HYBRID DISTRIBUTED STOCHASTIC GRADIENT DESCENT FOR FEDERATED LEARNING

By Xiaofeng Lin

A thesis submitted to the Graduate Program in Electrical & Computer Engineering

in conformity with the requirements for the Degree of Master of Applied Science

Queen's University
Kingston, Ontario, Canada
July, 2020

Abstract

With the advancement of information technology in the past decades, the world embraces the era of 'Big Data', in which large volumes of data are being produced in high velocity, while there is an increasing demand in processing these data. Such environment sets up a perfect playground for deep learning, which is able to utilize the large volumes of data to achieve various tasks. However, as both the volumes of data and the complexity of neural network architecture rises, it becomes increasingly expensive to train the model on a single machine. Federated learning becomes a hot research topic in recent years, which decentralizes the conventional deep learning architecture by distributing both data storage and/or computation operations to multiple machines, while it requires no exchange of information about the local training data so that the data privacy is preserved. In the literature, two different transmission approaches for federated learning, analog-based transmissions and digital-based transmissions, were studied and it was shown that the analog-based approach considerably outperforms the digital-based approach by utilizing the waveform superposition principle of the wireless access medium.

In this thesis, we propose the Hybrid Distributed Stochastic Gradient Descent (Hybrid DSGD) algorithm, a training scheme for federated learning which utilizes the advantages of both digital and analog transmissions to reduce communication

overhead and latency. We demonstrate why the conventional analog-based transmission schemes perform poorly when the number of workers participating the training and/or the power available for each worker are restricted. We then explain how our scheme addresses such issue. We will show through experiments that the hybrid DSGD is able to outperform the conventional analog-based transmission scheme under such circumstance.

Acknowledgments

I would like to first thank my supervisor Prof. Il-Min Kim for providing this research opportunity, and his underlying patient and academic supports, and also for reviewing this thesis.

I would also like to provide my gratitude to my parents, who unconditionally provided me supports in various aspects, and encouraged me through hard times.

Contents

List of Figures

Chapter 1

Introduction

1.1 Deep learning

The intrinsic idea of deep learning is connectionism, that is, by connecting a large number of simple layers, the neural network may have intelligent behaviors [1]. Since Hinton et al's work in [2] in 2006, research on deep learning has bloomed in the past decade, and achieved breakthroughs in various fields, including image classification [3]–[5], natural language processing [6]–[9], speech recognition [10]–[13], etc.

Albeit deep learning is data hungry, thanks to the rapid development of information technology in general over the past decades, there are increasing volumes of data being produced everyday in various fields, which brings the era of so-called 'Big Data'. To utilize such large volume of data for deep learning, a great depth of layers for approximating complex problems is required to produce a deep learning model that performs well. As a result, both the storage and computation resource on a single machine are usually not sufficient for training a large-scaled deep learning model. A natural solution to this is decentralizing the whole process by distributing the training datasets and computation operations to multiple devices.

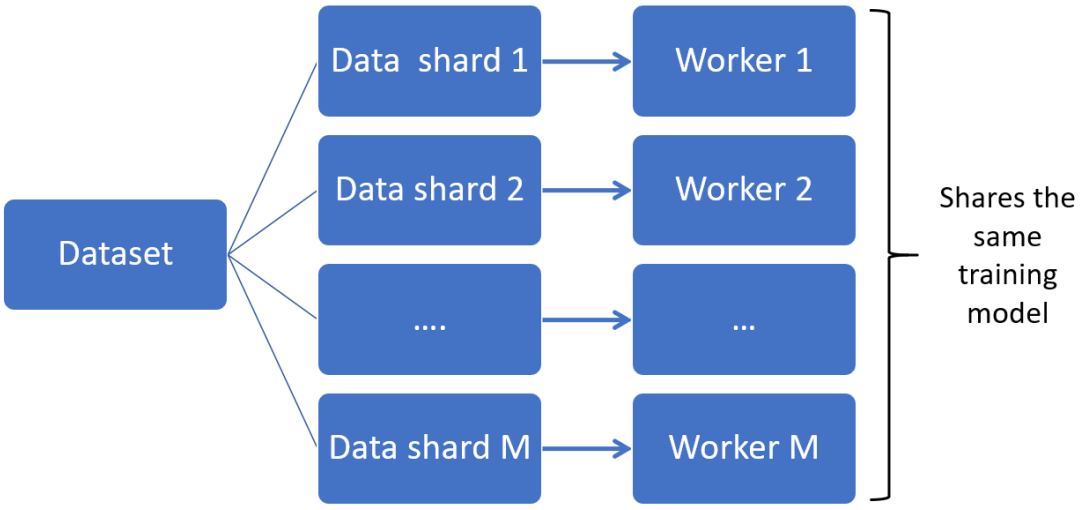
1.1.1 Distributed deep learning

Inspired by microcolumns of neurons in cerebral cortex, Ciregan et al. in their work [14] shows that training a group of relatively small models and averaging their predictions can outperform the conventional deep neural network trained with a single model. This indicates the possibility of parallel training for deep learning, as each small model can be distributed to a distinct local machine, i.e. a worker, and train it locally to achieve similar performance as a single centralized large training model. Since then research on various parallelism schemes for deep learning has been conducted. There are two major parallelism schemes that are well-studied:

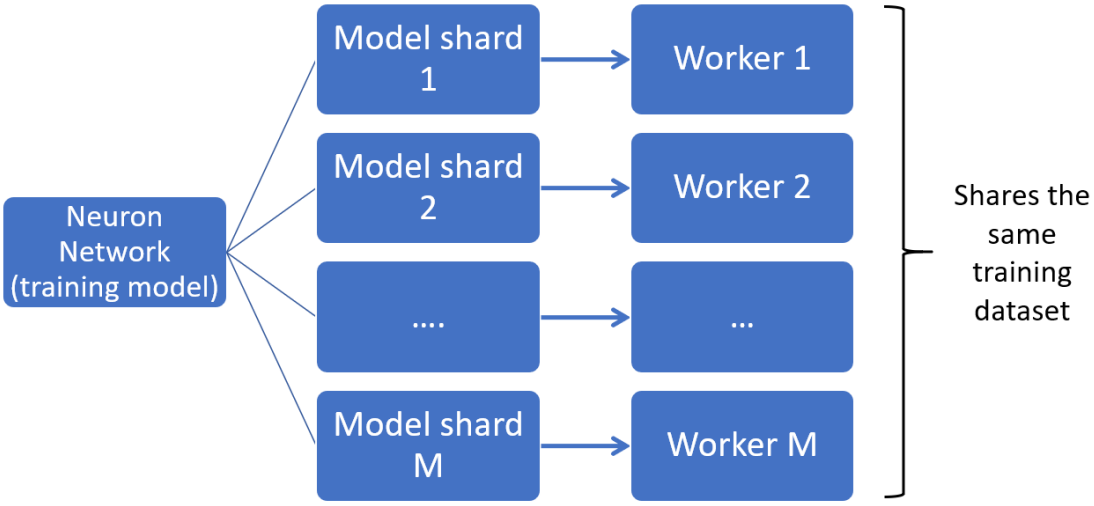
- **Data parallelism:** Workers have identical local training models and train them on non-overlapping local training data, as shown in Figure 1.1a.
- **Model parallelism:** Each worker holds a replica of different shard of training model, and trains its local model with identical local data, as shown in 1.1b.

There also exist a number of hybrid schemes combining data and model parallelisms such as pipeline parallelism [15], [16]. Among them, data parallelism is the easiest one to implement as it does not require further knowledge about the training model itself.

Data parallelism scheme can be either centralized or decentralized: in a centralized architecture, a parameter server (PS) is involved for managing global model and coordinating parameter synchronization. Each worker synchronizes its local trainable parameters with the PS at the beginning of every training iteration, and transmits its computed parameters to the PS at the end of training iteration; whereas in decentralized setting, the workers communicate with each other directly, e.g., through



(a) data parallelism



(b) model parallelism

Figure 1.1: Diagrams of data parallelism (top) and model parallelism (bottom) in distributed deep learning

ALLREDUCE operations [17].

1.2 Federated learning

In practical applications, however, the conventional distributed deep learning has two issues:

- **Scalability:** In the fields such as Internet of Things (IoT), workers are usually low-cost and low-powered. Therefore, communications between the PS and workers in a centralized system, or between workers in a decentralized system are usually slow in large-scaled distributed systems, which hinders the model training speed. Furthermore, it is costy and inefficient to transfer and store massive volume of data collected from workers on the PS in a centralized system.

- **Regulations:** In the scenarios where workers are also clients from different parties, sharing collected data has the potential risk of compromising data privacy even with data encryption. As a result, regulations such as the General Data Protection Regulation (GDPR) are enforced to protect data privacy by banning autonomous operations, including data collection and sharing without user permissions. Therefore, in many cases it is infeasible to merge local data from multiple parties to an integrated one at the PS in a centralized system without violating policies and regulations. This leads to the phenomenon commonly referred to as "isolated data island". [18], [19]

The state-of-the-art distributed solution to these challenges is Federated Learning [20], a decentralized parallel training scheme allowing multiple parties to train a shared model without exchanging information about local data itself.

1.2.1 Taxonomy

A survey written by Li et al. [19] provides an overview of existing Federated Learning systems (FLSs) found on Google Scholar and arXiv by comparing their infrastructures. They categorize the FLSs by six aspects:

- **Data partitioning**: Different applications of FLSs may have different distributions over the feature space and sample space. Data can either be horizontally partitioned or vertically partitioned: in the former case, clients share the same or similar feature space but differ in sample space, and vice versa in the latter case. Some FLSs may have hybrid of horizontal and vertical partitioning.

- **Training model**: Depending on the applications, the FLSs may have different machine learning models, neural network architectures, various regression models, decision trees, etc.

- **Privacy mechanism**: There are two major approaches, including data encryption and differential privacy. Data encryption prevents outsiders from obtaining the true meaning of data even if the data transmission is compromised, while differential privacy adds random noise to data so that outsiders can not know if an individual data is used for training.

- **Communication architecture**: A FLS can have either a centralized or decentralized communication architecture: in centralized setting, a PS is involved, which aggregates parameter updates from workers, updates the global model, and transmits updated parameters to each worker. On the contrary, the decentralized FLSs update parameters through direct communications among workers. A FLS can also be either synchronous or asynchronous.

- **Scale of federation**: The FLSs can be either cross-silo or cross-device. In cross-silo setting, workers are usually servers owned by organizations or companies, which have relatively good computation power and large volumes of training data. As for cross-device setting, workers are usually mobile devices which have strict energy and computation power limits, as well as much smaller training data size.

- **Motivation of federation**: Worker/client's motivation for using federated learning can be regulation or incentives. For organizations, they use federated learning to avoid violating laws and regulations, while for individuals they may decide to participate in Federated Learning to get better services.

1.2.2 Example system model

We considered a centralized and synchronized federated learning system, involving a single PS managing the global training model, and multiple local devices called workers. Each of them shares a same training model architecture as the PS, and computes the gradients locally.

One of the commonly used algorithms for federated learning is Federated Averaging (FedAvg) [20]. Consider the following generalized objective function in machine learning:

$$F(\boldsymbol{\theta}) = \frac{1}{N}\sum_{n=1}^{N} f(\boldsymbol{\theta}, B_n), \tag{1.1}$$

where $\boldsymbol{\theta}$ is the vector of trainable model parameters (i.e. the weights in the neural network), B_n is the n^{th} data batch, N is the total number of data batches, and $f(\cdot)$ is the objective function selected. In a federated learning system with cross-device setting, the data batches are stored on workers by either data partitioning or local

data collecting. Let $m = 1, 2, \cdots, M$ be the m^{th} worker out of M workers. Each worker has local training data batch B_m stored locally. Then the objective function can now be written as follows:

$$F(\boldsymbol{\theta}) = \frac{1}{M} \sum_{m=1}^{M} f(\boldsymbol{\theta}, B_m). \tag{1.2}$$

That is, $F(\boldsymbol{\theta})$ is the average of the aggregated objective function from all workers.

The goal is to minimize $F(\boldsymbol{\theta})$ by selecting proper values for model parameter $\boldsymbol{\theta}$. This is usually done by using stochastic gradient descent (SGD) to 'train' the model for multiple iterations as follows:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \rho_t \nabla F(\boldsymbol{\theta}_t), \tag{1.3}$$

where ρ_t is the learning rate at iteration t, $\boldsymbol{\theta}_t$ is the model parameter vector at iteration t, and $\nabla F(\boldsymbol{\theta}_t)$ denotes the gradient vector of the average of the aggregated objective function with respect to the model parameters at iteration t. In each iteration, $F(\boldsymbol{\theta}_t)$ is calculated according to eq. (1.2) with $\boldsymbol{\theta}_t$, that is:

$$F(\boldsymbol{\theta}_t) = \frac{1}{M} \sum_{m=1}^{M} f(\boldsymbol{\theta}_t, B_m). \tag{1.4}$$

And now eq. (1.3) can be written as:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \rho_t \frac{1}{M} \sum_{m=1}^{M} \nabla f(\boldsymbol{\theta}_t, B_m), \tag{1.5}$$

$$= \boldsymbol{\theta}_t - \rho_t \frac{1}{M} \sum_{m=1}^{M} \boldsymbol{g}_m(\boldsymbol{\theta}_t), \tag{1.6}$$

where $g_m(\boldsymbol{\theta}_t) = \nabla f(\boldsymbol{\theta}_t, B_m)$ is the local gradient computed by worker m with respect to the parameters $\boldsymbol{\theta}_t$. The model parameters are updated with the average of the aggregated gradients calculated with respect to current model parameters $\boldsymbol{\theta}_t$ and local data B_m.

