Privacy-Aware Federated Learning with Global Differential Privacy

Spoorthi Airody Suresh

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Engineering

Yang Cindy Yi, Chair
Yaling Yang
Lingjia Liu

December 9th, 2022
Blacksburg, Virginia

Keywords: federated learning, communication constraints, differential privacy, spiking neural networks, training variance

Privacy-Aware Federated Learning with Global Differential Privacy

Spoorthi Airody Suresh

(ABSTRACT)

There is an increasing need for low-power neural systems as neural networks become more widely used in embedded devices with limited resources. Spiking neural networks (SNNs) are proving to be a more energy-efficient option to conventional Artificial neural networks (ANNs), which are recognized for being computationally heavy. Despite its significance, there has been not enough attention on training SNNs on large-scale distributed Machine Learning techniques like Federated Learning (FL). As federated learning involves many energy-constrained devices, there is a significant opportunity to take advantage of the energy efficiency offered by SNNs. However, it is necessary to address the real-world communication constraints in an FL system and this is addressed with the help of three communication-reduction techniques, namely, model compression, partial device participation, and periodic aggregation. Furthermore, the convergence of federated learning systems is also affected by data heterogeneity.

Federated learning systems are capable of protecting the private data of clients from adversaries. However, by analyzing the uploaded client parameters, confidential information can still be revealed. To combat privacy attacks on the FL systems, various attempts have been made to incorporate differential privacy within the framework. In this thesis, we investigate the trade-offs between communication costs and training variance under a Federated Learning system with Differential Privacy applied at the parameter server (curator model).

Privacy-Aware Federated Learning with Global Differential Privacy

Spoorthi Airody Suresh

(GENERAL AUDIENCE ABSTRACT)

Federated Learning is a decentralized method of training neural network models; it employs several participating devices to independently learn a model on their local data partition. These local models are then aggregated at a central server to achieve the same performance as if the model had been trained centrally. But with Federated Learning systems there is a communication overhead accumulated. Various communication reductions can be used to reduce these costs. Spiking Neural Networks, being the energy-efficient option to Artificial Neural Networks, can be utilized in Federated Learning systems. This is because FL systems consist of a network of energy-efficient devices.

Federated learning systems are helpful in preserving the privacy of data in the system. However, an attacker can still obtain meaningful information from the parameters that are transmitted during a session. To this end, differential privacy techniques are utilized to combat privacy concerns in Federated Learning systems. In this thesis, we compare and contrast different communication costs and parameters of a federated learning system with differential privacy applied to it.

Contents

List of Figures

Chapter 1

Introduction

1.1 Overview

Mobile phones, wearable gadgets, and self-driving vehicles are just a few examples of ubiquitous edge devices in today's distributed networks that produce copious volumes of data every day. The number of connected devices is predicted to increase to approximately 30 billion by the end of 2030 [88] as shown in Figure 1.1. The ability to store data locally and move network computing to the edge is becoming more and more alluring due to the increasing computational capacity of these devices and worries about transferring private information.

In fact, basic query processing across distributed, low-power devices has been studied for decades in the context of query processing in various fields such as sensor networks, computing at the edge, etc. Recent research has also looked at the possibility of centralized training of ML models while distributing and storing them locally on edge devices, for instance, this is a typical strategy in modeling and customization of mobile users [60]. However, with the increase in storage and computational capacities of distributed networks' devices, it is possible to utilize improved local parameters on each device. Additionally, it is also important that user-generated data stay on local devices due to privacy issues around the transmission of raw data. Due to this, federated learning, which investigates directly training machine learning models [70] on remote devices, is gaining popularity.

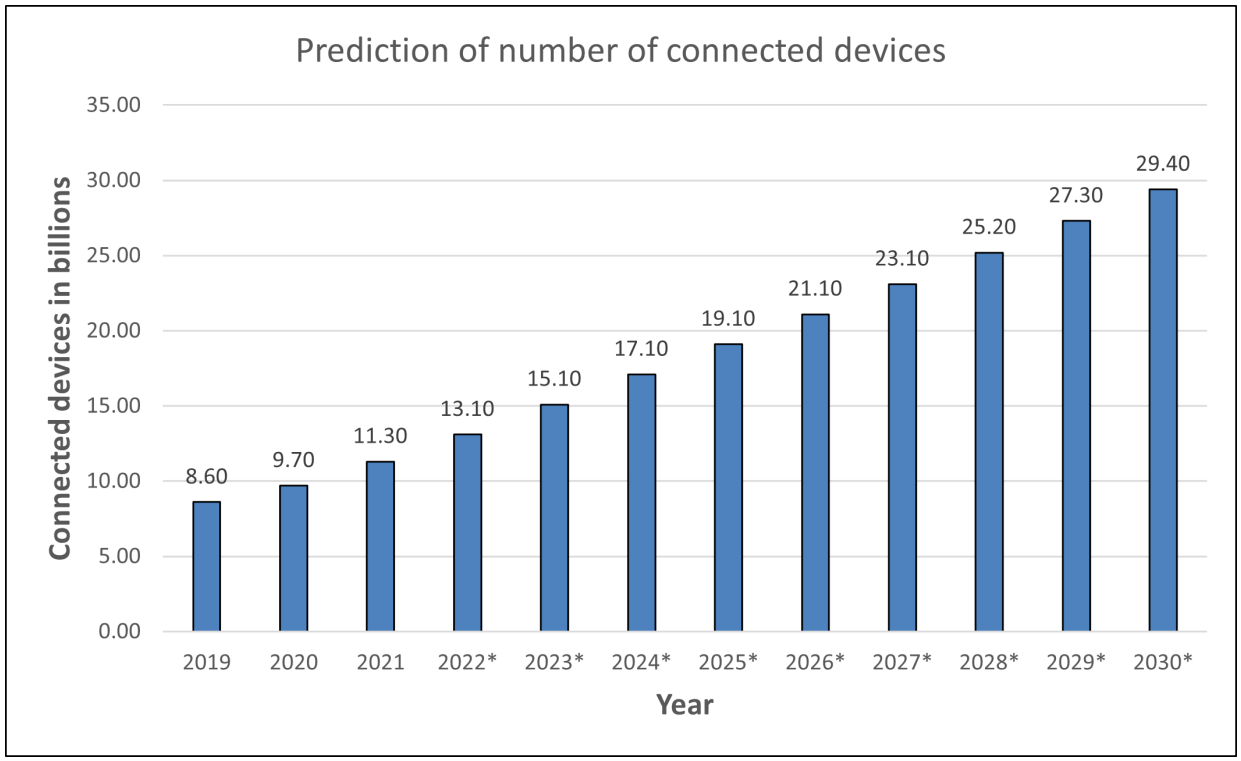

Figure 1.1: Prediction of number of connected devices from 2019 - 2030.

Originally, federated learning was introduced by Google to perform next-word prediction activities on smartphones [51]. In the current world, federated learning is being used in various domains. For example, in the healthcare sector, Nvidia works with medical organizations to forecast the COVID-19 symptoms of specific individuals by leveraging FL [54]. Federated learning also plays a significant role in the Internet of Things networks, which include wearable technology, autonomous vehicles, and smart homes that use sensors to collect and respond to incoming data in real-time [47, 79, 91]. In the case of autonomous vehicles, a vehicle requires updated information on traffic, pedestrians, and any obstruction for it to perform in an efficient and safe manner.

1.2 Motivation

Various organizations gather sensitive information, such as medical records, mobile phone records, and streaming service preferences, in order to create valuable predictions. These days, after gathering all of this data in a data center, various machine learning algorithms are applied. The model is subsequently trained on robust servers. However, the process of gathering data frequently violates data privacy policies. Machine learning can be challenging in some cases since many people do not want to give businesses access to their personal information. One way of improving data privacy is with data anonymization. Here, following collection, the data is preprocessed to achieve complete anonymity before being made available to businesses and research communities for analysis. Data anonymization makes it less possible for anyone to fully reconstruct the data from its original form using only the data itself.

In 2007, Netflix held a competition to find the best algorithm for collaborative filtering, which anticipates a user's rating of a movie based on ratings from previous users. All user information in the databases, including movie titles and other items, was changed to numbers assigned specifically for the competition. However, Narayanan et. al. [75] demonstrated how they were able to uncover nearly all of the sensitive data from the anonymized dataset provide by Netflix, including user identities, by merely using supplemental public data from the Internet Movie Database (IMDb).

In reality, anonymizing the data-points is a fairly poor method of protecting an individual's privacy. About 600K randomly chosen users' anonymized search logs were made available by AOL in 2006 for research usage. Because the user's personal information was included in the logs, the anonymization was insufficient.Barbaro et. al. [26] were able to demonstrate how users' identities might be stolen by comparing these specifics with phone book entries.

Another noteworthy instance included the de-anonymization of medical records in 1997; Barth-Jones et. al. [27] describes how analysts were able to extract the Massachusetts Governor's medical information. The analysts effectively matched numerous datasets of anonymized medical records with publicly accessible voter registration records.

By retaining local datasets on local devices and only exchanging local upgrades with the server, Federated Learning is able to protect some levels of privacy. However, it has been demonstrated that this is insufficient to protect the privacy of the data because the parameters can provide information about the training data. Therefore, federated learning by itself can only be implemented with trustworthy participants, and further measures should be taken to extend it for secure and privacy-preserving situations.

1.3 Objectives

The main objectives of the thesis can be summarized as below:

- Evaluate a federated learning framework with global differential privacy for Spiking Neural networks.
- Study the effects of communication reduction strategies under the presence of data heterogeneity on the privacy preserving federated learning system.

1.4 Thesis Organization

The chapters of this thesis are organized in such a way that it introduces the concept of federated learning, provides insights on how differential privacy can be used in context with federated learning and analyzes the implications of global differential privacy in a federated

learning environment. Chapter 2 summarizes federated learning and discusses the real-world challenges and ways to mitigate the same. Chapter 3 overviews various approaches used for incorporating differential privacy in federated learning systems. The proposed framework of global differential privacy is introduced in chapter 4. The experimental results are discussed in chapter 5. Chapter 6 concludes the work presented in this thesis and proposed directions for future research.

