DynFed: Dynamic Test-Time Adaptation for Federated Learning with Adaptive Rate Networks

Anonymous authors
Paper under double-blind review

Abstract

Test-Time Personalized Federated Learning (TTPFL) has emerged as a promising approach for adapting models to distribution shifts in federated learning (FL) environments without relying on labeled data during testing. However, existing methods often struggle with heterogeneous shifts across clients and lack the flexibility to handle diverse distribution changes effectively. In this paper, we introduce DynFed, a novel algorithm that dynamically optimizes test-time adaptation (TTA) in FL scenarios with heterogeneous distribution shifts. Our method leverages Adaptive Rate Networks (ARNs) to generate client-specific adaptation rates, enabling more effective handling of diverse shift types, including label skew and feature shifts. DynFed employs an innovative iterative adaptation process, where ARNs continuously refine adaptation rates based on the current adaptation state, without direct access to raw client data. Crucially, we uncover a fundamental dichotomy: optimal adaptation strategies for one-type and multi-type distribution shifts are diametrically opposed. DynFed navigates this challenge by automatically adjusting its approach based on the nature of the encountered shifts. Extensive experiments and theoretical analysis demonstrate that DynFed significantly outperforms existing TTPFL and TTA methods across various shift scenarios. Our method shows particularly robust performance in complex multi-type shift environments, where previous approaches often struggle. This work opens new avenues for adaptive and resilient FL in real-world applications where distribution shifts are diverse and unpredictable.

1 Introduction

Federated Learning (FL) has emerged as a powerful paradigm for distributed machine learning, enabling models to be trained across multiple decentralized edge devices or servers holding local data samples without the need to exchange them McMahan et al. (2017); Zhao et al. (2018); Mohri et al. (2019); Wang et al. (2021b; 2023; 2024c;b;a). This paradigm ensures data privacy and security by keeping data localized. However, the performance of FL models often degrades when confronted with distribution shifts between training and test data, a challenge exacerbated by the heterogeneous nature of client data in real-world scenarios.

Test-time Adaptation (TTA) has shown promise in addressing this issue by allowing models to adapt to new distributions during inference Zhang et al. (2022); Wang et al. (2021a; 2022). TTA methods enable models to adjust to unseen data distributions, thus enhancing robustness and performance. Recently, Test-time Personalized Federated Learning (TTPFL) has been proposed to combine the benefits of TTA with FL Bao et al. (2024), allowing for unsupervised local adaptation of global models during test time. TTPFL methods provide a framework for individualized model adjustments based on the local data characteristics of each client. However, Existing methods for handling distribution heterogeneity in FL face a critical limitation as depicted as Figure 1 (b). These approaches typically operate under the assumption that all clients encounter data conditions of a consistent type, known as one-type distribution heterogeneity.

This implies that the heterogeneous distributions across clients are uniform, such as all clients experiencing either label skew or all experiencing feature shift. In real-world scenarios, it is impossible to preset the type of heterogeneous distribution each client may encounter. Multi-type distribution heterogeneity is

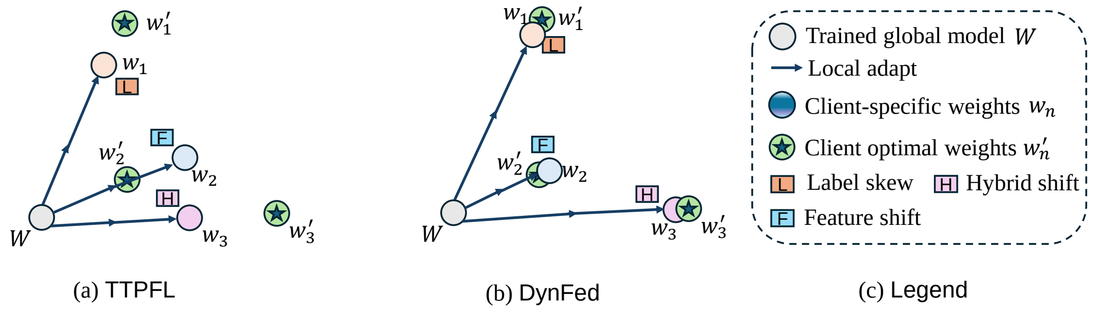


Figure 1: Comparison of TTPFL and DynFed adaptation processes. (a) TTPFL applies uniform adaptation across all clients, resulting in identical local models after each round. (b) DynFed employs client-specific adaptation, leading to diverse local models tailored to each client's distribution. The varying sizes of local model weights (w_1, w_2, w_3) in (b) illustrate the personalized nature of DynFed's adaptation. (c) Legend explaining the symbols used in the figure, where W represents the global model, and w_n denotes n th client-specific weights after adaptation.

common in FL settings, where some clients may face label skew while others face feature shift. In response to this complex heterogeneous environment, current methods employ fixed adaptation rates for all clients and model components, severely restricting their ability to effectively adapt to heterogeneous distribution shifts. These fixed rates fail to account for the varying degrees of distribution shifts experienced by different clients, resulting in suboptimal adaptation and performance. To address this challenge, we introduce DynFed, a novel algorithm designed to dynamically optimize test-time adaptation in FL system characterized by heterogeneous distribution shifts. Our approach leverages Adaptive Rate Networks (ARNs) to generate client-specific adaptation rates, enabling more effective handling of diverse shift types. The comparison illustrated in Figure 1 highlights the key advantage of our proposed DynFed framework over traditional TTPFL. Our method facilitate client-specific, dynamic adaptation of the global model, in contrast to the uniform adaptation approach employed by conventional TTPFL methods. DynFed employs an innovative iterative adaptation process, where ARNs continuously refine adaptation rates based on the current adaptation state. This process ensures that the adaptation rates are tailored to the specific batch data of each client, thereby enhancing the overall performance and robustness of the federated system in the presence of diverse distribution shifts. The main contributions of our work are as follows:

- We propose a new framework for TTPFL that explicitly accounts for multi-type distribution shifts across clients, including both label skew and feature shifts.
- We introduce Adaptive Rate Networks (ARNs) that incremental generate personalized adaptation rates for each client and model component, significantly improving upon fixed-rate adaptation methods.
- We uncover a critical insight: the optimal adaptation strategies for one-type and multi-type distribution shifts are fundamentally opposed. Our method, DynFed, effectively navigates this dichotomy by automatically adjusting its strategy based on the nature of the distribution shifts encountered.
- Extensive experiments on various FL benchmarks show that DynFed significantly outperforms traditional TTA and the state-of-the-art TTPFL methods across different types of distribution shifts.

