Task-Boundary Agnostic Continuous Federated Learning using Online Variational Bayes

A Project Report

submitted by

SHIVAKANTH REDDY

in partial fulfilment of the requirements
for the award of the degree of

MASTER OF TECHNOLOGY

ELECTRONICS AND COMMUNICATION ENGINEERING
INDRAPRASTHA INSTITUTE OF INFORMATION TECHNOLOGY DELHI

NEW DELHI- 110020

August, 2024

THESIS CERTIFICATE

This is to certify that the thesis titled **Task-Boundary Agnostic Continuous Federated Learning using Online Variational Bayes**, submitted by **Shivakanth Reddy**, to the Indraprastha Institute of Information Technology, Delhi, for the award of the degree of **Master of Technology**, is a bonafide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr.Ranjitha Prasad
Thesis Supervisor
Associate Professor
Dept. of Electronics and Communication
IIIT Delhi, 110020

Place: New Delhi

Date: 04th August 2024

ACKNOWLEDGEMENTS

I extend my deepest appreciation to my advisor, Dr. Ranjitha Prasad, for her supervision, guidance, and unwavering support throughout my thesis. Her mentorship has been invaluable, especially during challenging academic times, and I am profoundly grateful for the opportunities she has provided.

I am extremely thankful to Bhaswanth Gudimella (M.Tech) for his assistance and support in this project. It gives me immense pleasure to thank all of the members of the Intellicom Lab for their inspiration and patient support. Additionally, I extend my sincere gratitude to the IIITD administration for facilitating all necessary facilities for my thesis.

Lastly, I express heartfelt thanks to my parents and friends for their constant encouragement and motivation throughout this journey.

ABSTRACT

Federated learning (FL) is a privacy-preserving machine learning approach that enables the training of models across multiple decentralized edge devices without exchanging raw data. However, local models trained only on local data often fail to generalize well to unseen samples. Moreover, in the context of an end-to-end ML model at scale, it is not feasible to repeatedly train from scratch whenever new data arrives. Therefore, it is essential to employ continual learning to update models on the fly instead of retraining them from scratch. Continual Federated Learning enhances the efficiency, privacy, and scalability of federated learning systems by learning new tasks while preventing catastrophic forgetting of previous tasks. The primary challenge of Continual Federated Learning is global catastrophic forgetting, where the accuracy of the global model trained on new tasks declines on the old tasks.

In this work, we propose a novel strategy, Bayesian Gradient Descent in Continual Federated Learning(CFL-BGD) to overcome catastrophic forgetting. We derive new local optimization problems, based on Bayesian continual learning and FL principles. We conduct extensive experiments on Permuted MNIST and Split MNIST without task boundaries, demonstrating the effectiveness of our method in handling non-IID data distributions with varying levels of heterogeneity, and in mitigating global catastrophic forgetting. Unlike other continual learning methods like EWC, which take some core action based on task boundaries, our approach does not require any knowledge of task boundaries, making it more versatile and practical. The results show that our method significantly improves the performance and robustness of the global model across various tasks, highlighting the potential of our strategy in real-world federated learning applications.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOTATION

α	Dirichlet distribution parameter defines degree of hetereogenity
θ	Parameters of the model
$\mathcal{D}^n$	Dataset at the n-th task
$\mathbb{E}_x$	Expectation taken with respect to x
$\boldsymbol{\phi}$	Parameters of the parametric distribution
$\boldsymbol{\phi}^*$	Optimal parameters of the parametric distribution
$\mathcal{C}_t$	Set of iterations at which model aggregation occurs
$\mu_{k,t}(m)$	Mean parameter of client k's local model in round t
$\sigma_{k,t}(m)$	Standard deviation parameter of client k's local model in round t
η	Learning rate, controlling the step size in parameter updates
$L^n_{k,t}(\theta)$	Loss function of client k on n th task, dependent on model parameters θ
$\epsilon_{k,t}(m)$	Noise term specific to client k in round t
$\mu^{n-1}_{w,t}(m)$	Aggregated mean of global model parameters at round t
$\sigma^{n-1}_{w,t}(m)$	Aggregated variance of global model parameters at round t
K	Total number of clients participating in the federated learning process
$p(D, \theta)$	Joint probability distribution of dataset D and parameters θ
$p(D\|\theta)$	Likelihood function of the dataset D given parameters θ
$p(\theta)$	Prior distribution of the parameters θ
$p(\theta\|D)$	Posterior distribution of the parameters θ given dataset D
$Q(\theta\|\boldsymbol{\phi})$	Parametric distribution used in variational Bayes
$\mathcal{N}(\mu, \sigma)$	Gaussian distribution with mean μ and standard deviation σ

CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

General neural networks have the ability to learn the task at hand Mittal *et al.* (2021); however, when the same model is trained on new tasks, it tends to override the earlier ones, a problem known as catastrophic forgetting. Addressing this issue is important for enabling models to adapt to real-world scenarios. Continual learning, inspired by the human learning process, allows models to train on sequentially arriving tasks, mitigating catastrophic forgetting Yoon *et al.* (2021).

With the advent of technology, vast amounts of data are being generated by millions of devices, creating rich sources for training models in areas such as image, language, and speech processing. However, storing this data and ensuring its availability for training poses significant challenges, including privacy concerns for users. Federated Learning (FL) was introduced to address these issues by enabling end devices to train on their data without sharing it with a central server, thus simultaneously solving data privacy and storage challenges McMahan *et al.* (2023)

Continual Federated Learning combines the principles of continual learning and federated learning to address the unique challenges posed by both domains. It enables decentralized models to learn from sequentially arriving tasks across multiple edge devices without exchanging raw data, thereby preserving user privacy and improving model adaptability. This approach not only prevents catastrophic forgetting but also enhances the scalability and efficiency of the learning process.

In this work, we propose **CFL_BGD**, which refers to Continual federated learning using Bayesian gradient descent in a task-agnostic scenario.

- Addressing the challenge of catastrophic forgetting in Task-Agnostic Continual Federated Learning, where client-side adaptation occurs without explicit task boundaries.
- Using Online Variational Bayes to come up with an update rule for client models and global model.

1.2 Federated Learning

Federated learning trains machine learning models in a decentralized manner, where many clients each have their own data. The global model, shared by the server among clients, is sent to K clients for remote training using their local data, with each client having its own n_k samples. After training, the clients update their weights according to their local data. These updated weights are then sent back to the global server. The server aggregates all the models to update the global model. This method ensures data protection on the client side, as only model updates are shared, not the raw data. Unlike centralized training, where all data must be sent to a single location for training, federated learning maintains data privacy by keeping it decentralized. There is no direct communication between clients; they only communicate with the server. This approach leverages diverse datasets while ensuring data integrity and confidentiality, making it particularly useful in privacy-sensitive applications like healthcare, finance, and mobile devices.

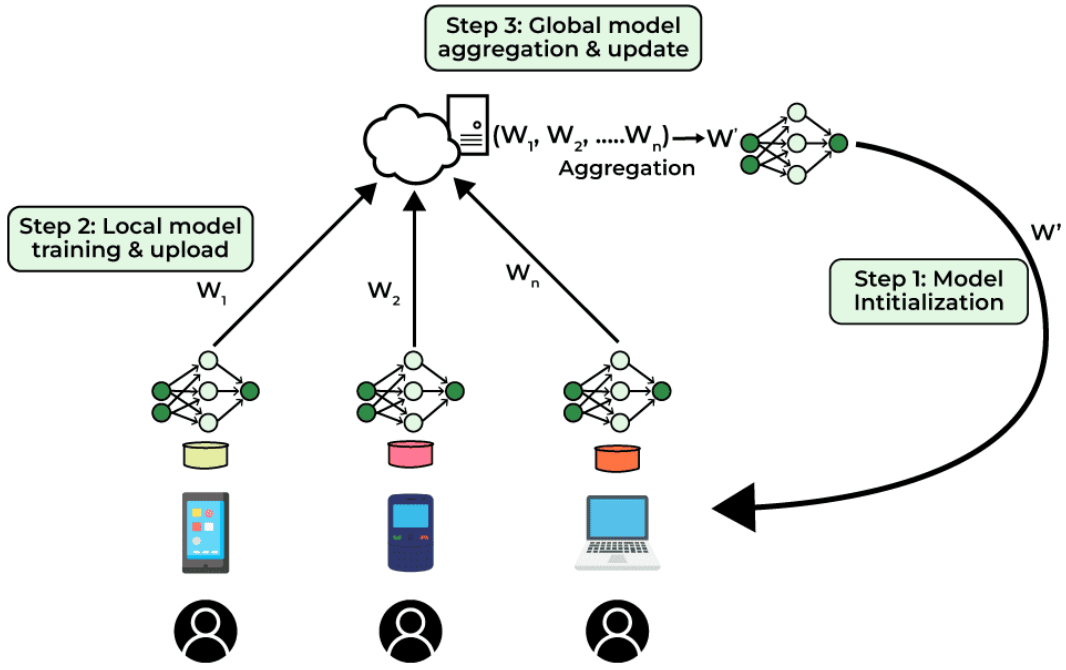


Figure 1.1: Federated learning process

The following are the detailed steps of federated learning in one communication round:

- **Step 1:** According to the use case, select a global model.

