

T.R.
GEBZE TECHNICAL UNIVERSITY
GRADUATE SCHOOL

**EFFICIENCY IMPROVEMENT VIA NEW METHODS FOR DATA
VALUATION, PRIVACY AND COMMUNICATION IN
FEDERATED LEARNING**

EMRE ARDIÇ

A THESIS OF DOCTORATE
DEPARTMENT OF COMPUTER ENGINEERING

ADVISOR: ASSOC. PROF. YAKUP GENÇ

MAY 2025

T.R.
GEBZE TECHNICAL UNIVERSITY
GRADUATE SCHOOL

**EFFICIENCY IMPROVEMENT VIA NEW METHODS
FOR DATA VALUATION, PRIVACY AND
COMMUNICATION IN FEDERATED LEARNING**

EMRE ARDIÇ

A THESIS OF DOCTORATE
DEPARTMENT OF COMPUTER ENGINEERING

ADVISOR: ASSOC. PROF. YAKUP GENÇ

MAY 2025

T.C.
GEBZE TEKNİK ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

YENİ VERİ DEĞERLENDİRME, GİZLİLİK VE
HABERLEŞME YÖNTEMLERİYLE FEDERE
ÖĞRENMEDE VERİMLİLİK ARTIRIMI

EMRE ARDIÇ

DOKTORA TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

DANIŞMAN: DOÇ. DR. YAKUP GENÇ

MAYIS 2025

DOCTORATE JURY APPROVAL FORM

A thesis submitted by Emre Ardıc, defended on 13/05/2025 before the jury formed with the 05/05/2025 date and 2025/25 numbered decision of the GTU Graduate Administration Board, has been accepted as a DOCTOR of PHILOSOPHY thesis in the Department of Computer Engineering.

JURY

MEMBER

(THESIS ADVISOR) : Assoc. Prof. Dr. Yakup GENÇ

MEMBER

: Assoc. Prof. Dr. Mehmet GÖKTÜRK

MEMBER

: Prof. Dr. Fatih Erdoğan SEVİLGİN

MEMBER

: Prof. Dr. Haluk Rahmi TOPÇUOĞLU

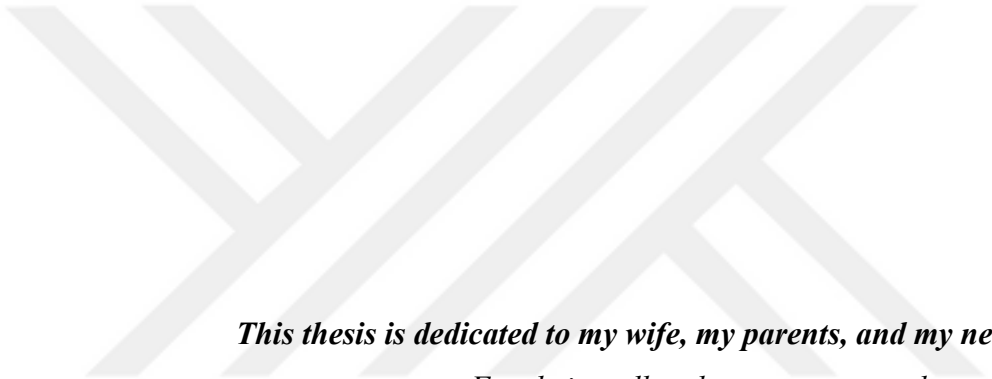
MEMBER

: Assoc. Prof. Dr. Hâbil KALKAN

APPROVAL

Gebze Technical University Graduate Administration Board
...../...../..... date and/..... numbered decision.

SIGNATURE/SEAL



*This thesis is dedicated to my wife, my parents, and my newborn son.
For their endless love, support, and encouragement.*

ABSTRACT

Federated Learning (FL) is a distributed machine learning method where multiple devices collaboratively train a model under the management of a central server without sharing underlying data. However, its performance is often hindered by redundant, malicious, or abnormal samples, leading to accuracy degradation. Another challenge is the communication bottleneck caused by variations in connection speed and bandwidth across devices. Thus, it is critical to reduce the size of transmitted data during training. Additionally, there is a potential risk of exposing sensitive information through the model or gradient analysis during training.

This thesis combines Differential Privacy (DP) with adaptive quantization to address both privacy and communication efficiency. Specifically, Laplacian-based DP is used to enhance privacy, which is relatively underexplored in FL and offers tighter privacy guarantees than Gaussian-based DP. Two novel adaptive quantization methods are introduced: a global bit-length scheduler employing round-based cosine annealing and a client-based scheduler that dynamically adjusts quantization precision according to dataset entropy, thereby prioritizing clients with more informative datasets. Experimental evaluations conducted on standard datasets, including CIFAR10, MNIST, and specialized medical imaging datasets, show the effectiveness of the proposed methods. Results show significant reductions in communication overhead, up to 52.64% for MNIST, 45.06% for CIFAR10, and between 31% and 37% for medical imaging datasets, while maintaining competitive model accuracy and robust privacy protection.

This thesis further presents novel sample selection methods to identify and filter redundant, malicious, or abnormal local samples that reduce model performance during training. A Multi-Task Autoencoder (MTAE) architecture, leveraging loss and feature analysis, is used to estimate individual sample contributions. Unsupervised outlier detection methods managed by a central server, such as One-Class Support Vector Machine (OCSVM), Isolation Forest (IF) and Adaptive Thresholding (AT), are proposed to filter noisy samples on clients. A multi-class deep Support Vector Data Description (SVDD) loss function is also proposed to improve feature-based sample selection in FL.

Moreover, the effectiveness of Shapley Value (SV) and Leave-One-Out (LOO) methods under noise is empirically evaluated. Experiments on the MNIST dataset with 40% closed-set noise show notable accuracy gains for both methods. Gradient Shapley and Truncated Monte Carlo Shapley achieve accuracy gains of 8.51% and 7.11%, respectively, although with significant computational overhead. In contrast, LOO-based methods, particularly LOO-Sum, yield comparable accuracy gains with substantially lower computational demands. To evaluate MTAE-based sample selection, extensive experiments are performed on the non-IID distributed CIFAR10 and MNIST datasets under various client counts and noise levels of up to 40%. Loss-based sample selection achieved notable accuracy improvements, including gains up to 7.02% on CIFAR10 using OCSVM and 1.83% on MNIST using AT. Moreover, the federated SVDD-based approach further improved feature-based selection, resulting in accuracy gains of up to 0.99% on CIFAR10 with OCSVM. Collectively, these results underline the potential of the developed methods to significantly enhance FL model accuracy, robustness, and resilience under challenging data distributions and privacy constraints.

Keywords: Differential Privacy, Adaptive Quantization, Sample Selection, Federated Learning

ÖZET

Federe Öğrenme (FL), birden fazla cihazın merkezi bir sunucunun yönetiminde, verilerini paylaşmadan ortak bir model eğittiği dağıtık bir makine öğrenimi yöntemidir. Ancak FL performansı, çoğu zaman gereksiz, zararlı veya anormal örneklerin varlığı nedeniyle düşmekte ve model doğruluğunun azalmasına yol açmaktadır. Bir diğer önemli zorluk ise, cihazlar arasındaki bağlantı hızları ve bant genişliklerindeki farklılıkların neden olduğu iletişim darboğazıdır. Bu nedenle, eğitim sırasında iletilen veri miktarının azaltılması büyük önem taşımaktadır. Ayrıca, model ya da gradyan analizleri yoluyla hassas bilgilerin açığa çıkma riski de mevcuttur.

Bu tezde, gizlilik koruması ve iletişim verimliliği sağlamak amacıyla Diferansiyel Gizlilik (DP) ve adaptif kuantizasyon yöntemleri önerilmektedir. Özellikle, FL literatüründe az çalışılmış Laplace tabanlı DP yöntemi tercih edilmiştir. Bu yöntem, Gaussian tabanlı yaklaşımlara göre daha sıkı gizlilik garantileri sunmaktadır. Ayrıca, iki yeni adaptif kuantizasyon yöntemi önerilmektedir. Bunlardan biri, eğitim turlarına göre kosinüs azalımı kullanan küresel bir bit uzunluğu zamanlayıcısı; diğeri ise veri kümesi entropisine bağlı olarak bit hassasiyetini dinamik biçimde ayarlayan istemci tabanlı zamanlayıcıdır. CIFAR10, MNIST ve bazı özel tıbbi görüntüleme veri kümeleri üzerinde yapılan deneyler, önerilen yöntemlerin etkinliğini ortaya koymaktadır. Sonuçlar MNIST için %52,64'e, CIFAR10 için %45,06'ya ve tıbbi veri kümeleri için %31 ila %37 arasında iletişim yükünde azalma sağlandığını, model doğruluğunun korunduğunu ve güçlü gizlilik koruması elde edildiğini göstermektedir.

Ayrıca, eğitim sürecinde model performansını düşüren gereksiz, zararlı veya anormal yerel verileri tespit edip filtrelemek üzere yeni örnek seçme yöntemleri geliştirilmiştir. Örnek değerini hem kayıp hem de özellik analizlerine göre tahmin eden çok görevli bir otomatik kodlayıcı (MTAE) mimarisi önerilmektedir. Düşük kaliteli ya da zararlı örneklerin denetimsiz olarak tespiti için, merkezi olarak yönetilen fakat istemcilerde yerel olarak çalışan One-Class Support Vector Machine (OCSVM), Isolation Forest (IF) ve Adaptif Eşikleme (AT) yöntemleri önerilmiştir. Ayrıca, FL ortamlarında özellik tabanlı örnek seçimini geliştirmek amacıyla çok sınıflı Deep Support Vector Data Description (SVDD) kayıp fonksiyonu önerilmektedir.

Ayrıca, hesaplama maliyeti yüksek olsa da Shapley Value (SV) ve Leave-One-Out (LOO) tabanlı örnek seçim yöntemleri geliştirilmiştir. MNIST üzerinde %40'lık kapalı-set gürültüsü ile yapılan deneyler, her iki yöntemin de doğrulukta anlamlı artış sağladığını göstermektedir. Gradient Shapley ve Truncated Monte Carlo Shapley ise sırasıyla %8,51 ve %7,11 doğruluk artışı sağlamıştır. Buna karşılık, özellikle LOO-Sum yöntemi olmak üzere LOO tabanlı yaklaşımlar, SV'ye göre çok daha düşük hesaplama maliyetiyle benzer doğruluk artışları sağlamıştır. MTAE tabanlı örnek seçimini değerlendirmek için non-IID dağıtılmış CIFAR10 ve MNIST veri kümelerinde %40'lık gürültü seviyesiyle farklı istemci sayıları için kapsamlı deneyler gerçekleştirilmiştir. Kayıp tabanlı örnek seçimi, CIFAR10 için OCSVM ile %7,02, MNIST için AT ile %1,83 doğruluk artışı sağlamıştır. Ayrıca, SVDD tabanlı yaklaşım, özellikle CIFAR10 üzerinde OCSVM ile %0,99 ek doğruluk artışı elde ederek, özellik tabanlı seçimi daha da iyileştirmiştir. Bu sonuçlar, geliştirilen yöntemlerin zorlu veri dağılımları ve gizlilik gereksinimleri altında model başarımını önemli ölçüde artırma potansiyelini ortaya koymaktadır.

Anahtar Kelimeler: Diferansiyel Gizlilik, Adaptif Kuantizasyon, Örnek Seçimi, Federe Öğrenme

ACKNOWLEDGEMENTS

I would like to sincerely thank my advisor, Assoc. Prof. Yakup Genç, for his consistent support and encouragement during my Ph.D. journey. His mentorship guided the direction of this thesis and helped me grow as a researcher.

I would also like to extend my warmest gratitude to Prof. Fatih Erdoğan Sevilgen, Prof. Haluk Rahmi Topcuoğlu, and Assoc. Prof. Mehmet Göktürk. Their guidance has significantly contributed to the quality of this work.



TABLE OF CONTENTS

	<u>Page</u>
ABSTRACT	vi
ÖZET	vii
ACKNOWLEDGEMENT	viii
TABLE of CONTENTS	ix
LIST of ABBREVIATIONS and ACRONYMS	xi
LIST of FIGURES	xii
LIST of TABLES	xiii
1. INTRODUCTION	1
1.1. Background and Motivation	1
1.2. Problem Definition	3
1.2.1. Communication Efficiency	3
1.2.2. Device and Data Heterogeneity	4
1.2.3. Privacy Concerns	4
1.2.4. Data Valuation	4
1.3. Research Objectives	5
2. LITERATURE REVIEW	7
2.1. Deep Learning	7
2.2. Federated Learning	10
2.2.1. Communication Efficiency	12
2.2.2. Privacy	15
2.2.3. Data Valuation	16
3. METHODOLOGY	18
3.1. FL Baseline	18
3.2. Datasets and Noise Injection Strategies	18
3.3. Multi-Task Autoencoders	20
3.4. Classification Models	21
3.5. Shapley Value	22
3.5.1. TMC-Shapley	24
3.5.2. Gradient Shapley	24
3.5.3. Leave-One-Out	25
3.6. Adaptive Thresholding	26
3.7. Unsupervised Outlier Detection	27
3.8. Federated Multi-Class SVDD Loss	30
3.9. Stochastic Uniform Quantization	31
3.10. Adaptive Quantization	33
3.11. Differential Privacy	34

4. EXPERIMENTS AND RESULTS	37
4.1. Experimental Setup	37
4.2. Sample Selection Baseline	38
4.3. Loss-based Sample Selection	41
4.4. Feature-based Sample Selection	48
4.5. SV-based Sample Selection	57
4.6. Laplacian-based Differential Privacy	58
4.7. Adaptive Quantization	62
4.7.1. Performance on Medical Image Datasets	69
5. CONCLUSION	74
5.1. Summary of Contributions	74
5.2. Future Directions	75
REFERENCES	76
BIOGRAPHY	84
PUBLICATIONS WITHIN the SCOPE of THESIS STUDY	85

LIST of ABBREVIATIONS and ACRONYMS

GTU	:	Gebze Technical University
FL	:	Federated Learning
DL	:	Deep Learning
IoT	:	Internet of Things
SGD	:	Stochastic Gradient Descent
FedAvg	:	Federated Averaging
DP	:	Differential Privacy
CNN	:	Convolutional Neural Network
GPU	:	Graphics Processing Unit
Non-IID	:	Non-Independent and Identically Distributed
AE	:	Auto Encoder
OCSVM	:	One-Class Support Vector Machine
IF	:	Isolation Forest
AT	:	Adaptive Threshold
SV	:	Shapley Value
LOO	:	Leave-one-out
MPC	:	Multi-party Computation
MSE	:	Mean Squared Error
CE	:	Cross Entropy
IR	:	Image Reconstruction
MTAE	:	Multi-task Auto Encoder
SVDD	:	Support Vector Data Description
RQM	:	Randomized Quantization Mechanism
FC	:	Fully Connected
ReLU	:	Rectified Linear Unit
BN	:	Batch Normalization
P2P	:	Peer to Peer
PSNR	:	Peak Signal-to-Noise Ratio
SSIM	:	Structural Similarity Index

LIST OF FIGURES

	<u>Page</u>
Figure 1.1: An FL training example for multiple clients and a single server.	2
Figure 2.1: An example of a two-layered CNN architecture.	7
Figure 2.2: An example of AE with an encoder and decoder for an input X .	8
Figure 2.3: Transposed convolution with 2×2 kernel.	9
Figure 2.4: Centralized, peer-to-peer, and hierarchical FL networks.	11
Figure 2.5: An example of vertical and horizontal FL.	12
Figure 2.6: An illustration of vanilla split learning.	13
Figure 3.1: An illustration of open-set and closed-set noise types.	19
Figure 3.2: The proposed MTAE models for MNIST and CIFAR10.	21
Figure 3.3: A two-layered CNN for MNIST.	22
Figure 3.4: A four-layered VGG network for CIFAR10.	22
Figure 3.5: Computation steps of the proposed SVDD algorithm.	30
Figure 3.6: The bit-length scheduling based on cosine annealing.	33
Figure 4.1: Test accuracy trends across various privacy budgets.	60
Figure 4.2: Round-wise average bit-length for the MNIST and CIFAR10.	62
Figure 4.3: Accuracies and communication costs for various λ_h values.	64
Figure 4.4: Test accuracies across various bit-lengths and client counts.	67
Figure 4.5: BACC scores and average bit-lengths for medical imaging datasets.	72

LIST OF TABLES

	<u>Page</u>
Table 3.1: The datasets used in this thesis.	20
Table 4.1: MNIST results without noise and sample selection.	38
Table 4.2: CIFAR10 results without noise and sample selection.	38
Table 4.3: MNIST results with 40% noise and no sample selection.	39
Table 4.4: CIFAR10 results with 40% noise and no sample selection.	40
Table 4.5: MNIST results with 40% closed-set noise and sample selection.	42
Table 4.6: MNIST results with 40% EMNIST noise and sample selection.	43
Table 4.7: MNIST results with ImageNet32 noise and sample selection.	44
Table 4.8: CIFAR10 results with 40% closed-set noise and sample selection.	45
Table 4.9: CIFAR10 results with 40% SVHN noise and sample selection.	46
Table 4.10: CIFAR10 results with ImageNet32 noise and sample selection.	47
Table 4.11: MNIST results with 40% noise, 1000 clients, and extra metrics.	48
Table 4.12: CIFAR10 results with 40% noise, 1000 clients, and extra metrics.	49
Table 4.13: CIFAR10 results with SVHN noise and feature-based selection.	52
Table 4.14: CIFAR10 results on ImageNet32 noise for feature-based selection.	53
Table 4.15: MNIST results with EMNIST noise and feature-based selection.	54
Table 4.16: MNIST results with ImageNet32 noise and feature-based selection.	55
Table 4.17: CIFAR10 and MNIST results with 40% noise and extra metrics.	56
Table 4.18: Test accuracies for SV-based and LOO-based methods.	58
Table 4.19: MNIST and CIFAR10 results across various gradient norms.	59
Table 4.20: MNIST and CIFAR10 results with a gradient norm of $\xi = 100$.	60
Table 4.21: MNIST and CIFAR10 results across various privacy budgets.	61
Table 4.22: Accuracies and communication costs for 50 and 100 clients.	65
Table 4.23: Accuracies and communication costs for 200 and 1000 clients.	66
Table 4.24: Classification datasets used in medical imaging.	70
Table 4.25: Accuracies and communication costs on medical image datasets.	73

1. INTRODUCTION

1.1. Background and Motivation

Modern distributed networks, such as autonomous vehicles, provide a huge volume of data. As the computational, storage, and battery capabilities of these devices are enhanced, it becomes easier and safer to store and process data locally. This trend has accelerated the adoption of Federated Learning (FL), which is a Machine Learning (ML) approach where numerous clients, including mobile devices or institutions, jointly train a model under the management of a central server while keeping their data local and decentralized [1]. As shown in Figure 1.1, clients keep their raw data locally, sending only model updates to a global server to enhance a learning objective, thus strengthening data privacy. This approach plays a vital role in today's IoT ecosystems, like smart homes, autonomous vehicles, and wearable devices, where protecting sensitive personal data has become more important than ever.

Communication, device heterogeneity, privacy, and statistical heterogeneity of data are the key problems of this learning method [2]. Each device in an FL network has different storage, communication, and computation capabilities due to heterogeneity in hardware (CPU, GPU, RAM), battery capacity, and network connectivity (3G, WiFi). Due to varying connection speed, latency, and bandwidth of the devices in the network, the communication must be as efficient as possible in terms of the total communication rounds and the transmitted package size [3]. Privacy is an important step in developing an FL method, as recent studies have shown the possible exposure of sensitive information through the gradient analysis during the communication rounds of a training process [4]. Methods such as Multi-Party Computation (MPC) and Differential Privacy (DP) offer privacy safeguards but come with additional computational power and possible reduction in the accuracy of deep learning models [5–7]. Statistical heterogeneity of data is another challenge that must be taken into account when training an FL model. Individual devices in an FL network generate training data based on their local environment and usage patterns. Both the size and the distribution of the local datasets typically differ heavily across the devices. This is called non-IID data distribution and introduces bias into the training, which potentially causes slow convergence or even training failures [8].

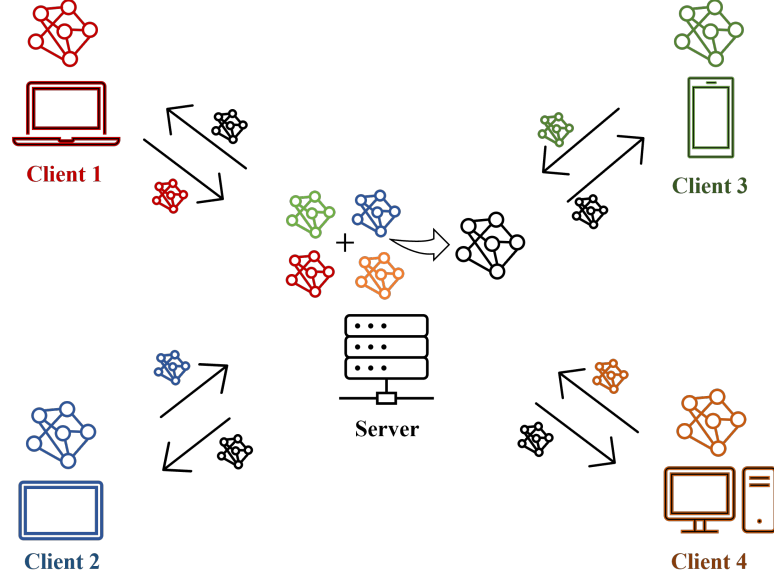


Figure 1.1: An FL training example for multiple clients and a single server.

During an FL training, there may be noisy, malicious, or anomalous samples on clients, which can degrade the accuracy of the global model. Thus, estimating the contribution of local samples to the global model is an important but often overlooked aspect of FL [9]. In this thesis, local sample contributions are predicted directly on clients. This allows us to remove the problematic samples of clients during training [9, 10]. This approach improves model accuracy as well as its robustness against adversarial attacks. In this setting, autoencoders (AEs) are especially effective, as they can capture the intrinsic data structure via input reconstruction, thereby allowing problematic samples to be detected via reconstruction loss. This approach is also crucial for tasks like anomaly detection [11], robust learning [12, 13], and client contribution estimation [9, 14].

This thesis addresses the limitations of current FL data valuation approaches. Although Shapley Value (SV) provides strong theoretical estimations, its computational cost makes it impractical for clients with limited resources [15]. Thus, previous approaches often rely on loss or gradient norm to estimate sample contributions [9, 16]. However, these methods have been relatively underexplored in large-scale, non-IID settings involving a high number of clients (up to 1000), particularly in unsupervised settings. Moreover, the use of AEs in FL, especially under non-IID conditions and at scale, has received limited attention [17]. To overcome these limitations, this thesis presents a novel dual-loss strategy combining classification and reconstruction losses, allowing the detection of mislabeled samples and structural abnormalities, respectively.

Existing studies in FL appear to have largely overlooked the data valuation, differential privacy, and adaptive quantization on non-IID datasets, especially at large scales (e.g., 1000 clients). This gap is essential in real-world FL contexts, where data is frequently non-IID and client participation is highly heterogeneous, both privacy preservation and communication efficiency are significant challenges. This thesis proposes novel solutions by addressing these challenges. The proposed methods are particularly advantageous for applications that require stringent privacy protocols, have limited communication capabilities, and involve noisy data.

1.2. Problem Definition

The primary goal of FL is to collaboratively learn a global model from a large collection of remote devices that generate private data locally. Instead of sharing raw data, only intermediate model updates or gradients are periodically transferred to a central server. While FL is a naturally privacy-preserving method, it has several challenges limiting its applicability and effectiveness in real-world scenarios, particularly in large-scale, non-IID settings. The primary challenges are discussed in the following subsections.

1.2.1. Communication Efficiency

The periodic transfer of model updates or gradients between clients and a central server is a necessary process in FL. However, this may incur considerable communication overhead, affected by factors such as bandwidth constraints, network latency, and different communication protocols (e.g., WiFi, 4G, 5G). To reduce communication costs while preserving learning accuracy, it is crucial to optimize both the total communication rounds and the size of model updates [3]. These issues are even more significant in large-scale FL networks, where the volume of data being transmitted and the number of clients involved greatly increase communication costs. This thesis seeks to minimize the size of model weights exchanged between the server and clients, thereby improving communication efficiency without compromising model accuracy.