Chapter 2

Background

2.1 Centralized Machine Learning

Traditionally, machine learning models collect data from various sources into to a central server. At this server, an algorithm like a decision tree or even a neural network is used to train on the data that has been collected. The resulting model is then run on the central server or can be transmitted to various sources. Even though the approach is straightforward, there are some disadvantages associated with it. Such as a single point of failure since all of the training data is on one centralized machine, high uplink communication cost incurred as raw data is transmitted from the local devices, the training process is also slowed down due to the amount of data available, an attacker may be able to access private and sensitive content since the raw data is directly transmitted from the devices [2].

2.2 Distributed Machine Learning

Algorithms for distributed learning are created to overcome the computing issues posed by complicated algorithms on massive datasets. In comparison to centralized ML, distributed ML algorithms are more effective and scalable. In the case of distributed learning, a model is trained using a dataset on the various devices available rather than at a central location. The model updates are then sent to the central server where they are averaged [59].

Similar to the concept of distributed learning, model training is performed independently in Federated Learning. However, in FL, each device trains the model with the help of the data which is available locally on the device. Due to this a federated learning system has a non-independent and identical setting, unlike distributed learning.

2.3 Federated Learning

The way we interact with existing digital devices has completely changed as a result of advancements in machine learning and deep learning over the past several years. For instance, we could never have guessed that deep learning applications would pave the way for autonomous vehicles and digital assistants like Alexa, Siri, and Google Assistant, which are now a part of our day-to-day lives. This achievement is largely because of the availability of massive training infrastructures and an abundance of training data. However, as consumers and providers of machine learning solutions have become concerned about the privacy implications of this increasingly data-intensive process, efforts to implement privacy-preserving measures have been made by ML service providers. In addition to privacy concerns, the data localization paradigm, which requires that data be processed in the same place where it was initially collected and stored, is quickly becoming a crucial machine learning component due to energy efficiency concerns.

In order to train models without ever having the data leave the users' devices, federated learning (FL), a new realm of machine learning, promises to overcome these problems. Federated Learning shifts the computation to the clients' devices, allowing clients to simultaneously train a shared model. Local training on the device itself has been made possible because of the recent improvements in the storage capacity, data accessibility, and computational capabilities of edge devices.

Google first used the term ”federated learning” in 2016, as the use and abuse of personal data were getting attention on a worldwide scale. The findings of Konečný et. al. [58] and McMahan et. al. [70] formed the basis for the current form of federated learning. Federated learning allows for the remote sharing of data by numerous individuals in order to jointly train a single deep learning model and iteratively enhance it. The model, which is often a foundation model that has already been trained, is then downloaded by each participant from a central cloud server. They then summarize and encrypt the model's new configuration after training it on their personal datasets. The model updates are transmitted back to the cloud where they are decrypted, aggregated, and integrated into the main model. The collaborative training continues iteration after iteration until the model is fully trained.

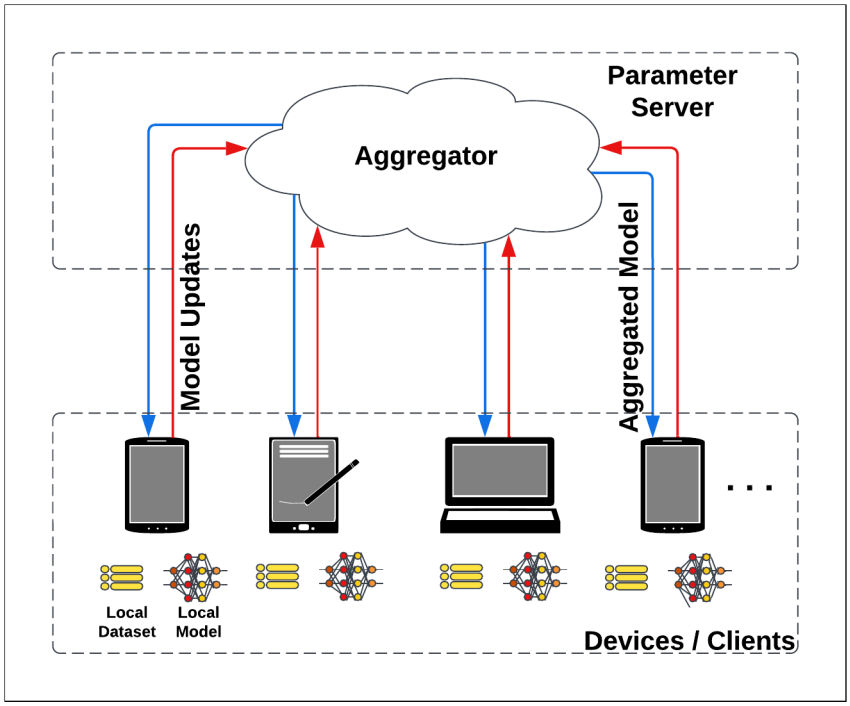

Figure 2.1: Architecture of a general Federated Learning system.

2.3.1 Federated Learning Algorithms

Figure 2.1 depicts the architecture of the traditional framework for federated learning systems. The system is made up of N local devices that are linked to a parameter server through a wireless network, which connects all the devices to train a single shared model. For each communication round, $k \in [K]$, participating devices, which are selected from the pool of available devices, download the aggregated data (x_k, c_k) from the parameter server, where x_k is the global model and c_k is the global control variate. Each device, i, from a pool of participating devices, S_k, determines the local model update, $\Delta x_k^{(i)}$, and control variate updates, $\Delta c_k^{(i)}$ for some local functions. This training process is repeated and is as shown in Algorithm 1.

Algorithm 1 Traditional Federated Learning

Input: global learning rate $\eta_{k,g}$ for $k \in [K]$
Initialize: model parameters $x_0 \in \mathbb{R}^d$
  for each communication round $k = 0, ..., K-1$ **do**
    on local each device i:
      calculate $\Delta x_k^{(i)}$
      calculate $\Delta c_k^{(i)}$
      send $\Delta x_k^{(i)}$ and $\Delta c_k^{(i)}$ to the parameter server
    on the parameter server:
      collect the local updates from devices in S_k
      calculate $x_{k+1} = x_k + \eta_{k,g}/M$
      calculate c_{k+1}
      transmit x_{k+1} and c_{k+1} to all devices
  end for

Federated Stochastic Gradient Descent Algorithm

In the Federated Stochastic Gradient Descent algorithm [35], for each communication round, $k \in K$, every device in the network is participating, that is all N devices are considered active. Local model updates are transmitted for every local SGD step. However, no control

variate updates are sent. As the algorithm necessitates complete involvement of devices and communication of local updates at every training session, it may result in prohibitive communication expenses.

Federated Averaging

The Federated Averaging algorithm, FedAvg, [64] takes into account techniques that help in reducing communication costs. At each communication round, $M < N$ devices are considered as participating and the model updates are transmitted periodically.

We consider E to be the number of local iterations and device, $i \in S_k$, where S_k is the set of participating devices. The local model updates generated by device i for communication round, k, and local iterations, $t \in E$ can be given as:

$$x_{k,0}^{(i)} = x_k \tag{2.1}$$

$$x_{k,t}^{(i)} = x_{k,t-1}^{(i)} - \eta_k^l \tilde{\nabla} f_i(x_{k,t-1}^{(i)}) \tag{2.2}$$

The local model update:

$$\Delta x_k^{(i)} = x_{(k,E)}^{(i)} - x_k \tag{2.3}$$

Federated Learning method with Periodic Averaging and Quantization

FedPAQ [81] utilizes both periodic averaging and partial participation as in FedAvg. Along with these techniques, the FedPAQ, also utilizes model compressors to compress the local model updates that have to be transmitted. Thus, further reducing communication costs.

The local model updates after quantization is as below:

$$\Delta x_k^{(i)} = Q(x_{(k,E)}^{(i)} - x_k) \tag{2.4}$$

2.3.2 Real World Challenges

Communication Bottleneck

Data generated on each device must stay on the respective device because communication in federated networks is a major bottleneck [32], and transferring raw data raises privacy issues. Due to the presence of enormous numbers of devices such as millions of smartphones present in the federated learning environment, communication within the network may be much slower than the iterations done locally on the devices. The difference in computational speed may be because of limited resources such as power and bandwidth [89]. In order to create efficient models that process the data generated by the devices effectively, it is crucial to design communication efficient techniques that send model parameters or updates during the training of the model instead of sending large datasets over the federated network. Various communication reduction strategies are employed to enhance the communication efficiency of federated learning systems such as:

- **Model Compression:**

  Machine learning systems typically have large models, making it prohibitively expensive to transfer model parameters with complete precision. In this regard, a popular tactic is to broadcast only a few crucial local parameters or to quantize the local model updates using low-precision compressors. Compression can greatly lower the overall transmission overhead.

- **Periodic Aggregation:**

  All active devices must be synchronized for the aggregation of local models. As with conventional distributed learning, aggregating during each iteration of training leads to significant communication overhead. As a result, regular model synchronization could become impractical and use up a lot of system resources for communication. To reduce communication costs, it is common practice to do numerous local iterations before syncing with the parameter server.

Systems heterogeneity

Each device in federated networks may have different storage, computational, and communication capabilities because of variations in hardware (CPU and RAM), network connectivity, and power [62]. Additionally, only a small portion of the total number of devices, such as a few hundred in a network with millions of devices, are normally active at any given time due to system limitations and network size. Additionally, it is usual for a participating device to stop working at a specific communication round because of connectivity or battery issues. These system-level traits significantly worsen problems like the straggler effect. As a result, it is not recommended in reality to wait for model upgrades from all devices. Furthermore, the capacity of the underlying MAC channel is often restricted and does not rise linearly with the addition of more transmitting devices. Partial participation through scheduling, where a subset of devices (M) from a set of participating devices (N) is scheduled to communicate during each communication slot, is an effective way to mitigate this problem. Additionally, a practical federated learning method should be

- prepared for limited participation of devices,
- tolerant of heterogeneous hardware,

- resistant to dropped devices in the communication network.

Statistical heterogeneity

Devices typically generate and gather data in a very non-identical distributed manner throughout the system, for instance, users of mobile phones utilize language differently when completing a task requiring next-word prediction. Additionally, a statistical structure may be present that represents the interaction between devices and their related distributions, in addition to the fact that the number of data points may vary widely between devices. This data-generation paradigm may complicate issue modeling, theoretical analysis, and empirical solution evaluation since it contradicts generally held independent and identically distributed assumptions in distributed optimization. Furthermore, statistical heterogeneity calls into doubt the usefulness of a single global model for different consumers. By implementing appropriate FL personalization strategies, such as simultaneously learning various local models via multitasking learning techniques [50, 84], these problems are frequently resolved. For instance, by taking into consideration certain speech patterns of particular users, the usefulness of a next-word prediction model may often be improved.