- **Step 2:** Send the global model among K clients.
- **Step 3:** Train the local model at each client on their respective local data.
- **Step 4:** After local training, send the updates to the server.
- **Step 5:** Aggregate all local model updates at the server.

1.2.1 Benefits of Federated Learning

- **Enhance Privacy and Security:** Federated learning allows machine learning models to be trained without sending raw data from local devices to a central server. By ensuring that critical data stays on the user's device, this decentralized approach lowers the possibility of data breaches and protects user privacy
- **Reduce Data Transfer Costs** Federated learning reduces the quantity of data that must be transferred over networks by processing data locally on devices and only communicating model updates to a central server. This reduction in data transfer can lead to lower bandwidth usage and cost savings, particularly important in environments with limited connectivity.
- **Improved Data Diversity:** Models trained in federated learning use data from many different contexts and sources. In contrast to models trained on centralized data, which might not capture all conceivable changes, this diversified data might assist in constructing more robust and generalizable models that function well across numerous circumstances.
- **Compliance with Regulations:** Many industries, such as healthcare and finance, are subject to strict regulations regarding data privacy and protection. Federated learning helps organizations comply with these regulations by ensuring that sensitive data never leaves the local devices, making it easier to adhere to legal and regulatory requirements while still leveraging the power of machine learning.

1.2.2 Challenges in Federated Learning :

- **Statistical heterogeneity:** As federated learning is a decentralized framework, it may have local devices in many different geographical locations, and the data at the different clients may have different distributions; because of this non-IID nature in the data, the global model faces challenges, we have calculated global fisher in order to reduce the effect of this problem.
- **Communication Efficiency:** In the federated learning framework, communication between server and client is challenging; we expect the global model to converge faster in a few rounds so that communication between server and clients decreases.
- **Scalability:** There are significant challenges with an increase in the number of clients in federated learning, such as communication overhead and latency in communication, and resource management is also important when the number of clients is increased.

1.3 Continual Learning

Continual learning addresses the challenge of training machine learning models on new data without needing to retrain from scratch. We have situations where we train a model and right after the training new data comes in, now if we want to train the same model we need to add this new data to the old dataset and we need to train the model from scratch which is a cumbersome task to do because we know training deep learning models is highly computational expensive hence Incrementally training our model on continuously arriving new data, we can update models sequentially on the stream of tasks.

In a typical scenario, we might have a neural network that is trained on Task A data, the weights in the network are trained such that it performs well on Task A, and when new data Task B arrives the same model which is trained on Task A data is now trained on Task B data the then the weight of the network which are crucial in performing Task A are now changed to perform Task B, this phenomenon is called as Catastrophic forgetting, Kirkpatrick *et al.* (2017). Continual learning methods help to reduce Catastrophic forgetting.

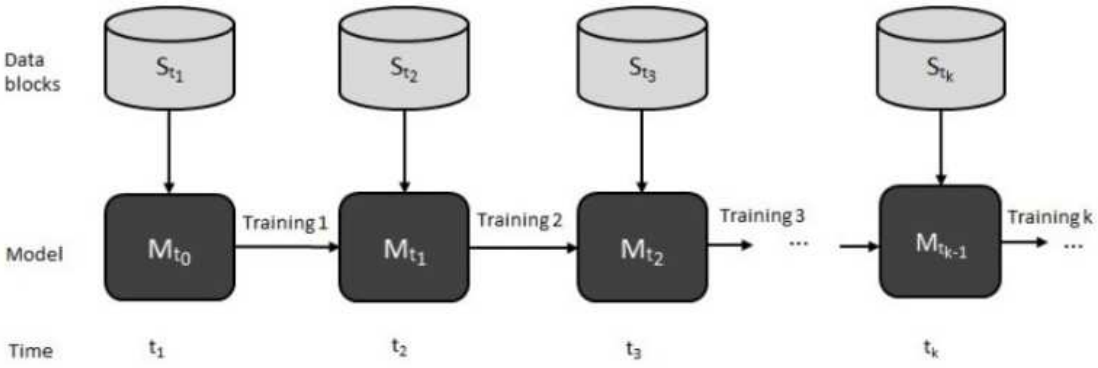


Figure 1.2: In this figure, we can see that model M_t is getting updated with continuous streams of data

1.3.1 Benefits of Continual Learning

- **Adaptability :** Federated learning allows machine learning models to be trained without sending raw data from local devices to a central server. By ensuring that critical data stays on the user's device, this decentralized approach lowers the possibility of data breaches and protects user privacy
- **Scalability :**It allows for the efficient use of computational resources by updating

models incrementally rather than retraining from scratch, making it scalable for large datasets and complex tasks.

- **Resource Efficiency:** Continual learning can reduce the need for extensive data storage and repeated training cycles, optimizing memory and processing power usage.

1.3.2 Challenges in Continual Learning :

Catastrophic Forgetting: One of the most significant challenges is catastrophic forgetting, training on new tasks causes the model to forget previously learned tasks. This happens because the model's weights are modified for the new task.

Balancing Stability and Plasticity: In the federated learning framework, communication between server and client is challenging; we expect the global model to converge faster in a few rounds so that communication between server and clients decreases.

Incremental learning can be more promising in modern machine learning; instead of training the model from scratch, we can use incremental learning to update the model on continuously evolving tasks. That way, we can make machine learning models adaptable and scalable. If we consider some real-world examples, it can be used in Autonomous vehicles where the model observes different contexts(for example, different new obstacles and lanes) from the fleets, sends the data to its servers, and updates the model with the new data. If we consider Language modeling as an example using incremental learning helps in learning different patterns in the data.

1.4 Contributions

We are incorporating incremental learning in the federated setup to leverage the incremental learning benefits, we have client models that have new data coming in continuously here comes the problem as we discussed earlier is Catastrophic forgetting as the client model trains on a new set of tasks it tends to forget the old tasks, in this work we ensure the client model continuously learn the evolving tasks with the decrease in forgetting previously learned tasks without knowing the task boundaries also combine these incrementally trained client models in federated setting, we develop a new method

where combined global model performs well on previously trained tasks which reduce catastrophic forgetting at the global level over the communication rounds.

1.4.1 Notations

Small letters denote scalars, boldface small letters denote vectors. $\mathbf{I}$ denotes an identity matrix whose size is as per context. The ℓ_2-norm of a vector $\mathbf{x}$ is denoted as $||\mathbf{x}||$. $\mathcal{P}$ represents sets and $|\mathcal{P}|$ represents size of the set.

CHAPTER 2

RELATED WORKS

In this chapter, we discuss papers in literature that are related to the proposed framework. Essentially, we first review well-known approaches in federated learning and discuss how none of these address the challenges in continual learning. Then, we review ***

2.1 Federated Learning

Federated algorithms have widespread applications such as predicting credit risk Xu *et al.* (2023) where the data is distributed across different banks, healthcare applications Joshi *et al.* (2022) for classifying medical images, applied in object recognition tasks Hegiste *et al.* (2023) where the data is distributed across multiple devices, robotics applications for training models to control robotic systems, smart home applications Zhang *et al.* (2022*a*) for predicting energy consumption, predicting stock prices, autonomous vehicle applications Chellapandi *et al.* (2023) for predicting road conditions, and several privacy preserving scenarios. The first technique that came about was FedAvg, proposed by McMahan *et al.* (2023), which is a simple and efficient federated learning algorithm that uses a weighted average of the local model updates from participating clients (not all clients are active all the time) to update the global model. Despite its simplicity, FedAvg has achieved good performance in various applications.

FedProx, proposed by Li *et al.* (2020), introduces a proximal term to the FedAvg algorithm to account for the heterogeneity in client data and improve the performance of the federated learning system. The proximal term encourages the clients to update their models towards a proximal solution of the global model, which helps to mitigate the effect of non-IID data distribution. FedBN Li *et al.* (2021) is a federated learning technique to address the problem of non-IID data distribution in the context of batch normalization (BN) layers in deep neural networks. When using BN layers in federated learning, clients with non-IID data distributions may have different means and variances

of their input data, which might lead to problems. FedBN solves this issue by enabling the clients to update the BN parameters locally, which are combined to produce the global BN parameters. As a result, clients with varied data distributions can employ various BN parameters, which enhances the model's performance when applied to local data. Among Bayesian methods, pFedBayes, proposed by Zhang *et al.* (2022*b*), is a federated learning algorithm designed to address the challenges posed by heterogeneity in client data. Unlike traditional federated learning algorithms that primarily focus on aggregating models, pFedBayes leverages Bayesian learning principles (Variational Inference) to personalize the learning process for each client. Although these methods effectively address privacy-preserving machine learning, they do not address challenges posed by continual learning.