In general, our work represents a significant step towards making the TTPFL framework more robust and adaptable in real-world scenarios, where distribution shifts are diverse and unpredictable. By addressing the limitations of current TTPFL methods, DynFed paves the way for more effective and flexible FL systems capable of handling the complexities of real-world data distributions. The rest of this paper is organized as follows: We first review related work in Section 2. Section 3 formally defines the problem setting. We then detail our proposed method in Section 4. Experimental results are presented in Section 5.

2 Related Work

2.1 Federated Learning.

Federated Learning (FL) has emerged as a transformative paradigm in distributed machine learning, addressing critical concerns of privacy and data locality. Introduced by McMahan et al. (2017), FL enables model training across decentralized edge devices or servers holding local data samples, without the need for direct data exchange. The seminal FedAvg algorithm, which aggregates locally computed gradients to update a global model, laid the foundation for this field. Since its inception, FL has undergone rapid development across various aspects, including communication efficiency Konečnỳ et al. (2016), privacy preservation Geyer et al. (2017).

The core challenge in real-world FL applications is data heterogeneity, commonly referred to as the non-IID (not independently and identically distributed) problem Zhao et al. (2018). This issue arises as clients often possess diverse label and feature distributions, reflecting their varied behaviors and habits Tan et al. (2022b); Luo et al. (2021). To address this challenge, several innovative approaches have proposed. Clustering-based FL methods aim to group clients based on their data similarity Ghosh et al. (2019); Ma et al. (2022), allowing for more targeted model updates. Concurrently, meta-learning techniques have been leveraged to enhance the personalization capabilities of local models Fallah et al. (2020). Additionally, model decoupling schemes have been introduced to facilitate better personalization while maintaining the benefits of collaborative learning Tan et al. (2023); Bao et al. (2023); Baek et al. (2023).

2.2 Test-Time Adaption.

Test-Time Adaptation (TTA) has emerged as a crucial method in machine learning for addressing distribution shifts between training and test data without necessitating model retraining or access to labeled test samples. The core principle of TTA involves adapting the model during inference using only unlabeled test data. Various approaches have been developed to tackle this challenge effectively. Entropy minimization, introduced by Wang et al. (2021a), stands as a fundamental TTA technique. This method fine-tunes the model to generate more confident predictions on test data, thereby aligning the model with the target distribution. Building upon this concept, self-supervised learning has been integrated into TTA frameworks to more efficiently utilize unlabeled test data Chen et al. (2022). This integration allows for the creation of auxiliary tasks that guide adaptation without relying on explicit labels. Recent advancements in TTA include more sophisticated techniques. SAR Niu et al. (2023) enhances adaptation by eliminating high-gradient samples and promoting weights that lead to flat minima, thus improving generalization. In the realm of pseudo labeling-based methods, PL Lee et al. (2013) refines model parameters using confidently predicted pseudo labels. SHOT Liang et al. (2020) takes a hybrid approach, combining entropy minimization strategies with pseudo labeling techniques to achieve robust adaptation. These diverse methods collectively represent the ongoing efforts to develop TTA approaches that are both effective and widely applicable across various domains and types of distribution shifts.

2.3 Test-Time Adaptation in Federated Learning.

Test-Time Adaptation in Federated Learning (TTA-FL) presents a more complex scenario compared to traditional end-to-end TTA, primarily due to the inherent data distribution challenges in FL environments Jiang & Lin (2023); Tan et al. (2024); Wan et al. (2024). While conventional TTA methods focus on adapting models to shifts between a single source and target domain, TTA-FL must contend with multiple, heterogeneous client distributions, It can also be called Test-Time Personlized Federated Learning (TTPFL) Bao et al. (2024), this means that the source model is a trained global model, but personalized to the data of individual clients. Existing methods assume that all clients face consistent data type conditions, referred to as one-type distribution heterogeneity, where the heterogeneous distributions on each client are the same, such as either uniformly label skew or uniformly feature shift. However, in real-world scenarios, it is impossible to preset the type of heterogeneous distribution on each client. Multi-type distribution heterogeneity is quite common, where in a FL scenario, some clients experience label skew while others experience feature shift.

Existing TTPFL Bao et al. (2024) methods utilize a uniform adaptation rate for all participating clients, which is evidently inadequate for addressing the needs of multi-type distribution heterogeneity scenarios.

3 Preliminaries

In this section, we first introduced the notion of TTPFL antecedents. Then the limitations of the current framework for TTPFL are discussed.

