From Algorithm to Application: Enhancing Federated Learning with Adaptive Aggregation and Gradient Protection

Yi Hu
[redacted]

Submitted to Swansea University in fulfilment
of the requirements for the Degree of Doctor of Philosophy

Swansea University
Prifysgol Abertawe

Department of Computer Science
Swansea University

March 14, 2025

Declarations

This work has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed..... [redacted] .....

Date..... 14/03/2025 .....

This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed..... [redacted] .....

Date..... 14/03/2025 .....

I hereby give consent for my thesis, if accepted, to be available for electronic sharing

Signed..... [redacted] .....

Date..... 14/03/2025 .....

The University's ethical procedures have been followed and, where appropriate, that ethical approval has been granted.

Signed..... [redacted] .....

Date..... 14/03/2025 .....

To my husband, Hanchi Ren, whose unwavering support and encouragement have been invaluable throughout my Ph.D. journey. Thank you, my love.

Abstract

Federated Learning (FL) has emerged as a promising paradigm for decentralised machine learning, allowing multiple clients to collaboratively train models without sharing their private data. This distributed approach is particularly relevant in domains with stringent data privacy requirements, such as finance, healthcare, and edge computing. However, despite its advantages, FL faces three critical challenges: (1) efficient model aggregation, (2) protection against privacy leakage, and (3) real-world applicability in complex domains. This thesis addresses these challenges by proposing novel strategies to optimise aggregation mechanisms, enhance its privacy protection capabilities, and explore its application in financial modelling.

First, we introduce Element-Wise Weights Aggregation for FL (EWWA-FL), a novel optimisation technique that improves global model convergence by adopting element-wise adaptive weighting. Traditional FL aggregation methods, such as FedAvg, assign a single proportion to each local model without considering the varying importance of individual model parameters. In contrast, EWWA-FL assigns unique proportions to each element within local model weights, ensuring more precise updates that account for dataset heterogeneity among clients. Experimental results demonstrate that EWWA-FL significantly improves both convergence speed and final model accuracy, outperforming FedAvg, FedOpt, and FedCAMS across multiple benchmark datasets. By incorporating an element-wise approach, EWWA-FL provides a more adaptive and fine-grained aggregation strategy that enhances FL's performance in both Independent and Identically Distributed (IID) and Non-IID settings.

Second, we propose AdaDefence, a privacy-preserving defence mechanism against gradient leakage attacks in FL. While FL eliminates the need for raw data sharing, recent attacks have demonstrated that an adversary can reconstruct private training data from shared gradients, posing a severe privacy risk. To counteract this, AdaDefence introduces a gradient stand-in approach, wherein local clients replace actual gradients with modified gradients before sending them to the server. This method prevents attackers from reconstructing private data while maintaining model utility. AdaDefence effectively defends against state-of-the-art attacks such

as Deep Leakage from Gradients (DLG), Generative Regression Neural Network (GRNN), and Inverting Gradient (IG) without significantly compromising model accuracy. Our extensive empirical analysis shows that AdaDefence provides strong privacy guarantees while ensuring minimal performance degradation, making it a practical and scalable solution for real-world FL deployments.

Finally, we explore the real-world application of FL in financial modelling, particularly in Cross-Stock Trend Integration (CSTI) for enhancing stock price prediction. Traditional financial models suffer from data fragmentation, where different financial institutions and stock markets operate in silos, limiting predictive power. To overcome this, we develop a FL-based approach that enables multiple financial institutions to collaboratively train stock prediction models without exposing sensitive trading data. This approach leverages cross-stock trend integration, allowing predictive models to learn patterns from multiple stocks while preserving privacy. Our experimental results demonstrate that federated cross-stock learning improves predictive accuracy and model robustness, outperforming conventional single-stock prediction methods. By enabling secure, multi-institution collaboration, this work highlights the potential of FL in advancing financial modelling while ensuring regulatory compliance and data confidentiality.

By addressing these fundamental aspects, optimisation and protection, this thesis makes significant contributions to the field of FL. The proposed methodologies collectively enhance FL's efficiency, security, and real-world applicability. Through EWWA-FL, FL models achieve faster and more reliable convergence. By introducing AdaDefence, FL gains stronger privacy protections against gradient-based attacks. Finally, by demonstrating FL's potential in cross-stock trend integration, this thesis showcases how FL can be deployed in privacy-sensitive financial applications. These contributions pave the way for more efficient, secure, and scalable FL systems, advancing their adoption in a wide range of domains, including healthcare, autonomous systems, and financial technology.

Acknowledgements

The journey of my PhD has been one of the toughest yet most rewarding experiences of my life. There were countless moments of doubt, exhaustion, and frustration, but what kept me going was the unwavering support of the people around me. Without my family and my supervisors, I don't think I would have made it to the finish line. Their encouragement, patience, and sacrifices carried me through the hardest days, and looking back now, I know I couldn't have done it alone.

In the second year of my PhD, my life changed in the most profound way, I became a mother. It was a moment of overwhelming joy, but also one of immense challenge. My son was fragile and needed constant care, and suddenly, the already demanding task of doing a PhD became twice as hard. When I returned from maternity leave, I felt lost. The research field had moved forward without me, and catching up felt almost impossible. There were nights when I questioned whether I could still do this, whether I had the energy to keep going.

In those moments of doubt, my supervisor, Dr. Jingjing Deng, and Prof. Xianghua Xie, became my guiding lights. They didn't just offer academic advice. They were patient, understanding, and incredibly supportive. No matter how busy they were, they always made time to help me find my footing again. Dr. Jingjing Deng even helped care for my son at times, allowing me to focus on my work when I needed it the most. Their kindness and belief in me reminded me that I wasn't alone, that I had people in my corner, cheering me on.

I owe so much to my parents and my sister, who travelled across great distances to help me with my son. Their support wasn't just practical, it was emotional. Knowing they were there, taking care of my child, gave me the freedom to focus on my research without constant worry. No matter what struggles I faced, they always reassured me that I wasn't alone, and their love gave me the strength to keep going.

Among all the people who have supported me, the one I am most grateful for is my husband. Despite having his own demanding career, he never hesitated to be there for me. He stayed up

late with me to proofread my thesis, had long discussions with me about my experiments, and was always ready to listen, offering both intellectual and emotional support. When I struggled with postpartum depression, he was my quiet protector, always patient, always gentle, never letting me feel like I had to face it alone. He reminded me that challenges are just part of life's journey, and one day, I would look back and see how they had made me stronger. His belief in me never wavered, even when my own confidence did.

But the greatest source of strength throughout has been my son. On the hardest days, when I was overwhelmed by stress and uncertainty, his laughter reminded me why I was pushing forward. His tiny hands reaching for mine, his peaceful face as he fell asleep beside me. Those moments grounded me. They reminded me that this PhD wasn't just about research; it was about resilience, love, and the life I was building for my family. His unexpected arrival in the middle of my studies didn't derail my journey, but became the most beautiful part of it.

And finally, I want to acknowledge myself. There were so many moments when I could have given up, when exhaustion and self-doubt almost won. But I kept going. This PhD is not just an academic achievement, it's proof of my strength, my determination, and my ability to push through even the hardest days.

This thesis is not just the result of research and hard work. It is a reflection of the love and support of the incredible people in my life. To all of them, from the bottom of my heart. Thank you!

Contents

List of Acronyms

AI	Artificial Intelligence
ARIMA	Autoregressive Integrated Moving Average
BERT	Bidirectional Encoder Representations from Transformers
BN	Batch Normalisation
CE	Cross Entropy
CMA-ES	Covariance Matrix Adaptation Evolution Strategy
CNN	Convolutional Neural Network
CSTI	Cross-Stock Trend Integration
DFT	Discrete Fourier Transform
DL	Deep Learning
DLG	Deep Leakage from Gradients
DNN	Deep Neural Network
DP	Differential Privacy
DT	Decision Tree
EWWA-FL	Element-Wise Weights Aggregation Method for FL
FedAdp	Federated Adaptive Weighting
FedAvg	Federated Averaging
FedBoosting	Federated Boosting
FFN	Feed-Forward Network
FFT	Fast Fourier Transform

FL	Federated Learning
FNSPID	Financial News and Stock Price Integration Dataset
GAN	Generative Adversarial Network
GDPR	General Data Protection Regulation
GGL	Generative Gradient Leakage
GLAUS	Gradient Leakage Attack Using Sampling
GPT	Generative Pre-trained Transformers
GRNN	Generative Regression Neural Network
GRU	Gated Recurrent Unit
HE	Homomorphic Encryption
HIPAA	Health Insurance Portability and Accountability Act
iDLG	Improved DLG
IG	Inverting Gradient
IID	Independent and Identically Distributed
IoT	Internet of Things
KNN	K-Nearest Neighbour
LPIPS	Learned Perceptual Image Patch Similarity
LSTM	Long Short-Term Memory
MAPE	Mean Absolute Percentage Error
MHSA	Multi-Head Self-Attention
ML	Machine Learning
MLP	Multi-Layered Perceptron
MMS	MinMax Sampling
MPC	Secure Multi-Party Computation
MSE	Mean Square Error
NLP	Natural Language Processing
non-IID	Non-Independent and Identically Distributed

PRECODE	Privacy Enhancing Module
PSNR	Peak Signal-to-Noise Ratio
ReLU	Rectified Linear Unit
RF	Random Forest
RL	Reinforcement Learning
RNN	Recurrent Neural Network
SGD	Stochastic Gradient Descent
SSIM	Structure Similarity Index Measure
SVM	Support Vector Machine
UGS	Unbiased Gradient Sampling-based
YOLO	You Only Look Once

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivations

Artificial Intelligence (AI) has become a cornerstone of modern computing, driving automation, intelligent decision-making, and predictive analytics across areas such as healthcare, finance, autonomous systems, and edge computing. The rapid growth of AI applications has been enabled by access to large-scale datasets, high-performance computing, and advances in Deep Learning (DL) architectures. Yet, as these systems increasingly rely on distributed and sensitive data, concerns about privacy, security, and computational efficiency have grown more pressing. Federated Learning (FL) [1–6] has emerged as a promising response. It allows multiple parties (such as mobile devices, hospitals, or financial institutions) to train shared models while keeping their data local. Unlike traditional centralised Machine Learning (ML), which requires raw data to be pooled into a single server, FL shares only model updates (gradients or weights). This not only reduces communication costs but also helps organisations comply with strict regulations such as the General Data Protection Regulation (GDPR) in Europe and Health Insurance Portability and Accountability Act (HIPAA) in the United States. Still, deploying FL in practice is far from straightforward. Real-world systems face three persistent challenges: (1) aggregating models effectively when client data are highly diverse, (2) protecting against privacy risks hidden in shared gradients, and (3) showing that federated approaches can work in demanding domains such as finance. These three challenges form the backbone of this thesis, which is guided by a single overarching goal, making FL more **secure, efficient, and practical for real-world deployment**. To achieve this, the work in this thesis follows three directions. First, it proposes a fine-grained element-wise aggregation method that adapts to non-IID data

and accelerates model convergence. Second, it introduces a defence mechanism that prevents gradient leakage attacks without the heavy cost of encryption or the accuracy loss seen in noise-based methods. Third, it demonstrates how FL can be applied to financial modelling through a federated cross-stock prediction framework, enabling institutions to collaborate while respecting data confidentiality. Taken together, these contributions connect theory and practice by improving the efficiency of federated training, strengthen its privacy guarantees, and show its value in a complex real-world application. The aim is not just to solve isolated problems, but to move FL closer to being a trustworthy technology that can be deployed with confidence.

Topic 1: Model Aggregation in Heterogeneous Settings

A central limitation in FL is the way local model updates from diverse clients are aggregated. In practice, data across clients is rarely Independent and Identically Distributed (IID). It often reflects regional biases, user-specific behaviour, class imbalance, or domain-specific features. This heterogeneity has a direct impact on global updates. The widely used Federated Averaging (FedAvg) [1] assumes that all clients contribute equally and applies a single weight to each local model, regardless of the quality or representativeness of the data. While this simplifies training, it does not reflect real-world conditions. As a result, global models trained under uniform weighting frequently converge slowly, generalise poorly across clients, and can become unstable when client updates diverge significantly. Clients with skewed data distributions or small datasets may even distort the aggregated model, reducing its robustness and fairness.

To address these issues, researchers have proposed adaptive aggregation methods that assign different weights to each client based on criteria such as data volume, similarity of distributions, or update divergence [7–9]. These methods mark an important step forward, but they still treat the model as a whole that each client's update is assigned a single scalar weight. This coarse adjustment overlooks the fact that not all parameters within a model contribute equally to learning. Parameters tied to well-represented classes or stable features may carry more useful information, while others may be noisy or biased by local peculiarities.

This raises the core research question of Chapter 3 that can aggregation be improved by moving beyond model-level weighting to parameter-level weighting, where each parameter is evaluated individually for its contribution to global learning. Our answer is the proposed **Element-Wise Weights Aggregation Method for FL (EWWA-FL)**, which departs from model-wise weighting schemes. Instead of assigning a single weight per client, it assigns adaptive weights to each parameter across local models. This fine-grained approach allows the global update to down-weight parameters distorted by Non-Independent and Identically

Distributed (non-IID) effects and amplify those that are consistent and reliable across clients. Hence, EWWA-FL aims to achieve faster convergence, reduced bias, and more stable training in heterogeneous and privacy-preserving environments.

Topic 2: Defending Gradient Leakage Attacks

FL addresses a fundamental privacy concern by avoiding raw data exchange, since training occurs locally on client devices and only model updates are shared. While this provides stronger privacy than centralised ML, it does not make FL immune to attack. In particular, gradient leakage has emerged as a serious vulnerability. By analysing the gradients sent to the server, adversaries can reconstruct sensitive inputs with surprising accuracy. Methods such as Deep Leakage from Gradients (DLG) [10], Improved DLG (iDLG) [11], Inverting Gradient (IG) [12], Generative Regression Neural Network (GRNN) [13], and Generative Gradient Leakage (GGL) [14] demonstrate that both images and labels can be faithfully recovered from shared updates. These findings raise significant concerns in domains such as healthcare and finance, where any leakage of underlying records may have legal or ethical consequences.

Existing defence strategies attempt to mitigate this risk but fall short in practice. Differential Privacy (DP) offers formal privacy guarantees by adding noise to the gradients, resulting in accuracy reduction, especially under tight privacy budgets. Homomorphic Encryption (HE) protects confidentiality by encrypting updates end-to-end, but its computational and communication costs are prohibitive for large models and resource-limited edge devices. In other words, one approach sacrifices utility, while the other sacrifices practicality.

This motivates the research question at the core of chapter 4 that can we design a defence that conceals sensitive information in gradients without incurring the accuracy loss of noise injection or the heavy costs of encryption. The proposed answer is **AdaDefence**, which introduces the idea of gradients stand-in. Instead of sending raw gradients to the server, each client replaces them with modified gradients derived through the Adam optimiser's first-order and second-order moment estimates. These stand-ins preserve enough learning signal for effective model training but obscure the direct mapping between gradients and private data. AdaDefence achieves a stronger balance in the privacy–utility trade-off. It prevents adversaries from reconstructing inputs while avoiding the performance drop typical of DP and the overhead of HE. The aim is to make FL both secure and practical for deployment in real-world, privacy-sensitive environments.

Topic 3: Real-World Applicability in Financial Modelling

FL has become a promising paradigm in ML, especially for applications where privacy and regulatory compliance are paramount. Its decentralised architecture addresses a long-standing

tension in data science, which is how to train accurate, generalisable models without exposing sensitive or proprietary data. This challenge is particularly acute in domains such as healthcare and finance, where strict legal frameworks restrict how data can move across institutions or jurisdictions. Although these sectors rely heavily on data-driven insights for decision-making, concerns about confidentiality and compliance often prevent them from fully exploiting their data resources.

In finance, the case for adopting FL is compelling. Banks, investment firms, insurance companies, and stock exchanges generate enormous volumes of trading records, transaction histories, and market signals. These datasets contain valuable predictive patterns, but they are rarely shared due to competitive sensitivities and regulatory oversight. As a result, conventional ML approaches that rely on centralised aggregation cannot be applied effectively. FL helps overcome this barrier by enabling institutions to train models collaboratively without moving raw data, offering new opportunities for fraud detection, cross-institutional forecasting, and algorithmic trading while preserving compliance and confidentiality.

However, even with FL, most existing approaches in finance still treat each institution's data in isolation, limiting the scope of what can be learned. Standard federated models tend to capture only local patterns, missing the broader market relationships that emerge when signals from multiple stocks or institutions are considered together. This leads to a precise research problem that how can federated models move beyond isolated, stock-specific learning to capture cross-stock dependencies that improve predictive performance. This motivates the development of Cross-Stock Trend Integration (CSTI). The idea is to integrate trends across different stocks during the federated training process, so that the global model can learn shared structures in financial time series while still respecting institutional privacy. CSTI goes beyond standard FL by allowing predictive models to exploit correlations across assets and markets, which conventional federated approaches overlook. In the end, CSTI provides two novel insights. First, privacy-preserving collaboration can uncover richer financial signals than any institution can achieve alone. Second, incorporating cross-stock information enhances both the accuracy and robustness of forecasting models. This contribution demonstrates that FL in finance need not be limited to protecting data silos but can actively generate new knowledge through cross-stock integration.

1.2 Overview and Contributions

This thesis presents novel methods to adaptive aggregation, gradient protection, and application of FL in real-world settings. The key contributions of this thesis are as follows:

1.2.1 Optimising Model Aggregation: EWWA-FL

The success of FL relies on an effective model aggregation strategy that can efficiently combine local model updates from multiple clients into a robust global model. Traditional FL aggregation methods, such as FedAvg, treat all model parameters uniformly, applying a single proportion across all local updates without considering the heterogeneity of client data. However, real-world FL applications often involve non-IID datasets, where clients train on distinct data distributions. Such heterogeneity causes significant weight divergence, leading to slow convergence, reduced generalisation, and unstable global model performance. To address these limitations, we introduce EWWA-FL in Chapter 3, a novel fine-grained aggregation technique that assigns individual proportions to each model parameter based on its significance in global learning. Unlike FedAvg, FedOpt, and FedCAMS, which aggregate entire models without distinction, EWWA-FL dynamically evaluates each parameter's contribution and adjusts its aggregation weight accordingly. This adaptive element-wise approach ensures that critical parameters receive higher aggregation priority, accelerating convergence and enhancing model performance. Our experimental results demonstrate that EWWA-FL offers substantial improvements in both IID and non-IID settings, outperforming conventional aggregation methods in terms of:

- Faster convergence: By optimising aggregation weights at a fine-grained level, EWWA-FL reduces the number of global communication rounds required for model convergence.
- Higher accuracy: The adaptive nature of EWWA-FL ensures that the global model captures diverse data patterns, improving overall predictive performance.
- Robustness to data heterogeneity: By assigning appropriate weights to local updates, EWWA-FL effectively mitigates weight divergence, making it well-suited for FL applications with highly non-IID data distributions.

These advantages make EWWA-FL a practical and scalable solution for real-world FL deployments, particularly in healthcare, finance, and edge computing, where data heterogeneity is a major challenge. Below is the publication for this chapter:

- Hu, Y., Ren, H., Hu, C., Deng, J., & Xie, X. (2023, December). An Element-Wise Weights Aggregation Method for Federated Learning. In 2023 IEEE International Conference on Data Mining Workshops (ICDMW) (pp. 188-196). IEEE.

1.2.2 Gradient Protection: AdaDefence for Gradient Leakage Defence

While FL is designed to protect user privacy by keeping data local, recent research has revealed that shared model updates (gradients) can still leak sensitive information about the underlying training data. Gradient leakage attacks, such as DLG, iDLG, GRNN, GGL, and IG, allow adversaries to reconstruct high-resolution training images and recover class labels directly from gradient updates. This vulnerability poses a critical threat to privacy-sensitive FL applications, such as medical diagnostics, facial recognition, and financial modelling. Existing privacy-preserving techniques, such as DP and HE, attempt to mitigate gradient leakage but suffer from severe trade-offs:

- DP-based approaches introduce random noise to gradients, but excessive noise can degrade model accuracy, leading to significant performance loss.
- HE-based encryption methods offer strong security guarantees but incur high computational overhead, making them impractical for large-scale FL systems with limited computing resources.

To overcome these challenges, we propose AdaDefence in Chapter 4, a novel gradients stand-in mechanism that replaces actual gradients with modified versions before they are shared with the global server. AdaDefence achieves strong privacy protection without introducing excessive computational costs by:

- Obfuscating private training data: The transformation of gradients prevents attackers from reconstructing sensitive information while preserving model utility.
- Maintaining high model accuracy: Unlike DP, which injects noise, AdaDefence ensures that the aggregated gradients retain meaningful learning signals.
- Minimal computational overhead: Compared to HE-based solutions, AdaDefence operates efficiently without requiring complex encryption schemes.

Our empirical evaluation demonstrates that AdaDefence successfully defends against DLG, GRNN and IG attacks while achieving better accuracy and lower computational cost than

traditional privacy-preserving techniques. This makes it an effective and practical defence mechanism for deploying FL in privacy-critical applications. The completed paper for this chapter can be found:

- Yi, H., Ren, H., Hu, C., Li, Y., Deng, J., & Xie, X. (2024). Gradients Stand-in for Defending Deep Leakage in Federated Learning. Computer Vision. IET. Under-reviewing

1.2.3 Applying FL to Financial Modelling: CSTI

The financial sector increasingly relies on ML for tasks such as stock price prediction, risk assessment, and fraud detection. However, financial institutions are subject to strict regulatory constraints that prohibit the sharing of proprietary market data. This fragmentation of financial datasets limits the predictive power of stock models, as they lack access to broader market trends and correlations across institutions. To address this limitation, we develop a FL-based financial modelling framework in Chapter 5, enabling multiple financial institutions to collaboratively train predictive models while preserving data privacy. Our approach, CSTI, allows stock prediction models to learn from multiple sources without direct data exchange, leveraging FL to improve financial forecasting accuracy. The key benefits of CSTI include:

- Enhanced predictive accuracy: By integrating cross-stock trends, our approach captures broader market patterns, outperforming traditional single-stock models.
- Privacy preservation: Financial data remains on local servers, ensuring compliance with regulatory requirements such as GDPR and HIPAA.
- Scalability and adaptability: Our FL framework can be easily extended to various financial applications, including portfolio optimisation and algorithmic trading.

Our empirical analysis demonstrates that federated cross-stock learning achieves significantly higher accuracy than stand-alone stock models while maintaining data confidentiality. This work highlights the potential of FL in privacy-sensitive financial applications, paving the way for secure and collaborative ML in the financial industry. The details of the paper from this chapter is:

- Yi, H., Ren, H., Deng, J., & Xie, X. (2025). From Local Patterns to Global Understanding: Cross-Stock Trend Integration for Enhanced Predictive Modeling. Ready for submitting.

1.2.4 Summary of Contributions

To summarise, this thesis makes the following key contributions:

- Adaptive Aggregation: We propose EWWA-FL, an element-wise aggregation method that improves global model convergence, robustness, and adaptability to non-IID data distributions.
- Gradient Protection: We introduce AdaDefence, a novel privacy-preserving approach that prevents gradient leakage while maintaining model utility, outperforming existing defences.
- Application: We develop CSTI, a FL-based framework for stock prediction, demonstrating how FL can be effectively applied in the financial sector.

These contributions advance the field of FL by enhancing key topics in model aggregation, gradient protection, and real-world application. By bridging the gap between theoretical advancements and practical deployments, this research provides a foundation for more efficient, secure, and scalable FL systems in the future.

Chapter 2

Background

2.1 Introduction

FL has emerged as a decentralised ML paradigm that enables multiple clients to collaboratively train a shared model while keeping their data local. By transmitting model updates instead of raw data, FL facilitates privacy-aware collaboration across data silos and is well-suited to domains such as healthcare, finance, and edge computing. Real-world deployment of FL has three major topics: a. model aggregation in heterogeneous settings, b. defending gradient leakage attacks, and c. applicability in regulatory-constrained domains like finance. This chapter presents the background required to understand these topics and motivates the contributions of this thesis. Section 2.2 introduces the foundational concepts in ML and DL, highlighting key architectures such as Convolutional Neural Network (CNN)s used in Chapter 3 & 4, Long Short-Term Memory (LSTM)s and transformers used in Chapter 5. Section 2.3 provides an overview of FL, its core mechanisms, and the problem of non-IID data. Section 2.4 examines gradient leakage attacks that threaten data privacy in FL. Section 2.5 discusses existing defence mechanisms and their limitations. Together, these sections provide a comprehensive foundation for the research presented in subsequent chapters.

2.2 Machine Learning and Deep Learning

ML and DL are two major subfields of AI that have revolutionised data-driven decision-making across various domains, including finance, healthcare, and autonomous systems. ML encompasses a broad spectrum of algorithms designed to learn patterns from data and make predictions

without explicit programming [15, 16]. The primary objective of ML is to enable systems to improve their predictive performance based on experience, rather than relying on hardcoded rules.

Traditional ML techniques can be broadly categorised into three learning paradigms: supervised learning, unsupervised learning, and reinforcement learning. In supervised learning, models are trained using labelled datasets, where each input sample is associated with a corresponding target output. Popular algorithms in this category include Decision Tree (DT)s, Support Vector Machine (SVM)s, K-Nearest Neighbour (KNN), and ensemble methods such as random forests and gradient boosting. These models are widely used for tasks such as classification and regression. Unsupervised learning, in contrast, operates on unlabelled data, aiming to discover hidden patterns or structures without explicit supervision. Clustering algorithms such as k-means, hierarchical clustering, and density-based spatial clustering are commonly employed for this purpose. These techniques find applications in market segmentation, anomaly detection, and recommendation systems. Reinforcement Learning (RL) [17] is another branch of ML where agents learn optimal actions by interacting with an environment and receiving feedback in the form of rewards or penalties. RL has shown success in fields such as robotics, game playing, and financial trading. Despite their effectiveness, traditional ML approaches have limitations, particularly when dealing with high-dimensional, unstructured data such as images, audio, and text. These models typically require manual feature engineering, which involves domain expertise to extract relevant features from raw data. The reliance on handcrafted features often limits scalability and adaptability to diverse tasks.

To overcome these limitations, DL has emerged as a powerful subset of ML that leverages Deep Neural Network (DNN)s to automatically extract hierarchical features from raw data. The hierarchical structure of DNNs enables automatic feature extraction, eliminating the need for manual feature engineering [18, 19]. DL models consist of multiple layers of interconnected neurons, where each layer captures increasingly complex representations of the input data. The adoption of DL has led to significant breakthroughs in various domains. In computer vision, CNNs have demonstrated state-of-the-art performance in image classification [20, 21], object detection [22], and image segmentation [23]. CNNs leverage spatial hierarchies in images by applying convolutional filters, allowing them to detect edges, textures, and complex patterns at different levels of abstraction. In speech recognition, Recurrent Neural Network (RNN)s and their variants, such as LSTM networks, have enabled accurate speech-to-text transcription [24]. By capturing temporal dependencies in audio signals, these models have facilitated

advancements in virtual assistants, automatic transcription systems, and multilingual speech processing. Natural Language Processing (NLP) has also witnessed a transformation with the advent of transformer-based architectures [25], such as Bidirectional Encoder Representations from Transformers (BERT) [26] and Generative Pre-trained Transformers (GPT) [27]. Unlike traditional sequence models, transformers utilise self-attention mechanisms to model long-range dependencies efficiently. These models have significantly improved tasks such as machine translation, sentiment analysis, text summarisation, and question answering.

While DL offers remarkable advantages in feature learning and performance scalability, it also presents challenges. Training DNNs requires large-scale labelled datasets and substantial computational resources. The training process is often computationally expensive, relying on hardware accelerators such as GPUs and TPUs. Furthermore, DL models are susceptible to over-fitting, where they memorise training data rather than generalising to unseen samples. Regularisation techniques, including dropout [20], Batch Normalisation (BN), and weight decay, are commonly employed to mitigate this issue. Another significant challenge in DL is interpretability. Unlike traditional ML models, which offer explainability through decision rules or feature importance scores, DNNs function as black-box models, making it difficult to understand their internal decision-making processes. This lack of interpretability raises concerns in critical applications such as healthcare and finance, where model decisions impact human lives and regulatory compliance. Despite these challenges, ongoing research in DL continues to push the boundaries of AI-driven automation. Hybrid approaches combining ML and DL are gaining traction, leveraging the strengths of both paradigms. Techniques such as transfer learning [28] allow pre-trained DL models to be fine-tuned for specific tasks, reducing data and computational requirements. Additionally, advancements in FL enable privacy-preserving training of DL models across distributed devices without exposing raw data, making it a promising solution for real-world applications in edge computing and personalised AI.