For each iteration of training, each worker transmits only its local gradient $\boldsymbol{g}_m(\boldsymbol{\theta}_t)$ to the PS. After receiving all the local gradients, the PS then updates the global model according to eq. (1.6), and broadcasts the updated parameters $\boldsymbol{\theta}_{t+1}$ to all workers for next iteration of training, as shown in Fig. 1.2.

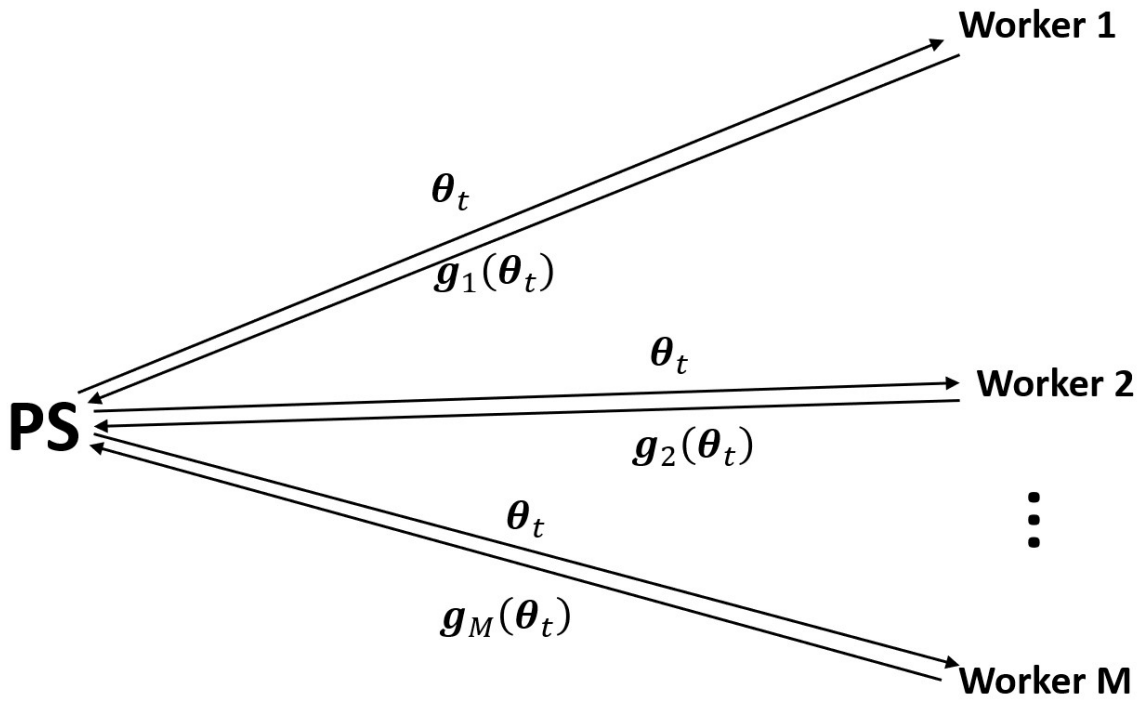


Figure 1.2: Architecture of centralized distributed learning. A parameter server (PS) transmits the trainable parameters $\boldsymbol{\theta}_t$ to each local machine called a worker, and each worker computes the gradients of $\boldsymbol{\theta}_t$ with respect to the local training data and send back its gradient $\boldsymbol{g}_m(\boldsymbol{\theta}_t)$

The advantages of federated learning is obvious: it is more scalable in terms of

memory and computation cost compared to the conventional centralized distributed learning system, and is less likely to compromise data privacy, as no information about the training data itself leaves the local device. The latter is especially important in applications where data is transmitted wirelessly, such as Internet of Things (IoT), where the worker units could be devices such as mobile phones or smart watch, and the training data are records that contain private information about the device owners.

1.2.3 Industrial applications

Federated learning has been implemented in various industries. For example, the Google keyboard application Gboard on Android uses federated learning to improve suggestion query by caching history of user interactions with suggestion messages [21]. Gboard uses combination of random rotations and quantizations for compressing transmission overhead [22], and uses FedAvg for model updates. MELLODDY, a drug-discovery consortium conducted a 3-year project, which brought together 17 parties [23], including 10 pharmaceutical companies to use federated learning and block-chain to train on the data of 10 million chemical compounds.

1.2.4 Challenges

Nothing is perfect. Some core challenges federated learning is facing include data and/or system heterogeneity, data privacy, and communication efficiency.

In practice, many local data are not independent and identically distributed (IID) and unbalanced in size. Without careful tuning, the conventional FedAvg optimization does not guarantee model convergence. Moreover, since the PS does not have

direct control to workers, it may lose connections to workers due to hardware malfunctioning or slow connection speed, which is common for devices such as mobile phones as they have different hardware specifications. Numerous researches [24]–[27] show that unbalanced, non-IID data distribution as well as partial participants hinder the convergence speed of the FedAvg model.

Although federated learning pushes one step further to reduce risks of privacy leakage compared to the conventional distributed deep learning architecture, extensive studies on privacy breaching show that federated learning system may still not provide sufficient guarantee for privacy. For example, it was shown that a malicious participant can perform various inference attacks by observing difference between updated parameter at different iterations, such as class representive inference [28], membership inference [29], [30], and training input/label inference [31], [32].

In this work, we focus on the third challenge, i.e., improving the communication efficiency. There are two directions in the communications for federated learning. One is the wireless communication from workers to the PS, which is referred to as the uplink communication. The other is the wireless communication from the PS to workers, which is referred to as the downlink communication. In this thesis, we focus on the uplink because the downlink communication is relatively easier and its overhead is (much) smaller.

1.3 Problem Statement

In the context of wireless/mobile edge network, each worker has limited power constraint and bandwidth, while the size of the training model may be large. It is known that compared to the actual training process, communication is the major

bottleneck of distributed training [33], [34], whereas additional operations such as encryption would further raise the communication cost. Therefore, reducing the uplink communication overhead becomes one of the focuses of recent research in federated learning.

Model compression is one of the most commonly used methods for improving the communication efficiency. Typical model compression techniques include reducing size of parameter updates by using quantization or sparsification on data transmitted. To improve the model performance, delayed update is usually used in tandem, in which each worker stores untransmitted data locally and transfer them in the later communication round.

While most federated learning studies have previously assumed digital transmission is used for communications, Zhu et al. recently proposed Broadband Analog Aggregation algorithm [35], which builds on the concept of over-the-air computation. The analog transmission scheme considerably reduces communication latency by exploiting the superposition property of multiple access channel.

1.4 Contribution

In this thesis, we propose a distributed stochastic gradient descent algorithm with hybrid of digital and analog transmissions, namely the hybrid DSGD, which is able to effectively compress the transmission data, while providing excellent training performance when the bandwidth and the power available to each worker are limited. Such scheme significantly improves the model performance with low sparsity rate, low bandwidth, small number of workers, and limited power constraint compared to the conventional analog-based transmission scheme, and thus the conventional digital

transmission scheme.

1.5 Thesis Outline

The rest of the thesis is organized as follow: Chapter 2 provides an overview on recent research of improving communication efficiency for federated learning. Chapter 3 presents the baseline analog transmission scheme in Section 3.1, discuses the motivation of our proposed scheme in Section 3.2, and presents our proposed hybrid DSGD scheme in Section 3.3. Finally, Chapter 4 presents our experiment results.

Chapter 2

Related Research

In this chapter, we introduce the related research on improving the uplink communication efficiency of federated learning. Section 2.1 provide an overview on techniques for reducing communication overhead, and Section 2.2 discusses recent research on communication schemes using combinations of techniques to improve communication efficiencies and overall model performance.

2.1 Reducing communication overhead

Quantization is one of the most commonly used techniques for reducing communication overhead. The one-bit quantization developed by Seide et al. [36] quantizes each local update to one bit by setting entries in the update (i.e. parameters or gradients) vector larger than 0 to 1 and the others to 0, then calculates the means of these two groups to obtain μ_1 and μ_0 respectively. The quanized local updates are transmitted along with the two mean values. Updates are reconstructed by setting 1-valued entries to μ_1 and the rest to μ_0. Errors introduced by quantization are stored locally and added to the local updates in the next round. Similarly, Strom proposed

the threshold quantization [37], which also uses local error accumulations and mean-value quantization. It differs from the one-bit quantization by using a fixed threshold τ and quantizing only the entries whose absolute value is larger than τ, and does not transmit the remaining entries. Inspired by this scheme, Dryden et al. in their work [38] suggest to use a fixed proportion π to quantize only π of all entries in the local update with largest absolute value instead.

Another popular technique used is gradient dropping [39]. Similar to the selection method of adaptive quantization, it drops a fixed proportion of gradients that are not significant before the transmission, and accumulates those dropped gradients for next communication round. Gradient dropping can be applied either locally and globally. When applied globally, it is found that using layer normalization is essential for good performance.

Konecný et al. in their works [40] proposed that uplink communication costs can also be reduced by finding a low rank representation of local updates. The original gradient/parameter matrix can be expressed as the product of two lower-rank matrices: a reconstruction matrix that is pre-defined and randomly generated, and a projection matrix. Since the reconstruction matrix is pre-defined and therefore known by all workers, each worker only needs to compute and transmit its projection matrix.

2.2 Communication schemes

In this section, we introduce state-of-the-art communication schemes for federated learning using either digital or analog transmission. They differ from each other by either encoding the local updates, or utilizing analog channel superposition to improve

communication efficiency.

2.2.1 Digital-based transmission schemes

Digital-based transmission schemes are the communication schemes most research considered, where digital modulations and demodulations are used during the communication. Here we introduce two research that use digital-based transmission schemes.

Deep Gradient Compression

Deep gradient compression (DGC) proposed by Lin et al. [33] is an application of the minibatch SGD [41], which focuses on using momentum SGD with large minibatch to train the model with sparsified gradients. Sparsification would effectively reduce the power consumption for transmission. Gradients are sparsified by using gradient dropping. The non-zero entries of each local gradient are encoded to 32-bits binary value while zero entries are encoded to 16-bits run length of zero value, and they are transmitted to other workers through ALLREDUCE operation.

However, vanilla momentum SGD cannot be directly applied to sparsified gradients when error accumulation is used. Consider the vanilla momentum SGD updates scheme:

$$\boldsymbol{u}_t = \beta \boldsymbol{u}_{t-1} + \sum_{m=1}^{M} \boldsymbol{g}_m(\boldsymbol{\theta}_t), \tag{2.1}$$

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \rho_t \boldsymbol{u}_t, \tag{2.2}$$

where β is the momentum term, and $\boldsymbol{u}_t$ is the aggregated momentum-accumulated

gradient vector. When sparsification is used, we have;

$$\boldsymbol{v}_{m,t} = \boldsymbol{v}_{m,t-1} + \boldsymbol{g}_m(\boldsymbol{\theta}_t), \tag{2.3}$$

$$\boldsymbol{u}_t = \beta \boldsymbol{u}_{t-1} + \sum_{m=1}^{M} sparse(\boldsymbol{g}_m(\boldsymbol{\theta}_t)), \tag{2.4}$$

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \rho \boldsymbol{u}_t, \tag{2.5}$$

where $\boldsymbol{v}_{m,t}$ is the local accumulated gradient term of worker m at iteration t, and $sparse(\cdot)$ is the sparsification method selected by the user. However, the term $\boldsymbol{u}_t$ here is not equivalent to the one in eq. (2.1), since the gradients that are set to zero are not accumulating the momentum. In other words, the accumulated momentum term, referred to as the accumulated discount factor in [33], is cleared by the sparsification step, which leads to model performance deterioration.

Therefore, a momentum corrected version of the momentum SGD is used to guarantee that all the momentum terms are updated to be the same as if no sparsification is applied:

$$\boldsymbol{u}_{m,t} = \beta \boldsymbol{u}_{m,t-1} + \boldsymbol{g}_m(\boldsymbol{\theta}_t), \tag{2.6}$$

$$\boldsymbol{v}_{m,t} = \boldsymbol{v}_{m,t-1} + \boldsymbol{u}_{m,t}, \tag{2.7}$$

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \rho_t \sum_{m=1}^{M} sparse(\boldsymbol{v}_{m,t}), \tag{2.8}$$

that is, we apply the momentum before sparsification so that the momentum is always accumulated, as shown in eq. (2.6).

It was found in the simulations that momentum correction would help the training curves follow the vanilla momentum SGD more closely, especially for the cases where

learning rate ρ_t in the later training process is significantly larger than in the current iteration [41].