Privacy Concerns

In FL applications, privacy is frequently a chief concern, which is one of the reasons why Federated Learning has gained popularity in recent years. Federated Learning provides a method that takes a step in the right direction toward securing local data by transmitting model changes, such as gradient information, rather than the raw data [37, 38], although it is not always assured that such updates do not include sensitive client information. It was demonstrated that under specific cases, the central server or a third-party business might

divulge personal information from the shared model changes. Several tools and methods, such as differential privacy[34], safe multi-party computation[36], homomorphic encryption [52], and trusted execution environments [73], can be used to improve FL algorithms with formal privacy guarantees.

2.4 Differential Privacy

Differential privacy is a technology that gives researchers and data analysts the ability to access databases containing people's private information and retrieve relevant information without disclosing the identity of any particular individual. This can be accomplished by inducing some amount of disruption in the data provided by the database. The introduced disruption is significant enough to safeguard privacy while also being constrained enough to maintain the value of the information provided to analysts. Differential privacy can be defined as the process of making data anonymous by purposefully adding noise to the dataset. It enables data analysts to carry out every statistical study conceivable without identifying any personal data.

Formally, differential privacy is a definition of privacy in terms of mathematics. It is a property that a process can have rather than a specific procedure like de-identification. It is possible to demonstrate, for instance, that a certain algorithm "satisfies" differential privacy. Informally, differential privacy ensures that each person who provides data for analysis will receive the following guarantee: whether or not you provide your data, the outcome of a differentially private analysis will be substantially the same. A mechanism is commonly referred to as a differentially private analysis, and we refer to it as M.

Figure 2.2 illustrates the principle of differential privacy. Here, the differential privacy mechanism M is said to be satisfied if, for all databases, D1 and D2 which differ by one individual's

data have output A and output B that are indistinguishable. This means that any person who sees output A and output B will not be able to distinguish if Alice's data was used or not, or if any of the databases even contained Alice's data.

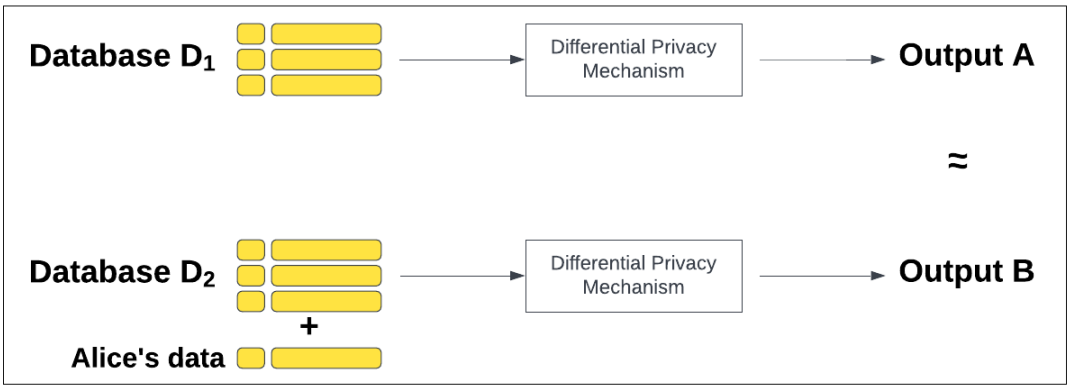

Figure 2.2: Principle of Differential Privacy

By adjusting the privacy parameter ϵ, also known as a privacy loss or privacy budget, we can regulate the strength of the privacy guarantee. The outcomes are more indistinguishable and, hence, more protected when the value of the parameter is lower.

In order to achieve data privacy, some random noise is added to the response of a query. Choosing where and how much noise to add presents a challenge. There are various strategies for choosing where to add the noise, such as:

- **Global Differential Privacy**

  A reliable data curator is an essential element of the global model. Each person gives the data curator access to their private information, which is then kept in one place. We believe the data curator to be trustworthy because we know they won't access sensitive information directly, won't share it with anybody, and can't be compromised by outside adversaries. With this model, we essentially presume that the server that has the sensitive data cannot be compromised.

Typically, we introduce noise to the outputs of the database queries in the global model. The benefit of this model is that it enables algorithms to contribute the least amount of noise and, as a result, deliver findings with the highest degree of precision permitted under differential privacy. We put the privacy barrier between the third party and the trusted data curator as shown in Figure 2.3; to its right, only differentially private results can be seen, negating the necessity for the third party to be trusted.

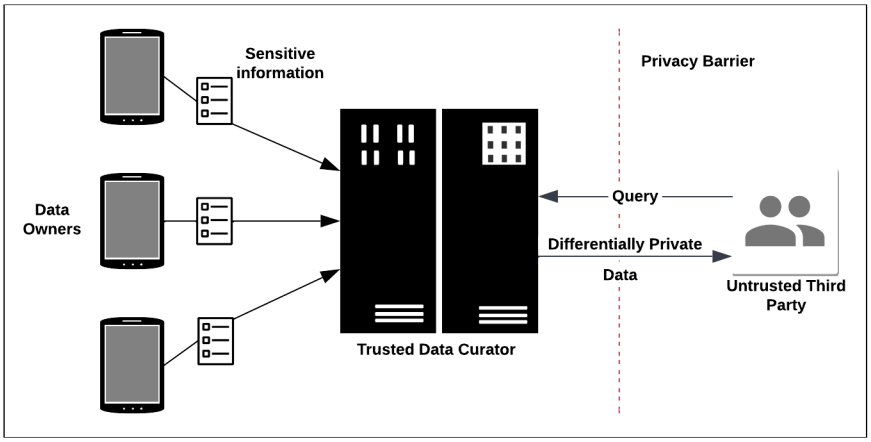

Figure 2.3: Global Differential Privacy

- **Local Differential Privacy**

The local differential privacy model addresses the security problem in the global model by removing the need for a trusted data curator. Before transmitting their own data to the data curator, each person contaminates it with noise. Since the data curator never sees the sensitive information, they may be trusted. The local model is shown in the figure 2.4, where each data owner and the data curator are separated by a privacy barrier.

If the data curator's server is compromised, the hackers will only be able to access noisy information that already meets differential privacy, avoiding the security concerns of

the global model. However, compared to the global model, the local model provides less accurate results. Since each participant in the local model contributes enough noise to satisfy differential privacy, it results in a far higher total amount of noise than the global model's single noise sample.

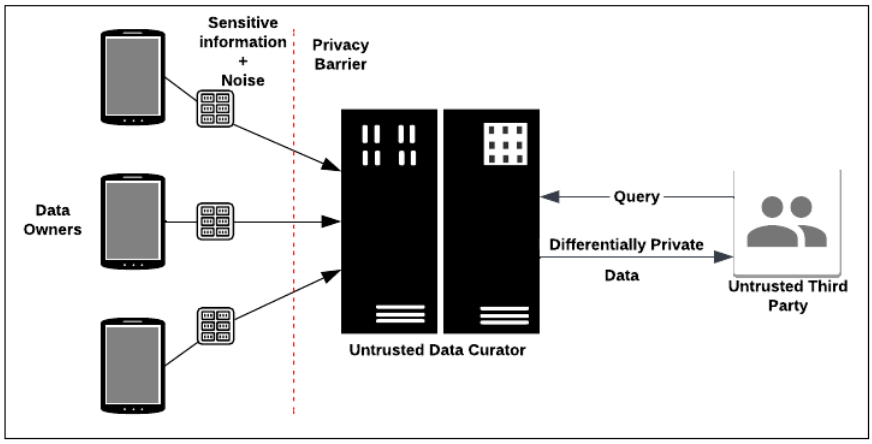

Figure 2.4: Local Differential Privacy

2.4.1 Privacy Mechanisms

The local parameters can be altered using a randomization technique before being released to the parameter server in order to achieve data privacy in FL, or alternatively, in a centralized manner, the trusted aggregating server can introduce noise to the averaged updates. The local updates are kept private and don't reveal sensitive information thanks to the addition of noise. Usually, Laplace Mechanism or Gaussian Mechanism is used in the generation of the noise to be added.

The idea of differential privacy is to guarantee bounds on how much information may be revealed by someone's participation in a database. These restrictions known as privacy parameters are defined by ϵ and δ. The privacy budget, ϵ, is the total amount of information

an attacker may gain with respect to an individual. Whereas, δ, is a term added to the privacy budget. In an ideal scenario, δ is considered to be 0 and hence achieves $(\epsilon, 0)$-differential privacy. If not under ideal conditions, (ϵ, δ)-differential privacy is achieved. Smaller the δ, the lesser the risk of losing the privacy guarantee.

Laplace Mechanism

The Laplace Mechanism was introduced by Dwork et al and it adds noise drawn from a Laplace distribution. It is defined as below:

$$\mathcal{M}_{Lap}(x, \epsilon) = f(x) + Lap(0, \frac{\Delta f}{\epsilon}) \tag{2.5}$$

Laplace mechanism depends on the l_1 sensitivity function. As seen from the above equation, for a larger value of ϵ, less noise is added and vice versa. A larger value of ϵ equates to a weak privacy guarantee. This mechanism preserves $(\epsilon, 0)$-differential privacy or ϵ-differential privacy.

Gaussian Mechanism

In contrast to the Laplace mechanism, the Gaussian mechanism introduces Gaussian noise rather than Laplacian noise. This mechanism is dependent on the l_2 sensitivity function and provides (ϵ, δ)-differential privacy. It is defined as below:

$$\mathcal{M}_{Gauss}(x, \epsilon, \delta) = f(x) + \mathcal{N}(0, \sigma^2) \tag{2.6}$$

$$\sigma^2 = \frac{2ln(1.25/\delta).(\Delta f)^2}{\epsilon^2} \tag{2.7}$$

We primarily use the Gaussian mechanism to generate the noise added to the aggregated model update in this work.

2.5 Machine Learning & Differential Privacy

Machine Learning has recently been at the heart of a variety of attacks. In the security domain, there are various adversarial attacks such as In Poisoning attack [30], by manipulating the training data or its labels, the attacker can make the model perform poorly during deployment. An attacker may contaminate the data by injecting malicious samples during the training process, which would then disrupt or affect the machine learning system while re-training. Another such attack is known as the Evasion attack [31] where during deployment, the attacker tampers with the data to trick classifiers that have already been trained.