3.1 Test-time personalized federated learning

3.1.1 Global and Personalized Federated Learning

Global Federated Learning (GFL) aims to find a single global model that minimizes the expected loss over the client population:

$$\mathcal{L}(w_G) = \mathbb{E}_{P\sim Q}[\mathcal{L}_P(w_G)], \quad \text{where} \quad \mathcal{L}_P(w_G) = \mathbb{E}_{(x,y)\in P}\ell(f(x; w_G); y) \tag{1}$$

where w_G represents the parameters of the global model, $\mathcal{L}(w_G)$ is the overall loss function for the global model, P represents the data distribution of an individual client, Q is the distribution of client distributions, $\mathcal{L}_P(w_G)$ is the loss function for a specific client with distribution P, x and y are input features and labels respectively, $\ell(\cdot;\cdot)$ represents the loss function (e.g., cross-entropy), $f(\cdot;\cdot)$ represents the model function parameterized by w_G, and $\mathbb{E}[\cdot]$ denotes the expectation operator. GFL mandates that each client uses the same global model for prediction, precluding adaptation to each client's unique data distribution. In contrast, personalized federated learning (PFL) customizes the global model w_G using the client's labeled data and employs the personalized model for prediction. However, most PFL algorithms Tan et al. (2022a); Deng et al. (2020); Fallah et al. (2020) assume that the target client also possesses additional labeled data, a stronger assumption compared to GFL.

3.1.2 Test-Time Personalized Federated Learning

In the paper Bao et al. (2024), they introduced a novel paradigm called test-time personalization federated learning (aka TTPFL). TTPFL focuses on adapting a trained global model to each target client's unlabeled data during test-time, utilizing an adaptation rule A that operates solely on unlabeled data. The objective function can be formulated as:

$$\mathcal{L}(\mathbf{w}_G, \mathrm{A}) = \mathbb{E}_{P\sim Q}[\mathcal{L}_P(\mathbf{w}_G, \mathrm{A})], \quad \text{where} \quad \mathcal{L}_P(\mathbf{w}_G, \mathrm{A}) = \mathbb{E}_{(x,y)\in P}\ell(f(x; \mathrm{A}(\mathbf{w}_G, X)); y) \tag{2}$$

which can be unbiasedly estimated by the average loss over M target clients unseen during training:

$$\hat{\mathcal{L}}(\mathbf{w}_G, \mathrm{A}) = \frac{1}{M}\sum_{j=1}^{M}\hat{\mathcal{L}}_{T_j}(\mathbf{w}_G, \mathrm{A}), \quad \text{where} \quad \hat{\mathcal{L}}_{T_j}(\mathbf{w}_G, \mathrm{A}) = \frac{1}{m_j}\sum_{r=1}^{m_j}\ell(f(x^r_{T_j}; \mathrm{A}(\mathbf{w}_G, X^r_{T_j})); y^r_{T_j}) \tag{3}$$

where $\hat{\mathcal{L}}_{P_j}(w_G, \mathrm{A})$ denotes the empirical loss on the j-th target client. The adaptation rule A modifies the global model using unlabeled samples $X^r_{T_j}$. TTPFL consider two standard settings: Test-Time Batch Adaptation (TTBA) and Online Test-Time Adaptation (OTTA)Liang et al. (2024). TTBA individually adapts the global model to each batch of unlabeled samples, where $X^r_{T_j}$ represents the data batch containing $x^r_{T_j}$. OTTA adapts the global model in an online manner, where $X^r_{T_j}$ encompasses all data batches arriving before or concurrently with $x^r_{T_j}$.

3.2 Limitation of TTPFL

Despite the promising advancements in TTPFL, current approaches suffer from several key limitations that hinder their effectiveness in real-world scenarios. Existing methods often presume a uniform distribution shift across all clients, neglecting the reality of heterogeneous shifts in federated environments. This oversimplification is compounded by the prevalent use of static, predefined adaptation rates for all clients and model components, significantly limiting the ability to adapt to diverse and dynamic distribution shifts. To formally express this limitation, we can formulate the TTPFL objective as:

$$\hat{\mathcal{L}}(\mathbf{w}_G, \alpha_j j = 1^M, \delta_j j = 1^M) = \frac{1}{M} \sum j = 1^M \hat{\mathcal{L}}_{P_j}(\mathbf{w}_G, \alpha_j, \delta_j) \tag{4}$$

where $\mathbf{w}_G$ represents the global model parameters, $\alpha_j j = 1^M$ denotes the set of adaptation rates for M clients, $\delta_j j = 1^M$ represents the set of distribution shifts for each client, and $\hat{L}_{P_j}(w_G, \alpha_j, \delta_j)$ is the empirical loss on client j using adaptation rate α_j under distribution shift δ_j. This formulation highlights the key limitation: each client j may require a different adaptation rate α_j to handle its specific distribution shift δ_j, which is challenging to achieve effectively in practice, especially for multi-type distribution shifts and dynamically changing environments. Furthermore, many TTPFL algorithms lack mechanisms to gradually adjust adaptation strategies based on evolving client data distributions, potentially leading to suboptimal performance over time Zhang et al.. The inefficient use of historical adaptation information also represents a missed opportunity for more informed and efficient adaptation processes. These interrelated issues collectively impair the effectiveness of TTPFL in complex, real-world federated learning environments Zeng et al. (2023); Zhang et al. (2024), where distribution shifts are often multifaceted, diverse, and continually evolving.

