEAL thanks to benefit of unconstrained communication. With QAW-GPR, 93.5 % of average accuracy and 10.7 variance is observed. Client scheduling utilizing all $B = 6$ RBs offers the advantage for the aforementioned performance over all the other methods considering the communication limitations. It can be also be seen that QAW and QUNAW have almost equal means (92 %) and variances of 16.9 and 19.5, respectively. Scheduling clients to train over a larger dataset in QAW provides a lower variance in accuracy over QUNAW. Although RAND is CSI-agnostic, it yields an average accuracy of 92.8 % and the highest variance of 19.41. This indicates that the client scheduling without any insight of datasize distribution and CSI cannot provide high training accuracy or fairness under communication constraints.

4.6 The impact of CSI correlation on the GPR-based prediction

Fig. 14, compares complementary cumulative distribution function (CCDF) plots of the GPR-based CSI prediction errors for the Rayleigh distributed channels with different correlations. The channels are generated with Clark-Gans model as described in section 1.2.5. Lower Doppler frequency (f_m), implies higher correlation between channel samples. Thus, by changing f_m three level of correlated channels are generated with average of channels' absolute values, 1.076.

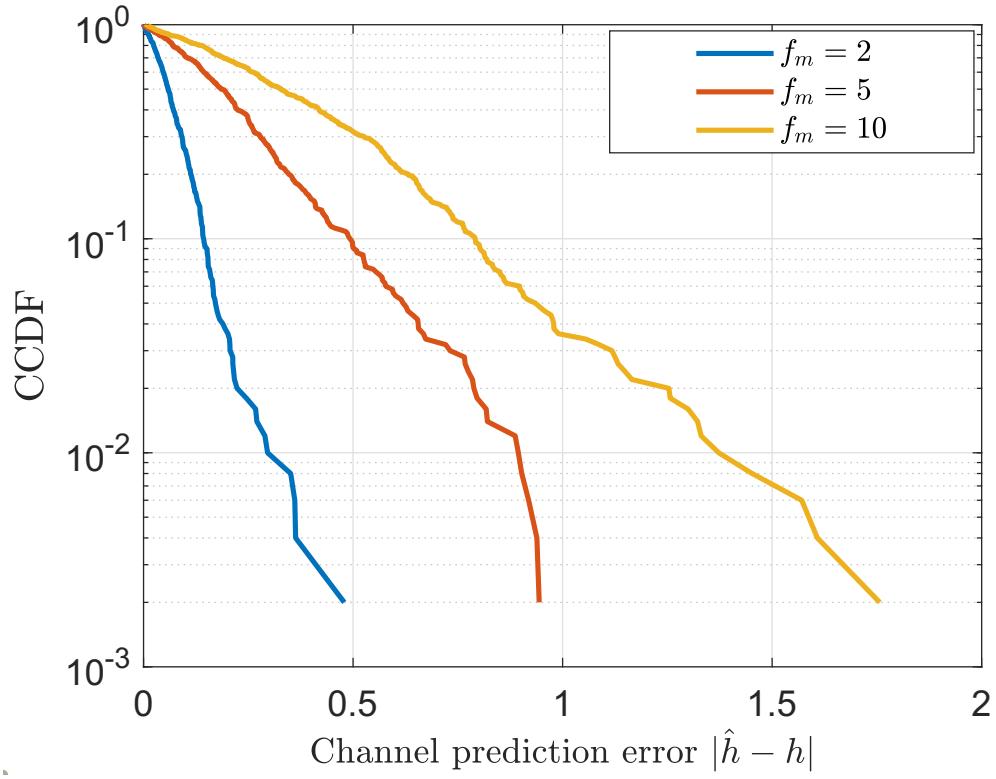

Figure 14. The error CCDF of GPR-based predictions for three correlated Rayleigh distributed channels with different correlations.