The adversary's objective in the privacy domain is to obtain sensitive, private data about the model's underlying training set or the model itself. The opponent in Membership Inference [80, 83], for instance, needs to determine if a data point, he has is a part of the training dataset. In many instances, the attackers can execute membership inference attacks by simply monitoring the output of the machine learning model without knowledge of its parameters. Model Inversion attacks, on the other hand, aim directly at the learned model. They seek to recreate a model's internal composite representation [33, 46]. Defense strategies [82, 87] that address these vulnerabilities are mostly based on the concept of Differential Privacy.

Chapter 3

Review of Literature

There have been several works [1, 28, 85] where authors heavily rely on introducing the additive noise mechanism (Gaussian or Laplace) to the gradients again at every SGD step. The fundamental composition theorem and its more complex variations [39, 40, 41, 55] predict that the privacy loss will rise with each step. We can measure this degradation thanks to the concept of privacy loss, which also makes it possible to analyze and manage cumulative privacy loss across numerous computations.

In [1], the authors suggested the Moments Accountant to do this. One can determine "how much privacy budget is being spent" in this manner. This is accomplished by interpreting privacy loss as a random variable, using a moment-generating function to determine higher moments of this variable, and then binding these moments to provide a DP guarantee. They may monitor the cumulative loss at each stage in this way, allowing the training process to be stopped whenever a specific ϵ or δ threshold value has been reached. Additionally, in order to illustrate this point, they train deep neural networks for classification using DP-SGD [28, 85], a variation of SGD. As a result of their experimental findings, which show great accuracy and little privacy loss, training models under privacy assurances has advanced to the current state of the art.

Papernot et. al.[77] suggest the PATE framework for deep learning. By transmitting knowledge from an ensemble of parent models, which are trained on subsets of private data, to a child model during the learning process, they safeguard the privacy of the training set.

Authors have subsequently improved on this work as described in [78] to demonstrate the system's potential using larger, real-world datasets. Since numerous attacks and defenses that compromise privacy in that environment have been put forth, differential privacy and federated learning systems have drawn more attention. In the article by Hitaj et. al. [53], the authors suggest a Generative Adversarial Network (GAN) that may partially recreate a private dataset, while in [72], they show that periodically exchanging model updates can accidentally reveal participants' training data.

Strong DP guarantees are proposed by Bhowmick et. al. [29] for training decentralized models to prevent some attackers from reconstructing an individual's data. It can also be observed that the insights provided in [49] and [71] are quite similar.Recurrent networks are used by McMahan et. al. [71] to learn DP language models, and convolutional networks are used by Geyer et. al. [49] to do DP picture recognition. Both of them, similar to what we have presented, share the learning task among N different clients by utilizing the Federated Learning architecture as proposed in [70]. In an effort to do away with the necessity of substantial parameter adjusting in federated setups, a recent expansion of work by McMahan et. al.[71] in [86] addresses the idea of adaptive clipping of clients' updates.

These methods can be described as an output perturbation of the optimization process. Each client builds a model from a locally available dataset that is not strictly private. The authors create a new model by averaging the local ones, which is then augmented with calibrated noise to assure differential privacy. We observe that a large number of communication rounds between the involved devices and the server must be carried out in order to generate a suitably accurate model. As a result, the privacy mechanism must be used throughout the rounds.The moments accountant introduced in [1] is used by the authors to account for this privacy loss, and the procedure is stopped when a predetermined value of ϵ or δ is achieved. The parameter server that averages the client's models is taken into account by the authors

in both scenarios as a reliable third party.

Numerous studies on the convergence analysis of federated learning techniques using various strategies to decrease communication overhead have been conducted. Li et. al.[64] investigated the FedAvg algorithm's convergence rate in a non-i.i.d. data situation with partial participation and periodic averaging based on the FedAvg algorithm first proposed in [70]. The authors, however, did not take quantization into account model compression techniques. In [57], a like set of findings was revealed.

The FedPaq framework was presented and examined in [81] using each of the three communication reduction strategies. The authors, however, presupposed homogeneous data. FedProx, an adaption of FedAvg, was created by Li et. al. [63] to counteract the impact of data heterogeneity. In FedProx, each local objective function is given a proximal term. These two works, however, did not take quantization into account.

The SCAFFOLD framework was finally introduced by [56], and this is possibly the most recent study. It includes a variance-reduction technique to counteract the impact of heterogeneous data. To limit the drifts among various devices, some control variates are specifically implemented in this framework. Although Karimireddy et. al.[56] demonstrates that SCAFFOLD performs better than FedAvg, the performance advantage is insignificant in the presence of moderate heterogeneity. But the communication cost is observed to be twofold for each round in SCAFFOLD because all active devices must send the local control variate updates along with the model parameter of the same size to the parameter server. FedPaq is still a viable option for federated learning in practice. The convergence analysis in the original study introduced in [81], however, is based on the unrealistic homogeneous data assumption. FedPaqDP[74] utilizes the techniques used in FEDPaq in a non-iid setting. It also incorporates differential privacy by adding a Gaussian noise at the local device level. However, the analysis is done on a conventional neural network setting.

It is crucial to get the FedPaq convergence findings under data heterogeneity in order to examine the trade-offs between communication overhead and training variance. In order to accomplish this, we build on current techniques like FedPaq and suggest a new framework that makes use of a curator model for spiking neural networks in order to deliver end-to-end privacy assurance.

Several works conducted by the Brain Inspired Computing and Circuits (BRICC) Lab at the MICS, Virginia Tech revolves around the hardware and software aspects of Brain Inspired computing. There has been significant work with respect to neural encoding [90, 92, 93, 95, 97, 102, 105], neuromorphic computing in both communications [15, 16, 22, 23, 61, 65, 66, 76, 94] and FPGA domains [9, 19, 24, 25, 42, 43, 44, 48, 101, 104], and also Memristor based Neuromorphic computing [3, 4, 5, 6, 7, 8, 10, 11, 12, 13, 14, 21, 45].

As neural networks are increasingly used in embedded devices with constrained resources, there is a growing demand for low-power neural systems. Artificial neural networks are known to be computationally intensive, however, spiking neural networks are proven to be a more energy-efficient alternative [17, 18, 96, 98, 103]. There have been efforts made in creating efficient neural networks on edge devices [20, 67, 68]. This thesis explores the use of spiking neural networks on edge devices along with examining the trade-offs between the communication reduction techniques in a federated learning environment.

Chapter 4

Methodology

4.1 Dataset

The dataset used for the implementation of global differential privacy in federated learning systems is the very well-known MNIST dataset. The reason MNIST was chosen for the trials is that it is frequently used as a testbed for novel ideas and serves as a proof of concept for many studies in the ML-Privacy research community. As a result, the given work can easily be compared to other cutting-edge ideas. For training purposes, the dataset includes of 60,000 examples of labeled, handwritten digits (0–9), and an additional 10,000 examples are used for evaluation.

Since the design is to be evaluated for spiking neural networks, we construct the neural networks using the MNIST dataset with 25 Leaky Integrate and Fire neurons in the intermediate layer and 10 more Leaky Integrate and Fire neurons in the output layer. These 10 neurons on the output layer correspond to the 10 digits from 0 – 9 in the MNIST dataset.

4.2 Implementation

The Federated Learning system is made up of N local devices that are connected to a parameter server through a wireless network and work together to train a single shared model.

The parameter server first sends the aggregated data, that is, x_k, the global model, received from the previous communication round to all active devices in a given communication round, $k \in K$. However, for a device, i, that is a part of a group of active or participating devices, S_k, that particular device, i , will determine the local model update, $\Delta x_k^{(i)}$ for some local functions, V_i, where $i \in S_k$. We consider the number of local iterations on each device, $i \in S_k$ to be E. After E iterations, the clients send a quantized version of the local updates through the quantizer, $Q(.)$, for additional compression and a decrease in communication overhead.

Consider the local model updates for local iterations, $t \in E$, as follows:

$$x_{k,0}^{(i)} = x_k \tag{4.1}$$

$$x_{k,t}^{(i)} = x_{k,t-1}^{(i)} - \eta_k^l \tilde{\nabla} f_i(x_{k,t-1}^{(i)}) \tag{4.2}$$

The local model update after quantization is given by:

$$\Delta x_k^{(i)} = Q(x_{(k,E)}^{(i)} - x_k) \tag{4.3}$$

The parameter receives the quantized local model updates for aggregation. Once the local updates have been combined, a random noise generated from a Gaussian distribution is applied to perturb them in order to maintain the desired level of privacy.

The global model update after being perturbed by the noise, z_k at the parameter server is given by:

$$x_{k+1} = x_k + \frac{1}{M} \Sigma_{i \in S_k} \Delta x_k^{(i)} + z_k \tag{4.4}$$

The algorithm for implementing federated learning systems with global differential privacy

is shown in Algorithm 2.

Algorithm 2 Federated Learning with Global Differential Privacy

Input: learning rate η_k for $k \in [K]$
Initialize: model parameters $x_0 \in \mathbb{R}^d$
  for each round $k = 1, ..., K$ **do**
    on each device $i \in S_k$:
      initialize local model, $x_{k,0}^{(i)} = x_k$
      for each local iteration $t = 1, ..., E$
        calculate $x_{k,t}^{(i)} = x_{k,t-1}^{(i)} - \eta_k^l \tilde{\nabla} f_i(x_{k,t-1}^{(i)})$
      end for
      send quantized model updates $\Delta x_k^{(i)} = Q(x_{(k,E)}^{(i)} - x_k)$ to the parameter server
    on the parameter server:
      collect the local updates from devices in S_k
      calculate noise z_k
      calculate $x_{k+1} = x_k + \frac{1}{M} \Sigma_{i \in S_k} \Delta x_k^{(i)} + z_k$
      broadcast x_{k+1} to all devices
  end for

Chapter 5

Experiments & Results

As mentioned earlier, the dataset used is the well-known MNIST dataset. The 10-digit images are distributed among 100 local devices. However, the degree of data heterogeneity is controlled by allowing a local device to have access to training samples for a fraction of all 10 classes which would lead to ten datasets. The images are transformed to spikes by applying ISI encoding [99, 100] to the pixel values with the spikes spanning 10 discrete time steps.

5.1 Experiment Setup

The total number of iterations, E, is considered to be 10 and the total number of communication rounds, K, is 100 unless otherwise stated. Tests conducted under a non-privatized setting mean that the subsampling ratio and clipping are not considered.