Other techniques that can improve the overall training performance are also explored. Local gradient clipping [42] is used to normalize the gradient distribution across each local device. Momentum factor masking [43] and warm-up training [44] are also used to stop momentum for delayed gradients, and make updates less aggressive in early stage of training, respectively. The experiment results show that momentum correction and local gradient clipping improve the local gradient accumulation, while momentum factor masking and warm-up training effectively eliminate the gradient staleness effect. Deep gradient compression therefore has similar training performance to that of the baseline model in terms of top-1 and top-5 accuracy, while having a much higher compression ratio in various training models for image classification, language modeling, and speech recognition.

Sparse Binary Compression

Sattler et al. proposed Sparse Binary Compression (SBC) in their work[45]. They consider a centralized distributed system model in which a central PS is used. SBC uses a variant of one-bit quantization: For each worker, after calculating the gradient vector in the current iteration, all entries whose absolute values pass a set threshold are selected and quantized to $\max\{\mu_+, \mu_-\}$, where μ_+ and μ_- are the mean values of the positive and negative entries that are selected, respectively. The remaining unselected entries are set to 0. Since all non-zero entries in the resulting vector have the same value, the resulting vector is transmitted to the PS by sending the mean values of the vector (which is already computed) and an indicator vector, which uses

value 1 to indicate the position of the non-zero entries. Such vector is encoded with Golomb encoding to minimize bits used for transmission. The entries that are not transmitted are accumulated for later transmission. Their experiment shows that with 4 workers, SBC is able to achieve over 2 times of compression rate compared to the conventional gradient dropping scheme while achieving similar top-1 error rate.

2.2.2 Analog-based transmission schemes

Broadband Analog Aggregation

Broadband Analog Aggregation (BAA) is a modified version of Orthogonal Frequency Division Multiple Access (OFDMA) scheme for federated learning, proposed by Zhu et al. in [35]. Inspired by the over-the-air computation technique [46], BAA uses amplitude modulations and exploits the waveform superposition property of multiple access channel. BAA divides the whole bandwidth to multiple orthogonal sub-channels/sub-carriers. However, unlike the conventional OFDMA, instead of assigning each sub-channel to each individual worker, BAA fully utilizes the broadband by dedicating each sub-channel to the transmission of one individual model parameter. Due to the waveform superposition, the PS receives the sum of parameters from each worker. This significantly reduces communication latency, as no summation and averaging operations are performed on the PS, and therefore the latency of BAA does not depend on the number of workers. This is achieved by making the following changes to conventional OFDMA scheme: BAA replaces the digital modulation with linear analog modulation, and adds truncated channel inversion at the transmitter side for signal amplitude alignments, while also guaranteeing that the transmission

power does not exceed a given power constraint. It also replaces the digital demodulator with a post-processing operator to recover the signal before scaling.

With fixed power constraint and broadband bandwidth, the receive Signal-to-Noise Ratio (SNR) is affected by two factors: the distance between the PS and the workers, and the truncate ratio regulated by the truncate threshold. For the former factor, since the signals from different workers are required to have the same amplitude (which is achieved by amplitude alignment), the signal amplitude depends heavily on the path loss of the furthest worker. To tackle this issue, BAA has two possibilities for scheduling: the PS either receives the transmissions from all workers (All-Inclusive Scheduling), or receives the ones within certain distance (Cell-Interior Scheduling). For the latter scheduling, since the transmissions from workers physically located further from the distance threshold are discarded, a tradeoff occurs. It is suggested that in a high-mobility network, where the workers can change their position rapidly over time, with proper increase to the communication rounds, cell-interior scheduling can still be effective. As for low-mobility networks, a hybrid scheduling scheme which alternates between cell-interior and all-inclusive scheduling can achieve a balance between the advantages and disadvantages of these two scheduling schemes.

As for the latter factor, the truncate threshold cannot be too high or too low: if it is too high, although the receive SNR is increased, so does the truncate ratio, which may degrade the learning performance as not enough parameters are used for training; if it is too low, the receive SNR is low as well, which brings inaccurate transmission result. In the experiment, the optimal truncate threshold is found by using a grid search.

Analog Distributed Stochastic Gradient Descent

In Amiri et al.'s work [47], they proposed the Analog Distributed SGD (A-DSGD), which can be considered as an extension of BAA. Different from BAA, A-DSGD first uses gradient dropping [39] for sparsification, and employs random projection and a compressed sensing algorithm for signal compression and reconstruction, respectively. It employs a random projection matrix $\boldsymbol{A}$, which projects the sparsified gradient vector to the channel bandwidth. They also scale the gradient vector with a constant $\alpha_{m,t}$ to satisfy the average power constraint of the m-th worker at iteration t. Then, both the scaled gradient vector and the square root of the scaling constant $\sqrt{\alpha_{m,t}}$ are transmitted to the PS without any digital modulation (i.e. analog transmission). Again, due to the superposition property of the Multiple Access Channel (MAC), the PS receives the summation $\boldsymbol{y}_t$ of the scaled compressed gradient vector of all workers, and the summation $\sqrt{\alpha_t}$ of power scaling constant of all workers. The PS then reconstructs the summation of unscaled sparse gradient by using approximate message passing algorithm (AMP) [48] on $\frac{\boldsymbol{y}_t}{\sqrt{\alpha_t}}$. However, due to the nature of compressed sensing, there is a trade-off between sparsity rate and reconstruction accuracy of the original signal: AMP would not reconstruct the signal accurately if the sparsity rate is too high, while the model would not be trained properly if the sparsity rate is too low.

Two power allocation schemes are proposed to decide how the power scaling constant $\alpha_{m,t}$ is chosen: Equal Power Allocation (EPA) and Unequal Power Allocation (UPA). In EPA, each worker scales its sparsified gradient vector with the same value, i.e. $\alpha_{m,t}$ is the same for all workers. In this case, $\alpha_{m,t}$ is known by both the PS and all workers, and therefore each worker does not need to transmit its $\alpha_{m,t}$ to the PS.

For UPA, $\alpha_{m,t}$ is calculated according to the transmission power constraint of each worker. Therefore, $\alpha_{m,t}$ is different for each worker and thus must be transmitted to the PS as well. To further reduce the transmission power consumption, it is also proposed to remove the mean value of the gradient vector $\mu_{m,t}$ before scaling, and transmit it to the PS. In this case the PS would receive the aggregated compressed gradient $\boldsymbol{y}_t$, the aggregated mean value of gradients μ_t, and the aggregated power scaling constant $\sqrt{\alpha_t}$ if UPA is used.

2.2.3 Digital-based transmissions vs. analog-based transmissions

Many research have shown that analog-based transmission schemes can achieve at least comparable training performance compared to the digital-based transmission schemes, and even outperform the latter in certain applications. Zhu et al. in [49] compared the training performance in terms of test accuracy and communication latency between the analog-based transmission scheme based on over-the-air computation [46] and the digital-based transmission scheme using conventional OFDMA modulation. They found that for a 4-layer convolutional neural network with two convolutional layers, with a single PS and 100 workers, the analog-based transmission scheme achieves a test accuracy similar to that of the digital-based transmission scheme, while reducing communication latency thanks to the channel superposition of the multiple access channel. Experiments on BAA show a similar result on both IID and non-IID dataset when compared to the digital-based transmission scheme using OFDMA modulation.

The above two research both assume that there are sufficient bandwidth resource, i.e. sufficient number of orthogonal subchannels, for both the digital-based and

analog-based transmission schemes. However, in Ameri et al.'s work [47], they find that when there is limited channel bandwidth, the performance of the conventional digital-based transmission such as SBC [45] deteriorates as the number of workers increases, which is not the case for A-DSGD. This is because for the digital-based transmission scheme, as the number of workers increases, the bandwidth/number of orthogonal subchannels allocated for each worker decreases. As a result, each worker sends less accurate training results, which induces performance deterioration. On the contrary, the performance of A-DSGD does not suffer from limited bandwidth as much as that of the digital-based transmission scheme, because each orthogonal sub-channel is dedicated for a single parameter instead of an individual worker, and it is beneficial for the analog-based transmission scheme to have more workers, since more workers bring better SNR of the aggregated signals.

Since limited bandwidth and on-device power consumption are common issues in practice such as mobile networks, we conclude that analog-based transmission schemes have overall better performance compared to the digital-based transmission schemes, and therefore in this thesis we focus only on comparing our proposed scheme to the analog-based transmission scheme.

Chapter 3

Hybrid Distributed Stochastic Gradient Descent

3.1 Baseline Analog Transmission Scheme

The baseline scheme we consider is similar to that of [47], that is, a synchronized distributed learning system with a central PS and M homogeneous workers using the FedAvg update scheme. The pseudo-code for the whole transmission scheme is described in Algorithm 1. In this scheme, each worker shares the same average power constraint P and the same training model (i.e. the neural network architecture) with the PS. At the beginning of the t^{th} round of training, the PS transmits the current trainable parameters $\boldsymbol{\theta}_t \in \mathbb{R}^d$, where d is the total number of trainable model parameters, and a random projection matrix $\boldsymbol{A}_t \in \mathbb{R}^{(s-1)\times d}$, whose elements are distributed by independent Gaussian $N(0, \frac{1}{s-1})$, to each worker. Here, s is the number of orthogonal sub-channels, where each one of them is dedicated for transmitting a single element in the gradient vector. We assume that the channel bandwidth is a scarce resouce, i.e. $s \ll d$. We also assume that each worker receives $\boldsymbol{A}_t$ and $\boldsymbol{\theta}_t$ without any errors; that is, the broadcasting channel from the PS to workers is assumed to be error-free. Then each worker trains its local model on local dataset

B_m and computes its local gradients $\boldsymbol{g}_m(\boldsymbol{\theta}_t) \in \mathbb{R}^d$ for $m = 1, 2, \cdots, M$.

Algorithm 1 Baseline Analog Transmission Scheme [47]

Initialize $\boldsymbol{\theta}_1 = 0$ and $\triangle_{1,0} = ... = \triangle_{M,0} = 0$
for $t = 1, ..., T$ **do**
  for PS **do**
    generate $\boldsymbol{A}_t \sim N(0, \frac{1}{s-1})$
  end for

  for Workers $m = 1, ..., M$ in parallel **do**
    Compute $\boldsymbol{g}_m(\boldsymbol{\theta}_t)$ w.r.t B_m
    $\boldsymbol{g}_m^{ec}(\boldsymbol{\theta}_t) = \boldsymbol{g}_m(\boldsymbol{\theta}_t) + \triangle_{m,t-1}$
    $\boldsymbol{g}_m^{sp}(\boldsymbol{\theta}_t) = sparse_k(\boldsymbol{g}_m^{ec}(\boldsymbol{\theta}_t))$
    $\triangle_{m,t} = \boldsymbol{g}_m^{ec}(\boldsymbol{\theta}_t) - \boldsymbol{g}_m^{sp}(\boldsymbol{\theta}_t)$ ▷ Local gradients accumulation
    $\tilde{\boldsymbol{g}}_m(\boldsymbol{\theta}_t) = \boldsymbol{A}_t \boldsymbol{g}_m^{sp}(\boldsymbol{\theta}_t)$
    $\alpha_{m,t} = \frac{P}{||\tilde{\boldsymbol{g}}_m(\boldsymbol{\theta}_t)||_2^2 + 1}$
    $\boldsymbol{x}_{m,t}(\boldsymbol{\theta}_t) = \sqrt{\alpha_{m,t}} \tilde{\boldsymbol{g}}_m(\boldsymbol{\theta}_t)$
  end for

  for PS **do**:
    $\hat{\boldsymbol{g}}(\theta_t) = AMP_{\boldsymbol{A}_t}(\frac{\boldsymbol{y}(\boldsymbol{\theta}_t)}{\sqrt{\bar{\alpha}_t}})$
    $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \rho_t \hat{\boldsymbol{g}}(\boldsymbol{\theta}_t)$
  end for
end for

To reduce the communication overhead, each worker first performs sparsification by using gradient dropping as follows [39]. Let $\triangle_{m,t} \in \mathbb{R}^d$ denote the accumulated error vector of gradients for worker m at iteration t, which stores the local gradients that are not transmitted in the previous rounds, and let $\boldsymbol{g}_m^{ec}(\boldsymbol{\theta}_t) = \boldsymbol{g}_m(\boldsymbol{\theta}_t) + \triangle_{m,t-1}$ be the error accumulated gradient vector. Also, let $k \in (0, 1]$ be the sparsity rate that is defined as the proportion of elements in a vector with non-zero values. First, each worker sets all elements of $\boldsymbol{g}_m^{ec}(\boldsymbol{\theta}_t)$ except the ones whose absolute values are among the largest $d \times k$ to zero. This gives the sparse gradient vector $\boldsymbol{g}_m^{sp}(\boldsymbol{\theta_t})$, which will be transmitted from worker m to the PS for training. Such sparsification scheme is

referred to as top-k selection in this work and is denoted by $sparese_k()$. The error vector $\triangle_{m,t}$ is then updated by $\triangle_{m,t} = \boldsymbol{g}_m^{ec}(\boldsymbol{\theta}_t) - \boldsymbol{g}_m^{sp}(\boldsymbol{\theta}_t)$.