1.2.2. Device and Data Heterogeneity

Devices in an FL network often show considerable variation in their computational power, storage capacity, battery life, and networking capabilities. This diversity makes it challenging to aggregate model updates. Furthermore, the data produced by these devices is usually non-IID, indicating that the data distribution can differ significantly among clients. This non-IID distribution can introduce biases during model training, potentially resulting in slower convergence, reduced model performance, or even training failures [8]. Consequently, effective model aggregation and generalization in these contexts pose significant problems. This thesis aims to improve communication efficiency and user privacy under non-IID data distributions.

1.2.3. Privacy Concerns

The key advantage of FL is the preservation of data privacy while keeping data local to the clients. However, there are still risks associated with the transmission of model updates. The recent works have shown that adversarial clients may infer sensitive information by analyzing the model updates shared with the server during training [18, 19]. To improve privacy, well-known techniques are MPC and DP [5, 7]. DP adds systematic noise to model updates, and this makes it a popular choice due to its simple implementation and user-friendliness. Although these techniques provide privacy guarantees, they generally result in higher computational overheads and possible reductions in model accuracy, leading to a trade-off between safeguarding privacy and preserving learning accuracy. This thesis prioritizes DP due to its ease of use and lower computational cost compared to alternatives like MPC.

1.2.4. Data Valuation

Identifying the contribution of individual data samples on clients is crucial for optimizing FL performance [9]. The client may have low-quality samples, such as noisy, malicious, or abnormal data, which can reduce the accuracy of FL models, especially on non-IID datasets. Although gradient norm and loss-based methods are commonly used for sample contribution estimation, their performance has not been comprehensively

studied in federated settings characterized by non-IID data and a large number of clients. Moreover, traditional anomaly detection methods for FL are limited in their ability to effectively handle the challenges posed by non-IID data. This thesis focuses on computationally efficient methods for estimating sample contributions on clients while preserving privacy and communication efficiency.

1.3. Research Objectives

In this thesis, one of the key objectives is to develop a novel unsupervised method for local data valuation in FL. The proposed method employs a Multi-Task Autoencoder (MTAE) framework with Image Classification (IC) and Image Reconstruction (IR) purposes. This enables the estimation of each sample’s contribution based on either loss or feature representations. Using these contribution estimates, outlier samples are detected during local training on clients using unsupervised anomaly detection models such as One-Class Support Vector Machine (OCSVM) and Isolation Forest (IF). These models are periodically updated at the central server using either loss values or extracted features from participating clients, depending on the selected valuation method. Additionally, a loss-based adaptive threshold (AT) method, inspired by FedBalancer [16], is employed to detect outlier samples on clients. This approach uses a weighted combination of IR and IC losses for contribution estimates. Moreover, inspired by the work of Cheng et al. [20], a novel multi-class deep Support Vector Data Description (SVDD) algorithm is proposed to enhance feature-based sample selection in FL.

Another objective is to develop novel differential privacy and adaptive quantization methods to address both privacy and communication efficiency. Laplacian-based DP is used to strengthen privacy protection, which is relatively underexplored in FL and offers stricter privacy guarantees than Gaussian-based DP. An efficient global bit-length scheduling strategy is presented, using round-wise cosine annealing to progressively adjust precision levels during training. In addition, a client-specific scheduler is designed to dynamically refine bit-length allocation by assessing client importance, estimated through the entropy of their local datasets. Specifically, different quantization methods are used for server-to-client and client-to-server communications. For server-to-client transmission, a round-based cosine annealing schedule is employed, starting with 32-bit precision and gradually reducing it as training progresses. To optimize client-to-server

communication, a combination of round-based cosine annealing and dataset entropy is used to adjust the bit length for clients. The Shannon entropy is computed on the local dataset of each client to efficiently measure the amount of information in the dataset [21]. The high entropy indicates that the dataset is balanced and contains multiple classes, thereby leading to increased accuracy and easier convergence of models. The clients with higher entropy tend to have less compression of their weights, thus contributing more to the global model.

In experiments, the proposed FL method is tested under different degrees of open-set and closed-set noises, with the training data partitioned among clients following a non-IID distribution. The main objectives of this thesis are:

- An unsupervised sample selection method using MTAE architecture is proposed to improve the performance of FL models by utilizing OCSVM and IF algorithms.
- A novel multi-class deep SVDD loss specifically designed for FL is proposed to improve feature-based outlier detection.
- A simple and efficient global bit-length scheduler using round-based cosine annealing to adjust bit-length globally during training, thereby enhancing communication efficiency without compromising model accuracy.
- A novel adaptive quantization approach using the Shannon entropy to adjust bit-length dynamically on clients during training, thereby prioritizing clients with more information in their datasets.

2. LITERATURE REVIEW

2.1. Deep Learning

Deep Learning (DL) is a subfield of machine learning that automatically learns data representations at various levels of abstraction by using multi-layered artificial neural networks [22]. DL eliminates the need for hand-engineered features by learning them directly from data, enabling scalable solutions in high-dimensional and unstructured data environments. Thanks to DL, various applications, including computer vision, speech recognition, and natural language processing, have advanced significantly. The main DL concepts and architectures relevant to this thesis are described in this section.

Convolutional Neural Networks (CNNs) are widely used in computer vision tasks due to their ability to extract local spatial features from images [23]. As illustrated in Figure 2.1, the convolution operation forms the foundation of CNN architectures, where the convolution layers employ learnable filters on input data to extract local spatial features such as edges, textures, and object structures. This hierarchical extraction can capture both low-level and high-level representations in images. In contrast to fully connected (FC) layers, CNNs use weight sharing and local connectivity to minimize the number of parameters, enabling more effective and scalable training.

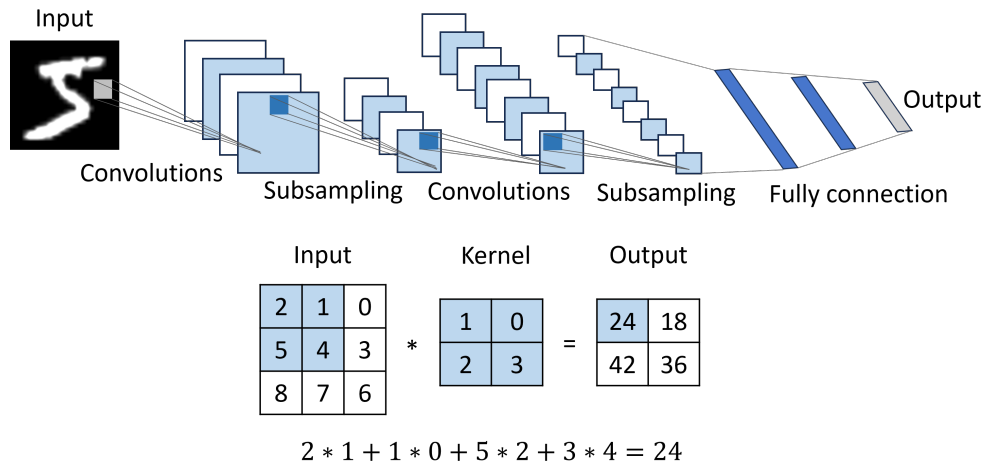


Figure 2.1: An example of a CNN architecture with convolution, subsampling, and FC layers, along with the step-by-step computation of a 2D convolution using a 2×2 kernel.

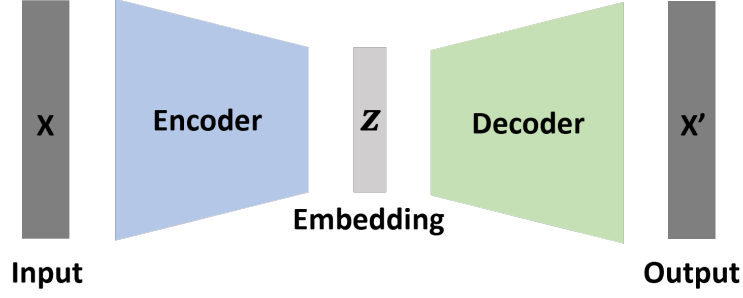


Figure 2.2: An illustration of AE architecture featuring an encoder, decoder, and embedding Z for input X and output X' .

Autoencoders are a type of neural network that is trained to reconstruct its input, as shown in Figure 2.2. They consist of an encoder that compresses the input into a latent representation, also known as an embedding vector, and a decoder that reconstructs the original data from this embedding [24]. The training objective is to minimize the reconstruction error. This error is usually computed by the Mean Squared Error (MSE) or binary Cross-Entropy (CE) functions. Applications including dimensionality reduction, anomaly detection, and representation learning make AEs rather common. Particularly in encoder-decoder systems involving image reconstruction or generation, transposed CNN layers, also known as deconvolution layers, are usually used [25]. As shown in Figure 2.3, these layers learn to map low-resolution feature maps back to high-resolution outputs, enabling applications such as image reconstruction and segmentation.

Training deep neural networks relies on iteratively adjusting model parameters in response to optimization techniques designed to minimize a predefined loss function. Equation (2.1) shows that Stochastic Gradient Descent (SGD) is a fundamental optimization method whereby gradients are computed using mini-batches of data [26]. The learning rate is indicated here η ; the gradient of the loss function with regard to the model parameters is $\nabla_{\theta} \mathcal{L}_{\text{batch}}$. Variants, including SGD with momentum, RMSProp, and Adam, add further mechanisms for adaptive learning rates and momentum-based updates, thereby enhancing convergence behavior and robustness.

$$\theta_{t+1} = \theta_t - \eta \nabla_{\theta} \mathcal{L}_{\text{batch}}(\theta_t) \quad (2.1)$$

Deep network representational power depends on non-linear activation functions, which also allow them to learn intricate decision boundaries [27]. The Rectified Linear Unit

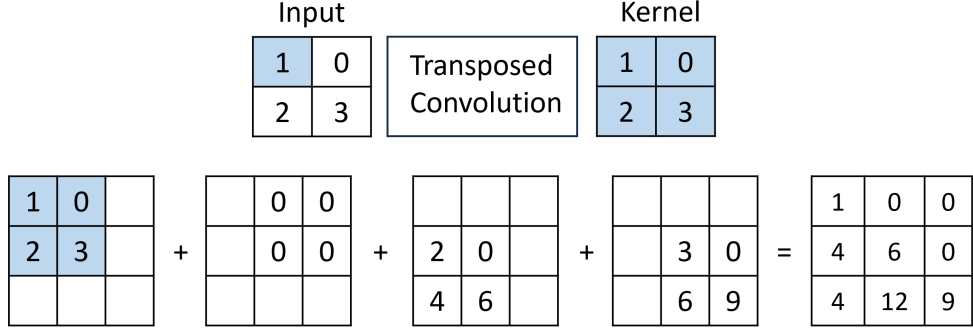


Figure 2.3: Transposed convolution with 2x2 kernel. The blue areas are a portion of an intermediate tensor, as well as the input and kernel tensor elements.

(ReLU), which is widely used due to its simplicity and ability to mitigate the vanishing gradient problem, is defined in Equation (2.2).

$$\text{ReLU}(x) = \max(0, x) \quad (2.2)$$

Usually, in multi-class classification problems, the softmax function is preferred for output layers [28]. Equation (2.3) shows how raw output scores are transformed into probabilities that sum to one, enabling probabilistic interpretation of predictions.

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}} \quad (2.3)$$

Another often used activation function is the Sigmoid, defined in Equation (2.4), mapping input values into the interval $(0, 1)$. It is especially appropriate for binary classification problems and probabilistic interpretations. Its use in hidden layers is sometimes limited, though, by its sensitivity to saturation and vanishing gradients in deep layers.

$$\text{Sigmoid}(x) = \frac{1}{1 + e^{-x}} \quad (2.4)$$

Batch Normalization (BN) is a regularization and training acceleration technique that normalizes the inputs of each layer by adjusting and scaling the activations [29]. Given an input activation x in a mini-batch, BN transforms it by Equation (2.5), where μ and σ^2 are the mini-batch mean and variance, and γ, β are learnable parameters. BN helps to stabilize and speed up training by reducing internal covariate shift. It also enables the use of higher learning rates and acts as a form of regularization, often reducing the need for dropout.

$$\hat{x} = \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}}, \quad y = \gamma\hat{x} + \beta \quad (2.5)$$

Still another essential element of CNN designs is max pooling [30]. It chooses the maximum value inside a predefined local region, usually a 2×2 window, so lowering the spatial resolution of feature maps. As shown in Equation (2.6), this operation lowers computational cost, improves translational invariance, and helps to prevent overfitting.

$$y_{i,j} = \max_{(m,n) \in \Omega(i,j)} x_{m,n} \quad (2.6)$$

These elements, taken together, provide the basis for DL models applied throughout this thesis. The development of advanced architectures and optimization strategies in the framework of FL depends on an understanding of their roles and interactions.

2.2. Federated Learning

Although DL has demonstrated remarkable success across various fields, it typically relies on large, centrally collected datasets to build robust models. However, in many real-world applications, privacy, security, or legal constraints restrict the centralization of data, resulting in naturally distributed datasets across multiple devices or institutions. FL has emerged as a promising paradigm to overcome these issues by using a collaborative model training directly on decentralized data sources without exchanging raw data [1].

A typical FL framework involves a central server managing training across a network of client devices, each possessing its own local dataset. In each communication round, a subset of clients performs local training and sends only the model updates back to the server, which aggregates them to update a global model. Federated Averaging (FedAvg), the most commonly used aggregation algorithm, calculates a weighted average of these local model updates based on the local data size of each client [1]. This approach ensures data privacy and significantly reduces communication overhead compared to traditional centralized training methods.

The network topology employed in FL typically follows a star topology, with a central server directly communicating with each client, as shown in Figure 2.4. However, other topologies such as Peer-to-Peer (P2P) [31] and hierarchical structures [32] have recently gained attention to further optimize communication and scalability. P2P topologies eliminate the need for a central server by allowing clients to directly exchange updates, thus improving scalability and fault tolerance. On the other hand, hierarchical topologies

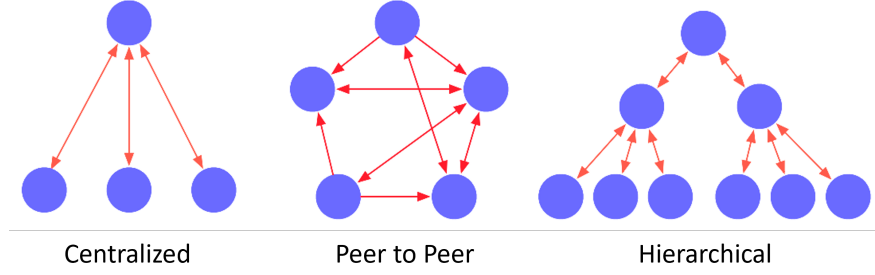


Figure 2.4: Centralized, peer-to-peer, and hierarchical FL networks.

organize clients into clusters, each having a local aggregator, thereby reducing the load on the central server and enhancing scalability for large-scale deployments.

System heterogeneity introduces further complexity in FL systems. Disparities in hardware capabilities, communication bandwidth, and device availability lead to the adoption of asynchronous communication protocols [33]. However, this can create a bias towards faster devices. Deciding which client to select per round is crucial, as it can significantly impact learning accuracy and total training time. A common approach is to select devices based on criteria such as network and computational capacity in each round [34]. Fault tolerance is also crucial in FL, requiring measures to handle device dropouts during training. Simple methods either ignore dropped devices or select only those with good network quality to ensure stable training [35].

Moreover, FL can be categorized based on data partitioning approaches, namely horizontal and vertical FL [36]. Horizontal FL is used when datasets across devices share the same features but vary in samples (e.g., mobile users), whereas vertical FL is suitable when datasets across institutions differ in features but align in their sample identifiers (e.g., banks and insurance companies with common customers).

Split learning is another innovative approach closely related to FL, where a DL model is partitioned between clients and servers, allowing model parts rather than raw data or entire models to be communicated [37]. As illustrated in Figure 2.6, in vanilla split learning, the model is partitioned such that initial layers are executed on the client side, and intermediate activations are communicated to the server. The server computes forward propagation using these activations, calculates the loss, and performs backpropagation to compute gradients, which are then sent back to the client for updating

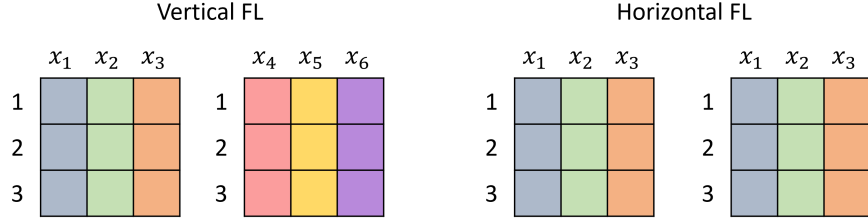


Figure 2.5: An illustration of vertical and horizontal FL architectures, showing how data is partitioned across clients based on features and samples.

its local model. This method further reduces communication overhead and enhances privacy by keeping sensitive portions of the model and raw data locally at client devices.

This thesis investigates the challenges and advanced techniques in FL to enhance the system’s privacy, scalability, and efficiency under realistic and constrained settings.

2.2.1. Communication Efficiency

Communication bottlenecks can arise due to variations in connection speed and bandwidth across devices in an FL network. It can be expensive and unreliable if the edge devices, such as mobile phones, have metered connections and their availability cannot be predicted. In order to reduce total training time and speed up the convergence of deep learning models in FL settings, the communication should be optimized in terms of the total time elapsed in communication and the total size of messages transferred during the training rounds. In addition, using a single global server can create a bottleneck for an FL network with millions of devices. This has led to a significant recent interest in reducing the communication cost of FL, which can be categorized into two groups: decreasing the total number of communication rounds and data compression.

In vanilla FL, each client sends a local update to a global server after completing its training. The training process of a typical deep learning model generally takes hundreds of epochs. This significant overhead can be reduced by increasing the communication period. FedAvg and other similar methods execute multiple epochs of local training before the communication stage [1]. The communication interval is a tunable hyperparameter that greatly affects the final accuracy of the global model. While short intervals are better for model performance, long intervals increase the convergence speed of the

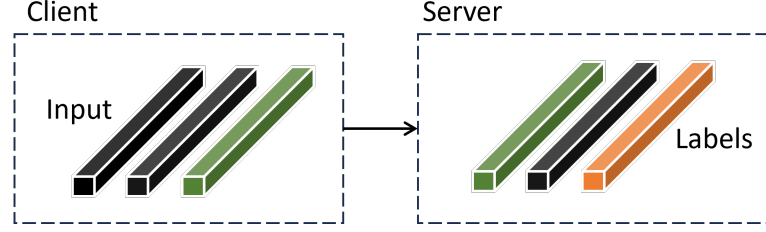


Figure 2.6: A vanilla split learning, showing how the layers are partitioned between client and server, enabling intermediate activations and gradients to be communicated while maintaining data privacy.

model at the cost of gradient computation. Konecny et al. propose long communication intervals with random sparsification and probabilistic quantization to reduce both the total number of communication rounds and the size of messages [38].

In addition to reducing the total number of communication rounds, there are efficient methods such as sparsification, subsampling, and quantization to compress the messages transmitted at each iteration [38–41]. However, most of these methods are analyzed in data center environments that do not have the FL challenges like the low participation of devices, non-IID local data, and local updating schemes. Nevertheless, there are several FL approaches, such as quantization with random rotations [40], lossy compression with federated dropout [42], and lossless encoding [8]. Furthermore, several works propose convergence guarantees with low-precision training for non-IID data, but they do not regard low device participation or local optimization methods like FedAvg [43].

Compression can be applied to the gradients computed at clients or the model weights on the server. As clients typically have more limited upload bandwidths compared to their download bandwidths, gradient compression proves highly practical for reducing both total training time and the overall size of transmitted messages. Similarly, reducing the size of the global model is necessary since the clients in an FL network download the latest weights before computing local gradients. In order to compress the gradients computed at clients, sparsification methods restrict model updates to a small subset of parameters that are randomly selected or based on predefined criteria [39]. It is primarily used to transmit only essential gradients to the server. Although its compression rate is much higher than quantization, it introduces extra computation overhead during the training, which can be a problem for devices with low battery capacity or performance.

Quantization methods try to reduce the gradients to a smaller set of values. In other words, the original gradients are represented by a low-precision data type such as 1 byte or 1 bit. Since 32-bit is the frequently used data type in deep learning, the maximum compression rate is restricted to $1/32$. Convergence speed is also much slower due to information loss, as discussed in [2]. In contrast, the compression rate of sparsification can be up to $1/100$ without a major cost of reduced convergence speed and accuracy. Bernstein et al. propose signSGD to quantize each gradient update to its binary sign, which reduces the size of each gradient value by a factor of 32 [44]. Theoretically, it is shown that this compression method has a convergence guarantee on IID data. Other methods like TernGrad [45], QSGD [46], and ATOMO [40] propose probabilistic approaches to quantize the gradients before sending them to the server.

Recent studies have focused on adaptive quantization methods that change quantization precision in response to the training phase or model characteristics. For example, AdaQuantFL [47] employs a stochastic uniform quantization approach to dynamically adjust quantization levels, thereby striking a balance between communication efficiency and quantization error. Likewise, FedDQ [48] introduces a descending quantization technique that accelerates training convergence by reducing bit lengths as model updates decrease during training. However, this technique assumes uniform client contributions, overlooking dataset diversity and heterogeneity. In more recent work, FedAQ [49] proposes a combined uplink and downlink adaptive quantization approach designed to optimize communication in resource-limited settings; however, it overlooks issues related to non-IID distributions or privacy. To address these drawbacks, this thesis proposes a novel descending quantization strategy for both uplink and downlink communications, incorporating a new mechanism for client importance estimation that effectively manages non-IID datasets. Additionally, unlike previous studies, a comprehensive evaluation is conducted on a large scale, featuring experiments with as many as 1000 clients. This innovative design closes the critical gaps in downlink optimization and data heterogeneity identified in existing methods, all while significantly enhancing communication efficiency and scalability.

2.2.2. Privacy

The primary motivation is to make sure that raw data on each client remains local in an FL network. It is shown that the gradients or model updates can reveal sensitive information about the data [18, 19]. For example, Carlini et al. show that sensitive text data, such as credit card numbers, can be revealed by analyzing recurrent neural network models trained on the language data of users. In the literature, recent works use differential privacy, secure multiparty computation, and other cryptographic algorithms to preserve the privacy of transmitted messages and local data on each client. In FL, low device participation tolerance, computation, and communication efficiency are necessary without a significant decrease in the model accuracy. In this section, privacy issues in FL networks and possible solutions are explained.

DP is a frequently used randomized method to reduce the relationship between the input and output of deep learning models. Ideally, it is expected that a change in one input feature does not cause too much difference in the distribution of the output. The goal is to obscure whether a specific sample is used in the learning process or not. This sample-level privacy can be used in many machine learning applications [38, 50, 51]. For gradient-based learning, a simple and popular approach is the random perturbation of the outputs at each iteration [51]. In order to restrict the effect of each sample on the entire update, the gradients can be clipped. The perturbation can be done by adding noise with probabilistic methods such as Laplacian [52], Gaussian [50], or Binomial [53]. Since increasing the perturbation rate can decrease the accuracy of the model, it can be hard to balance this trade-off.

Recent research on privacy in FL is categorized into two types: global and local privacy [4–7, 19, 54–56]. Global privacy ensures that model updates during training remain inaccessible to all untrusted third parties, except the global server. In contrast, local privacy extends this protection by keeping updates private even from the server. Cryptographic protocols, such as secure multiparty computation, enhance FL privacy without sacrificing learning accuracy, as shown by Bonawitz et al. [7]. Nonetheless, implementing these protocols can lead to significant communication overhead.

Numerous studies have explored strategies to enhance both privacy and communication efficiency in FL. Lang et al. introduce JoPEQ, which integrates lossy compression

with privacy improvements via vector quantization and local DP [57]. However, it relies on static quantization rather than adaptive methods tailored to client or parameter importance. Likewise, MSPDQ-FL implements a model-splitting technique with dynamic quantization to strengthen privacy and lower communication costs in non-IID data scenarios [58]. Still, its dynamic quantization targets submodel parameters rather than encompassing the entire parameter spectrum. Youn et al. present the Randomized Quantization Mechanism (RQM), which ensures privacy by merging randomized subsampling with rounding of quantization levels, aligning with Renyi DP [59]. Despite its novel approach to privacy-preserving quantization, RQM mainly concentrates on trade-offs between privacy and accuracy, lacking adaptive strategies. Lastly, Nguyen et al. offer a framework that combines quantization and binomial noise to refine privacy and communication metrics, though it, too, lacks adaptive quantization techniques and a clear scalability analysis for extensive FL scenarios [60].