2.2 Continual Learning

As a general notion in human learning, if one is exposed to new information, he/she tends to forget the old concepts. However, in humans, the old concept manifests in various ways, particularly in the context of memory and learning. It is expected that a similar phenomenon occurs with machine learning models, where, if it is trained on old samples, the model needs to remember some concepts and learning even when exposed to new information. However, machines do not have any intrinsic mechanism to preserve learning in previous tasks, and this is aspect is technically termed as "Catastrophic forgetting". Ideally, we would prefer that the model remembers old concepts, and hence, several methods to mitigate catastrophic forgetting have been proposed.

To solve this problem of catastrophic forgetting, literature consists of rehearsal-based methods which incorporates a fixed set of memory called memory buffer the samples from the old task are stored. While training samples from this memory buffer are included, i.e., we pass a few old samples along with new samples and hence, it becomes an active recall for the models and hence this avoids catastrophic forgetting Masana *et al.* (2020). Another class of methods include regularization-based methods in which we introduce new terms in the loss function along with normal cross-entropy loss, now we minimize the combined loss function classification loss on old samples and distillation loss for old and new samples; this process is done by using knowledge

distillation specifically self-distillation any deep learning architecture can adopt this incremental learning only requirement is to modify the loss function with incremental loss. We also delve into Bayesian methods for continual learning.

2.2.1 Rehearsal Based Methods

To address the challenge of catastrophic forgetting, rehearsal-based methods have evolved as promising approaches. These methods utilize memory buffers, also known as exemplar sets, to store the previous samples of data, which are used in training to rehearse previously learned experiences to consolidate learning and also help in performing better on previous tasks. One of the well-known contributions in this field is GEM Lopez-Paz and Ranzato (2022), where memory buffers are maintained to store the past data samples, and during the training of new tasks GEM constraints the updates by projecting the gradients onto subspace spanned by past gradients stored in the memory buffer. This helps the model to learn the new tasks while retaining the knowledge learnt in previous training. Instead of storing the previous data samples directly and using them in training the current model, authors in Shin *et al.* (2017) propose a novel technique which is the combination of exemplar memory and generative modeling to avoid the catastrophic forgetting problem. Here, synthetic data is generated using generative modeling and these generated synthetic data is used in training deep learning models so as they serve as good as exemplar memory.

In class incremental learning problem setting, there are per-class fixed-size sets used in training the model Mittal *et al.* (2021). Class-incremental learning aims to develop a unified classifier from sequentially arriving data of different classes. Data is received incrementally in batches, each containing images from specific classes. Each batch is considered a task, and with each incremental step, new task data containing samples of new classes is introduced. At each step, only the complete data for the new classes is available, while a small amount of exemplar data from previous classes is retained in a limited-size memory buffer. The model is then expected to classify all the classes encountered so far, additionally Mittal *et al.* (2021) uses knowledge distillation.

2.2.2 Regularization Based Methods

Regularization-based methods introduce new terms in the loss function in addition to the standard cross-entropy loss. The combined loss function minimizes both classification loss on old samples and distillation loss for both old and new samples. This process is achieved through knowledge distillation, specifically self-distillation, where any deep learning architecture can adopt incremental learning by modifying its loss function to include an incremental loss.

As the storing data from the previous task has memory limitations so as to avoid storing data, In Dhar *et al.* (2019), the authors propose a method for reducing memorization in machine learning models. The collection of previous data is also not advisable as edge devices have limited memory. The novel approach here is the idea of preserving base class information without storing the samples, which is incorporated using the Attention distillation loss. It introduces the student-teacher model where the teacher is trained using base class, and the student incrementally trains when new data arrives and also ensures it performs well on the base task. In Rosasco and Villa (2015), the authors introduce a scheme where the regularization parameters are updated incrementally. This algorithm focuses on improving learning by optimizing added regularization term to the loss function. They also discuss the importance of early stopping in incremental updates to reduce the empirical risk. For Class Incremental Learning, the authors in Yan *et al.* (2021) propose a new method for representation learning which is a two-stage learning approach that uses dynamically expanded representation to enable incremental idea modeling to be more successful. Existing incremental learning techniques have a stability-plasticity problem; in simpler terms, high stability means not learning new concepts, while high plasticity means performance decreases while learning new concepts, referred to as catastrophic forgetting. In this framework, the authors introduce a method of representing data that essentially retains previous knowledge, and when new data arrives, the old ones are frozen, and it adapts to new class features as it maintains feature extractors of different sizes.

In general, continual learning literature has several approaches based on representation learning, and these can be categorized into three main categories namely regularization-based Zhao *et al.* (2024), distillation-based Li *et al.* (2024), and structure-based methods Kumar *et al.* (2020); after regularization, it is important to take care of

class imbalance issues; different approaches are used to avoid class imbalance Kim *et al.* (2020). Similarly, Elastic weight Consolidation (EWC) is inspired by synaptic intelligence; EWC mimics the behavior of the brain by restricting network parameters that are important for achieving previous tasks to stay close to their old values. In the context of DNNs, Learning, this translates to adjusting the weights and biases to optimize performance. This importance given to weights is calculated using the Fisher Information matrix, which adds weightage to the loss function of each parameter according to their significance in previous tasks. EWC adeptly balances stability (retaining old knowledge) and plasticity (learning new knowledge), enabling models to acquire new information without substantially forgetting prior knowledge Kirkpatrick *et al.* (2017) Aich (2021).

2.2.3 Bayesian Methods

In continual learning, data arrives sequentially. In Bayesian methods, the idea is to provide a solution to the continual learning problem using Baye's rule. Essentially, the posterior distribution is updated as and when new data comes in, using the posterior for the previous task as prior for the current task. In Nguyen *et al.* (2018), the authors combine online variational inference and Monte Carlo VI for neural networks and use Bayes by backprop. In Blundell *et al.* (2015), authors use mean-field approximation assumption where the weight distributions are independent. In the Bayesian Incremental learning approach Kochurov *et al.* (2018), a Bayesian approach is used to update posterior on sequential tasks. Specifically, it uses several approximations to calculate posterior such as fully factorized Gaussian approximation, channel factorized Gaussian approximation, and multiplicative normalizing flow approximations. These algorithms are designed such that an appropriate action will be taken based on the task switch.

2.3 Federated Continual Learning

Federated learning has been a promising method for privacy-preserving machine learning where clients train on their local data, while sharing only the parameters with the server. Further, a global model is obtained at the server, and this global model is shared with the clients. In real-world scenarios, the data at the client is not constant and it varies

according to the time, and the global model now does not remember the previous tasks on which it trained, which constitutes catastrophic forgetting in FL. Since centralised methods cannot be directly used, the authors in CFeD proposed Ma *et al.* (2022) where they discuss i) Client division mechanism where the tasks are divided as reviewing old tasks and learning new tasks so the clients will be assigned with one of these tasks. ii) Knowledge distillation on surrogate Data, it uses surrogate dataset for each client and it distills the knowledge from old tasks to a new task, thus it tries to mitigate the problem of catastrophic forgetting and combining these two methods client division mechanism and knowledge distillation.

Among existing continual federated learning methods, Yoon *et al.* (2021) proposes the decomposition of model parameters into dense global parameters and sparse task-specific parameters to retain knowledge from past tasks. To tackle catastrophic forgetting, Dong *et al.* (2022) uses global and local level loss compensations to tackle local as well as global forgetting. On the other hand, Bakman *et al.* (2023) projects the global updates of new tasks into orthogonal subspace of previous tasks. Dupuy *et al.* (2023*b*) proposes a way to quantify catastrophic forgetting and alleviate it using replay memory. Federated continual learning poses new challenges in terms of learning as the different clients are involved and there are problems such as inter-client interference. Clients learn on continuously arriving tasks sequence but the task order of tasks is unknown, and making this learning happen effectively without interference is important, also in the federated setting we always need to see the communication cost for the transferring parameters between clients and server and this will be crucial when it has the large number of tasks to train. To tackle catastrophic forgetting, Dong *et al.* (2023) uses global and local level loss compensations to tackle local as well as global forgetting. On the other hand, Bakman *et al.* (2023) projects the global updates of new tasks into the orthogonal subspace of previous tasks. Dupuy *et al.* (2023*a*) proposes a way to quantify catastrophic forgetting and alleviate it using replay memory.

Despite several recent advances in this area, current literature lacks task-boundary agnostic methods for Continual Federated Learning (CFL) algorithms that effectively alleviate catastrophic forgetting.