The integration of ML and DL has facilitated transformative progress in various industries. In finance, predictive analytics powered by DL enhances risk assessment [29], fraud detection [30], and algorithmic trading [31]. In healthcare, DL-based medical imaging systems aid in diagnosing diseases such as cancer and diabetic retinopathy [32] with high accuracy. Autonomous systems, including self-driving cars [33] and robotic control [34], benefit from DL's ability to process sensor data in real time. As the field of AI continues to evolve, the synergy between ML and DL is expected to drive further innovation. With increasing accessibility to large-scale datasets and advancements in hardware acceleration, DL will continue to play a

crucial role in shaping the future of intelligent systems.

2.2.1 CNN

CNNs have become the standard architecture for image classification and related vision tasks [35, 36]. Their key strength lies in the convolutional layer, which extracts hierarchical spatial features such as edges, textures, and shapes, enabling models to generalise effectively in complex visual domains. Pooling layers reduce redundancy and improve robustness, while fully connected layers and Softmax output layers complete the classification pipeline. In Chapters 3 and 4, CNNs serve as the backbone for experiments on federated aggregation and privacy defence. Since both EWWA-FL and AdaDefence are evaluated on image classification benchmarks, CNNs provide the foundation on which the effects of aggregation and gradient protection can be observed.

2.2.2 LSTM Network

LSTM networks [37] extend recurrent architectures by incorporating gating mechanisms to retain long-term dependencies in sequential data. This makes them highly effective for tasks such as time-series forecasting, speech recognition, and anomaly detection, where the order of data points matters. In Chapter 5, LSTMs are employed to capture temporal dependencies in financial time-series data. Their ability to model long-range patterns makes them a natural choice for stock prediction tasks, and they form part of the baseline models used to evaluate the effectiveness of cross-stock trend integration.

2.2.3 Transformer and Self-Attention Mechanism

The transformer architecture [25] revolutionised sequence modelling by replacing recurrence with self-attention, allowing global dependencies to be captured in parallel. Multi-head attention further enhances contextual representation, while positional encodings provide temporal ordering. Transformers have since achieved state-of-the-art performance in natural language processing, speech, and time-series forecasting. In Chapter 5, transformers are applied to financial time-series forecasting. Their ability to capture long-range dependencies complements LSTM-based models and highlights how CSTI can integrate cross-stock information within both recurrent and attention-based sequence frameworks.

2.3 Federated Learning

FL is a distributed ML paradigm that enables multiple clients to collaboratively train a global model while keeping their data stored locally. As shown in Figure 2.1, this decentralised approach contrasts with traditional centralised learning, where all data is transmitted to a central server for training. By ensuring that raw data remains on the client devices and only model updates, such as gradients or weights, are shared, FL enhances data privacy and security, making it particularly suitable for applications in healthcare, finance, and mobile edge computing. The architecture of FL aligns with privacy regulations such as the GDPR and HIPAA, which impose strict guidelines on data sharing and user consent. The standard FL framework consists of a central server and multiple participating clients. Each client independently trains a local model using its private dataset and periodically transmits model updates to the central server. The server aggregates these updates and refines the global model, which is then sent back to the clients for further local training. This iterative process continues until the model converges or a predefined performance threshold is achieved.

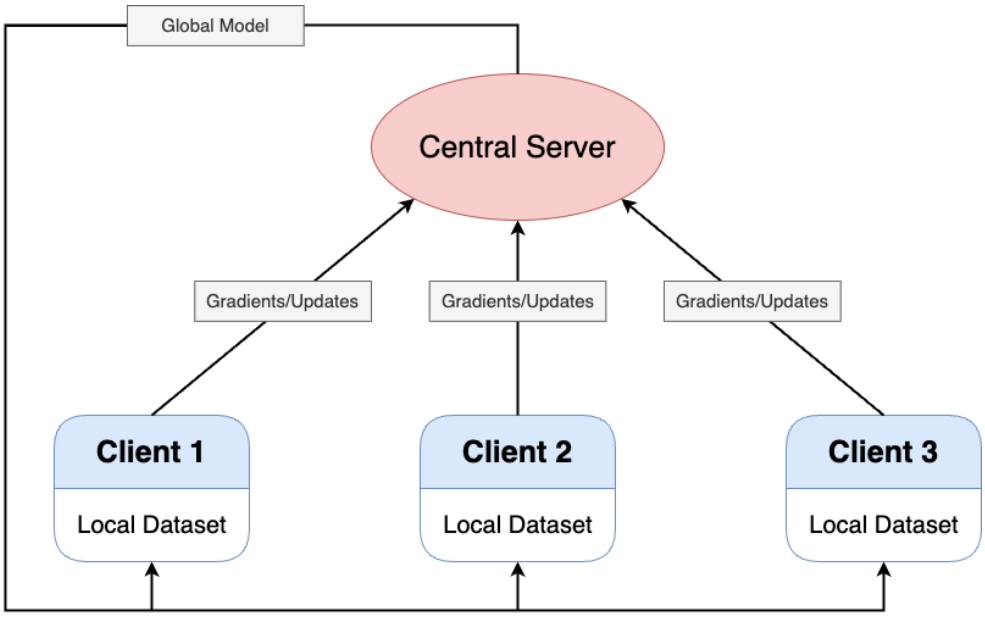


Figure 2.1: Illustration of the FL Framework. Clients perform local training on their private datasets and share only model updates with the central server, which aggregates them to form a global model for next round of local training.

The most widely adopted FL aggregation method is FedAvg, introduced by McMahan et al. [4]. FedAvg updates the global model parameters by computing a weighted average

of the local model updates received from participating clients. The weight assigned to each client's update is proportional to the number of local training samples, ensuring that clients with larger datasets have a greater influence on the global model. Mathematically, the global model parameters θ^t at training round t are updated as follows:

$$\theta^t = \sum_{i=1}^{N} \frac{n_i}{N} \theta_i^t, \tag{2.1}$$

where N is the total number of clients, n_i represents the number of local training samples for client i, and θ_i^t denotes the model parameters trained on client i at round t. During each communication round, the global model θ^t is distributed to all participating clients, who perform local updates based on their dataset. Each client updates its local model using Stochastic Gradient Descent (SGD) or other optimisation algorithms, which can be expressed as:

$$\theta_i^{t+1} = \theta^t - \eta \nabla L_i(\theta^t), \tag{2.2}$$

where η is the learning rate and $\nabla L_i(\theta^t)$ represents the gradient of the local loss function computed on client i. The locally updated parameters θ_i^{t+1} are then sent back to the server, where FedAvg performs aggregation to obtain the new global model.

One of the fundamental challenges in FL is the presence of non-IID data across clients. In real-world scenarios, clients often possess datasets with distinct feature distributions, label distributions, and sample sizes. This heterogeneity can lead to weight divergence, where locally trained models become significantly different from each other, resulting in poor generalisation of the global model. The performance degradation caused by non-IID data can be attributed to several factors. First, when different clients have imbalanced class distributions, the local models learn biased representations, which can negatively affect the global model's accuracy. Second, variations in feature distributions across clients may cause local models to converge to different optima, making aggregation less effective. Finally, clients with limited computational resources may perform fewer training iterations, exacerbating the problem of inconsistent model updates. These issues motivate the need for more fine-grained and adaptive aggregation, which is addressed in this thesis by **EWWA-FL** (Chapter 3). By weighting parameters individually rather than assigning a single scalar per client, EWWA-FL stabilises training and improves convergence in heterogeneous settings.

Another critical challenge is privacy leakage. Although FL avoids raw data exchange, gradients themselves have been shown to reveal sensitive information through reconstruction attacks [10, 12, 13]. To address this, differential privacy (via noise injection) and homomorphic

encryption have been proposed, but both have a limitation that noise often reduces accuracy, while encryption introduces prohibitive computational costs. To bridge this gap, this thesis proposes **AdaDefence** (Chapter 4), which replaces raw gradients with obfuscated "gradients stand-in" derived from Adam's moment estimates. This method conceals private information while retaining training utility, offering a more practical defence in federated settings.

Communication efficiency and scalability also remain significant hurdles. Frequent exchanges of updates between server and clients create high bandwidth demands, especially in mobile and edge networks. Model compression, quantisation, and sparsification reduce overhead, but they do not fully exploit the potential of collaborative training in domain-specific scenarios. In finance, for example, conventional FL approaches treat each institution's data silo independently, overlooking correlations across markets and assets. This motivates the contribution of **CSTI** (Chapter 5), which integrates cross-stock signals during FL training. By capturing dependencies across institutions without increasing communication load, CSTI demonstrates how FL can move from generic benchmarks to real-world financial forecasting.

In summary, while FL enables privacy-preserving collaborative learning, it faces persistent challenges: convergence under non-IID data, leakage of private information through gradients, and communication burdens in large-scale, domain-specific applications. The three core contributions of this thesis, including EWWA-FL, AdaDefence, and CSTI, are designed to tackle these issues, advancing the overarching aim of secure and efficient FL for real-world deployment.

2.4 Gradient Leakage Attacks

FL is designed to enhance data privacy by decentralising model training, ensuring that raw data remains on local devices. However, despite eliminating the need for direct data sharing, FL is not inherently immune to security risks. Model updates, particularly gradients exchanged between clients and the central server, can inadvertently reveal private training data. Attackers can exploit these shared gradients to reconstruct sensitive input samples, leading to privacy violations. One of the most significant threats in this domain is gradient leakage, where adversaries use mathematical optimisation techniques to infer the original training data from the gradients transmitted during the FL process [10–14]. Gradient leakage occurs because gradients encode information about the training data that generated them. During back-propagation, the computed gradients reflect the loss function's sensitivity with respect to the model's parameters. These gradients are directly influenced by the input data and its corresponding labels. If an attacker gains access to the gradients, it becomes possible to reverse-engineer the original input

data by solving an inverse problem that minimises the difference between the computed and observed gradients.

DLG [10] is one of the most well-known attacks that successfully reconstructs input data from shared gradients. In this method, an attacker initiates the reconstruction process by generating a random noise vector as a substitute for the original data. This noise vector is iteratively optimised using gradient descent to minimise the difference between the observed gradients from the original training data and the gradients computed from the generated noise. Mathematically, this optimisation process can be expressed as follows:

$$\hat{x}, \hat{y} = \arg\min_{x,y} \|\nabla_\theta L(f_\theta(x), y) - \nabla_\theta L(f_\theta(x_t), y_t)\|^2, \tag{2.3}$$

where $\hat{x}$ and $\hat{y}$ are the reconstructed input and label, x_t and y_t represent the original training sample and label, $f_\theta(x)$ denotes the model's output, and L is the loss function. The optimisation seeks to iteratively refine $\hat{x}$ and $\hat{y}$ so that the computed gradients align closely with the original gradients. Once this alignment is achieved, the reconstructed data is often visually and semantically indistinguishable from the original training data. The success of DLG and its variants depends on the type of data and model architecture. Image data, for instance, is particularly vulnerable since the structured nature of images allows efficient gradient matching. High-resolution images can be reconstructed with striking accuracy, exposing personally identifiable information in medical imaging or biometric applications. Text data in NLP models is also susceptible, where token embeddings and gradient updates can be exploited to infer words and sentence structures. Other gradient inversion techniques extend the capabilities of DLG by improving stability and accuracy. IG [12] introduces a similarity measure based on cosine distance instead of Euclidean distance to improve reconstruction robustness. This modification ensures that gradients are matched based on directional similarity rather than magnitude alone, making the attack more resilient to variations in gradient scaling. In another approach, GRNN [13] enhances gradient leakage attacks by incorporating a generative model that learns an optimal mapping between gradients and original inputs. This approach enables more efficient data recovery, even when only partial gradients are available.

Gradient leakage attacks are particularly concerning in FL due to the distributed nature of the training process. Since different clients train on distinct datasets, an adversary controlling the central server can analyse incoming gradient updates to infer statistical properties of each client's private data. This type of attack is more severe when applied iteratively across multiple training rounds, as cumulative gradient analysis can reveal more information about local datasets. Attackers may also combine gradient leakage with model inversion attacks, where

reconstructed gradients are fed into another neural network trained to refine and enhance the reconstructed data. The implications of gradient leakage extend beyond individual privacy violations. In sectors such as healthcare, finance, and legal analytics, where FL is employed to enable collaborative learning without compromising sensitive records, data reconstruction poses a significant ethical and legal concern. Personal health records, financial transactions, and confidential legal documents could be inferred from gradient updates, violating regulatory standards. The continuous evolution of gradient leakage attacks highlights the need for ongoing research into privacy-preserving FL mechanisms. Future advancements may include adversarial training strategies where models are explicitly trained to be resilient against reconstruction attacks. Combining multiple defence mechanisms, such as DP with gradient masking, may also provide stronger protection without excessive computational costs. Although FL represents a significant advancement in privacy-aware ML, the threat posed by gradient leakage attacks underscores the necessity of implementing robust security protocols. By addressing these vulnerabilities, FL can be effectively deployed in sensitive domains without compromising data confidentiality.

2.5 Gradient Leakage Defence Mechanisms

FL has been proposed as a privacy-preserving ML paradigm; however, gradient leakage attacks pose a significant threat by allowing adversaries to reconstruct private training data from shared model updates. To mitigate these risks, various privacy-enhancing techniques have been developed, including cryptographic methods, statistical obfuscation approaches, and representation-space defences. The effectiveness of these defence mechanisms is often evaluated through generative gradient leakage audits that test the resilience of FL systems against adversarial reconstruction attempts.

2.5.1 Traditional Methods

Secure Multi-Party Computation (MPC) is a cryptographic protocol that enables multiple clients to jointly compute a function over their private inputs without revealing them. In FL, MPC allows clients to encrypt their gradients before transmitting them to the central server, ensuring that the model aggregation process does not expose sensitive information. The aggregated

gradient G is computed without any client directly revealing their local gradient g_i:

$$G = \sum_{i=1}^{N} g_i. \tag{2.4}$$

This is achieved by leveraging additive secret sharing, where each client splits its gradient into multiple encrypted shares and distributes them among other clients. The reconstruction of G is performed in an encrypted domain, ensuring privacy. Despite its strong security guarantees, MPC introduces significant computational and communication overhead due to repeated cryptographic operations, making it less practical for large-scale FL systems with resource-constrained devices.

DP introduces controlled noise into gradients to prevent an attacker from inferring the presence or absence of specific data points in a training set. A mechanism M satisfies ε-DP if for any two datasets D and D' that differ by one element:

$$\frac{P[M(D) \in S]}{P[M(D') \in S]} \leq e^{\varepsilon} \tag{2.5}$$

for all subsets S in the output space. In FL, DP is applied by perturbing model updates with Gaussian noise:

$$\tilde{g} = g + \mathcal{N}(0, \sigma^2), \tag{2.6}$$

where $\mathcal{N}(0, \sigma^2)$ is Gaussian noise with variance σ^2 calibrated to the privacy budget ε. While DP effectively mitigates gradient leakage, it often degrades model accuracy, particularly when high noise levels are required to ensure privacy. Techniques such as adaptive noise scaling and moment accounting are employed to balance privacy and model utility.

HE is a cryptographic technique that allows mathematical operations to be performed on encrypted data without requiring decryption. This property makes HE particularly useful in FL, where gradients can be encrypted before being transmitted to the central server. The server aggregates encrypted gradients using the additive homomorphic property:

$$E(G) = E(g_1) + E(g_2) + \cdots + E(g_N). \tag{2.7}$$

After aggregation, the decrypted result provides the same outcome as performing computations on unencrypted data. Fully HE [38] offers the strongest security guarantees, but its computational overhead makes it impractical for real-time FL. To mitigate this, Leveled HE [39] and Partially HE [40] have been explored, reducing computational complexity while maintaining privacy.

2.5.2 Advanced Defence Mechanisms

Recent studies have demonstrated that traditional privacy defences can still be vulnerable under certain conditions. The paper [14] introduced an auditing framework that applies generative adversarial models to reconstruct training data from shared gradients. This auditing approach quantitatively evaluates the robustness of privacy-preserving mechanisms by simulating real-world attacks. The study found that while DP and MPC provide strong privacy guarantees in theory, they may still leak meaningful information when gradients contain residual structural dependencies. As a result, the auditing process underscores the necessity of continuous evaluation and adaptation of privacy-preserving techniques. One of the main limitations of DP is its trade-off between privacy and model accuracy. The paper [41] proposed a novel approach that perturbs gradients in a manner that preserves their contribution to model updates while obfuscating sensitive training information. This method introduces perturbations based on an optimisation framework that aligns noise vectors with gradient directions, ensuring minimal accuracy loss:

$$\tilde{g} = g + \alpha \cdot P_{\perp}(g), \tag{2.8}$$

where $P_{\perp}(g)$ represents the projection of g onto a randomly generated noise subspace, and α is a scaling factor. By ensuring that the noise remains orthogonal to the primary gradient direction, this method achieves privacy protection without sacrificing training performance. Soteria [42] introduced a novel perspective on protecting FL models by securing representation spaces rather than solely relying on obfuscating gradients. The core idea behind Soteria is to modify the intermediate feature representations within the neural network, ensuring that information leakage is minimised at a deeper level. By constraining the representational capacity of neural networks through adversarial regularisation, Soteria prevents adversaries from reconstructing meaningful input features while preserving model performance. The defence mechanism works by optimising a loss function that enforces representation obfuscation:

$$L_{\text{soteria}} = L_{\text{task}} + \lambda L_{\text{privacy}}, \tag{2.9}$$

where L_{task} represents the standard training loss, L_{privacy} measures the similarity between intermediate representations of different inputs, and λ is a weighting factor that balances utility and privacy. By operating at the representation level, Soteria offers a complementary approach to existing gradient-based defences. This method effectively prevents privacy leakage by ensuring that even if an adversary successfully recovers model gradients, the reconstructed representations remain uninformative.

Gradient leakage attacks pose a significant challenge in FL, necessitating robust and efficient privacy-preserving mechanisms. MPC and HE provide strong cryptographic protections but suffer from high computational costs. DP offers statistical guarantees but introduces accuracy degradation. Advanced defence mechanisms, such as accuracy-lossless perturbation and representation-space protection, provide new ways to mitigate privacy risks while maintaining model utility. The emergence of generative gradient leakage auditing highlights the need for continuous evaluation and refinement of privacy defences in FL. Future research directions will focus on optimising the trade-offs between privacy, efficiency, and model performance, ensuring that FL can be securely deployed in real-world applications.

2.6 Conclusion

This chapter has reviewed the foundations of FL, its roots in ML and DL, and the main challenges that currently limit its widespread adoption. While FL enables collaborative training without sharing raw data, several research gaps remain unresolved.

First, the effectiveness of model aggregation under non-IID data is still a major barrier. Existing methods such as FedAvg and its adaptive variants mitigate client heterogeneity only at the model level, which often results in slow convergence and biased updates. This gap motivates Chapter 3, which introduces element-wise adaptive aggregation to achieve finer-grained weighting across parameters.

Second, although decentralisation reduces direct exposure of private data, gradient leakage attacks demonstrate that sensitive information can still be reconstructed from shared updates. Current defences, including noise injection or encryption, trade off either accuracy or efficiency. This motivates Chapter 4, which proposes a lightweight gradients stand-in approach that conceals private information while preserving training utility.

Finally, most applications of FL focus on generic benchmarks and overlook sector-specific requirements. In finance, for example, existing FL methods fail to exploit interdependencies across stocks and institutions, limiting predictive performance. This motivates Chapter 5, which presents a cross-stock trend integration framework to show how FL can both preserve confidentiality and deliver novel predictive insights in a real-world setting.

In summary, the remaining chapters of this thesis build directly on the gaps highlighted here. By advancing adaptive aggregation (Chapter 3), privacy defence (Chapter 4), and financial applications (Chapter 5), the thesis aims to move FL closer to secure, efficient, and practical deployment in real-world environments.

Chapter 3

Element-Wise Weights Aggregation for Federated Learning

In this chapter, we focus on topic 1 in FL, which is model aggregation in heterogeneous settings. A central challenge in FL is the effective aggregation of local model weights from disparate and potentially unbalanced participating clients. Existing methods often treat each client indiscriminately, applying a single proportion to the entire local model. However, it is empirically advantageous for each weight to be assigned a specific proportion. This chapter introduces an innovative EWWA-FL aimed at optimising learning performance and accelerating convergence speed. Unlike traditional FL approaches, EWWA-FL aggregates local weights to the global model at the level of individual elements, thereby allowing each participating client to make element-wise contributions to the learning process. By taking into account the unique dataset characteristics of each client, EWWA-FL enhances the robustness of the global model to different datasets while also achieving rapid convergence. The method is flexible enough to employ various weighting strategies. Through comprehensive experiments, we demonstrate the advanced capabilities of EWWA-FL, showing significant improvements in both accuracy and convergence speed across a range of backbones and benchmarks.

3.1 Introduction

As the digital world continues to expand at an unprecedented rate, the world is inundated with a massive amount of data, distributed across various devices, sensors, and platforms. With the growing adoption of ML algorithms, the demand for efficient, secure, and decentralised learning

processes has become increasingly critical. FL [1–4] has emerged as a promising solution to address these challenges. It enables the deployment of learning algorithms on decentralised data sources while safeguarding data privacy. FL focuses on training ML models across a multitude of dispersed devices or clients, each holding their own local datasets, eliminating the need for data exchange. This approach effectively addresses privacy and security concerns, as it obviates the need to transfer potentially sensitive data to a centralised location. However, a key challenge in FL lies in the aggregation of model weights. The process of combining model weights from multiple, disparate clients is inherently complex due to the heterogeneous nature of their data distributions [43–45]. In an FL network, each client utilises its local data to train an independent model. Consequently, these local models may capture different data patterns, posing a challenge to the creation of a well-generalised global model. Various strategies, such as FedAvg, have been developed to mitigate these issues. Nonetheless, devising an efficient and robust aggregation mechanism remains a significant challenge in the field of FL.

There has been a surge in recent research focused on adaptive weight aggregation. Reddi *et al.* [46] proposed a method called FedOpt. They provide a theoretical analysis of the model's convergence on heterogeneous data for non-convex optimisation problems, as well as the relationship between dataset heterogeneity and communication efficiency. Three specific optimisation methods, termed FedAdam, FedAdagrad, and FedYogi, are employed by the authors. These methods modify the global update rule of FedAvg from one-step SGD to one-step adaptive gradient optimisation. Conversely, the work presented in [47] aims to address the challenge of high communication cost in FL. The authors propose a novel communication-efficient adaptive FL method called FedCAMS and also provide a theoretical analysis to guarantee model convergence. They first improve upon FedAdam by incorporating AMSGrad [48] with max stabilisation. Both FedOpt and FedCAMS aggregate local updates and obtain an averaged gradient, upon which global model aggregation is conducted. In other words, both methods treat each client equally when generating the global updates. However, the underlying philosophy of adaptive optimisation generally favours treating each individual weight independently. While FedOpt and FedCAMS both use adaptive techniques for global model aggregation, their averaging processes do not account for the varied contributions of local models trained on different datasets. This is noteworthy because different datasets result in different levels of convergence [7, 49].

In the work of Federated Boosting (FedBoosting) [49], the authors proposed an adaptive gradient aggregation method based on the boosting algorithm. They discovered that the gen-

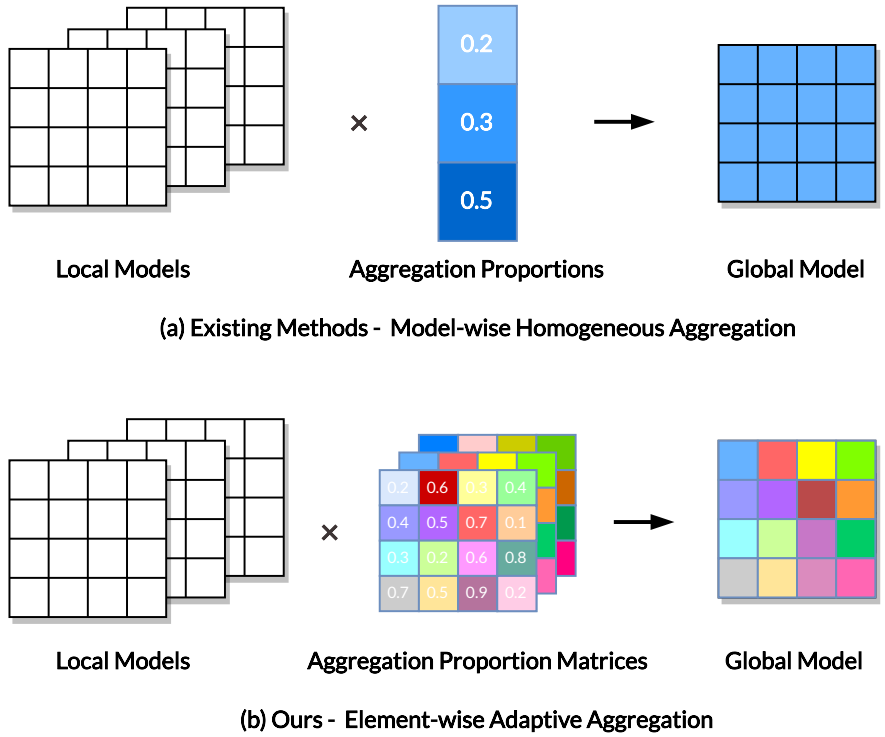


Figure 3.1: Illustration for our proposed EWWA-FL. (a) shows traditional aggregation, where each client contributes a single scalar weight to the global model, treating all parameters equally. (b) presents EWWA-FL's aggregation proportion matrix, which assigns distinct weights to individual parameters, enabling element-wise integration of local models. This fine-grained weighting better captures client-specific data characteristics, improving robustness to heterogeneity and enhancing convergence and generalization on non-IID data.

eralisation ability of the global model on non-IID data is unsatisfactory due to the presence of weight divergence, particularly when employing the FedAvg strategy. Consequently, each client participating in the training receives a unique aggregation percentage. Similarly, Wu *et al.* [7] found that in FL, the path that minimises the local objective does not necessarily align with the path of global minimisation. This implies that each client's contribution to global aggregation will differ. Based on this observation, Wu *et al.* proposed Federated Adaptive Weighting (FedAdp), a method that measures the contributions of participating clients based on the correlation between local and global gradients. All of the above studies have one thing in common that they treat all model parameters equally when aggregating the global model. Specifically, both FedOpt and FedCAMS perform a simple averaging of local model weights prior to subsequent computations. Although FedBoosting and FedAdp assign different proportions to each local model, they still allocate the same proportion to each parameter within these

models. This approach may not be the most intuitive or efficient way to handle local models.

Key to the FL process is the merging of model weights from different clients, which is inherently intricate and poses several challenges. The main reason for this complexity is the heterogeneity of the data distribution of the participating entities, cause each client's local dataset has different statistical properties. For example, one client's dataset may contain one or more specific classes, while another client does not. This heterogeneity can result in non-IID data, which poses a significant challenge in aggregating local updates in a way that is representative and conducive to global model performance and generalisation. To mitigate the effects of different data distributions and to ensure robust model aggregation, many sophisticated algorithms and techniques are proposed [1,7,46,49]. Building a balanced and harmonious model requires not only rigorous mathematical or algorithmic knowledge, but also a comprehensive understanding of the differences and nuances inherent in the data landscape of different clients. Based on the findings from Wu *et al.* [7], we would like to expand the idea to a more grained level that the elements in each local model have their personalised path to minimise the local objective. It is conceivable that each client should have a different weight, and likewise, each parameter within the local model should also have a unique weight. In the context of convex optimisation for learning, the goal is to update the model's weights to achieve convergence. A model comprises various parameters, the values of which fluctuate depending on the feature space of the local dataset. Within the framework of FL, individual local models, trained on distinct datasets, display unique patterns and directions of convergence. Hence, the same parameter across these local models may have vastly different values and may not align closely with each other. Additionally, each parameter may follow a unique trend and orientation toward convergence. As a result, using a uniform proportion to aggregate all parameters into a global model may not be the most suitable approach. Based on this understanding, we introduce EWWA-FL, which assigns a different aggregation proportion to each parameter in the local model. As shown in Figure 3.1, we proposed an aggregation proportion matrix generated by Adam algorithm. Each entry in this matrix assigns a distinct weight to each parameter in the local models, enabling the global model to integrate information at an element-wise level. This finer granularity allows the aggregation process to adapt to client-specific data characteristics, effectively mitigating the adverse effects of client heterogeneity. By capturing parameter-level variations rather than collapsing them into a single proportion, EWWA-FL enhances both convergence stability and generalization across non-IID data. Experimental results show that our method outperforms FedAvg, FedCAMS, and FedOpt across various neural networks,

benchmark datasets, and experimental settings. The contributions of this chapter are fourfold:

- We introduce a new perspective on element-wise weight combination for FL. This approach assigns a specific proportion to each parameter in the local model, aiming to improve aggregation. Experimental results confirm the novelty of our proposed EWWA-FL.
- A comprehensive evaluation is conducted. We test the model's generalisation ability using various neural networks on different benchmark datasets, employing both IID and non-IID strategies.
- The adaptive element-wise aggregation paradigm demonstrates faster convergence compared to other recent works.
- We disclose the implementation details of the proposed algorithm to ensure its reproducibility.