We further reduce the communication overhead by finding a lower rank representation of $\boldsymbol{g}_m^{sp}(\boldsymbol{\theta}_t)$, denoted by $\tilde{\boldsymbol{g}}_m(\boldsymbol{\theta}_t)$. This is done by projecting $\boldsymbol{g}_m^{sp}(\boldsymbol{\theta}_t)$ to a smaller space with $\boldsymbol{A}_t$ [22], that is:

$$\tilde{\boldsymbol{g}}_m(\boldsymbol{\theta}_t) = \boldsymbol{A}_t \boldsymbol{g}_m^{sp}(\boldsymbol{\theta}_t) \in \mathbb{R}^{s-1}. \tag{3.1.1}$$

To ensure the transmit power is P, we scale $\tilde{\boldsymbol{g}}_m(\boldsymbol{\theta_t})$ with a power scaling coefficient. For worker m at iteration t, its power scaling coefficient $\alpha_{m,t}$ is computed as follows:

$$\alpha_{m,t} = \frac{P}{||\tilde{\boldsymbol{g}}_m(\boldsymbol{\theta}_t)||_2^2 + 1}, \tag{3.1.2}$$

where $||\cdot||_2$ is the l_2 norm. The number 1 in the denominator is used since we also need to send the information of $\alpha_{m,t}$ itself to the PS to recover $\tilde{\boldsymbol{g}}_m(\boldsymbol{\theta_t})$. Then we scale $\tilde{\boldsymbol{g}}_m(\boldsymbol{\theta}_t)$ with $\sqrt{\alpha_{m,t}}$, which gives $\boldsymbol{x}_{m,t}(\boldsymbol{\theta}_t) = \sqrt{\alpha_{m,t}}\tilde{\boldsymbol{g}}_m(\boldsymbol{\theta}_t)$.

Now we need to transmit both $\boldsymbol{x}_{m,t}(\boldsymbol{\theta}_t)$ and $\sqrt{\alpha_{m,t}}$ to the PS. We first use linear amplitude modulation to modulate $\boldsymbol{x}_{m,t}(\boldsymbol{\theta}_t)$ and $\sqrt{\alpha_{m,t}}$, then transmit them to the PS with suppressed-carrier transmission through a noisy multiple access channel (MAC), in the presence of Additive White Gaussian Noise (AWGN), $N(0, \sigma^2)$.

Let $\boldsymbol{z}_t^{s-1} \in \mathbb{R}^{s-1}$ denote the channel noise vector, whose elements are distributed by independent Gaussian $N(0, \sigma^2)$. Due to the superposition property of MAC, the

PS receives the aggregated scaled compressed signal $\boldsymbol{y}(\boldsymbol{\theta}_t) \in \mathbb{R}^{s-1}$ as follows:

$$\boldsymbol{y}(\boldsymbol{\theta}_t) = \sum_{m=1}^{M} \boldsymbol{x}_{m,t}(\boldsymbol{\theta}_t) + \boldsymbol{z}_t^{s-1}, \tag{3.1.3}$$

$$= \sum_{m=1}^{M} \sqrt{\alpha_{m,t}} \tilde{\boldsymbol{g}}_m(\boldsymbol{\theta}_t) + \boldsymbol{z}_t^{s-1}. \tag{3.1.4}$$

Recall that when using FedAvg, the PS needs to update the model with average aggregated gradients according to eq. (1.6), which in our case is the average of sparse gradient vector $\frac{1}{M}\sum_{m=1}^{M} \boldsymbol{g}_{m,t}^{sp}(\boldsymbol{\theta}_t)$. In order to reconstruct $\frac{1}{M}\sum_{m=1}^{M} \boldsymbol{g}_{m,t}^{sp}(\boldsymbol{\theta}_t)$, the PS then performs Approximated Message Passing (AMP) algorithm [48], a compressed sensing algorithm used for recovering sparse signals, which is described as follows:

Let $AMP_{\boldsymbol{A}}(\boldsymbol{y})$ denote the AMP algorithm with projection matrix $\boldsymbol{A}$ and input signal $\boldsymbol{y}$, where $\boldsymbol{A} \in \mathbb{R}^{a \times b}$ is the matrix used for projecting the original signal to a smaller space ($a \ll b$), and $\boldsymbol{y} \in \mathbb{R}^a$ is the compressed signal. AMP reconstructs the original sparse signal $\boldsymbol{x} \in \mathbb{R}^b$ by iteratively computing the following:

$$\tilde{\boldsymbol{x}}_{\tau+1} = \eta(\boldsymbol{A}^T \boldsymbol{r}_\tau + \tilde{\boldsymbol{x}}_\tau), \tag{3.1.5}$$

$$\boldsymbol{r}_{\tau+1} = \boldsymbol{y} - \boldsymbol{A}\tilde{\boldsymbol{x}}_{\tau+1} + \frac{1}{\delta}\boldsymbol{r}_\tau < \eta'(\boldsymbol{A}^T \boldsymbol{r}_\tau + \tilde{\boldsymbol{x}}_\tau) >, \tag{3.1.6}$$

where $\tilde{\boldsymbol{x}}_\tau$ is the estimated signal at AMP iteration τ, $\boldsymbol{A}^T$ denotes the transpose of $\boldsymbol{A}$, $\boldsymbol{r}_\tau \in \mathbb{R}^a$ is the residual term which is defined iteratively, and $\delta = \frac{a}{b}$ is the ratio between the dimensions of $\boldsymbol{x}$ and $\boldsymbol{y}$. Also, $\eta(\boldsymbol{\mu})$ is the soft threshold function applied

to input vector $\boldsymbol{\mu}$ component-wise:

$$\eta(\mu) = \begin{cases} \mu - \lambda & \text{if } |\mu| > \lambda, \\ 0 & \text{otherwise,} \end{cases} \tag{3.1.7}$$

where λ is the threshold parameter. Furthermore, $< \cdot >$ denotes vector mean, and $\eta'(\boldsymbol{\mu})$ is the component-wise first order derivative of $\eta(\boldsymbol{\mu})$. The process halts either after a fixed number of reconstruction iterations, or whenever the power of residual $||\boldsymbol{r}_\tau||_2^2$ is sufficiently small. AMP requires $O(d)$ iterations for reconstruction, and its per-iteration computation complexity is $O(d^2)$.

In our case, we reconstruct $\frac{1}{M}\sum_{m=1}^{M} \boldsymbol{g}_{m,t}^{sp}(\boldsymbol{\theta}_t)$ by computing $AMP_{\boldsymbol{A}_t}(\frac{\boldsymbol{y}(\boldsymbol{\theta}_t)}{\sqrt{\bar{\alpha}_t}})$, i.e. the AMP algorithm with random projection matrix $\boldsymbol{A}_t$ generated at the beginning of t^{th} iteration, and input $\frac{\boldsymbol{y}(\boldsymbol{\theta}_t)}{\sqrt{\bar{\alpha}_t}}$, where

$$\sqrt{\bar{\alpha}_t} = \sum_{m=1}^{M} \sqrt{\alpha_{m,t}} + z_t \tag{3.1.8}$$

is the aggregated power scaling coefficient the PS receives.

Then the estimation $\hat{\boldsymbol{g}}_t(\boldsymbol{\theta}_t)$ is produced by the AMP algorithm. Then, the PS updates the model with $\hat{\boldsymbol{g}}_t(\boldsymbol{\theta}_t)$ and proceeds to the next round of training. That is, the PS first updates the model parameters by

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \rho_t \hat{\boldsymbol{g}}_t(\boldsymbol{\theta}_t), \tag{3.1.9}$$

and then broadcasts $\boldsymbol{\theta}_{t+1}$ to all workers. After receiving the updated model parameters $\boldsymbol{\theta}_{t+1}$, each worker locally updates it using its own local data set.

3.2 Motivation

In this section, we discuss the fundamental motivation for the proposed scheme. We first note that $\frac{\boldsymbol{y}(\boldsymbol{\theta}_t)}{\sqrt{\bar{\alpha}_t}}$, which is the input to the AMP algorithm, can be written as:

$$\frac{\boldsymbol{y}(\boldsymbol{\theta}_t)}{\sqrt{\bar{\alpha}_t}} = \frac{\sum_{m=1}^{M} \boldsymbol{x}_{m,t}(\boldsymbol{\theta}_t) + \boldsymbol{z}_t^{s-1}}{\sum_{m=1}^{M} \sqrt{\alpha_{m,t}} + z_t}, \tag{3.2.1}$$

$$= \frac{\sum_{m=1}^{M} \sqrt{\alpha_{m,t}} \tilde{\boldsymbol{g}}_m(\boldsymbol{\theta}_t)}{\sum_{m=1}^{M} \sqrt{\alpha_{m,t}} + z_t} + \frac{\boldsymbol{z}_t^{s-1}}{\sum_{m=1}^{M} \sqrt{\alpha_{m,t}} + z_t}, \tag{3.2.2}$$

where the first and second terms in eq. (3.2.2) can be considered as the signal and noise components, respectively. From this equation, we can make two important observations. First, when the transmit power constraint P is large, $\sqrt{\alpha_{m,t}}$ will be large according to eq. (3.1.2). In this case, the second term becomes small, meaning that the noise becomes small. As for the first term, if P is sufficiently large such that the summation of $\sqrt{\alpha_{m,t}}$ dominates in the denominator, the first term approximates the average of the lower rank representation of the original sparse signal, i.e. $\frac{1}{M}\sum_{m=1}^{M} \tilde{\boldsymbol{g}}_{m,t}(\boldsymbol{\theta}_t)$, when d and $|B_m|$ are sufficiently large [47]. Overall, the larger P, the more accurate the reconstruction of the average of aggregated sparse gradient, i.e. $\frac{1}{M}\sum_{m=1}^{M} \boldsymbol{g}_{m,t}^{sp}(\boldsymbol{\theta}_t)$. The fact that stronger transmit power gives accurate reconstruction might not sound very surprising; but, the next observation is somewhat interesting.

Second, when the number of workers M is larger, the second term becomes smaller, meaning that the noise becomes smaller. Meanwhile, the first term again approximates $\frac{1}{M}\sum_{m=1}^{M} \tilde{\boldsymbol{g}}_{m,t}(\boldsymbol{\theta}_t)$ when M is sufficiently large, because the summation in the denominator dominates over the noise z_t. That is, the larger M, the more accurate the reconstruction of the original signal, i.e. the average of aggregated sparse gradient

$\frac{1}{M}\sum_{m=1}^{M} \boldsymbol{g}_{m,t}^{sp}(\boldsymbol{\theta}_t)$.

Thus, at the PS, the reliability of reconstructing $\frac{1}{M}\sum_{m=1}^{M} \boldsymbol{g}_{m,t}^{sp}(\boldsymbol{\theta}_t)$ depends on the values of both M and P. Fig 3.1a shows that when M and/or P are small, the training performance deteriorates. The simulation In this simulation, the distributed learning model is a three layers neural network. We trained it on the MNIST handwritten digit dataset, which is comprised of 60,000 training samples and 10,000 testing samples, where each sample is a black and white 28×28 image [50]. This results in $d = 7850$ trainable parameters. We fixed $s = 0.05d$, $k = 0.04$, $SNR = 5$ dB, and set rest of the configurations the same as in Section 4.1, where the SNR is defined by:

$$\text{SNR} = \frac{P}{\sigma^2}. \tag{3.2.3}$$

In Figures 3.1 and 3.2, the green dashed curve depicts the test accuracy of the model where the received value of aggregated power scaling coefficient is not distorted by channel noise, i.e. there is no noise term z_t in eq. (3.1.8); and the solid blue curve represents the test accuracy with the distorted value of aggregated power scaling coefficient, i.e. the noise term z_t exists and is non-zero in eq. (3.1.8). It can be seen that in the former case, the training performance in terms of final test accuracy is improved by 6% compared to the latter. Even with relatively high SNR, small value of M still has significant impact on training performance, as seen in Fig. 3.1b. With $SNR = 15$ dB and a single worker, although in both cases the training performance is improved, the case where the received value of aggregated power scaling coefficient is noiseless still significantly outperforms the other case with over 15% of improvement in terms of the final test accuracy.

With SNR fixed, increasing M boosts the training performance for both cases, as

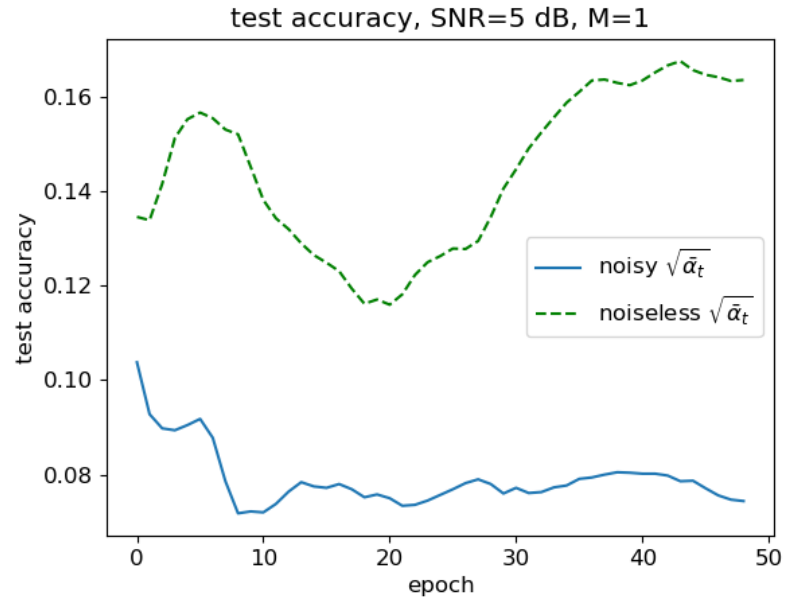
shown in Fig. 3.2. Indeed, when the value of accumulated power scaling coefficient is distorted by channel noise, greater M gives a greater summation in the denominator of the second term in eq. (3.2.2), resulting in smaller noise, leading to more accurate reconstruction, and therefore better training performance. However, when the received value of aggregated power scaling coefficient is noiseless, the model benefits from significantly more boost in training performance.