CHAPTER 3

METHODOLOGY

3.1 Introduction

Continual learning allows models to learn from a sequence of tasks over time, mitigating the issue of catastrophic forgetting where learning new tasks can interfere with previously learned ones. In task-boundary agnostic scenarios, traditional methods of continual learning are often unsuitable because they depend on detecting task switches. Since the number of tasks is unknown, identifying task switches becomes challenging.

3.1.1 Continual Learning and Task Boundaries

Task boundaries are explicit demarcations between different tasks during training. Task-boundary aware methods such as EWC and online EWC are designed to preserve knowledge from previous tasks by selectively slowing down the learning on certain weights that are important for previous tasks Kirkpatrick *et al.* (2017). Techniques like LwF leverages knowledge distillation, where the model is trained on new tasks while using predictions from the model trained on previous tasks to ensure it retains knowledge of the old tasks Li and Hoiem (2017).These task-boundary aware approaches helps maintain performance on old tasks while learning new ones.

In contrast, task-boundary agnostic methods like the Online Variational Bayes do not require explicit task boundaries Zeno *et al.* (2021). Instead, they adapt continuously without knowing when tasks switch. This is beneficial in real-world scenarios where task boundaries may not be clearly defined or available, enabling a more fluid and adaptable learning process.

The proposed Task-Agnostic Continuous Federated Learning distinguishes itself by utilizing online variational Bayes, which allows for continual adaptation in a federated learning environment where data arrives in a non-stationary and task-agnostic manner.

This approach aims to reduce catastrophic forgetting and improve scalability and adaptability without relying on explicit task boundaries.

3.2 Mathematical Preliminaries

3.2.1 Federated Learning

In a typical Federated Learning (FL) setting, N edge devices periodically communicate the parameters $\boldsymbol{\theta}$ (or incremental updates) to a central server to solve the following finite-sum unconstrained optimization problem:

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^d} f(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}\in\mathbb{R}^d} \sum_{i=1}^{N} p_i f_i(\boldsymbol{\theta}),$$

where $f_i(\boldsymbol{\theta})$ is the local loss function at the i-th client and $f(\boldsymbol{\theta})$ is the global loss function. Each client independently computes gradients on its local static data and subsequently transmits these gradients to the central server. The central server receives these updates from all clients and aggregates them using some predefined strategy, such as averaging the parameter updates.

3.2.2 Continual Learning using a Bayesian Framework

The process of Bayesian inference fundamentally revolves around creating a robust probability model that characterizes the uncertainty about both the dataset and the model parameters. Central to this process is the joint probability distribution, which integrates these aspects into a cohesive framework. This joint probability distribution can be expressed as the product of two distributions:

$$p(\mathcal{D}, \boldsymbol{\theta}) = p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta}), \tag{3.1}$$

where $p(\mathcal{D}|\boldsymbol{\theta})$ denotes the likelihood function of the dataset $\mathcal{D}$, and $p(\boldsymbol{\theta})$ represents the prior distribution of the parameters $\boldsymbol{\theta}$. The posterior distribution is calculated using Baye's rule:

$$p(\boldsymbol{\theta}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathcal{D})}, \tag{3.2}$$

where $p(\mathcal{D})$ is obtained using the sum rule.

In Continual learning, data arrives sequentially $(\mathcal{D}^1, \mathcal{D}^2, \ldots, \mathcal{D}^n)$ and the posterior distribution is updated each time new data comes in. At each step, the previous posterior distribution is used as the new prior distribution for the current task. Thus, according to Bayes' theorem, the posterior probability distribution at time n is given by:

$$p(\boldsymbol{\theta}|\mathcal{D}^n) = \frac{p(\mathcal{D}^n|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{D}^{n-1})}{p(\mathcal{D}^n)}. \tag{3.3}$$

However, calculating the posterior distribution is often intractable for practical probability models. Therefore, we approximate the true posterior using variational methods.

3.2.3 Variational Bayes

Variational Bayes, as proposed in Graves (2011), is used to approximate the posterior distribution $p(\boldsymbol{\theta}|\mathcal{D})$ using a parametric distribution $q(\boldsymbol{\theta}|\boldsymbol{\phi})$, for practical neural network models, where $\boldsymbol{\theta}$ are model parameters and $\boldsymbol{\phi}$ are the variational parameters. The objective is to minimize the Kullback-Leibler (KL) divergence between the approximate distribution and the true posterior distribution, written as:

$$\mathrm{KL}(q(\boldsymbol{\theta}|\boldsymbol{\phi})\|p(\boldsymbol{\theta}|\mathcal{D}) = \mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\phi})}\left[\log q(\boldsymbol{\theta}|\boldsymbol{\phi}) - \log p(\boldsymbol{\theta}|\mathcal{D})\right] \tag{3.4}$$

We find the best approximation of $q(\boldsymbol{\theta}|\boldsymbol{\phi})$ such that the KL divergence is minimized. Therefore, we need those parameters $\boldsymbol{\phi}$ of the distribution $q(\boldsymbol{\theta}|\boldsymbol{\phi})$ which minimize the KL divergence. The optimization problem can be stated as:

$$\varphi^* = \arg\min_{\boldsymbol{\phi}} \mathrm{KL}(Q(\boldsymbol{\theta}|\boldsymbol{\phi})\|P(\boldsymbol{\theta}|\mathcal{D})) \tag{3.5}$$

3.3 Proposed Formulation

In the context of Federated Learning (FL), we consider a scenario where there are K clients and n tasks arriving sequentially at each client. The goal is to update the posterior distribution at each client based on the received data.

For the n-th task at the k-th client, we have

$$p_k(\boldsymbol{\theta}_k|\mathcal{D}_k^n) = \frac{p_k(\mathcal{D}_k^n|\boldsymbol{\theta}_k)p_k(\boldsymbol{\theta}_k|\mathcal{D}_k^{n-1})}{p_k(\mathcal{D}_k^n)}. \tag{3.6}$$

Using the variational Bayes framework:

$$\begin{aligned}
\boldsymbol{\phi}_k^* &= \arg\min_{\boldsymbol{\phi}_k} \int q_k^n(\boldsymbol{\theta}_k|\boldsymbol{\phi}_k) \log \frac{q_k^n(\boldsymbol{\theta}_k|\boldsymbol{\phi}_k)}{p_k(\boldsymbol{\theta}_k|\mathcal{D}_k^n)} d\boldsymbol{\theta}_k \\
&= \arg\min_{\boldsymbol{\phi}_k} \int q_k^n(\boldsymbol{\theta}_k|\boldsymbol{\phi}_k) \log \frac{q_k^n(\boldsymbol{\theta}_k|\boldsymbol{\phi}_k)}{p_k(\mathcal{D}_k^n|\boldsymbol{\theta}_k)p_k(\boldsymbol{\theta}_k)} d\boldsymbol{\theta}_k \\
&= \arg\min_{\boldsymbol{\phi}_k} \int q_k^n(\boldsymbol{\theta}_k|\boldsymbol{\phi}_k) \log \frac{q_k^n(\boldsymbol{\theta}_k|\boldsymbol{\phi}_k)}{p_k(\boldsymbol{\theta}_k)} d\boldsymbol{\theta}_k - \mathbb{E}_{\boldsymbol{\theta}_k \sim q_k^n(\boldsymbol{\theta}_k|\boldsymbol{\phi}_k)}[\log p(\mathcal{D}_k^n|\boldsymbol{\theta}_k)].
\end{aligned} \tag{3.7}$$

We assume that (according to BGD) $p_k(\boldsymbol{\theta}_k) = q_k^{n-1}(\boldsymbol{\theta}_k|\mathcal{D}_k^{n-1})$, that is prior is set to the approximate posterior of the previous task.

$$\boldsymbol{\phi}_k^* = \arg\min_{\boldsymbol{\phi}_k} \int q_k^n(\boldsymbol{\theta}_k|\boldsymbol{\phi}_k) \log \frac{q_k^n(\boldsymbol{\theta}_k|\boldsymbol{\phi}_k)}{q_k^{n-1}(\boldsymbol{\theta}_k|\mathcal{D}_k^{n-1})} d\boldsymbol{\theta}_k - \mathbb{E}_{\boldsymbol{\theta}_k \sim q_k^n(\boldsymbol{\theta}_k|\boldsymbol{\phi}_k)}[\log p(\mathcal{D}_k^n|\boldsymbol{\theta}_k)]. \tag{3.8}$$

In addition, (according to pfedbayes) at the end of the previous task, the posterior is set to the global posterior distribution, i.e., $q_k^{n-1}(\boldsymbol{\theta}_k|\mathcal{D}_k^{n-1}) = w^{n-1}(\boldsymbol{\theta}_k|\mathcal{D}_k^{n-1})$. Hence, the above equation can be rewritten as

$$\boldsymbol{\phi}_k^* = \arg\min_{\boldsymbol{\phi}_k} \mathbb{E}_{\boldsymbol{\theta}_k \sim q_k^n(\boldsymbol{\theta}_k|\boldsymbol{\phi}_k)} \left[\log q_k^n(\boldsymbol{\theta}_k|\boldsymbol{\phi}_k) - \log w^{n-1}(\boldsymbol{\theta}_k|\mathcal{D}_k^{n-1}) - \log p(\mathcal{D}_k^n|\boldsymbol{\theta}_k)\right]. \tag{3.9}$$