The chapter is organised as follows. Section 3.2 reviews previous studies related to adaptive weight aggregation in FL. Preliminaries on vanilla FL and the Adam optimisation algorithm are then discussed. Our proposed approach is elaborated upon in Section 3.3. Section 3.4 provides insights into the experiments, offers in-depth discussions, and suggests potential mitigation methods. Finally, concluding remarks are presented in Section 3.5.

3.2 Related Work

A fundamental challenge in FL is the efficient aggregation of model weights from diverse and potentially non-IID data sources to produce a globally consistent model. Adaptive weight aggregation addresses this challenge by assigning different proportions to local model weights based on their quality or relevance, as opposed to treating them equally. This approach recognises the inherent heterogeneity present in real-world FL environments. It optimises the performance of the global model by leveraging the more informative weights from local models and potentially mitigates the negative impact of less reliable participants.

In the work [46], the authors provide a comprehensive discussion on adaptive weight aggregation for FL and propose a flexible framework called FedOpt. This framework is capable of incorporating multiple optimisation algorithms. The authors specialise FedOpt into FedAdam, FedAdagrad, and FedYogi by employing three example optimisation algorithms: Adam [50],

Adagrad [51], and YOGI [52]. This approach closely parallels the FedAvg process, diverging only in the final stage of weight aggregation. After obtaining the averaged local gradients, denoted as $\hat{g}$, the first-order momentum matrices m are computed for FedAdam, FedAdagrad, and FedYogi, as detailed in (3.1). However, the computation of the second-order variance matrices v varies depending on the algorithm. Specifically, FedAdam employs (3.2), while FedAdagrad and FedYogi utilise (3.3) and (3.4), respectively, to derive their second-order matrices.

$$m_r = \beta_1 m_{r-1} + (1-\beta_1)\hat{g}_r \tag{3.1}$$

$$v_r = \beta_2 v_{r-1} + (1-\beta_2)\hat{g}_r^2 \tag{3.2}$$

$$m_r = m_{r-1} + \hat{g}_r^2 \tag{3.3}$$

$$v_r = \beta_2 v_{r-1} + (1-\beta_2)\cdot \hat{g}_r^2 \cdot sign(v_{r-1} - \hat{g}_r^2) \tag{3.4}$$

where, r is the training round, β_1 and β_2 are two momentum parameters, $sign()$ is the symbolic functions. In the end, all those three methods employ Eq.3.5 for weights aggregation.

$$\omega_r = \omega_{r-1} + \eta_r \frac{m_r}{\sqrt{v_r}+\varepsilon} \tag{3.5}$$

where, η_r is the adaptive learning rate, calculated by:

$$\eta_r = \eta_0 \frac{\sqrt{1-\beta_2^r}}{1-\beta_1^r} \tag{3.6}$$

where η_0 denotes the initial learning rate, while β_1^r and β_2^r represent the r-th powers of the parameters β_1 and β_2, respectively. The authors provide a theoretical analysis to demonstrate the superiority of the proposed FedOpt in comparison to other methods. The primary distinction between FedOpt and our proposed method, EWWA-FL, lies in the location of the optimisation algorithm. Specifically, FedOpt employs the optimisation algorithm after averaging the local models, whereas EWWA-FL performs the optimisation after each local training. As a result, FedOpt treats each client equally and assigns the same aggregation proportion to each local model through averaging. In contrast, our method treats each parameter in every local model differently. Building upon FedOpt, Wang *et al.* [47] introduced FedCAMS with the objective of reducing communication costs. The optimisation algorithm in FedCAMS occupies the same position as in FedOpt, thereby ensuring that all local weights are aggregated equally.

Unlike FedOpt and FedCAMS, FedBoosting [49] and FedAdp [7] assign different proportions to each local model to perform adaptive weight aggregation. FedBoosting computes the

aggregation proportion based on the results of local training T_r^i and cross-validation $V_r^{i,j}$. The authors first sum all the validation results for a local model across all other local model validation datasets. Then, they calculate the weight of this sum of validation results. Finally, a *Softmax* function is applied to derive the final proportion for each local model. Equations.3.7, 3.8, and 3.9 provide the local weight aggregation proportion p_r^i for the i-th local model in training round r:

$$p_r^{(i)} = softmax(softmax(T_r^{(i)}) \cdot \sum_{j \neq i}^{N} V_r^{(i,j)}) \tag{3.7}$$

$$softmax(T_r^{(i)}) = \frac{exp(T_r^{(i)})}{\sum_{j=1}^{N} exp(T_r^{(j)})} \tag{3.8}$$

$$V_r^{(i,j)} = \begin{pmatrix} V_r^{(1,1)} & V_r^{(1,2)} & \cdots & V_r^{(1,j)} \\ V_r^{(2,1)} & V_r^{(2,2)} & \cdots & V_r^{(2,j)} \\ \vdots & \vdots & \ddots & \vdots \\ V_r^{(i,1)} & V_r^{(i,2)} & \cdots & V_r^{(i,j)} \end{pmatrix} \tag{3.9}$$

On the other hand, FedAdp focuses on the angle of convergence between the updated local weight and the global weight. In particular, they quantify the contribution of each client in each round of global observations according to the angle θ^i:

$$\theta^{(i)} = arccos(\frac{< G, g^{(i)} >}{||G|| \cdot ||g^{(i)}||}) \tag{3.10}$$

where G is the global gradient, $< \cdot >$ is the inner product operation and $|| \cdot ||$ denotes the L2 normalisation. To suppress instability caused by instantaneous angular randomness, the angle θ_r^i is then averaged over previous training rounds r:

$$\hat{\theta}_r^{(i)} = \begin{cases} \theta_r^{(i)} & \text{if} \quad r = 1 \\ \frac{r-1}{r} \hat{\theta}_{r-1}^{(i)} + \frac{1}{r} \theta_r^{(i)} & \text{if} \quad r > 1 \end{cases}$$

The authors then designed a non-linear mapping function that quantifies each client's contribution based on angular information. Inspired by the *Sigmoid* function, they use a variant of *Gompertz* function [53]:

$$\mathscr{F}(\hat{\theta}) = \alpha(1 - \frac{1}{exp(exp(\alpha(1 - \hat{\theta})))}) \tag{3.11}$$

where α is a hyper-parameter. The final proportions for each local model are calculated by giving each client's contribution value into the *Softmax* function.

In comparison to FedOpt and FedCAMS, although FedBoosting and FedAdp provide different aggregation proportions for local clients, their aggregation proportions are still at the model level. In contrast, our proposed EWWA-FL makes progress in this regard by providing a more fine-grained, element-wise aggregation level. This feature enhances the adaptability and convergence of the global model, especially considering that the local models come from different datasets.

3.3 Methodology

In this section, we first present a preliminary discussion on FedAvg, as it is the most commonly used method in FL applications. Subsequently, we briefly explain the Adam optimisation algorithm, highlighting that Adam provides an adaptive learning rate for each parameter, in contrast to the SGD algorithm. Finally, we introduce the EWWA-FL algorithm, which enables element-wise global weight aggregation in FL.

3.3.1 FedAvg

It is the most basic method behind all the recent proposed FL methods. Assuming we have many clients $\mathscr{C}$ and their local datasets $\mathscr{D}$. The task is formulated as $\mathscr{F}$ with weights ω. So the local gradient g is:

$$g_i = \frac{1}{||d_i||}\nabla_\omega \sum_j \mathscr{L}(\mathscr{F}(x^{(j)};\omega), y^{(j)}); \forall i \in \mathscr{C} \tag{3.12}$$

Where $||d||$ denotes the number of samples, x and y are the samples and their relevant labels in the i-th local dataset d_i. The server gathers all the local gradients and conduct the averaging process to generate the next round of global model ω_r. We assume that $||d_i|| = ||d_k||; \forall d_i, d_k \in \mathscr{D}$.

$$\omega_r = \omega_{r-1} - \frac{1}{\mathscr{C}} \sum_{i=1}^{\mathscr{C}} g_i \tag{3.13}$$

The FedAvg algorithm [1] aims to create a unified model by averaging gradients from various local clients. While this method is effective for centralizing distributed learning, it is not without shortcomings. Specifically, inherent differences in data distributions among clients lead to diverse convergence directions for local model weights. This diversity arises from the incoherent feature spaces of the data, posing challenges for FedAvg. When local datasets differ significantly in their data distributions, this can induce a considerable bias in

local model weights. Consequently, the simplistic averaging mechanism employed by FedAvg may not yield optimal results, particularly in the presence of significant data biases or extreme outliers. recognising these limitations, we have started exploring more nuanced aggregation strategies, such as weighted averages or other adaptive mechanisms. These approaches gauge the contribution of each local model based on factors like data distribution or its relevance to the overall learning objective [54–56]. Employing these refined techniques produces a global model that is both resilient and accurate, skilfully navigating the complexities of differing data distributions and overcoming some limitations inherent to traditional FedAvg methods.

3.3.2 Element-Wise Aggregation for FL

Prior to detail our proposed method, we would like to briefly introduce the Adam algorithm [50] firstly. It is a SGD optimisation method based on the momentum idea. Before each iteration, the first-order and second-order moments of the gradient are computed and the sliding average is computed to update the current parameters. This idea combines the ability of the Adagrad [51] algorithm to handle sparse data with the properties of the RMSProp [57] algorithm to deal with non-smooth data. Finally, it achieves very good test performance on both traditional convex optimisation problems and DL optimisation problems. More details of Adam is shown in Algorithm 1.

Algorithm 1: Adam

Require: initial learning rate α
Require: exponential decay rates $\beta_1, \beta_2 \in [0,1]$
Require: maximum iteration number I
initial weights ω_0
initial 1^{st}-order moment vector $m_0 \leftarrow 0$
initial 2^{nd}-order moment vector $v_0 \leftarrow 0$
for each iteration $i = 1,2,3,...,I$ **do**
  $g_i \leftarrow \nabla_\omega \mathscr{L}(\mathscr{F}(\omega_{i-1}))$ ▶ get gradients at iteration i
  $m_i \leftarrow \beta_1 \cdot m_{i-1} + (1-\beta_1) \cdot g_i$ ▶ update biased 1^{st}-order moment estimate
  $v_i \leftarrow \beta_2 \cdot v_{i-1} + (1-\beta_2) \cdot (g_i \odot g_i)$ ▶ update biased 2^{nd}-order moment estimate
  $\hat{m}_i \leftarrow \frac{m_i}{1-\beta_1^i}$ ▶ compute biased-corrected 1^{st}-order moment estimate
  $\hat{v}_i \leftarrow \frac{v_i}{1-\beta_2^i}$ ▶ compute biased-corrected 2^{nd}-order moment estimate
  $\omega_i \leftarrow \omega_{i-1} - \frac{\alpha}{\sqrt{\hat{v}_i}+\varepsilon} \cdot \hat{m}_i$ ▶ update weights
end for
return ω_I

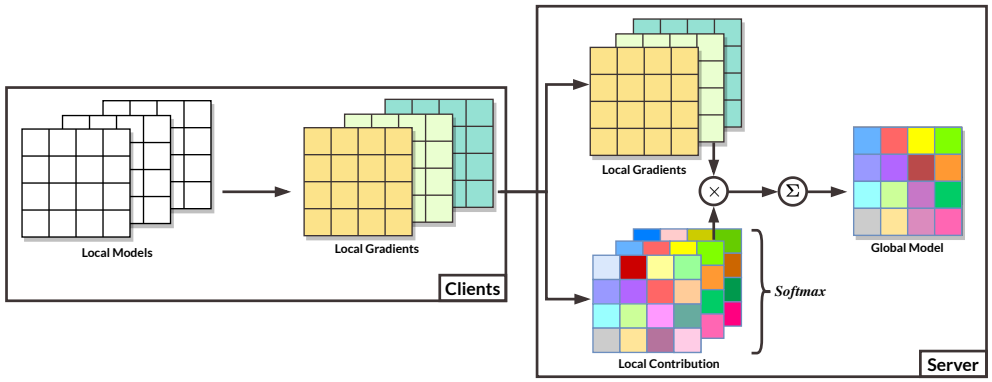


Figure 3.2: Diagram of EWWA-FL. The left part is the client that performs the local training. The right side is the server calculating the local contributions and aggregating a new global model based on the local contributions and gradients. All calculations on the server end are done on an element-by-element basis.

All of the above works [1, 7, 46, 47, 49] either average the local models or assign dynamic proportions to the entire local model for global model aggregation. The learning process for DL model can be viewed as a convex optimisation problem where the weights in the model are trained to reach a minimum point. The basic components of the model are a number of parameters whose values can vary greatly with the different feature spaces of different local datasets. In the FL scenario, local models trained on different local datasets obtain different degrees and directions of convergence. In these local models, the values of the same parameters may be completely different or even not close to each other, leading to convergence heterogeneity in the local models. Therefore, giving one proportion for all the parameters in the local model is not the best way to aggregate the global model. From another perspective, in contrast to SGD, the adaptive learning methods calculate the learning rate from the point of view of the elements. That is to say, in common model training scenarios, Adam provides an element-wise adaptive learning rate. We would like to follow Adam's idea and introduce element-level adaptive aggregation to FL. As shown in Figure 3.2, the left part is the client that performs the local training. The right side is the server, which calculates the local contributions and aggregates a new global model based on the local contributions and gradients. All computations on the server end are performed on an element-by-element basis. Specifically, Algorithm 2 is the pseudo-code of our proposed EWWA-FL. In the first round of global training, the global weights ω_0, first-order moment vectors m_0 and second-order moment vectors v_0 are initialised. Then, the server end assigns weights to each local model for local training. Once the local training is complete, The server collects the gradients of all local models to update the biased first-order

moments and second-order moments m_r and v_r and compute the unbiased estimates $\hat{m}_r$ and $\hat{v}_r$. Then obtain the contribution b_r of the local model to the new global model. Finally, the aggregation proportion for each local model is computed using *Softmax*. All the computations related to the first-order moment, second-order moment, local contribution and final aggregation proportion are element-wise, which means each parameter in each local model will receive a specific aggregation proportion in each round, rather than one proportion for all parameters in the local model.

Algorithm 2: EWWA-FL

Require: initial local learning rate α
Require: exponential decay rates $\beta_1, \beta_2 \in [0,1]$
Require: global training round R
initial weights ω_0
initial 1^{st}-order moment vector $m_0 \leftarrow 0$
initial 2^{nd}-order moment vector $v_0 \leftarrow 0$
for each round $r = 1, 2, ..., R$ **do**
  for each client $c \in C$ **do**
    $g_r^{(c)} \leftarrow \nabla_\omega \mathscr{L}(\mathscr{F}(\omega_{r-1}))$
    $m_r^{(c)} \leftarrow \beta_1 \cdot m_{r-1} + (1-\beta_1) \cdot g_r^{(c)}$
    $v_r^{(c)} \leftarrow \beta_2 \cdot v_{r-1} + (1-\beta_2) \cdot (g_r^{(c)} \odot g_r^{(c)})$
    $\hat{m}_r^{(c)} \leftarrow \frac{m_r^{(c)}}{1-\beta_1^r}$
    $\hat{v}_r^{(c)} \leftarrow \frac{v_r^{(c)}}{1-\beta_2^r}$
    $b_r^{(c)} \leftarrow \frac{\alpha}{\sqrt{\hat{v_r^{(c)}}}+\varepsilon} \cdot \hat{m}_r^{(c)}$
  end for
  $p_r^{(c)} \leftarrow \frac{exp(b_r^{(c)})}{\sum_{i=c}^{C} exp(b_r^{(i)})}; \forall c \in C$
  $G_r \leftarrow \sum_{i=c}^{C} p_r^{(i)} \cdot g_r^{(i)}$
end for

3.4 Experiments

In this section, we first describe the settings of all experiments. Then, we introduce the backbone neural networks and the datasets for benchmark evaluation. After that, we present multiple sets of experiments to access the performance of our proposed EWWA-FL against other state-of-the-art methods.

3.4.1 Settings, Backbones and Datasets

We utilised the PyTorch framework [58] to implement all neural network models. For anyone interested in replicating our results, the source code is open to the public and can be accessed here[1]. For our proposed EWWA-FL method, the global aggregation learning rate for Adam, Adagrad, and Yogi was set at 1.0 based on the FedOpt reference [46]. We chose the two momentum parameters of 0.9 and 0.999. Local training was conducted using the SGD optimisation algorithm, accompanied by a consistent learning rate of 0.01 and a momentum of 0.9. The local batch size was set at 64. The ILSVRC2012 dataset was given a training round number of 100, while the other datasets were subjected to 500 rounds. The code we used to reproduce the FedOpt and FedCAMS results was taken from FedOpt's official GitHub repository[2].

The neural networks of choice include *LeNet* [59], *ResNet-18*, *ResNet-20*, *ResNet-32*, and *ResNet-34* [21]. *LeNet* is comprised of three convolutional layers with each succeeded by a *Sigmoid* activation function. The output layer is a fully-connected layer. Its input dimension stands at $32*32*3$. On the other hand, *ResNet-18* and *ResNet-34* are derivatives from PyTorch's official offerings, having an input dimension of $224*224*3$. As for *ResNet-20* and *ResNet-32*[3], the developers are tailored specifically for the CIFAR-10 and CIFAR-100 datasets [60]. The input size for them is $32*32*3$.

Specifically, we employed the MNIST dataset [61] for *LeNet*. The CIFAR-10 and CIFAR-100 datasets underwent experimentation using *ResNet-18*, *ResNet-20*, *ResNet-32*, and *ResNet-34*. It is worth noting that for *ResNet-18* and *ResNet-34*, the sample dimensions were upsampling to $224*224*3$. The ILSVRC2012 dataset [62] was exclusively tested using *ResNet-18* and *ResNet-34*. The datasets were partitioned in a $9:1$ ratio for training and testing. Subsequently, the training data is distributed to three local clients, following either an IID or non-IID distribution. The test samples are retained on the server end to assess the performance of the current round of the global model.

3.4.2 Accuracy on IID data

Table 3.1 presents a comprehensive comparison of top-1 classification accuracies for various FL methods, employing different backbone neural networks, and tested on multiple benchmark datasets where clients are assumed to have an IID distribution. The table employs colour-coding

[1] https://github.com/Rand2AI/EWWA-FL
[2] https://github.com/yujiaw98/FedCAMS
[3] https://github.com/akamaster/pytorch_resnet_cifar10

Table 3.1: Top-1 classification accuracy (%) across different methods, backbone neural networks and benchmark datasets with IID distribution on local clients. "C-10", "C-100" and "IC-12" stand for CIFAR-10, CIFAR-100 and ILSVRC2012, respectively. FedAMS is derived from FedCAMS and has no efficient communication settings. The red ones are the highest accuracy and the blue ones are the next highest. Crossed symbols indicate experimental failure, *i.e.* no global model converged in five trials.

Model	*LeNet* (32*32)	*ResNet-20* (32*32)		*ResNet-32* (32*32)		*ResNet-18* (224*224)			*ResNet-34* (224*224)		
Dataset	MNIST	C-10	C-100	C-10	C-100	C-10	C-100	IC-12	C-10	C-100	IC-12
FedAvg	98.14	91.20	58.58	91.37	61.91	89.50	68.59	65.50	89.27	68.30	67.91
FedAMS	98.77	86.57	54.58	87.21	54.23	91.59	66.07	×	91.91	65.41	×
FedCAMS	×	76.77	41.65	84.7	41.77	91.35	66.62	×	91.44	66.41	×
FedOpt Adam	×	73.59	22.31	74.84	×	78.07	×	×	81.71	×	×
FedOpt Adagrad	×	64.63	×	67.36	×	×	×	×	×	×	×
FedOpt YOGI	×	65.93	×	71.90	×	73.88	×	×	71.8	×	×
Ours Adam	98.00	89.73	64.14	90.17	65.63	90.77	70.98	65.64	91.23	70.15	68.23
Ours Adagrad	97.99	89.74	64.16	90.17	65.84	91.17	70.34	65.64	91.13	69.77	68.33
Ours YOGI	98.00	89.43	64.10	90.10	65.38	90.88	70.78	65.54	91.13	70.34	68.27

to highlight significant results; values highlighted in red indicate the highest performance for each dataset, while those in blue signify the second highest performance.

Performance on Large-Class Datasets: One of the most noteworthy observations is that our proposed EWWA-FL algorithm exhibits exceptional performance on datasets that have a large number of classes. Specifically, it outshines the competition on the CIFAR-100 and ILSVRC2012 datasets, both of which have a large number of classes, 100 and 1000 respectively. For instance, when employing the *ResNet-20* architecture on the CIFAR-100 dataset, our EWWA-FL model, when optimised using the Adagrad optimiser, achieved an accuracy of 64.16%. This is considerably better than the next best performing method, FedAvg, which achieved an accuracy of 58.58%. The 9.53% margin reflects how element-wise weighting is able to suppress noisy parameter updates from clients with skewed class distributions, thereby reducing variance in class-specific parameters and producing a more stable global model. Similarly, when using *ResNet-32* as the backbone architecture on CIFAR-100, EWWA-FL notched an accuracy of 65.84%, surpassing FedAvg's 61.91% by a margin of 6.35%. Although the performance gain is smaller than that observed with ResNet-20, the results still highlight that per-parameter adaptation consistently outperforms uniform averaging, especially in non-IID settings where client updates diverge. On the larger-scale ILSVRC2012 dataset, EWWA-FL continues to show an advantage. With the *ResNet-18* architecture it reached an accuracy

of 65.64%, and with *ResNet-34* it achieved 68.33%. Both are the highest among the tested methods for their respective architectures. Here the gains are more modest, which can be attributed to the broader class coverage in ILSVRC2012 that reduces the severity of update divergence. Nevertheless, the consistent improvement demonstrates that EWWA-FL's finer-grained aggregation is beneficial across scales. In contrast, methods such as FedOpt and FedAvg rely on model-wise or uniform weighting, which amplifies the effect of divergent updates when data are non-IID. This explains their unstable convergence patterns compared to EWWA-FL, which adaptively balances contributions at the parameter level. In the end, in large-class and large-scale datasets such as ILSVRC2012, client updates tend to diverge significantly due to highly imbalanced and complex distributions. Model-wise methods like FedAvg cannot resolve these discrepancies, whereas EWWA-FL's parameter-level adaptation selectively emphasises stable and representative parameters across clients, leading to a more robust global model.

Competitive Results on Smaller-Class Datasets: Although EWWA-FL does not achieve the absolute highest accuracy on datasets like MNIST and CIFAR-10, the results show that the algorithm is highly competitive. For the MNIST dataset, when using the *LeNet* architecture, the highest accuracy was achieved by FedAMS with 98.77%. However, EWWA-FL followed closely with an accuracy of 98.00%, a difference of only 0.79%. This small gap can be explained by the relatively homogeneous and simple nature of MNIST: when client updates are already well aligned due to low data complexity, the benefit of element-wise weighting is less pronounced. On the CIFAR-10 dataset, the performance gaps are also narrow. For instance, the differences in accuracy rates when comparing EWWA-FL to the best-performing methods are 1.63%, 1.33%, 0.46%, and 0.75% for architectures *ResNet-20*, *ResNet-32*, *ResNet-18*, and *ResNet-34*, respectively. These modest gaps highlight that in lower-class, balanced datasets such as CIFAR-10, most aggregation methods perform relatively well since update divergence is limited. Even so, EWWA-FL consistently stays within a very small margin of the best results, which demonstrates that its per-parameter weighting strategy does not introduce instability and remains effective across architectures. Overall, the results on MNIST and CIFAR-10 confirm that EWWA-FL remains robust and competitive in simpler or more balanced settings, while its advantages become more substantial on challenging datasets with high class diversity or stronger non-IID effects (as seen on CIFAR-100 and ILSVRC2012).

In conclusion, our proposed EWWA-FL method exhibits convincing performance, especially in challenging scenarios involving large classes of datasets. Although it is not necessarily the absolute best in all cases, it maintains a competitive edge in various benchmarks.

Compared with FedOpt, that is also capable of employing various global aggregation optimisation algorithms, our proposed EWWA-FL significantly outperforms in terms of stability and convergence. In our experiments, the average standard variance for EWWA-FL was remarkably low, at only 0.1094%. Additionally, the minimum and maximum variances were confined to a tight range, specifically between 0.0047% and 0.2673%, respectively. This indicates that EWWA-FL produces highly consistent performance across scenarios. The stability of this algorithm stems from its element-wise weighting. By adjusting contributions at the parameter level, it effectively downweights noisy or divergent updates from clients with skewed data. This prevents these updates from disproportionately influencing the global model. In contrast, FedOpt exhibited much higher variability. Its average standard variance was 3.5200%, more than thirty times higher than that of EWWA-FL, with minimum and maximum variances of 2.0950% and 4.9550%, respectively. This instability arises because FedOpt applies uniform or model-wise adjustments that amplify noisy updates under non-IID conditions. Without fine-grained control, local divergences accumulate and cause fluctuations in global performance. These findings demonstrate that EWWA-FL's parameter-level adaptation not only improves accuracy but also ensures smoother and more reliable convergence, particularly in heterogeneous federated environments. It is worth noting that FedOpt encountered significant issues during our testing phase. Despite conducting at least five separate attempts, none of the FedOpt trials converged as expected. This suggests that FedOpt may have fundamental limitations when it comes to achieving reliable convergence. We trialled code from the official FedCAMS GitHub repository as well as our own deployed code. Similarly, FedAMS and FedCAMS demonstrated a lack of robustness in our experiments. Specifically, FedAMS failed to converge on the ILSVRC2012 dataset, while FedCAMS failed on both the MNIST and ILSVRC2012 datasets. These failures further underscore the superiority of EWWA-FL in achieving stable and consistent results across various benchmark datasets.

3.4.3 Accuracy on non-IID data

The circumstance on non-IID data exhibits distinct challenges compared to those on IID data. In my experiments, non-IID partitions were generated by allocating disjoint sets of classes to different clients. As the results presented in Table 3.2, our proposed EWWA-FL model consistently outperformed other methods across various benchmarks, including MNIST and CIFAR-10 datasets. For these experiments, We employed Adam as the global optimisation algorithm for both FedOpt and EWWA-FL for a fair comparison. Focusing on the MNIST

Table 3.2: Percentage (%) of top-1 classification accuracy across methods, backbone neural networks, and benchmark datasets in non-IID conditions on local clients. The values highlighted in red demonstrate the highest performance. The global aggregation optimization algorithm for FedOpt and EWWA-FL is Adam.