In this thesis, Laplacian-based DP is preferred rather than Gaussian-based DP, as it offers stronger privacy guarantees under ℓ_1 -sensitivity, making it suitable for bounded updates in FL [61]. Initially, Laplace noise is introduced to protect model updates, followed by adaptive quantization, which dynamically adjusts bit lengths according to an innovative client importance estimation mechanism. This sequential approach ensures effective integration of added noise into the quantized updates, achieving a balance among privacy, accuracy, and communication costs, especially in non-IID and large-scale FL settings.

2.2.3. Data Valuation

The research on data and client valuation within FL is relatively scarce, particularly when applying SV techniques. For instance, Wang et al. introduce a Federated Shapley Value (FSV) method, which is a variant of SV that is a well-known concept in cooperative game theory used to calculate the total contributions of a set of players [62]. They treat FL as a cooperative game involving multiple clients or data sources [15]. Similarly, Wang et al. compute the contributions of participants in both vertical and horizontal FL by using SV for vertical FL and a deletion method for horizontal FL to estimate the importance of grouped features and instances [63]. Ardic et al. adopt methods based on

leave-one-out and SV to estimate sample contributions on clients, but these approaches require frequent retraining of local models, making them computationally expensive and difficult to apply in FL environments [10].

An alternative method for estimating sample contributions involves using loss or gradient norms, which are calculated by performing a forward pass of the model [9, 16, 64]. Although SV methods are accurate, loss-based methods provide a faster, more efficient solution, especially given the limited computational capabilities of FL clients. For example, Li et al. propose using gradient norms to estimate the importance of samples with a single forward pass, by summing the importance of each client’s local samples [9]. During training, the server selects a subset of important clients and their high-quality samples in each round to improve accuracy. However, compared to loss-based approaches, gradient norm calculations impose higher computational and memory overhead. Shyn et al. propose a basic model to estimate client contributions by considering only data size [65], while Tuor et al. suggest a benchmark model trained on a small dataset to handle noisy and irrelevant data within FL, but their approach is task-specific and not easily generalized [66].

Anomaly detection within FL has gained growing interest due to its importance across various fields, including cybersecurity, healthcare, and industrial systems [67–69]. A common strategy is to use autoencoders to detect noisy or malicious samples by evaluating reconstruction loss [11, 12, 70–72]. Nardi et al. propose an unsupervised anomaly detection method for decentralized FL, where clients are grouped into communities based on similar data patterns, and a shared autoencoder-based anomaly detection model is trained within each group [17]. However, this approach is not well-suited for large-scale FL scenarios involving a high number of clients and has been evaluated only within fully connected networks. Bhat et al. propose a performance-driven aggregation approach for FL, in which client models are weighted according to their F1 scores to enhance anomaly detection and reduce the influence of clients with poor performance. [72]. However, this method does not consider non-IID data distributions, lacks evaluation in large-scale client settings, and depends on the availability of a test dataset on each client to calculate F1 scores.

3. METHODOLOGY

3.1. FL Baseline

In this thesis, a standard synchronous centralized FL network is employed, consisting of a server and N clients, represented as $\mathcal{N} = \{\zeta_1, \dots, \zeta_N\}$. In this network, each client ζ_i has a local dataset $D_i = \{(x_{ij}, y_{ij})\}_{j=1}^{n_i}$, where x_{ij} denotes j -th sample on client i , $y_{ij} \in [c_1, \dots, c_K]$ is the label, and $n_i = |D_i|$ is the total sample count. The primary goal is to develop a unified global model using the data produced by clients. Only model updates are periodically sent back and forth between the clients and the server during this process. In each round, the FedAvg algorithm is used by the server to aggregate the local models gathered from remote devices [1]. For a local sample $(x_{ij}, y_{ij}) \in D_i$, the loss function can be defined as $\ell(f(x_{ij}; \theta), y_{ij})$, where x_{ij} is an input sample, y_{ij} is the ground-truth label, and θ is the parameters of the model that is defined by $f(x_{ij}; \theta)$. Accordingly, the model is formulated as a function of the input x_{ij} , with θ representing its parameters. Equation (3.1) defines the overall loss function $\mathcal{L}_i(\theta)$ for a client ζ_i . Each client seeks to optimize its local loss function $\mathcal{L}_i(\theta)$ during the training phase.

$$\mathcal{L}_i(\theta) = \frac{1}{n_i} \sum_{j=1}^{n_i} \ell(f(x_{ij}; \theta), y_{ij}) \quad (3.1)$$

As shown in Equation (3.2), the primary objective of FL is to minimize the global loss function [2]. Each client uses the SGD optimizer on its local model to train over several epochs in order to solve this objective function in a distributed manner.

$$\hat{\theta} = \arg \min_{\theta} \frac{\sum_{i=1}^N n_i \mathcal{L}_i(\theta)}{\sum_{i=1}^N n_i} \quad (3.2)$$

3.2. Datasets and Noise Injection Strategies

In an FL network, the training process can be intentionally or unintentionally disrupted by the clients that generate noisy, redundant, or abnormal samples. In this thesis, it is assumed that the clients are trusted and cooperative. In FL, however, the data is inaccessible and remains distributed across clients. Thus, it is challenging to select and prioritize only valuable samples during the training. To simulate such a scenario,

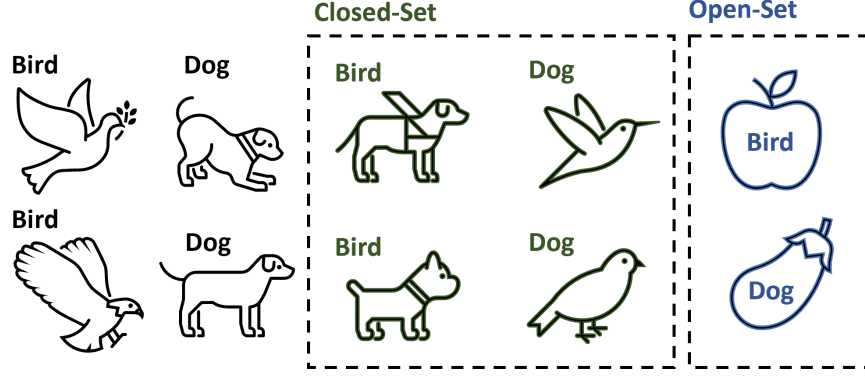


Figure 3.1: An illustration of open-set and closed-set noise types. The samples of unknown classes that come from another dataset are called open-set noise, whereas incorrectly labeled samples are known as closed-set noise.

open-set and closed-set noise are injected into the training datasets and distributed to clients in a non-IID manner. These noise types, shown in Figure 3.1, are as follows:

1. *Closed-set noise:* This refers to cases where a sample from a known class is incorrectly labeled as another class that is also part of the known set.
2. *Open-set noise:* This occurs when a sample from an unknown or unseen class is mistakenly labeled as belonging to one of the known classes.

In this thesis, closed-set noise is simulated by randomly changing the labels in the training dataset, while open-set noise is introduced by replacing certain training samples with images sourced from unrelated datasets. The proposed FL models are trained and tested by using MNIST [73] and CIFAR10 [74] datasets. MNIST is a famous dataset in ML, including 28x28 grayscale images of handwritten digits collected from 1,000 individuals. CIFAR10, another well-known dataset, includes 32x32 color images across 10 categories, like cars, birds, and trucks. To simulate open-set noise, additional datasets are incorporated, including Street View House Number (SVHN) [75], ImageNet32 [76], and Extended MNIST (EMNIST) [77]. Moreover, the federated variant of EMNIST, proposed by Google AI [78], is also used. SVHN contains 32x32 cropped digit images extracted from house numbers in Google Street View images, across 10 classes. EMNIST extends MNIST by including upper-case and lower-case English characters. Additionally, ImageNet32, a downscaled version of ImageNet, consists of 32x32 color images spanning 1,000 classes. More details of these datasets are provided in Table 3.1.

Table 3.1: The datasets used in this thesis, where CIFAR10 and MNIST are for training, while SVHN, EMNIST, and ImageNet32 are open-set noise datasets.

Dataset	Image Size	Train Examples	Test Examples	Classes
CIFAR10	32x32 RGB	50,000	10,000	10
MNIST	28x28 Gray	60,000	10,000	10
SVHN	32x32 RGB	73,257	26,032	10
EMNIST	28x28 Gray	671,585	77,483	62
ImageNet32	32x32 RGB	1,281,167	50,000	1,000

3.3. Multi-Task Autoencoders

A sample from the dataset D can be depicted as $(x_j, y_j) \in D$, where x_j is the input image and y_j is its label. A multi-task autoencoder architecture employed in this thesis consists of an encoder, a decoder, and a classifier, defined respectively as $f : x_j \rightarrow z_j$, $g : z_j \rightarrow \hat{x}_j$, and $h : z_j \rightarrow \hat{y}_j$. In this setup, the reconstructed image is given by $\hat{x}_j = g(f(x_j))$, while the predicted label is denoted as $\hat{y}_j = h(f(x_j))$. The intermediate representation z_j is the embedding of x_j within the feature space $\mathcal{F}$. The reconstruction loss $\mathcal{L}_{\text{rec}}(x_j, \hat{x}_j)$ is computed by mean squared error, and the classification loss $\mathcal{L}_{\text{cls}}(y_j, \hat{y}_j)$ is computed by cross-entropy. The combined loss $\mathcal{L}$, as defined in Equation 3.3, aims to jointly minimize both losses through $(\min_{f,g,h} \mathcal{L})$. The hyperparameters λ_{rec} and λ_{cls} are employed to balance the contributions of reconstruction and classification losses, thereby controlling the model’s sensitivity to pixel-level and label noise, respectively.

$$\mathcal{L} = \lambda_{\text{rec}} \cdot \mathcal{L}_{\text{rec}}(x_j, \hat{x}_j) + \lambda_{\text{cls}} \cdot \mathcal{L}_{\text{cls}}(y_j, \hat{y}_j) \quad (3.3)$$

The MTAE architecture designed for the MNIST is shown in Figure 3.2a. An encoder based on a two-layer CNN with 32 and 64 filters is used to align with the literature, similar to the one used in the FedAvg study [1]. In addition, the MTAE model designed for the CIFAR10 dataset with 32x32 color images is shown in Figure 3.2b. In this model, an encoder with a four-layered CNN architecture consisting of 64, 128, 128, and 256 filters is used, each having a kernel size of 3x3, a stride of 2, and a padding of 1. The sigmoid activation function in the final layer of the decoder is also used to produce output values between 0 and 1.

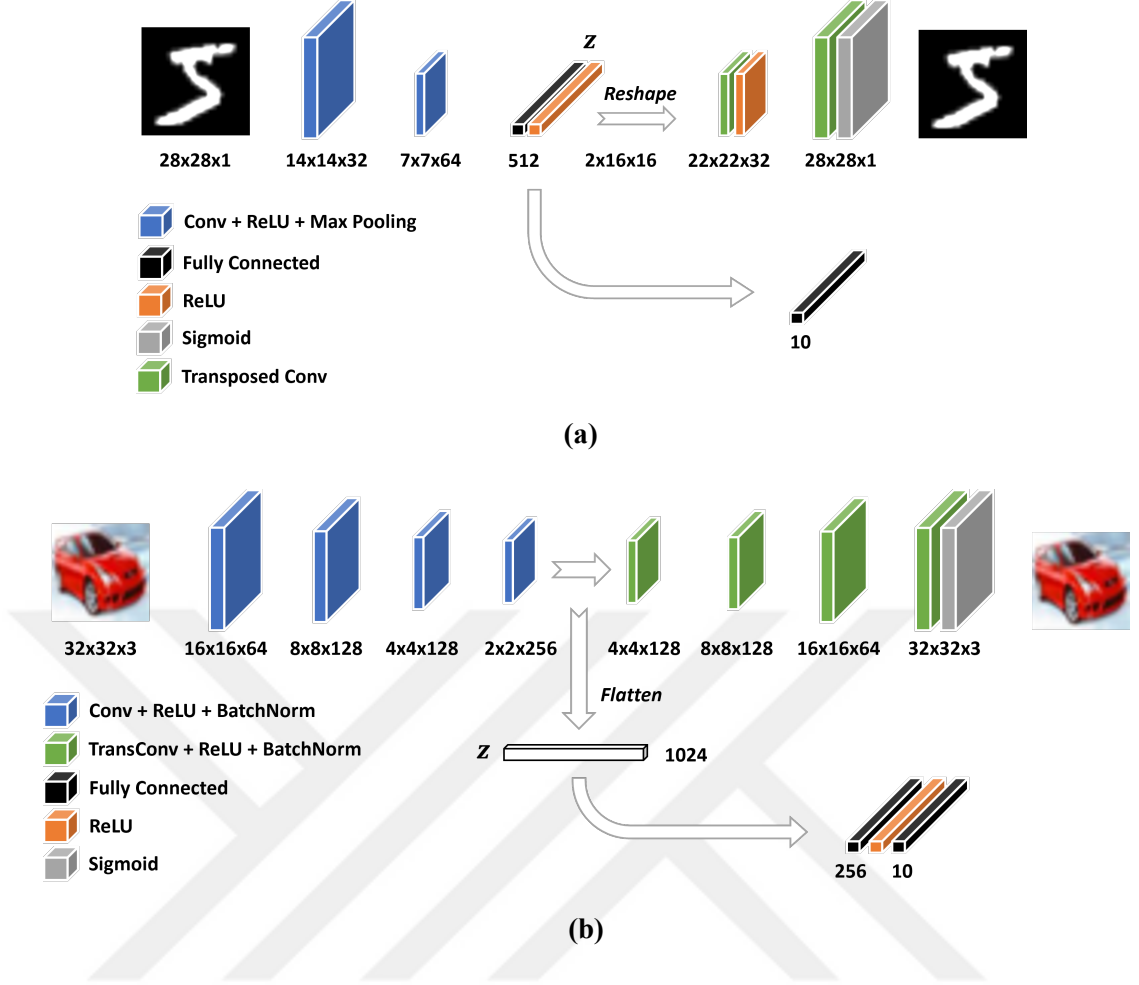


Figure 3.2: The proposed MTAE architectures for MNIST (a) and CIFAR10 (b) datasets. Best viewed in color.

3.4. Classification Models

In this section, the classification models designed for the MNIST and CIFAR10 datasets are presented. The MNIST model is presented in Figure 3.3. A simple two-layered CNN consisting of 32 and 64 filters is used to align with the literature, which is also used in the FedAvg study [1]. The kernel size, stride, and padding parameters of the CNN layers are set to 5, 1, and 2. Following each convolution layer, the ReLU activation function and a max-pooling layer with a kernel size of 2x2 and a stride of 2 are used. The final convolutional layer's output is flattened and passed to an FC layer with ReLU activation. This gives a 1D output of length 512. An FC layer with a Softmax activation function is employed for the final classification, which assigns outputs to one of ten classes.

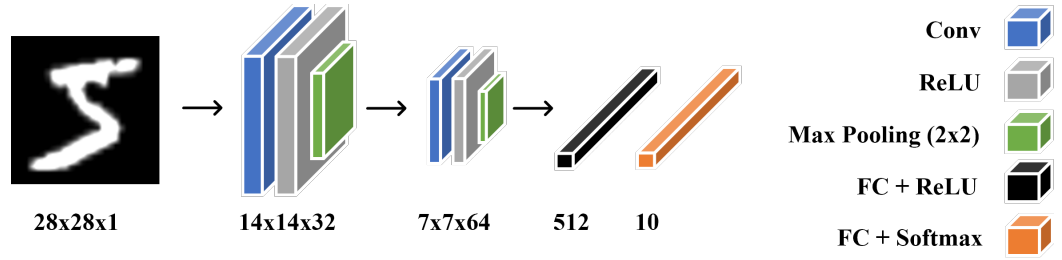


Figure 3.3: A two-layered CNN designed for 28x28 images in the MNIST dataset. Recommended to view in color.

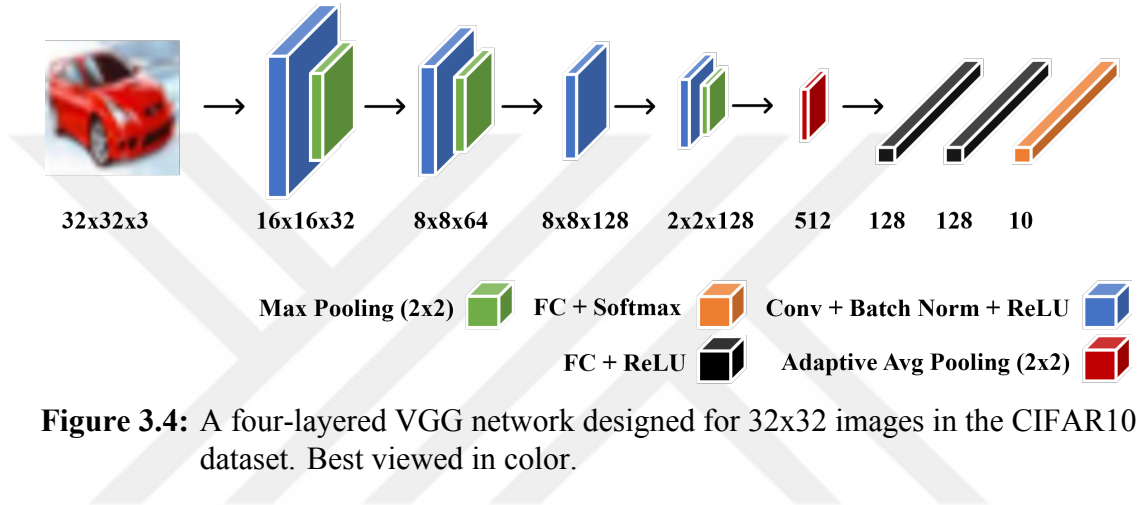


Figure 3.4: A four-layered VGG network designed for 32x32 images in the CIFAR10 dataset. Best viewed in color.

The CIFAR10 model, designed for 32x32 color images, is shown in Figure 3.4. The model is a four-layered CNN with 32, 64, 128, and 128 filters, where each convolutional layer utilizes a 3x3 kernel, a stride of 1, and a padding of 1. First, the ReLU function and then batch normalization are used after each convolutional layer to ensure stable and efficient training. The output of the final convolutional layer is passed to adaptive average pooling with a window size of 2x2. This gives an output size of 2x2x128. This output is then flattened into a 1D vector of length 512. Following this, two FC layers with the ReLU function are employed. This produces a 1D vector of length 128. The final classification is performed by an FC layer with the Softmax function.

3.5. Shapley Value

The SV is a concept from cooperative game theory that provides a principled way to fairly attribute the total utility of a system to individual contributors. In ML, it quantifies

the contribution of each data sample to the model's overall performance. This allows for data valuation by measuring how much each sample improves or degrades the accuracy or loss when included in the training set. Due to its desirable fairness properties, as outlined below, the SV has gained popularity in data-centric applications such as FL, where evaluating the impact of potentially noisy data is crucial [62, 79]:

- *Efficiency*: The total gain is fully distributed among all contributors.
- *Proportionality*: Contributions are allocated in accordance with actual influence.
- *Additivity*: Values accumulate naturally when a sample is used across multiple evaluations.

The exact computation of the SV involves evaluating all 2^n subsets of the training set $D = \{(x_i, y_i)\}_{i=1}^n$, which is computationally infeasible for large datasets. Thus, two approximation methods, Truncated Monte Carlo Shapley (TMC-Shapley) and Gradient Shapley (G-Shapley), are used in this thesis to estimate sample contributions [79].

$$\phi_i = C \sum_{S \subseteq D \setminus \{i\}} \frac{V(S \cup \{i\}) - V(S)}{\binom{n-1}{|S|}}, \quad (3.4)$$

Given a learning algorithm A and a performance evaluation function V , the data SV of the i -th data point is defined by Equation 3.4, where C is a normalization constant, S denotes all possible subsets not containing the i -th point, and $V(S)$ denotes the model performance when trained on subset S .

To make the computation tractable, Equation 3.4 can be reformulated as Equation 3.5, which represents an expectation over all permutations π of the dataset, where S_i^π denotes the subset of data points preceding i in permutation π , and Π is the uniform distribution over all $n!$ permutations.

$$\phi_i = \mathbb{E}_{\pi \sim \Pi} [V(S_i^\pi \cup \{i\}) - V(S_i^\pi)], \quad (3.5)$$

In the context of FL and this thesis, TMC-Shapley, G-Shapley, and LOO-based methods are executed locally on each client's dataset. While this design maintains data privacy, it also imposes significant computational overhead on resource-constrained client devices.

3.5.1. TMC-Shapley

The TMC-Shapley in Algorithm 1 works by sampling permutations from Π and averaging their marginal contributions [79]. Each permutation involves visiting data points sequentially and recording the change in model performance with the addition of every new sample, which is considered that sample's contribution. By repeating this across various random permutations, the final SV for each data point is determined as the average of its contributions. To improve efficiency, once the performance difference $|V(D) - V(S)|$ falls below a predefined threshold, further marginal contributions within that permutation are assumed negligible and set to zero. This strategy significantly reduces computational demands while maintaining fidelity.

Algorithm 1: The TMC-Shapley algorithm for estimating sample importance.

```

1 TMC-Shapley( $D, \mathcal{A}, V$ )
2   Initialize  $\phi_i \leftarrow 0$  for  $i = 1, \dots, n$ 
3    $t \leftarrow 0$ 
4   while convergence criteria not met do
5      $t \leftarrow t + 1$ 
6      $\pi^t$  Random permutation of train data
7      $v_0^t \leftarrow V(\emptyset, \mathcal{A})$ 
8     for  $j = 1$  to  $n$  do
9       if  $|V(D) - v_{j-1}^t| < \textit{tolerance}$  then
10         $v_j^t \leftarrow v_{j-1}^t$ 
11       else
12         $v_j^t \leftarrow V(\{\pi^t[1], \dots, \pi^t[j]\}, \mathcal{A})$ 
13         $\phi_{\pi^t[j]} \leftarrow \frac{t-1}{t} \phi_{\pi^{t-1}[j]} + \frac{1}{t} (v_j^t - v_{j-1}^t)$ 
14   return  $\phi$ 

```

3.5.2. Gradient Shapley

In scenarios involving machine learning models with a high number of parameters and large datasets, even TMC-Shapley computations may become prohibitively expensive. Thus, a gradient-based approximation called G-Shapley, as shown in Algorithm 2, can be preferred. Instead of training a model multiple times, G-Shapley estimates the

marginal contribution of a sample using single-step gradient updates during a simulated one-pass training.

$$\theta_j = \theta_{j-1} - \alpha \nabla_{\theta} L(x_{\pi[j]}, y_{\pi[j]}; \theta_{j-1}), \quad (3.6)$$

Let θ_0 be the initial parameters of the model before the training process. For each permutation π of the training samples, the parameters are updated iteratively as defined by Equation 3.6, where L is the loss function, α is the learning rate, and θ_j denotes the model parameters after the j -th update.

$$\phi_{\pi[j]} = V(\theta_j) - V(\theta_{j-1}). \quad (3.7)$$

The marginal contribution of sample $\pi[j]$ is computed by Equation 3.7. These contributions are averaged over multiple permutations to obtain the final SV estimates. Despite its simplifications, G-Shapley has been shown to produce meaningful data valuations with substantially lower runtime, especially in large-scale settings.