In the real world applications, especially in mobile network scenario, there is no guarantee that there would be a sufficient number of workers participating the training process, i.e. the value of M can be small. Furthermore, because each participating worker (e.g. wireless device) may have limited power available, the SNR is often low. Therefore, it is necessary to consider improving system performance under such circumstances.

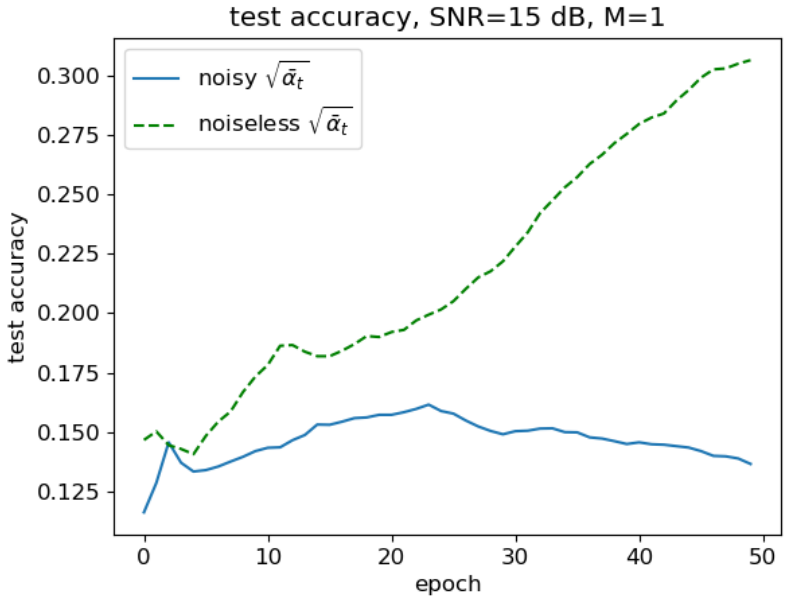
3.3 The hybrid DSGD

In the last section, we demonstrated that the noise added to the power scaling coefficient degrades the training performance, especially when SNR is low and/or the number M of workers small. In order to address this issue, it is important for each worker to reliably transmit the power scaling coefficient to the PS such that the received power scaling coefficient has small or no errors. To this end, in this paper, instead of using pure analog transmission, we propose using a hybrid of digital and analog transmissions, where the digital part of the transmission is dedicated for transmitting the important power scaling coefficient $\sqrt{\alpha_{m,t}}$. We name this scheme the hybrid DSGD.

More specifically, each worker first modulates its power scaling coefficient $\sqrt{\alpha_{m,t}}$

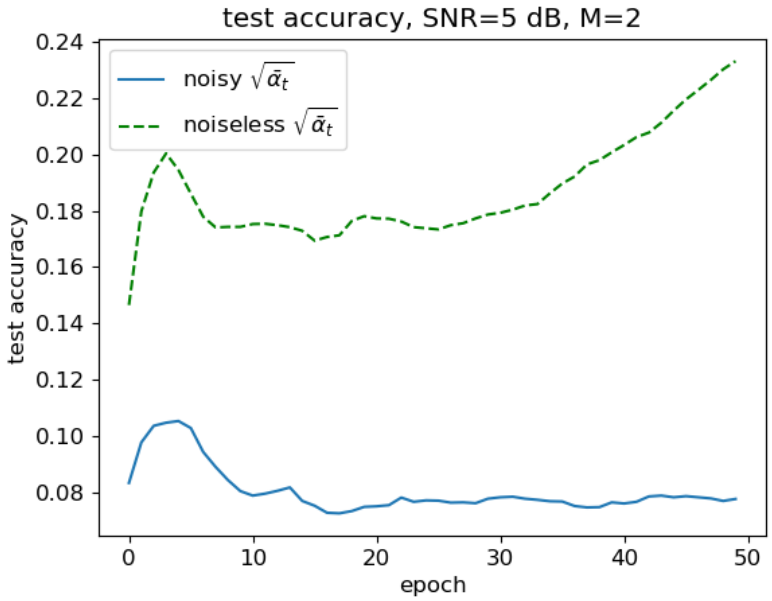


(a) SNR=5 dB

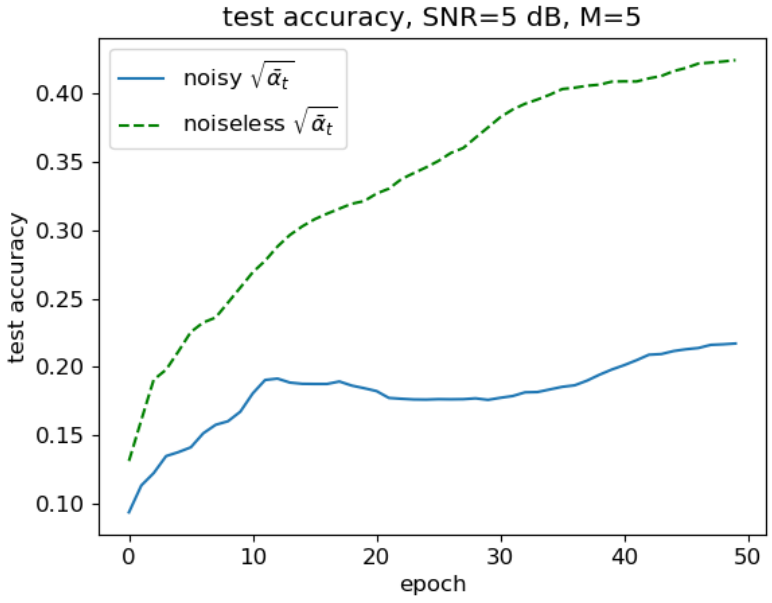


(b) SNR=15 dB

Figure 3.1: Test accuracy of models with noiseless aggregated power scaling coefficient (green, dahsed) and noisy aggregated power scaling coefficient (blue, solid) received at the PS, respectively, with fixed number of workers $M = 1$ and AWGN $N(0, 1)$.



(a) M=2



(b) M=5

Figure 3.2: Test accuracy of models with noiseless aggregated power scaling coefficient (green, dahsed) and noisy aggregated power scaling coefficient (blue, solid), with AWGN $N(0, 1)$ and fixed signal-to-noise ratio SNR = 5 dB.

using the modulation scheme adopted for the communication, then transmits it in worker's designated time slots one at a time. That is, each $\sqrt{\alpha_{m,t}}$ is transmitted by a time-division digital communications. This will induce a communication latency which scales linearly with the number of workers. Note that other techniques such as error coding can be applied to reinforce signal's resilience to the channel noise.

After all values of $\sqrt{\alpha_{m,t}}$ are transmitted, all workers then simultaneously transmit the compressed signal $\boldsymbol{x}_{m,t}(\boldsymbol{\theta}_t)$ with the same analog transmission scheme as that of the baseline scheme. Fig. 3.3 shows the architecture comparison between the baseline analog transmission scheme and the proposed hybrid transmission scheme, in which $\boldsymbol{x}_{m,t}^{[i]}$ denotes the i^{th} element of $\boldsymbol{x}_{m,t}(\boldsymbol{\theta}_t)$. Since computation for summation of power scaling coefficient has complexity of $O(M)$, the whole system can still be considered to have low latency.

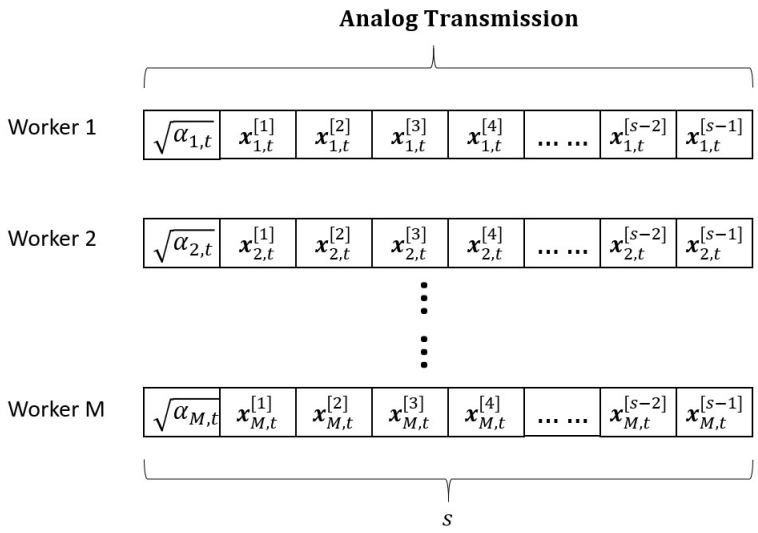
Let the total time duration for uplink transmission in each communication round be $\mathcal{T}$, and $\beta \in (0, 1)$ be the proportion of time duration for digital transmission. Then, in the proposed scheme, the total transmit energy and the average transmit power of each worker are given by

$$\mathcal{T}P = \beta\mathcal{T}P + (1-\beta)\mathcal{T}P, \tag{3.3.1}$$

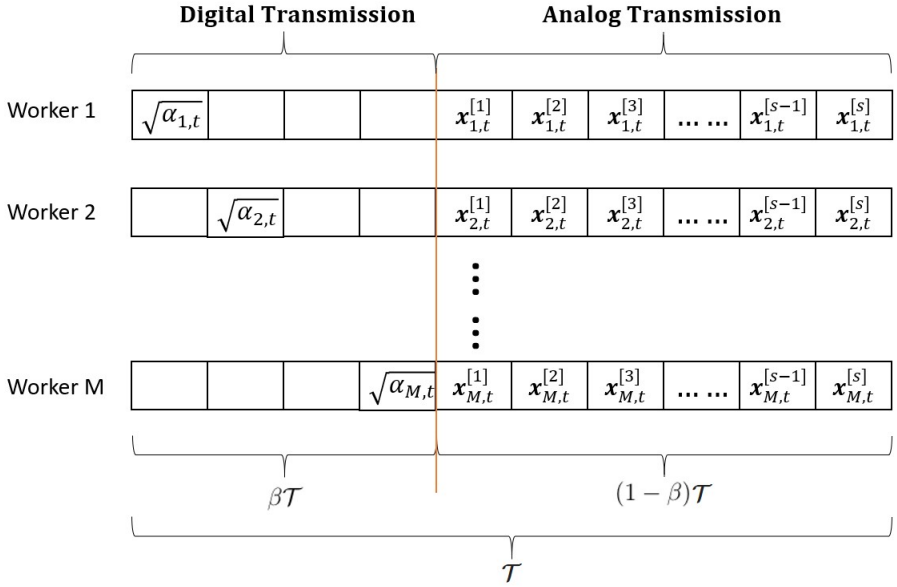
$$P = \beta P + (1-\beta)P, \tag{3.3.2}$$

$$= P_d + P_a, \tag{3.3.3}$$

where $P_d = \beta P$ and $P_a = (1-\beta)P$ are the average power consumptions of each worker for digital and analog transmissions, respectively. We now define the SNR of



(a) Baseline Analog Transmission Scheme



(b) Hybrid DSGD

Figure 3.3: Figures demonstrating the architecture differences between the baseline analog transmission scheme (top) and proposed hybrid transmission scheme (bottom). The orange line in the diagram of proposed hybrid scheme indicates the split point between digital (left) and analog (right) transmission segments.

the aggregated compressed analog signal considering all workers as follows:

$$SNR_a = \frac{MP_a}{\sigma^2}, \tag{3.3.4}$$

$$= \frac{M(1-\beta)P}{\sigma^2}, \tag{3.3.5}$$

which is a useful performance measure because it accommodates both the SNR (i.e. $\frac{P}{\sigma^2}$) of each worker and the number of all workers for analog transmission of gradients.

Assuming each worker has the same length of time slot for digital transmission, then each worker has $\frac{\beta\mathcal{T}}{M}$ of time to transmit its own digital signal, i.e. the value of $\sqrt{\alpha_{m,t}}$. Therefore, maintaining the same total energy consumption as the baseline analog scheme, each worker can use up to $M \times P$ of power to transmit the digital signal such that $MP \times \frac{\beta\mathcal{T}}{M} = \beta\mathcal{T}P$, which also guarantees worker's average power consumption being P. Also, we assume each worker quantizes the value of its power scaling coefficient $\sqrt{\alpha_{m,t}}$ to a q-bits binary number, then the energy per bit would be $E_b = \frac{\beta\mathcal{T}P}{q}$. Since AWGN is a zero mean and wide-sense stationary process, it has two-sided power spectral density $\frac{N_0}{2}$ for all frequencies, where N_0 is the noise power per unit bandwidth, whereas its variance is $\sigma^2 = \frac{N_0}{2}$ [51].