Essentially, we need to optimize for the parameters of $q_k^n(\boldsymbol{\theta}_k|\boldsymbol{\phi}_k)$ at each client, under the assumption that $\log p(\mathcal{D}_k^n|\boldsymbol{\theta}_k)$ is available at each client as the log-likelihood, and $p_k(\boldsymbol{\theta}_k|\mathcal{D}_k^{n-1})$ is available from the previous task. We assume that $w_k^{n-1}(\boldsymbol{\theta}_k|\mathcal{D}_k^{n-1})$ is Gaussian whose mean and variance at round t can be computed as

$$\mu_w^{n-1}(m) = \frac{1}{K}\sum_{k=1}^{K} \mu_k^{n-1}(m), \tag{3.10}$$

$$(\sigma_w^{n-1}(m))^2 = \frac{1}{K}\sum_{k=1}^{K}\left[(\sigma_k^{n-1}(m))^2 + (\mu_k^{n-1}(m))^2 - (\mu_w^{n-1}(m))^2\right]. \tag{3.11}$$

These updates of the parameters happen at the server. At the k-th client, the parameters are continuously updated based on the previous task. The update equations are given as

$$\mu_k(m) = \mu_k(m) - \eta\sigma_k^2(m)\mathbb{E}_\epsilon\left[\frac{\partial L_k^n(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_k}\right], \tag{3.12}$$

$$\sigma_k(m) = \sigma_k(m)\sqrt{1 + \left(\frac{1}{2}\sigma_k(m)\mathbb{E}_\epsilon\left[\frac{\partial L_k^n(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_k}\epsilon_k(m)\right]\right)^2} - \frac{1}{2}\sigma_k^2(m)\mathbb{E}_\epsilon\left[\frac{\partial L_k^n(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_k}\epsilon_k(m)\right], \tag{3.13}$$

where $\epsilon_k(m) \sim \mathcal{N}(0,1)$ and $L_k^n(\boldsymbol{\theta}) = -\log p(\mathcal{D}_k^n|\boldsymbol{\theta}_k)$.

The expectations are estimated using Monte Carlo method, with $\boldsymbol{\theta}_k^{(t)} = \mu_k(m) + \epsilon_k^{(t)}(m)\sigma_k(m)$:

$$\mathbb{E}_\epsilon\left[\frac{\partial L_k^n(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_k}\right] \approx \frac{1}{K}\sum_{t=1}^{K}\frac{\partial L_k^n(\boldsymbol{\theta}^{(t)})}{\partial \boldsymbol{\theta}_k}, \tag{3.14}$$

$$\mathbb{E}_\epsilon\left[\frac{\partial L_k(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_k}\epsilon_k(m)\right] \approx \frac{1}{K}\sum_{t=1}^{K}\frac{\partial L_k^n(\boldsymbol{\theta}^{(t)})}{\partial \boldsymbol{\theta}_k}\epsilon_k^{(t)}(m). \tag{3.15}$$

3.3.1 Algorithm

The following steps summarize the proposed scheme:

Algorithm 1 Proposed CFL-BGD Scheme

1: **for** each round of communication $t \in [\mathcal{C}_t]$ **do**

2: Initialize client models with the Server model

3: **for** each client k **do**

4: Compute local gradients and update local model parameters on every iteration(mini-batch):

$$\mu_{k,t}(m) = \mu_{k,t}(m) - \eta\sigma_{k,t}^2(m)\mathbb{E}_\epsilon\left[\frac{\partial L_k^n(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_k}\right], \tag{3.16}$$

$$\begin{aligned}\sigma_{k,t}(m) = \sigma_{k,t}(m)\sqrt{1 + \left(\frac{1}{2}\sigma_{k,t}(m)\mathbb{E}_\epsilon\left[\frac{\partial L_k^n(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_k}\epsilon_{k,t}(m)\right]\right)^2} \\ - \frac{1}{2}\sigma_{k,t}^2(m)\mathbb{E}_\epsilon\left[\frac{\partial L_k^n(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_k}\epsilon_k(m)\right]\end{aligned} \tag{3.17}$$

5: Transmit updated parameters to the server.

6: **end for**

7: Server aggregates the received parameters:

$$\mu_{w,t+1}^{n-1}(m) = \frac{1}{K}\sum_{k=1}^{K}\mu_{k,t}^{n-1}(m), \tag{3.18}$$

$$(\sigma_{w,t+1}^{n-1}(m))^2 = \frac{1}{K}\sum_{k=1}^{K}\left[(\sigma_{k,t}^{n-1}(m))^2 + (\mu_{k,t}^{n-1}(m))^2 - (\mu_{w,t}^{n-1}(m))^2\right], \tag{3.19}$$

8: Server transmits the global parameters to the clients.

9: **end for**

CHAPTER 4

EXPERIMENTAL SETUP & RESULTS

In this section, we demonstrate the performance of the proposed FCL-BGD framework on standard datasets used in continual and federated learning. The novelty also lies in the data partitioning, where data is partitioned in a task-boundary agnostic manner, with tasks transitioning gradually over time. This task-based dataset is later distributed in an IID and non-IID manner to the clients, hence simulating diverse task and data distributions across federated nodes.

4.1 Dataset and Partitioning

We have used two variants of MNIST dataset namely

- **Permuted-MNIST**: This variant of MNIST involves randomly permuting the pixels of each image. Every task uses a different permutation of the pixels from the previous task. For example, if we consider N tasks, we have $N-1$ permutations and 1 task as the original MNIST.
- **Split-MNIST**: This set of tasks is constructed by pairing digits from the MNIST dataset. For instance, the first task involves classifying digits 0 and 1, the second task involves classifying digits 2 and 3, and so on. This approach results in a total of five tasks.

4.1.1 IID and Non-IID Partitioning

In our experiments, we employed the Dirichlet partitioning technique to distribute the data among the participating clients. This method allows us to regulate the data heterogeneity in Federated Learning using the Dirichlet parameter α. When $\alpha \to 0$, the data distribution becomes highly heterogeneous (Non-IID), reflecting the real-world scenario where clients have distinct data distributions. Conversely, when $\alpha \to \infty$, the data distribution among clients becomes homogeneous (IID), simulating a scenario where all clients have similar data.

For our method `CFL_BGD`, we used three values of α to generate Non-IID and IID data distributions: $\alpha = 0.1$, $\alpha = 0.01$ and $\alpha = 10^5$. The value $\alpha = 0.1$ was chosen to create a non-IID setting, introducing moderate heterogeneity among the clients' data distributions, and value $\alpha = 0.01$ was chosen to create a non-IID setting it mimics real-world data heterogeneity, providing a challenging scenario for evaluating our method. On the other hand, $\alpha = 10^5$ was used to create an IID setting, ensuring that all clients have nearly identical data distributions. This helps in comparing the performance of our method under both homogeneous and heterogeneous data conditions.

4.1.2 Continuous Task- Agnostic Scenario

Tasks arrive continuously over the training phase, and transitions between tasks occur gradually over time rather than suddenly. As a result, task boundaries are not known during training. The output heads are shared among all tasks. In continuous task-agnostic learning, the concept of an epoch does not exist since we cannot define 'passing over the whole dataset'. Instead, we define "iterations per virtual epoch" to indicate how many iterations constitute a single epoch. The figure below illustrates the proposed transition of tasks that occurs over the training phase Zeno *et al.* (2021).

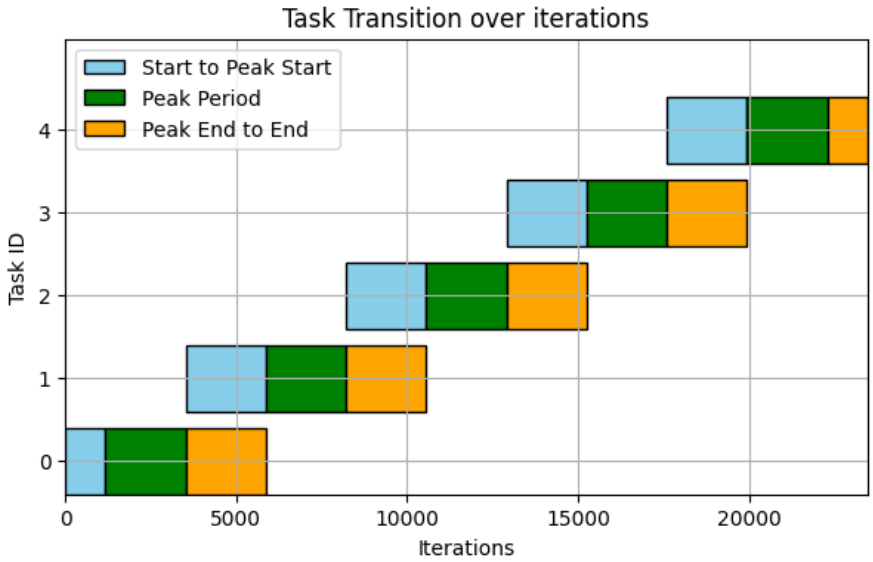


Figure 4.1: Evolving tasks over the training phase. This example considers 5 tasks, with T iterations (T = iterations_per_virtual_epoch $\times$ max_epochs).