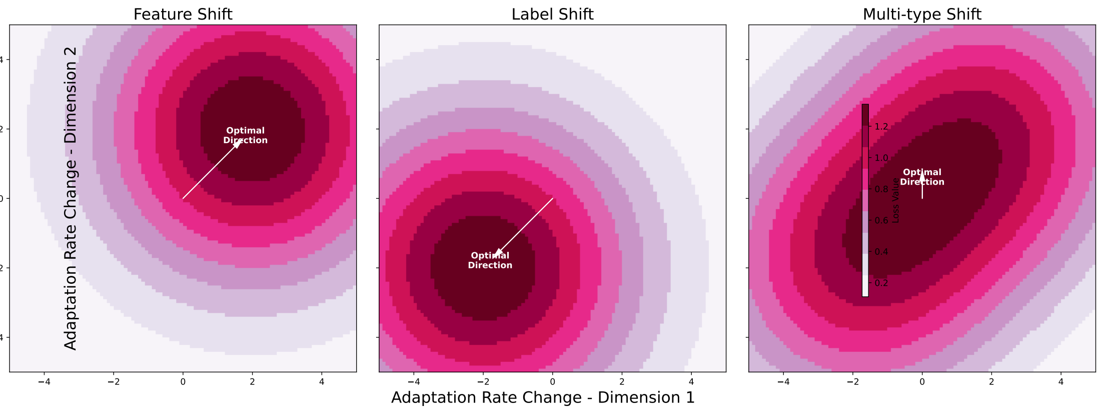


Figure 2: Loss landscapes and optimal adaptation directions for different types of distribution shifts. Left: Feature shift only. Middle: Label shift only. Right: Multi-type distribution shift (combined feature and label shifts in different clients, And the two-type shifts don't overlap). White arrows indicate the optimal adaptation directions. Note that the optimal direction for multi-type shift differs significantly from single-type shifts in FL, justifying the need for different adaptive scaling in participant clients.

3.3 Challenges

The primary challenge in TTPFL lies in effectively addressing the not only one-type distribution heterogeneity prevalent in real-world FL environments. This challenge is significantly more complex than dealing with one-type distribution heterogeneity, as the optimization directions for different types of shifts can be inconsistent or even conflicting. Our experiment observations, as illustrated in Figure 2, reveal a critical insight: Even the direction of the adaption is not consistent across one-type distribution shifts. When multiple distinct distribution shifts occur simultaneously across different clients, the optimal adaptation direction without server communication can be diametrically opposed to the direction in scenarios with a single, uniform shift. This phenomenon underscores the inadequacy of conventional TTPFL methods that assume a homogeneous distribution shift across all clients.

[Adaptation Strategy Dichotomy] Let $\mathcal{S}_1, \ldots, \mathcal{S}_N$ denote a set of clients experiencing a one-type distribution shift (either purely feature or purely label shift), and let $\mathcal{M}_1, \ldots, \mathcal{M}_N$ denote a set of clients experiencing multi-type (mixed) distribution shifts. Then, under strong convexity assumptions, the optimal adaptation rate vectors $\alpha^*_{\mathcal{S}}$ for one-type shifts and $\alpha^*_{\mathcal{M}}$ for multi-type shifts satisfy

$$\alpha^*_{\mathcal{S}} \cdot \alpha^*_{\mathcal{M}} < 0, \tag{5}$$

where "$\cdot$" denotes the dot product, indicating that the optimal adaptation directions are negatively correlated.

The core challenge, therefore, is to develop a unified approach that can effectively handle diverse types of distribution shifts occurring across different clients. This approach must be capable of gradually adjusting adaptation strategies to accommodate these heterogeneous shifts. Furthermore, this solution must operate within the constraints of FL, maintaining data privacy and minimizing communication overhead. The ability to address these multifaceted challenges without compromising the core principles of FL is the cornerstone of our proposed DynFed method.

4 Method

In this section, we introduce DynFed, a novel framework that gradually generates client-specific adaptation rates for each module in FL settings. Our approach addresses the challenges of heterogeneous distribution shifts across clients, offering a more flexible and efficient solution compared to existing methods. We detail the training and testing phases of DynFed, followed by a discussion of its advantages.

4.1 Training Phase: Learning Adaptation Rates with Source Clients

Following the training stage of Bao et al. (2024), our training phase builds upon the TTPFL framework while introducing key innovations. DynFed employs a communication protocol similar to FedAvg McMahan et al. (2017) to initialize and refine adaptation rates. The training process is iterative, with each round consisting of local unsupervised adaptation, supervised refinement, and server aggregation. Similar to previous works, we consider the model processes a data batch $X^k_{Si} = \{x_{Si,k,b}\}^B_{b=1}$ at a time where B is the batch size, i is the client index and k is the batch index. In the following, we omit the superscript Si for clarity, e.g. $X^k_{Si} \to X^k$, as unsupervised adaptation and supervised refinement operate identically across all source clients.

Unsupervised Adaptation: We consider a neural network model $f(\cdot; \mathbf{w}_G)$ with global model parameter $\mathbf{w}_G \in \mathbb{R}^D$. The network comprises d modules, each with parameters $w^{[1]}, \dots, w^{[d]}$. During unsupervised adaptation, we allow each module $w^{[l]}$ to have a distinct adaptation rate $\alpha^{[l]}$, enabling fine-grained control over the adaptation process.

Update Trainable Parameters: For each trainable parameter module $w^{[l]}$, we compute the unsupervised update direction as the negative gradient of the entropy loss:

$$h^{[l]}_k = -\nabla_{w^{[l]}} \ell_H(f(X_k; \mathbf{w}_G)) \tag{6}$$

where $\ell_H(\hat{Y}) = -\frac{1}{B}\sum_{b=1}^{B}\sum_{c=1}^{C} \hat{y}_{b,c} \log \hat{y}_{b,c}$ is the entropy loss. This approach allows the model to adapt without requiring labeled data, making it suitable for real-world scenarios where labels may be scarce or unavailable.

Update Running Statistics: For batch normalization layers, which play a crucial role in adapting to distribution shifts, we define the update direction as:

$$h^{[l]}_k = \hat{w}^{[l]}_k - w^{[l]}_G \tag{7}$$

where $\hat{w}^{[l]}_k$ represents the statistics for the current batch of inputs, and $w^{[l]}_G$ is the running statistics. This update ensures that the batch normalization layers can effectively adapt to the local data distribution of each client.