Algorithm 2: The G-Shapley algorithm for estimating sample importance.

```

1 GradientShapley( $\mathcal{L}, D, V$ )
2   Initialize  $\phi_i \leftarrow 0$  for  $i = 1, \dots, n$ 
3    $t \leftarrow 0$ 
4   while convergence criteria not met do
5      $t \leftarrow t + 1$ 
6      $\pi^t$    Random permutation of train data
7      $\theta_0^t$  Random initial parameters
8      $v_0^t = V(\theta_0^t)$ 
9     for  $j = 1$  to  $n$  do
10       $\theta_j^t \leftarrow \theta_{j-1}^t - \alpha \nabla_{\theta} \mathcal{L}(\pi^t[j]; \theta_{j-1}^t)$ 
11       $v_j^t = V(\theta_j^t)$ 
12       $\phi_{\pi^t[j]} \leftarrow \frac{t-1}{t} \phi_{\pi^{t-1}[j]} + \frac{1}{t} (v_j^t - v_{j-1}^t)$ 
13   return  $\phi$ 

```

3.5.3. Leave-One-Out

The Leave-One-Out (LOO) strategy assesses the impact of individual training samples by measuring the change in model performance, such as accuracy or F1 scores, when

a specific sample is excluded from the dataset. The LOO value for the i -th sample is defined in Equation 3.8 [79].

$$\phi_i^{\text{LOO}} = V(D) - V(D \setminus \{(x_i, y_i)\}) \quad (3.8)$$

Although LOO is computationally more efficient than exact SV, it may suffer from lower accuracy and sensitivity to label noise. In this thesis, several LOO-based extensions are proposed to improve its robustness.

$$\delta^i = V_c(D) - V_c(D \setminus \{i\}) = [\delta_1^i, \delta_2^i, \dots, \delta_k^i] \quad (3.9)$$

Let $V_c(D) = [\lambda_1(D), \lambda_2(D), \dots, \lambda_k(D)]$ be the vector of class-wise accuracies when the model is evaluated on k classes. Then for each sample i , the change vector can be defined by Equation 3.9. Using this vector, two variants of LOO are constructed:

- *LOO-Sum*: Aggregates the total change in class-wise accuracy after removing sample i :

$$\phi_i^{\text{Sum}} = \sum_{j=1}^k \delta_j^i \quad (3.10)$$

- *LOO-StdDev*: Introduces class-wise weights based on the standard deviation σ_j across all training samples, giving more importance to unstable classes:

$$\phi_i^{\text{StdDev}} = \sum_{j=1}^k \delta_j^i \cdot \sigma_j \quad (3.11)$$

3.6. Adaptive Thresholding

Motivated by FedBalancer [16], an adaptive threshold approach is developed for loss-based sample selection. Specifically, the sample selection algorithm of FedBalancer is employed, with the deadline control mechanism removed. To compute the loss values for each sample, the overall loss function is employed, which combines image reconstruction and classification losses into a weighted sum, with loss weights $\lambda_{\text{rec}} = 1$ and $\lambda_{\text{cls}} = 0.05$. The sample selection process (*SelectSample*) is described in Algorithm 3. Here, D_i is the local dataset of client i and loss_i contains the loss values of local samples. During the R -th communication round, client i partitions its local samples into two sets, Under-Threshold (UT_R^i) and Over-Threshold (OT_R^i), based on the current threshold lt_R .

The samples in OT_R^i are considered outliers or important samples, which are selected randomly (Line 24), with the selection amount determined by $p \in [0.0, 1.0]$. In contrast, the samples in UT_R^i are presumed to be normal and included in the training.

Algorithm 3: Adaptive thresholding algorithm for server and clients.

```

1 Server:
2 CalculateLt( $LLow_R, LHigh_R, ltr$ )
3    $ll = \min(LLow_R)$ 
4    $lh = \text{mean}(LHigh_R)$ 
5    $lt_{R+1} = ll + (lh - ll) \cdot ltr$ 
6   return  $lt_{R+1}$ 
7 ControlLtr( $U, LossSum_R, L_R, ltr, lss, t_w, R$ )
8    $U_R = LossSum_R / L_R$ 
9    $U.\text{add}(U_R)$ 
10  if  $R \bmod t_w \equiv 0$  then
11    if  $\sum U(R - 2t_w : R - t_w) > \sum U(R - t_w : R)$  then
12       $ltr = \min(ltr + lss, 1)$ 
13    else
14       $ltr = \max(ltr - lss, 0)$ 
15  return  $ltr$ 
16 Client:
17 SelectSample( $D_i, loss_i, lt_R, p$ )
18    $OT^i, UT^i = []$ 
19   for  $x = 0$  to  $|D_i| - 1$  do
20     if  $loss_i[x] \geq lt_R$  then
21        $OT^i.\text{insert}(D_i[x])$ 
22     else
23        $UT^i.\text{insert}(D_i[x])$ 
24   return  $UT^i \cup \text{randSample}(OT^i, p \cdot |OT^i|)$ 

```

3.7. Unsupervised Outlier Detection

In this thesis, unsupervised outlier detection techniques, specifically OCSVM [80] and IF [81], are employed to identify anomalous samples either in a two-dimensional loss space, constructed using weighted CE and MSE losses, or within the feature space. The proposed sample selection approach using OCSVM or IF is detailed in Algorithms 4 and

5 for server and clients, respectively. Here, T denotes the total rounds, P is the number of clients selected per round, $\mathcal{N}$ denotes the entire client set, $\mathcal{P}_t$ indicates the subset of clients selected at round t , E is the number of local epochs, B is the batch size, η is the learning rate, $\mathcal{M}$ is the outlier detection model, t_s specifies the round after which sample selection begins, and t_w is the interval at which the outlier detection model is updated.

Algorithm 4: Server-side algorithm for sample selection and quantization in FL.

```

1 Server( $T, P, \mathcal{N}, E, B, \eta, t_s, t_w, b_{max}, b_{min}$ ):
2   initialize global model parameters  $\theta_0$ 
3   initialize outlier detection model  $\mathcal{M}$ 
4   for  $t = 0$  to  $T - 1$  do
5     select  $P$  clients randomly as  $\mathcal{P}_t \subset \mathcal{N}$ 
6      $b_t = b_{min} + (b_{max} - b_{min}) \frac{1 + \cos(\frac{\pi t}{T-1})}{2}$ 
7      $\hat{\theta}_t, \mathcal{S}_t \leftarrow \text{Quantize}(\theta_t, b_t)$ 
8      $n_{max}^t = \max_{\zeta_i \in \mathcal{P}_t} |D_i|$ 
9      $n = 0, ssTrainSet = \emptyset$ 
10    for each client  $i \in \mathcal{P}_t$  do
11       $\hat{\theta}_t^i, n_i, \kappa_i, \mathcal{S}_t^i = \text{Client}(i, \theta_t, \eta, E, B, \mathcal{M}, t_s, n_{max}^t, \mathcal{S}_t)$ 
12       $\theta_t^i \leftarrow \text{Dequantize}(\hat{\theta}_t^i, \mathcal{S}_t^i)$ 
13       $ssTrainSet = ssTrainSet \cup \zeta_i$ 
14       $n += n_i$ 
15    if  $t \geq t_s$  and  $(t \bmod t_w = 0 \text{ or } t = t_s)$  then
16       $\mathcal{M}.fit(ssTrainSet)$ 
17     $\theta_{t+1} = \sum_{i \in \mathcal{P}_t} \frac{n_i \theta_t^i}{n}$ 
18  return  $\theta_T$ 

```

To reduce the computational burden on client devices, the training of the outlier detection models is performed centrally at the server, where the losses or features from each client are collected to train these models. The updated outlier detection models are then broadcast to clients along with the new global model. Prior to local training, each client computes the relevant loss values (Line 18) and filters out samples identified as outliers by the detection model (Line 20). To improve computational efficiency, the retraining of the outlier detection model is performed every t_w rounds (Line 13) instead

of after each round. A similar procedure is applied in the case of feature-based sample selection, where sample embeddings (Line 18) are used rather than loss values, and outlier detection is performed within the feature space.

Algorithm 5: Client-side algorithm for sample selection, quantization, and DP.

```

1 Client( $i, \hat{\theta}_t, \eta, E, B, \mathcal{M}, t_s, n_{max}^t, \mathcal{S}_t$ ):
2    $\kappa_i = []$ 
3    $\theta_t^i \leftarrow \text{Dequantize}(\hat{\theta}_t, \mathcal{S}_t)$ 
4   if  $t \geq t_s$  then
5      $\kappa_i = \text{ExtractLossesOrFeatures}(D_i, \theta_t^i)$ 
6   if  $t > t_s$  then
7      $D_i = \text{RemoveOutlierSamples}(D_i, \mathcal{M}(\kappa_i))$ 
8    $n_i = |D_i|$ 
9    $\mathcal{B}_i = \text{split } D_i \text{ into batches of size } B$ 
10  for  $e = 0$  to  $E - 1$  do
11    for  $\text{batch } (x, y) \in \mathcal{B}_i$  do
12       $\theta_t^i \leftarrow \theta_t^i - \eta \nabla f(x; \theta_t^i) / \max(\|\nabla f(x; \theta_t^i)\|_1, \xi)$ 
13      where  $\xi$  is the clipping bound.
14  generate noise for DP:  $w_t^i \sim \text{Lap}\left(0, \frac{T_i}{\epsilon} \Xi_t^i\right)$ 
15   $\theta_t^i \leftarrow \theta_t^i + w_t^i$ 
16  compute client utility score  $\nu_i$  by Eq. (3.19)
17  compute quantization bit-length  $b_t^i$  by Eq. (3.18)
18   $\hat{\theta}_t, \mathcal{S}_t^i \leftarrow \text{Quantize}(\theta_t^i, b_t^i)$ 
19  return  $\theta_t, n_i, \kappa_i, \mathcal{S}_t^i$ 

```

Additionally, the sample selection process is intentionally delayed until a predefined warm-up period (t_s) has elapsed, allowing the model sufficient time to learn from normal samples during the early stages of training. The choice of t_s is especially important as prolonged training rounds make it difficult to distinguish anomalous samples in the CE-MSE loss space due to the risk of overfitting. In addition, in the feature-based selection approach, it is crucial to give the model enough time to learn a meaningful embedding space for separating dissimilar training samples.

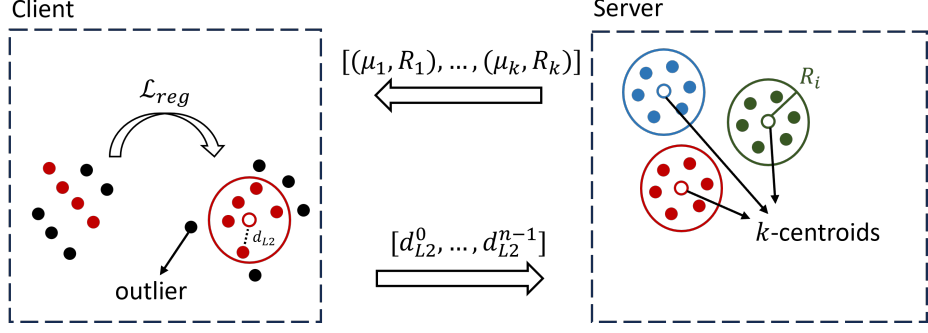


Figure 3.5: The procedure for evaluating the multi-class federated SVDD loss on a client and server, where n denotes the local sample count and d_{L2} represents the Euclidean distance in the embedding space. Best viewed in color.

3.8. Federated Multi-Class SVDD Loss

Let $f : x_{ij} \rightarrow z_{ij}$ and $g : z_{ij} \rightarrow \hat{x}_{ij}$ be the encoder and decoder of an autoencoder for a client i , where z_{ij} is the embedding vector of input x_{ij} in the feature space $\mathcal{F}$, and $\hat{x}_{ij}$ is the reconstructed input x_{ij} . In this thesis, the multi-class federated SVDD loss, formulated in Equation 3.12, is introduced to distinguish outlier samples within the feature space $\mathcal{F}$. The objective is to optimize f such that embeddings $\{z_{ij}\}_{j=1}^{n_i}$ become more suited for outlier detection in multi-class settings. The Deep SVDD loss is integrated as a regularization component into the overall autoencoder loss $\mathcal{L}$, as defined in Equation 3.3. The combined loss is defined as $\mathcal{L}' = \mathcal{L} + \lambda_{\text{reg}} \mathcal{L}_{\text{reg}}$, where λ_{reg} is a hyperparameter that controls the distortion of feature space. The goal is to learn a minimal hypersphere with radius $R_k > 0$ and centroid $\mu_k \in \mathcal{F}$ for a specific class $k \in [1, K]$, where K is the number of classes, n_i is the total samples, n_{ik} is the number of samples in class k , and $\mathbb{I}_{y_{ij}=k}$ is 1 if $y_{ij} = k$ and 0 otherwise. This approach aims to ensure that most normal data points lie within the hypersphere, while anomalous ones are pushed outside its boundary.

$$\mathcal{L}_{\text{reg}} = \frac{1}{K} \sum_{k=1}^K \left(R_k^2 + \frac{1}{n_{ik}} \sum_{j=1}^{n_i} \mathbb{I}_{y_{ij}=k} \max\{0, \|f(x_{ij}) - \mu_k\|^2 - R_k^2\} \right) \quad (3.12)$$

Another key contribution of this thesis is the adaptation of the Deep SVDD Loss [20] to FL, as illustrated in the computation steps in Figure 3.5. In this approach, the server calculates the class centroids $[\mu_1, \dots, \mu_k]$ and corresponding radii $[R_1, \dots, R_k]$ using a publicly available test dataset once the global model reaches a predefined target criterion,

such as a validation accuracy, loss value, or training round. These centroids and radii are then transmitted to the selected clients at the beginning of a training round. On the client side, the proposed regularization loss $\mathcal{L}_{\text{reg}}$ is incorporated into the overall loss function $\mathcal{L}$ during local model updates. Following local training, each client computes the Euclidean distances $[d_{L2}^j]_{j=1}^{n_i}$ between the embeddings of its samples and the class centroids $\mu_{1\dots k}$, and sends these distance values back to the server. The server subsequently updates the radii $R_{1\dots k}$ by computing the q -th quantile of the aggregated distance values, where $q = 1 - \nu$ is a hyperparameter that defines the proportion of data points to be classified as outliers.

3.9. Stochastic Uniform Quantization

To enhance communication efficiency, this thesis uses stochastic uniform quantization, which reduces the precision of model weights by mapping continuous values to a finite set of discrete levels, following approaches in prior studies [46–48]. This method introduces controlled randomness into the quantization process, ensuring that the resulting quantization error is distributed uniformly across the possible value range. Due to the uniform noise, the method mitigates the impact of quantization artifacts, resulting in a more stable and generalizable representation, which is particularly valuable in scenarios requiring both precision and robustness. However, this form of quantization can introduce information loss, which may adversely affect the convergence speed, as noted in [2]. The quantization process consists of two basic functions, outlined as follows:

- **Quantization:** This step transforms a real-valued number into its corresponding discrete integer representation, such as converting from 32-bit floating point (FP32) to 8-bit integer (INT8).
- **Dequantization:** In this step, the quantized integer values are mapped back to approximate real numbers, restoring them to formats like FP32 from INT8.

Let $[\beta, \alpha]$ denote the range of real values that can be represented, and let b be the target bit-width for the signed integer representation. Uniform quantization maps an input value $x \in [\beta, \alpha]$ to the discrete range $[-2^{(b-1)}, 2^{(b-1)} - 1]$, where values falling outside this interval are clipped to the closest boundary. In the uniform quantization, a general form $g(x) = s \cdot x + z$ or a simplified form $g(x) = s \cdot x$ can be preferred, where

$x, s, z \in \mathbb{R}$. These formulations correspond to affine and symmetric transformations, respectively, as described in [82].

$$Q(x, s, b) = \text{clip}(\rho(x * s), b) \quad (3.13)$$

In this thesis, symmetric quantization ($z = 0$) with stochastic rounding is employed, involving a scale-based transformation to map values between the input and integer ranges. Both ranges are symmetric with respect to zero, ensuring a balanced representation of positive and negative values. This type of quantization is defined in Equation (3.13), where ρ is the stochastic rounding operation, s is the scale factor, and clip is the clipping function.

The scale factor s is computed by Equation (3.14) where b is the target bit-length and $\alpha \in \mathbb{R}$ is the maximum real value in the tensor with arbitrary dimension.

$$s = \frac{2^{(b-1)} - 1}{\alpha} \quad (3.14)$$

The clip function $\text{clip}(y, b)$ is defined in Equation (3.15) where $y \in \mathbb{Z}$ denotes an integer value. The stochastic rounding function $\rho : \mathbb{R} \rightarrow \mathbb{Z}$, defined in Equation (3.16), is employed to convert floating-point values into fixed-point or integer representations.

$$\text{clip}(y, b) = \begin{cases} -2^{(b-1)} + 1 & \text{if } y < -2^{(b-1)} + 1 \\ 2^{(b-1)} - 1 & \text{if } y > 2^{(b-1)} - 1 \\ y & \text{otherwise} \end{cases} \quad (3.15)$$

This rounding method probabilistically maps a real number to one of its two nearest integer values, with the selection probability being proportional to the number's distance from those values.

$$\rho(x) = \begin{cases} \lfloor x \rfloor & \text{with probability } p = \lceil x \rceil - x \\ \lceil x \rceil & \text{with probability } 1 - p = x - \lfloor x \rfloor \end{cases} \quad (3.16)$$

The dequantization process, defined in Equation (3.17), restores a real-valued number from its quantized integer representation $x_q \in \mathbb{Z}$. The corresponding scale factor, used in this conversion, is determined during the quantization phase as defined in Equation (3.14).

$$DQ(x_q, s) = \frac{x_q}{s} \quad (3.17)$$

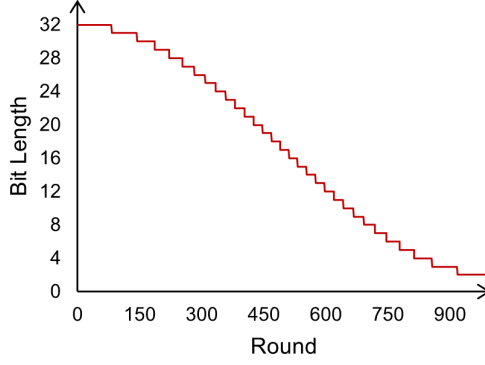


Figure 3.6: Cosine-annealed bit-length scheduling from 32 bits to 2 bits.

Quantization parameters can be shared among multiple tensor elements, a concept referred to as quantization granularity [82]. The highest granularity level assigns distinct quantization parameters to each individual tensor element, which, while precise, becomes computationally intensive for large-scale tensors. In this thesis, a per-tensor granularity approach is assumed, where a single set of quantization parameters is applied uniformly to each three-dimensional tensor within every model layer, balancing efficiency and accuracy.

3.10. Adaptive Quantization

This thesis introduces two straightforward yet effective adaptive quantization strategies that dynamically change quantization precision during training. The first approach employs a cosine annealing schedule to progressively reduce the bit length used in quantization. This enables the model to start training with a higher precision level (b_{max}) and gradually transition to a lower precision level (b_{min}) as training continues. This smooth reduction in bit length mitigates potential accuracy degradation caused by quantization error, enabling a more stable convergence. This descending scheduling approach is suggested by Qu et al. [48] for the subsequent reasons:

- Higher bit-lengths in the initial training stages accelerate loss minimization, while lower precisions, though communication-efficient, tend to slow convergence.
- As the model stabilizes in later rounds, higher precisions are not needed, making a gradual reduction in bit-length more suitable for FL.

The bit-length scheduling for a client i in the t -th round is formulated in Equation (3.18), where T is the total communication rounds, v_i is the importance score of a client i , b_{max} is the initial bit-length, and b_{min} is the final bit-length during training. Initially, a uniform scheduling policy is applied, where $v_i = 1$ for the server and all clients. The bit-length decreases gradually from b_{max} to b_{min} according to the cosine annealing curve, as illustrated in Figure 3.6.

$$b_t^i = b_{min} + v_i(b_{max} - b_{min}) \frac{1 + \cos\left(\frac{\pi t}{T}\right)}{2} \quad (3.18)$$

In addition to the uniform scheduling, this thesis proposes a hybrid strategy that combines cosine annealing with client-based Shannon entropy, as formulated in Equation (3.18). This strategy uses the cosine annealing schedule to control the overall bit-length reduction, while per-client adjustments based on dataset entropy are made to ensure adaptation to each client's data distribution and characteristics. Specifically, clients with larger and more homogeneous datasets are assigned higher bit lengths to speed up convergence and maintain accuracy. The importance score v_i incorporates two factors: dataset homogeneity and relative dataset size, as defined in Equation (3.19). Here, p_k^i denotes the probability of a class k , $\lambda_h \in [0, 1]$ denotes the weight of dataset homogeneity score, and n_{max}^t is the number of samples of the client with the most samples in the t -th round, calculated as $n_{max}^t = \max_{i \in P_t} |D_i|$.

$$v_i = \lambda_h \frac{-\sum_{k=1}^K p_k^i \log_2(p_k^i)}{\log_2(K)} + (1 - \lambda_h) \frac{|D_i|}{n_{max}^t} \quad (3.19)$$

The hyperparameter λ_h determines how much weight is given to dataset homogeneity versus dataset size when computing client importance. By combining these factors, the proposed scheduling approach efficiently balances model accuracy and computational efficiency, which is especially beneficial in non-IID FL settings.

3.11. Differential Privacy

Although a key benefit of FL is its capability to facilitate localized training without requiring the exchange of personal data between the server and clients, adversaries may still uncover sensitive information to some degree by analyzing the distinctions in the model weights or gradients uploaded by the clients [83–85]. DP methods add

artificial noise to prevent information leakage, but increased noise leads to enhanced privacy protection at the cost of reduced accuracy. To avoid information leakage, each client should locally perturb its trained parameters by deliberately introducing noise before uploading them for aggregation at the server. Moreover, increasing the number of participating clients in FL can enhance convergence performance while maintaining a fixed level of privacy protection.

A randomized function $M : X \rightarrow R$, mapping input domain X to output range R , satisfies (ϵ, δ) -differential privacy if, for any pair of neighboring datasets $D, D' \in X$ differing by one record, and for any subset $S \subseteq R$, the inequality in Equation (3.20) holds. In this context, $\epsilon > 0$ is the privacy budget, balancing privacy strength against model utility, whereas $\delta \geq 0$ is the probability of a privacy breach. Higher ϵ values weaken privacy guarantees by increasing distinguishability between neighboring datasets.

$$\Pr[M(D) \in S] \leq e^\epsilon \Pr[M(D') \in S] + \delta \quad (3.20)$$

Within FL frameworks, there are two ways of applying DP: Local DP, where clients apply noise before communication, and Central DP, where noise is added post-aggregation on the server. This thesis uses a Local DP approach with $\delta = 0$ during training, ensuring that client data cannot be predicted even if server-side exposures exist.

$$\Xi = \max_{\forall \theta, D': \|D-D'\|_1=1} \|\Phi(\theta, D) - \Phi(\theta, D')\|_1 \quad (3.21)$$

During local training, gradient clipping is applied to bound the l_1 -norm of the gradients of the loss function ℓ , which plays a crucial role in sensitivity analysis, as explained in [61]. Given the statistical heterogeneity of client datasets resulting from non-IID distributions, the L_1 -sensitivity of client ζ_i in the t -th round, denoted by Ξ_t^i , is computed using Equation (3.22), where ξ is the l_1 -norm gradient clipping, η is the learning rate, E is the local epochs, $n_i = |D_i|$ is the total samples, and E_0 is the minimum integer such that $(1 + \lambda_i \eta)^{E_0} \geq 1 + n_i$.

$$\Xi_t^i = \begin{cases} \frac{2\xi E \eta}{n_i}, & \text{if } \lambda_i = 0, \\ \frac{2\xi}{\lambda_i n_i} ((1 + \lambda_i \eta)^E - 1), & \text{if } \lambda_i > 0, E < E_0, \\ 2\xi + 2\eta\xi(E - E_0), & \text{if } \lambda_i > 0, E \geq E_0. \end{cases} \quad (3.22)$$

After local training on a client ζ_i , a systematic noise w_t^i is generated based on the sensitivity function Ξ_t^i and this noise is added to the model weights θ_t^i before communicating

with the server. The noise is defined by $w_t^i = \text{Lap}(0, \frac{T_i}{\epsilon} \Xi_t^i)$, where $\text{Lap}(\mu, s)$ denotes the Laplace distribution with mean μ and scale s of the same dimension as w_t^i . The sensitivity is scaled by $T_i = \frac{PT}{NE}$, where P is the number of selected clients in round t , T is the total communication rounds, and N is the total clients.