From Fig. 14, it can be noted that GPR-based prediction accuracy improves with the channel correlation. Channel prediction error is calculated taking the the absolute value of the difference between prediction and the actual ($|\hat{h} - h|$). For the higher correlation ($f_m = 2$) all the predictions are precise, in which the prediction errors are below of 0.479. For medium correlation ($f_m = 5$) the maximum error is 0.9442 where as for the lower correlation ($f_m = 10$), it reaches up to 1.757, which is the highest from all three correlations.

Further from Fig. 14, it is shown that, there is 99.9 % reliability that prediction errors with higher correlation are only up to 0.296. With lower correlation, it is 99.9 % reliable that prediction errors are below 1.373 where as with medium correlation it is 0.8944.

When the correlation is higher channel shows less fluctuations. Thus, prediction captures the pattern of channel fluctuations with less previous data and predict channel more accurately. Hence, the GPR prediction accuracy have an impact with the correlation of channels.

5 CONCLUSION AND FUTURE EXTENSIONS

In this thesis, a joint client scheduling and RB allocation policy for FL over wireless links under imperfect CSI is proposed. The problem of client scheduling and RB allocation was cast to minimize the training loss compared to centralized approach and the CSI uncertainties. Resorting to GPR-based channel prediction method and deriving an upper bound for the loss of accuracy in FL compared to a centralized approach, the stochastic optimization problem was solved using Lyapunov optimization. With extensive set of simulations, we evaluated the performance of the proposed methods for both perfect and imperfect CSI. Results show that the performance of the proposed methods outperforms the baseline client scheduling and RB allocation methods adopted in FL, especially when training data is unevenly distributed among clients.

In this work, for the convenience of optimal client scheduling policy derivation, it is assumed that all the clients have the ability to solve its local NN model training subproblem upto a certain accuracy irrespective to the dataset size and computational capabilities. Practically, with the dataset size and due to computational capability differences, optimality that each worker solve its subproblem, is not the same. Thus, analyzing the impact of computation and communication on FL accuracy are potential future extensions.

6 REFERENCES

[1] McCarthy J. (2004) What is artificial intelligence? .

[2] Silver D., Huang A., Maddison C.J., Guez A., Sifre L., Van Den Driessche G., Schrittwieser J., Antonoglou I., Panneershelvam V., Lanctot M. et al. (2016) Mastering the game of go with deep neural networks and tree search. nature 529, p. 484.

[3] Mitchell T.M. (1999) Machine learning and data mining. Communications of the ACM 42.

[4] Dhar V. (2012) Data science and prediction .

[5] Sze V., Chen Y.H., Emer J., Suleiman A. & Zhang Z. (2017) Hardware for machine learning: Challenges and opportunities. In: 2017 IEEE Custom Integrated Circuits Conference (CICC), IEEE, pp. 1–8.

[6] Konečnỳ J., McMahan H.B., Yu F.X., Richtárik P., Suresh A.T. & Bacon D. (2016) Federated learning: Strategies for improving communication efficiency. arXiv preprint arXiv:1610.05492 .

[7] Hard A., Rao K., Mathews R., Beaufays F., Augenstein S., Eichner H., Kiddon C. & Ramage D. (2018) Federated learning for mobile keyboard prediction. arXiv preprint arXiv:1811.03604 .

[8] Van den Stock J., Righart R. & De Gelder B. (2007) Body expressions influence recognition of emotions in the face and voice. Emotion 7, p. 487.

[9] Millard-Ball A. (2018) Pedestrians, autonomous vehicles, and cities. Journal of Planning Education and Research 38, pp. 6–12.

[10] Gautam A., Naik V.S., Gupta A., Sharma S. & Sriram K. (2015) An smartphone-based algorithm to measure and model quantity of sleep. In: 2015 7th International Conference on Communication Systems and Networks (COMSNETS), IEEE, pp. 1–6.

[11] Myers B.A. & Beigl M. (2003) Handheld computing. Computer 36, pp. 27–29.

[12] Chai Z., Fayyaz H., Fayyaz Z., Anwar A., Zhou Y., Baracaldo N., Ludwig H. & Cheng Y. (2019) Towards taming the resource and data heterogeneity in federated learning. In: 2019 {USENIX} Conference on Operational Machine Learning (OpML 19), pp. 19–21.