In order to consider the errors induced to the q bits used to transmit the value of $\sqrt{\alpha_{m,t}}$, the Bit Error Rate (BER) must be computed considering the ratio E_b/N_0 and the modulation scheme adopted for communications. For example, when Quadrature

Phase Shift Keying (QPSK) is used, the BER is given by [52]:

$$BER = Q\left(\sqrt{\frac{2 \times E_b}{N_0}}\right), \tag{3.3.6}$$

$$= Q\left(\sqrt{\frac{2 \times \frac{\beta \mathcal{T} P}{q}}{2\sigma^2}}\right), \tag{3.3.7}$$

$$= Q\left(\sqrt{\frac{\beta \mathcal{T} P}{q\sigma^2}}\right), \tag{3.3.8}$$

where $Q(\cdot)$ is the Q-function. When other modulations schemes (e.g., M-ary Phase Shift Keying (PSK) or square/rectangular M-ary Quadrature Amplitude Modulation (QAM)) are used, the bit error rates can also be determined [53]–[55]. Other than increasing M or P, the BER can be improved by increasing the value of β, increasing the total transmission time $\mathcal{T}$, or employ error control coding.

Note that there exists a trade-off between the BER for digital signal and the SNR, SNR_a, for analog signal with given $\mathcal{T}$: if β is small, the BER for each $\sqrt{\alpha_{m,t}}$ will be high according to eq. (3.3.8), while SNR_a will be high according to eq. (3.3.5); on the other hand, if β is large, the BER for each $\sqrt{\alpha_{m,t}}$ will be low, while SNR_a will be low. Therefore, in order to optimize the performance, we need to find the largest β such that the BER for digital signal is small enough to reliably transmit the power scaling coefficients to the PS, and SNR_a is high enough to allow the compressed signal to be reliably reconstructed at the PS, while minimizing $\mathcal{T}$ to maintain low communication latency.

The pseudo-code of the proposed hybrid transmission scheme is shown in Algorithm 2. Here, $floatToBin()$ and $binToFloat()$ are the functions which perform conversions between float and binary number as their names suggest.

Algorithm 2 Hybrid DSGD

Initialize $\boldsymbol{\theta}_1 = 0$ and $\triangle_{1,0} = ... = \triangle_{M,0} = 0$
for $t = 1, ..., T$ **do**
  for PS **do**
    generate $\boldsymbol{A}_t \sim N(0, \frac{1}{s-1})$
  end for

  for Workers $m = 1, ..., M$ in parallel **do**
    Compute $\boldsymbol{g}_m(\boldsymbol{\theta}_t)$ w.r.t B_m
    $\boldsymbol{g}_m^{ec}(\boldsymbol{\theta}_t) = \boldsymbol{g}_m(\boldsymbol{\theta}_t) + \triangle_{m,t-1}$
    $\boldsymbol{g}_m^{sp}(\boldsymbol{\theta}_t) = sparse_k(\boldsymbol{g}_m^{ec}(\boldsymbol{\theta}_t))$
    $\triangle_{m,t} = \boldsymbol{g}_m^{ec}(\boldsymbol{\theta}_t) - \boldsymbol{g}_m^{sp}(\boldsymbol{\theta}_t)$ ▷ Local gradients accumulation
    $\tilde{\boldsymbol{g}}_m(\boldsymbol{\theta}_t) = \boldsymbol{A}_t \boldsymbol{g}_m^{sp}(\boldsymbol{\theta}_t)$
    $\alpha_{m,t} = \frac{P}{||\tilde{\boldsymbol{g}}_m(\boldsymbol{\theta}_t)||_2^2 + 1}$
    $\boldsymbol{x}_{m,t}(\boldsymbol{\theta}_t) = \sqrt{\alpha_{m,t}} \tilde{\boldsymbol{g}}_m(\boldsymbol{\theta}_t)$
    $bin_{m,t} = floatToBin(\sqrt{\alpha_{m,t}})$
  end for

  for PS **do**:
    for $m = 1, ..., M$ in parallel **do**
      $\sqrt{\hat{\alpha}_{m,t}} = binToFloat(\bar{bin}_{m,t})$
    end for
    $\sqrt{\bar{\alpha}_t} = \sum_{m=1}^{M} \sqrt{\hat{\alpha}_{m,t}}$
    $\hat{\boldsymbol{g}}(\theta_t) = AMP_{\boldsymbol{A}_t}(\frac{\boldsymbol{y}(\boldsymbol{\theta}_t)}{\sqrt{\bar{\alpha}_t}})$
    $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \rho_t \hat{\boldsymbol{g}}(\boldsymbol{\theta}_t)$
  end for
end for

Chapter 4

Experiment Results

4.1 Experiment Setup

For the experiments, we ran all the simulations on Tensorflow 2.x. The experiment setup is similar as that of [47]: a three layers neural network is trained on the MNIST dataset [50] with fixed learning rate $\rho_t = 0.01, \forall t$, mini-batch size 32, and the ADAM optimizer [56] for 50 epochs, resulting in $d = 7850$ of trainable model parameters. A random dropout layer with dropout rate 0.5 is added after the hidden layer to prevent overfitting. We set $s = 0.05d$ and $k = 0.04$. With such small sparsity rate, we expect the training performance for both scheme would not be high, especially when the number of workers M is small. Prior to the training process , the training dataset is shuffled and then equally divided into M subsets, and each subset is assigned to a distinct worker. The channel noise is set to be distributed by AWGN $N(0, 1)$. For the proposed scheme, we convert the float values in the sparse gradient vector to 32-bits IEEE-754 binary number[57] and use QPSK modulation. The total uplink transmission duration $\mathcal{T}$ is adopted to guarantee that the BER for digital signal is lower than 0.05. Two sets of experiments with different setups are conducted.

4.2 First set of experiments

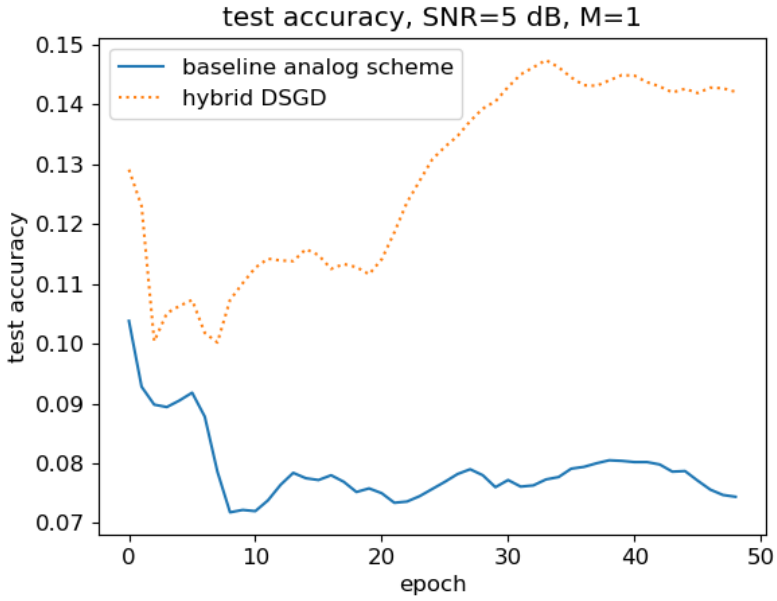
In the first set of experiments, we wish to see how the proposed hybrid transmission scheme improves the performance of training compared to the baseline analog transmission scheme with increasing M when SNR is low. We trained the neural network with $M = [1, 2, 5, 10, 20]$, SNR $= 5$ dB and $\beta = 1 \times 10^{-4}$. We plot the test accuracy versus training epoch curves of the baseline analog scheme and the proposed hybrid scheme during training, as shown in Fig. 4.1, where the orange dotted curves represent the ones for the proposed hybrid scheme, and the blue solid curves depict the ones for the baseline analog scheme. With small number of workers, the proposed hybrid scheme provides significant performance boost over the baseline analog scheme, as shown in Figs. 4.1a–4.1d. However, when M is sufficiently large, the SNR for aggregated power scaling coefficient in the baseline scheme is sufficiently large so that the PS is now able to reliably reconstruct the signal, resulting in similar performance compared to the proposed hybrid scheme as expected, as shown in Fig. 4.1e.

4.3 Second sets of experiments

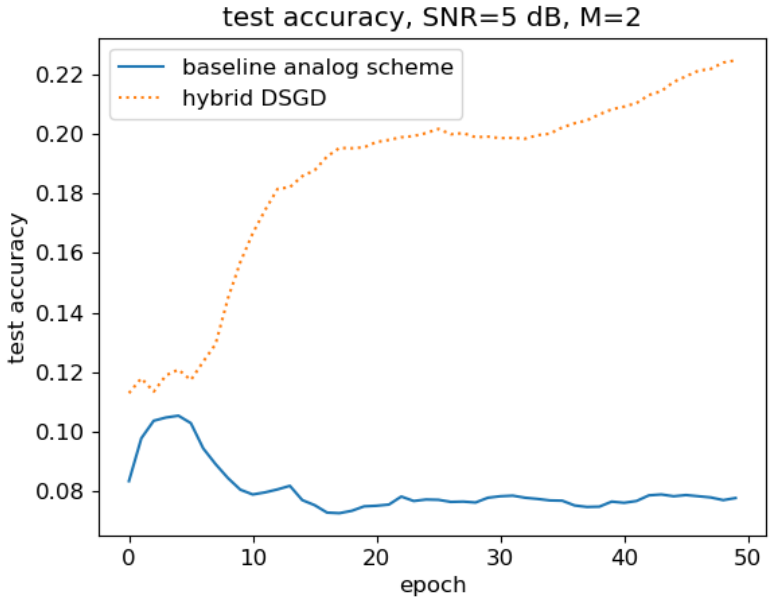
From the first set of experiments, we saw that the proposed hybrid scheme outperforms the baseline analog scheme up to certain value of M. In the second set of the experiments, we wish to find such critical value of M with different average power constraint. In the second set of experiments, the neural network is trained with different values of β. The largest β that provides the best training performance is found by running simulations starting from $\beta = 0.1$, then exponentially reducing this value until the model performance is no longer improving. The model is trained with

$M = [1, 2, ..., 20]$ and $SNR = [0 \text{ dB}, 5 \text{ dB}, 10 \text{ dB}, 15 \text{ dB}]$. We consider the Top-1 final test accuracy as the performance evaluation metric. Also, early stopping is employed such that if the test accuracy is not improved for 10 epochs during training, the training halts and the final test accuracy would be the highest test accuracy so far. The experiment results are shown in Figs. 4.2a–4.2d, in which we plot the performance of the baseline analog scheme and the proposed hybrid scheme with the best performance and the largest β. It can be seen that when the SNR is low, the proposed hybrid scheme outperforms the baseline analog scheme, while as M increases, the performance boosts by using the proposed hybrid scheme diminish, which matches the observations from the previous set of experiments. As each worker has increasingly higher SNR, the SNR for the power scaling coefficient is significantly improved for the baseline analog scheme, leading to a fast decrease in the performance advantages for the proposed hybrid scheme. Therefore, it is expected that after passing a critical value of M, the baseline analog scheme would outperform our proposed scheme, as now the baseline analog scheme is now able to reliably transmits the power scaling coefficients, while also have higher SNR for transmitting the scaled compressed gradient signal.

Another observation is that as the SNR increases, the largest value of β which provides the best training performance in terms of final test accuracy increases from $\beta = 1 \times 10^{-4}$ to $\beta = 5 \times 10^{-4}$, which is as expected since the SNR for the analog signal, i.e. SNR_a, is also improved, allowing more power being allocated for the digital transmission.