After computing the update directions, each module is updated using its corresponding adaptation rate:

$$w_k^{[l]} \leftarrow w_G^{[l]} + \alpha^{[l]} h_k^{[l]} \tag{8}$$

Supervised Refinement: Following the unsupervised adaptation, we refine the adaptation rates using gradient descent on the cross-entropy loss:

$$\alpha \leftarrow \alpha - \eta_\alpha \nabla_\alpha \ell_{CE}(f(X_k; w_k), Y_k) \tag{9}$$

where η_α is the learning rate for adaptation rates, and ℓ_{CE} is the cross-entropy loss. This step allows the model to fine-tune its adaptation strategy based on labeled data, enhancing its performance on the target task.

Server Aggregation: To leverage the collective knowledge of all clients and improve generalization, we employ federated aggregation to periodically combine the local adaptation rates. This step is crucial for ensuring that the learned adaptation strategies are effective across a diverse range of clients and data distributions.

4.2 Testing Phase: Exploiting Dynamic Adaptation on Target Clients

During the testing phase, each target client receives the global model, the Adaptive Rate Network (ARN), and the initial adaptation rates. We consider two settings, following TTPFL Bao et al. (2024): DynFed-batch for test-time batch adaptation (TTBA) and DynFed-online for online test-time adaptation (OTTA).

Adaptive Rate Network (ARN): The cornerstone of our incremental adaptation mechanism is the ARN, denoted as $g(\cdot; \theta)$. This neural network is designed to capture and model the complex relationships between adaptation rates and model performance across various distribution shifts. The ARN takes the current adaptation rates as input and generates refined, context-aware rates tailored to each client's specific data distribution.

We formally define the ARN as a compact neural network with non-linear activations:

$$g(x; \theta) = \tau \cdot \sigma(f_i(f_1(x)))) \tag{10}$$

where f_i represents the i-th layer of the network, σ is the sigmoid activation function, and τ is a scaling factor of hyperparameter that ensures the output adaptation rates fall within an appropriate range. Each layer f_i is defined as:

$$f_i(x) = \sigma(W_i x + b_i) \tag{11}$$

Here, W_i and b_i are learnable weight matrices and bias vectors, respectively, and σ is the rectified linear unit activation function. This architecture enables the ARN to learn complex, non-linear mappings from current adaptation rates to optimized rates, allowing for highly flexible and context-specific adaptation strategies.

Dynamic Adaptation Rate Computation: We introduce a novel mechanism to dynamically compute the adaptation rates. The process starts with an initial set of adaptation rates, which are then iteratively updated for each batch of data on each client. Let α_t denote the adaptation rates at time step t. We compute the adaptation rates for the next time step as follows:

$$\alpha_{t+1} = \beta \cdot \alpha_t + (1 - \beta) \cdot g(\alpha_t, x) \tag{12}$$

where β is a learnable parameter, $g(\cdot, \cdot)$ is a function implemented by the ARN that generates new adaptation rates based on the current rates and batch data distribution, and x represents the distribution of the current batch data. The initial adaptation rates α_t are set to predetermined values or can be learned during a warm-up phase. This formulation allows for a dynamic balance between maintaining the current adaptation strategy and adjusting to the characteristics of the incoming data batch, enabling the model to continuously refine its adaptation approach as it processes more data.

Algorithm 1 FedDyn-Training

```
Input: Global model $\mathbf{w}_G$, initial adaptation rates $\alpha_0^G$
Parameter: Number of communication rounds $T$, number of clients per round $C$
Output: Final adaptation rates $\alpha_G^T$
 1: Broadcast $w^G$ to all source clients
 2: for communication round $t = 1$ to $T$ do
 3:     $S_t \leftarrow$ (random set of $C$ source clients)
 4:     for all source client $S_i \in S_t$ in parallel do
 5:         $\alpha_i^t \leftarrow \textsc{ClientTrain}(S_i, \alpha_{t-1}^G)$
 6:     end for
 7:     $\alpha_G^t = \frac{1}{C}\sum_{S_i \in S_t} \alpha_i^t$
 8: end for
 9: return $\alpha_G^T$,

    ClientTrain($S_i, \alpha$):
10: for local epoch $e = 1$ to $E$ do
11:     for all batch $k$ in $D^{S_i}$ do
12:         $\alpha_{k+1} = \beta \cdot \alpha_k + (1-\beta) \cdot g(\alpha_k, X_k^{S_i};)$
13:         $h_k^{S_i} \leftarrow -\nabla_{w_G} \ell_H(f(X_k^{S_i}; w_G))$
14:         $w_k^{S_i} \leftarrow w^G + \mathrm{A}) \odot h_k^{S_i}$
15:         $\alpha \leftarrow \alpha - \eta \nabla_\alpha \ell_{CE}(f(X_k^{S_i}; w_k^{S_i}), Y_k^{S_i})$
16:     end for
17: end for
18: return $\alpha$
```

DynFed-batch: In the TTBA setting, each target client processes data batches independently. For each batch, DynFed-batch first generates adaptation rates using the ARN, then conducts unsupervised adaptation, and finally makes predictions. This approach allows for quick adaptation to batch-specific characteristics while maintaining consistency with the global model.

DynFed-online: For OTTA, we propose a modified version of the averaged adaptation mechanism to mitigate batch dependency:

$$w_k^T \leftarrow w_G + (A_{g(\alpha,x;\theta)}) \odot \frac{1}{k}\sum_{s=1}^{k} h_s^T \tag{13}$$

where $g(\alpha, x; \theta)$ generates the dynamic adaptation rates. By using the average of previous updates, we effectively simulate updating with a larger batch size, utilizing historical data while controlling the number of update steps. This approach allows for more stable adaptation in online scenarios, where data arrives sequentially and may exhibit temporal dependencies.

Algorithm 1 followed the previous work Bao et al. (2024) and Algorithm 2 provide a detailed summary of testing phases of DynFed, respectively.

4.3 Discussion

DynFed offers several key advantages over existing TTPFL methods. Firstly, our approach is inherently more flexible due to the use of the Adaptive Rate Network, which can gradually adjust adaptation rates based on the specific characteristics of each client's data distribution. This flexibility allows DynFed to handle a wider range of distribution shifts more effectively than static adaptation methods.