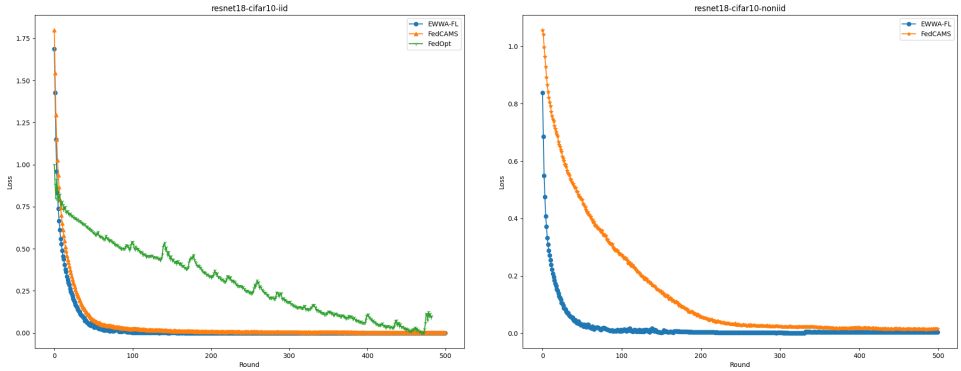
Model	*LeNet* (32*32)	*ResNet-20* (32*32)		*ResNet-32* (32*32)		*ResNet-18* (224*224)			*ResNet-34* (224*224)		
Dataset	MNIST	C-10	C-100	C-10	C-100	C-10	C-100	IC-12	C-10	C-100	IC-12
FedAvg	96.30	72.57	55.49	75.76	57.11	82.55	66.26	50.68	82.64	65.02	46.13
FedAMS	96.78	54.78	37.33	54.46	39.71	55.71	33.71	29.26	44.13	32.12	24.09
FedCAMS	×	47.76	×	46.47	26.87	40.03	25.57	×	41.27	24.27	×
FedOpt	×	61.16	×	46.75	×	×	×	×	×	×	×
Ours	96.88	76.04	56.07	78.50	57.56	83.86	66.74	51.67	84.36	65.97	52.50

dataset, EWWA-FL achieved an accuracy of 96.88%. This figure is marginally but importantly higher by 0.1033% when compared to the 96.78% reported for FedAMS. Although the margin is small, it reflects the relative simplicity and homogeneity of MNIST, where most aggregation methods perform well. Even so, the fact that EWWA-FL still edges ahead demonstrates that per-parameter weighting can fine-tune updates in a way that avoids the minor biases introduced by non-IID splits. On the CIFAR-10 dataset, when using *ResNet-20*, EWWA-FL significantly outperformed its closest competitor, FedAvg, by achieving an accuracy of 76.04%. This was a notable 4.7816% improvement over FedAvg's 72.57%. The larger gain here stems from the greater heterogeneity in CIFAR-10 compared to MNIST, that class imbalance and diverse feature distributions across clients cause uniform averaging in FedAvg to amplify divergent updates. By contrast, EWWA-FL's element-wise aggregation down-weights inconsistent parameters and stabilises convergence, yielding higher overall accuracy. The performance gains extended to more challenging datasets as well. For example, on the ILSVRC2012 dataset, EWWA-FL reached an accuracy of 52.50%, which was 13.8088% higher than the 46.13% managed by FedAvg. This substantial improvement highlights the scalability of EWWA-FL. Together, these results suggest that EWWA-FL is not only competitive on simpler datasets but also provides clear advantages in more complex and heterogeneous environments, where its fine-grained aggregation mechanism mitigates divergence more effectively than model-wise approaches. In the course of our experimental evaluation, it became evident that certain algorithms like FedAMS, FedCAMS, and FedOpt encountered significant difficulties in reaching convergence. This was consistent with their performance on IID data. Specifically, FedAMS and FedCAMS yielded unsatisfactory results in numerous tests, such as achieving only 37.33% accuracy using

ResNet-20 on the CIFAR-100 dataset and 40.03% accuracy using *ResNet-18* on the CIFAR-10 dataset. Moreover, FedAMS scored as low as 24.09% when tested using *ResNet-34* on the ILSVRC2012 dataset. Similarly, FedOpt struggled in several experiments, underscoring the limitations of current global optimisation techniques when dealing with non-IID data distributions. As elaborated in Section 3.3, the model divergence owing to the heterogeneity of local feature spaces is substantial when only a single aggregation proportion is provided for the entire local model. Given that each parameter in the local model can have its own unique direction and level of convergence, the one-size-fits-all approach falls short. In contrast, an element-wise global model aggregation strategy, as implemented in EWWA-FL, offers the adaptability and flexibility needed to facilitate better convergence across a range of neural networks and benchmark datasets.

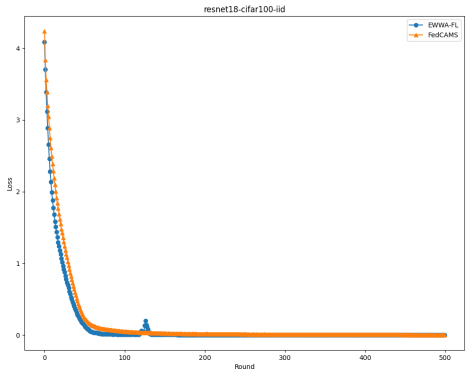
3.4.4 Convergence Speed

Convergence speed is a critical metric in evaluating the effectiveness of adaptive aggregation strategies in FL. Figure 3.3 presents the visualisation of the average training loss across all local clients, illustrating how quickly each method converges over 500 global aggregation rounds. The subfigures in the left column correspond to IID data settings across different model architectures and datasets, while those in the right column reflect experiments under non-IID conditions. As observed, our proposed EWWA-FL consistently achieves faster convergence compared to FedCAMS and FedOpt across all scenarios. In the IID setting, both EWWA-FL and FedCAMS demonstrate rapid loss reduction, particularly when using shallower models like *ResNet-18*, as shown in Figure 3.3(a) and 3.3(c). However, the advantage of EWWA-FL becomes more pronounced when using deeper models such as *ResNet-32* (Figure 3.3(e)). This indicates that parameter-wise weighting is able to exploit the additional representational capacity of large models more effectively than scalar, model-wise schemes. On large-class datasets such as CIFAR-100, this fine-grained weighting reduces variance among class-specific parameters, allowing the global model to learn more balanced representations and improving convergence speed relative to FedCAMS. By contrast, FedOpt exhibits unstable convergence and significantly higher loss throughout, indicated by the green curves. To facilitate visual comparison, the loss values from FedOpt have been normalised to the [0, 1] range. Despite this adjustment, its convergence remains poor and inconsistent, often stagnating or fluctuating instead of improving over time. This instability arises because FedOpt applies uniform updates across parameters, which amplifies the effect of noisy or divergent gradients. Under the non-IID

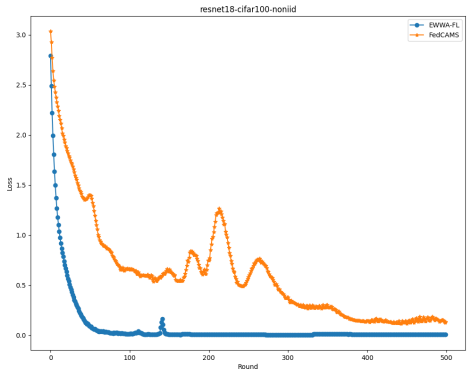


(a) ResNet-18, CIFAR-10, IID

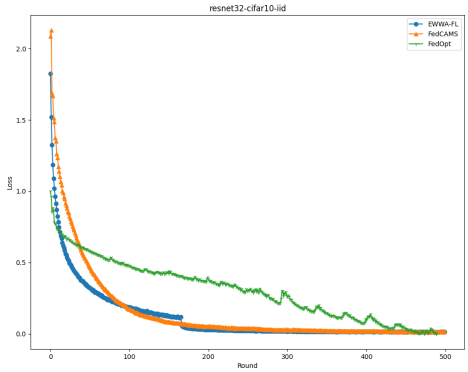
(b) ResNet-18, CIFAR-10, non-IID



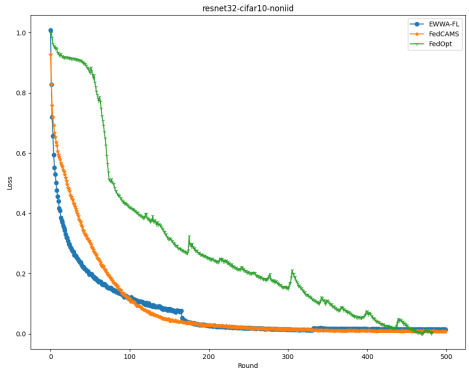
(c) ResNet-18, CIFAR-100, IID



(d) ResNet-18, CIFAR-100, non-IID



(e) ResNet-32, CIFAR-10, IID



(f) ResNet-32, CIFAR-10, non-IID

Figure 3.3: Visualization of the average training loss across all local clients is shown. The horizontal axis represents the number of global aggregation rounds, while the vertical axis indicates the average loss for the current round. The blue line represents our proposed EWWA-FL, the orange line is from FedCAMS, and the green line represents FedOpt.

setting (Figures 3.3(b), 3.3(d), and 3.3(f)), EWWA-FL maintains its advantage, converging faster and more smoothly than FedCAMS. Notably, FedCAMS experiences a considerable degradation in convergence when using *ResNet-18* on both CIFAR-10 and CIFAR-100, highlighting its sensitivity to data heterogeneity. However, when using *ResNet-32* on the CIFAR-10 dataset, FedCAMS regains some stability and achieves competitive convergence speed, which may be attributed to the increased parameter capacity of deeper models. Importantly, EWWA-FL benefits in this setting from its element-wise Softmax weighting, which systematically down-weights parameters associated with noisy or divergent updates. This leads to smoother training dynamics and avoids the oscillations seen in model-wise schemes. Interestingly, FedOpt fails to converge entirely in several non-IID experiments, as indicated by the absence of green lines in Figures 3.3(b), 3.3(c), and 3.3(d). As noted in Section 3.4.2, this failure further demonstrates the limitations of FedOpt in handling heterogeneous data distributions. Overall, these results provide strong empirical support for the efficacy of our EWWA-FL framework. Its adaptive element-wise aggregation mechanism not only accelerates convergence under ideal IID conditions but also ensures robustness in more realistic non-IID environments. By mitigating class-specific variance and suppressing noisy updates, EWWA-FL achieves faster and smoother convergence than competing methods across a variety of architectures and datasets.

3.5 Conclusion

In this chapter, we presented EWWA-FL, an adaptive element-wise global weights aggregation method for FL. Unlike traditional aggregation schemes, EWWA-FL assigns individual weights to each model parameter, enabling more fine-grained and data-aware updates. Extensive empirical evaluations across multiple datasets and neural network architectures demonstrate that EWWA-FL consistently achieves faster convergence and improved performance compared to baseline methods such as FedAvg, FedOpt, and FedCAMS. A brief theoretical analysis grounded in the principles of the Adam optimisation algorithm further supports the design rationale of our approach. The implementation of EWWA-FL is made publicly available to facilitate reproducibility and further research.

Chapter 4

Gradient Leakage Defence in Federated Learning

In the previous chapter, we addressed one of the central barriers in FL, that is how to achieve effective aggregation under heterogeneous, non-IID client data. By introducing element-wise weighting, EWWA-FL demonstrated that finer-grained aggregation can significantly improve convergence and stability compared to model-wise schemes. While this advances the efficiency of FL, improving aggregation alone does not guarantee secure deployment. A second and equally pressing challenge lies in privacy protection. Even though raw data never leaves the client devices, recent studies have shown that shared gradients can still be exploited to reconstruct sensitive training data. This vulnerability, known as gradient leakage, undermines the assumption that FL is inherently private and raises serious concerns for applications in healthcare, finance, and other sensitive domains. In this chapter, we turn to this problem and introduce **AdaDefence**, a novel defence mechanism designed to counter gradient leakage attacks while maintaining model performance. Instead of transmitting raw gradients to the server, AdaDefence generates and shares gradient *stand-in*, which act as substitutes during aggregation. These stand-in are derived from Adam's optimisation moments, enabling obfuscation of private information while preserving the signal needed for efficient global training.

The remainder of this chapter is structured as follows. We begin with an overview of gradient leakage attacks and existing defences, motivating the need for a lightweight alternative. We then present the design of AdaDefence, supported by theoretical justification for why gradients stand-in disrupt inversion attacks. Finally, we evaluate AdaDefence on benchmark datasets, showing that it can preserve accuracy while significantly reducing the success of state-of-the-art

leakage attacks. Through this, we position AdaDefence as a practical and effective step towards secure FL.

4.1 Introduction

The rapid advancements in DNNs have significantly influenced various DL applications, driving progress across multiple domains. These advancements are largely fuelled by the availability of extensive training datasets. However, the increasing volume of data generation in recent years includes a substantial amount of sensitive personal information, which raises critical privacy concerns and legal restrictions on data sharing. The introduction of the GDPR in Europe has reinforced the need for stringent data protection measures, ensuring the integrity and security of personal data while regulating its exchange. As a result, leveraging such datasets for DL model training must align with strict legal and ethical standards. This has led to a growing emphasis on developing techniques that enable the utilization of large-scale data while maintaining compliance with privacy regulations. Addressing these challenges is crucial for advancing DL applications in a secure and legally compliant manner.

For this purpose, FL [1–4] is employed as a decentralised model training framework where the central server accumulates local gradients from clients without requiring the exchange of local data. This approach ensures that data remains at its source, ostensibly preserving privacy because only gradient information is shared. Traditionally, these gradients have been assumed secure for sharing, making FL a prime application in privacy-sensitive scenarios. Nonetheless, recent studies have exposed vulnerabilities within this gradient-sharing framework. L. Zhu, Z. Liu and S. Han [10] introduced DLG, a technique for reconstructing the original training data by iteratively optimising randomly initialised input images and labels to match the shared gradients, rather than updating the model parameters directly. However, the performance of DLG is limited by factors such as the training batch size and the resolution of images, which can lead to instability in data recovery. To address these instabilities, J. Geiping, H. Bauermeister, H. Dröge and M. Moeller [12] proposed IG, which utilises a magnitude-invariant cosine similarity as the loss function to enhance the stability of training data reconstruction. This method has proven effective in recovering high-resolution images ($224 * 224$ pixels) from gradients of large training batches (up to 100 samples). Further refining the approach, iDLG by B. Zhao, K.R. Mopuri and H. Bilen [11] simplifies the label recovery process within DLG by analytically deriving the ground-truth labels from the gradients of the loss function relative to the *Softmax* layer outputs, thus improving the precision of the reconstructed data. Expanding on

generative methods, H. Ren, J. Deng and X. Xie [13] developed GRNN, which integrates two generative model branches: one, a Generative Adversarial Network (GAN)-based branch for creating synthetic training images, and another, a fully-connected layer designed to generate corresponding labels. This model facilitates the alignment of real and synthetic gradients to reveal the training data effectively. Recently, Z. Li, J. Zhang, L. Liu and J. Liu [14] introduced GGL, a novel method that also utilises a pre-trained GAN to generate fake data. This approach leverages the iDLG concept to deduce true labels and adjusts the GAN input sequence based on the gradient matching, thereby producing fake images that closely resemble the original training images. The paper [63] introduces a novel attack method named Gradient Leakage Attack Using Sampling (GLAUS) which targets Unbiased Gradient Sampling-based (UGS) secure aggregation methods in FL. These UGS, e.g. MinMax Sampling (MMS) [64], methods are designed to enhance security by using unbiased random transformations and gradient sampling to prevent direct access to the real gradients during model training, thereby obscuring private client data. These developments underscore the ongoing need to enhance the security measures in FL systems against sophisticated attacks that seek to exploit gradient information, thus compromising the privacy of sensitive data. The continuous evolution of defensive techniques is crucial to safeguarding data integrity in federated environments.

These issues prompted us to evaluate the reliability of the FL system. Concerns about gradient leakage have triggered various defence strategies, such as gradient perturbation [10, 13, 41, 42, 65], data obfuscation or sanitization [66–70], along with other techniques [49, 71–74]. Nonetheless, these methods generally involve compromises between privacy protection and computational efficiency, often requiring substantial computational resources due to the complexities of encryption technologies. The study in [75] delves into the factors in an FL system that could influence gradient leakage, such as batch size, image resolution, the type of activation function used, and the number of local iterations prior to the exchange of gradients. Various studies corroborate these findings; for instance, the DLG method indicates that activation functions need to be bi-differentiable. IG techniques can reconstruct images with resolutions up to $224 * 224$, whereas GRNN are compatible with batch sizes of 256 and image resolutions of $256 * 256$, also assessing how the frequency of local iterations might impact privacy leakage. Moreover, recent advancements by D. Scheliga, P. Mäder and M. Seeland [76] introduced an innovative model extension, named Privacy Enhancing Module (PRECODE), designed specifically to enhance privacy safeguards against potential leakages in FL systems. This adaptability highlights the evolving trend of FL security measures that

seek to enhance the balance between performance and privacy without imposing excessive computational requirements.

In traditional model training strategies, various optimisation algorithms can alter the low-level representations of gradients while preserving their high-level representations. For instance, unlike SGD which utilises raw gradients for model updates [77], Adam [50] adjusts gradient values using adaptive estimates based on first-order and second-order moments. This chapter introduces an alternative approach by employing gradients stand-in for global gradient aggregation, aimed at preventing the transmission of private information to the parameter server. After an epoch of local training, the local gradients are modified through the Adam optimisation algorithm, which utilises adaptive estimates of lower-order moments. Critical information, such as first-order and second-order moments, is retained locally, thereby rendering it impossible for an adversary to execute the forward-backward procedure without access to this data. Consequently, reconstructing local training data from the shared gradients becomes infeasible within the FL framework. The gradients stand-in calculated in this manner not only mitigate the risk of gradient leakage but also preserve model performance. Additionally, the simplicity and computational efficiency of the method, make it particularly attractive. The versatility of the approach is evident as it does not impose constraints on the model architecture, the local training optimisation strategy, the dataset, or the FL aggregation technique. Comprehensive evaluations on various benchmark networks and datasets have demonstrated that our method, termed AdaDefence, effectively addresses the identified privacy concerns. The implementation of AdaDefence is available on GitHub[1].

The structure of this chapter is organised as follows: Section 4.2 reviews related work on gradient leakage attacks along with existing defence mechanisms. Section 4.3 details the proposed method for defending against gradient leakage. Experimental results are presented in Section 4.4, where we compare the efficacy of our proposed AdaDefence with state-of-the-art attack methods. Finally, Section 4.5 concludes the chapter and outlines future research directions.

[1] `https://github.com/Rand2AI/AdaDefence`

4.2 Related Work

4.2.1 Attack Methods

Gradient leakage attacks pose a significant threat to privacy in both centralised and collaborative DL systems by potentially exposing private training data through leaked gradients. Such attacks are particularly prevalent in centralised systems, often through membership inference attacks as discussed by [78–83].

The work [10] was pioneer in examining data reconstruction from leaked gradients within collaborative systems, an effort furthered by [12] which introduced an optimisation-based method to enhance the stability of such attacks. They utilised magnitude-invariant cosine similarity measurements for the loss function and demonstrated that incorporating prior knowledge could significantly augment the efficiency of gradient leakage attacks. Expanding on these findings, the work [84] argued that gradient information alone might not suffice for revealing private data, thus they proposed the use of a pre-trained model, GIAS, to facilitate data exposure. In [85], the authors found that in image classification tasks, ground-truth labels could be easily discerned from the gradients of the last fully-connected layer, and that BN statistics markedly enhance the success of gradient leakage attacks by disclosing high-resolution private images.

Alternative approaches involve generative models, as explored in [86], which employed a GAN-based method for data recovery that replicates the training data distribution. Similarly, the work [87] developed mGAN-AI, a multitask discriminator-enhanced GAN, to reconstruct private information from gradients. GRNN, introduced [13], is capable of reconstructing high-resolution images and their labels, effectively handling large batch sizes. Likewise, in the work of GGL [14], a GAN with pre-trained and fixed weights was used. GGL differs in its use of the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) as the optimiser, reducing variability in the generated data, which, while not exact replicas, closely resemble the true data. This characteristic significantly enhances GGL's robustness, allowing it to counteract various defensive strategies including gradient noising, clipping, and compression. This series of developments highlights the dynamic and evolving nature of combating gradient leakage attacks in DL.

GLAUS [63] demonstrates a significant vulnerability in UGS methods by showing that they can still be susceptible to gradient leakage attacks. This is done by reconstructing private data points with considerable accuracy despite the supposed robust security measures of UGS. GLAUS circumvents the safeguards by approximating the gradient using leaked indices and

signs from the UGS's crafted random transformations. It effectively reduces the security of UGS frameworks to that of basic FL models without additional protections. It is capable of adaptively inferring the gradient without needing the exact gradient, utilizing an approximate gradient reconstructed through several steps. These include narrowing the gradient search range, estimating the magnitude of each gradient value, and revising the gradient signs. This method showcases that even when real gradients are not directly accessible, sensitive data can still be reconstructed, posing a serious security risk.

As shown in Table 4.1, gradient leakage attacks differ in how they exploit gradients but share a common vulnerability. They rely on access to raw, unmodified gradients containing directional and magnitude information. In this chapter we focus in detail on IG, GRNN, and GLAUS, since these represent state-of-the-art attacks that threaten practical FL deployments. AdaDefence is specifically designed to mitigate these by replacing raw gradients with stand-in derived from Adam's moment estimates, thereby concealing the very information these attacks require.

Table 4.1: Summary of representative gradient leakage attacks and their limitations.

Attack	**Mechanism**	**Limitation**
DLG [10]	Recovers inputs by optimising randomly initialised data until their gradients match the shared gradients.	Requires many optimisation steps and often produces noisy reconstructions on complex datasets.
iDLG [11]	Simplifies DLG by analytically inferring labels from gradients before optimisation.	Works mainly for simple models; accuracy decreases with deeper networks; batch size has to be 1.
IG [12]	Uses cosine similarity between observed and candidate gradients to guide image inversion.	Sensitive to gradient obfuscation; fails when precise magnitude information is hidden.
GRNN [13]	Employs a generative neural network to reconstruct inputs directly from gradients.	The reconstruction process for training the generative model is slow.
GLAUS [14]	Audits gradient leakage risks by aligning gradient-space representations.	Specifically tailored to attack UGS secure aggregation methods

4.2.2 Defence Methods

Numerous strategies [83] have been developed to safeguard private data against potential leakage via gradient sharing in FL. Techniques such as gradient perturbation, data obfuscation

or sanitization, DP, HE, and MPC have been employed to protect both the private training data and the publicly shared gradients [72].

The work in [10] assessed the efficacy of Gaussian and Laplacian noise types in protecting data, discovering that the magnitude of the noise's distribution variance is crucial, with a variance threshold above 10^{-2} effectively mitigating leakage attacks at the cost of significant performance degradation. The work [70] introduced a data perturbation method that maintains model performance while securing the privacy of training data. This approach transforms the input data matrix into a new feature space through a multidimensional transformation, applying variable scales of transformation to ensure adequate perturbation. However, this technique depends on a centralised server for generating global perturbation parameters and may distort the structural information in image-based datasets. The method proposed in [74] implemented DP to introduce noise into each client's training dataset using a per-example-based DP method, termed Fed-CDP. They proposed a dynamic decay method for noise injection to enhance defence against gradient leakage and improve inference performance. Despite its effectiveness in preventing data reconstruction from gradients, this method significantly reduces inference accuracy and incurs high computational costs due to the per-sample application of DP.

Both PRECODE [76] and FedKL [88] aim to prevent input information from propagating through the model during gradient computation. PRECODE achieves this by incorporating a probabilistic encoder-decoder module ahead of the output layer, which normalises the feature representations and significantly hinders input data leakage through gradients. This process involves encoding the input features into a sequence, normalising based on calculated mean and standard deviation values, and then decoding into a latent representation that feeds into the output layer. While effective, the additional computational overhead from the two fully-connected layers required by PRECODE limits its applicability to shallow DNNs due to high computational costs. In contrast, FedKL method introduces a key-lock module that manages the weight parameters with a hyper-parameter controlling the input dimension and an output dimension optimised for specific architectures; 16 for *ResNet-20* and *ResNet-32* on CIFAR-10 and CIFAR-100, and 64 for *ResNet-18* and *ResNet-34* on the ILSVRC2012 dataset. This configuration not only secures the gradients against leakage but also maintains manageable computational demands, making it feasible for more complex DNNs. Both FedKL and PRECODE require architectural modifications: FedKL introduces a key-lock mechanism at the batch normalisation layer, while PRECODE adds an encoder–decoder module ahead of the output layer. These changes bring extra computational cost and limit the applicability of the defence to certain

network types. AdaDefence, by contrast, does not alter the model structure or rely on encryption. Instead, it simply replaces raw gradients with Adam-based gradients stand-in, which already contain sufficient learning signals for aggregation but conceal the information that leakage attacks exploit.

Unlike existing methods, our proposed AdaDefence incorporates the Adam optimisation algorithm as an integrated component to enhance privacy protection in FL. Instead of transmitting raw gradients, AdaDefence transforms the original local gradients into a gradients stand-in, which is subsequently used for global aggregation. This stand-in retains the high-level latent representations of the original gradients, ensuring that model performance remains largely unaffected during each aggregation round. Moreover, the computation of the gradients stand-in, along with the retention of both first-order and second-order moment estimates at the local client level, prevents adversaries from reconstructing private training data. Since the crucial optimisation parameters remain inaccessible to potential attackers, gradient leakage becomes infeasible. This approach not only strengthens privacy safeguards within FL environments but also preserves the integrity and efficiency of the overall learning process.

4.3 Methodology

In the initial subsection, we provide an overview of how gradient leakage can expose private training data. Subsequently, we performed a mathematical analysis to demonstrate the efficacy of our proposed gradients stand-in in defending leakage attacks. This analysis explains how the gradients stand-in protects sensitive information while maintaining model performance. Finally, we present a comprehensive introduction to the proposed AdaDefence method.

4.3.1 Gradient Leakage

In the context of ML, particularly in neural network training, the gradient represents the direction of the steepest increase in the loss or cost function. To minimise the loss function and optimise model parameters, the training process follows the opposite direction of the gradient, i.e. gradient descent. This iterative optimisation technique adjusts model parameters step by step, progressively reducing the loss function to achieve convergence.

Definition 1 Gradient: Consider a function $f : \mathbb{R}^n \to \mathbb{R}^m$, which maps a vector x in n-dimensional space to a vector y in m-dimensional space, defined by $f(x) = y$ where $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$. The gradient of f with respect to x is represented by the Jacobian matrix as follows:

$$\frac{\partial f}{\partial x} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{pmatrix}$$

This matrix consists of partial derivatives where the element in the i-th row and j-th column, $\frac{\partial f_i}{\partial x_j}$, represents the partial derivative of the i-th component of y with respect to the j-th component of x.

The *Chain Rule* is a rule in calculus that describes the derivative of a composite function. In simple terms, if a variable z depends on y, and y depends on x, then z, indirectly through y, depends on x, and the chain rule helps in finding the derivative of z with respect to x. If $z = f(y)$ and $y = g(x)$, then:

$$\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx}$$

Many recent studies have highlighted the vulnerability of gradient data to leakage attacks. In their research, the work [12] demonstrated that for fully-connected layers, the gradients of the loss with respect to the outputs can reveal information about the input data. By employing the *Chain Rule*, it is possible to reconstruct the inputs of a fully-connected layer using only its gradients, independent of the gradients from other layers. Expanding on this, in FedKL [88], the authors explored this vulnerability in convolutional and BN layers across typical supervised learning tasks. They provided a detailed analysis of how gradients carry training data, which can be exploited by an attacker to reconstruct this data. Their results confirmed a strong correlation between the gradients and the input training data across linear and non-linear neural networks, including CNN and BN layers. Ultimately, they concluded that in image classification tasks, the gradients from the fully-connected, convolutional, and BN layers contain ample information about the input data and the true labels. This richness of information enables an attacker to reconstruct both the inputs and labels by regressing the gradients. We can conclude that:

Proposition 1 In image classification tasks, the gradients from the fully connected, convolutional and BN layers contain a great deal of information about the input data and the actual labels. These details allow an attacker to approximate these gradients and effectively reconstruct the corresponding inputs and labels.

4.3.2 Theoretical Analysis on Gradients Stand-in

Model Performance: In this part, we explore the influence of using gradients stand-in on model performance, and provide justification for employing them during global aggregation in a FL context. A gradients stand-in is a surrogate update vector computed locally on each client to replace the raw gradient during communication with the server. In this work, the stand-in is derived from Adam's optimisation process. Instead of transmitting the raw gradient g_t, the client transmits $\hat{g}_t$, computed by the first-order moment m_t and second-order moment v_t, which form the gradients stand-in used for aggregation. Unlike raw gradients, which carry direct directional and magnitude information that adversaries can invert to reconstruct training data, moment-based stand-ins obscure this information. The exponential averaging in Adam smooths out client-specific noise and conceals fine-grained patterns tied to individual data samples. As a result, stand-ins preserve the learning signal necessary for convergence but remove the sharp features that leakage attack methods rely on. To motivate this design, we begin by reviewing the Adam [50] optimisation algorithm, a method of SGD that incorporates momentum concepts. Each iteration in Adam involves computing the first-order and second-order moments of the gradients, followed by calculating their exponential moving averages to update the model parameters. This approach merges the strengths of the Adagrad [51] algorithm, which excels in handling sparse data, with those of the RMSProp [57] algorithm, designed to manage non-smooth data effectively. Collectively, these features enable Adam to deliver robust performance across a wide range of optimisation problems, from classical convex formulations to complex DL tasks. Further details on the Adam algorithm are presented in Algorithm 3.

In the local training phase of FL, we concentrate on the aggregated gradients from each training round rather than the gradients from individual iterations. Specifically, the local gradients that are transmitted to the global server represent the cumulative sum of all gradients produced during the local training iterations. Common FL aggregation methods, such as FedAvg [1], FedAdp [7], and FedOpt [46], employ these cumulative local gradients for global model updates. This approach effectively outlines the model's convergence path and direction within an FL framework. Moreover, studies such as Adam [50] optimisation algorithm has demonstrated superior convergence properties compared to traditional SGD, underscoring its widespread adoption in the DL domain. Based on these insights, we propose a novel method of representing local gradients using the Adam algorithm to enhance privacy and maintain model efficacy in FL. Experimental outcomes discussed in Section 4.4 confirm the viability of using these modified gradients for global aggregation, thereby supporting the integrity and

Algorithm 3: Adam optimisation Algorithm

Require: Initial learning rate α
Require: Exponential decay rates for moment estimates $\beta_1, \beta_2 \in [0,1]$
Require: Small constant for numerical stability ε
Require: Maximum number of iterations I
initialise weights ω_0
initialise the first-order moment vector $m_0 \leftarrow 0$
initialise the second-order moment vector $v_0 \leftarrow 0$
initialise the time-step $t \leftarrow 0$
for $t = 1$ to I **do**
  $g_t \leftarrow \nabla_\omega \mathscr{L}(\omega_{t-1})$ ▶Get gradients w.r.t. stochastic objective at time-step t
  $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1-\beta_1) \cdot g_t$ ▶Update biased first-order moment estimate
  $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1-\beta_2) \cdot g_t^2$ ▶Update biased second-order moment estimate
  $\hat{m}_t \leftarrow \frac{m_t}{1-\beta_1^t}$ ▶Compute bias-corrected first-order moment estimate
  $\hat{v}_t \leftarrow \frac{v_t}{1-\beta_2^t}$ ▶Compute bias-corrected second-order moment estimate
  $\omega_t \leftarrow \omega_{t-1} - \alpha \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\varepsilon}$ ▶Update parameters
end for
return ω_I ▶Output the optimised parameters

performance of the FL model.