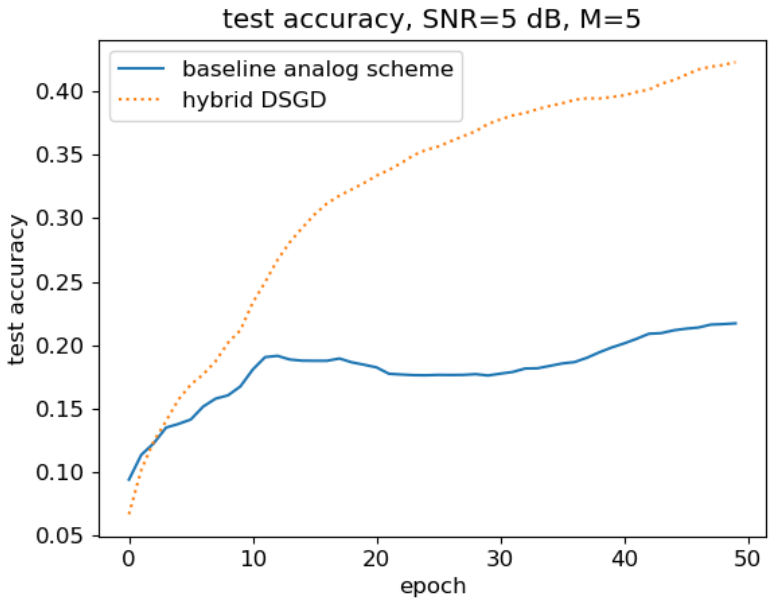


(a) M=1

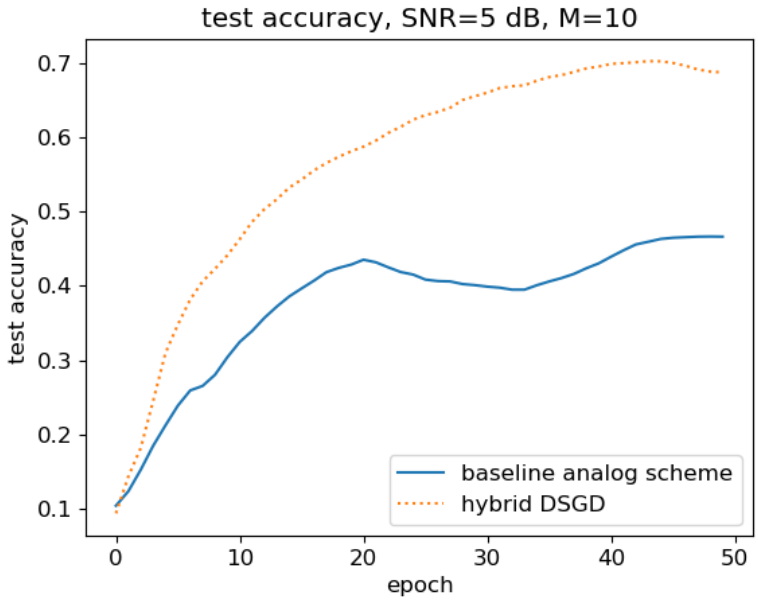


(b) M=2

Figure 4.1: Test accuracy of the proposed hybrid scheme (orange, dotted) and the baseline analog scheme (blue, solid), with fixed signal-to-noise ratio SNR $= 5dB$

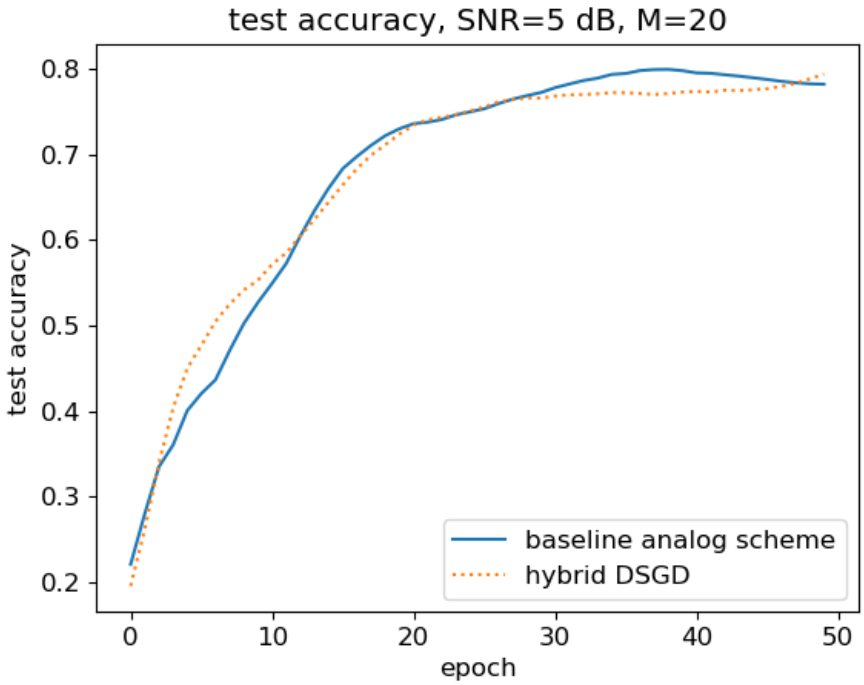


(c) M=5



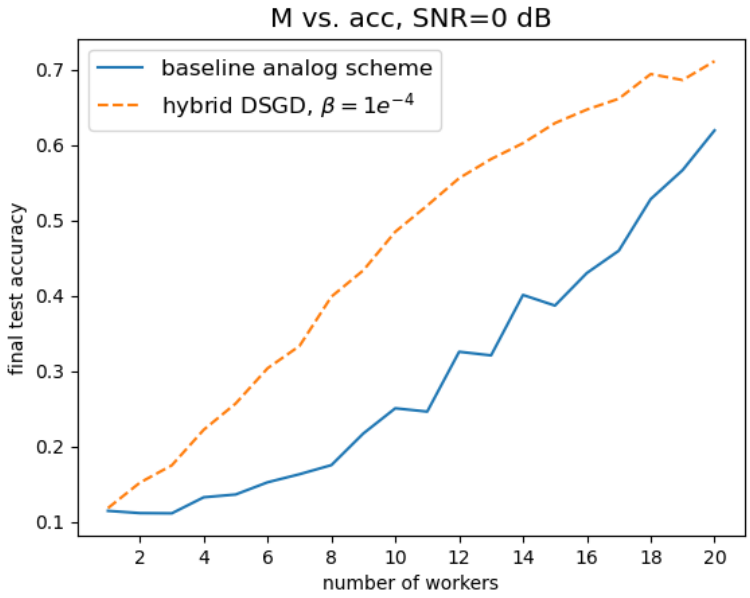
(d) M=10

Figure 4.1: Continued

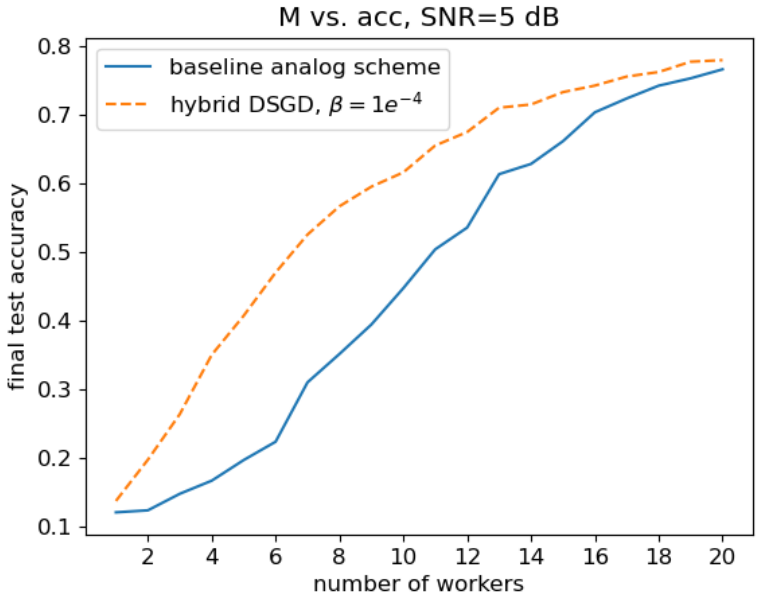


(e) M=20

Figure 4.1: Continued

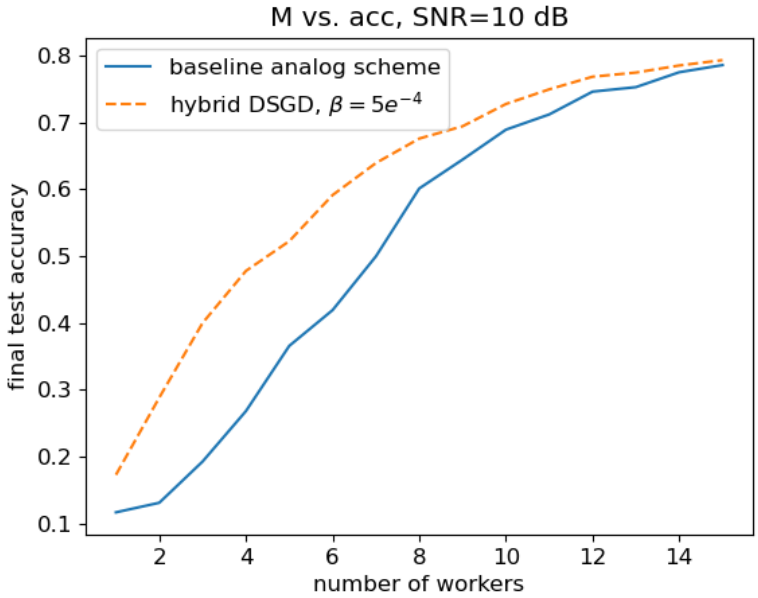


(a) SNR=0dB

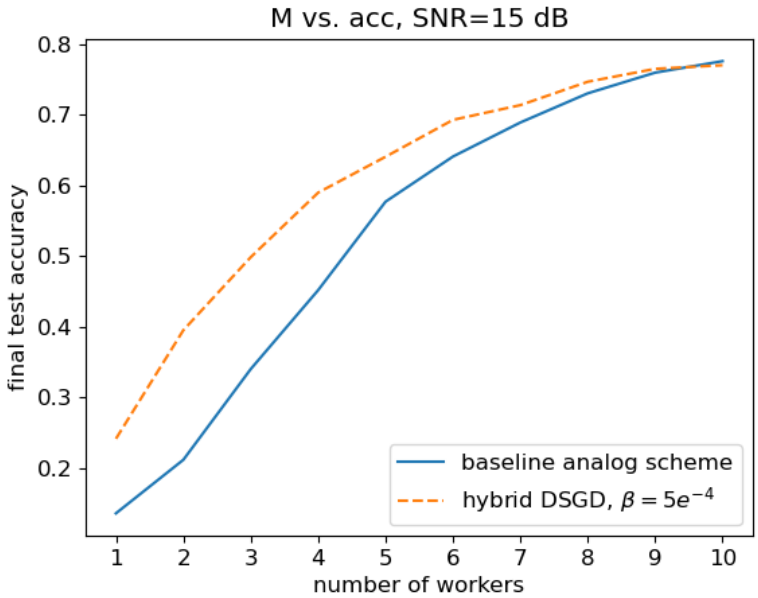


(b) SNR=5dB

Figure 4.2: Number of workers versus final test accuracy comparison between the proposed hybrid scheme (orange, dotted) and the baseline analog scheme (blue, solid). Channel bandwidth s and local sparsity k are fixed with $0.05d$, $0.04d$, respectively.



(c) SNR=10dB



(d) SNR=15dB

Figure 4.2: Continued

Chapter 5

Conclusion and Future Work

In this work, a synchronized federated learning model involving a central parameter server governing the global model and multiple homogeneous workers (i.e. local devices) dedicated for computing model updates corresponding to their local datasets is considered. In the scenario where the workers have limited bandwidth resource and power constraints, an analog approach for compressing and transmitting model updates to reduce communication overhead was studied in the literature, and it was demonstrated that the pure analog approach significantly outperforms the pure digital approach. In this thesis, however, we demonstrated that when the number of workers and power constraint are both low, such pure analog scheme could not reconstruct the compressed signal accurately, resulting in poor model performance. Such performance deterioration is caused by the low SNR of the power scaling coefficient, which scales the compressed model updates to fit the power constraint on each worker. It was found through experiments that the model would have significant performance improvement if the power scaling coefficient can be reconstructed without any error at the parameter server.

To tackle this issue, we proposed a modified version of the pure analog transmission

scheme, which utilizes digital encoding for transmitting the important power scaling coefficient of each worker, while still using analog transmission for transmitting compressed signal of the model updates. Each worker individually transmits its power scaling coefficient in a time-division manner with a digital modulation, and then all workers simultaneously transmit their analog signals without any digital modulation to the parameter server. Within a single transmission time frame, different length of time slot for transmitting digital signal results in different combinations of BER and SNR for the digital and analog signals, respectively. It was found through experiments that with proper setting, the proposed transmission scheme would drastically improve model performance when the number of workers is small and the power constraint on each worker is low.

In this work, we only considered AWGN channel. Although AWGN channel is a simple mathematical model and does not reflect the complex real-world communication environment, it is easy for simulations and system analysis, and it can be extended to a complex channel in the future, such as a fast fading channel with path loss and line-of-sight components, which better simulates the real-world communication environment. In the future research, other than simulating a more complex channel, a deeper neural network architecture and a more complex training dataset should be considered.

Bibliography

[1] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[2] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A Fast Learning Algorithm for Deep Belief Nets," *Neural Comput.*, vol. 18, no. 7, 1527–1554, Jul. 2006, ISSN: 0899-7667. DOI: 10.1162/neco.2006.18.7.1527. [Online]. Available: https://doi.org/10.1162/neco.2006.18.7.1527.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Commun. ACM*, vol. 60, no. 6, 84–90, May 2017, ISSN: 0001-0782. DOI: 10.1145/3065386. [Online]. Available: https://doi.org/10.1145/3065386.

[4] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv 1409.1556*, Sep. 2014.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *CoRR*, vol. abs/1512.03385, 2015. arXiv: 1512.03385. [Online]. Available: http://arxiv.org/abs/1512.03385.

[6] R. Collobert and J. Weston, "A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning," in *Proceedings*

of the 25th International Conference on Machine Learning, ser. ICML '08, Helsinki, Finland: Association for Computing Machinery, 2008, 160–167, ISBN: 9781605582054. DOI: 10.1145/1390156.1390177. [Online]. Available: https://doi.org/10.1145/1390156.1390177.