We discuss how each parameter such as gradient clipping, C, subsampling ration γ, privacy budget ϵ, along with the communication reduction techniques affects the performance of the FL system with global differential privacy.

5.2 Effect of Partial Participation

When the clients per round, M is set to a low value, such as M = 1, as illustrated in figure 5.1, training may become incredibly unstable, especially during the first few training iterations.

It can also be observed that the performance when clients per round, M = 10, is much more stable and incurs lesser loss than M = 1.

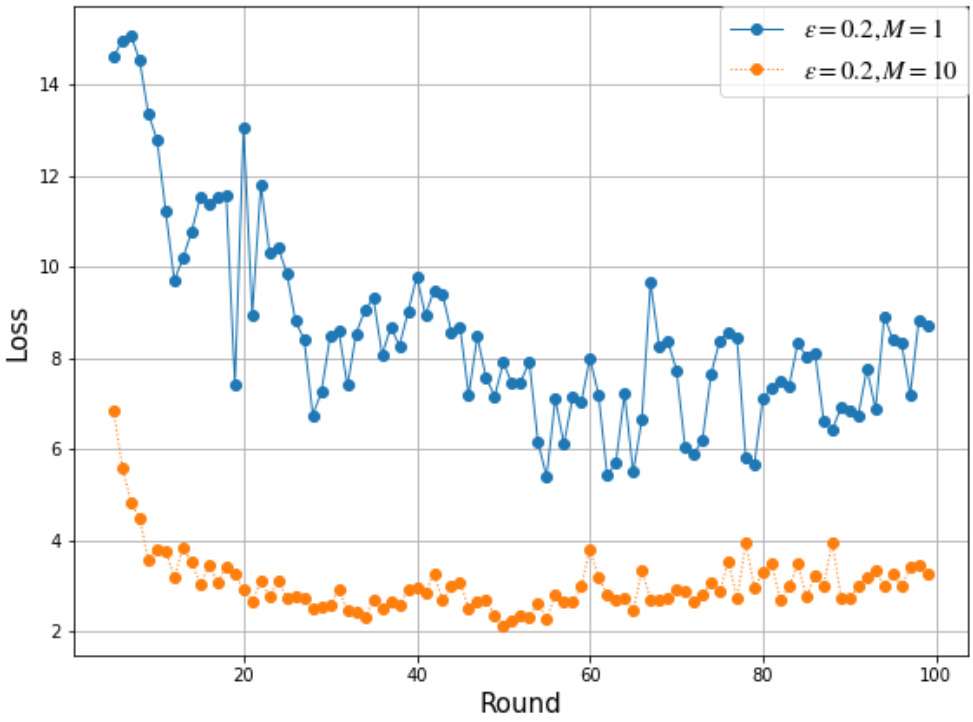

Figure 5.1: The effect of Partial Participation.

5.3 Effect of Periodic Aggregation

We see the effect of local iterations, E as depicted in the 5.2 when we consider data heterogeneity. We note that with a larger E per round, the model's accuracy increases. This

is referring to the situation where a set number of communication rounds is realistically required. However, a larger E causes the model's convergence at each iteration to be delayed.

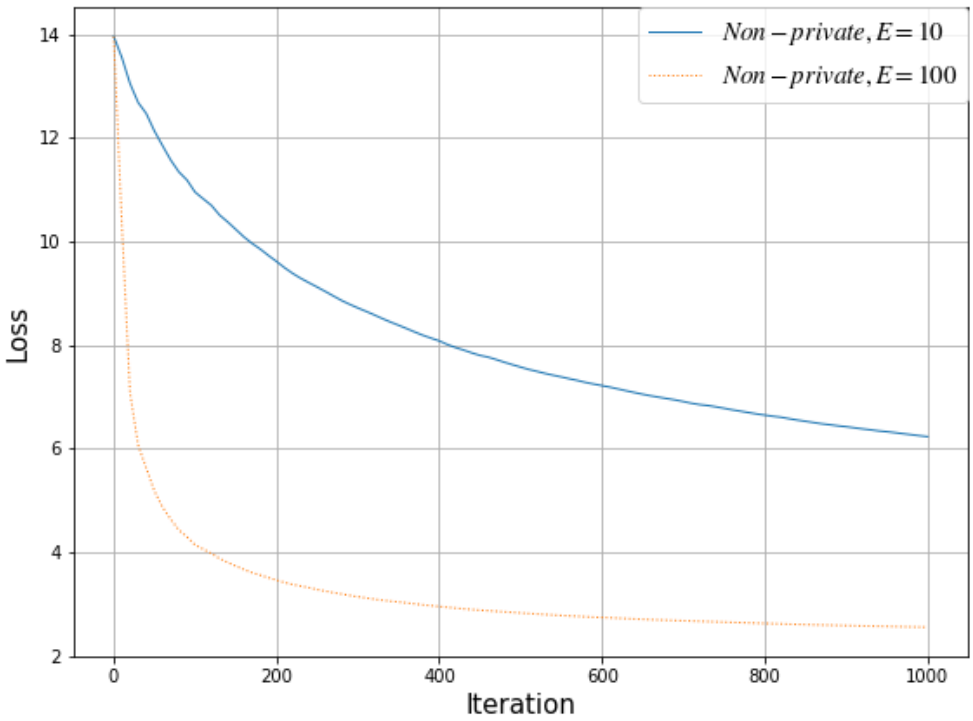

Figure 5.2: The effect of Periodic Aggregation.

5.4 Effect of Model Compression

From Figure 5.3, it can be observed that the loss for quantization level, 10, is almost the same as the one without any quantization. The performance decrease is minimal even for s = 1, and the training is largely steady. This allows us to adopt lossy quantization techniques as there are other parameters such as the number of clients per round and local iterations that have more impact on the convergence.

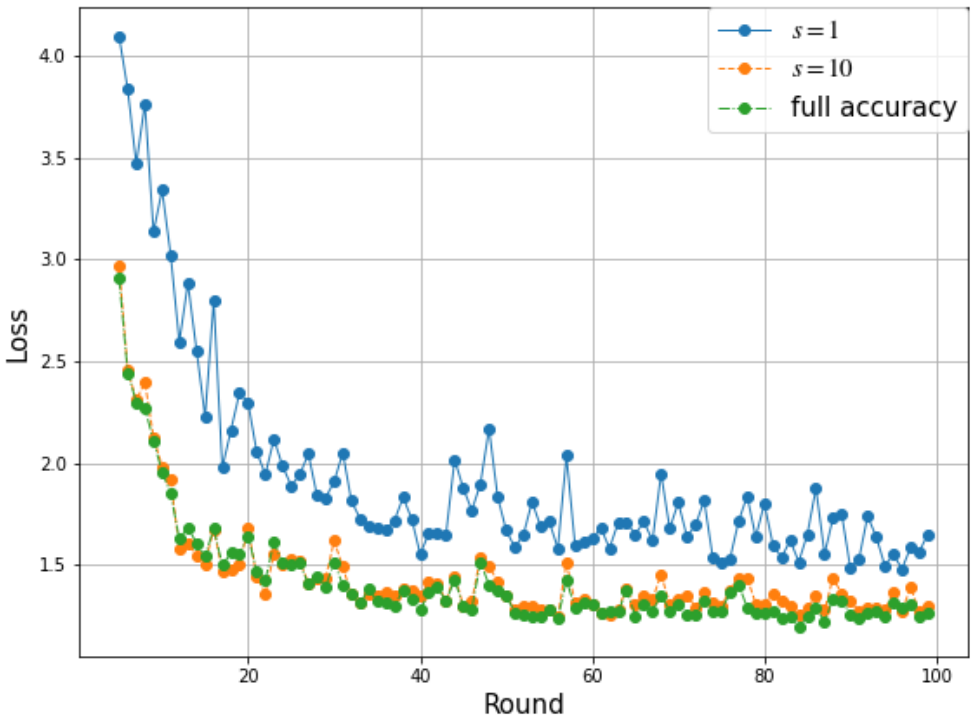

Figure 5.3: The effect of quantization, s.

5.5 Effect of Privacy Budget

As the privacy parameters, ϵ and δ, are increased, the amplitude of the additive noise decreases, improving accuracy but putting client privacy at risk. In each of the tests, we have fixed δ to be $10^{(-3)}$. Figure 5.4 shows how the privacy budget affects the model's performance. Setting a very low privacy budget, ϵ, can cause the utility to fall noticeably while boosting the privacy guarantee.

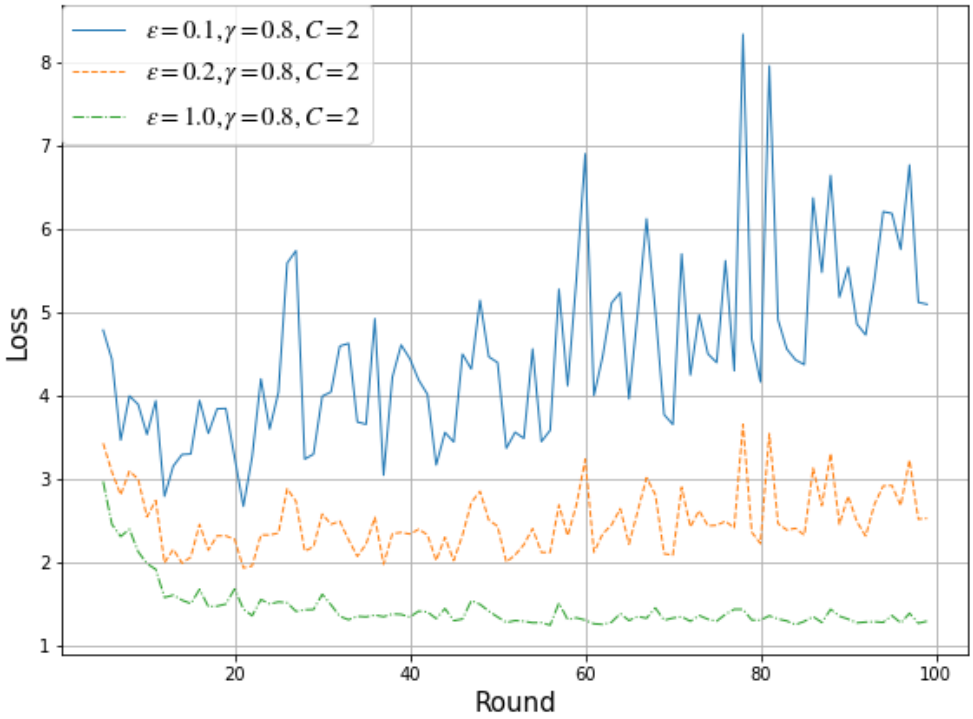

Figure 5.4: The effect of Privacy Budget,ϵ.