[13] Rappaport T.S. et al. (1996) Wireless communications: principles and practice, vol. 2. prentice hall PTR New Jersey.

[14] Neely M.J. (2010) Stochastic network optimization with application to communication and queueing systems. Synthesis Lectures on Communication Networks 3, pp. 1–211.

[15] Janocha K. & Czarnecki W.M. (2017) On loss functions for deep neural networks in classification. arXiv preprint arXiv:1702.05659 .

[16] Ghojogh B. & Crowley M. (2019) The theory behind overfitting, cross validation, regularization, bagging, and boosting: Tutorial. arXiv preprint arXiv:1905.12787 .

[17] Smith V., Forte S., Ma C., Takac M., Jordan M.I. & Jaggi M. (2016) CoCoA: A general framework for communication-efficient distributed optimization. arXiv preprint arXiv:1611.02189 .

[18] Sra S., Nowozin S. & Wright S.J. (2012) Optimization for machine learning. Mit Press.

[19] Sutskever I., Martens J., Dahl G. & Hinton G. (2013) On the importance of initialization and momentum in deep learning. In: International conference on machine learning, pp. 1139–1147.

[20] Polyak B.T. & Juditsky A.B. (1992) Acceleration of stochastic approximation by averaging. SIAM Journal on Control and Optimization 30, pp. 838–855.

[21] Hinton G., Srivastava N. & Swersky K. (2012) Overview of mini-batch gradient descent. Neural Networks for Machine Learning 575.

[22] Chen T., Giannakis G., Sun T. & Yin W. (2018) LAG: Lazily aggregated gradient for communication-efficient distributed learning. In: Advances in Neural Information Processing Systems, pp. 5050–5060.

[23] Yang H.H., Liu Z., Quek T.Q. & Poor H.V. (2019) Scheduling policies for federated learning in wireless networks. arXiv preprint arXiv:1908.06287 .

[24] Chen M., Yang Z., Saad W., Yin C., Poor H.V. & Cui S. (2019) A joint learning and communications framework for federated learning over wireless networks. arXiv preprint arXiv:1909.07972 .

[25] Reisizadeh A., Mokhtari A., Hassani H., Jadbabaie A. & Pedarsani R. (2019) FedPAQ: A communication-efficient federated learning method with periodic averaging and quantization. arXiv preprint arXiv:1909.13014 .

[26] Samarakoon S., Bennis M., Saad W., Debbah M. & Latva-Aho M. (2016) Ultra dense small cell networks: Turning density into energy efficiency. IEEE Journal on Selected Areas in Communications 34, pp. 1267–1280.

[27] Urgaonkar R., Urgaonkar B., Neely M.J. & Sivasubramaniam A. (2011) Optimal power cost management using stored energy in data centers. In: Proceedings of the ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems, ACM, pp. 221–232.

[28] Clarke R. (1968) A statistical theory of mobile-radio reception. Bell system technical journal 47, pp. 957–1000.

[29] Semeter J. (2012) Basic radar signal processing. Boston University.[Online]. Available: https://eiscat3d. se/drupal/sites/default/files/IISRWS2011/08-Semeter Radar Signal Processing. pdf .

[30] Mansfield K. (2007) The White Book: The Beatles, the Bands, the Biz: An Insider's Look at an Era. Harper Collins.

[31] Williams C.K. & Rasmussen C.E. (2006) Gaussian processes for machine learning, vol. 2. MIT press Cambridge, MA.

[32] Pérez-Cruz F., Vaerenbergh S.V., Murillo-Fuentes J.J., Lázaro-Gredilla M. & Santamaría I. (2013) Gaussian processes for nonlinear signal processing: An overview of recent advances. IEEE Signal Processing Magazine 30, pp. 40–50.

[33] Rasmussen C.E. & Williams C.K.I. (2005) Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning). The MIT Press.

[34] Karaca M., Ercetin O. & Alpcan T. (2016) Entropy-based active learning for wireless scheduling with incomplete channel feedback. Computer Networks 104, pp. 43–54.