To determine the appropriate noise magnitude for each client, the sensitivity function Ξ_t^i is used to maintain the balance between privacy and learning accuracy [61]. The sensitivity is affected by the local dataset size and a regularization parameter λ_i . This parameter denotes the local Lipschitz smoothness of the loss function ℓ on client ζ_i , effectively constraining gradient magnitudes. This smoothness constraint is crucial for controlling the sensitivity of the local updates, thereby enabling smaller noise magnitudes needed by the DP while maintaining model utility and preserving privacy.

The local Lipschitz smoothness λ_i is computed using Equation (3.23), which is based on l_1 -norm to be consistent with the L_1 -sensitivity of Laplace noises. Here, $\nabla f(\theta)$ and $\nabla f(\theta')$ represent the gradients of the model before and after local training, respectively. In this context, the learning model is f , while θ and θ' represent the initial and updated model parameters before and after local training, respectively.

$$\lambda_i = \frac{\|\nabla f(\theta) - \nabla f(\theta')\|_1}{\|\theta - \theta'\|_1} \quad (3.23)$$

As mini-batch training is performed over several local epochs, the variations in gradients and model parameters are observed across these epochs for each client. Based on these tracked changes, λ_i values are calculated for each batch, with the maximum value selected as the final λ_i used in sensitivity calculations.

4. EXPERIMENTS AND RESULTS

4.1. Experimental Setup

For the experimental evaluations in this thesis, the FedML framework is used due to its capability of simulating FL environments on a single machine. Additionally, it offers a flexible programming interface with standard implementations for optimizers, models, and datasets [86]. Within this framework, the proposed algorithms described in Algorithms 4 and 5 are implemented, integrating the proposed sample selection, Laplacian-based DP, and adaptive quantization methods, along with the models discussed in Sections 3.3 and 3.4.

This thesis uses the FedAvg algorithm to combine local models collected from participating clients at each round. The FL trainings are performed over a total of 1000 communication rounds across different numbers of clients. To evaluate the global model on the central server, training and testing accuracies are computed every 10 rounds to save computational cost. As the testing accuracy fluctuates due to non-IID distribution across clients during FL training, the best-performing rounds are tracked and recorded based on global testing accuracy for fairer comparisons. As discussed in Section 2.1, the clients use the SGD optimizer with a learning rate of 0.1 and a weight decay of 0.001. Each client trains locally for five epochs per round with a batch size of 64.

The experiments are performed by using a total of 1000, 200, 100, and 50 clients, with 100, 20, 10, and 5 clients selected per round, respectively. A fixed selection ratio of 0.1 is maintained across all experiments to ensure consistency. Besides, the same random seed is employed across all runs to ensure reproducibility and fair comparisons.

The objective function of MTAE, defined in Equation 3.3, is calculated using mean squared error and cross-entropy, with loss weights set at $\lambda_{rec} = 1$ and $\lambda_{cls} = 0.05$. The weight assigned to CE loss is intentionally kept lower than that of MSE to avoid blurry generated images and make sure that the models capture fine-grained details. Every 10 training rounds, train and test accuracies for the classification results are evaluated, along with the Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) metrics for the reconstructed images.

4.2. Sample Selection Baseline

In this section, the proposed MTAE models are trained on the MNIST and CIFAR10 datasets across various noise settings and client counts without using the proposed sample selection methods to create a basis for future experiments. The main objective of sample selection is to improve global test accuracy, which is calculated on the server, by using sample selection techniques in the presence of noisy samples.

Table 4.1: The highest performance achieved by the MTAE models trained on the MNIST dataset without applying noise and sample selection across different client counts.

Total Clients	Accuracy	PSNR	SSIM	Best Round
1000	94.60	21.99	92.55	920
200	96.62	22.62	94.23	990
100	97.34	23.40	94.88	970
50	97.49	24.09	95.86	970

Table 4.2: The highest performance achieved by the MTAE models trained on the CIFAR10 dataset without applying noise and sample selection across various client counts.

Total Clients	Accuracy	PSNR	SSIM	Best Round
1000	57.95	18.54	43.92	960
200	66.81	20.15	57.60	999
100	69.75	20.87	62.08	970
50	71.05	21.55	66.26	900

The proposed MTAE models are initially trained with diverse client numbers, without introducing both noise perturbation and sample selection procedures. The experimental results in Tables 4.1 and 4.2 demonstrate that the test accuracy increases when the client count is reduced. This is expected because the training datasets, CIFAR10 or MNIST,

Table 4.3: The highest performance achieved by the MTAE models trained on the MNIST dataset for various client counts and types of 40% noise without sample selection.

Noise Type	Total Clients	Accuracy	PSNR	SSIM	Best Round
Closed-Set	1000	90.14	21.38	91.56	640
	200	92.28	21.42	91.86	470
	100	88.36	22.40	92.84	460
	50	85.58	21.96	92.69	260
EMNIST	1000	91.43	21.12	91.45	920
	200	95.22	22.69	93.69	960
	100	94.48	21.95	92.77	990
	50	94.53	23.09	94.78	990
ImageNet32	1000	93.27	21.00	89.39	980
	200	95.60	22.46	92.33	970
	100	95.14	21.58	91.14	850
	50	95.13	23.02	93.95	990

are distributed to a lower number of clients, thereby increasing the local data size of clients. This improves the global model in terms of accuracy and image quality.

In the following experiments, the proposed MTAE models are trained on the CIFAR10 and MNIST datasets with 40% open-set and closed-set noise across various numbers of clients. According to the results in Tables 4.3 and 4.4, accuracy in closed-set noise experiments for the MNIST and CIFAR10 improves as client numbers decrease, but eventually declines after a certain threshold is reached. This threshold appears to be at 200 clients for the MNIST and 100 clients for the CIFAR10. The accuracy drop beyond these thresholds is mainly a result of the higher number of noisy labels on clients.

In the MNIST experiments, it is observed that closed-set noise reduces accuracy to 85.58% for 50 clients, the lowest value in all settings. Comparing the EMNIST and ImageNet32 noises, there is a higher drop in accuracy when the EMNIST is used for open-set noise. This is because of the similarity between the EMNIST and MNIST datasets. For example, the letter O resembles the digit zero. In addition, the inclusion

Table 4.4: The highest performance achieved by the MTAE models trained on the CIFAR10 dataset, including various client counts and types of 40% noise without sample selection.

Noise Type	Total Clients	Accuracy	PSNR	SSIM	Best Round
Closed-Set	1000	39.74	18.04	40.77	820
	200	40.32	19.78	55.04	980
	100	41.36	20.29	59.71	890
	50	38.59	20.26	59.47	550
SVHN	1000	50.99	18.16	41.21	970
	200	57.46	19.54	53.30	970
	100	60.86	20.13	58.23	930
	50	60.58	20.49	60.91	900
ImageNet32	1000	47.47	18.52	43.67	990
	200	52.68	19.84	55.04	890
	100	55.25	20.43	60.39	890
	50	57.53	20.90	63.06	900

of open-set noise increases the PSNR and SSIM scores for 50 clients. This shows that increased size and diversity of local samples are beneficial for autoencoders in FL.

In the CIFAR10 experiments, the largest drop in accuracy is observed under closed-set noise. This shows that even if the classification loss weight λ_{cls} is much lower than the reconstruction loss weight λ_{rec} , label noise reduces both accuracy and quality of reconstructed images. Considering the results with open-set noise, ImageNet32 reduces accuracy more than SVHN, regardless of the client count. This is because of the resemblance of images in the CIFAR10 and ImageNet32 datasets. In addition, the lowest accuracy is 38.59% for 50 clients under closed-set noise. As the number of clients is reduced, the accuracy and image quality metrics, particularly for ImageNet32, show improvement. However, for SVHN and closed-set noise, accuracy starts to drop when there are fewer than 100 clients. This is likely due to the increased occurrence of noisy samples among clients.

4.3. Loss-based Sample Selection

As detailed in Sections 3.6 and 3.7, this thesis introduces novel loss-based sample selection methods. These methods are based on AT, OCSVM, and IF algorithms, which are specifically designed for FL. To evaluate these methods, various types of 40% noise are injected into the training datasets to better observe the utility of sample selection. Since the loss distribution changes during training, these methods are retrained on the server every five rounds. The server collects the weighted losses from each client to adapt to the new loss distribution for more effective sample selection. Careful selection of the update interval is important, as the high computational cost of training OCSVM and IF can substantially extend the overall training time. If the update period is excessively prolonged, the sample selection methods may struggle to adjust to the loss distribution of the current FL model, which could result in the inclusion of noisy samples.

Extensive experiments are conducted on the MNIST and CIFAR10 datasets to tune the loss coefficients λ_{rec} and λ_{cls} . The configuration $\lambda_{rec} = 1$ and $\lambda_{cls} = 0.05$ are selected to balance the trade-off between the reconstruction and classification tasks. Empirical results show that placing more emphasis on reconstruction improves visual quality and helps the model identify noisy inputs more effectively. Conversely, assigning a smaller weight to the classification term helps mitigate overfitting while preserving high PSNR and SSIM scores. The contamination parameters of the OCSVM and IF algorithms are crucial for effective outlier detection. Thus, these parameters are set to 0.4 for both algorithms as a noise rate of 40% is used in the experiments. In addition, the number of estimators in the IF algorithm is selected based on the square root of the data size. This heuristic allows the IF to capture sufficient variability in the dataset while avoiding unnecessary computational cost. The Radial Basis Function (RBF) kernel is chosen due to its proven effectiveness in modeling complex, non-linear relationships. This improves the model’s ability to distinguish outlier samples from regular ones in high-dimensional feature representations.

The proposed AT method, detailed in Section 3.6, has three parameters: loss step size l_{ss} , update period t_w , and a probability p that can be tuned for target tasks. Inspired by FedBalancer [16], these parameters are determined through extensive experiments on the MNIST and CIFAR10 datasets. As a result, a loss step size of $l_{ss} = 0.1$, an update

Table 4.5: The highest performance achieved by the MTAE models trained on the MNIST dataset, applying 40% noise and sample selection under closed-set noise.

Total Clients	Method	Accuracy (Gain)	PSNR	SSIM	Best Round
1000	AT	91.75 (+1.61)	20.88	90.21	570
	OCSVM	91.77 (+1.63)	21.29	91.36	780
	IF	91.83 (+1.69)	21.27	91.37	770
200	AT	93.85 (+1.57)	21.13	91.21	510
	OCSVM	<u>91.56</u> (−0.72)	21.60	92.32	700
	IF	92.73 (+0.45)	21.62	92.36	770
100	AT	90.19 (+1.83)	22.38	92.77	550
	OCSVM	88.71 (+0.35)	22.53	93.05	660
	IF	89.82 (+1.46)	22.50	93.00	660
50	AT	85.58 (+0.00)	21.96	92.69	260
	OCSVM	85.58 (+0.00)	21.96	92.69	260
	IF	85.58 (+0.00)	21.96	92.69	260

period of $t_w = 5$, and a probability of $p = 0.75$ are selected, which are appropriate for gradual threshold modifications and adapt to the new loss distribution. The update period $t_w = 5$ is also consistent with the update intervals of both OCSVM and IF models. The probability $p = 0.75$ is used to preserve high-loss samples, which may be either useful or contain noise. Given that the AT method relies on a single loss function, the final loss represents the weighted combination of both CE and MSE losses.

In the early stages of training, the model has not yet learned to effectively differentiate between informative and noisy samples. As a result, both clean and outlier instances tend to produce similar gradient norms [9]. Based on this observation, sample selection is delayed and applied after the 400th round during training. This is the point at which accuracy improvements start to plateau, indicating model stabilization. Choosing the right starting round is an important hyperparameter that impacts the effectiveness of the sample selection. Due to the underfitting encountered in early rounds, the model tends to produce high losses. In contrast, later rounds encounter overfitting, where noisy

Table 4.6: The highest performance achieved by the MTAE models trained on the MNIST dataset, applying 40% noise and sample selection for open-set noise type EMNIST.

Total Clients	Method	Accuracy (Gain)	PSNR	SSIM	Best Round
1000	AT	91.95 (+0.52)	20.91	90.95	880
	OCSVM	92.06 (+0.63)	20.71	90.19	930
	IF	<u>89.26</u> (−2.17)	20.62	89.97	880
200	AT	<u>94.99</u> (−0.23)	21.83	92.54	990
	OCSVM	<u>94.03</u> (−1.19)	21.79	92.42	990
	IF	<u>92.55</u> (−2.67)	21.82	92.50	990
100	AT	95.13 (+0.65)	22.49	93.38	960
	OCSVM	95.06 (+0.58)	22.35	93.34	940
	IF	<u>93.58</u> (−0.90)	22.52	93.59	999
50	AT	95.03 (+0.50)	22.77	94.26	860
	OCSVM	95.65 (+1.12)	22.87	94.43	950
	IF	<u>93.49</u> (−1.04)	22.50	93.87	610

samples are memorized, thus producing lower losses. Both underfitting and overfitting reduce the distinction in losses, making it harder to differentiate between valuable and noisy samples.

In the following sample selection experiments, underlined results indicate that using a corresponding sample selection method degrades model accuracy compared to the reference results, which are presented in Tables 4.3 and 4.4 for the MNIST and CIFAR10 datasets, respectively.

The accuracy improvements achieved by the proposed sample selection methods on the MNIST dataset with closed-set noise are presented in Table 4.5. The results show that the highest accuracy improvement, which is 1.83%, is observed for 100 clients when AT is employed, with a minor drop in PSNR and SSIM scores. Except for 50 clients, IF also improves accuracy up to 1.69%, which is observed for 1000 clients. Except for 50 and 200 clients, OCSVM improves accuracy up to 1.63%, which is achieved for 1000

Table 4.7: The highest performance of MTAE models trained on the MNIST dataset with 40% open-set ImageNet32 noise and sample selection.

Total Clients	Method	Accuracy (Gain)	PSNR	SSIM	Best Round
1000	AT	93.31 (+0.04)	20.77	88.65	980
	OCSVM	93.47 (+0.20)	20.61	88.74	930
	IF	<u>92.38</u> (−0.89)	20.83	89.46	950
200	AT	<u>95.32</u> (−0.28)	21.72	91.02	990
	OCSVM	<u>95.35</u> (−0.25)	21.72	91.67	990
	IF	<u>94.61</u> (−0.99)	21.91	92.20	990
100	AT	95.70 (+0.56)	22.18	91.36	960
	OCSVM	95.97 (+0.83)	22.36	92.59	940
	IF	95.27 (+0.13)	22.56	93.20	999
50	AT	95.54 (+0.41)	22.79	93.57	860
	OCSVM	96.04 (+0.91)	22.96	93.90	950
	IF	95.79 (+0.66)	23.34	94.59	970

clients. However, OCSVM causes a 0.72% decline in accuracy for 200 clients. This is probably due to the RBF kernel’s tendency to overfit, which can misclassify outliers.

The accuracy improvements achieved by the proposed sample selection methods on the MNIST dataset with EMNIST open-set noise are presented in Table 4.6. The results show that OCSVM has higher improvements than the others for 50 and 1000 clients, with the highest improvement of 1.12% that is observed for 50 clients. In contrast, no accuracy improvements are recorded for IF. This is probably due to the similarity of the MNIST and EMNIST datasets, which makes it harder to distinguish between noisy and regular samples.

The accuracy improvements achieved by the proposed sample selection methods on the MNIST dataset with ImageNet32 open-set noise are presented in Table 4.7. The results demonstrate that OCSVM has the highest improvements across all settings, except for 200 clients, with the highest improvement of 0.91% observed for 50 clients. Similarly, AT shows improvements for 50, 100, and 1000 clients, but is less effective than OCSVM.

Table 4.8: The highest performance of MTAE models trained on the CIFAR10 dataset with 40% closed-set noise and sample selection.

Total Clients	Method	Accuracy (Gain)	PSNR	SSIM	Best Round
1000	AT	40.94 (+1.20)	18.14	41.83	830
	OCSVM	44.94 (+5.20)	18.33	43.17	999
	IF	45.39 (+5.65)	18.29	43.12	999
200	AT	<u>39.87</u> (−0.45)	19.56	53.56	880
	OCSVM	46.76 (+6.44)	19.48	52.76	940
	IF	46.17 (+5.85)	19.40	52.24	920
100	AT	<u>40.48</u> (−0.88)	20.26	59.26	890
	OCSVM	46.70 (+5.34)	20.03	57.61	930
	IF	45.01 (+3.65)	19.92	57.06	990
50	AT	38.60 (+0.01)	20.52	61.64	890
	OCSVM	45.61 (+7.02)	20.43	60.85	890
	IF	45.24 (+6.65)	20.42	60.47	890

In contrast, IF yields no gains for 1000 and 200 clients, but shows improvements for 50 and 100 clients. The highest improvement for IF is 0.66% recorded for 50 clients. All methods fail to improve accuracy for 200 clients. This is probably because of the late start of sample selection, which results in overfitting on noisy samples and inaccurate outlier classifications.

The accuracy improvements achieved by the proposed sample selection methods on the CIFAR10 dataset with closed-set noise are presented in Table 4.8. The results indicate that OCSVM and IF methods consistently improve accuracy across all settings. The highest improvement, 7.02%, is observed for 50 clients when OCSVM is employed. Similarly, IF has the highest improvement of 6.65% for 50 clients. In contrast, AT yields accuracy drops for 100 and 200 clients, with decent image quality. Compared to the MNIST experiments, the results show that OCSVM and IF are more effective in enhancing model accuracy on more complex datasets like CIFAR10.

The accuracy improvements achieved by the proposed sample selection methods on

Table 4.9: The highest performance of the MTAE models trained on the CIFAR10 dataset with 40% open-set SVHN noise and sample selection.

Total Clients	Method	Accuracy (Gain)	PSNR	SSIM	Best Round
1000	AT	<u>49.95</u> (−1.04)	18.16	41.17	990
	OCSVM	51.40 (+0.41)	18.02	40.83	960
	IF	<u>50.84</u> (−0.15)	17.88	40.08	940
200	AT	<u>56.84</u> (−0.62)	19.47	52.70	999
	OCSVM	<u>57.44</u> (−0.02)	19.14	50.58	999
	IF	<u>56.82</u> (−0.64)	19.05	49.91	980
100	AT	61.22 (+0.36)	20.03	57.66	930
	OCSVM	60.86 (+0.00)	19.76	56.00	990
	IF	<u>60.34</u> (−0.52)	19.70	55.56	990
50	AT	61.14 (+0.56)	20.49	61.19	880
	OCSVM	61.94 (+1.36)	20.21	59.01	890
	IF	61.15 (+0.57)	20.33	60.54	990

the CIFAR10 dataset with SVHN noise are presented in Table 4.9. The findings show that OCSVM is beneficial for 50 and 1000 clients, but shows no improvements for 100 and 200 clients. Considering all results, however, the highest improvement, 1.36%, is achieved by OCSVM with 50 clients. Similarly, AT improves accuracy only for 50 and 100 clients, while also maintaining higher image quality compared to the other methods. For IF, the only improvement, 0.57%, is observed for 50 clients. These limited improvements are likely due to overfitting, as SVHN is simpler than CIFAR10, making it more difficult to distinguish informative samples within the loss space.

The accuracy improvements achieved by the proposed sample selection methods on the CIFAR10 dataset with ImageNet32 noise are presented in Table 4.10. In contrast to the experiments where SVHN is used as the open-set noise, both OCSVM and IF yield higher accuracy improvements across all settings. The highest improvement, 3.61%, is observed for 200 clients when IF is employed. However, the effectiveness of AT, based on a single loss value, is found to be more limited compared to other methods

Table 4.10: The highest performance achieved by the MTAE models trained on the CIFAR10 dataset, applying 40% noise and sample selection under open-set noise ImageNet32.

Total Clients	Method	Accuracy (Gain)	PSNR	SSIM	Best Round
1000	AT	<u>47.00</u> (−0.47)	18.43	42.88	960
	OCSVM	50.89 (+3.42)	18.43	43.27	999
	IF	50.66 (+3.19)	18.38	43.08	999
200	AT	<u>52.56</u> (−0.12)	19.90	55.69	999
	OCSVM	55.99 (+3.31)	19.70	54.00	999
	IF	56.29 (+3.61)	19.64	53.64	999
100	AT	<u>53.37</u> (−1.88)	20.24	59.02	890
	OCSVM	57.79 (+2.54)	20.14	58.04	930
	IF	58.72 (+3.47)	20.20	58.58	970
50	AT	58.09 (+0.56)	21.08	63.49	940
	OCSVM	59.04 (+1.51)	20.62	61.75	890
	IF	59.02 (+1.49)	20.63	62.28	900

using two different losses for outlier detection. This limitation is more pronounced on complex datasets such as CIFAR10, as opposed to simpler ones like MNIST. Moreover, the non-IID nature of the data across clients leads to fluctuations in loss values and complicates the use of a single global threshold to identify noisy samples.

For a more fine-grained evaluation of classification performance across 1000 clients, the proposed sample selection methods are further assessed using additional metrics, including macro-averaged precision, recall, and F1-score. The experimental results are provided in Tables 4.11 and 4.12. Macro-averaging means that each class is independently evaluated and then the results are averaged, ensuring equal weight is given to all classes regardless of their sample size. The results reveal that OCSVM delivers superior performance compared to the other approaches, especially on the CIFAR10 dataset, where it shows notable gains across precision, recall, and F1-score. For example, under closed-set noise, OCSVM improves precision by 5.55% and F1-score by 5.64% relative to the baseline. While AT yields strong results on the MNIST,

Table 4.11: The highest performance achieved by the MTAE models trained on the MNIST dataset with 40% noise and 1000 clients, using macro-averaged metrics.

Noise Type	Method	Prec. (Gain)	Recall (Gain)	F1 (Gain)	Best R.
Closed-Set	–	90.23	89.98	89.98	640
	AT	91.86 (+1.63)	91.62 (+1.64)	91.67 (+1.69)	570
	OCSVM	91.88 (+1.65)	91.70 (+1.72)	91.66 (+1.68)	780
	IF	91.85 (+1.62)	91.77 (+1.79)	91.74 (+1.76)	770
EMNIST	–	91.73	91.28	91.36	920
	AT	92.14 (+0.41)	91.93 (+0.65)	91.92 (+0.56)	880
	OCSVM	92.06 (+0.33)	91.99 (+0.71)	92.01 (+0.65)	930
	IF	<u>89.36</u> (−2.37)	<u>89.20</u> (−2.08)	<u>89.20</u> (−2.16)	880
ImageNet32	–	93.43	93.19	93.23	980
	AT	93.50 (+0.07)	93.22 (+0.03)	93.27 (+0.04)	980
	OCSVM	93.44 (+0.01)	93.45 (+0.26)	93.43 (+0.20)	930
	IF	<u>92.41</u> (−1.02)	<u>92.30</u> (−0.89)	<u>92.32</u> (−0.91)	950

its performance reduces dramatically on the CIFAR10 when SVHN is used as open-set noise, at times falling below baseline levels. Conversely, IF achieves promising outcomes on the CIFAR10 but experiences substantial degradation on the MNIST with EMNIST noise, where all three metrics decrease by more than 2%. Overall, OCSVM appears to be the most robust method across various datasets and noise types. These findings are also consistent with the accuracy-based evaluations.

4.4. Feature-based Sample Selection

In this section, feature-based sample selection is employed based on the embeddings obtained from MTAE models. The unsupervised outlier detection algorithms explained in Section 3.7 are employed with the same configuration as the loss-based sample selection experiments, trained on the embeddings of local samples by the central server every five rounds. Since closed-set noise is not detectable in the embedding space, only 40% open-set noise is applied during the experiments.

Table 4.12: The highest performance achieved by the MTAE models trained on the CIFAR10 dataset with 40% noise and 1000 clients, using macro-averaged metrics.