Algorithm 2 FedDyn-Testing

Input: Target client T_j, global model $\mathbf{W}_G$, adaptation rates α, ARN $g(\cdot;\theta)$
Parameter: Adaptation mode (TTBA or OTTA)
Output: Adapted predictions $\hat{Y}^T_{jk}$

```
1: Initialize $h_{\text{history}} \leftarrow 0$
2: for all batch $k$ in test dataset $D^T_j$ do
3:     $\alpha_t \leftarrow g(\alpha, x; \theta)$, $\alpha_{t+1} = \beta \cdot \alpha_t + (1-\beta) \cdot g(\alpha_t, x)$
4:     $h^T_{jk} \leftarrow -\nabla_{w_G} \ell(f(D^T_j; \mathbf{w}_G))$
5:     if TTBA then
6:         $w^T_{jk} \leftarrow w_G + \mathrm{A} \odot h^T_{jk}$
7:     else
8:         $h_{\text{history}} \leftarrow \frac{k-1}{k} h_{\text{history}} + \frac{1}{k} h^T_{jk}$
9:         $w^T_{jk} \leftarrow w_G + \mathrm{A} \odot h_{\text{history}}$
10:    end if
11:    $\hat{Y}^T_{jk} = f(D^T_{jk}; w^T_{jk})$
12: end for
13: return $\{\hat{Y}^T_{jk}\}$
```

4.4 Theoretical Properties of DynFed

Our approach not only demonstrates empirical improvements but also possesses strong theoretical guarantees. We provide the key theoretical results here while full proofs are given in the Appendix.

4.4.1 Convergence Analysis

[Bounded Gradients] The expected squared norms of the stochastic gradients are bounded, i.e.,

$$\mathbb{E}\left\|\nabla_{\mathbf{w}_G}\mathcal{L}_{P_j}(\mathbf{w}_G, \alpha_j) - \nabla_{\mathbf{w}_G}\mathcal{L}(\mathbf{w}_G, \{\alpha_j\}_{j=1}^M)\right\|^2 \le \sigma_w^2,$$

and

$$\mathbb{E}\left\|\nabla_{\alpha_j}\mathcal{L}_{P_j}(\mathbf{w}_G, \alpha_j) - \nabla_{\alpha_j}\mathcal{L}(\mathbf{w}_G, \{\alpha_j\}_{j=1}^M)\right\|^2 \le \sigma_\alpha^2.$$

[Convergence of DynFed] Under Assumptions 1 and 2, with appropriately chosen learning rates η_w and η_α for the model parameters and adaptation rates respectively, the expected decrease in the Lyapunov function after T rounds is bounded by:

$$\mathbb{E}[V_T] - \mathbb{E}[V_0] \le -\sum_{t=0}^{T-1}\left(\frac{\eta_w}{2}\|\nabla_{\mathbf{w}_G}\mathcal{L}(\mathbf{w}_G^t, \{\alpha_j^t\}_{j=1}^M)\|^2 + \frac{\gamma\eta_\alpha}{2}\sum_{j=1}^M\|\nabla_{\alpha_j}\mathcal{L}(\mathbf{w}_G^t, \{\alpha_j^t\}_{j=1}^M)\|^2\right) + C, \quad (14)$$

where C is a constant depending on η_w, η_α, β, σ_w^2, and σ_α^2.

This theorem guarantees that despite the additional complexity of client-specific adaptation rates, DynFed converges at a rate comparable to standard federated learning methods.

4.4.2 Generalization Guarantees

[Generalization] Let $\mathcal{H} = \{\alpha : \|\alpha\|_2 \le R\}$ be the hypothesis space of adaptation rates, N be the number of source clients, and K be the number of data batches per source client. Assuming (1) the model is L-Lipschitz and (2) the update directions for each module are bounded by H, then for any fixed global model $\mathbf{w}_G$ and any $\epsilon > 0$, we have

$$\Pr\Big(\sup_{\alpha\in\mathcal{H}} |\varepsilon(\alpha) - \hat{\varepsilon}(\alpha)| \ge \epsilon\Big) \le \left(\frac{12LHR}{\epsilon}\right)^d \cdot 4\exp\Big(-\frac{NK\epsilon^2}{2(\sqrt{K}+1)^2}\Big), \quad (15)$$

where $\hat{\varepsilon}(\alpha)$ is the average post-adaptation error rate on source clients and $\varepsilon(\alpha)$ is the expected post-adaptation error rate over the client population, and d is the dimensionality of the adaptation parameter vector.

This theorem demonstrates that DynFed can generalize well to unseen clients despite its enhanced expressiveness through client-specific adaptation. The generalization error is controlled by the capacity of the adaptation parameter space and decreases with more source clients and more batches per client.

4.4.3 Adaptation Mechanisms for Different Distribution Shifts

[Adapting the Last Layer for Label Shift] Consider two distributions p and q with

$$p(\mathbf{x}|\mathbf{y}) = q(\mathbf{x}|\mathbf{y}) \quad \text{and} \quad p(\mathbf{y}) \neq q(\mathbf{y}). \tag{16}$$

If a neural network is calibrated on p, i.e., $f(\mathbf{x};\mathbf{w}) = p(\cdot|\mathbf{x})$, then it can be calibrated on q by adding $\log \frac{q(\mathbf{y})}{p(\mathbf{y})}$ to the bias term of the final layer.

[Adapting the BN Layer for Feature Shift] Assume that the feature shift causes differences only in the first and second order moments of the feature activations $\mathbf{z} = g(\mathbf{x})$, and that the activations are independent. Then, the feature shift can be removed by adapting the running mean and variance of the Batch Normalization (BN) layer.