This scenario is achieved by dynamically generating task probabilities at each iteration. For n tasks, a task probability vector of dimension $T \times n$ is calculated, where

T is the total number of iterations and n represents the number of tasks. Thus, at each iteration, a probability vector of dimension $1 \times n$ is obtained, containing the probabilities of selecting each task for that specific iteration. For instance, the probability vector for the i-th iteration can be represented as $[p_{i1}, p_{i2}, ..., p_{in}]$, where p_{ij} denotes the probability of selecting task j at iteration i. This approach introduces tasks sequentially with overlapping boundaries. Figure 4.1 illustrates this concept by dividing the process into three regions: (i) start to peak start, where the task's probability gradually increases; (ii) peak period, where the task's probability reaches its maximum; and (iii) peak end to end, where the task's probability decreases to zero. Data is distributed among clients using Dirichlet distribution, and then for every client, their corresponding data is made available for them in an agnostic manner as described above.

4.2 Implementation Details and Baseline

The implementation of **CFL_BGD** is carried out using the PyTorch framework and utilizes an NVIDIA Quadro RTX 6000 GPU. During training, K clients are simulated, enabling efficient communication between clients and the server through the PyTorch back-end. In our experiments, we used $K = 5$ number of clients, number of tasks $T = 5$ and the number of local epochs to $E = 30$, and the number of communication rounds to $C = 5$ over the training phase, unless stated otherwise. For all experiments, the batch size is $B = 128$. The model used for training is a fully connected linear network with two hidden layers each of width 200, trained with `mean_eta` = 1 for `CFL_BGD` and `CFL_BGD` with a learning rate of $\eta = 0.01$. We have a total of T iterations in the entire training phase. From these iterations, we compute a set of aggregation points, denoted as $\mathcal{C}_t$. Each element c in $\mathcal{C}_t$ represents the iteration at which a task is fully processed divided by the number of aggregations (set to 1). The total number of aggregations is equal to the number of tasks times number of aggregations. The server model is aggregated every c iterations. Clients continuously update their models using data up to the c-th iteration before transmitting updates to the server. The server combines these client updates to create a new server model.

Due to the inapplicability of previous algorithms in task agnostic scenario, we chose `FedAvg` aggregation with `SGD` as the optimizer in a continual task-agnostic scenario

as our baseline for comparison, referring to it as `CFL_SGD`.

4.3 Metrics

For T as the total number of tasks, we define Accuracy and Forgetting as follows :

Average Accuracy (Acc): This metric measures the average performance of the global model across all tasks at the end of the final task in the continuous federated learning (CFL) process. It provides an overall accuracy score for how well the model has performed on all tasks. It is mathematically computed as follows:

$$\text{Acc} = \frac{1}{T}\sum_{i=1}^{T} A_{T,i} \tag{4.1}$$

where $A_{T,i}$ is the accuracy of the global model on task i at the end of the final task T.

Forgetting: This metric measures the extent to which the model forgets previously learned tasks as it learns new ones. A lower value of forgetting indicates that the model retains its performance on old tasks even after being trained on new tasks.
It is mathematically computed as follows:

$$\text{Forget} = \frac{1}{T-1}\sum_{i=1}^{T-1}(A_{i,i} - A_{T,i}) \tag{4.2}$$

where $A_{i,i}$ is the accuracy on ith task right after training on task i, and $A_{T,i}$ is the accuracy on task i at the end of the final task T.

4.4 Results and Discussion

Table 4.1: Performance metrics on P-MNIST dataset for IID and Non-IID ($\alpha = 0.1$) and Non-IID ($\alpha = 0.01$) settings.

	P-MNIST					
	IID		Non-IID		Non-IID	
	Acc	Forget	Acc	Forget	Acc	Forget
CFL_SGD	51.91	56.62	38.35	40.65	22.41	11.78
CFL_BGD	**85.83**	**13.55**	**66.24**	**18.46**	**36.72**	**9.12**

Table 4.2: Performance metrics on S-MNIST dataset for IID and Non-IID ($\alpha = 0.1$) and Non-IID ($\alpha = 0.01$) settings.

	Split-MNIST					
	IID		Non-IID		Non-IID	
	Acc	Forget	Acc	Forget	Acc	Forget
CFL_SGD	60.12	48.38	60.9	46.66	60.95	46.66
CFL_BGD	**67.14**	**40.01**	**69.27**	**37.47**	**68.91**	**37.71**

Table 4.1 and Table 4.2 present the average accuracy across tasks (Acc) and average forgetting of all tasks (Forget) for the Permuted-MNIST and Split-MNIST datasets, respectively, under different settings: IID ($\alpha = 10^5$), mild Non-IID ($\alpha = 0.1$), and highly Non-IID ($\alpha = 0.01$). These metrics are used to compare the proposed method, CFL_BGD, with the baseline CFL_SGD.

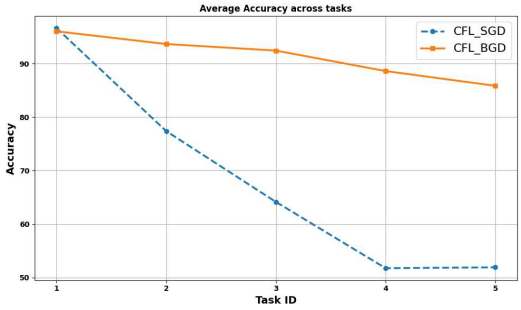
- The results indicate that CFL_BGD (proposed) consistently outperforms CFL_SGD (baseline) in both accuracy and forgetting across all data distribution settings (IID and Non-IID).
- In the permuted MNIST case, we can see the proposed method is performing significantly higher accuracy lower forgetting than the baseline, which supports our analysis that the method significantly alleviates catastrophic forgetting.
- In the case of Split MNIST, we could see that consistently the proposed method outperforms the baseline, but the difference margin is less compared to permuted MNIST.
- Even in highly Non-IID cases, CFL_BGD demonstrates better performance compared to baseline.

To analyse the performance of CFLBGD, and to compare with baseline. We have cinsidered two types of plots:

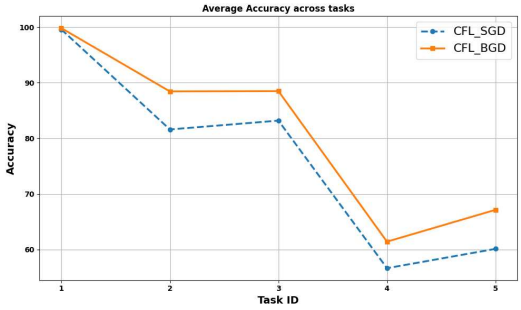
- **Average Global Accuracy plot:** This plot visualizes the mean accuracy across all tasks up to a specific task ID. The x-axis represents the task ID, and the y-axis represents the average accuracy.
- **Task-Wise Accuracies Over Rounds plot:** This plot illustrates the accuracy of a particular task as it progresses through multiple rounds. The x-axis represents the round number, and the y-axis represents the accuracy.