Noise Type	Method	Prec. (Gain)	Recall (Gain)	F1 (Gain)	Best R.
Closed-Set	–	40.56	39.74	39.15	820
	AT	40.65 (+0.09)	40.94 (+1.20)	40.41 (+1.26)	830
	OCSVM	46.11 (+5.55)	44.94 (+5.20)	44.79 (+5.64)	999
	IF	46.14 (+5.58)	45.39 (+5.65)	45.00 (+5.85)	999
SVHN	–	50.69	50.99	50.51	970
	AT	<u>49.97</u> (−0.72)	<u>49.95</u> (−1.04)	<u>49.41</u> (−1.10)	990
	OCSVM	50.70 (+0.01)	51.40 (+0.41)	50.61 (+0.10)	960
	IF	50.90 (+0.21)	<u>50.84</u> (−0.15)	<u>50.30</u> (−0.21)	940
ImageNet32	–	48.23	47.47	47.02	990
	AT	<u>46.39</u> (−1.84)	<u>47.00</u> (−0.47)	<u>45.85</u> (−1.17)	960
	OCSVM	51.15 (+2.92)	50.89 (+3.42)	50.15 (+3.13)	999
	IF	50.76 (+2.53)	50.66 (+3.19)	49.82 (+2.80)	999

To improve the effectiveness of feature-based sample selection on clients, a federated multi-class SVDD loss, explained in Section 3.8, is employed. In clients, this loss is added to the main loss as a regularization term with a weight of $\lambda_{reg} = 10^{-5}$. This weight parameter is empirically chosen to preserve training stability while effectively incorporating the SVDD objective. The empirical analysis shows that higher values tend to deform the embedding space and reduce the overall performance.

The embeddings are not distinctive in early rounds of training, as the model is trained from scratch. Thus, a warm-up duration is needed to achieve meaningful distinctiveness in the embedding space during training. In this thesis, experiments show that the 600th round is appropriate to start the feature-based sample selection. However, the proposed SVDD loss is applied earlier, activating at the 500th round to fine-tune the embedding space. This strategy reduces the chances of mistakenly classifying important samples as outliers, resulting in a more refined embedding space that helps in selecting samples accurately and effectively.

The accuracy improvements achieved by applying the SVDD loss with feature-based sample selection on the CIFAR10 dataset with SVHN noise are presented in Table 4.13. The results show that the proposed SVDD loss decreases accuracy for 50 and 100 clients, but increases accuracy for larger numbers of clients. While IF and OCSVM show similar improvements, the highest improvement, 0.59%, is observed for the 200 clients when OCSVM is employed. The reconstructed image qualities also get slightly better when the SVDD loss is applied, except for 50 clients. As a result, these findings indicate that the SVDD loss improves the outlier detection performance.

The accuracy improvements achieved by applying the SVDD loss with feature-based sample selection on the CIFAR10 dataset with ImageNet32 noise are provided in Table 4.14. As also observed in the experiments with SVHN noise, the outcomes show that the SVDD loss becomes more effective as the client count increases. The maximum gain, 0.99%, is achieved by OCSVM for 200 clients. However, accuracy drops are observed for 50 and 100 clients, especially for OCSVM. This is probably due to the increased local dataset size as the number of clients decreases. A larger local dataset amplifies the influence of the SVDD loss, which may, in turn, distort the embedding space by overfitting to a greater number of noisy samples.

The accuracy improvements achieved by applying the SVDD loss with feature-based sample selection on the MNIST dataset are provided in Tables 4.15 and 4.16. The only improvement, 0.15%, is achieved by OCSVM for 200 clients, which is statistically trivial. A decrease in image quality is also observed, in terms of PSNR and SSIM, indicating the need for hyperparameter optimization of the method, including adjustments to the starting round and the loss weight λ_{reg} .

For a more fine-grained evaluation of classification performance across 1000 clients, the proposed feature-based sample selection methods are further evaluated using more metrics, including macro-averaged precision, recall, and F1-score. The detailed results are demonstrated in Table 4.17. The results are similar to the accuracy-based findings that are presented in Tables 4.13, 4.14, 4.15, and 4.16. The results show that applying the federated SVDD loss results in a 0.58% boost in the F1 score for the CIFAR10 dataset with SVHN noise, and a 0.55% gain for the ImageNet32 dataset with the same noise. IF also shows a notable increase of 0.86% in the F1 score under the ImageNet32 noise. This shows that the federated SVDD loss is capable of improving the effectiveness

of feature-based sample selection in complex datasets like CIFAR10. Conversely, for the MNIST dataset with EMNIST noise, this method reduces the F1 score. Here, the aforementioned overfitting issues may have led to a performance drop by causing noisy sample embeddings to cluster nearer to class centroids, thus hindering effective outlier detection. This indicates the importance of timing the addition of the SVDD loss, considering the complexity of the dataset and the characteristics of the noise.

In summary, the results indicate that feature-based sample selection, with or without the SVDD loss, reduces model accuracy across all scenarios compared to the baseline experiments presented in Tables 4.3 and 4.4. These findings imply that identifying noisy samples in a feature space of 512 or 1024 dimensions is not as effective as expected. This could arise from the increased noise rate and the memorization of noisy samples, making it difficult to differentiate samples in the feature space. However, in the CIFAR10 experiments, it is important to note that the federated SVDD loss improves the effectiveness of feature-based sample selection in terms of accuracy and image quality for larger client numbers, like 200 and 1000, in comparison to experiments without it.

Table 4.13: The highest performance achieved by the MTAE models when feature-based sample selection is applied after the 500th round on the CIFAR10 dataset with 40% SVHN noise, across various client counts. An underlined value shows an accuracy drop with the SVDD.

Total Clients	Method	Without SVDD				With SVDD			
		Accuracy	PSNR	SSIM	Best R.	Accuracy (Gain)	PSNR	SSIM	Best R.
1000	OCSVM	47.24	17.49	36.74	930	47.64 (+0.40)	17.67	38.16	980
	IF	47.95	17.60	38.10	960	48.36 (+0.41)	17.68	38.23	960
200	OCSVM	53.60	18.45	47.90	800	54.19 (+0.59)	18.80	49.02	999
	IF	53.07	18.25	47.16	690	53.51 (+0.44)	18.85	49.00	940
100	OCSVM	58.02	19.49	54.68	930	<u>57.43</u> (−0.59)	19.52	53.93	600
	IF	58.07	19.46	54.17	720	<u>57.43</u> (−0.64)	19.52	53.93	600
50	OCSVM	60.24	20.11	58.43	940	<u>58.30</u> (−1.94)	19.90	57.34	550
	IF	58.72	19.96	58.24	720	<u>58.30</u> (−0.42)	19.90	57.34	550

Table 4.14: The highest performance achieved by the MTAE models when feature-based sample selection is applied after the 500th round on the CIFAR10 dataset, with 40% ImageNet32 noise, across various client counts. An underlined value shows an accuracy drop with the SVDD.

Total Clients	Method	Without SVDD				With SVDD			
		Accuracy	PSNR	SSIM	Best R.	Accuracy (Gain)	PSNR	SSIM	Best R.
1000	OCSVM	46.81	17.87	39.05	820	46.98 (+0.17)	18.11	40.88	930
	IF	46.82	18.07	40.66	960	47.30 (+0.48)	18.21	41.63	990
200	OCSVM	50.43	18.98	50.68	690	51.42 (+0.99)	19.37	52.70	970
	IF	51.07	19.17	52.46	999	<u>49.45</u> (−1.62)	19.38	52.70	999
100	OCSVM	54.48	20.08	58.15	930	<u>53.06</u> (−1.42)	19.99	57.70	930
	IF	52.44	19.99	56.75	600	53.24 (+0.80)	19.88	56.90	970
50	OCSVM	56.04	20.26	59.49	900	<u>55.41</u> (−0.63)	20.66	61.78	940
	IF	55.28	20.68	61.52	940	<u>54.20</u> (−1.08)	20.47	60.87	940

Table 4.15: The highest performance achieved by the MTAE model when feature-based sample selection is applied after the 500th round on the MNIST dataset, with 40% EMNIST letter noise, across various client counts. An underlined value shows an accuracy drop with the SVDD.

Total Clients	Method	Without SVDD				With SVDD			
		Accuracy	PSNR	SSIM	Best R.	Accuracy (Gain)	PSNR	SSIM	Best R.
1000	OCSVM	89.40	20.09	89.02	920	<u>88.90</u> (−0.50)	20.00	87.71	540
	IF	89.11	20.16	88.41	540	<u>88.90</u> (−0.21)	20.00	87.71	540
200	OCSVM	92.90	21.20	91.21	600	<u>92.59</u> (−0.31)	21.08	90.99	600
	IF	92.90	21.20	91.21	600	<u>92.59</u> (−0.31)	21.08	90.99	600
100	OCSVM	92.99	21.49	91.77	670	<u>92.33</u> (−0.66)	21.23	91.02	670
	IF	92.32	21.21	91.22	960	<u>91.60</u> (−0.72)	21.19	91.06	420
50	OCSVM	92.86	21.52	92.02	330	92.86 (+0.00)	21.52	92.02	330
	IF	94.32	22.20	93.41	610	<u>93.05</u> (−1.27)	21.80	92.59	610

Table 4.16: The highest performance achieved by the MTAE models when feature-based sample selection is applied after the 500th round on the MNIST dataset, with 40% ImageNet32 noise, across various client counts. An underlined value shows an accuracy drop with the SVDD.

Total Clients	Method	Without SVDD				With SVDD			
		Accuracy	PSNR	SSIM	Best R.	Accuracy (Gain)	PSNR	SSIM	Best R.
1000	OCSVM	93.04	20.26	87.43	930	<u>92.02</u> (−1.02)	19.63	82.82	920
	IF	91.91	19.77	83.89	930	<u>91.22</u> (−0.69)	19.24	80.67	920
200	OCSVM	94.14	21.05	89.88	900	94.29 (+0.15)	20.85	88.64	990
	IF	94.06	20.95	89.67	600	<u>93.30</u> (−0.76)	20.57	87.36	600
100	OCSVM	94.56	21.53	89.87	960	<u>93.83</u> (−0.73)	21.30	87.83	960
	IF	93.41	21.06	87.11	960	<u>92.55</u> (−0.86)	20.95	89.04	420
50	OCSVM	95.27	22.05	91.82	950	<u>93.87</u> (−1.40)	21.93	92.00	460
	IF	95.59	22.13	92.12	610	<u>93.87</u> (−1.72)	21.93	92.00	460

Table 4.17: The highest performance achieved by the MTAE models when feature-based sample selection is used after the 500th round on the CIFAR10 and MNIST datasets with 40% noise, using 1000 clients and more metrics. An underlined value shows an accuracy drop with the SVDD.

Train Dataset	Noise Type	Method	Without SVDD				With SVDD			
			Precision	Recall	F1	Best R.	Precision (Gain)	Recall (Gain)	F1 (Gain)	Best R.
CIFAR10	SVHN	OCSVM	46.74	47.24	46.72	930	47.81 (+1.07)	47.64 (+0.40)	47.30 (+0.58)	980
		IF	47.20	47.95	47.19	960	47.60 (+0.40)	48.36 (+0.41)	47.37 (+0.18)	960
	ImageNet32	OCSVM	46.40	46.81	46.14	820	47.12 (+0.72)	46.98 (+0.17)	46.69 (+0.55)	930
		IF	46.21	46.82	45.76	960	48.02 (+1.81)	47.30 (+0.48)	46.62 (+0.86)	990
MNIST	EMNIST	OCSVM	89.77	89.27	89.37	920	<u>89.09</u> (−0.68)	<u>88.74</u> (−0.53)	<u>88.76</u> (−0.61)	540
		IF	89.21	88.97	88.98	540	<u>89.09</u> (−0.12)	<u>88.74</u> (−0.23)	<u>88.76</u> (−0.22)	540
	ImageNet32	OCSVM	93.08	93.01	93.00	930	<u>92.15</u> (−0.93)	<u>91.98</u> (−1.03)	<u>92.00</u> (−1.00)	920
		IF	91.97	91.84	91.85	930	<u>91.34</u> (−0.63)	<u>91.13</u> (−0.71)	<u>91.15</u> (−0.70)	920

4.5. SV-based Sample Selection

To empirically evaluate the effectiveness of SV-based and LOO-based sample selection strategies under noisy conditions, a series of experiments is conducted on the MNIST dataset with a closed-set noise ratio of 40% using the 2-layered CNN model illustrated in Figure 3.3. All implementations are performed on the FedML library. Due to the high computational complexity of SV-based methods, particularly TMC-Shapley and G-Shapley, the dataset is partitioned among five clients in a non-IID manner. FL training is executed for 100 communication rounds using the FedAvg aggregation algorithm. On each client, SGD is used as the local optimizer, with a learning rate of 0.03, a batch size of 128, and 10 local update steps per round.

For the purpose of data valuation, a fixed evaluation subset is constructed from the test set. Specifically, it consists of 100 images selected by sampling 10 instances from each digit class (0–9), thereby ensuring class balance within the subset. This evaluation set is held out on each client and is not involved in training. It is used in the evaluation function $V()$, which measures the performance of a model trained on a given subset of training data. In this way, $V()$ quantifies the contribution of each data point by evaluating the model’s effectiveness on this evaluation subset.

For TMC-Shapley and G-Shapley, convergence of the SV estimates is determined using the criterion provided in Equation 4.1. In this expression, n represents the number of local training samples, t denotes the current iteration, and ϕ_i^t corresponds to the estimated contribution of the i -th sample at iteration t . This condition ensures that the average relative change in the contribution estimates across the most recent three iterations falls below a threshold of 0.1, thereby indicating convergence. Although stricter convergence thresholds can potentially lead to more accurate valuation results, they considerably increase the total computation time required.

$$\frac{1}{n} \sum_{i=1}^n \frac{|\phi_i^t - \phi_i^{t-3}|}{|\phi_i^t|} < 0.1 \quad (4.1)$$

A baseline performance is first established by training the global model on the noisy dataset without applying any data valuation method. Under this configuration, a test accuracy of 69.27% is achieved, while training on the clean dataset yields an accuracy of 89.72%. The experimental outcomes are summarized in Table 4.18.

Table 4.18: The test accuracies with SV-based and LOO-based methods after 100 rounds. The accuracy improvements compared to the noisy setting are given in parentheses.

Method	Test Accuracy (%)	Training Time (h)
Baseline	69.27	0.5
G-Shapley	77.78 (+8.51)	23
TMC-Shapley	76.38 (+7.11)	140
LOO-Sum	78.08 (+8.81)	
LOO-StdDev	77.00 (+7.73)	2
LOO	77.78 (+8.51)	

It can be observed that all evaluated sample valuation methods resulted in notable gains in test accuracy compared to the noisy baseline. Among the SV-based approaches, G-Shapley achieved the highest accuracy gain (8.51%), while TMC-Shapley followed closely with a 7.11% gain. However, these improvements came at the cost of considerably longer training times, 23 and 140 hours, respectively.

In contrast, LOO-based methods provided comparable or even superior accuracy gains with substantially lower computational requirements. Specifically, LOO-Sum yielded the highest accuracy among all methods (78.08%), improving upon the original LOO accuracy of 77.78%. LOO-StdDev, on the other hand, resulted in a slightly lower performance (77.00%), which may be attributed to non-IID distribution across clients.

Despite the relaxed convergence criteria applied to SV-based methods, the LOO-based methods remained highly competitive in terms of both performance and efficiency. These findings suggest that, while SV-based methods are theoretically strong, their practical application in FL may be constrained by scalability and runtime limitations.

4.6. Laplacian-based Differential Privacy

In this thesis, the Laplacian-based DP algorithm, defined in Section 3.11, is evaluated under different numbers of clients and privacy budgets ϵ . The classification models defined in Section 3.4 are employed in these experiments. Since the DP method requires

Table 4.19: The highest accuracies for the MNIST and CIFAR10 datasets, using 100 clients across different gradient clipping norms.

Gradient Norm ξ	MNIST		CIFAR10	
	Accuracy	Best Round	Accuracy	Best Round
10	80.95	810	61.05	930
20	82.49	810	67.86	930
50	87.38	960	73.05	930
100	93.71	970	75.30	930
∞	98.93	830	81.97	930

gradient clipping with l_1 -norm, different norm values are tested for 100 clients to find the optimal one that maintains model accuracy. The results in Table 4.19 show that decreasing the norm value ξ leads to a decrease in model accuracy. This result is reasonable as lower norm values mean lower learning capacity. Compared to an unbounded norm ($\xi = \infty$), the lowest accuracy drop is achieved with a norm of $\xi = 100$ for the CIFAR10 and MNIST datasets. Specifically, the accuracy decreases by 6.67% on the CIFAR10 and by 5.22% on the MNIST. As described in Equation 3.22, higher norm values enhance privacy but also distort true gradient signals during training, which causes less accurate model updates and divergence issues.

In the following experiments, a norm of $\xi = 100$ is preferred to balance privacy protection and model accuracy. As shown in Table 4.20, the highest accuracies achieved during 1000 training rounds are reported for varying numbers of clients. The results show that model accuracy increases as the number of clients is reduced. This is expected as fewer clients correspond to more local data per client with greater diversity, resulting in increased accuracy.

To examine the effect of privacy budget on model accuracy for 100 clients, different privacy budgets ϵ are tested on the CIFAR10 and MNIST datasets, using a gradient norm of $\xi = 100$. Test accuracy trends across 1000 training rounds for both datasets are presented in Figure 4.1. For both datasets, the results show that the test accuracy fluctuation increases when smaller privacy budgets ($\epsilon = 10^4$) are used. This is expected as smaller privacy budgets increase the noise rate of DP, thereby reducing learning

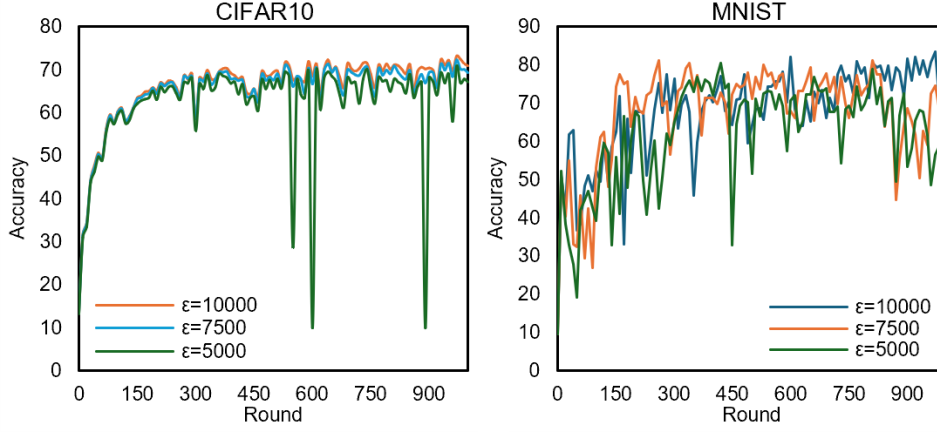


Figure 4.1: The test accuracy trends across different privacy budgets for 100 clients on the CIFAR10 and MNIST datasets. Best viewed in color.

accuracy. The DP formulation says that increasing the total number of clients increases the risk of privacy leaks. For 100 clients, a privacy budget of $\epsilon = 10^4$ can be preferred for both datasets as it yields higher accuracy and more stable training than the others.

Table 4.20: The highest accuracies for the MNIST and CIFAR10 datasets, using a gradient norm of $\xi = 100$ across different numbers of clients.

	MNIST			CIFAR10	
	Total Clients	Accuracy	Best Round	Accuracy	Best Round
	50	95.06	850	76.48	900
	100	93.71	970	75.30	930
	200	92.78	990	72.47	880
	1000	88.17	990	62.02	970

To determine an appropriate privacy budget without causing excessive accuracy loss, experiments are performed across different client counts and privacy budgets. The results are presented in Table 4.21, which confirms that a higher ϵ generates less noise, thereby yielding better model accuracy. The results indicate that larger models, such as the one used for MNIST, are more sensitive to noise. Moreover, the accuracy loss increases as the number of clients increases. This is mainly caused by the increased

Table 4.21: The highest accuracies observed for the MNIST and CIFAR10 datasets under different privacy budgets and client counts.

		MNIST		CIFAR10	
Total Clients	$\epsilon \times 10^3$	Accuracy	Best Round	Accuracy	Best Round
50	5	90.60	330	74.36	940
	7.5	92.39	460	75.34	940
	10	93.22	460	75.46	990
100	5	80.59	420	70.68	970
	7.5	81.21	810	72.32	970
	10	83.49	970	73.11	970
200	5	73.36	900	59.97	890
	7.5	71.60	790	62.27	640
	10	72.49	620	63.23	900
1000	5	63.97	820	22.00	10
	7.5	65.39	990	24.72	20
	10	66.50	760	31.19	40
	500			55.15	930
	750	—	—	58.02	930
	1000			59.35	920

noise level for clients having fewer local samples (n_i). According to the Equation 3.22, the noise level should increase because a smaller dataset leads to larger per-sample influence, which requires adding more noise to maintain the same privacy guarantee. This formulation also generates higher noise as the number of clients per round increases. Thus, applying DP for a larger number of clients with fewer local samples becomes more challenging. For 1000 clients, the MNIST and CIFAR10 models diverge for privacy budgets of $\epsilon \leq 10^3$ and $\epsilon \leq 10^5$, respectively. However, empirical studies for the CIFAR10 show that privacy budgets of $\epsilon \geq 5 \times 10^5$ yield a good balance between accuracy and privacy.

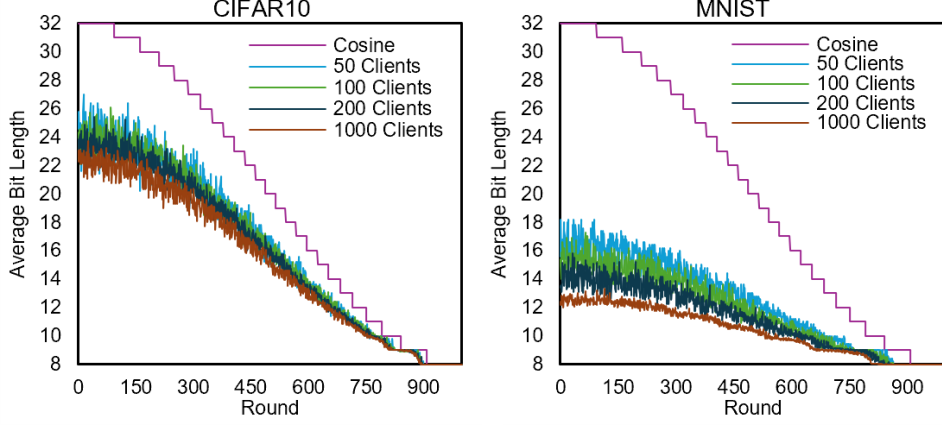


Figure 4.2: The round-wise average bit-length across varying numbers of clients on the CIFAR10 and MNIST, using cosine annealing bit-length scheduling with client importance weighting ($\lambda_h = 0.75$, $b_{min} = 8$). Best viewed in color.

4.7. Adaptive Quantization

In this part of the study, the adaptive quantization method introduced in Section 3.10 is tested with different client counts and bit-length scheduling strategies, using the models from Section 3.4. Except for the CIFAR10 setup with 1000 clients, where $\epsilon = 10^6$ is used, all experiments are run with a privacy budget of $\epsilon = 10^4$ as described in Section 3.11. These values are selected because they offer a better trade-off between model accuracy and privacy, as detailed in Section 4.6.

The proposed method uses a global bit-length scheduler based on cosine annealing. This scheduler adjusts the precision of model updates depending on the training round. The server follows this schedule when quantizing the global model before sending it to clients. Clients either use the global precision value directly or adjust it using a weighting term based on their own importance ν_i , which is determined from the size and homogeneity of their local data. In the experiments, the term Cosine refers to using the global schedule as-is, while Dynamic refers to applying client-based weighting. The average bit-lengths achieved per round are shown in Figure 4.2 for various client counts.

To examine how changing λ_h values in the set 0.25, 0.5, 0.75, 1.0 affects both accuracy and total communication cost, extensive experiments are performed. The related results on the CIFAR10 and the MNIST are given in Figure 4.3. Diamond markers show the highest accuracies. As expected, increasing the number of clients leads to higher

total communication costs. For large-scale setups like 1000 clients, smaller λ_h values help reduce the overall communication volume. The experiments also show that as the number of clients increases, dataset size has a stronger effect on accuracy and communication than data homogeneity. For instance, $\lambda_h = 0.25$ gives the lowest communication cost in the CIFAR10 experiments with 200 and 1000 clients, and also in the MNIST with 1000 clients. This is consistent with the fact that local data size becomes more influential when client numbers are high.