5.6 Effect of Gradient Clipping

The global sensitivity significantly affects the algorithm's rate of convergence. Figure 5.5 demonstrates that a greater C causes more noise, which delays the privatized algorithm's convergence. In fact, it may be argued that choosing a bigger privacy budget, ϵ to achieve faster convergence is less preferable than picking a lower C for clipping the gradients. Although, figure 5.7 suggests that this isn't always the case, though.

5.7 Effect of Sub-sampling

We are utilizing sub-sampling as a privacy amplification strategy to decrease the amount of noise that needs to be introduced in order to retain a particular degree of privacy guarantee,

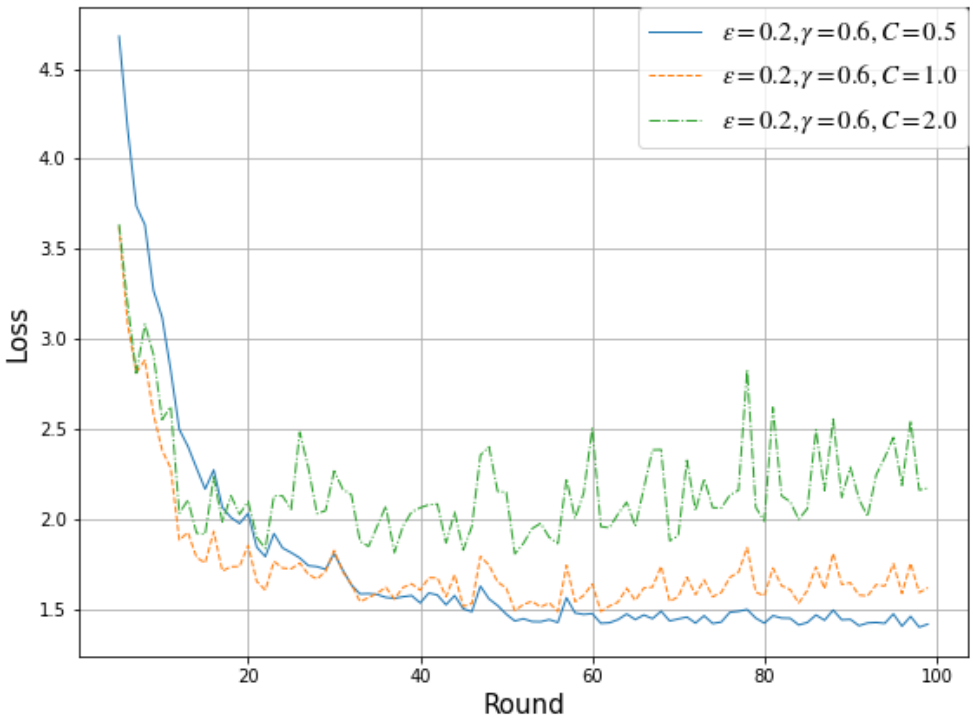

Figure 5.5: The effect of Gradient Clipping, C.

enhancing the utility. Figure 5.6 demonstrates how, with higher levels of E, a bigger sub-sampling ratio has a more pronounced degrading effect. It shows the training loss of a few, otherwise identical, private models for different values of the sub-sampling ratio, γ.

The performance of the model is shown as a result of how γ and C interact in figure 5.7. One can see that depending on which value of γ is chosen, a smaller clipping of C = 0.5 produces either the best or one of the worst results in a privatized setting.

Clipping the gradients obtained on such a small sub-sample would greatly worsen convergence, as would be the case with a non-privatized algorithm, despite the fact that small C leads to relatively minimal additional noise. A more conservative clipping, such as C = 1, would be more immune to any undesirable effects of sub-sampling.

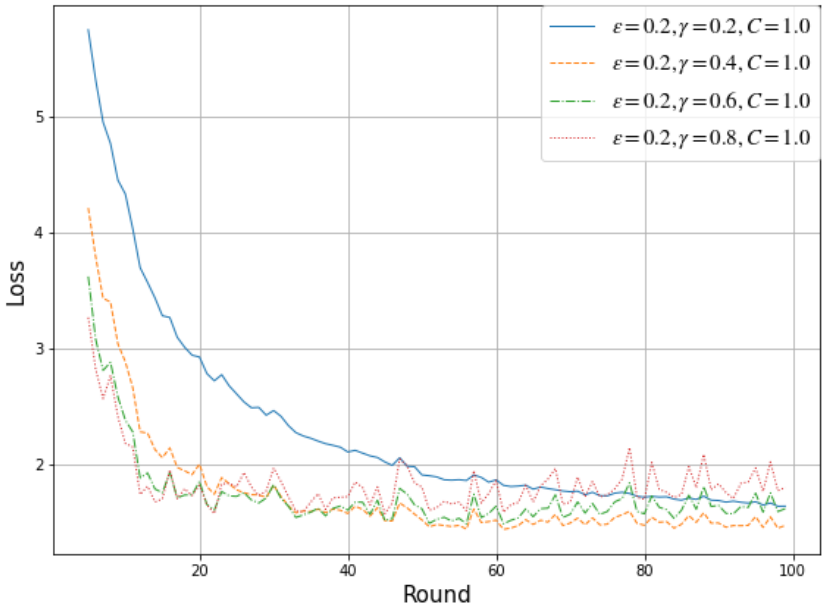

Figure 5.6: The effect of Sub-sampling, γ.

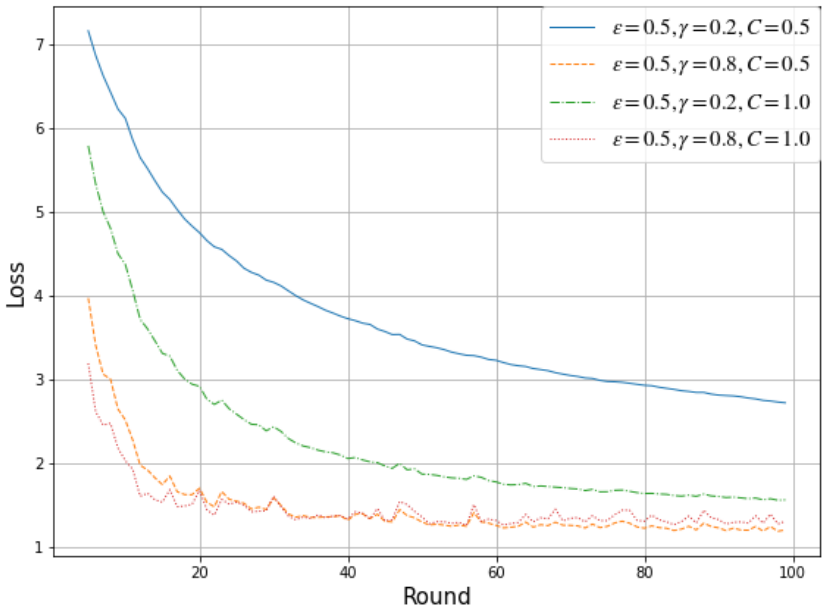

Figure 5.7: The effect of Sub-sampling, γ, and Gradient Clipping, C.

Chapter 6

Conclusions

In this chapter, we draw conclusions from the work done in this thesis along with providing some insights for possible future work.

6.1 Conclusions

The thesis summarizes the observations and analysis of a federated learning algorithm with global differential privacy for spiking neural networks while considering data heterogeneity and various communication reduction techniques. Some insights with respect to the communication reduction strategies were developed, like:

- No matter how heterogeneous the problem is, the choice of level of quantization has little effect on convergence. This promotes the widespread usage of low-precision quantizers in order to reduce the underlying communication costs.

- It can be observed that a larger number of local iterations, E, results in better performance. Although, it adds to the communication costs.

- In a private setting, we see that the lesser the number of clients per round, M, the more unstable the training process.

6.2 Future Work

Although the presented work gives insights into how a communication-efficient federated learning system with global differential privacy can be formulated, it does not provide an analysis of how much privacy can be achieved. There are various strategies such as membership interference tests and other privacy auditing techniques that can be utilized to gauge the privacy guarantee of the system [69]. A comparative study of the effects of such attacks on various differential privacy mechanisms in a federated learning environment will help deliver better privacy-aware systems.

Bibliography

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. *In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, 2016.

[2] altexsoft. Federated learning: The shift from centralized to distributed on-device model training, 2022. URL `https://www.altexsoft.com/blog/federated-learning/`.

[3] Hongyu An, M. Amimul Ehsan, Zhen Zhou, and Yang Yi. Electrical modeling and analysis of 3d synaptic array using vertical rram structure. *2017 18th International Symposium on Quality Electronic Design (ISQED)*, pages 1–6, 2017. doi: 10.1109/ISQED.2017.7918283.

[4] Hongyu An, Jialing Li, Ying Li, Xin Fu, and Yang Yi. Three dimensional memristor-based neuromorphic computing system and its application to cloud robotics. *Comput. Electr. Eng.*, 63:99–113, 2017.

[5] Hongyu An, Zhen Zhou, and Yang Yi. Memristor-based 3d neuromorphic computing system and its application to associative memory learning. *2017 IEEE 17th International Conference on Nanotechnology (IEEE-NANO)*, pages 555–560, 2017. doi: 10.1109/NANO.2017.8117459.

[6] Hongyu An, Zhen Zhou, and Yang Yi. 3d memristor-based adjustable deep recurrent neural network with programmable attention mechanism. *Proceedings of the Neuromorphic Computing Symposium*, 2017.

[7] Hongyu An, Mohammad Shah Al-Mamun, Marius K. Orlowski, and Yang Yi. Learning accuracy analysis of memristor-based nonlinear computing module on long short-term memory. *Proceedings of the International Conference on Neuromorphic Systems*, 2018.

[8] Hongyu An, M. Amimul Ehsan, Zhen Zhou, Fangyang Shen, and Yang Yi. Monolithic 3d neuromorphic computing system with hybrid cmos and memristor-based synapses and neurons. *Integration*, 65:273–281, 2019.