These propositions formally justify why different model components require different adaptation strategies for various types of distribution shifts, which is a core principle behind our DynFed approach.

5 Experiments

5.1 Experiments setting

We evaluate our proposed framework on a variety of models, datasets and distribution shifts. We first evaluate on CIFAR-10(-C) with a standard three-way split Yuan et al. (2021): we randomly split the dataset to 300 clients: 240 source clients and 60 target clients. Each source client has 160 training samples and 40 validation samples, while each target client has 200 unlabeled testing samples. We simulate four kinds of one-type distribution shifts: feature shift, label shift, hybrid shift and multi-type distribution shift (MT shift), there are different shifts within a federated system that operate on different clients. It is worth noting that unlike the previous hybrid shift, multi-type shifts are not overlapping. For feature shift, we follow Hendrycks & Dietterich (2019); Jiang & Lin (2023), randomly apply 15 different kinds of corruptions to the source clients, and 4 new kinds of corruptions to the target clients to test the generalization of ATP. For label shift, we use the step partition Chen & Chao (2020), where each client has 8 minor classes with 5 images per class, and 2 major classes with 80 images per class. For the hybrid shift, we apply both step partition and feature perturbations, it is equals basically two overlapping shifts on each client. We also test ATP with two different architectures: a five-layer CNN on CIFAR-10(-C) and ResNet-18 on CIFAR-100(-C). Baseline models we chose several common models: the BN-adapt Schneider et al. (2020), Tent Wang et al. (2021a), T3A Iwasawa & Matsuo (2021), MEMO Zhang et al. (2022) and the current state-of-the-art framework ATP Bao et al. (2024) in TTPFL. To simplify the expression, we use DynFed-B as well as DynFed-O to denote the TTBA and OTTA.

5.2 Experiment results

5.2.1 Performance comparison

We evaluate DynFed against state-of-the-art test-time adaptation techniques on CIFAR-10 and CIFAR-100 datasets under various distribution shifts. The results are shown in Table 1. Our method consistently outperforms all baselines, including the current state-of-the-art ATP, across all shift types. On CIFAR-10, DynFed-OTTA achieves accuracy improvements of 1.95%, 2.51%, 1.30%, and 1.78% over ATP for feature, label, hybrid, and multi-type shifts, respectively. The gains are even more significant on CIFAR-100, with improvements of 1.63%, 2.05%, 1.36%, and 2.86% for the same shift types. Both evaluation variants of our

	CIFAR-10				CIFAR-100			
Method	Feature	Label	Hybrid	MT	Feature	Label	Hybrid	MT
No adapt	64.36±.13	69.42±.24	63.68±.24	57.71±.20	35.12±.22	38.88±.32	33.68±.30	26.50±.24
BN adapt	65.52±.22	64.54±.10	60.02±.39	54.02±.18	36.52±.27	35.54±.15	32.02±.45	25.80±.22
Tent	65.76±.09	70.13±.21	63.42±.26	57.62±.24	36.76±.13	39.13±.28	34.05±.33	27.30±.28
T3A	64.53±.08	71.70±.32	62.17±.17	58.48±.21	35.53±.12	40.70±.38	33.17±.23	28.00±.25
MEMO	62.43±.22	78.45±.15	63.07±.29	55.49±.12	34.43±.30	44.31±.29	34.07±.37	26.30±.17
ATP	66.54±.12	76.67±.22	68.37±.35	60.69±.19	39.54±.27	48.97±.30	40.12±.40	31.19±.25
DynFed-B	**68.12±.18**	**78.54±.19**	**69.56±.31**	**62.23±.17**	**40.49±.22**	**50.19±.25**	**41.18±.35**	**33.72±.20**
DynFed-O	**68.49±.25**	**79.18±.27**	**69.67±.36**	**62.47±.19**	**41.17±.28**	**51.02±.32**	**41.48±.38**	**34.05±.22**

Table 1: Accuracy (%) on target clients under various distribution shifts on CIFAR-10 (CNN) and CIFAR-100 (ResNet-18). Best results in **bold**.

method consistently outperform other approaches, with the OTTA variant showing slightly better results, particularly in complex shift scenarios. These results demonstrate DynFed's effectiveness in handling various distribution shifts in FL environments. The substantial improvements over ATP, especially in challenging scenarios like multi-type shifts and on CIFAR-100, highlight our method's robustness and flexibility. DynFed not only advances the state-of-the-art in test-time adaptation for FL but also shows remarkable consistency across different shift types, underscoring its potential for real-world applications with unknown or mixed distribution shifts.

5.2.2 Hyperparameter analysis

In the component of ARN, There is a critical hyperparameter τ, because according to our experiments, we found that τ takes values in the opposite direction in multi-type distribution shifts versus one-type distribution shifts. Figure 3 illustrates the crucial role of the τ parameter in our Adaptive Rate Network (ARN) across different distribution shift scenarios. For multi-type distribution shifts (red line), we observe optimal performance at $\tau = 0.3$, with accuracy peaking at 62.33%. The performance rapidly declines as τ increases beyond this point, stabilizing at a lower level for τ >1. Conversely, for one-type distribution shifts, exemplified by feature shift (blue line), the model's accuracy improves as τ increases, reaching its optimal value of 68.27% at $\tau = 1.7$. This insightful performance remains stable for higher τ values. This contrasting behavior underscores the adaptive capability of our ARN, demonstrating its ability to effectively handle diverse distribution shift scenarios by appropriately tuning the τ parameter. The significant performance differences highlight the importance of dynamic adaptation in TTPFL environments with heterogeneous distribution shifts.