Leakage defence: In the FedKL [88], it was demonstrated that private training data leakage through gradients sent to a global server is a significant concern. The authors of FedKL introduced a novel approach involving a key-lock pair to generate shift and scale parameters in the BN layer, which are typically trainable within the model. Crucially, in FedKL, these generated parameters are retained on the local clients. The primary aim of FedKL is to sever the transmission of private training data via gradients, thereby safeguarding against potential leakage by malicious servers. This method involves modifications to the network architecture, including additional layers that are responsible for generating the shift and scale parameters. While these changes have minimal impact on model performance, they significantly reduce the system's efficiency due to the increased computational overhead. In this chapter, we adhere to the principle of preventing the propagation of private training data through gradients. However, unlike FedKL, we achieve this without altering the network architecture or adding extra training layers, thus avoiding additional computational burdens. The proposed approach maintains system efficiency while still protecting against data leakage.

Claim 1 The gradients stand-in transmitted to the global server no longer contains information from which details of the private training data can be inferred.

Proof. Assuming g_r is the local gradients in the r-th training round and $\hat{g}_r$ is the gradients stand-in for global gradients aggregation. By applying the Adam in Algorithm 3, we have:

$$m_r = \beta_1 m_{r-1} + (1-\beta_1) g_r \tag{4.1}$$

$$v_r = \beta_2 v_{r-1} + (1-\beta_2) g_r^2 \tag{4.2}$$

$$\hat{m}_r = \frac{m_r}{1-\beta_1^r} \tag{4.3}$$

$$\hat{v}_r = \frac{v_r}{1-\beta_2^r} \tag{4.4}$$

$$\hat{g}_r = \frac{\hat{m}_r}{\sqrt{\hat{v}_r} + \varepsilon} \tag{4.5}$$

where β_1 and β_2 are set to 0.9 and 0.999, separately; the values in g_0 are initialised to all zeros; $g_r \in \mathbb{R}^D$, $m_r \in \mathbb{R}^D$, $v_r \in \mathbb{R}^D$, $\hat{m}_r \in \mathbb{R}^D$, $\hat{v}_r \in \mathbb{R}^D$ and $\hat{g}_r \in \mathbb{R}^D$, D is the dimension of the parameter space, m_r is the biased first-order moment estimate in the r-th training round, v_r is the biased second-order moments estimate, $\hat{m}_r$ is the bias-corrected first-order moment estimate and $\hat{v}_r$ is the bias-corrected second-order moment estimate. The derivative of the gradients stand-in, $\hat{g}_r$, w.r.t. the original local gradients, g_r, can be expressed as:

$$\frac{\partial \hat{g}_r}{\partial g_r} = \frac{\partial \hat{g}_r}{\partial \hat{m}_r}\frac{\partial \hat{m}_r}{\partial g_r} + \frac{\partial \hat{g}_r}{\partial \hat{v}_r}\frac{\partial \hat{v}_r}{\partial g_r} \tag{4.6}$$

Given (4.1) and (4.2), differentiating m_r and v_r w.r.t. g_r:

$$\frac{\partial m_r}{\partial g_r} = 1-\beta_1 \tag{4.7}$$

$$\frac{\partial v_r}{\partial g_r} = 2(1-\beta_2) g_r \tag{4.8}$$

Given (4.3) and (4.4), differentiating $\hat{m}_r$ and $\hat{v}_r$ w.r.t. g_r:

$$\frac{\partial \hat{m}_r}{\partial g_r} = \frac{\frac{\partial m_r}{\partial g_r}}{1-\beta_1^r} = \frac{1-\beta_1}{1-\beta_1^r} \tag{4.9}$$

$$\frac{\partial \hat{v}_r}{\partial g_r} = \frac{\frac{\partial v_r}{\partial g_r}}{1-\beta_2^r} = \frac{2(1-\beta_2) g_r}{1-\beta_2^r} \tag{4.10}$$

Given (4.5), using the quotient rule, where $u = \hat{m}_r$ and $z = \sqrt{\hat{v}_r} + \varepsilon$:

$$\frac{\partial \hat{g}_r}{\partial g_r} = \frac{\frac{\partial \hat{m}_r}{\partial g_r} z - u \frac{\partial}{\partial g_r}(\sqrt{\hat{v}_r} + \varepsilon)}{z^2} \tag{4.11}$$

Calculating $\frac{\partial}{\partial g_r}(\sqrt{\hat{v}_r}+\varepsilon)$:

$$\begin{aligned}\frac{\partial\sqrt{\hat{v}_r}}{\partial g_r} &= \frac{1}{2\sqrt{\hat{v}_r}}\cdot\frac{\partial\hat{v}_r}{\partial g_r}\\ &= \frac{1}{2\sqrt{\hat{v}_r}}\cdot\frac{2(1-\beta_2)g_r}{1-\beta_2^r}\\ &= \frac{(1-\beta_2)g_r}{\sqrt{\hat{v}_r}(1-\beta_2^r)}\end{aligned} \tag{4.12}$$

Then, differentiating $\hat{g}_r$ w.r.t. g_r:

$$\frac{\partial\hat{g}_r}{\partial g_r} = \frac{(\frac{1-\beta_1}{1-\beta_1^r})(\sqrt{\hat{v}_r}+\varepsilon)-\hat{m}_r\cdot\frac{(1-\beta_2)g_r}{\sqrt{\hat{v}_r}(1-\beta_2^r)}}{(\sqrt{\hat{v}_r}+\varepsilon)^2} \tag{4.13}$$

Substitute $\hat{m}_r$ and $\hat{v}_r$ in (4.13) referring to (4.1), (4.2), (4.3) and (4.4). To simplify the equation. We define:

$$\begin{aligned}V &= \sqrt{\hat{v}_r}\\ &= \sqrt{\frac{\beta_2 v_{r-1}+(1-\beta_2)g_r^2}{1-\beta_2^r}}\end{aligned} \tag{4.14}$$

Then, the equation would be:

$$\frac{\partial\hat{g}_r}{\partial g_r} = \frac{(1-\beta_1)(V+\varepsilon)-(\frac{(1-\beta_2)\beta_1 m_{r-1}+(1-\beta_2)(1-\beta_1)g_r}{V\sqrt{1-\beta_2^r}})g_r}{(1-\beta_1^r)(V+\varepsilon)^2} \tag{4.15}$$

Given $\beta_1 = 0.9$, $\beta_2 = 0.999$ and ε is a very small value that can be ignored, considering both large or small r, we approximate:

$$(V+\varepsilon)\approx\alpha g_r \tag{4.16}$$

where α is a non-zero scaling variant. In the end, we have:

$$\frac{\partial\hat{g}_r}{\partial g_r} = \frac{-\beta_1 m_{r-1}}{\alpha(1-\beta_1^r)}\cdot\frac{1}{g_r^2} \tag{4.17}$$

In (4.17), the parameters α and $(1-\beta_1)$ are non-zero values, while m_{r-1} is exclusively preserved within the confines of local clients. This specific configuration ensures that the original local gradients, g_r, are safeguarded against deep leakage from gradients by potential malicious attackers. Consequently, the proposed gradients stand-in, $\hat{g}_r$, effectively obstructs any attempts to deduce or reconstruct private training data from the local gradients. This safeguarding mechanism enhances the security of the data during the FL training process, providing a robust defence against potential data privacy breaches. ■

Proof of the Approximation: In the context of simplifying the derivative $\frac{\partial \hat{g}_r}{\partial g_r}$ and considering whether V can be approximated by αg_r, we need to analyse the terms carefully.

Claim 2 For large values of r, the model has undergone sufficient training to approach convergence. At this stage, V can be closely approximated by αg_r, which simplifies the expression for the derivative $\frac{\partial \hat{g}_r}{\partial g_r}$.

Proof. Given the value $\beta_2 = 0.999$, evaluate the assumption that $V \approx \alpha g_r$, where:

$$V = \sqrt{\frac{\beta_2 v_{r-1} + (1-\beta_2) g_r^2}{1-\beta_2^r}}$$

Given the value of β_2, examine this approximation more critically. We first evaluate β_2^r. For larger r, β_2^r becomes quite small because $\beta_2 = 0.999$ is very close to 1. The decay term $1-\beta_2^r$ is crucial to understanding V's behaviour. To compute $1-\beta_2^r$, say $r = 1000$ for example:

$$\beta_2^{1000} = (0.999)^{1000}$$

This can be computed as:

$$\beta_2^{1000} \approx e^{1000 ln(0.999)}$$

Referring to Taylor expansion $ln(1-x) \approx -x$ for small x, we have:

$$ln(0.999) \approx -0.001$$
$$1000 ln(0.999) \approx -1$$

Hence,

$$\beta_2^{1000} \approx e^{-1} \approx 0.3679$$

Given the computation above, we have:

$$V \approx \sqrt{\frac{\beta_2 v_{r-1} + (1-\beta_2) g_r^2}{0.6321}}$$

Assuming $v_{r-1} \approx g_r^2$ for simplicity (which can be the case as gradients do not change dramatically over large iterations), then:

$$V \approx \sqrt{\frac{0.999 g_r^2 + 0.001 g_r^2}{0.6321}} = \sqrt{\frac{g_r^2}{0.6321}} \approx \frac{g_r}{0.795} \approx 1.258 g_r$$

With $\beta_2 = 0.999$, the term V is approximate $1.258 g_r$ when assuming $v_{r-1} \approx g_r^2$ at larger r. So in this case, $\alpha \approx 1.258$ ■

Claim 3 Given the condition of small r, the model does not converge effectively. Under this circumstance, the variable V can be accurately approximated by αg_r.

Proof. With the values $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and considering small r, such as $r = 1$ or $r = 2$, we explore how V behaves. This scenario is typical at the beginning of the training, where initial values for moving averages m_0 and v_0 are set to zero, and the initial gradient g_0 is randomly initialised.

Calculation for $r = 1$:

$$
\begin{aligned}
m_1 &= \beta_1 m_0 + (1-\beta_1) g_0 = 0.1 g_0 \\
v_1 &= \beta_2 v_0 + (1-\beta_2) g_0^2 = 0.001 g_0^2 \\
\hat{m}_1 &= \frac{m_1}{1-\beta_1^1} = g_0 \\
\hat{v}_1 &= \frac{v_1}{1-\beta_2^1} = g_0^2 \\
V &= \sqrt{\hat{v}_1} = \sqrt{g_0^2} = g_0
\end{aligned}
$$

Calculation for $r = 2$, assume $g_1 \approx g_0$ (for simplicity, assuming the gradient does not change too much):

$$
\begin{aligned}
m_2 &= \beta_1 m_1 + (1-\beta_1) g_1 = 0.19 g_0 \\
v_2 &= \beta_2 v_1 + (1-\beta_2) g_1^2 = 0.002 g_0^2 \\
\hat{m}_2 &= \frac{m_2}{1-\beta_1^2} \approx g_0 \\
\hat{v}_2 &= \frac{v_2}{1-\beta_2^2} \approx g_0^2 \\
V &= \sqrt{\hat{v}_2} \approx \sqrt{g_0^2} = g_0
\end{aligned}
$$

With the values $\beta_1 = 0.9$ and $\beta_2 = 0.999$, the term V is approximate g_r when assuming $g_1 \approx g_0$ at the beginning of the training (small r). So in this case, $\alpha \approx 1$. ■

Derivation Process: Now, we are going to provide specific derivation process of (4.17). Given that $(V + \varepsilon) \approx \alpha g_r$ and considering ε to be negligible for simplification, we will further simplify the derivative of $\hat{g}_r$ w.r.t. g_r:

$$
\hat{g}_r = \frac{\hat{m}_r}{\sqrt{\hat{v}_r} + \varepsilon} \approx \frac{\hat{m}_r}{\sqrt{\hat{v}_r}} = \frac{\hat{m}_r}{V} = \frac{\hat{m}_r}{\alpha g_r}
$$

According to the quotient rule $(\frac{u}{v})' = \frac{u'v-uv'}{v^2}$, we have:

$$\hat{m}_r = \frac{\beta_1 m_{r-1} + (1-\beta_1) g_r}{1-\beta_1^r}$$
$$\frac{\partial \hat{m}_r}{\partial g_r} = \frac{1-\beta_1}{1-\beta_1^r}$$

Hence,

$$\begin{aligned}
\frac{\partial \hat{g}_r}{\partial g_r} &= \frac{(\frac{1-\beta_1}{1-\beta_1^r})(\alpha g_r) - (\frac{\beta_1 m_{r-1}+(1-\beta_1)g_r}{1-\beta_1^r})(\alpha)}{(\alpha g_r)^2} \\
&= \frac{\frac{\alpha(1-\beta_1)g_r - \alpha(\beta_1 m_{r-1}+(1-\beta_1)g_r)}{1-\beta_1^r}}{\alpha^2 g_r^2} \\
&= \frac{\alpha(1-\beta_1)g_r - \alpha\beta_1 m_{r-1} - \alpha(1-\beta_1)g_r}{\alpha^2 g_r^2 (1-\beta_1^r)} \\
&= \frac{-\beta_1 m_{r-1}}{\alpha g_r^2 (1-\beta_1^r)}
\end{aligned}$$

4.3.3 AdaDefence in FL

Federated Learning: The foundational method underpinning recent FL techniques assumes the presence of multiple clients, represented as $\mathscr{C}$, each possessing their own local datasets $\mathscr{D}$. The learning task is defined by a model $\mathscr{F}$ with parameters ω. The local gradient g_i for each client i is computed as follows:

$$g_i = \frac{1}{||d_i||} \nabla_\omega \sum_j \mathscr{L}(\mathscr{F}(x^{(j)}; \omega), y^{(j)}) \quad \forall i \in \mathscr{C}, \tag{4.18}$$

where $||d||$ denotes the number of samples, used here to normalise the gradients by the size of the dataset d_i. In this formula, $x^{(j)}$ and $y^{(j)}$ are the input features and corresponding labels of the j-th example in the i-th local dataset.

After computing the local gradients, the server aggregates these gradients and performs an averaging process to update the global model parameters ω_r. This process is mathematically represented as:

$$\omega_r = \omega_{r-1} - \frac{1}{|\mathscr{C}|} \sum_{i=1}^{|\mathscr{C}|} g_i, \tag{4.19}$$

where $|\mathscr{C}|$ denotes the total number of clients. It is assumed that all local datasets are of equal size, i.e., $||d_i|| = ||d_k||$ for any $d_i, d_k \in \mathscr{D}$. This assumption simplifies the model updating process by maintaining uniform influence for each client's gradient.

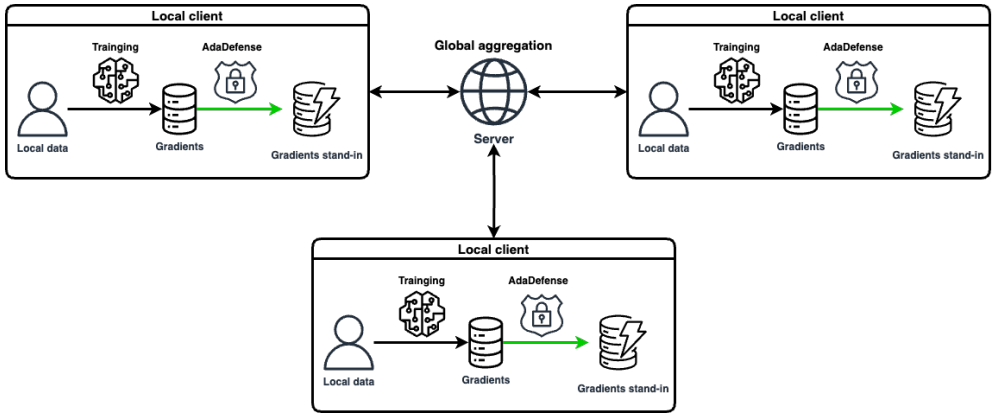


Figure 4.1: Illustration of AdaDefence in FL

AdaDefence: emphasising the innovative nature of AdaDefence is essential, as it integrates seamlessly into existing frameworks without altering the fundamental architecture of the model or the FL strategy it supports. AdaDefence is designed to act as an unobtrusive extension, focusing on the manipulation of local gradients. This process involves the creation of gradients stand-in that is used in the global aggregation phase. In a standard FL setup, the global model is initially crafted on a central server and then distributed to various clients. These clients perform secure local training and then send their local gradients stand-in back to the server. At the server, these gradients stand-in are aggregated to update and refine the global model. This cycle of local training and global updating repeats, with the server coordinating the aggregation of gradients to produce each new iteration of the global model. During this process, the server remains oblivious to all clients' first-order and second-order moments information. This omission is intentional and essential for security. This structure ensures that gradients stand-in cannot be exploited to reconstruct private, local training data. By maintaining this separation, AdaDefence strengthens the FL system against potential data reconstruction attacks. Consequently, in every round of local training that follows, each client adjusts its model based on the updated global model distributed by the server, thus advancing the collective learning process without compromising on privacy or security. See Figure 4.1 for an illustration of AdaDefence in FL.

4.3.4 Theoretical Justification of Gradients Stand-in

The central idea behind AdaDefence is that by transmitting Adam's moment estimates rather than raw gradients, clients prevent adversaries from recovering private data. Gradient leakage attacks rely on precise information about the direction and magnitude of raw gradients to solve

an inverse problem that reconstructing the input that produced them. In contrast, the gradients stand-in replaces raw gradients with their bias-corrected first-order and second-order moments, respectively. This transformation has three privacy-preserving effects:

- **Obfuscation of magnitude:** The exponential moving average smooths gradients across timesteps, concealing the sharp per-sample variations that leakage attacks exploit.
- **Loss of direct directionality:** Moment estimates aggregate gradient history rather than exposing the exact directional vector at each iteration. This prevents attacks such as IG or GRNN from aligning candidate inputs with the observed gradient direction.
- **Preservation of learning signal:** Although the raw gradient is hidden, the aggregated moment estimates still capture enough optimisation information for effective global training.

In summary, by keeping only Adam's moment estimates on each client, the updates sent to the server no longer carry the information required for gradient inversion. Without access to the precise gradient vector, an adversary cannot solve the inverse mapping from gradient space back to input space. This explains why AdaDefence is effective against a wide range of leakage attacks, a claim that is formally supported by the derivative proofs provided.

4.4 Experiments

In this section, we begin by describing the datasets and metrics used for benchmark evaluation. We then detail a series of experiments designed to evaluate the performance of DNNs across two aspects. The first set of experiments investigates the impact of using the proposed gradients stand-in on the prediction accuracy. The second set assesses the effectiveness of this approach in defending against state-of-the-art gradient leakage attacks, specifically those involving GRNN, IG and GLAUS.

4.4.1 Benchmarks and Metrics

In our study, we conducted experiments using four well-known public datasets: MNIST [61], CIFAR-10 [60], CIFAR-100 [60], and ILSVRC2012 [62]. To assess the quality of images generated by our method, we employed four evaluation metrics: Mean Square Error (MSE), Peak Signal-to-Noise Ratio (PSNR), Learned Perceptual Image Patch Similarity (LPIPS) [89], and

Structure Similarity Index Measure (SSIM) [90]. To optimise the conditions for gradient leakage attacks and effectively evaluate the defence capabilities, we set the batch size to one. This configuration allows for a direct comparison between the reconstructed image and the original image. The MSE metric quantifies the pixel-wise L2 norm difference between the reconstructed image $\hat{X}$ and the original image X, defined as $MSE(X,\hat{X}) = ||X - \hat{X}||_2$. PSNR, another criterion for image quality assessment, is calculated using the formula $PSNR(X,\hat{X}) = 10 \cdot \lg(\frac{255^2}{MSE(X,\hat{X})})$, where a higher PSNR value indicates a higher similarity between the two images. For perceptual similarity, we utilised the LPIPS metric, which measures the similarity between two image patches using pre-trained networks such as *VGGNet* [91] and *AlexNet* [20]. This metric aligns with human visual perception, where a lower LPIPS score indicates greater perceptual similarity between the compared images. The SSIM index evaluates the change in image structure, where a higher SSIM value suggests less distortion and thus a better quality of the reconstructed image. All neural network models were implemented using the PyTorch framework [58], ensuring robust and efficient computation.

4.4.2 Model Performance

Table 4.2: Testing accuracy (%) of models trained under various configurations. The highest accuracy achieved is highlighted in red, while the second highest is denoted in blue. Here, "C-10" and "C-100" refer to CIFAR-10 and CIFAR-100 datasets, respectively. "AD" stands for the AdaDefense module. The accuracy figures listed in the "Reference" row are sourced from corresponding papers.

Model	*LeNet* (32*32)	*ResNet-20* (32*32)	*ResNet-32* (32*32)	*ResNet-18* (224*224)	*ResNet-34* (224*224)	*VGG-16* (224*224)
Dataset	MNIST	C-10 C-100	C-10 C-100	C-10 C-100	C-10 C-100	C-10 C-100
Centralized	98.09	91.63 67.59	92.34 70.35	91.62 72.15	92.20 73.21	89.13 63.23
FedAvg	98.14	91.20 58.58	91.37 61.91	89.50 68.59	89.27 68.30	88.13 61.91
FedAvg w/ AD	97.80	89.15 61.81	90.20 63.64	89.29 68.76	89.38 68.38	87.13 58.32
Reference	99.05 [92]	91.25 - [21]	92.49 - [21]	- -	- -	- -

To comprehensively assess the impact of gradients stand-in on model performance, we designed and evaluated three training paradigms: centralised training, standard federated training using the FedAvg algorithm, and federated training with the proposed AdaDefence mechanism, where stand-in gradients are used in place of true gradients during aggregation. For our experiments, we selected six widely used DNN architectures, *LeNet* [92], *ResNet-20*, *ResNet-32*, *ResNet-18*, *ResNet-34* [21], and *VGGNet-16* [91], across three benchmark datasets:

MNIST, CIFAR-10, and CIFAR-100. These datasets vary in resolution, ranging from 32×32 to 224×224 pixels, and differ in complexity, with CIFAR-100 representing a more challenging classification task due to its larger number of classes. Across all experimental configurations, we trained a total of thirty-three models under eleven distinct settings, ensuring a robust and fair comparison between the different training approaches. The full results are summarised in Table 4.2, where we report the testing accuracy (%) of each model and highlight the highest and second-highest values for clarity. This extensive evaluation allows us to quantify the relative performance of traditional centralised training, standard FL, and our gradients stand-in approach under diverse model and data conditions. As expected, centralised training consistently yields the highest accuracy across most settings, benefiting from direct access to the entire training dataset. For example, the *LeNet* model trained on MNIST achieves 99.05% accuracy, reported in paper [92], while *ResNet-32* achieves 92.49% on CIFAR-10, reflecting the strong predictive capability of centralised setups. However, a noticeable drop in accuracy is observed when moving from CIFAR-10 to CIFAR-100 for all models, confirming that the increased class granularity in CIFAR-100 makes the task more complex. Models such as *ResNet-34* and *VGG-16*, which are deeper in architecture, also exhibit this drop, though they maintain relatively strong performance, with accuracies of 73.21% and 63.23% on CIFAR-100, respectively.

Under the FedAvg framework, model performance remains competitive despite the lack of access to centralised data. In many cases, the accuracy achieved by FedAvg approaches that of the centralised baseline. For instance, *LeNet* on MNIST achieves 98.14%, only slightly below the centralised result. Additionally, FedAvg secures four out of eleven second-highest testing accuracies, demonstrating that it retains strong predictive performance under decentralised conditions. This result reinforces the practicality of FL for scenarios where data sharing is restricted. Interestingly, integrating the AdaDefence module, which replaces true gradients with stand-in gradients during aggregation, does not significantly degrade model accuracy. On the contrary, our experiments indicate that the use of stand-in gradients can preserve and, in some instances, improve predictive performance. For example, *ResNet-20* trained on CIFAR-100 using AdaDefence achieves 61.81% accuracy, outperforming the standard FedAvg baseline by 5.51%. This suggests that AdaDefence not only provides a layer of privacy protection against gradient leakage attacks but also contributes to more stable training under data heterogeneity. Furthermore, five out of eleven second-highest accuracies were obtained using AdaDefence, validating its robustness across different architectures and datasets. In summary, the results presented in Table 4.2 show that while centralised training still offers the best performance

overall, both FedAvg and AdaDefence deliver competitive results under privacy-preserving federated settings. Notably, the gradients stand-in mechanism used in AdaDefence enables secure model training without sacrificing model utility, offering a practical balance between privacy and performance in FL environments.

4.4.3 Defence Performance

In evaluating the robustness of defence mechanisms against gradient leakage attacks, our study incorporated four advanced attack methodologies. These included GRNN [13], IG [12] and GLAUS [63]. For the datasets MNIST, CIFAR-10, and CIFAR-100, experiments were conducted using a batch size of one and an image resolution of $32 * 32$. However, for the ILSVRC2012 dataset, a higher resolution of $256 * 256$ was selected rather than the usual $224 * 224$. This adjustment was necessary because GRNN can only process image resolutions that are powers of two. To accommodate this requirement, images of lower resolutions were upscaled through linear interpolation to match the required dimensions. This approach ensures that the integrity and comparability of the results across different datasets and resolution settings are maintained.

In Table 4.3, we present qualitative results that demonstrate the comparative effectiveness of various defence mechanisms against targeted leakage attacks. These defences include GRNN and IG, which have been implemented across different backbone networks and benchmark datasets. The table provides a clear view of how each defence strategy performs, allowing for an assessment of their efficacy in a controlled environment.

As discussed in the paper [13], DLG is inadequate in several experimental scenarios, leading us to exclude its results from our analysis. Instead, we focused on GRNN, which has proven more effective. Our proposed AdaDefence that efficiently safeguards against the leakage of true images during gradient-based reconstruction processes. In our experiments using GRNN with a *ResNet-18* model at an image resolution of $256 * 256$, the reconstructed images retained only minimal and non-identifiable details from the original images. This indicates a significant protection against the exposure of private data. For other architectures, such as *LeNet* and *ResNet-20*, the results were even more promising, with the reconstructed images being completely unrecognisable and distorted. On the other hand, the GRNN is capable of revealing true labels from gradients. Consequently, we have included these labels in Table 4.3 for comprehensive analysis. It is evident from our observations that when models are trained using our AdaDefence module, no correct labels are inferred from gradients. This further indicates the effectiveness of AdaDefence in enhancing model security against gradient-based label inference attacks.

Table 4.3: Comparison of image reconstruction using GRNN and IG without and with AdaDefense.

Model		*LeNet* (32*32)			*ResNet-20* (32*32)			*ResNet-18* (256*256)		
Dataset		MNIST	C-10	C-100	MNIST	C-10	C-100	C-10	C-100	ILSVRC
GRNN	True									
		four	truck	keyboard	five	frog	cattle	bird	sunflower	tench
	w/o									
		four	truck	keyboard	five	frog	cattle	bird	sunflower	tench
	w/									
		zero	ship	bowl	four	horse	mushroom	dog	oak tree	soccer ball
IG	True									
		two	airplane	willow	zero	horse	rose	truck	beaver	jay
	w/o									
		-	-	-	-	-	-	-	-	-
	w/									
		-	-	-	-	-	-	-	-	-

Furthermore, we compared the reconstruction capabilities of IG with those of GRNN. The findings confirmed that GRNN outperforms IG in safeguarding private training data across all tested configurations. Overall, our AdaDefence consistently demonstrated robust protection of private training information from gradient-based reconstructions in our experiments.

In the recent publication on the method GLAUS, it is noted that the official code supports only a specially designed architecture, *MNIST-CNN*, for use with the MNIST dataset. Attempts to adapt this method for other neural networks and datasets were unsuccessful. Consequently, the results presented in Table 4.5 are based solely on experiments conducted using the original, unmodified official code. This table highlights certain inherent shortcomings of GLAUS, even in the absence of our defence method. These limitations are generally considered acceptable, given that GLAUS is specifically tailored to attack UGS secure aggregation methods within the FL framework. Notably, when our AdaDefence module is integrated into the system, GLAUS is rendered ineffective in all trials concerning image reconstruction and label inference.