The best accuracy on the MNIST with 1000 clients is achieved at $\lambda_h = 0.5$. In the CIFAR10, the change in accuracy due to λ_h is minimal, remaining within 1.12% for 200 clients. However, the MNIST model is more sensitive to this parameter, with a 9.75% difference between the best and worst settings. Overall, $\lambda_h = 0.75$ or $\lambda_h = 0.5$ appears to provide a good compromise between accuracy and communication cost. Therefore, $\lambda_h = 0.75$ is used in the rest of the experiments.

Further experiments compare the Cosine and Dynamic bit-length schedulers across various client counts using the CIFAR10 and MNIST. Static quantization with 4, 8, and 16 bits is also included to better understand how bit precision affects performance. Full-precision (32-bit) results are also reported as a baseline. The highest accuracy over 1000 rounds and the total communication cost are summarized in Tables 4.22 and 4.23, while Figure 4.4 shows the global test accuracy over time.

As expected, using a lower bit-length value reduces communication cost but may decrease accuracy. This is especially true for 4-bit quantization, which causes instability due to large quantization errors, particularly in non-IID settings. In contrast, 8-bit quantization offers a strong balance. Both Cosine and Dynamic scheduling improve this trade-off by achieving similar accuracy while significantly cutting communication volume. For example, the Cosine reduces total communication by about 37.46% compared to full-precision. The Dynamic performs even better, saving between 49.54% and 52.64% on the MNIST and between 43.45% and 45.06% on the CIFAR10, depending on the number of clients.

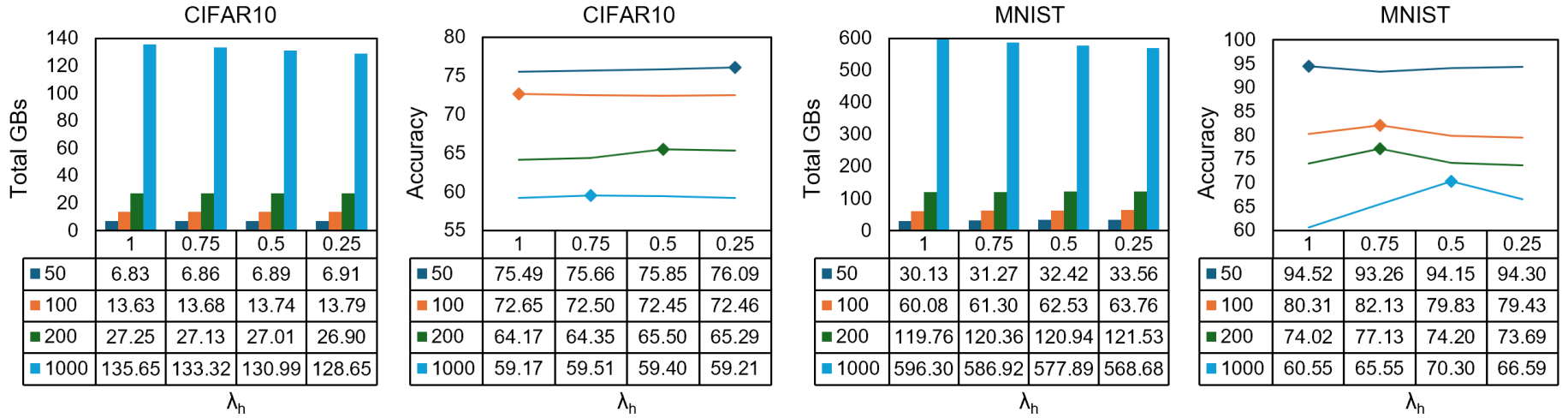


Figure 4.3: The total communicated gigabytes and highest accuracies after 1000 training rounds for the CIFAR10 and MNIST datasets, evaluated under varying λ_h values and number of clients, with a privacy budget of $\epsilon = 10^4$, except for the case of 1000 clients on the CIFAR10, where $\epsilon = 10^6$ is applied. Best viewed in color.

Table 4.22: The total communicated gigabytes and highest accuracies after 1000 training rounds on the MNIST and CIFAR10 datasets using 50 and 100 clients. A privacy budget of $\epsilon = 10^4$ is used, except for the case of 1000 clients on the CIFAR10, where $\epsilon = 10^6$ is applied. *Cosine* means cosine annealing without client importance ($b_{min} = 8$), while *Dynamic* denotes cosine annealing with client importance ($\lambda_h = 0.75$, $b_{min} = 8$).

		MNIST			CIFAR10		
Total Clients	Bit Length	Total GBs	Accuracy	Best Round	Total GBs	Accuracy	Best Round
50	INT4	7.75	76.65	860	1.52	50.56	340
	INT8	15.49	92.85	460	3.03	75.76	900
	INT16	30.98	94.61	610	6.07	75.74	900
	FP32	61.97	93.22	460	12.13	75.46	990
	Cosine	38.75	93.04	460	7.59	75.97	870
	Dynamic	31.27	93.26	460	6.86	75.66	900
100	INT4	15.49	60.49	380	3.03	46.31	140
	INT8	30.98	81.50	730	6.07	72.70	970
	INT16	61.97	78.08	340	12.13	72.67	970
	FP32	123.93	83.49	970	24.27	73.11	970
	Cosine	77.50	81.77	770	15.18	72.34	970
	Dynamic	61.30	82.13	890	13.68	72.50	640

Table 4.23: The total communicated gigabytes and highest accuracies after 1000 training rounds on the MNIST and CIFAR10 datasets using 200 and 1000 clients. A privacy budget of $\epsilon = 10^4$ is employed, except for the case of 1000 clients on the CIFAR10, where $\epsilon = 10^6$ is used. *Cosine* means cosine annealing without client importance ($b_{min} = 8$), and *Dynamic* denotes cosine annealing with client importance ($\lambda_h = 0.75$, $b_{min} = 8$).

		MNIST			CIFAR10		
Total Clients	Bit Length	Total GBs	Accuracy	Best Round	Total GBs	Accuracy	Best Round
200	INT4	30.98	49.37	230	6.07	41.74	50
	INT8	61.97	76.49	540	12.13	65.24	790
	INT16	123.93	74.96	710	24.27	64.85	510
	FP32	247.86	72.49	620	48.53	63.23	900
	Cosine	155.01	76.99	990	30.35	65.19	760
	Dynamic	120.36	77.13	540	27.13	64.35	890
1000	INT4	154.91	40.17	450	30.33	34.41	240
	INT8	309.83	64.40	990	60.67	59.33	999
	INT16	619.65	65.76	980	121.34	59.49	999
	FP32	1239.31	66.50	760	242.67	59.35	920
	Cosine	775.03	64.85	820	151.76	59.54	920
	Dynamic	586.92	65.55	610	133.32	59.51	930

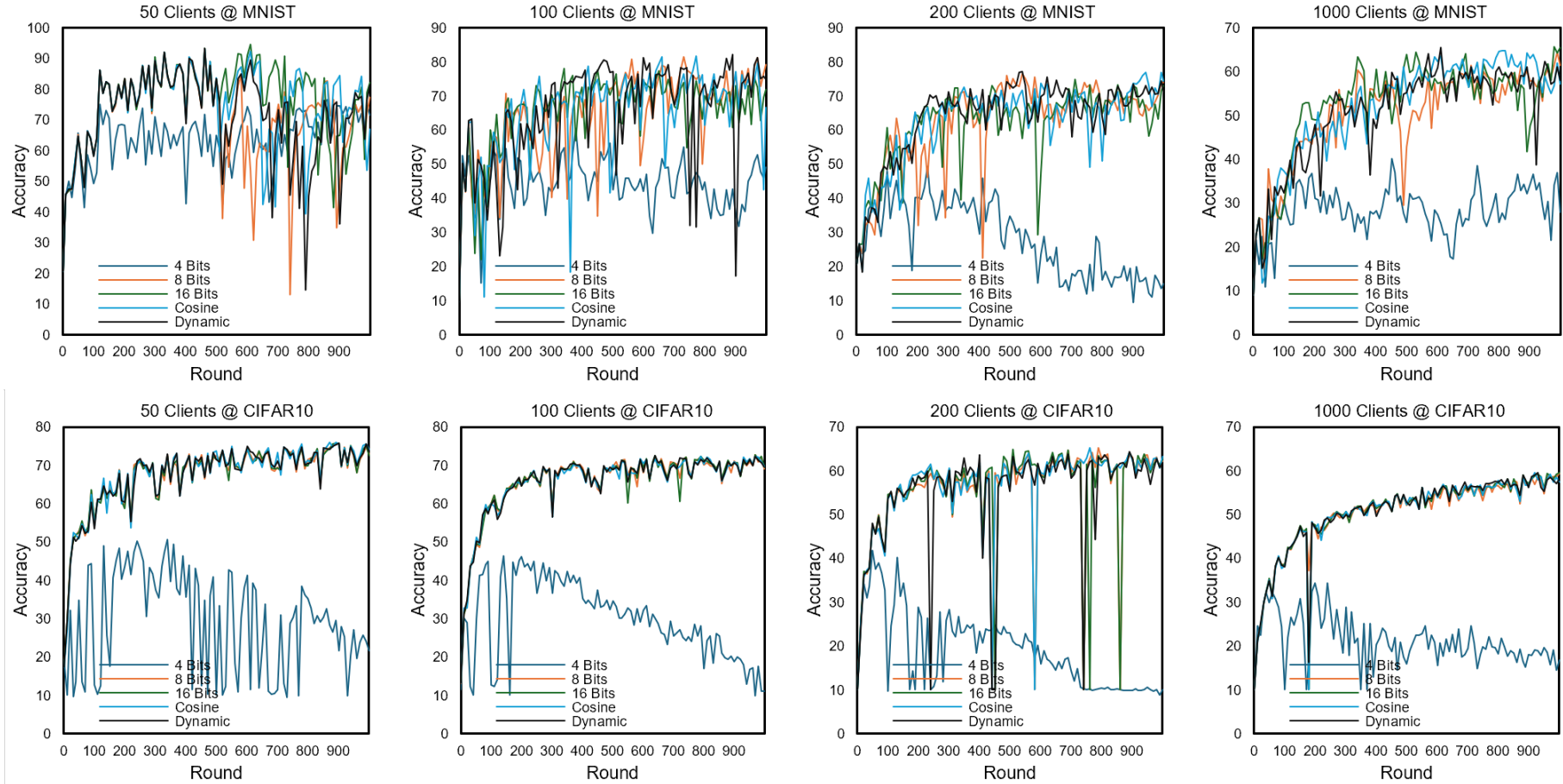


Figure 4.4: The test accuracies on the MNIST and CIFAR10 datasets across varying client counts and bit-lengths, using a privacy budget of $\epsilon = 10^4$, except for the CIFAR10 with 1000 clients, where $\epsilon = 10^6$ is used. *Cosine* denotes cosine annealing with $b_{min} = 8$, while *Dynamic* integrates client importance ($\lambda_h = 0.75$) into the same schedule. Best viewed in color.

When using 50 clients, the Cosine achieves 93.04% accuracy on the MNIST and 75.97% on the CIFAR10, with total communication costs of 38.75 GB and 7.59 GB, respectively. These are better than the 32-bit results, which reach similar accuracy but with higher costs: 61.97 GB for the MNIST and 12.13 GB for the CIFAR10. The Dynamic performs even more efficiently, reducing the costs to 31.27 GB for the MNIST and 6.86 GB for the CIFAR10 while maintaining comparable accuracies of 93.26% and 75.66%, respectively. In the MNIST, 16-bit quantization gives the best accuracy at 94.61% and reduces the communication cost by half. This may be due to the regularization effect of moderate quantization error. At 100 clients, the Dynamic reaches 82.13% accuracy on the MNIST with 61.30 GB of communication, outperforming 16-bit settings in both accuracy and efficiency. The CIFAR10 results remain stable at around 72% for both Cosine and Dynamic, while cutting cost nearly in half compared to full-precision. Overall, 8-bit quantization gives surprisingly good performance at this scale.

For 200 clients, the Dynamic reaches 77.13% accuracy on the MNIST with 120.36 GB of communication, significantly better than the 32-bit setting's 72.49% at 247.86 GB. In the CIFAR10, the Cosine gives 65.19% accuracy, and the Dynamic is close at 64.35%. Interestingly, the Dynamic with $\lambda_h = 0.5$ reaches 65.50%, the best score at this scale. This trend suggests that the local dataset size becomes more critical as client numbers increase. In both datasets, 8-bit quantization continues to give strong performance while cutting communication by a factor of four.

When scaled to 1000 clients, communication becomes a major concern, and both Cosine and Dynamic schedulers show that they scale well. The Dynamic delivers 65.55% on the MNIST and 59.51% on the CIFAR10, reducing communication by 52.64% and 45.06%, respectively. The Cosine slightly outperforms the Dynamic in the CIFAR10 accuracy (59.54%) but with a higher cost. In contrast, the Dynamic performs better on the MNIST in both accuracy and efficiency, showing that client importance is especially useful at large scales. Static 8-bit and 16-bit quantization methods also work well at this size, striking a good balance between cost and performance.

To summarize, the proposed Cosine and Dynamic scheduling methods manage to reduce communication cost while preserving model accuracy, even at large scales such as 1000 clients. By adjusting precision globally with cosine annealing and locally using entropy-based importance, the methods save over 50% of communication on the MNIST and 45%

on the CIFAR10 compared to full-precision, while still reaching competitive accuracy. This shows that adaptive quantization is not only efficient but also scalable.

A major benefit of the methods is their simplicity. They are easy to implement and can be used in many types of applications. The entropy calculation used for client weighting runs in linear time, making it efficient. Since they do not rely on dataset-specific structures, they can be adapted easily to other settings. One limitation is that the Dynamic requires tuning λ_h to get the best results. Another is that neither method considers the quality of the local data, which could affect performance. This issue might be addressed in future work using client or data valuation techniques [9, 10, 15, 87].

Finally, the two types of noise used in this thesis, Laplacian noise for DP and stochastic noise from quantization, are both unbiased. The Laplace noise is zero-mean, so it does not influence the expected value of model updates. The quantization method uses stochastic rounding, which also preserves the original expected value, as explained by Alistarh et al. [46]. Because both noise sources are unbiased, they do not introduce systematic bias, even though they increase variance in the updates.

4.7.1. Performance on Medical Image Datasets

In this section, the adaptive quantization methods are tested on three popular medical image classification datasets: PAP-Smear [88], Chest X-ray (Pneumonia) [89], and BreakHisV1 [90]. Table 4.24 lists the details of these datasets. The purpose is to examine how well the methods balance communication efficiency and model accuracy in FL. To assess performance, several metrics are used, including total communicated gigabytes, accuracy, precision, recall, F1 score, and balanced accuracy (BACC).

The EfficientNet-B0 model, pretrained on ImageNet, is used for these experiments via the FedML framework [86]. This model is widely adopted for medical image tasks due to its efficiency and strong feature representation. In each experiment, the FedAvg algorithm is used across 10 clients for 100 communication rounds. Local training uses the Adam optimizer with a learning rate of 3×10^{-4} and weight decay of 5×10^{-4} . Each client performs two local epochs with a batch size of 64. Evaluation is done every three rounds, and the highest test BACC observed during training is reported as the "Best Round" to mitigate the effects of accuracy fluctuations caused by non-IID data,

Table 4.24: Classification datasets used in medical imaging.

Dataset	PAP-Smear	Pneumonia	BreakHisV1
Image Size	128x128	224x224	128x128
Train Samples	3,049	5,232	5,361
Test Samples	1,000	624	2,548
# Classes	5	2	2

which can make relying on the final round unfair. For all configurations, $\epsilon = 10^4$ and $\xi = 1000$ are used, based on prior findings indicating a good balance between privacy and utility.

To simulate non-IID conditions, data is distributed using a Dirichlet distribution with $\alpha = 0.5$, following the setup used in earlier CIFAR10 and MNIST experiments. Both Cosine and Dynamic bit-length scheduling methods are applied to these datasets. Static quantization with 12-bit and 16-bit fixed lengths is also tested to better understand the communication-accuracy trade-off. A minimum bit length of $b_{\min} = 12$ is used to prevent model collapse at lower values. For the Dynamic method, $\lambda_h = 0.75$ and $b_{\min} = 12$ yield good results. For comparison, full-precision training (32-bit float) is included. Table 4.25 summarizes the results in terms of the best BACC over 100 rounds and total gigabytes communicated.

The findings show that both Cosine and Dynamic methods reduce communication compared to the 32-bit baseline while keeping F1 and BACC scores competitive. On the PAP-Smear dataset, the Dynamic method cuts communication by 37%, with only a small drop in F1 (from 90.94% to 89.41%) and BACC (from 91.03% to 89.59%). For the Pneumonia dataset, the Cosine method achieves 31% lower communication and a similar F1 score (93.79% vs. 93.99%). However, on BreakHisV1, the Dynamic method reduces communication by 39%, with F1 decreasing from 91.44% to 87.03% and BACC from 91.45% to 88.36%, likely caused by the loss of fine-grained histopathological features due to quantization, along with high intra-class variability, which makes the model more sensitive to precision degradation.

Interestingly, in some cases, 16-bit static quantization provides better results than the

adaptive methods. For instance, in the PAP-Smear dataset, the 16-bit setup outperforms both Cosine and Dynamic in F1 and BACC while using less data. This shows that using a consistent, moderate bit length can be more effective in some situations. Adaptive methods adjust bit length throughout training, which may sometimes cause higher communication due to a higher bit length usage. Additionally, the bit-length variation may lead to unstable model performance due to fluctuating quantization noise. As shown in Figure 4.5, test BACC values under adaptive quantization methods fluctuate more than the 32-bit baseline.

Overall, these results highlight the trade-off between communication cost and model quality. While fixed-bit quantization, like 16-bit, can yield consistent results with low communication, adaptive methods offer flexibility that becomes useful as client diversity increases. However, this flexibility does not always lead to better performance or lower overhead. Moreover, combining Laplacian noise with quantization noise can lead to instability in training. Laplacian noise adds randomness, and quantization reduces numerical precision. Together, they can increase performance variability, as seen in Figure 4.5. Therefore, while adaptive quantization is promising, its effectiveness depends on careful tuning to handle the balance between precision, noise, and communication.

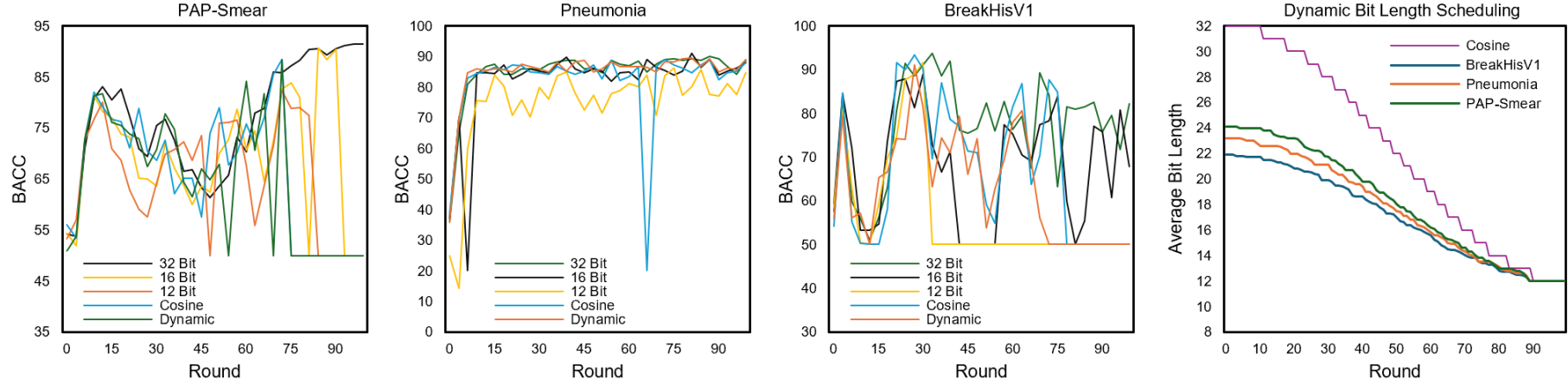


Figure 4.5: The BACC scores on the global test set and average per-round bit-lengths for the medical imaging datasets across various bit-length settings, using 10 clients and a privacy budget of $\epsilon = 10^4$. The average bit length is measured as the mean across clients. *Cosine* applies cosine annealing with ($b_{min} = 12$), while *Dynamic* incorporates client importance ($\lambda_h = 0.75$, $b_{min} = 12$) into this schedule. Best viewed in color.

Table 4.25: The total communicated GBs and highest performances after 100 training rounds on the medical image datasets, including 10 clients under different bit-lengths and a privacy budget of $\epsilon = 10^4$. *Cosine* employs $b_{min} = 12$, and *Dynamic* employs $\lambda_h = 0.75$ and $b_{min} = 12$.

Dataset	Bit Length	Total GBs	Accuracy	Precision	Recall	F1 Score	BACC	Best Round
PAP-Smear	INT12	11.33	86.20	87.88	86.28	86.31	86.28	75
	INT16	15.11	90.90	91.12	91.03	90.94	91.03	81
	FP32	30.22	89.90	90.72	90.07	89.96	90.07	87
	Cosine	20.85	88.80	90.18	88.99	88.95	88.99	87
	Dynamic	19.00	89.40	90.21	89.59	89.41	89.59	81
Pneumonia	INT12	11.33	92.95	93.40	91.54	92.32	91.54	30
	INT16	15.11	91.51	93.48	88.93	90.50	88.93	30
	FP32	30.22	94.39	94.19	93.80	93.99	93.80	33
	Cosine	20.85	94.23	94.22	93.42	93.79	93.42	27
	Dynamic	18.73	92.63	93.27	91.03	91.94	91.03	27
BreakHisV1	INT12	11.33	81.00	80.42	82.27	80.57	82.27	72
	INT16	15.11	90.23	89.28	90.64	89.80	90.64	84
	FP32	30.22	91.95	91.43	91.45	91.44	91.45	99
	Cosine	20.85	86.70	86.18	88.39	86.40	88.39	72
	Dynamic	18.43	87.44	86.52	88.36	87.03	88.36	72

5. CONCLUSION

5.1. Summary of Contributions

This thesis presents novel methods for FL to improve communication efficiency, preserve privacy, and improve model performance through data valuation. Within the scope of data valuation, a multi-task autoencoder architecture was proposed, combining both reconstruction and classification tasks to efficiently detect and filter malicious, redundant, or low-quality local data samples during training through loss analysis. Specifically, unsupervised sample selection methods were developed to enhance the effectiveness of FL models during the training phase without relying on preprocessing or pretrained models. These methods incorporated unsupervised outlier detection algorithms such as OCSVM and IF, along with adaptive thresholding techniques that were centrally controlled but run locally on clients. This approach makes a notable contribution to the field, particularly in large-scale and non-IID FL. Comprehensive experiments on the MNIST and CIFAR10 datasets proved the effectiveness of this approach in terms of model accuracy, robustness, and resilience to noisy data.

This thesis proposes a novel loss function based on multi-class deep SVDD, specifically tailored for FL. This approach effectively separated outlier and normal samples in feature space and improved the effectiveness of feature-based sample selection. Extensive experiments on the CIFAR10 showed that this approach achieved modest yet consistent accuracy gains, emphasizing its utility in improving model robustness in challenging training conditions involving diverse and noisy data distributions.

Additionally, despite their high computational cost, this thesis presents both SV-based and LOO-based data valuation methods for FL. SV-based methods, such as TMC-Shapley and G-Shapley, provide theoretically strong estimates of individual sample contributions, yet their high computational cost significantly restricts scalability. However, LOO-based methods offer a more practical alternative through localized exclusion-based evaluations. Experimental results showed that, despite their simplicity, LOO-based methods achieved comparable accuracy improvements relative to SV-based methods, while incurring significantly lower computational cost.

This thesis presents two innovative adaptive quantization methods designed to enhance communication efficiency while preserving privacy in FL. The first method systematically reduces quantization bit-length over the training process by using a global bit-length scheduler based on round-based cosine annealing, so balancing communication efficiency with model accuracy. Moreover, a new client-based adaptive quantization method is proposed, dynamically changing bit lengths based on dataset entropy and size, so prioritizing clients with larger and more informative data. These quantization methods are combined with a Laplacian-based DP mechanism to provide better privacy guarantees and so reduce privacy risks without appreciably compromising model performance. In addition, the effectiveness of these methods was tested on standard datasets such as CIFAR10 and MNIST, as well as specialized medical image datasets such as PAP-Smear, Pneumonia (Chest X-ray), and BreakHisV1. The experiments showed that these methods significantly reduced communication overhead, even in the presence of noise introduced by Laplacian-based DP, which is particularly critical for sensitive medical datasets. These results emphasized the viability and effectiveness of combining adaptive quantization with privacy-preserving methods in complex FL environments.