4.4.1 IID ($\alpha = 10^5$)

Under IID conditions, the proposed CFL_BGD method significantly outperforms the baseline CFL_SGD in both average and task-wise accuracies. As depicted in Figures

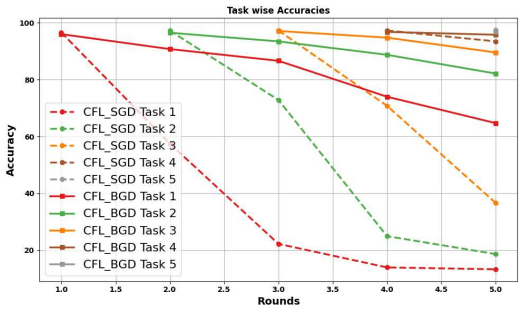


(a) P-MNIST average accuracy (iid)

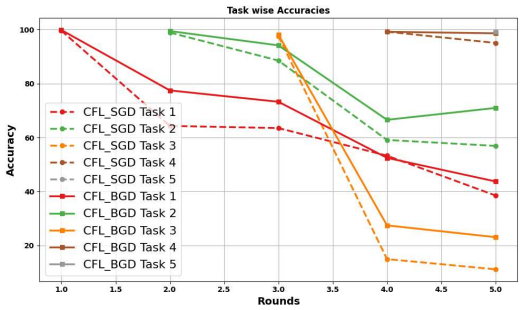


(b) Split-MNIST Average Accuracy (iid)

Figure 4.2: Average accuracy across tasks



(a) P-MNIST task-wise accuracy (iid)

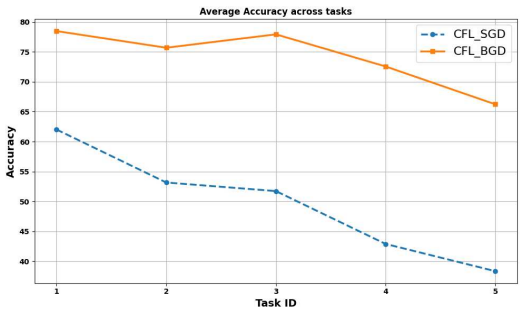


(b) Split-MNIST task-wise accuracy (iid)

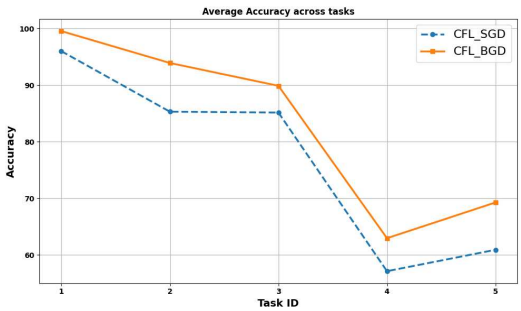
Figure 4.3: Task wise accuracy across Rounds

4.2 and 4.3, CFL_BGD demonstrates superior performance by maintaining a more stable accuracy level across tasks and rounds, while CFL_SGD exhibits a rapid decline. Although both methods encounter challenges on the Split-MNIST dataset, CFL_BGD's resilience is more evident, indicating its effectiveness in preserving task-specific knowledge.

4.4.2 Non-IID ($\alpha = 0.1$)

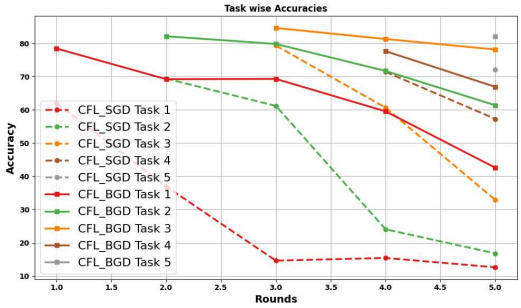


(a) P-MNIST average accuracy (non-iid ($\alpha = 0.1$)

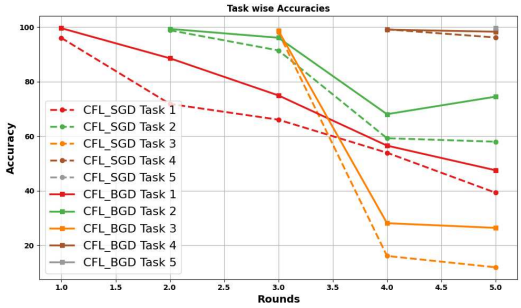


(b) Split-MNIST average accuracy (non-iid ($\alpha = 0.1$)

Figure 4.4: Average accuracy across tasks



(a) P-MNIST task wise accuracy (non-iid ($\alpha = 0.1$)

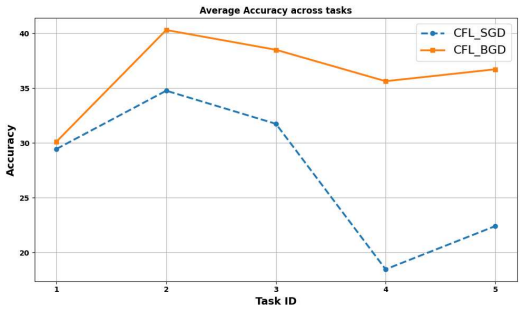


(b) Split-MNIST task wise accuracy (non-iid ($\alpha = 0.1$)

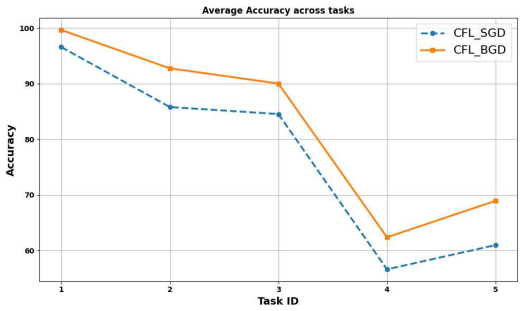
Figure 4.5: Task wise accuracy across Rounds

The introduction of mild Non-IID conditions ($\alpha = 0.1$) intensifies the catastrophic forgetting problem for both CFL_BGD and CFL_SGD. However, Figures 4.4 and 4.5 reveal that CFL_BGD continues to outperform CFL_SGD by maintaining a more gradual decline in accuracy. While the Split-MNIST dataset remains challenging under these conditions, CFL_BGD demonstrates better adaptability to data heterogeneity.

4.4.3 Non-IID ($\alpha = 0.01$)

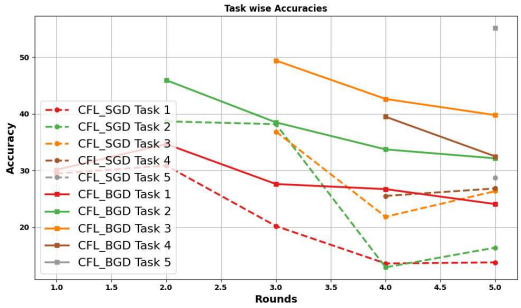


(a) P-MNIST average accuracy (non-iid ($\alpha = 0.01$)

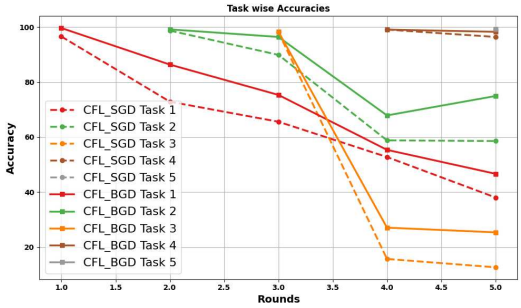


(b) Split-MNIST average accuracy (non-iid ($\alpha = 0.01$)

Figure 4.6: Average accuracy across tasks



(a) P-MNIST task wise accuracy (non-iid ($\alpha = 0.01$)



(b) Split-MNIST task wise accuracy (non-iid ($\alpha = 0.01$

Figure 4.7: Task wise accuracy across Rounds

As data distribution shifts to highly Non-IID conditions ($\alpha = 0.01$), the performance of both CFL_BGD and CFL_SGD deteriorates significantly. Figures 4.6 and 4.7 illustrate a pronounced decline in task-wise accuracy for both methods. Despite the challenging environment, CFL_BGD maintains a performance advantage over CFL_SGD, although the gap narrows compared to previous settings. This suggests that while CFL_BGD is more robust to data heterogeneity, extreme Non-IID conditions pose significant challenges for both algorithms.

CHAPTER 5

CONCLUSION

In this work we have explored the challenges and advancements in Continual Federated Learning (CFL). By developing and analyzing a novel strategy using Online Variational Bayes, we have addressed the critical issue of catastrophic forgetting in federated learning systems. This approach provides a robust solution for updating models continuously without the need for retraining from scratch, thus enhancing the efficiency, privacy, and scalability of these systems. The results of our experiments, which include comparisons of Average accuracy and task-wise accuracies and average forgetting across various levels of data heterogeneity, demonstrate the effectiveness of our method. This work not only contributes to the theoretical understanding of CFL but also offers practical insights for its implementation in real-world applications. These findings provide a strong foundation for future research and development, paving the way for more advanced and resilient federated learning models capable of handling the dynamic nature of real-world data.

REFERENCES

1. **Aich, A.** (2021). Elastic Weight Consolidation (EWC): Nuts and Bolts. URL http://arxiv.org/abs/2105.04093. ArXiv:2105.04093 [cs, stat].

2. **Bakman, Y. F.**, **D. N. Yaldiz**, **Y. H. Ezzeldin**, and **S. Avestimehr** (2023). Federated Orthogonal Training: Mitigating Global Catastrophic Forgetting in Continual Federated Learning. URL http://arxiv.org/abs/2309.01289. ArXiv:2309.01289 [cs].

3. **Blundell, C.**, **J. Cornebise**, **K. Kavukcuoglu**, and **D. Wierstra** (2015). Weight Uncertainty in Neural Networks. URL http://arxiv.org/abs/1505.05424. ArXiv:1505.05424 [cs, stat].

4. **Chellapandi, V. P.**, **L. Yuan**, **C. G. Brinton**, **S. H. Zak**, and **Z. Wang** (2023). Federated Learning for Connected and Automated Vehicles: A Survey of Existing Approaches and Challenges. URL http://arxiv.org/abs/2308.10407. ArXiv:2308.10407 [cs].