[7] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A Convolutional Neural Network for Modelling Sentences," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Baltimore, Maryland: Association for Computational Linguistics, Jun. 2014, pp. 655–665. DOI: 10.3115/v1/P14-1062. [Online]. Available: https://www.aclweb.org/anthology/P14-1062.

[8] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," *CoRR*, vol. abs/1409.3215, 2014. arXiv: 1409.3215. [Online]. Available: http://arxiv.org/abs/1409.3215.

[9] J. An and S. Cho, "Variational Autoencoder based Anomaly Detection using Reconstruction Probability," 2015. [Online]. Available: http://dm.snu.ac.kr/static/docs/TR/SNUDM-TR-2015-03.pdf10:26PM6/28/2020.

[10] O. Abdel-Hamid, A. Mohamed, H. Jiang, and G. Penn, "Applying Convolutional Neural Networks concepts to hybrid NN-HMM model for speech recognition," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 4277–4280.

[11] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four

Research Groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[12] A. Graves and N. Jaitly, "Towards End-to-End Speech Recognition with Recurrent Neural Networks," in *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ser. ICML'14, Beijing, China: JMLR.org, 2014, II–1764–II–1772.

[13] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen, J. Chen, J. Chen, Z. Chen, M. Chrzanowski, A. Coates, G. Diamos, K. Ding, N. Du, E. Elsen, J. Engel, W. Fang, L. Fan, C. Fougner, L. Gao, C. Gong, A. Hannun, T. Han, L. Johannes, B. Jiang, C. Ju, B. Jun, P. LeGresley, L. Lin, J. Liu, Y. Liu, W. Li, X. Li, D. Ma, S. Narang, A. Ng, S. Ozair, Y. Peng, R. Prenger, S. Qian, Z. Quan, J. Raiman, V. Rao, S. Satheesh, D. Seetapun, S. Sengupta, K. Srinet, A. Sriram, H. Tang, L. Tang, C. Wang, J. Wang, K. Wang, Y. Wang, Z. Wang, Z. Wang, S. Wu, L. Wei, B. Xiao, W. Xie, Y. Xie, D. Yogatama, B. Yuan, J. Zhan, and Z. Zhu, "Deep Speech 2 : End-to-End Speech Recognition in English and Mandarin," in *Proceedings of The 33rd International Conference on Machine Learning*, M. F. Balcan and K. Q. Weinberger, Eds., ser. Proceedings of Machine Learning Research, vol. 48, New York, New York, USA: PMLR, 2016, pp. 173–182. [Online]. Available: http://proceedings.mlr.press/v48/amodei16.html.

[14] D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3642–3649.

[15] Y. Huang, Y. Cheng, D. Chen, H. Lee, J. Ngiam, Q. V. Le, and Z. Chen, "GPipe: Efficient Training of Giant Neural Networks using Pipeline Parallelism," *CoRR*, vol. abs/1811.06965, 2018. arXiv: 1811.06965. [Online]. Available: http://arxiv.org/abs/1811.06965.

[16] A. Harlap, D. Narayanan, A. Phanishayee, V. Seshadri, N. R. Devanur, G. R. Ganger, and P. B. Gibbons, "PipeDream: Fast and Efficient Pipeline Parallel DNN Training," *CoRR*, vol. abs/1806.03377, 2018. arXiv: 1806.03377. [Online]. Available: http://arxiv.org/abs/1806.03377.

[17] *MPI: A Message-Passing Interface StandardVersion 3.1*, https://www.mpi-forum.org/docs/mpi-3.1/mpi31-report.pdf, 2015.

[18] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated Machine Learning: Concept and Applications," *CoRR*, vol. abs/1902.04885, 2019. arXiv: 1902.04885. [Online]. Available: http://arxiv.org/abs/1902.04885.

[19] Q. Li, Z. Wen, and B. He, "Federated Learning Systems: Vision, Hype and Reality for Data Privacy and Protection," *ArXiv*, vol. abs/1907.09693, 2019.

[20] H. B. McMahan, E. Moore, D. Ramage, and B. A. y Arcas, "Federated Learning of Deep Networks using Model Averaging," *CoRR*, vol. abs/1602.05629, 2016. arXiv: 1602.05629. [Online]. Available: http://arxiv.org/abs/1602.05629.

[21] B. McMahan and D. Ramage, *Federated Learning: Collaborative Machine Learning without Centralized Training Data*, https://ai.googleblog.com/2017/04/federated-learning-collaborative.html, 2017.

[22] J. Konecný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated Learning: Strategies for Improving Communication Efficiency," *CoRR*, vol. abs/1610.05492, 2016. arXiv: 1610.05492. [Online]. Available: http://arxiv.org/abs/1610.05492.

[23] C. Rhodes, *Perfect Harmony: Pharmas MELLODDY Consortium Joins Forces with NVIDIA to Supercharge AI Drug Discovery*, https://blogs.nvidia.com/blog/2019/08/08/pharma-melloddy-ai-drug-discovery-consortium, 2019.

[24] M. Duan, "Astraea: Self-balancing Federated Learning for Improving Classification Accuracy of Mobile Deep Learning Applications," *CoRR*, vol. abs/1907.01132, 2019. arXiv: 1907.01132. [Online]. Available: http://arxiv.org/abs/1907.01132.

[25] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated Learning with Non-IID Data," *CoRR*, vol. abs/1806.00582, 2018. arXiv: 1806.00582. [Online]. Available: http://arxiv.org/abs/1806.00582.

[26] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the Convergence of FedAvg on Non-IID Data," 2019. arXiv: 1907.02189 [stat.ML]. [Online]. Available: https://arxiv.org/abs/1907.02189.

[27] A. K. Sahu, T. Li, M. Sanjabi, M. Zaheer, A. Talwalkar, and V. Smith, "On the Convergence of Federated Optimization in Heterogeneous Networks," *CoRR*, vol. abs/1812.06127, 2018. arXiv: 1812.06127. [Online]. Available: http://arxiv.org/abs/1812.06127.

[28] B. Hitaj, G. Ateniese, and F. Pérez-Cruz, "Deep Models Under the GAN: Information Leakage from Collaborative Deep Learning," *CoRR*, vol. abs/1702.07464, 2017. arXiv: 1702.07464. [Online]. Available: http://arxiv.org/abs/1702.07464.

[29] L. Melis, C. Song, E. D. Cristofaro, and V. Shmatikov, "Exploiting Unintended Feature Leakage in Collaborative Learning," 2018. arXiv: 1805.04049 [cs.CR]. [Online]. Available: https://arxiv.org/abs/1805.04049.

[30] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning," *2019 IEEE Symposium on Security and Privacy (SP)*, 2019. DOI: 10.1109/sp.2019.00065. [Online]. Available: http://dx.doi.org/10.1109/SP.2019.00065.

[31] L. Zhu, Z. Liu, and S. Han, "Deep Leakage from Gradients," *CoRR*, vol. abs/1906.08935, 2019. arXiv: 1906.08935. [Online]. Available: http://arxiv.org/abs/1906.08935.

[32] B. Zhao, K. R. Mopuri, and H. Bilen, "iDLG: Improved Deep Leakage from Gradients," 2020. arXiv: 2001.02610 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2001.02610.

[33] Y. Lin, S. Han, H. Mao, Y. Wang, and W. J. Dally, "Deep Gradient Compression: Reducing the Communication Bandwidth for Distributed Training," *CoRR*, vol. abs/1712.01887, 2017. arXiv: 1712.01887. [Online]. Available: http://arxiv.org/abs/1712.01887.

[34] H. Tang, C. Zhang, S. Gan, T. Zhang, and J. Liu, "Decentralization Meets Quantization," *CoRR*, vol. abs/1803.06443, 2018. arXiv: 1803.06443. [Online]. Available: http://arxiv.org/abs/1803.06443.

[35] G. Zhu, Y. Wang, and K. Huang, "Low-Latency Broadband Analog Aggregation for Federated Edge Learning," *CoRR*, vol. abs/1812.11494, 2018. arXiv: 1812.11494. [Online]. Available: http://arxiv.org/abs/1812.11494.

[36] F. Seide, H. Fu, J. Droppo, G. Li, and D. Yu, "1-bit stochastic gradient descent and application to data-parallel distributed training of speech dnns," in *Interspeech 2014*, 2014. [Online]. Available: https://www.microsoft.com/en-us/research/publication/1-bit-stochastic-gradient-descent-and-application-to-data-parallel-distributed-training-of-speech-dnns/.

[37] N. Strom, "Scalable distributed DNN training using commodity GPU cloud computing," *INTERSPEECH*, pp. 1488–1492, 2015. [Online]. Available: https://www.isca-speech.org/archive/interspeech_2015/i15_1488.html.

[38] N. Dryden, T. Moon, S. A. Jacobs, and B. Van Essen, "Communication Quantization for Data-Parallel Training of Deep Neural Networks," in *2016 2nd Workshop on Machine Learning in HPC Environments (MLHPC)*, 2016, pp. 1–8.

[39] A. F. Aji and K. Heafield, "Sparse Communication for Distributed Gradient Descent," *CoRR*, vol. abs/1704.05021, 2017. arXiv: 1704.05021. [Online]. Available: http://arxiv.org/abs/1704.05021.

[40] J. Konecný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated Learning: Strategies for Improving Communication Efficiency," *CoRR*, vol. abs/1610.05492, 2016. arXiv: 1610.05492. [Online]. Available: http://arxiv.org/abs/1610.05492.

[41] P. Goyal, P. Dollár, R. B. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He, "Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour," *CoRR*, vol. abs/1706.02677, 2017. arXiv: 1706.02677. [Online]. Available: http://arxiv.org/abs/1706.02677.

[42] R. Pascanu, T. Mikolov, and Y. Bengio, "On the Difficulty of Training Recurrent Neural Networks," in *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ser. ICML'13, Atlanta, GA, USA: JMLR.org, 2013, III–1310–III–1318.

[43] I. Mitliagkas, C. Zhang, S. Hadjis, and C. Ré, "Asynchrony begets Momentum, with an Application to Deep Learning," 2016. arXiv: 1605.09774 [stat.ML]. [Online]. Available: https://arxiv.org/abs/1605.09774.

[44] P. Goyal, P. Dollár, R. B. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He, "Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour," *CoRR*, vol. abs/1706.02677, 2017. arXiv: 1706.02677. [Online]. Available: http://arxiv.org/abs/1706.02677.

[45] F. Sattler, S. Wiedemann, K. Müller, and W. Samek, "Sparse Binary Compression: Towards Distributed Deep Learning with minimal Communication," *CoRR*, vol. abs/1805.08768, 2018. arXiv: 1805.08768. [Online]. Available: http://arxiv.org/abs/1805.08768.

[46] O. Abari, H. Rahul, and D. Katabi, "Over-the-air Function Computation in Sensor Networks," *CoRR*, vol. abs/1612.02307, 2016. arXiv: 1612.02307. [Online]. Available: http://arxiv.org/abs/1612.02307.

[47] M. M. Amiri and D. Gündüz, "Machine Learning at the Wireless Edge: Distributed Stochastic Gradient Descent Over-the-Air," *CoRR*, vol. abs/1901.00844, 2019. arXiv: 1901.00844. [Online]. Available: http://arxiv.org/abs/1901.00844.

[48] D. L. Donoho, A. Maleki, and A. Montanari, "Message Passing Algorithms for Compressed Sensing," *CoRR*, vol. abs/0907.3574, 2009. arXiv: 0907.3574. [Online]. Available: http://arxiv.org/abs/0907.3574.

[49] G. Zhu, D. Liu, Y. Du, C. You, J. Zhang, and K. Huang, "Toward an Intelligent Edge: Wireless Communication Meets Machine Learning," *IEEE Communications Magazine*, vol. 58, no. 1, pp. 19–25, 2020.

[50] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/.

[51] M. Safak, "Digital Communications," in *Wireless Communications*. John Wiley & Sons, Ltd, 2017, 174–175.

[52] A. Goldsmith, "Performance of Digital Modulation over Wireless Channels," in *Wireless Communications*. Cambridge University Press, 2005, 172–203. DOI: 10.1017/CBO9780511841224.007.

[53] Kyongkuk Cho and Dongweon Yoon, "On the general BER expression of one- and two-dimensional amplitude modulations," *IEEE Transactions on Communications*, vol. 50, no. 7, pp. 1074–1080, 2002.

[54] "Coded Communication Over Fading Channels," in *Digital Communication Over Fading Channels*. John Wiley & Sons, Ltd, 2002, ch. 12, pp. 495–534, ISBN: 9780471200697. DOI: 10.1002/0471200697.ch12. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/0471200697.ch12. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/0471200697.ch12.

[55] J. Proakis, *Digital Communications*, ser. McGraw-Hill series in electrical and computer engineering: communications and signal processing. McGraw-Hill, 2001, ISBN: 9780071181839. [Online]. Available: https://books.google.ca/books?id=aUp2QgAACAAJ.

[56] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *International Conference on Learning Representations*, 2014.

[57] "IEEE Standard for Floating-Point Arithmetic," *IEEE Std 754-2019 (Revision of IEEE 754-2008)*, pp. 1–84, 2019.