[35] Alpcan T. (2011) Dual control with active learning using gaussian process regression. arXiv preprint arXiv:1105.2211 .

[36] Karaca M., Alpcan T. & Ercetin O. (2012) Smart scheduling and feedback allocation over non-stationary wireless channels. In: 2012 IEEE International Conference on Communications (ICC), IEEE, pp. 6586–6590.

[37] Hiriart-Urruty J. & Lemaréchal C. (2004) Fundamentals of Convex Analysis. Grundlehren Text Editions, Springer Berlin Heidelberg.

[38] Xing E.P. (2015) Advanced Gaussian Processes. 10-708: Probabilistic Graphical Models, Carnegie Mellon School of Computer Science, University in Pittsburgh, Pennsylvania, 1–8 p.

7 APPENDICES

7.1 Proof of upper bound of $\varepsilon(T)$

After t and $t+1$ communication rounds the expected increment in the dual formulation of objective (15a) function is as below,

$$\mathbb{E}\Big(\psi(\boldsymbol{\theta}(t+1)) - \psi(\boldsymbol{\theta}(t))\Big) \geq \mathbb{E}(\psi(\boldsymbol{\theta}(t+1))) - \mathbb{E}(\psi(\boldsymbol{\theta}(t)))$$

This inequality holds since the expectations of difference is greater than the difference of the expectations. Thereafter, add and subtract the same phrase, which is the optimal dual function value to the R.H.S. for the formulation.

$$\begin{aligned}\mathbb{E}\Big(\psi(\boldsymbol{\theta}(t+1)) - \psi(\boldsymbol{\theta}(t))\Big) &\geq \mathbb{E}(\psi(\boldsymbol{\theta}^\star)) - \mathbb{E}(\psi(\boldsymbol{\theta}(t))) + \mathbb{E}(\psi(\boldsymbol{\theta}(t+1))) - \mathbb{E}(\psi(\boldsymbol{\theta}^\star)) \\ &= \sum_{k=1}^{K} \Delta\psi(\Delta\boldsymbol{\theta}_k^\star(t)) - \mathbb{E}(\psi(\boldsymbol{\theta}(t))) + \sum_{k=1}^{K} \Delta\psi(\Delta\boldsymbol{\theta}_k(t)) - \sum_{k=1}^{K} \Delta\psi(\Delta\boldsymbol{\theta}_k^\star(t))\end{aligned}$$

From (19) and following definition $\mathbb{E}(\psi(\boldsymbol{\theta}(t))) = \sum_{i=0}^{K} \Delta\psi(0)$ since its same as previous communication round update,

$$\begin{aligned}\mathbb{E}\Big(\psi(\boldsymbol{\theta}(t+1)) - \psi(\boldsymbol{\theta}(t))\Big) &\geq \sum_{k=1}^{K} \Delta\psi(\Delta\boldsymbol{\theta}_k^\star(t)) - \mathbb{E}(\psi(\boldsymbol{\theta}(t))) - \beta\Big\{\sum_{k=1}^{K} \Delta\psi(\Delta\boldsymbol{\theta}_k^\star(t)) - \mathbb{E}(\psi(\boldsymbol{\theta}(t)))\Big\} \\ &= (1-\beta)\Big\{\sum_{k=1}^{K} \Delta\psi(\Delta\boldsymbol{\theta}_k^\star(t)) - \mathbb{E}(\psi(\boldsymbol{\theta}(t)))\Big\}\end{aligned}$$

This is the ultimate bound when all the data points in $\mathcal{D}$ are utilized. However with scheduling and upto t number of communication iterations,

$$\begin{aligned}\mathbb{E}\Big(\psi(\boldsymbol{\theta}(t+1)) - \psi(\boldsymbol{\theta}(t))\Big) &\geq (1-\beta)\Big\{\sum_{k=1}^{K} \Delta\psi(\Delta\boldsymbol{\theta}_k^\star(t)) - \mathbb{E}(\psi(\boldsymbol{\theta}(t)))\Big\} \\ &\geq (1-\beta)\Big(\sum_{\tau=1}^{t}\sum_{i=1}^{K} \frac{D_k}{tD} s_k(t)\Big)\Big\{\sum_{k=1}^{K} \Delta\psi(\Delta\boldsymbol{\theta}_k^\star(t)) - \mathbb{E}(\psi(\boldsymbol{\theta}(t)))\Big\}\end{aligned}$$