[9] Hongyu An, Dong Sam Ha, and Yang (Cindy) Yi. Powering next-generation industry 4.0 by a self-learning and low-power neuromorphic system. In *Proceedings of the 7th ACM International Conference on Nanoscale Computing and Communication*, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450380836. doi: 10.1145/3411295.3411302. URL `https://doi.org/10.1145/3411295.3411302`.

[10] Hongyu An, Mohammad Shah Al-Mamun, Marius K. Orlowski, Lingjia Liu, and Yang Yi. Robust deep reservoir computing through reliable memristor with improved heat dissipation capability. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 40(3):574–583, 2021. doi: 10.1109/TCAD.2020.3002539.

[11] Hongyu An, Mohammad Shah Al-Mamun, Marius K. Orlowski, and Yang Yi. A three-dimensional (3d) memristive spiking neural network (m-snn) system. *2021 22nd International Symposium on Quality Electronic Design (ISQED)*, pages 337–342, 2021. doi: 10.1109/ISQED51717.2021.9424303.

[12] Hongyu An, Qiyuan An, and Yang Yi. Realizing behavior level associative memory learning through three-dimensional memristor-based neuromorphic circuits. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 5(4):668–678, 2021. doi: 10.1109/TETCI.2019.2921787.

[13] Hongyu An, Kangjun Bai, and Yang Yi. Three-dimensional memristive deep neural network with programmable attention mechanism. *2021 22nd International Symposium on Quality Electronic Design (ISQED)*, pages 210–215, 2021. doi: 10.1109/ISQED51717.2021.9424331.

[14] Hongyu An, Mohammad Shah Al-Mamun, Marius K. Orlowski, Lingjia Liu, and Yang Yi. Three-dimensional neuromorphic computing system with two-layer and low-variation memristive synapses. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(3):400–409, 2022. doi: 10.1109/TCAD.2021.3061481.

[15] Qiyuan An, Kangjun Bai, Lingjia Liu, Fangyang Shen, and Yang Yi. A unified information perceptron using deep reservoir computing. *Computers and Electrical Engineering*, 85:106705, 07 2020. doi: 10.1016/j.compeleceng.2020.106705.

[16] Kangjun Bai and Yang Yi. Dfr: An energy-efficient analog delay feedback reservoir computing system for brain-inspired computing. *J. Emerg. Technol. Comput. Syst.*, 2018.

[17] Kangjun Bai, Bradley, and Yang Yi. A path to energy-efficient spiking delayed feedback reservoir computing system for brain-inspired neuromorphic processors. *2018 19th International Symposium on Quality Electronic Design (ISQED)*, pages 322–328, 2018. doi: 10.1109/ISQED.2018.8357307.

[18] Kangjun Bai, Jialing Li, Kian Hamedani, and Yang Yi. Enabling an new era of brain-inspired computing: Energy-efficient spiking neural network with ring topology. *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*, pages 1–6, 2018. doi: 10.1109/DAC.2018.8465938.

[19] Kangjun Bai, Qiyuan An, and Yang Yi. Deep-dfr: A memristive deep delayed feed-

back reservoir computing system with hybrid neural network topology. In *2019 56th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6, 2019.

[20] Kangjun Bai, Shiya Liu, and Yang Yi. High speed and energy efficient deep neural network for edge computing. *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*, page 347–349, 2019.

[21] Kangjun Bai, Qiyuan An, Lingjia Liu, and Yang Yi. A training-efficient hybrid-structured deep neural network with reconfigurable memristive synapses. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 28(1):62–75, 2020. doi: 10.1109/TVLSI.2019.2942267.

[22] Kangjun Bai, Lingjia Liu, Zhou Zhou, and Yang Yi. Detection through deep neural networks: A reservoir computing approach for mimo-ofdm symbol detection. *Proceedings of the 39th International Conference on Computer-Aided Design*, 2020.

[23] Kangjun Bai, Yang Yi, Zhou Zhou, Shashank Jere, and Lingjia Liu. Moving toward intelligence: Detecting symbols on 5g systems through deep echo state network. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 10(2):253–263, 2020. doi: 10.1109/JETCAS.2020.2992238.

[24] Kangjun Bai, Lingjia Liu, and Yang Yi. Spatial-temporal hybrid neural network with computing-in-memory architecture. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 68(7):2850–2862, 2021. doi: 10.1109/TCSI.2021.3071956.

[25] Kangjun Bai, Clare Thiem, Nathan McDonald, Lisa Loomis, and Yang Yi. Toward intelligence in communication networks: A deep learning identification strategy for radio frequency fingerprints. In *2021 22nd International Symposium on Quality Electronic Design (ISQED)*, pages 204–209, 2021. doi: 10.1109/ISQED51717.2021.9424319.

[26] Michael Barbaro and Tom Zeller Jr. A face is exposed for aol searcher no. 4417749, 2006. URL `https://www.nytimes.com/2006/08/09/technology/09aol.html`.

[27] Daniel C Barth-Jones. The 're-identification' of governor william weld's medical information: A critical re-examination of health data identification risks and privacy protections, then and now. *SSRN Electronic Journal*, 2012.

[28] Raef Bassily, Adam Smith, and Abhradeep Thakurta. Differentially private empirical risk minimization: Efficient algorithms and tight error bounds. *arXiv preprint*, arXiv:1405.7085, 2014.

[29] Abhishek Bhowmick, John Duchi, Julien Freudiger, Gaurav Kapoor, and Ryan Rogers. Protection against reconstruction and its applications in private federated learning. *arXiv preprint*, arXiv:1812.00984, 2018.

[30] Battista Biggio, Blaine Nelson, and Pavel Laskov. Support vector machines under adversarial label noise. *In Asian Conference on Machine Learning*, pages 97–112, 2011.

[31] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. *Springer Berlin Heidelberg*, pages 387–402, 2013.

[32] K. A. Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloé M Kiddon, Jakub Konečný, Stefano Mazzocchi, Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. Towards federated learning at scale: System design. *arXiv preprint*, arXiv:1902.01046, 2019.

[33] Nicholas Carlini, Chang Liu, Ulfar Erlingsson, Jernej Kos, and Dawn Song. The

secret sharer: Evaluating and testing unintended memorization in neural networks. *Proceedings of the 28th USENIX Conference on Security Symposium*, page 267–284, 2019.

[34] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research (JMLR)*, Vol. 12, no. 3, 2011.

[35] Jianmin Chen, Xinghao Pan, Rajat Monga, Samy Bengio, and Rafal Jozefowicz. Revisiting distributed synchronous sgd. *arXiv preprint*, arXiv:1604.00981, 2016.

[36] W. Du and M. J. Atallah. Secure multi-party computation problems and their applications: a review and open problems. *Workshop on New Security Paradigms*, 2001.

[37] John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. Privacy aware learning. *arXiv preprint*, arxiv.1210.2085, 2012.

[38] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy, 2014.

[39] Cynthia Dwork and Guy N Rothblum. Concentrated differential privacy. *arXiv preprint*, arXiv:1603.01887, 2016.

[40] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. *In Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 486–503, 2006.

[41] Cynthia Dwork, Guy N Rothblum, and Salil Vadhan. Boosting and differential privacy. *In 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 51–60, 2010.

[42] M. Amimul Ehsan, Zhen Zhou, and Yang Yi. Three dimensional integration technology applied to neuromorphic hardware implementation. In *2015 IEEE International Symposium on Nanoelectronic and Information Systems*, pages 203–206, 2015. doi: 10.1109/iNIS.2015.72.

[43] M. Amimul Ehsan, Hongyu An, Zhen Zhou, and Yang Yi. Design challenges and methodologies in 3d integration for neuromorphic computing systems. In *2016 17th International Symposium on Quality Electronic Design (ISQED)*, pages 24–28, 2016. doi: 10.1109/ISQED.2016.7479151.

[44] M. Amimul Ehsan, Hongyu An, Zhen Zhou, and Yang Yi. Adaptation of enhanced tsv capacitance as membrane property in 3d brain-inspired computing system. In *2017 54th ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6, 2017. doi: 10.1145/3061639.3062196.

[45] Md Amimul Ehsan, Hongyu An, Zhen Zhou, and Yang Yi. A novel approach for using tsvs as membrane capacitance in neuromorphic 3d ic. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, PP:1–1, 10 2017. doi: 10.1109/TCAD.2017.2760506.

[46] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. *In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1322–1333, 2015.

[47] Chunyan Fu. An architecture for an intelligent edge management., 2021. URL `https://www.ericsson.com/en/blog/2021/10/edge-computing-management-use-cases`.

[48] Victor M. Gan, Yibin Liang, Lianjun Li, Lingjia Liu, and Yang Yi. A cost-efficient

digital esn architecture on fpga for ofdm symbol detection. *J. Emerg. Technol. Comput. Syst.*, 2021. doi: 10.1145/3440017. URL https://doi.org/10.1145/3440017.

[49] Robin C Geyer, Tassilo Klein, , and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint*, arXiv:1712.07557, 2017.

[50] Filip Hanzely and Peter Richtárik. Federated learning of a mixture of global and local models. *arXiv preprint*, arXiv:2002.05516, 2020.

[51] Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Kiddon. Federated learning for mobile keyboard prediction. *arXiv preprint*, arxiv.1811.03604, 2018.

[52] S. Hardy, W. Henecka, H. Ivey-Law, R. Nock, G. Patrini, G. Smith, and B. Thorne. Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. *arXiv preprint*, arXiv:1711.10677, 2017.

[53] Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. Deep models under the gan: information leakage from collaborative deep learning. *In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 603–618, 2017.

[54] Makenzie Holland. Nvidia brings federated learning to covid-19 patient data., 2020. URL https://www.techtarget.com/searchhealthit/news/252490606/Nvidia-brings-federated-learning-to-COVID-19-patient-data.

[55] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. The composition theorem for differential privacy. *In Proceedings of the 32nd International Conference on International Conference on Machine Learning*, 37:1376–1385, 2015.

[56] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank J. Reddi, Sebastian U. Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. *arXiv preprint*, arXiv:1910.06378, 2019.

[57] Ahmed Khaled, Konstantin Mishchenko, and Peter Richtárik. First analysis of local gd on heterogeneous data. *arXiv preprint*, arXiv:1909.04715, 2019.

[58] Jakub Konečný, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint*, arxiv.1610.05492, 2016.