Table 4.4: Quantitative comparison of GRNN and IG without and with AdaDefense module. The results are computed from the generated images and their associated true images.

Method	**Model**	**Dataset**	MSE↓		PSNR↑		LPIPS-V↓		LPIPS-A↓		SSIM↑	
			w/o	w/	w/o	w/	w/o	w/	w/o	w/	w/o	w/
GRNN	*LeNet* (32*32)	MNIST	0.65	173.44	50.11	25.74	0.05	0.75	0.01	0.56	1.000	-0.003
		C-10	1.97	143.19	45.28	26.57	0.00	0.56	0.00	0.35	0.997	0.000
		C-100	1.79	145.36	45.85	26.51	0.00	0.55	0.00	0.38	0.998	0.003
	ResNet-20 (32*32)	MNIST	3.55	125.81	44.04	27.14	0.12	0.70	0.01	0.47	0.954	0.003
		C-10	12.16	85.43	40.68	28.84	0.00	0.44	0.00	0.25	0.973	0.030
		C-100	22.30	86.43	37.37	28.84	0.00	0.43	0.00	0.24	0.958	0.053
	ResNet-18 (256*256)	C-10	17.36	70.76	37.55	29.69	0.01	0.43	0.01	0.57	0.955	0.151
		C-100	32.56	64.37	34.67	30.19	0.02	0.31	0.02	0.28	0.917	0.434
		ILSVRC	18.82	59.81	37.31	30.41	0.05	0.41	0.04	0.39	0.891	0.212
IG	*LeNet* (32*32)	MNIST	19.23	90.62	35.29	28.56	0.06	0.50	0.03	0.32	0.825	0.300
		C-10	22.16	101.09	34.68	28.08	0.18	0.63	0.06	0.30	0.637	0.060
		C-100	20.45	92.32	35.02	28.48	0.12	0.53	0.06	0.31	0.623	0.100
	ResNet-20 (32*32)	MNIST	49.25	60.81	31.23	30.30	0.22	0.27	0.15	0.24	0.361	0.125
		C-10	39.35	52.31	32.27	30.99	0.21	0.27	0.16	0.16	0.299	0.098
		C-100	38.19	47.25	32.37	31.41	0.22	0.31	0.10	0.20	0.309	0.110
	ResNet-18 (256*256)	C-10	64.38	45.54	30.11	32.08	0.35	0.32	0.22	0.21	0.325	0.476
		C-100	78.44	63.13	29.29	30.19	0.37	0.28	0.30	0.27	0.355	0.510
		ILSVRC	82.46	79.16	28.99	29.20	0.64	0.49	0.56	0.48	0.103	0.266

Table 4.5: Image reconstruction for GLAUS without and with AdaDefense.

Model		*MNIST-CNN* (28*28)				
Dataset		MNIST				
GLAUS	True					
		eight	four	nine	seven	five
	w/o					
		eight	four	nine	seven	five
	w/					
		five	five	five	five	five

4.4.4 Quantitative Analysis

To conduct a quantitative assessment of our proposed AdaDefence module in comparison to existing state-of-the-art methods for mitigating gradient leakage, we utilised four distinct evaluation metrics: MSE for pixel-level error, PSNR for overall image fidelity, LPIPS with both *VGGNet* and *AlexNet* for perceptual similarity, and SSIM for structural similarity. The evaluation results, as detailed in Table 4.4, were derived from comparisons between generated images and their respective original images. Our analysis consistently revealed that images produced by models incorporating the AdaDefence module exhibited significantly lower degrees of similarity when compared to those from models lacking this feature. This outcome demonstrates the efficacy of the AdaDefence module in providing robust defence against gradient leakage attacks.

To enhance the clarity of our analysis on the defensive capabilities of AdaDefence, we have concentrated our examination primarily on GRNN. This focus is driven by our observations that GRNN exhibits a significantly stronger attack capability compared to IG. Through this targeted analysis, we aim to demonstrate more effectively the robustness of AdaDefence against more

potent threats. Starting with MSE, we see that in every case, the introduction of AdaDefence results in a dramatic increase in MSE, which implies a higher error rate between generated and true images when the defence is active. For instance, the MSE for the *LeNet* model on the MNIST dataset jumps from 0.65 without AdaDefence to 173.44 with AdaDefence, representing an astonishing increase of approximately $26,600\%$. This trend of increased error rates with AdaDefence is consistent across all models and datasets, highlighting a significant protection of using AdaDefence in terms of gradients leakage. Conversely, the impact on PSNR, which measures image quality (higher is better), shows a general decline when AdaDefence is applied. For example, PSNR for the *ResNet-18* model on the ILSVRC dataset decreases from 37.31 to 30.41, a decrease of about 18.5%. LPIPS, evaluated using both *VGGNet* and *AlexNet*, measures perceptual similarity (lower is better). All the results from GRNN perform significantly worse when the AdaDefence module is adapted into the model. Lastly, the SSIM metric, which evaluates the structural similarity between the generated and true images, mostly shows a decline with AdaDefence. For the *ResNet-18* model on the CIFAR-10 dataset, SSIM decreases significantly from 0.955 to 0.151, about a 84.18% reduction. This supports the view that AdaDefence can protect against specific vulnerabilities.

Across all benchmark datasets and reconstruction attacks, the results show that AdaDefence significantly weakens an adversary's ability to recover input data. Compared with undefended baselines, we observe dramatic increases in MSE, confirming that pixel-level reconstruction error rises sharply under our defence. At the same time, PSNR values drop by approximately 25% to 40%, reflecting a substantial decline in overall image fidelity. Perceptual metrics show similar trends: LPIPS scores increase by a large margin, indicating greater perceptual dissimilarity between the reconstructed and true images, while SSIM values decrease consistently, demonstrating a loss of structural similarity. Taken together, these metrics confirm that gradients stand-in prevent accurate data reconstruction, forcing attackers to produce outputs that are noisy, visually degraded, and structurally inconsistent, while the global model still maintains its learning performance.

4.5 Conclusion

In this chapter, we introduce, AdaDefence, a novel defence mechanism against gradient leakage tailored for the FL framework. The gist of AdaDefence is to use Adam to generate local gradients stand-in for global aggregation, which effectively prevents the leakage of input data information through the local gradients. We provide a theoretical demonstration of how the

gradients stand-in secures input data within the gradients and further validate its effectiveness through extensive experimentation with various advanced gradient leakage techniques.

Our results, both quantitative and qualitative, indicate that integrating the proposed method into the model precludes the possibility of reconstructing input images from the publicly shared gradients. For GRNN, the attack relies on learning a mapping between raw gradients and the original images through a generative model. Because AdaDefence replaces raw gradients with stand-in that retain only the moment estimates, the fine-grained directional information needed by GRNN is no longer available, so reconstruction fails. For IG, the attack depends on cosine similarity between observed gradients and candidate gradients to iteratively refine the reconstructed image. Since the stand-in does not preserve exact magnitudes or directions, the cosine similarity objective cannot be satisfied, and IG fails to converge. For GLAUS, the method exploits unbiased gradients sampling to approximate real gradients and then reconstruct private data from these approximations. With AdaDefence, only stand-in is shared, so the unbiased sampling assumption is broken, making GLAUS ineffective in all cases. Furthermore, we evaluate the impact of the defence plug-in on the model's classification accuracy, ensuring that the defence mechanism does not detract from the model's performance. We are excited to make the implementation of AdaDefence available for further research and application.

In summary, the core contribution of this chapter is the introduction of AdaDefence, a defence mechanism that secures FL by replacing raw gradients with Adam-based gradients stand-in. This approach conceals sensitive information from leakage attacks while preserving model utility and convergence efficiency, offering a lightweight alternative to noise-injection or encryption-based methods. As a natural next step, future work will investigate layer-wise vulnerability, examining how different layers contribute to potential information leakage. Extending this analysis to heterogeneous client settings will further test the resilience of AdaDefence and guide the design of even more robust privacy-preserving strategies in FL.

Chapter 5

Cross-Stock Trend Integration for Enhanced Predictive Modelling in Federated Learning

In this chapter, we focus on the third topic in FL, namely its real-world applicability in financial modelling. Stock price prediction is a critical area of financial forecasting, traditionally approached by training models on the historical price data of individual stocks. While such single-stock models can capture local dynamics, they fail to exploit correlations across stocks that may provide richer predictive signals. This limitation narrows their ability to model broader market behaviour. The method proposed here, CSTI, addresses this challenge by merging local patterns into a global representation that captures dependencies across assets. Our strategy is inspired by FL, which enables decentralised training while preserving privacy. In CSTI, models are first trained on individual stock data and then integrated into a unified global model, which is subsequently fine-tuned to maintain local relevance. This allows parallel training of stock-specific models, efficient use of computational resources, and the aggregation of global insights without requiring data sharing. Importantly, CSTI builds on the FL techniques developed in the previous chapters. Chapter 3 showed how adaptive, element-wise aggregation can stabilise global models under heterogeneous data distributions. This principle carries over to CSTI, where aggregation across different stocks faces similar heterogeneity. Chapter 4 introduced AdaDefence to counter gradient leakage attacks, a concern that is particularly acute in finance, where transaction data are highly sensitive and tightly regulated. By drawing on these advances, CSTI demonstrates how federated aggregation and privacy protection can be combined in a

practical financial forecasting application. We conducted extensive experiments to evaluate CSTI, showing that it outperforms benchmark models and improves the predictive capability of state-of-the-art methods. These results highlight the efficacy of CSTI in advancing stock price prediction and illustrate how the broader techniques of secure and efficient FL developed in this thesis can be applied in a domain where both performance and confidentiality are paramount.

5.1 Introduction

Stock price prediction has long been a cornerstone of financial research, with its origins rooted in attempts to model and forecast market behaviour for informed decision-making. Early efforts in stock price prediction were dominated by statistical models, such as Autoregressive Integrated Moving Average (ARIMA) [93–95] and exponential smoothing [96, 97] methods, which relied on historical price data to identify patterns and trends. These models were effective within their time but were limited by their linear assumptions and inability to capture the complexities of financial markets.

The advent of computational advancements and the rise of ML brought a paradigm shift to stock price prediction. Techniques such as SVM [98], DT [99], and ensemble methods like Random Forest (RF) [100] gained popularity for their ability to model non-linear relationships in financial data. With the availability of large datasets and increased computational power, DL approaches, including RNN [101], LSTM [37], Gated Recurrent Unit (GRU) [102], and Transformer [25], emerged as dominant methods. These models excelled at capturing temporal dependencies and intricate patterns, significantly improving prediction accuracy. Despite these advancements, most traditional ML and DL approaches focus on training models using individual stock data, often neglecting the inter-dependencies between stocks. Based on these traditional approaches, many novel methods are proposed aiming to achieve higher accuracy of prediction, such as, FilterNet [103], FreTS [104], DLinear [105], TimesNet [106] and PatchTST [107]. FilterNet is tailored for time series forecasting, emphasising the extraction of informative temporal patterns through learnable frequency filters. By selectively amplifying or attenuating specific components of time series signals, FilterNet effectively captures both high-frequency and low-frequency information. This design addresses challenges such as vulnerability to high-frequency noise and inefficiencies in full-spectrum utilization, which are common in Transformer-based models. FreTS introduces a novel approach by applying Multi-Layered Perceptron (MLP) in the frequency domain. This method involves transforming time-domain signals into their frequency-domain representations using the Discrete Fourier

Transform (DFT). FreTS then employs redesigned MLP to learn the real and imaginary parts of these frequency components, capturing global dependencies and concentrating on key frequency components with compact signal energy. Operating on both inter-series and intra-series scales, FreTS effectively learns channel-wise and time-wise dependencies. DLinear is a simple and fast model for long-horizon forecasting that utilises linear layers for trend and seasonality components, offering competitive accuracy with reduced complexity. TimesNet is a CNN-based model that transforms time series data into 2D tensors using the Fast Fourier Transform (FFT), enabling the application of visual backbones like inception to capture temporal patterns effectively. PatchTST is a Transformer-based model that employs a patch-based technique inspired by computer vision, achieving state-of-the-art prediction results in long-term time series forecasting. While those approaches have produced good results, this single-stock perspective can cause the model to miss broader market trends or cross-stock relationships that can provide valuable predictive insights. Researchers have begun exploring collaborative and integrative strategies to address these limitations [108]. For example, K. J. Koa *et al.* [109] introduced a model that combines deep hierarchical variational auto-encoders with diffusion probabilistic techniques to predict stock prices over multiple steps, addressing the inherent stochastic nature of stock data; Z. Pei *et al.* [110] proposed a method that leverages time series decomposition and multi-scale CNNs to predict stock prices using Open, High, Low, Close, and Trading volume data from multiple stocks; Y. Dong and Y. Hao [111] presented a DNN framework that dynamically assigns weights to multidimensional features from various stocks, capturing the impact of each feature on stock prices; W. Liu *et al.* [112] introduced a method that calculates multiple factors, including Alpha158 and OCHLVC (Open, Close, High, Low, Volume, and Change) data, to predict stock prices using a GAN [113] combined with TrellisNet [114].

In recent years, the fusion of financial domain knowledge with sophisticated computational techniques has opened new avenues for stock price prediction. The incorporation of external data sources, such as news sentiment [115], macroeconomic indicators [116], and social media trends [117], alongside advancements in model architectures, continues to push the boundaries of what is possible in financial forecasting. As markets grow increasingly complex and interconnected, methods that transcend traditional single-stock modelling and embrace holistic, cross-stock approaches represent a promising direction for future research. The essence of FL lies in merging local models to create a global model that captures the latent feature space of each local dataset without compromising the privacy or exposing the details of any individual

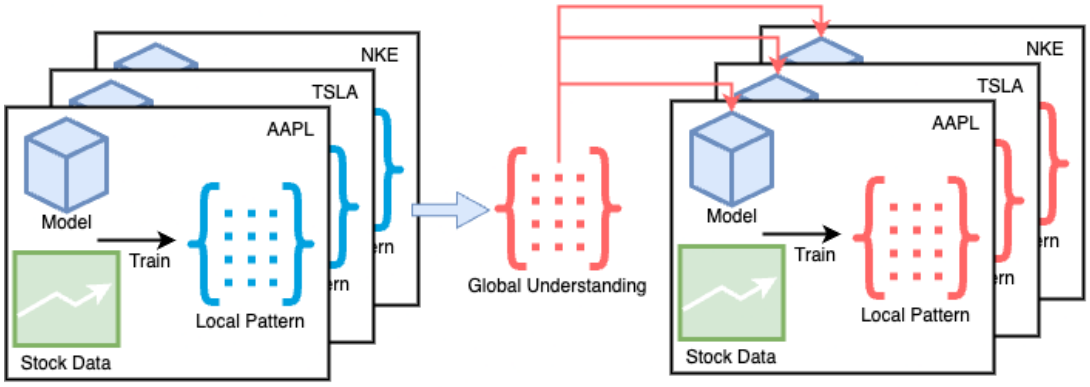


Figure 5.1: Illustration of CSTI

dataset. This concept inspired our approach, as we recognised that the trends of different stocks could be similarly merged to form a unified global understanding. Stock price movements, despite their unique characteristics, often share underlying latent patterns. By leveraging this idea, we aim to enhance prediction performance by integrating these shared patterns into a cohesive global model. In this chapter, we propose a novel training strategy designed to transform local patterns derived from individual stocks into a cohesive global understanding, thereby enhancing predictive modelling through the CSTI. Specifically, our approach involves training individual models on the historical price data of different stocks and then merging these models iteratively during training. This iterative merging process allows the global model to progressively capture shared patterns and relationships among stocks, fostering a comprehensive understanding of cross-stock price dynamics. Once the global model achieves a sufficient level of convergence, we fine-tune it on the data of each individual stock to adapt and optimise its predictions for specific stocks. As shown in Figure 5.1, this two-phase training strategy not only preserves the unique characteristics of each stock but also leverages the broader market relationships that enhance prediction accuracy. By integrating cross-stock trends into the learning process, our method addresses the limitations of traditional single-stock training and offers a robust framework for improving stock price prediction models. Our strategy inherently supports parallel training, allowing individual stock models to be trained simultaneously, which significantly improves computational efficiency and scalability. We have conducted comprehensive experiments on famous benchmark dataset, FNSPID, and evaluated our approach using multiple state-of-the-art models to validate its effectiveness. The results illustrate that our proposed training strategy significantly enhances overall performance, outperforming traditional single-stock training methods and existing benchmark models. By integrating cross-stock trends, our method not

only improves prediction accuracy but also demonstrates robustness across different market conditions and datasets.

5.2 Related Work

5.2.1 Federated Learning

Personal data protection and privacy-preserving issues have garnered significant attention from researchers [118–120]. Traditional ML approaches, which typically require centralised data for model training, are becoming increasingly challenging to implement due to stringent restrictions on data sharing. As a result, decentralised data-training approaches have emerged as a more attractive alternative, offering notable advantages in privacy preservation and data security. FL [1, 2] was introduced as a solution to these concerns, enabling individual data providers to collaboratively train a shared global model without the need for centralised data aggregation. [1] proposed a practical decentralised training method for DNNs based on averaging aggregation. Their experimental evaluations, conducted on a variety of datasets and architectures, demonstrated the robustness and effectiveness of FL in addressing privacy and security challenges.

5.2.2 Stock Trend Prediction

Traditionally, statistical approaches have been developed for time series forecasting, focusing on both the time and frequency domains. In recent years, DL methods have gained popularity in this field, owing to their ability to capture non-linear and complex relationships. These techniques leverage architectures such as RNN, LSTM, GRU and CNN to model dependencies within either the time or frequency domain. Furthermore, Transformer-based forecasting models have emerged as a powerful alternative, utilizing attention mechanisms to effectively model long-range dependencies.

FilterNet is designed to enhance time series forecasting by leveraging learnable frequency filters. It can be expressed as:

$$\mathscr{Z} = \mathscr{F}^{-1}(\mathscr{F}(\mathrm{Z}) \cdot \mathscr{H}_{filter})$$

where $\mathscr{F}$ is Fourier Transform, $\mathscr{F}^{-1}$ is inverse Fourier Transform and $\mathscr{H}_{filter}$ is learnable frequency filter. It addresses challenges such as vulnerability to high-frequency signals and inefficiencies in full-spectrum utilization, which are common in Transformer-based models.

FilterNet introduces two types of learnable filters: plain shaping filter (PaiFilter) and contextual shaping filter (TexFilter). The first one adopts a universal frequency kernel $\mathscr{H}_{Pai}$ for signal filtering and temporal modelling. $\mathscr{H}_{Pai}$ is a random initialised learnable weight. The PaiFilter processes the input time series to selectively pass or attenuate certain frequency components, effectively capturing essential temporal patterns.

$$\mathscr{Z} = \mathscr{F}^{-1}(\mathscr{F}(Z) \cdot \mathscr{H}_{Pai})$$

The second examines filtered frequencies in terms of their compatibility with input signals for dependency learning. It adapts to the specific context of the input data, allowing for more nuanced modelling of temporal dependencies.

$$\mathscr{Z} = \mathscr{F}^{-1}(\mathscr{F}(Z) \cdot \mathscr{H}_{Tex}(\mathscr{F}(Z)))$$

where $\mathscr{H}_{Tex}$ is a neural network acts as data-dependent frequency filter. FilterNet can approximate the linear and attention mappings widely adopted in time series literature. It effectively handles high-frequency noises and utilises the entire frequency spectrum beneficial for forecasting. Extensive experiments on eight time series forecasting benchmarks have demonstrated FilterNet's superior performance in both effectiveness and efficiency compared with state-of-the-art methods.

FreTS applies MLP in the frequency domain and involves transforming time-domain signals into their frequency-domain representations using the DFT:

$$\mathscr{X}(f) = \sum_{t=0}^{T-1} x(t) \cdot e^{-j2\pi ft}$$

where $x(t)$ is the time-series data, $e^{-2j\pi ft}$ is an exponential function that decomposes the signal into its frequency components. FreTS then employs redesigned MLP to learn the real ($Re(\mathscr{X})$) and imaginary ($Im(\mathscr{X})$) parts of these frequency components, capturing global dependencies and focusing on key frequency components with compact signal energy:

$$\hat{\mathscr{X}}(f) = MLP(Re(\mathscr{X})) + j \cdot MLP(Im(\mathscr{X}))$$

Operating on both inter-series and intra-series scales, FreTS effectively learns channel-wise and time-wise dependencies. Experiments on 13 real-world benchmarks, encompassing both short-term and long-term forecasting tasks, have shown that FreTS consistently outperforms many existing methods.

DLinear is a time series forecasting model designed to address the challenges of modelling both long-term trends and short-term seasonality in temporal data. It emphasises the importance of decomposing time series data into two components: trend and seasonal, and then applying linear modelling to each component separately for enhanced forecasting accuracy. The decomposition can be expressed as:

$$x(t) = x_{trend}(t) + x_{seansonal}(t)$$

where $x(t)$ is the original time series, $x_{trend}(t)$ captures the long-term trends, and $x_{seansonal}(t)$ represents the short-term periodic fluctuations. For the trend component, a simple linear model is used to approximate the general upward or downward trajectory over time. The linear trend is modelled as:

$$x_{trend}(t) = w_{trend} \cdot t + b_{trend}$$

where w_{trend} is the weight (slope) capturing the direction and magnitude of the trend, and b_{trend} is the bias term. For the seasonal component, periodic patterns are captured using a Fourier decomposition approach, which breaks the time series into sinusoidal components. The seasonal component can be modelled as:

$$x_{seansonal}(t) = \sum_{k=1}^{K} (a_k \cdot cos(\frac{2\pi kt}{T}) + b_k \cdot sin(\frac{2\pi kt}{T}))$$

where K is the number of harmonics (frequency components), T is the period of the seasonality (e.g., daily, weekly, etc.), and a_k, b_k are learnable coefficients representing the amplitude of the cosine and sine waves, respectively. This decomposition allows the model to adapt to various periodic patterns in the data. One of DLinear's key strengths is its simplicity and focus on linear relationships, which makes it computationally efficient compared to more complex DL models. By separating the linear components explicitly, DLinear avoids the over-fitting risks often associated with large neural networks while retaining the ability to capture essential temporal patterns. Additionally, the model can be trained in parallel for each component, enhancing scalability.

TimesNet is an advanced model for time series forecasting that innovatively transforms 1D time series data into 2D tensors, enabling it to capture both intra-period and inter-period variations effectively. This approach is particularly beneficial for modelling complex temporal patterns, including periodicity and irregular trends. The transformation involves reshaping the time series data $x(t)$ into a 2D tensor X based on a given period P, defined as:

$$X[i, j] = x(t + i \cdot P + j)$$

where i indexes the inter-period variations (across different cycles of the period P), and j captures intra-period variations (within a single cycle). This reshaping allows the model to treat periodic dependencies as spatial patterns, which can be efficiently processed using 2D convolution operations. The core building block of TimesNet is the TimesBlock, which employs parameter-efficient 2D convolutions to model temporal variations. For a given tensor input X , the TimesBlock applies a series of 2D convolution layers:

$$F = \sigma(Conv2D(X,W) + b)$$

where W and b are the learnable weights and biases of the convolution kernel, and σ represents an activation function such as Rectified Linear Unit (ReLU). The convolution operation captures local temporal dependencies by aggregating information across neighbouring intra-period and inter-period variations. To adaptively discover multi-periodicity in time series data, TimesNet incorporates an inception-like architecture within the TimesBlock, enabling it to process multiple kernel sizes simultaneously. This allows the model to extract features at different temporal resolutions. For instance, convolutions with larger kernels capture long-term dependencies, while smaller kernels focus on short-term variations. Additionally, TimesNet includes a periodicity discovery mechanism to identify the most relevant period P for each dataset. This mechanism involves analysing the autocorrelation of the time series and selecting the period with the highest periodicity score, ensuring that the reshaped tensor optimally represents temporal variations. The output of TimesNet is aggregated using global pooling and fed into a fully connected layer for prediction. The model is trained using a loss function, such as MSE.

PatchTST is a Transformer-based architecture specifically designed for time series forecasting. It introduces the concept of dividing time series data into non-overlapping patches, treating these patches as analogous to words in NLP tasks. This patching mechanism helps capture both local and global temporal dependencies efficiently. Let $x \in \mathbb{R}^{T \times d}$ represent a multivariate time series with T time steps and d features. PatchTST segments x into $N = \frac{T}{p}$ patches of size p, such that each patch P_i is defined as $P_i = \{x_{t_i}, x_{t_{i+1}}, \ldots, x_{t_{i+p-1}}\}$. These patches are then linearly embedded into a feature space using a learnable embedding matrix W_e:

$$Z_i = P_i \cdot W_e + b_e$$

where $W_e \in \mathbb{R}^{p \times d_z}$ maps each patch into a d_z-dimensional feature space, and b_e is a bias term. The patches are then passed through a Transformer encoder, where the self-attention mechanism

plays a pivotal role. The attention mechanism operates as:

$$\text{Attention}(Q,K,V) = \text{Softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V$$

where $Q = ZW_q$, $K = ZW_k$, and $V = ZW_v$. Here, $W_q, W_k, W_v \in \mathbb{R}^{d_z \times d_k}$ are learnable projection matrices for queries, keys, and values, respectively, and d_k is the dimensionality of the key vectors. The scaled dot-product attention computes the relationship between all patches, allowing the model to capture both short-term and long-term dependencies across the time series. To enhance the model's ability to focus on temporal patterns, positional encodings E are added to the patch embeddings Z, ensuring that the Transformer is aware of the order of patches: $\tilde{Z} = Z + E$. The Transformer encoder processes these embeddings through multiple layers of Multi-Head Self-Attention (MHSA) and Feed-Forward Network (FFN). Each encoder block can be expressed as:

$$\tilde{Z}' = \text{LayerNorm}(\tilde{Z} + \text{MHSA}(\tilde{Z}))$$
$$Z'' = \text{LayerNorm}(\tilde{Z}' + \text{FFN}(\tilde{Z}'))$$

The final output Z'' is flattened and passed through a regression head for forecasting. PatchTST's design, particularly the use of patching and Transformers, enables it to model both local temporal structures within patches and global dependencies across patches effectively.

While the aforementioned methods address various aspects of time series forecasting, including frequency-domain analysis (FilterNet, FreTS), linear modelling (DLinear), 2D variation modelling (TimesNet), and patch-based Transformer architectures (PatchTST), our approach sets itself apart by focusing on the integration of cross-stock patterns to enhance stock price prediction. To the best of my knowledge, there are no existing FL methods that target cross-stock integration for financial forecasting. Prior work [121, 122] on stock prediction largely trains single-asset models or centralised multi-asset models. FL papers [123–125] in finance mainly focus on privacy-preserving training across institutions without modelling cross-asset dependencies. The work in [123] proposes a FL framework for multiple traditional ML algorithms, expanding the utility of FL from DL to traditional ML. [124] proposed a new method, named Federated Learning Enhanced Multi-Layer Perceptron Long Short-Term Memory (Fed-MLP-LSTM). They combined MLP with LSTM, which the former part acts as feature extraction and the later is for sequence modelling. The authors of the work [125] designed F-LSTM to tackle the dispersion problem that usually occurs in data sources like cryptocurrency. Drawing inspiration from FL principles, our method merges local models trained on individual stocks into

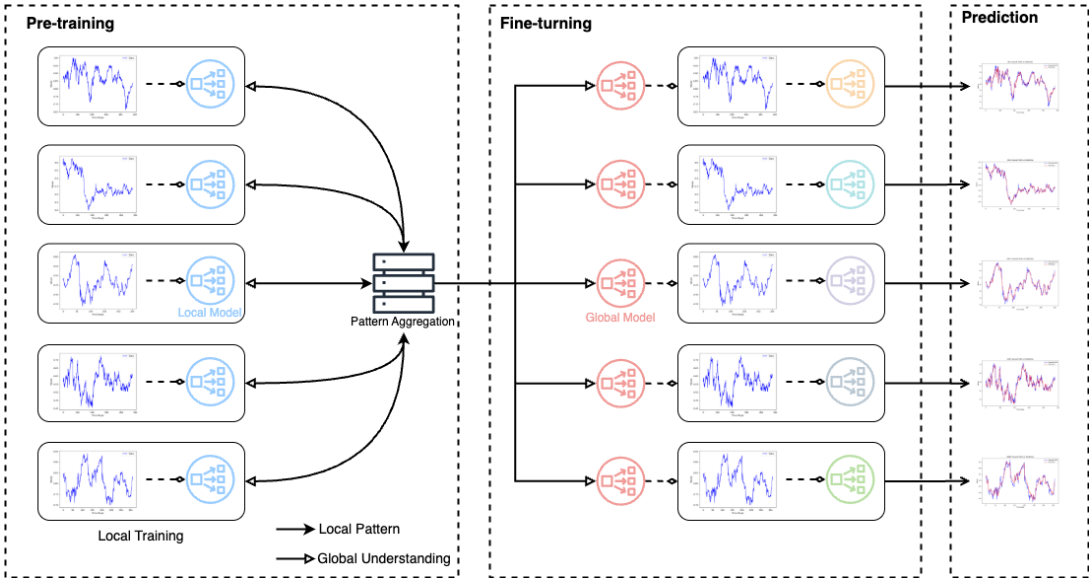


Figure 5.2: The overview architecture of our proposed CSTI.

a unified global model, effectively capturing inter-dependencies among different stocks without the need for centralised data aggregation. This approach enables the model to uncover shared patterns across stocks, potentially improving predictive performance in stock price forecasting. Moreover, our proposed strategy is highly versatile and can be seamlessly deployed on top of all the aforementioned models. By doing so, our method complements and enhances existing approaches, enabling them to learn cross-stock patterns and overcome their current limitations. This synergy has the potential to unlock even greater performance in stock price prediction tasks.