Now, following Appendix B in [23], $\Big\{\sum_{k=1}^{K} \Delta\psi(\Delta\boldsymbol{\theta}_k^\star(t)) - \mathbb{E}(\psi(\boldsymbol{\theta}(t)))\Big\} \geq \bar{s}\Big\{\psi(\boldsymbol{\theta}^\star) - \mathbb{E}(\psi(\boldsymbol{\theta}(t)))\Big\}$, where $\bar{s} \in (0,1)$, it can be derived,

$$\begin{aligned}\varepsilon(T) &= \mathbb{E}\Big(\psi(\boldsymbol{\theta}^\star) - \psi(\boldsymbol{\theta}(T+1))\Big) \\ &= \mathbb{E}\Big(\psi(\boldsymbol{\theta}^\star) - \psi(\boldsymbol{\theta}(T))\Big) - \mathbb{E}\Big(\psi(\boldsymbol{\theta}(T+1)) - \psi(\boldsymbol{\theta}(T))\Big) \\ &\leq \mathbb{E}\Big(\psi(\boldsymbol{\theta}^\star) - \psi(\boldsymbol{\theta}(T))\Big) - (1-\beta)\Big(\sum_{\tau=1}^{t}\sum_{i=1}^{K} \frac{D_k}{TD} s_k(t)\Big)\Big\{\psi(\boldsymbol{\theta}^\star) - \mathbb{E}(\psi(\boldsymbol{\theta}(T)))\Big\} \\ &= \Big(1 - (1-\beta)\sum_{\tau=1}^{T}\sum_{i=1}^{K} \frac{D_k}{TD} s_k(t)\Big)\Big\{\psi(\boldsymbol{\theta}^\star) - \mathbb{E}(\psi(\boldsymbol{\theta}(T)))\Big\} \\ &\vdots \\ &\leq \Big(1 - (1-\beta)\sum_{\tau=1}^{T}\sum_{i=1}^{K} \frac{D_k}{TD} s_k(t)\Big)^T \Big\{\psi(\boldsymbol{\theta}^\star) - \mathbb{E}(\psi(\boldsymbol{\theta}(0)))\Big\}\end{aligned}$$

In [17] it is proved that $\left\{\psi(\boldsymbol{\theta}^{\star}) - \mathbb{E}(\psi(\boldsymbol{\theta}(0)))\right\} < D$. Following that,

$$\varepsilon(T) \leq \left(1 - (1-\beta)\sum_{\tau=1}^{T}\sum_{i=1}^{K}\frac{D_k}{TD}s_k(t)\right)^T D$$

7.2 Proof of GPR uncertainty represents information content

Let the entropy function be $H(.)$. Thus, after sampling the channel $h(t)$ at communication iteration t, $H(\hat{h}|[h(t_n)]_{n\in\mathcal{N}}) = j(t) = 0$. Otherwise, $j(t) = H(\hat{h}(t)|t, [h(t_n)]_{n\in\mathcal{N}})$.

Note that, a posterior distribution of channel given for GPR is Gaussian distributes. Thus,

$$p(\hat{h}(t)|t, [h(t_n)]_{n\in\mathcal{N}}) \sim \mathcal{N}(\hat{h}(t); V(t))$$

where $\hat{h}(t)$ is the channel estimate and $V(t)$ is the GPR uncertainty, at communication iteration t. Since the entropy of this Gaussian distribution closed form expression is,

$$H(\hat{h}(t)|t, [h(t_n)]_{n\in\mathcal{N}}) = \frac{1}{2}\log(2\pi e V(t))$$

Since, information content $j(t) = \frac{1}{2}\log(2\pi e V(t))$, therefore for $j(t)$, $V(t)$ can be used as a measuring quantity of information content since log is a monotonically increasing function.