5.2. Future Directions

The computational overhead of the proposed methods will be studied, including lightweight and scalable solutions that maintain learning accuracy while reducing resource demands in large-scale and real-world FL cases. Optimal starting round strategies for sample selection and adaptive techniques for dynamically adjusting hyperparameters, such as contamination rates in OCSVM and IF, will be examined to enhance robustness across varying noise levels without manual tuning. The methods will also be refined to handle more complex data types, especially in real-world FL scenarios involving heterogeneous devices and non-IID data distributions.

Adaptive quantization strategies will be studied, exploring more advanced algorithms for accurately estimating client importance, with the aim of further improving the balance between communication efficiency and learning accuracy in large-scale FL. Additionally, advanced privacy-preserving mechanisms, such as SMC, will be investigated to strengthen privacy guarantees in FL. These enhancements aim to optimize communication and privacy trade-offs, making FL more applicable to real-world problems.

REFERENCES

- [1] McMahan H. B., Moore E., Ramage D., Hampson S., y Arcas B. A. A., (2017), Communication-efficient learning of deep networks from decentralized data, in “Proceedings of the 20th International Conference on Artificial Intelligence and Statistics”, Fort Lauderdale, FL, USA, 1273–1282.
- [2] Li T., Sahu A. K., Talwalkar A., Smith V., (2020), “Federated learning: Challenges, methods, and future directions”, IEEE Signal Processing Magazine, 37(3), 50–60.
- [3] Huang J., Qian F., Guo Y., Zhou Y., Xu Q., Mao Z. M., Sen S., Spatscheck O., (2013), “An in-depth study of lte: Effect of network protocol and application behavior on performance”, ACM SIGCOMM Computer Communication Review, 43(4), 363–374.
- [4] McMahan H. B., Ramage D., Talwar K., Zhang L., (2018), “Learning differentially private recurrent language models”, <https://arxiv.org/abs/1710.06963>, Accessed: 2025-05-20.
- [5] Geyer R. C., Klein T., Nabi M., (2017), “Differentially private federated learning: A client-level perspective”, <https://arxiv.org/abs/1712.07557>, Accessed: 2025-05-20.
- [6] Li J., Khodak M., Caldas S., Talwalkar A., (2019), “Differentially private meta-learning”, <https://arxiv.org/abs/1909.05830>, Accessed: 2025-05-20.
- [7] Bonawitz K., Ivanov V., Kreuter B., Marcedone A., McMahan H. B., Patel S., Ramage D., Segal A., Seth K., (2017), Practical secure aggregation for privacy-preserving machine learning, in “Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security”, Dallas, TX, USA, 1175–1191.
- [8] Sattler F., Wiedemann S., Muller K.-R., Samek W., (2020), “Robust and communication-efficient federated learning from non-i.i.d. data”, IEEE Transactions on Neural Networks and Learning Systems, 31(9), 3400–3413.
- [9] Li A., Zhang L., Tan J., Qin Y., Wang J., Li X.-Y., (2021), Sample-level data selection for federated learning, in “Proceedings of the IEEE Conference on Computer Communications (INFOCOM 2021)”, IEEE, Vancouver, BC, Canada, 1–10.
- [10] Ardiç E., Genç Y., (2023), Data valuation methods for federated learning, in “Proceedings of the 2023 31st Signal Processing and Communications Applications Conference (SIU)”, IEEE, Istanbul, Turkey, 1–4.
- [11] Novoa-Paradela D., Fontenla-Romero O., Guijarro-Berdiñas B., Orellana-Cañas D., (2023), A federated learning architecture for anomaly detection on the edge

- using deep autoencoders, in “Proceedings of the 2023 IEEE International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises”, IEEE, Paris, France, 1–6.
- [12] Li S., Cheng Y., Wang W., Liu Y., Chen T., (2020), “Learning to detect malicious clients for robust federated learning”, <https://arxiv.org/abs/2002.00211>, Accessed: 2025-05-20.
 - [13] Li X., Zhao H., Deng W., (24), “Iofl: Intelligent-optimization-based federated learning for non-iid data”, IEEE Internet of Things Journal, 11(9), 16693–16699.
 - [14] Zhao J., Zhu X., Wang J., Xiao J., (2021), Efficient client contribution evaluation for horizontal federated learning, in “Proceedings of the 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)”, IEEE, Virtual conference, Toronto, Ontario, Canada, 3060–3064.
 - [15] Wang T., Rausch J., Zhang C., Jia R., Song D., (2020), “A principled approach to data valuation for federated learning”, Springer International Publishing, Cham, Switzerland, 153–167.
 - [16] Shin J., Li Y., Liu Y., Lee S.-J., (2022), Fedbalancer: Data and pace control for efficient federated learning on heterogeneous clients, in “Proceedings of the 20th Annual International Conference on Mobile Systems, Applications and Services (MobiSys ’22)”, Portland, Oregon, USA, 436–449.
 - [17] Nardi M., Valerio L., Passarella A., (2022), Anomaly detection through unsupervised federated learning, in “Proceedings of the 2022 18th International Conference on Mobility, Sensing and Networking”, IEEE, Guangzhou, China, 495–501.
 - [18] Bhowmick A., Duchi J., Freudiger J., Kapoor G., Rogers R., (2018), “Protection against reconstruction and its applications in private federated learning”, <https://arxiv.org/abs/1812.00984>, Accessed: 2025-05-20.
 - [19] Carlini N., Liu C., Erlingsson Ú., Kos J., Song D., (2019), The secret sharer: evaluating and testing unintended memorization in neural networks, in “Proceedings of the 28th USENIX Security Symposium”, Santa Clara, CA, USA, 267–284.
 - [20] Cheng Z., Wang S., Zhang P., Wang S., Liu X., Zhu E., (2021), “Improved autoencoder for unsupervised anomaly detection”, International Journal of Intelligent Systems, 36(12), 7103–7125.
 - [21] Gray R. M., (2011), “Entropy and information theory”, 2nd Edition, Springer Science & Business Media, New York, NY, USA.
 - [22] LeCun Y., Bengio Y., Hinton G., (2015), “Deep learning”, Nature, 521(7553), 436–444.

- [23] LeCun Y., Bottou L., Bengio Y., Haffner P., (1998), “Gradient-based learning applied to document recognition”, *Proceedings of the IEEE*, 86(11), 2278–2324.
- [24] Goodfellow I., Bengio Y., Courville A., (2016), “Deep learning”, MIT Press.
- [25] Dumoulin V., Visin F., (2016), “A guide to convolution arithmetic for deep learning”, <https://arxiv.org/abs/1603.07285>, Accessed: 2025-05-20.
- [26] Ruder S., (2016), “An overview of gradient descent optimization algorithms”, <https://arxiv.org/abs/1609.04747>, Accessed: 2025-05-20.
- [27] Nair V., Hinton G. E., (2010), Rectified linear units improve restricted boltzmann machines, in “*Proceedings of the 27th International Conference on Machine Learning*”, Haifa, Israel, 807–814.
- [28] Bridle J. S., (1990), Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters, in “*Advances in Neural Information Processing Systems*”, Morgan Kaufmann, Denver, Colorado, USA, 211–217.
- [29] Ioffe S., Szegedy C., (2015), Batch normalization: Accelerating deep network training by reducing internal covariate shift, in “*Proceedings of the 32nd International Conference on Machine Learning (ICML 2015)*”, PMLR, Lille, France, 448–456.
- [30] Scherer D., Müller A., Behnke S., (2010), Evaluation of pooling operations in convolutional architectures for object recognition, in “*Proceedings of the 20th International Conference on Artificial Neural Networks (ICANN 2010)*”, Springer, Thessaloniki, Greece, 92–101.
- [31] Roy A., Siddiqui S., Pölsterl S., Navab N., Wachinger C., (2019), “Brain-torrent: A peer-to-peer environment for decentralized federated learning”, <https://arxiv.org/abs/1905.06731>, Accessed: 2025-05-20.
- [32] Liu L., Zhang J., Song S., Letaief K. B., (2020), Client-edge-cloud hierarchical federated learning, in “*Proceedings of the 2020 IEEE International Conference on Communications (ICC)*”, IEEE, Dublin, Ireland, 1–6.
- [33] Xie C., Koyejo S., Gupta I., (2019), “Asynchronous federated optimization”, <https://arxiv.org/abs/1903.03934>, Accessed: 2025-05-20.
- [34] Nishio T., Yonetani R., (2019), Client selection for federated learning with heterogeneous resources in mobile edge, in “*Proceedings of the 2019 IEEE International Conference on Communications (ICC)*”, IEEE, Shanghai, China, 1–7.
- [35] Bonawitz K., Eichner H., Grieskamp W., Huba D., Ingerman A., Ivanov V., Kiddon C., Konecny J., Mazzocchi S., McMahan H. B., et al. (2019), “Towards federated

learning at scale: System design”, <https://arxiv.org/abs/1902.01046>, Accessed: 2025-05-20.

- [36] Yang Q., Liu Y., Chen T., Tong Y., (2019), “Federated machine learning: Concept and applications”, *ACM Transactions on Intelligent Systems and Technology*, 10(2), 1–19.
- [37] Vepakomma P., Gupta O., Swedish T., Raskar R., (2018), “Split learning for health: Distributed deep learning without sharing raw patient data”, <https://arxiv.org/abs/1812.00564>, Accessed: 2025-05-20.
- [38] Konecny J., McMahan H. B., Yu F. X., Richtarik P., Suresh A. T., Bacon D., (2016), “Federated learning: Strategies for improving communication efficiency”, <https://arxiv.org/abs/1610.05492>, Accessed: 2025-05-20.
- [39] Wangni J., Wang J., Liu J., Zhang T., (2018), “Gradient sparsification for communication-efficient distributed optimization”, *Advances in Neural Information Processing Systems*, 31, 1306–1316.
- [40] Wang H., Sievert S., Liu S., Charles Z., Papailiopoulos D., Wright S., (2018), “Atomo: communication-efficient learning via atomic sparsification”, *Advances in Neural Information Processing Systems*, 31, 9872–9883.
- [41] Lin Y., Han S., Mao H., Wang Y., Dally W. J., (2017), “Deep gradient compression: reducing the communication bandwidth for distributed training”, <https://arxiv.org/abs/1712.01887>, Accessed: 2025-05-20.
- [42] Caldas S., Konecny J., McMahan H. B., Talwalkar A., (2018), “Expanding the reach of federated learning by reducing client resource requirements”, <https://arxiv.org/abs/1812.07210>, Accessed: 2025-05-20.
- [43] Tang H., Gan S., Zhang C., Zhang T., Liu J., (2018), “Communication compression for decentralized training”, *Advances in Neural Information Processing Systems*, 31, 7663–7673.
- [44] Bernstein J., Wang Y.-X., Azizzadenesheli K., Anandkumar A., (2018), Signsgd: Compressed optimisation for non-convex problems, in “Proceedings of the 35th International Conference on Machine Learning”, PMLR, Stockholmsmässan, Stockholm, Sweden, 560–569.
- [45] Wen W., Xu C., Yan F., Wu C., Wang Y., Chen Y., Li H., (2017), “Terngrad: ternary gradients to reduce communication in distributed deep learning”, *Advances in Neural Information Processing Systems*, 30, 1508–1518.
- [46] Alistarh D., Grubic D., Li J., Tomioka R., Vojnovic M., (2017), “Qsgd: communication-efficient sgd via gradient quantization and encoding”, *Advances in Neural Information Processing Systems*, 30, 1709–1720.

- [47] Jhunjhunwala D., Gadhikar A., Joshi G., Eldar Y. C., (2021), Adaptive quantization of model updates for communication-efficient federated learning, in “Proceedings of the 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)”, IEEE, Toronto, Ontario, Canada, 3110–3114.
- [48] Qu L., Song S., Tsui C.-Y., (2022), Feddq: Communication-efficient federated learning with descending quantization, in “Proceedings of the 2022 IEEE Global Communications Conference (GLOBECOM)”, IEEE, Rio de Janeiro, Brazil, 281–286.
- [49] Qu L., Song S., Tsui C.-Y., (2024), “Fedaq: Communication-efficient federated edge learning via joint uplink and downlink adaptive quantization”, <https://arxiv.org/abs/2406.18156>, Accessed: 2025-05-20.
- [50] Abadi M., Chu A., Goodfellow I., McMahan H. B., Mironov I., Talwar K., Zhang L., (2016), Deep learning with differential privacy, in “Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security”, ACM, Vienna, Austria, 308–318.
- [51] Iyengar R., Near J. P., Song D., Thakkar O., Thakurta A., Wang L., (2019), Towards practical differentially private convex optimization, in “Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP)”, IEEE, San Francisco, CA, USA, 299–316.
- [52] Agarwal N., Suresh A. T., Yu F. X. X., Kumar S., McMahan H. B., (2018), “Cpsgd: communication-efficient and differentially-private distributed sgd”, *Advances in Neural Information Processing Systems*, 31, 7564–7575.
- [53] Melis L., Danezis G., De Cristofaro E., (2015), “Efficient private statistics with succinct sketches”, <https://arxiv.org/abs/1508.06110>, Accessed: 2025-05-20.
- [54] Andrew G., Thakkar O., McMahan B., Ramaswamy S., (2021), “Differentially private learning with adaptive clipping”, *Advances in Neural Information Processing Systems*, 34, 17455–17466.
- [55] Fu J., Hong Y., Ling X., Wang L., Ran X., Sun Z., Wang W. H., Chen Z., Cao Y., (2024), “Differentially private federated learning: A systematic review”, <https://arxiv.org/abs/2405.08299>, Accessed: 2025-05-20.
- [56] Tariq A., Serhani M. A., Sallabi F. M., Barka E. S., Qayyum T., Khater H. M., Shuaib K. A., (2024), “Trustworthy federated learning: A comprehensive review, architecture, key challenges, and future research prospects”, *IEEE Open Journal of the Communications Society*, 5, 4920–4998.
- [57] Lang N., Sofer E., Shaked T., Shlezinger N., (2023), “Joint privacy enhancement and quantization in federated learning”, *IEEE Transactions on Signal Processing*, 71, 295–310.

- [58] Wang Y., Cao X., Jin S., Chow M.-Y., (2024), “A novel privacy enhancement scheme with dynamic quantization for federated learning”, <https://arxiv.org/abs/2405.16058>, Accessed: 2025-05-20.
- [59] Youn Y., Hu Z., Ziani J., Abernethy J., (2023), “Randomized quantization is all you need for differential privacy in federated learning”, <https://arxiv.org/abs/2306.11913>, Accessed: 2025-05-20.
- [60] Nguyen H. M., Chu N. H., Nguyen D. N., Hoang D. T., Nguyen V.-D., Ha M. H., Dutkiewicz E., Krunz M., (2023), “Optimal privacy preserving for federated learning in mobile edge computing”, <https://arxiv.org/abs/2211.07166>, Accessed: 2025-05-20.
- [61] Zhou Y., Wang R., Liu J., Wu D., Yu S., Wen Y., (2023), “Exploring the practicality of differentially private federated learning: a local iteration tuning approach”, *IEEE Transactions on Dependable and Secure Computing*, 21(4), 3280–3294.
- [62] Shapley L. S., (1953), A value for n-person games, in “Contributions to the theory of games”, Vol. II, Princeton University Press, Princeton, NJ, USA, 307–317.
- [63] Wang G., Dang C. X., Zhou Z., (2019), Measure contribution of participants in federated learning, in “Proceedings of the 2019 IEEE International Conference on Big Data”, IEEE, Los Angeles, CA, USA, 2597–2604.
- [64] Katharopoulos A., Fleuret F., (2018), Not all samples are created equal: Deep learning with importance sampling, in “Proceedings of the 35th International Conference on Machine Learning”, Vol. 80 of Proceedings of Machine Learning Research, PMLR, Stockholmsmässan, Stockholm, Sweden, 2525–2534.
- [65] Shyn S. K., Kim D., Kim K., (2021), “Fedccea: A practical approach of client contribution evaluation for federated learning”, <https://arxiv.org/abs/2106.02310>, Accessed: 2025-05-20.
- [66] Tuor T., Wang S., Ko B. J., Liu C., Leung K. K., (2021), Overcoming noisy and irrelevant data in federated learning, in “Proceedings of the 25th International Conference on Pattern Recognition”, IEEE, Milan, Italy, 5020–5027.
- [67] Luan Y., (2024), Network traffic anomaly detection based on federated learning, in “Proceedings of the 2024 4th International Conference on Neural Networks, Information and Communication Engineering”, IEEE, Guangzhou, China, 224–228.
- [68] Tran D. H., Nguyen V. L., Utama I. B. K. Y., Jang Y. M., (2022), An improved sensor anomaly detection method in iot system using federated learning, in “Proceedings of the 2022 13th International Conference on Ubiquitous and Future Networks”, IEEE, Barcelona, Spain, 466–469.
- [69] Kea K., Han Y., Min Y.-J., (2024), “A federated learning approach for ef-

ficient anomaly detection in electric power steering systems”, IEEE Access, 12, 67525–67536.

- [70] Li S., Cheng Y., Liu Y., Wang W., Chen T., (2019), “Abnormal client behavior detection in federated learning”, <https://arxiv.org/abs/1910.09933>, Accessed: 2025-05-20.
- [71] Kea K., Han Y., Kim T.-K., (2023), “Enhancing anomaly detection in distributed power systems using autoencoder-based federated learning”, PLoS ONE, 18(8), e0290337.
- [72] Bhat P., Pai M. M., Pai R. M., (2023), Anomaly detection using federated learning: A performance based parameter aggregation approach, in “Proceedings of the 2023 3rd International Conference on Intelligent Technologies”, IEEE, Hubli, India, 1–6.
- [73] Deng L., (2012), “The mnist database of handwritten digit images for machine learning research”, IEEE Signal Processing Magazine, 29(6), 141–142.
- [74] Krizhevsky A., Hinton G., (2009), Learning multiple layers of features from tiny images, Technical report, University of Toronto, Toronto, ON, Canada.
- [75] Netzer Y., Wang T., Coates A., Bissacco A., Wu B., Ng A. Y., et al. (2011), Reading digits in natural images with unsupervised feature learning, in “Proceedings of the NIPS Workshop on Deep Learning and Unsupervised Feature Learning”, Granada, Spain, 4.
- [76] Chrabaszcz P., Loshchilov I., Hutter F., (2017), “A downsampled variant of imagenet as an alternative to the cifar datasets”, <https://doi.org/10.48550/arXiv.1707.08819>, Accessed: 2025-05-20.
- [77] Cohen G., Afshar S., Tapson J., van Schaik A., (2017), Emnist: extending mnist to handwritten letters, in “Proceedings of the 2017 International Joint Conference on Neural Networks”, IEEE, Anchorage, Alaska, USA, 2921–2926.
- [78] Reddi S., Charles Z., Zaheer M., Garrett Z., Rush K., Konečný J., Kumar S., McMahan H. B., (2021), “Adaptive federated optimization”, <https://arxiv.org/abs/2003.00295>, Accessed: 2025-05-20.
- [79] Ghorbani A., Zou J., (2019), Data shapley: Equitable valuation of data for machine learning, in “Proceedings of the 36th International Conference on Machine Learning (ICML)”, PMLR, Long Beach, California, USA, 2242–2251.
- [80] Chang C.-C., Lin C.-J., (2011), “Libsvm: A library for support vector machines”, ACM Transactions on Intelligent Systems and Technology, 2(3), 27.
- [81] Liu F. T., Ting K. M., Zhou Z.-H., (2008), Isolation forest, in “Proceedings of the

2008 Eighth IEEE International Conference on Data Mining”, IEEE, Pisa, Italy, 413–422.

- [82] Wu H., Judd P., Zhang X., Isaev M., Micikevicius P., (2020), “Integer quantization for deep learning inference: Principles and empirical evaluation”, <https://arxiv.org/abs/2004.09602>, Accessed: 2025-05-20.
- [83] Wei K., Li J., Ding M., Ma C., Yang H. H., Farokhi F., Jin S., Quek T. Q. S., Poor H. V., (2020), “Federated learning with differential privacy: Algorithms and performance analysis”, IEEE Transactions on Information Forensics and Security, 15, 3454–3469.
- [84] Ma C., Li J., Ding M., Yang H. H., Shu F., Quek T. Q. S., Poor H. V., (2020), “On safeguarding privacy and security in the framework of federated learning”, IEEE Network, 34(4), 242–248.
- [85] Wang Z., Song M., Zhang Z., Song Y., Wang Q., Qi H., (2019), Beyond inferring class representatives: User-level privacy leakage from federated learning, in “Proceedings of the IEEE Conference on Computer Communications (INFOCOM 2019)”, IEEE, Paris, France, 2512–2520.
- [86] He C., Li S., So J., Zeng X., Zhang M., Wang H., Wang X., Vepakomma P., Singh A., Qiu H., Zhu X., Wang J., Shen L., Zhao P., Kang Y., Liu Y., Raskar R., Yang Q., Annamalai M., Avestimehr S., (2020), “Fedml: A research library and benchmark for federated machine learning”, <https://arxiv.org/abs/2007.13518>, Accessed: 2025-05-20.
- [87] Ardiç E., Genç Y., (2025), “Sample selection using multi-task autoencoders in federated learning with non-iid data”, Engineering Science and Technology, an International Journal, 61, 101920.
- [88] Plissiti M. E., Dimitrakopoulos P., Sfikas G., Nikou C., Krikoni O., Charchanti A., (2018), Sipakmed: A new dataset for feature and image-based classification of normal and pathological cervical cells in pap smear images, in “Proceedings of the 2018 25th IEEE International Conference on Image Processing (ICIP)”, IEEE, Athens, Greece, 3144–3148.
- [89] Kermany D., Zhang K., Goldbaum M., (2018), “Large dataset of labeled optical coherence tomography (oct) and chest x-ray images”, Mendeley Data, vol. 3, no. 10.17632, Accessed: 2025-05-20.
- [90] Spanhol F. A., Oliveira L. S., Petitjean C., Heutte L., (2016), “A dataset for breast cancer histopathological image classification”, IEEE Transactions on Biomedical Engineering, 63(7), 1455–1462.

BIOGRAPHY

Emre Ardiç is a Chief Researcher at the Scientific and Technological Research Council of Turkey (TÜBİTAK BİLGEM), specializing in deep learning. He has extensive professional experience and a strong academic background in computer engineering from Gebze Technical University, where he graduated at the top of his class in his undergraduate studies and earned high honors in his master's program. His research primarily focuses on deep learning, computer vision, and federated learning methodologies. Emre has held multiple roles at TÜBİTAK BİLGEM, where he has played a key role in developing sophisticated AI applications, including image enhancement, object detection, and tracking using cutting-edge technologies like PyTorch and TensorRT. Previously, he was responsible for designing secure, large-scale client-server systems and AI-driven forgery detection tools, notably the National Marker Controller for fuel sample detection and the FORDEX forensic document examination system. Emre has made academic contributions through publications on data valuation in federated learning and deep neural network classification techniques, and he co-authored the PHISTO bioinformatics tool for pathogen-host interaction analysis. Proficient in multiple programming languages and frameworks, his skillset includes software engineering, system optimization, and application development.

PUBLICATIONS WITHIN the SCOPE of THESIS STUDY

Ardıç, E. and Genç, Y., (2023), “Data Valuation Methods for Federated Learning,” in 31st Signal Processing and Communications Applications Conference (SIU), Istanbul, Türkiye, pp. 1–4, doi: 10.1109/SIU59756.2023.10223784.

Ardıç, E. and Genç, Y., (2025), “Sample Selection using Multi-Task Autoencoders in Federated Learning with Non-IID Data,” in Engineering Science and Technology, an International Journal, vol. 61, p. 101920, doi: 10.1016/j.jestch.2024.101920.

Ardıç, E. and Genç, Y., (2025), “Enhanced Privacy and Communication Efficiency in Non-IID Federated Learning with Adaptive Quantization and Differential Privacy,” in IEEE Access, vol. 13, pp. 54322–54337, doi: 10.1109/ACCESS.2025.3554138.