5. **Dhar, P.**, **R. V. Singh**, **K.-C. Peng**, **Z. Wu**, and **R. Chellappa** (2019). Learning without Memorizing. URL http://arxiv.org/abs/1811.08051. ArXiv:1811.08051 [cs].

6. **Dong, J.**, **H. Li**, **Y. Cong**, **G. Sun**, **Y. Zhang**, and **L. Van Gool** (2023). No One Left Behind: Real-World Federated Class-Incremental Learning. URL http://arxiv.org/abs/2302.00903. ArXiv:2302.00903 [cs].

7. **Dong, J.**, **L. Wang**, **Z. Fang**, **G. Sun**, **S. Xu**, **X. Wang**, and **Q. Zhu** (2022). Federated Class-Incremental Learning. URL http://arxiv.org/abs/2203.11473. ArXiv:2203.11473 [cs].

8. **Dupuy, C.**, **J. Majmudar**, **J. Wang**, **T. Roosta**, **R. Gupta**, **C. Chung**, **J. Ding**, and **S. Avestimehr** (2023*a*). Quantifying catastrophic forgetting in continual federated learning. URL https://www.amazon.science/publications/quantifying-catastrophic-forgetting-in-continual-federated-learning.

9. **Dupuy, C.**, **J. Majmudar**, **J. Wang**, **T. G. Roosta**, **R. Gupta**, **C. Chung**, **J. Ding**, and **S. Avestimehr**, Quantifying catastrophic forgetting in continual federated learning. *In ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2023*b*. ISSN 2379-190X.

10. **Graves, A.**, Practical Variational Inference for Neural Networks. *In Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011. URL https://papers.nips.cc/paper_files/paper/2011/hash/7eb3c8be3d411e8ebfab08eba5f49632-Abstract.html.

11. **Hegiste, V.**, **T. Legler**, and **M. Ruskowski** (2023). Federated Object Detection for Quality Inspection in Shared Production. URL http://arxiv.org/abs/2306.17645. ArXiv:2306.17645 [cs].

12. **Joshi, M.**, **A. Pal**, and **M. Sankarasubbu** (2022). Federated Learning for Healthcare Domain - Pipeline, Applications and Challenges. *ACM Transactions on Computing for Healthcare*, **3**(4), 1–36. ISSN 2691-1957, 2637-8051. URL http://arxiv.org/abs/2211.07893. ArXiv:2211.07893 [cs].

13. **Kim, C. D.**, **J. Jeong**, and **G. Kim** (2020). Imbalanced Continual Learning with Partitioning Reservoir Sampling. URL http://arxiv.org/abs/2009.03632. ArXiv:2009.03632 [cs, stat].

14. **Kirkpatrick, J.**, **R. Pascanu**, **N. Rabinowitz**, **J. Veness**, **G. Desjardins**, **A. A. Rusu**, **K. Milan**, **J. Quan**, **T. Ramalho**, **A. Grabska-Barwinska**, **D. Hassabis**, **C. Clopath**, **D. Kumaran**, and **R. Hadsell** (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, **114**(13), 3521–3526. ISSN 0027-8424, 1091-6490. URL http://arxiv.org/abs/1612.00796. ArXiv:1612.00796 [cs, stat].

15. **Kochurov, M.**, **T. Garipov**, **D. Podoprikhin**, **D. Molchanov**, **A. Ashukha**, and **D. Vetrov** (2018). Bayesian Incremental Learning for Deep Neural Networks. URL http://arxiv.org/abs/1802.07329. ArXiv:1802.07329 [cs, stat].

16. **Kumar, A.**, **S. Chatterjee**, and **P. Rai** (2020). Bayesian Structure Adaptation for Continual Learning. URL http://arxiv.org/abs/1912.03624. ArXiv:1912.03624 [cs, stat].

17. **Li, S.**, **T. su**, **X. Zhang**, and **Z. Wang**, *Continual Learning with Knowledge Distillation: A Survey*. 2024.

18. **Li, T.**, **A. K. Sahu**, **M. Zaheer**, **M. Sanjabi**, **A. Talwalkar**, and **V. Smith** (2020). Federated Optimization in Heterogeneous Networks. URL http://arxiv.org/abs/1812.06127. ArXiv:1812.06127 [cs, stat].

19. **Li, X.**, **M. Jiang**, **X. Zhang**, **M. Kamp**, and **Q. Dou** (2021). FedBN: Federated Learning on Non-IID Features via Local Batch Normalization. URL http://arxiv.org/abs/2102.07623. ArXiv:2102.07623 [cs].

20. **Li, Z.** and **D. Hoiem** (2017). Learning without Forgetting. URL http://arxiv.org/abs/1606.09282. ArXiv:1606.09282 [cs, stat].

21. **Lopez-Paz, D.** and **M. Ranzato** (2022). Gradient Episodic Memory for Continual Learning. URL http://arxiv.org/abs/1706.08840. ArXiv:1706.08840 [cs].

22. **Ma, Y.**, **Z. Xie**, **J. Wang**, **K. Chen**, and **L. Shou**, Continual Federated Learning Based on Knowledge Distillation. volume 3. 2022. URL https://www.ijcai.org/proceedings/2022/303. ISSN: 1045-0823.

23. **Masana, M.**, **X. Liu**, **B. Twardowski**, **M. Menta**, **A. D. Bagdanov**, and **J. van de Weijer** (2020). Class-incremental learning: survey and performance evaluation on image classification. URL https://arxiv.org/abs/2010.15277v3.

24. **McMahan, H. B.**, **E. Moore**, **D. Ramage**, **S. Hampson**, and **B. A. y. Arcas** (2023). Communication-Efficient Learning of Deep Networks from Decentralized Data. ArXiv:1602.05629 [cs].

25. **Mittal, S.**, **S. Galesso**, and **T. Brox** (2021). Essentials for Class Incremental Learning. URL http://arxiv.org/abs/2102.09517. ArXiv:2102.09517 [cs].

26. **Nguyen, C. V.**, **Y. Li**, **T. D. Bui**, and **R. E. Turner** (2018). Variational Continual Learning. URL http://arxiv.org/abs/1710.10628. ArXiv:1710.10628 [cs, stat].

27. **Rosasco, L.** and **S. Villa** (2015). Learning with incremental iterative regularization. URL http://arxiv.org/abs/1405.0042. ArXiv:1405.0042 [cs, math, stat].

28. **Shin, H.**, **J. K. Lee**, **J. Kim**, and **J. Kim** (2017). Continual Learning with Deep Generative Replay. URL http://arxiv.org/abs/1705.08690. ArXiv:1705.08690 [cs].

29. **Xu, Z.**, **J. Cheng**, **L. Cheng**, **X. Xu**, and **M. Bilal** (2023). MSEs Credit Risk Assessment Model Based on Federated Learning and Feature Selection. *Computers, Materials & Continua*, **75**(3), 5573–5595. ISSN 1546-2218, 1546-2226. URL https://www.techscience.com/cmc/v75n3/52593.

30. **Yan, S.**, **J. Xie**, and **X. He** (2021). DER: Dynamically Expandable Representation for Class Incremental Learning. URL http://arxiv.org/abs/2103.16788. ArXiv:2103.16788 [cs].

31. **Yoon, J.**, **W. Jeong**, **G. Lee**, **E. Yang**, and **S. J. Hwang** (2021). Federated Continual Learning with Weighted Inter-client Transfer. URL http://arxiv.org/abs/2003.03196. ArXiv:2003.03196 [cs, stat].

32. **Zeno, C.**, **I. Golan**, **E. Hoffer**, and **D. Soudry** (2021). Task Agnostic Continual Learning Using Online Variational Bayes with Fixed-Point Updates. *Neural Computation*, 1–39. ISSN 0899-7667, 1530-888X. URL http://arxiv.org/abs/2010.00373. ArXiv:2010.00373 [cs, stat].

33. **Zhang, T.**, **L. Gao**, **C. He**, **M. Zhang**, **B. Krishnamachari**, and **S. Avestimehr** (2022*a*). Federated Learning for Internet of Things: Applications, Challenges, and Opportunities. URL http://arxiv.org/abs/2111.07494. ArXiv:2111.07494 [cs].

34. **Zhang, X.**, **Y. Li**, **W. Li**, **K. Guo**, and **Y. Shao** (2022*b*). Personalized Federated Learning via Variational Bayesian Inference. URL http://arxiv.org/abs/2206.07977. ArXiv:2206.07977 [cs] PersonalizedFederatedLearning2022b.

35. **Zhao, X.**, **H. Wang**, **W. Huang**, and **W. Lin** (2024). A Statistical Theory of Regularization-Based Continual Learning. URL http://arxiv.org/abs/2406.06213. ArXiv:2406.06213 [cs, stat].