5.3 Methodology

In this chapter, we propose a novel training strategy, CSTI, which enhances stock price prediction by leveraging shared patterns and interdependencies among stocks. As shown in Figure 5.2, unlike traditional single-stock forecasting methods that rely solely on individual stock data, CSTI extracts local patterns from each stock and integrates them into a unified global understanding, enabling a more holistic representation of market dynamics. The proposed framework consists of three key phases: individual training, global model merging, and fine-tuning, each contributing to the development of a robust, generalisable model for stock prediction. A pseudocode is provided in Algorithm 4.

Algorithm 4: CSTI Framework

Input: Stock datasets $\{X_k, y_k\}_{k=1}^{K}$, number of stocks K, training epochs E, global learning rate η_g, adaptation coefficient α, regularization weight λ

Output: Fine-tuned models $\{f_k(\cdot;\theta_k')\}_{k=1}^{K}$ for each stock

Phase 1: Individual Training
forall *stock* S_k, *where* $k = 1$ *to* K ***in parallel*** **do**
  Initialize model $f_k(\cdot;\theta_k)$
  Update θ_k by minimizing:
  $\mathscr{L}_k = \frac{1}{T}\sum_{t=1}^{T}(f_k(X_{k,t};\theta_k) - y_{k,t})^2$
end
Phase 2: Global Model Merging
forall *stock* S_k **do**
  Compute weight w_k
end
Aggregate global model: $\theta_g = \frac{1}{K}\sum_{k=1}^{K} w_k\theta_k$
for $e = 1$ *to* E **do**
  Refine θ_g by repeating Phase 1 & 2.
end
Phase 3: Fine-Tuning for Stock-Specific Adaptation
forall *stock* S_k ***in parallel*** **do**
  Initialize $\theta_k \leftarrow \theta_g$
  for $e = 1$ *to* E **do**
    Update θ_k by minimizing:
    $\mathscr{L}_k' = \frac{1}{T}\sum_{t=1}^{T}(f_k(X_{k,t};\theta_k) - y_{k,t})^2 + \lambda\|\theta_k - \theta_g\|^2$
  end
  Save θ_k'
end
return Fine-tuned models $\{f_k(\cdot;\theta_k')\}_{k=1}^{K}$

5.3.1 Individual Training

The first phase of the CSTI framework focuses on training individual models for each stock using historical stock price data. Each stock, denoted as S_k, is represented by its time-series dataset $X_k \in \mathbb{R}^{T\times d}$, where T represents the length of the time horizon and d denotes the number of extracted features for each time step. The goal of this phase is to enable each stock-specific model to learn its own price patterns, volatilities, and dependencies on various market indicators. To train a local model $f_k(\cdot;\theta_k)$ for stock S_k, we define an objective function that optimises the model parameters θ_k to minimise the discrepancy between the predicted stock prices and their

actual observed values. The training process is based on the following loss function:

$$\mathscr{L}_k = \frac{1}{T}\sum_{t=1}^{T} \ell(f_k(X_{k,t};\theta_k), y_{k,t}), \tag{5.1}$$

where ℓ represents the stock price prediction loss function, $X_{k,t}$ denotes the feature set at time t, and $y_{k,t}$ represents the actual stock price at time t. The loss function is typically designed as the MSE to penalise large deviations in predictions:

$$\ell(f_k(X_{k,t};\theta_k), y_{k,t}) = (f_k(X_{k,t};\theta_k) - y_{k,t})^2. \tag{5.2}$$

To ensure robustness, each local model is trained using a combination of historical market indicators, technical indicators, and external financial variables. The feature set $X_{k,t}$ may include open price, close price, trading volume, moving averages, volatility measures, and macroeconomic indicators. Feature engineering techniques, such as time-series normalisation, log transformation, and seasonal adjustment, are applied to preprocess the data, ensuring stability in learning patterns. A key advantage of this phase is that training is conducted in parallel for all stocks, allowing for efficient scalability. Given a set of K stocks, their respective models $\{f_k\}_{k=1}^{K}$ are trained simultaneously on independent computing units, significantly reducing the computational overhead:

$$\mathscr{L} = \sum_{k=1}^{K} \mathscr{L}_k. \tag{5.3}$$

Parallel training enables the framework to handle a vast number of stocks without compromising computational efficiency. It also allows for the incorporation of multiple asset types, including equities, commodities, and exchange-traded funds, making the system flexible for large-scale financial modelling. Each local model learns stock-specific dependencies, capturing intrinsic price movement trends, cyclical patterns, and market anomalies. By training independently, the models are optimised to their respective stock behaviours, avoiding cross-stock contamination during initial training. This ensures that local training preserves fine-grained stock-specific insights before transitioning to the global model merging phase, where broader inter-stock relationships are integrated.

5.3.2 Global Model Merging

In the second phase, the locally trained models are iteratively merged to create a global model $f_g(\cdot;\theta_g)$. This merging process integrates the parameter spaces $\{\theta_k\}_{k=1}^{K}$ of individual stock

models, forming a generalised representation of cross-stock trends. Unlike single-stock models that learn in isolation, this global model captures inter-stock dependencies, enabling a more robust and holistic understanding of market-wide movements. To achieve this, the global model parameters θ_g are computed using a weighted aggregation of the locally trained models:

$$\theta_g = \frac{1}{K} \sum_{k=1}^{K} w_k \theta_k, \tag{5.4}$$

where w_k is a weighting factor assigned to stock S_k based on predefined metrics such as stock volatility, market capitalization, or historical prediction accuracy. This weighted aggregation ensures that more reliable and stable stocks contribute more significantly to the global model, while highly volatile or less predictive stocks have a reduced impact. The adaptive weighting mechanism prevents dominant stocks from disproportionately influencing the model and ensures that under-represented stocks retain relevance. The integration of local models into a global model enables the detection of market-wide patterns that extend beyond individual stock movements. By aggregating trends from multiple stocks, the global model learns shared patterns such as correlated price fluctuations, responses to macroeconomic indicators, and sectoral dependencies. These cross-stock relationships improve predictive robustness, particularly in highly dynamic market conditions where isolated stock models may struggle to generalise.

One of the challenges in global model merging is ensuring that the aggregation process does not overly smooth individual stock characteristics, which may lead to loss of stock-specific signals. To mitigate this, we employ an iterative refinement process where the merged global model is evaluated after each aggregation step to monitor performance convergence. The merging continues until the global model stabilises, ensuring that it retains both cross-stock insights and essential stock-specific variations. In our experiments, we initially applied equal weights to all individual models, assuming uniform importance across stocks. This baseline approach demonstrated stable performance; however, further refinements can be introduced by dynamically adjusting w_k using market-specific criteria. For instance, a volatility-aware weighting scheme assigns lower weights to highly volatile stocks to prevent extreme fluctuations from distorting the global model:

$$w_k = \frac{1}{\sigma_k + \varepsilon}, \tag{5.5}$$

where σ_k represents the standard deviation of stock S_k over the training period, and ε is a small constant to prevent division by zero. Another approach is accuracy-based weighting, where

stocks with historically higher predictive accuracy are given greater weight in the aggregation:

$$w_k = \frac{A_k}{\sum_{i=1}^{K} A_i}, \tag{5.6}$$

where A_k is the past performance metric of stock S_k, such as its MSE or Mean Absolute Percentage Error (MAPE) in prior predictions. Beyond improving model accuracy, the global model merging process enhances the stability of stock price predictions. Since individual stocks often experience periods of high volatility, sectoral shocks, or external economic impacts, a well-merged global model leverages the diversification effect to mitigate the impact of such anomalies. The iterative refinement ensures that the model does not become overly biased toward a subset of stocks while maintaining an accurate and generalised understanding of financial market behaviour. To further optimise the merging process, future work could explore alternative aggregation strategies, such as clustering-based grouping of stocks before aggregation. By grouping stocks with similar behavioural patterns, aggregation could be performed at an intermediate level before constructing the final global model, preserving sector-specific trends while still capturing market-wide insights. By integrating insights across multiple stocks and refining the aggregation process through adaptive weighting, the global model effectively enhances predictive power compared to traditional single-stock forecasting. The resulting model not only improves overall accuracy but also adapts dynamically to evolving financial conditions, making it a robust framework for large-scale financial forecasting applications.

5.3.3 Fine-Tuning for Stock-Specific Adaptation

Once the global model achieves sufficient convergence, the final phase involves fine-tuning the global model on individual stock data to ensure that stock-specific characteristics are preserved while leveraging the broader knowledge embedded in the global model. Although the global model provides a generalised understanding of market-wide trends, each stock exhibits unique behavioural patterns that may not be fully captured in a purely aggregated model. Fine-tuning addresses this limitation by reintroducing stock-specific learning while maintaining the benefits of cross-stock knowledge integration. The fine-tuning process begins by re-initialising the parameters of each local model θ_k using the global model parameters θ_g. This re-initialisation allows the local models to start from an optimised baseline rather than learning from scratch, significantly improving training efficiency and accelerating convergence. The stock-specific adaptation is then performed through further optimisation on the individual stock dataset,

refining the parameters to minimise the stock-level prediction loss:

$$\theta_k' = \arg\min_{\theta_k} \frac{1}{T} \sum_{t=1}^{T} \ell(f_k(X_{k,t}; \theta_k), y_{k,t}). \tag{5.7}$$

where ℓ represents the loss function, $X_{k,t}$ is the feature vector for stock S_k at time step t, and $y_{k,t}$ is the corresponding ground truth stock price. The fine-tuning process ensures that the model maintains the insights gained from the global model while adapting to the idiosyncratic features of each stock, preventing excessive bias toward overall market behaviour.

A critical challenge in this phase is achieving the optimal balance between global generalisation and stock-specific adaptation. If the fine-tuning step is too aggressive, the model may over-fit to individual stock data, losing the generalisation benefits gained from the global model. Conversely, if the fine-tuning is too minimal, the model may fail to capture the distinct price movement patterns of each stock. To address this, we introduce an adaptive learning rate strategy that controls the extent of fine-tuning. Specifically, we initialise the learning rate η_k for each stock S_k using a scaled version of the global learning rate η_g:

$$\eta_k = \alpha \cdot \eta_g, \tag{5.8}$$

where α is a tuning coefficient that determines the adaptation strength. A lower α ensures a smoother transition from the global model, preserving shared market trends, whereas a higher α allows greater flexibility for local adaptation. Another important aspect of fine-tuning is the regularization of model parameters to prevent divergence from the global model while still allowing necessary stock-specific refinements. We incorporate an L_2 regularization term that penalises excessive deviation from θ_g:

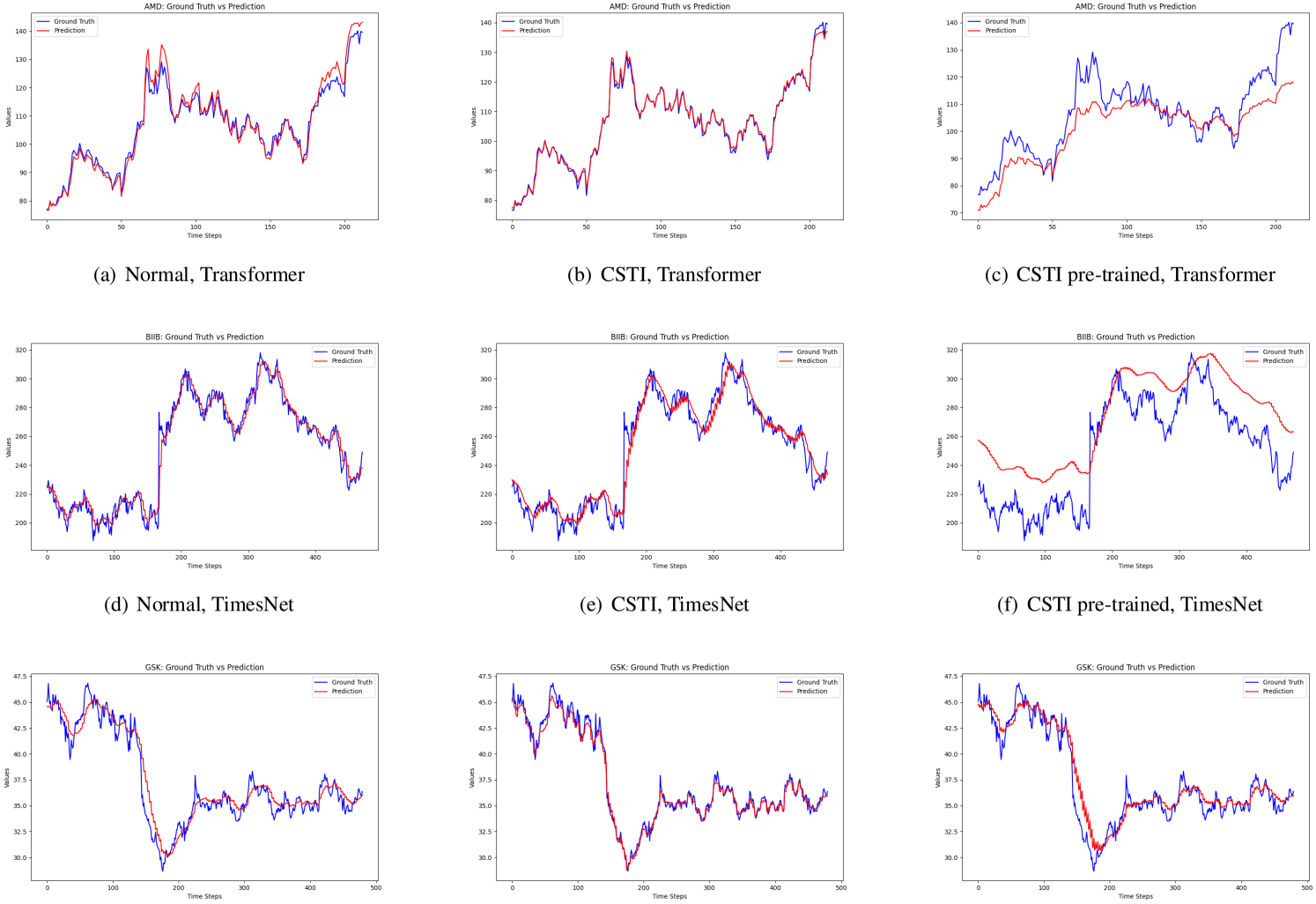
$$\mathscr{L}_k' = \frac{1}{T} \sum_{t=1}^{T} \ell(f_k(X_{k,t}; \theta_k), y_{k,t}) + \lambda \|\theta_k - \theta_g\|^2, \tag{5.9}$$

where λ is a regularization coefficient that controls the trade-off between maintaining the global model structure and adapting to stock-specific features. This constraint ensures that while fine-tuning personalises the model for individual stocks, the learned parameters remain aligned with the global market representation, thus avoiding drastic deviations that could lead to over-fitting. Beyond parameter fine-tuning, feature-level adaptation can be incorporated to enhance stock-specific learning. Fine-tuning is crucial for ensuring that each stock model benefits from the cross-stock knowledge integrated in the global model while preserving the unique dynamics that define its price behaviour. The resulting fine-tuned models exhibit improved accuracy in

predicting stock movements compared to models trained in isolation, as they incorporate both market-wide trends and individual price variations. This final adaptation phase plays a pivotal role in making the CSTI framework effective for real-world financial forecasting, balancing generalisation and stock-specific refinement for optimal predictive performance.

5.3.4 Advantages of the CSTI Framework

The CSTI framework improves on traditional single-stock training by combining local learning with global integration of cross-stock information. Its core novelty lies in the global merging and adaptive weighting phase, which enables the model to capture dependencies across assets that standard FL averaging overlooks. Parallel training of individual stock models ensures scalability, allowing efficient utilisation of computational resources across many stocks. During the merging phase, local models are combined into a global representation, where adaptive weighting gives greater influence to stocks whose updates are more representative of shared market behaviour. This mechanism allows the global model to learn sector-level correlations and macroeconomic patterns, improving generalisation across diverse market conditions. The fine-tuning stage then restores local specificity, ensuring that while benefiting from global insights, each stock model retains sensitivity to its own unique price dynamics. Convergence checks are incorporated to guarantee stability during the integration process, preventing noisy or underperforming stock updates from destabilising the global model. In this way, CSTI goes beyond conventional FL, which focuses primarily on decentralisation and privacy, by directly enhancing predictive performance. By adaptively merging and refining models, CSTI preserves stock-specific features while exploiting cross-stock trends, leading to more accurate and robust forecasts for large-scale financial applications.



(a) Normal, Transformer (b) CSTI, Transformer (c) CSTI pre-trained, Transformer

(d) Normal, TimesNet (e) CSTI, TimesNet (f) CSTI pre-trained, TimesNet

(g) Normal, DLinear (h) CSTI, DLinear (i) CSTI pre-trained, DLinear

Figure 5.3: The visualization of the predicted regression lines for models trained on 50 stocks without sentiment information is presented. The first column shows the results from the normal training strategy, the second column presents the results from our proposed methods, and the third column displays the results from pre-trained global models.

5.4 Experiments

5.4.1 Settings and Dataset

We utilised the PyTorch framework [58] to implement all neural network models. For reproducibility, the source code is publicly available and can be accessed at the following link[1]. Local training was conducted using various optimisation algorithms, all of which employed a consistent learning rate of 0.01 and a momentum of 0.9. The local batch size was set to 64, and the number of training epochs was fixed at 100. For our proposed strategy, we trained local models for 1 epoch before integrating them to generate a global model during the first 50 epochs. Following this, we fine-tuned the global model for the remaining 50 epochs. This approach ensures that both the normal training process and the CSTI framework have the same total number of model weight update iterations. To evaluate the performance of our method across different stock group sizes, we randomly selected three groups of sub-datasets, containing 5, 25, and 50 stocks, respectively. These experiments were conducted to assess the scalability and performance of our method with varying numbers of stocks. The backbone models used in our experiments include Transformer, TimesNet, DLinear, FreTS, PaiFilter, TexFilter, and PatchTST.

In our study, we conducted experiments using FNSPID, which is a comprehensive dataset designed to address key limitations in financial market analysis by integrating numerical stock data with sentiment-laden financial news. This dataset spans the years 1999 to 2023, encompassing over 29.7 million stock price entries and 15.7 million time-aligned financial news records for 4775 companies within the *S&P*500 index. The dataset uniquely combines quantitative and qualitative data to enhance predictive modelling capabilities, particularly in the realm of stock price forecasting. FNSPID was built through a rigorous process combining data scraping, cleaning, and integration. Stock price data was sourced from Yahoo Finance APIs, while financial news was collected via web scraping from platforms like NASDAQ. Ethical considerations, including adherence to copyright laws and avoidance of premium content, were prioritised throughout data collection. Sentiment scores were normalised and missing data gaps were handled through exponential decay methods to maintain temporal consistency. It sets a new standard for financial datasets, offering a rich resource for advancing research in financial market analysis, sentiment modelling, and stock price prediction. In out experiments, we used two sets of data features. The first set includes Open and Close prices and the second one includes Open, Close and Sentiment.

[1] `https://github.com/Rand2AI/CSTI`

Table 5.1: Experiment results using different training strategies over seven models on FNSPID dataset with or without sentiment information. The results highlighted in red are the better ones. Trans. means Transformer; TN represents TimesNet; DL is DLinear; FTS refers to FreTS; PF is PaiFilter; TF represents TexFilter and PTST meas PatchTST.

#	Model	Normal						Ours					
		w/ sentiment			w/o sentiment			w/ sentiment			w/o sentiment		
		MAE	MSE	R^2	MAE	MSE	R^2	MAE	MSE	R^2	MAE	MSE	R^2
5	Trans.	0.0702	0.0079	0.3067	0.0472	0.0045	0.5829	0.0535	0.005	0.5243	0.0406	0.0031	0.6126
	TN	0.1158	0.0368	0.444	0.0459	0.0034	0.5368	0.0702	0.0095	0.4669	0.0427	0.0029	0.6166
	DL	0.0424	0.0029	0.6014	0.0359	0.0021	0.7211	0.0216	0.0008	0.8772	0.0278	0.0013	0.825
	FTS	0.0361	0.0022	0.6864	0.0308	0.0016	0.7579	0.0323	0.0017	0.7499	0.0311	0.0016	0.7625
	PF	0.0489	0.0037	0.5051	0.0489	0.0037	0.5062	0.0484	0.0037	0.511	0.0388	0.0026	0.6026
	TF	0.0424	0.0029	0.622	0.0407	0.0026	0.653	0.0258	0.0014	0.7588	0.023	0.0009	0.8706
	PTST	0.0206	0.0008	0.8467	0.027	0.0012	0.8299	0.0235	0.001	0.8228	0.0262	0.0012	0.8371
25	Trans.	0.0381	0.0036	0.5946	0.0489	0.0047	0.5298	0.0373	0.0031	0.6585	0.0583	0.0066	0.4212
	TN	0.0341	0.0021	0.6664	0.0333	0.0021	0.7023	0.0358	0.0021	0.6388	0.0318	0.0017	0.7005
	DL	0.0229	0.001	0.7776	0.0259	0.0011	0.7605	0.0179	0.0007	0.9225	0.0223	0.0009	0.8344
	FTS	0.0195	0.0007	0.868	0.0255	0.0011	0.7858	0.0271	0.0013	0.7793	0.0253	0.0011	0.8029
	PF	0.0433	0.0031	0.4405	0.0433	0.0031	0.4398	0.0143	0.0005	0.9125	0.0165	0.0005	0.9096
	TF	0.0218	0.0009	0.7964	0.0289	0.0014	0.7123	0.0122	0.0003	0.9412	0.0167	0.0005	0.8954
	PTST	0.0143	0.0004	0.9198	0.023	0.0009	0.8382	0.0168	0.0005	0.9045	0.0254	0.0011	0.8007
50	Trans.	0.027	0.0017	0.8162	0.0536	0.0066	0.5834	0.0404	0.0045	0.653	0.0474	0.0051	0.5432
	TN	0.0283	0.0015	0.7673	0.0224	0.001	0.8449	0.0303	0.0016	0.7552	0.037	0.0025	0.6613
	DL	0.0168	0.0005	0.909	0.0237	0.001	0.8368	0.0154	0.0005	0.936	0.0234	0.001	0.8437
	FTS	0.0266	0.0031	0.8783	0.0276	0.0014	0.7959	0.0249	0.0011	0.8143	0.0273	0.0013	0.8
	PF	0.041	0.0028	0.5432	0.0411	0.0028	0.5425	0.0375	0.0024	0.6176	0.0313	0.0018	0.7232
	TF	0.0148	0.0005	0.9223	0.0233	0.001	0.8362	0.0123	0.0003	0.9498	0.017	0.0005	0.9112
	PTST	0.0149	0.0004	0.9325	0.0247	0.0011	0.8351	0.0164	0.0005	0.9186	0.0254	0.0011	0.8219

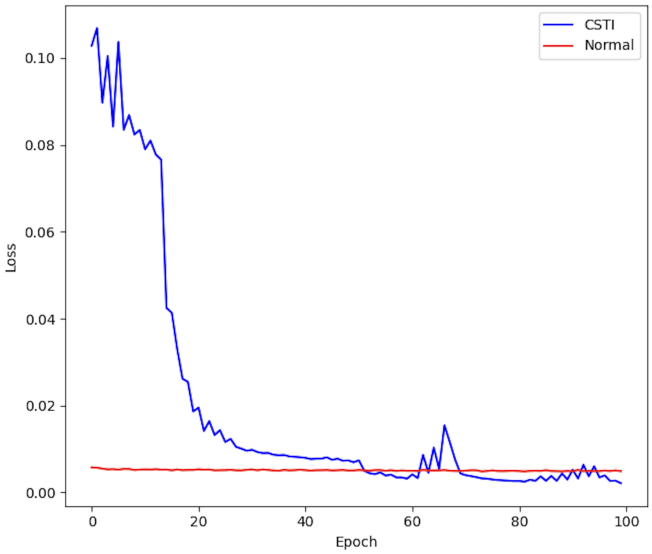
5.4.2 Results Analysis

A set of comprehensive experiments were conducted to evaluate the performance of our proposed method compared to the normal training strategy. Following the work in [115], the normal training strategy involves training a model on one stock's historical price data, then fine-tuning it on the next stock's data, iteratively, until all the stock data has been utilised. Table 5.1 presents the experimental results for seven models on the FNSPID dataset, with and without sentiment information. The results highlighted in red indicate the superior ones. Overall, our proposed training strategy, CSTI, outperforms the normal training strategy in most experiments. For example, with the PaiFilter model, CSTI achieved R^2 values of 0.9125 and 0.9096 on data with and without sentiment information, respectively, using 25 stocks. In contrast, the normal training strategy only achieved R^2 values of 0.4405 and 0.4398 under the same conditions. The substantial gap can be attributed to PaiFilter's reliance on capturing broad temporal patterns: the cross-stock integration stage in CSTI enriches its representation with sector-level correlations and shared market signals, which are otherwise missing in isolated single-stock training. However, our method does not always yield better results. In the case of TimesNet and PatchTST, the normal training strategy consistently outperformed CSTI across experiments using 25 and 50 stocks. This behaviour can be explained by the architectural design of these models. Both TimesNet and PatchTST employ strong sequence modelling and tokenisation mechanisms that already capture global context internally. When additional cross-stock aggregation is introduced through CSTI, the effect may become redundant or even introduce noise from less correlated stocks, leading to reduced predictive accuracy. Another clear trend is that models trained on data with sentiment information consistently outperformed those trained without sentiment features. This supports the claim made in [115] that textual and sentiment-based signals provide complementary information to historical price series. Importantly, the benefit of sentiment data interacts with cross-stock aggregation, when combined with CSTI, sentiment features help amplify shared market signals such as investor mood or sector-wide reactions, further improving performance. Taken together, these observations show that CSTI is most beneficial for architectures like PaiFilter that depend on broad trend information and are enhanced by additional cross-stock correlations. For models such as TimesNet or PatchTST, which already encode rich global dependencies, CSTI offers less advantage and may require more selective aggregation strategies to avoid incorporating noise.

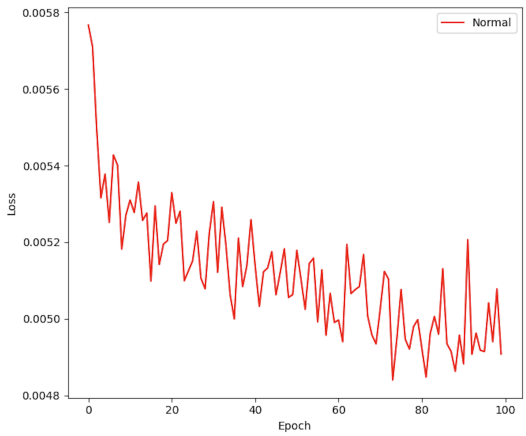
In Figure 5.3, we present visualisations of predicted regression lines from the normal training strategy, our proposed CSTI method, and CSTI pre-trained global models, trained on 50 stocks

without sentiment information. The first column displays results from the normal training strategy, the second column shows predictions from CSTI, and the third column presents outputs from the global model before fine-tuning. For the Transformer model on AMD stock (Figures (a)-(c)), both the normal training strategy and CSTI follow the ground truth closely. However, the pre-trained global model diverges in local details while retaining the overall up-and-down trend. This shows how CSTI's cooperative learning captures broad market patterns during pre-training, which fine-tuning then sharpens into stock-specific accuracy. For the TimesNet model on BIIB stock (Figures (d)-(f)), the normal training strategy aligns more closely with short-term fluctuations than CSTI. This illustrates a limitation of CSTI, that when a model architecture already encodes global dependencies internally, additional cross-stock aggregation may introduce noise that obscures fine-grained local signals. The pre-trained global model captures the main trajectory but lacks the refinement of the fine-tuned CSTI model, underscoring the complementary roles of global integration and local adaptation. For the DLinear model on GSK stock (Figures (g)-(i)), CSTI predictions reduce noise and align more closely with the ground truth than the normal training strategy. Here, CSTI improves performance by filtering out unstable local updates through its global merging and weighting stage, highlighting that simpler linear models benefit strongly from cross-stock integration. The pre-trained global model again captures overall directionality but misses local variations, showing that fine-tuning is necessary to recover stock-specific latent dynamics. Overall, these visual comparisons demonstrate how CSTI changes prediction behaviour. Pre-training on cross-stock information provides a strong global prior, while fine-tuning restores local precision. Depending on the model architecture, CSTI either enhances trend capture (e.g., Transformer, DLinear) or interacts with internal global mechanisms in ways that may reduce sensitivity to short-term fluctuations (e.g., TimesNet). This confirms that CSTI's methodological novelty and directly shapes predictive behaviour, rather than merely shifting performance metrics.