[59] Tim Kraska, Ameet S. Talwalkar, John C. Duchi, Rean Griffith, Michael J. Franklin, and Michael I. Jordan. Mlbase: A distributed machine-learning system. In *Conference on Innovative Data Systems Research*, 2013.

[60] T. Kuflik, J. Kay, , and B. Kummerfeld. Challenges and solutions of ubiquitous user modeling, in ubiquitous display environments. *Springer-Verlag*, page 7–30, 2012.

[61] Jialing Li, Kangjun Bai, Lingjia Liu, and Yang Yi. A deep learning based approach for analog hardware implementation of delayed feedback reservoir computing system. *2018 19th International Symposium on Quality Electronic Design (ISQED)*, pages 308–313, 2018. doi: 10.1109/ISQED.2018.8357305.

[62] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, pages 50–60, 2018.

[63] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *arXiv preprint*, arXiv:1812.06127, 2018.

[64] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. *arXiv preprint*, arXiv:1907.02189, 2019.

[65] Chunxiao Lin, Yibin Liang, and Yang Yi. Fpga-based reservoir computing with optimized reservoir node architecture. *2022 23rd International Symposium on Quality Electronic Design (ISQED)*, pages 1–6, 2022. doi: 10.1109/ISQED54688.2022.9806247.

[66] Shiya Liu, Yibin Liang, Victor Gan, Lingjia Liu, and Yang Yi. Accurate and efficient quantized reservoir computing system. *2020 21st International Symposium on Quality Electronic Design (ISQED)*, pages 364–369, 2020. doi: 10.1109/ISQED48828.2020.9136986.

[67] Shiya Liu, Lingjia Liu, and Yang Yi. Quantized reservoir computing on edge devices for communication applications. *2020 IEEE/ACM Symposium on Edge Computing (SEC)*, pages 445–449, 2020. doi: 10.1109/SEC50012.2020.00068.

[68] Shiya Liu, Dong Sam Ha, Fangyang Shen, and Yang Yi. Efficient neural networks for edge devices. *Computers and Electrical Engineering*, 92:107121, 2021.

[69] Brendan McMahan and Abhradeep Thakurta. Federated learning with formal differential privacy guarantees, 2022. URL `https://ai.googleblog.com/2022/02/federated-learning-with-formal.html`.

[70] H Brendan McMahan, Eider Moore, Daniel Ramage, and Seth Hampson. Communication-efficient learning of deep networks from decentralized data. *arXiv preprint*, arXiv:1602.05629, 2016.

[71] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. *In International Conference on Learning Representations*, 2018.

[72] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning. *arXiv preprint*, arXiv:1805.04049, 2018.

[73] F. Mo, A. S. Shamsabadi, K. Katevas, S. Demetriou, I. Leontiadis, A. Cavallaro, and H. Haddadi. Darknetz:towards model privacy at the edge using trusted execution environments. *International Conference on Mobile Systems, Applications, and Services (MobiSys)*, pages 161–174, 2020.

[74] Nima Mohammadi, Jianan Bai, Qiang Fan, Yifei Song, Yang Yi, and Lingjia Liu. Differential privacy meets federated learning under communication constraints. *CoRR*, abs/2101.12240, 2021. URL `https://arxiv.org/abs/2101.12240`.

[75] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. *IEEE Symposium on Security and Privacy*, pages 111–125, 2008.

[76] Fabiha Nowshin, Yuhao Zhang, Lingjia Liu, and Yang Yi. Recent advances in reservoir computing with a focus on electronic reservoirs. *2020 11th International Green and Sustainable Computing Workshops (IGSC)*, pages 1–8, 2020. doi: 10.1109/IGSC51522.2020.9290858.

[77] Nicolas Papernot, Martin Abadi, Ulfar Erlingsson, Ian Goodfellow, and Kunal Talwar. Semi-supervised knowledge transfer for deep learning from private training data. *arXiv preprint*, arXiv:1610.05755, 2016.

[78] Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Ulfar Erlingsson. Scalable private learning with pate. *In International Conference on Learning Representations*, arXiv:1802.08908, 2018.

[79] Xavier Pita. Divided attention could ease wireless congestion., 2021. URL `https://www.newswise.com/articles/divided-attention-could-ease-wireless-congestion`.

[80] Apostolos Pyrgelis, Carmela Troncoso, and Emiliano De Cristofaro. Knock knock, who's there? membership inference on aggregate location data. *arXiv preprint*, arXiv:1708.06145, 2017.

[81] Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. *arXiv preprint*, arXiv:1909.13014, 2019.

[82] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1310–1321, 2015.

[83] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. *IEEE Symposium on Security and Privacy*, pages 3–18, 2017.

[84] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet Talwalkar. Federated multi-task learning. *arXiv preprint*, arxiv.1705.10467, 2017.

[85] Shuang Song, Kamalika Chaudhuri, and Anand D Sarwate. Stochastic gradient descent with differentially private updates. *2013 IEEE Global Conference on Signal and Information Processing*, pages 245–248, 2013.

[86] Om Thakkar, Galen Andrew, and H. Brendan McMahan. Differentially private learning with adaptive clipping. *arXiv*, arXiv:1905.03871, 2019.

[87] Aleksei Triastcyn and Boi Faltings. Generating artificial data for private deep learning. *arXiv preprint*, arXiv:1803.03148, 2018.

[88] Lionel Sujay Vailshery. Number of internet of things (iot) connected devices worldwide from 2019 to 2021, with forecasts from 2022 to 2030, 2022. URL `https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/`.

[89] C.H. van Berkel. Multi-core for mobile phones. *Proceedings of Conference on Design, Automation and Test in Europe*, pages 1260–1265, 2009.

[90] Yang Yi, Yongbo Liao, Bin Wang, Xin Fu, Fangyang Shen, Hongyan Hou, and Lingjia Liu. Fpga based spike-time dependent encoder and reservoir design in neuromorphic computing processors. *Microprocess. Microsyst.*, page 175–183, oct 2016.

[91] Chuanting Zhang, Shuping Dang, Basem Shihada, and Mohamed-Slim Alouini. Dual attention-based federated learning for wireless traffic prediction. *IEEE Conference on Computer Communications*, pages 1–10, 2021.

[92] Chenyuan Zhao, Wafi Danesh, Bryant T. Wysocki, and Yang Yi. Neuromorphic encoding system design with chaos based cmos analog neuron. *2015 IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*, pages 1–6, 2015. doi: 10.1109/CISDA.2015.7208631.

[93] Chenyuan Zhao, Bryant T. Wysocki, Yifang Liu, Clare D. Thiem, Nathan R. McDonald, and Yang Yi. Spike-time-dependent encoding for neuromorphic processors. *J. Emerg. Technol. Comput. Syst.*, 12, 2015.

[94] Chenyuan Zhao, Jialing Li, Lingjia Liu, Lakshmi Sravanthi Koutha, Jian Liu, and Yang Yi. Novel spike based reservoir node design with high performance spike delay

loop. *Proceedings of the 3rd ACM International Conference on Nanoscale Computing and Communication*, 2016.

[95] Chenyuan Zhao, Jialing Li, and Yang Yi. Making neural encoding robust and energy efficient: An advanced analog temporal encoder for brain-inspired computing systems. *2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 1–6, 2016. doi: 10.1145/2966986.2967052.

[96] Chenyuan Zhao, Bryant Wysocki, Clare Thiem, Nathan McDonald, Jialing Li, Lingjia Liu, and Yang Yi. Energy efficient spiking temporal encoder design for neuromorphic computing systems. *IEEE Transactions on Multi-Scale Computing Systems*, PP, 09 2016. doi: 10.1109/TMSCS.2016.2607164.

[97] Chenyuan Zhao, Bryant Wysocki, Clare Thiem, Nathan McDonald, Jialing Li, Lingjia Liu, and Yang Yi. Energy efficient spiking temporal encoder design for neuromorphic computing systems. *IEEE Transactions on Multi-Scale Computing Systems*, PP, 2016. doi: 10.1109/TMSCS.2016.2607164.

[98] Chenyuan Zhao, Bryant T. Wysocki, Clare D. Thiem, Nathan R. McDonald, Jialing Li, Lingjia Liu, and Yang Yi. Energy efficient spiking temporal encoder design for neuromorphic computing systems. *IEEE Transactions on Multi-Scale Computing Systems*, 2(4):265–276, 2016. doi: 10.1109/TMSCS.2016.2607164.

[99] Chenyuan Zhao, Jialing Li, Hongyu An, and Yang Yi. Energy efficient analog spiking temporal encoder with verification and recovery scheme for neuromorphic computing systems. *2017 18th International Symposium on Quality Electronic Design (ISQED)*, pages 138–143, 2017. doi: 10.1109/ISQED.2017.7918306.

[100] Chenyuan Zhao, Yang Yi, Jialing Li, Xin Fu, and Lingjia Liu. Interspike-interval-based analog spike-time-dependent encoder for neuromorphic processors. *IEEE Transactions*

on Very Large Scale Integration (VLSI) Systems, 25(8):2193–2205, 2017. doi: 10.1109/TVLSI.2017.2683260.

[101] Chenyuan Zhao, Kian Hamedani, Jialing Li, and Yang Yi. Analog spike-timing-dependent resistive crossbar design for brain inspired computing. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 8(1):38–50, 2018. doi: 10.1109/JETCAS.2017.2765892.

[102] Chenyuan Zhao, Lingjia Liu, and Yang Yi. Design and analysis of real time spiking neural network decoder for neuromorphic chips. *Proceedings of the International Conference on Neuromorphic Systems*, 2019.

[103] Chenyuan Zhao, Qiyuan An, Kangjun Bai, Bryant Wysocki, Clare Thiem, Lingjia Liu, and Yang Yi. Energy efficient temporal spatial information processing circuits based on stdp and spike iteration. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 67(10):1715–1719, 2020. doi: 10.1109/TCSII.2019.2945690.

[104] Honghao Zheng, Juliet Anderson, and Yang Yi. Approaching the area of neuromorphic computing circuit and system design. In *2021 12th International Green and Sustainable Computing Conference (IGSC)*, pages 1–8, 2021. doi: 10.1109/IGSC54211.2021.9651627.

[105] Honghao Zheng, Nima Mohammadi, Kangjun Bai, and Yang Yi. Low-power analog and mixed-signal ic design of multiplexing neural encoder in neuromorphic computing. *2021 22nd International Symposium on Quality Electronic Design (ISQED)*, pages 154–159, 2021. doi: 10.1109/ISQED51717.2021.9424267.