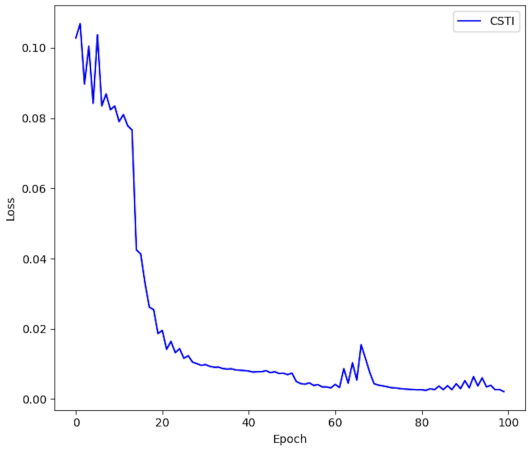
Figure 5.4 illustrates the training loss over 100 epochs for two Transformer models: one trained using the normal training strategy (red line) and the other trained with our proposed CSTI strategy (blue line). Both experiments were conducted using 50 stocks without sentiment information as the dataset. The normal training strategy exhibits a steady and smooth decline in training loss from the very first epoch. This is expected, as the Transformer model quickly adapts to the stock price data, which is relatively straightforward to regress. As a result, the initial loss is already quite low, and it decreases further in a gradual and consistent manner, stabilising towards the later epochs. In contrast, the CSTI approach shows a training loss pattern



(a) Both



(b) Normal



(c) CSTI

Figure 5.4: Training Loss Comparison

more typical of general DL models, with a sharper decline in the initial epochs followed by minor oscillations as training progresses. This behaviour arises due to the iterative merging of 50 local models into a unified global model at each epoch, which introduces variability and occasional fluctuations in the loss curve. The transient spikes observed in the CSTI loss curve indicate the periodic model merging, which slightly perturbs convergence but ultimately allows for broader market generalisation. Despite this, the CSTI strategy achieves a comparable final training loss as the normal training method by the end of the 100 epochs, demonstrating

its effectiveness in converging to an optimal solution. The figure highlights the differences in loss dynamics between the two strategies, with CSTI achieving similar performance while incorporating cross-stock interdependencies to enhance market-wide learning.

5.5 Conclusion

In conclusion, this chapter highlights the transformative potential of integrating cross-stock patterns into stock price prediction models using the proposed CSTI framework. Traditional methods, while effective, often neglect the inter-dependencies among stocks, leading to sub-optimal utilization of available market information and limiting the ability to capture broader financial trends. By leveraging a novel training strategy that iteratively merges local stock models into a unified global model, CSTI demonstrates its ability to enhance prediction accuracy and generalisation by learning shared market-wide representations. Extensive experiments conducted on the FNSPID dataset validate the effectiveness of the method, showcasing superior performance across various models and settings, particularly in capturing complex temporal dependencies and mitigating over-fitting. The proposed approach not only addresses the limitations of single-stock training but also integrates complementary benefits of pre-training and fine-tuning to capture both global trends and stock-specific nuances, ensuring a balanced trade-off between market-wide insights and individual stock characteristics. Furthermore, the results underscore CSTI's scalability, efficiency, and robustness, making it a viable solution for large-scale financial forecasting applications. This work paves the way for future advancements in federated financial modelling, opening new avenues for enhancing predictive performance while preserving privacy and computational efficiency in decentralised financial systems.

Chapter 6

Conclusions and Future Work

In this chapter, we summarise the key contributions of this thesis and discuss potential directions for future research.

6.1 Conclusions

This thesis advances the field of FL by addressing three fundamental challenges, which are efficient aggregation, gradient protection, and real-world applicability. By demonstrating how these aspects collectively move FL closer to practical deployment.

- First, the proposed **EWWA-FL** establishes a new approach to adaptive aggregation by weighting parameters individually rather than assigning a single scalar per client. This fine-grained mechanism directly addresses the convergence instability and bias introduced by non-IID data, one of the most persistent challenges in FL. Experimental results across multiple benchmarks showed faster convergence, reduced variance, and superior accuracy compared to FedAvg, FedOpt, and FedCAMS, confirming that parameter-level adaptation yields more stable and generalisable global models. The significance of this finding extends beyond the specific algorithm, which demonstrates that modelling heterogeneity at the granularity of parameters can fundamentally improve communication efficiency and learning robustness in decentralised systems.

- Second, the **AdaDefence** framework contributes a practical and theoretically motivated solution to privacy leakage in FL. By substituting raw gradients with Adam-based stand-in, it prevents adversaries from reconstructing sensitive training data while maintaining model

performance. The results in Chapter 4 evidenced substantial increases in reconstruction error (MSE) and decreases in image fidelity (PSNR, SSIM), confirming that AdaDefence effectively obfuscates information without the high computational cost of encryption or the accuracy loss caused by like DP or HE. This contribution demonstrates that lightweight, optimiser-driven obfuscation can achieve a favourable privacy–utility trade-off, providing a scalable defence for real-world FL applications.

- Finally, **CSTI** illustrates the applicability of FL in the financial domain, showcasing how aggregation and privacy techniques can be integrated to meet regulatory and operational constraints. By merging stock-specific models into a global representation and fine-tuning them locally, CSTI captures cross-stock dependencies that traditional single-stock models overlook. The improved R^2 scores across diverse financial architectures validate that cross-institutional collaboration can enhance predictive accuracy without compromising data confidentiality. More broadly, CSTI serves as a blueprint for deploying FL in other sensitive sectors where both performance and privacy are critical.

In combination, these advances demonstrate a coherent progression from theoretical innovation to practical application. Together they improve the efficiency, security, and utility of FL, showing that fine-grained aggregation, optimiser-based gradient protection, and cross-domain adaptation can jointly support the development of trustworthy and scalable federated systems.

6.2 Future Work

While this thesis advances FL across aggregation, privacy, and application, several open challenges remain. We outline three key directions for future exploration.

Gradient Leakage in Natural Language Processing:

Most gradient leakage studies focus on computer vision tasks, yet textual models in FL, including recurrent and transformer-based architectures, are equally vulnerable. Gradients in language models may encode semantic or syntactic cues, enabling partial reconstruction of private text. Future work should systematically test these risks by simulating inversion attacks on federated NLP tasks such as sentiment analysis and document classification. Extending AdaDefence to sequential data, for example by adapting stand-in to attention mechanisms, will be crucial to ensure privacy in language applications.

Lightweight Adaptive Aggregation:

Although EWWA-FL improves convergence under non-IID conditions, its per-parameter weighting introduces extra computation and communication overhead. A promising next step is to embed adaptive optimisation principles directly into the aggregation rule, using moment-based estimates, as in Adam, to approximate parameter stability without explicit variance computation. This approach could reduce complexity while retaining fine-grained adaptivity, enabling scalable deployment of adaptive aggregation in bandwidth-limited or heterogeneous settings.

Partial Gradient Leakage Analysis:

Initial evidence suggests that even incomplete gradient information may expose sensitive data. This raises an important question, that can full gradients be inferred from partial updates through structural correlations between layers? Future studies should model this as a reconstruction problem, estimating missing gradients using learned mappings between partial and complete updates. Understanding this relationship will clarify how much information partial gradients carry and guide the design of defences robust to partial disclosure.

Together, these directions aim to extend the robustness, scalability, and privacy guarantees of FL beyond the current scope of this thesis, moving toward more secure and efficient real-world deployments.

6.3 The End

The increasing reliance on large-scale AI models, particularly in the context of the AIGC era, has raised important ethical concerns surrounding monopolisation, algorithmic transparency, and data privacy. Centralised models, typically controlled by a handful of dominant technology companies, often operate as black boxes trained on proprietary data, limiting public oversight and fostering asymmetric power dynamics in the deployment and governance of AI. This concentration of control not only stifles competition and innovation from smaller entities but also exacerbates issues of bias, exclusion, and lack of accountability in decision-making systems. The inability of users and developers to influence or inspect such systems highlights the urgent need for more democratic and privacy-preserving approaches to AI development. FL presents a promising decentralised alternative that addresses these concerns by allowing collaborative training across multiple data custodians without requiring raw data sharing. In this framework,

sensitive information remains on local devices, and only model updates are exchanged and aggregated. This design inherently respects data sovereignty and regulatory requirements while enabling the collective training of powerful models. Furthermore, by removing the need for centralised data ownership, FL empowers diverse stakeholders, ranging from small enterprises to individual users, to participate in the advancement of AI, thereby fostering inclusivity and reducing dependency on monopolistic actors. FL also enhances transparency and accountability by encouraging the development of open protocols and modular design practices, where individual components of the learning pipeline can be independently audited, adjusted, and improved. These characteristics make it a strong candidate for establishing a more equitable AI ecosystem, especially in critical domains such as healthcare, finance, and public infrastructure, where trust, fairness, and security are paramount. The contributions of this thesis lay the groundwork for making FL more scalable, secure, and practically deployable. Specifically, the proposed methods address key challenges in model aggregation under non-IID conditions, robust privacy defence against gradient leakage, and cross-institutional collaboration in financial forecasting. These innovations collectively enhance the applicability and resilience of FL systems. However, realising the full potential of FL in the privacy-aware AI era will require ongoing efforts to improve aggregation efficiency, reduce communication overhead, strengthen defences against inference attacks, and tailor FL strategies to domain-specific needs. Future research should continue to explore these directions, ensuring that FL evolves as a core enabler of ethical, secure, and decentralised AI technologies.

Bibliography

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[2] J. Konecný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *ArXiv*, vol. abs/1610.05492, 2016.

[3] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[4] B. McMahan and D. Ramage, "Federated learning: Collaborative machine learning without centralized training data," *Google Research Blog*, vol. 3, 2017.

[5] L. Li, Y. Fan, M. Tse, and K.-Y. Lin, "A review of applications in federated learning," *Computers & Industrial Engineering*, vol. 149, p. 106854, 2020.

[6] J. Wen, Z. Zhang, Y. Lan, Z. Cui, J. Cai, and W. Zhang, "A survey on federated learning: challenges and applications," *International Journal of Machine Learning and Cybernetics*, vol. 14, no. 2, pp. 513–535, 2023.

[7] H. Wu and P. Wang, "Fast-convergent federated learning with adaptive weighting," *IEEE Transactions on Cognitive Communications and Networking*, vol. 7, no. 4, pp. 1078–1088, 2021.

[8] X. Ouyang, Z. Xie, J. Zhou, G. Xing, and J. Huang, "Clusterfl: A clustering-based federated learning system for human activity recognition," *ACM Transactions on Sensor Networks*, vol. 19, no. 1, pp. 1–32, 2022.

[9] N. Kotelevskii, M. Vono, A. Durmus, and E. Moulines, "Fedpop: A bayesian approach for personalised federated learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 8687–8701, 2022.

[10] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," *Advances in Neural Information Processing Systems*, vol. 32, pp. 14 774–14 784, 2019.

[11] B. Zhao, K. R. Mopuri, and H. Bilen, "idlg: Improved deep leakage from gradients," *arXiv preprint arXiv:2001.02610*, 2020.

[12] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients-how easy is it to break privacy in federated learning?" *Advances in neural information processing systems*, vol. 33, pp. 16 937–16 947, 2020.

[13] H. Ren, J. Deng, and X. Xie, "Grnn: Generative regression neural network–a data leakage attack for federated learning," *ACM Transactions on Intelligent Systems and Technology*, 2021.

[14] Z. Li, J. Zhang, L. Liu, and J. Liu, "Auditing privacy defenses in federated learning via generative gradient leakage," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 132–10 142.

[15] H. Kaur and V. Kumari, "Predictive modelling and analytics for diabetes using a machine learning approach," *Applied computing and informatics*, 2020.

[16] T. Shaikhina, D. Lowe, S. Daga, D. Briggs, R. Higgins, and N. Khovanova, "Machine learning for predictive modelling based on small data in biomedical engineering," *IFAC-PapersOnLine*, vol. 48, no. 20, pp. 469–474, 2015.

[17] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of artificial intelligence research*, vol. 4, pp. 237–285, 1996.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[19] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[22] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," *Proceedings of the IEEE*, vol. 111, no. 3, pp. 257–276, 2023.

[23] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image segmentation using deep learning: A survey," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 7, pp. 3523–3542, 2021.

[24] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," *Advances in neural information processing systems*, vol. 28, 2015.

[25] A. Vaswani, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017.

[26] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 2019, pp. 4171–4186.

[27] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz *et al.*, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, 2020, pp. 38–45.

[28] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big data*, vol. 3, pp. 1–40, 2016.

[29] A. Mashrur, W. Luo, N. A. Zaidi, and A. Robles-Kelly, "Machine learning for financial risk management: a survey," *Ieee Access*, vol. 8, pp. 203 203–203 223, 2020.

[30] A. Ali, S. Abd Razak, S. H. Othman, T. A. E. Eisa, A. Al-Dhaqm, M. Nasser, T. Elhassan, H. Elshafie, and A. Saif, "Financial fraud detection based on machine learning: a systematic literature review," *Applied Sciences*, vol. 12, no. 19, p. 9637, 2022.

[31] T.-V. Pricope, "Deep reinforcement learning in quantitative algorithmic trading: A review," *arXiv preprint arXiv:2106.00123*, 2021.

[32] A. Bora, S. Balasubramanian, B. Babenko, S. Virmani, S. Venugopalan, A. Mitani, G. de Oliveira Marinho, J. Cuadros, P. Ruamviboonsuk, G. S. Corrado *et al.*, "Predicting the risk of developing diabetic retinopathy using deep learning," *The Lancet Digital Health*, vol. 3, no. 1, pp. e10–e19, 2021.

[33] Q. Rao and J. Frtunikj, "Deep learning for self-driving cars: Chances and challenges," in *Proceedings of the 1st international workshop on software engineering for AI in autonomous systems*, 2018, pp. 35–38.

[34] R. Liu, F. Nageotte, P. Zanne, M. de Mathelin, and B. Dresp-Langley, "Deep reinforcement learning for the control of robotic manipulation: a focussed mini-review," *Robotics*, vol. 10, no. 1, p. 22, 2021.

[35] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural computation*, vol. 29, no. 9, pp. 2352–2449, 2017.

[36] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.

[37] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[38] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the 41st annual ACM symposium on Theory of computing (STOC)*. ACM, 2009, pp. 169–178.

[39] C. Gentry and S. Halevi, "Implementing gentry's fully homomorphic encryption scheme," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*. Springer, 2011, pp. 129–148.

[40] D. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM*, vol. 24, no. 2, pp. 84–90, 1981.

[41] X. Yang, Y. Feng, W. Fang, J. Shao, X. Tang, S.-T. Xia, and R. Lu, "An accuracy-lossless perturbation method for defending privacy attacks in federated learning," in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 732–742.

[42] J. Sun, A. Li, B. Wang, H. Yang, H. Li, and Y. Chen, "Soteria: Provable defense against privacy leakage in federated learning from representation perspective," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 9311–9319.

[43] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *ICC 2019-2019 IEEE international conference on communications (ICC)*. IEEE, 2019, pp. 1–7.

[44] T. Sery, N. Shlezinger, K. Cohen, and Y. C. Eldar, "Over-the-air federated learning from heterogeneous data," *IEEE Transactions on Signal Processing*, vol. 69, pp. 3796–3811, 2021.

[45] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[46] S. J. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečnỳ, S. Kumar, and H. B. McMahan, "Adaptive federated optimization," in *International Conference on Learning Representations*, 2020.

[47] Y. Wang, L. Lin, and J. Chen, "Communication-efficient adaptive federated learning," in *International conference on machine learning*. PMLR, 2022, pp. 22 802–22 838.

[48] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of adam and beyond," *arXiv*, 2019.

[49] H. Ren, J. Deng, X. Xie, X. Ma, and Y. Wang, "Fedboosting: Federated learning with gradient protected boosting for text recognition," *Neurocomputing*, vol. 569, p. 127126, 2024.

[50] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[51] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization." *Journal of machine learning research*, vol. 12, no. 7, 2011.

[52] M. Zaheer, S. Reddi, D. Sachan, S. Kale, and S. Kumar, "Adaptive methods for nonconvex optimization," *Advances in neural information processing systems*, vol. 31, 2018.

[53] M. N. Gibbs and D. J. MacKay, "Variational gaussian process classifiers," *IEEE Transactions on Neural Networks*, vol. 11, no. 6, pp. 1458–1464, 2000.

[54] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv preprint arXiv:1806.00582*, 2018.

[55] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," *arXiv preprint arXiv:1907.02189*, 2019.

[56] C. Xu, Z. Hong, M. Huang, and T. Jiang, "Acceleration of federated learning with alleviated forgetting in local training," *arXiv preprint arXiv:2203.02645*, 2022.

[57] G. Hinton, N. Srivastava, and K. Swersky, "Neural networks for machine learning lecture 6a overview of mini-batch gradient descent," 2012.

[58] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, and L. Antiga, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[59] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[60] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," 2009.

[61] Y. LeCun, "The mnist database of handwritten digits," *http://yann. lecun. com/exdb/mnist/*, 1998.

[62] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Ieee, 2009, pp. 248–255.

[63] Y. Yang, Z. Ma, B. Xiao, Y. Liu, T. Li, and J. Zhang, "Reveal your images: Gradient leakage attack against unbiased sampling-based secure aggregation," *IEEE Transactions on Knowledge and Data Engineering*, 2023.

[64] Y. Zhao, Y. Zhang, Y. Li, Y. Zhou, C. Chen, T. Yang, and B. Cui, "Minmax sampling: A near-optimal global summary for aggregation in the wide area," in *Proceedings of the 2022 International Conference on Management of Data*, 2022, pp. 744–758.

[65] L. Sun, J. Qian, and X. Chen, "Ldp-fl: Practical private aggregation in federated learning with local differential privacy," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, 2021.

[66] A. T. Hasan, Q. Jiang, J. Luo, C. Li, and L. Chen, "An effective value swapping method for privacy preserving data publishing," *Security and Communication Networks*, vol. 9, no. 16, pp. 3219–3228, 2016.

[67] M. A. P. Chamikara, P. Bertók, D. Liu, S. Camtepe, and I. Khalil, "Efficient data perturbation for privacy preserving and accurate data stream mining," *Pervasive and Mobile Computing*, vol. 48, pp. 1–19, 2018.

[68] M. Chamikara, P. Bertok, D. Liu, S. Camtepe, and I. Khalil, "Efficient privacy preservation of big data for accurate data mining," *Information Sciences*, vol. 527, pp. 420–443, 2020.

[69] H. Lee, J. Kim, S. Ahn, R. Hussain, S. Cho, and J. Son, "Digestive neural networks: A novel defense strategy against inference attacks in federated learning," *computers & security*, vol. 109, p. 102378, 2021.

[70] M. A. P. Chamikara, P. Bertok, I. Khalil, D. Liu, and S. Camtepe, "Privacy preserving distributed machine learning with federated learning," *Computer Communications*, vol. 171, pp. 112–125, 2021.

[71] Z. Bu, J. Dong, Q. Long, and W. J. Su, "Deep learning with gaussian differential privacy," *Harvard data science review*, vol. 2020, no. 23, 2020.

[72] Y. Li, Y. Zhou, A. Jolfaei, D. Yu, G. Xu, and X. Zheng, "Privacy-preserving federated learning framework based on chained secure multiparty computing," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6178–6186, 2020.

[73] K. Yadav, B. B. Gupta, K. T. Chui, and K. Psannis, "Differential privacy approach to solve gradient leakage attack in a federated machine learning environment," in *International Conference on Computational Data and Social Networks*. Springer, 2020, pp. 378–385.

[74] W. Wei, L. Liu, Y. Wut, G. Su, and A. Iyengar, "Gradient-leakage resilient federated learning," in *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2021, pp. 797–807.

[75] W. Wei, L. Liu, M. Loper, K.-H. Chow, M. E. Gursoy, S. Truex, and Y. Wu, "A framework for evaluating client privacy leakages in federated learning," in *Computer Security*. Springer, 2020, pp. 545–566.

[76] D. Scheliga, P. Mäder, and M. Seeland, "Precode-a generic model extension to prevent deep gradient leakage," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 1849–1858.

[77] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[78] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *2017 IEEE symposium on security and privacy (SP)*. IEEE, 2017, pp. 3–18.

[79] S. Truex, L. Liu, M. E. Gursoy, L. Yu, and W. Wei, "Towards demystifying membership inference attacks," *arXiv:1807.09173*, 2018.

[80] S. Truex, L. Liu, M. Gursoy, L. Yu, and W. Wei, "Demystifying membership inference attacks in machine learning as a service," *IEEE Transactions on Services Computing*, vol. 14, no. 6, pp. 2073–2089, 2021.

[81] C. A. Choquette-Choo, F. Tramer, N. Carlini, and N. Papernot, "Label-only membership inference attacks," in *International Conference on Machine Learning*. PMLR, 2021, pp. 1964–1974.

[82] G. Zhang, B. Liu, T. Zhu, M. Ding, and W. Zhou, "Label-only membership inference attacks and defenses in semantic segmentation models," *IEEE Transactions on Dependable and Secure Computing*, 2022.

[83] X. Xie, C. Hu, H. Ren, and J. Deng, "A survey on vulnerability of federated learning: A learning algorithm perspective," *Neurocomputing*, p. 127225, 2024.

[84] J. Jeon, K. Lee, S. Oh, J. Ok *et al.*, "Gradient inversion with generative image prior," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[85] H. Yin, A. Mallya, A. Vahdat, J. M. Alvarez, J. Kautz, and P. Molchanov, "See through gradients: Image batch recovery via gradinversion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 16 337–16 346.

[86] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the gan: information leakage from collaborative deep learning," in *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, 2017, pp. 603–618.

[87] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, "Beyond inferring class representatives: User-level privacy leakage from federated learning," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 2512–2520.

[88] H. Ren, J. Deng, X. Xie, X. Ma, and J. Ma, "Gradient leakage defense with key-lock module for federated learning," *arXiv preprint arXiv:2305.04095*, 2023.

[89] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 586–595.

[90] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[91] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv:1409.1556*, 2014.

[92] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[93] G. E. Box and D. A. Pierce, "Distribution of residual autocorrelations in autoregressive-integrated moving average time series models," *Journal of the American statistical Association*, vol. 65, no. 332, pp. 1509–1526, 1970.

[94] Y.-S. Lee and L.-I. Tong, "Forecasting time series using a methodology based on autoregressive integrated moving average and genetic programming," *Knowledge-Based Systems*, vol. 24, no. 1, pp. 66–72, 2011.

[95] Y. Funde and A. Damani, "Comparison of arima and exponential smoothing models in prediction of stock prices," *The Journal of Prediction Markets*, vol. 17, no. 1, pp. 21–38, 2023.

[96] E. S. Gardner Jr, "Exponential smoothing: The state of the art," *Journal of forecasting*, vol. 4, no. 1, pp. 1–28, 1985.

[97] E. De Faria, M. P. Albuquerque, J. Gonzalez, J. Cavalcante, and M. P. Albuquerque, "Predicting the brazilian stock market through neural networks and adaptive exponential smoothing methods," *Expert Systems with Applications*, vol. 36, no. 10, pp. 12 506–12 509, 2009.

[98] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intelligent Systems and their applications*, vol. 13, no. 4, pp. 18–28, 1998.

[99] B. De Ville, "Decision trees," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 5, no. 6, pp. 448–455, 2013.

[100] L. Breiman, "Random forests," *Machine learning*, vol. 45, pp. 5–32, 2001.

[101] J. L. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.

[102] K. Cho, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[103] K. Yi, J. Fei, Q. Zhang, H. He, S. Hao, D. Lian, and W. Fan, "Filternet: Harnessing frequency filters for time series forecasting," *arXiv preprint arXiv:2411.01623*, 2024.

[104] K. Yi, Q. Zhang, W. Fan, S. Wang, P. Wang, H. He, N. An, D. Lian, L. Cao, and Z. Niu, "Frequency-domain mlps are more effective learners in time series forecasting," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[105] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?" in *Proceedings of the AAAI conference on artificial intelligence*, vol. 37, 2023, pp. 11 121–11 128.

[106] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "Timesnet: Temporal 2d-variation modeling for general time series analysis," *arXiv preprint arXiv:2210.02186*, 2022.

[107] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A time series is worth 64 words: Long-term forecasting with transformers," *arXiv preprint arXiv:2211.14730*, 2022.

[108] J. Yan and J. Yu, "Cross-stock momentum and factor momentum," *Journal of Financial Economics*, vol. 150, no. 2, p. 103716, 2023.

[109] K. J. Koa, Y. Ma, R. Ng, and T.-S. Chua, "Diffusion variational autoencoder for tackling stochasticity in multi-step regression stock price prediction," in *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 2023, pp. 1087–1096.

[110] Z. Pei, J. Yan, J. Yan, B. Yang, Z. Li, L. Zhang, X. Liu, and Y. Zhang, "A stock price prediction approach based on time series decomposition and multi-scale cnn using ohlct images," *arXiv preprint arXiv:2410.19291*, 2024.

[111] Y. Dong and Y. Hao, "A stock prediction method based on multidimensional and multi-level feature dynamic fusion," *Electronics*, vol. 13, no. 20, p. 4111, 2024.

[112] W. Liu, Y. Ge, and Y. Gu, "Multi-factor stock price prediction based on gan-trellisnet," *Knowledge and Information Systems*, pp. 1–22, 2024.

[113] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[114] S. Bai, J. Z. Kolter, and V. Koltun, "Trellis networks for sequence modeling," *arXiv preprint arXiv:1810.06682*, 2018.

[115] Z. Dong, X. Fan, and Z. Peng, "Fnspid: A comprehensive financial news dataset in time series," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024, pp. 4918–4927.

[116] T.-W. Lee, P. Teisseyre, and J. Lee, "Effective exploitation of macroeconomic indicators for stock direction classification using the multimodal fusion transformer," *IEEE Access*, vol. 11, pp. 10 275–10 287, 2023.

[117] K. Nyakurukwa and Y. Seetharam, "The evolution of studies on social media sentiment in the stock market: Insights from bibliometric analysis," *Scientific African*, vol. 20, p. e01596, 2023.

[118] J. Liu and X. Meng, "Survey on privacy-preserving machine learning," *JCRD*, vol. 57, no. 2, p. 346, 2020.

[119] H. C. Tanuwidjaja, R. Choi, S. Baek, and K. Kim, "Privacy-preserving deep learning on machine learning as a service-a comprehensive survey," *IEEE Access*, vol. 8, pp. 167 425–167 447, 2020.

[120] N. Koti, M. Pancholi, A. Patra, and A. Suresh, "{SWIFT}: Super-fast and robust privacy-preserving machine learning," in *Security Symposium*, 2021.

[121] J. Shah, D. Vaidya, and M. Shah, "A comprehensive review on multiple hybrid deep learning approaches for stock prediction," *Intelligent Systems with Applications*, vol. 16, p. 200111, 2022.

[122] Y. Gao, R. Wang, and E. Zhou, "Stock prediction based on optimized lstm and gru models," *Scientific Programming*, vol. 2021, no. 1, p. 4055281, 2021.

[123] S. Pourroostaei Ardakani, N. Du, C. Lin, J.-C. Yang, Z. Bi, and L. Chen, "A federated learning-enabled predictive analysis to forecast stock market trends," *Journal of Ambient Intelligence and Humanized Computing*, vol. 14, no. 4, pp. 4529–4535, 2023.

[124] J. Kumarappan, E. Rajasekar, S. Vairavasundaram, K. Kotecha, and A. Kulkarni, "Federated learning enhanced mlp–lstm modeling in an integrated deep learning pipeline for stock market prediction," *International Journal of Computational Intelligence Systems*, vol. 17, no. 1, p. 267, 2024.

[125] N. Patel, N. Vasani, N. K. Jadav, R. Gupta, S. Tanwar, Z. Polkowski, F. Alqahtani, and A. Gafar, "F-lstm: Federated learning-based lstm framework for cryptocurrency price prediction." *Electronic Research Archive*, vol. 31, no. 10, 2023.