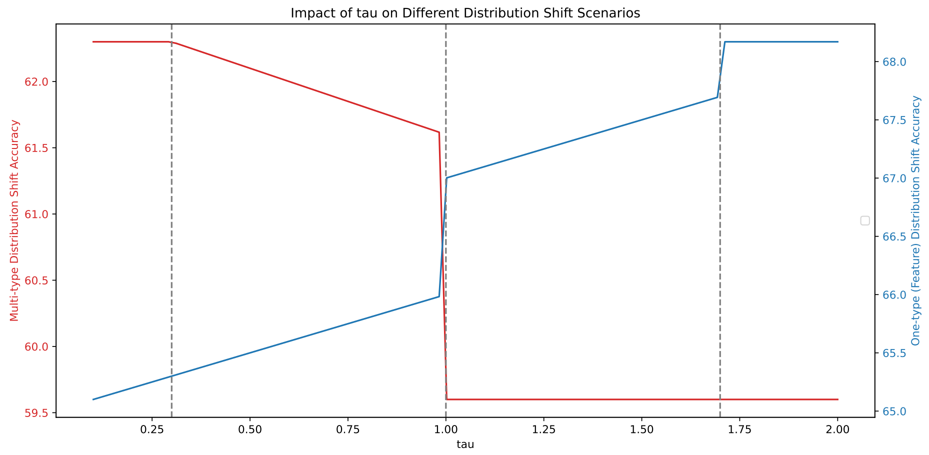
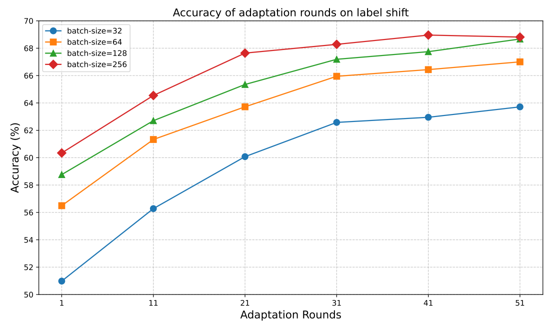
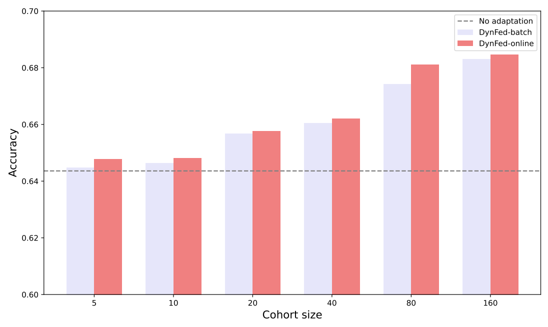


Figure 3: Impact of τ on model performance under different distribution shift scenarios. The red line represents accuracy for multi-type distribution shift, while the blue line shows accuracy for one-type distribution shift (feature shift).



(a) Impact of batch size on model accuracy across adaptation rounds for label shift scenarios.



(b) Impact of shard size on model accuracy for feature shift scenarios.

Figure 4: Effect of shard size and batch size.

5.2.3 Effect of batch size and shard size

Figure 4 illustrates the impact of batch size and shard size on our method's performance under different distribution shift scenarios. As shown in Figure 4a, larger batch sizes consistently yield higher accuracy in label shift conditions, with batch size 256 achieving the best performance. The performance gap between batch sizes narrows over time, suggesting diminishing returns for larger batches in later adaptation rounds. Figure 4b demonstrates that both DynFed variants significantly outperform the no-adaptation baseline across all shard sizes in feature shift scenarios, with DynFed-online showing superior performance, especially as shard size increases. These results highlight our method's adaptability and efficiency across various operational conditions, showcasing its robustness in diverse federated learning environments and its ability to effectively leverage larger data shards for improved adaptation.

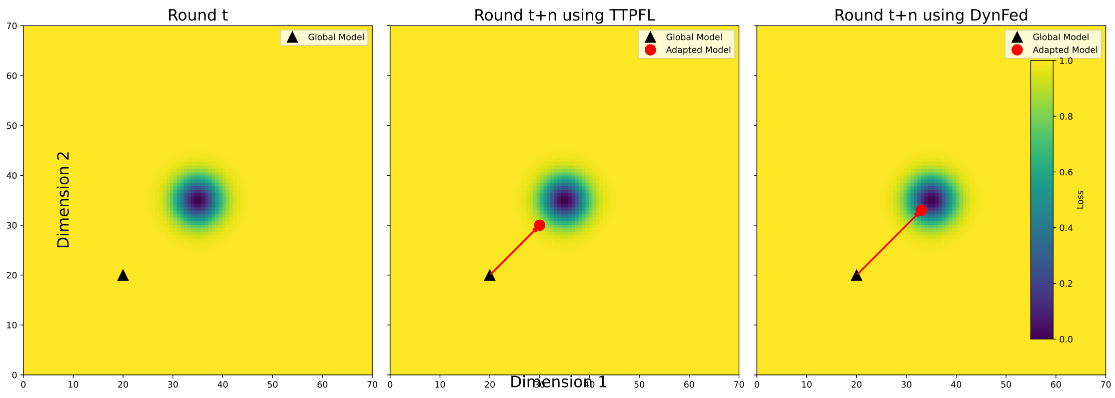


Figure 5: Optimization trajectories of TTPFL and DynFed in FL, projected onto two principal dimensions of adaptation rate changes. DynFed demonstrates superior convergence towards the optimal region (darker area) compared to TTPFL, illustrating its more effective adaptation to client-specific data distributions.

5.2.4 Optimization trajectories

Figure 5 illustrates the superiority of our proposed method over TTPFL in the context of FL. The figure shows the optimization trajectories in dimension space, the dimension capture the main directions of variation in the model parameters during the adaptation process. As observed, both methods start from the same initial global model position (round t). However, after n rounds of adaptation (comparison experiments conducted with our selected $n = 50$) , DynFed achieves a position closer to the optimal region (darker area) compared to TTPFL. This visual representation demonstrates DynFed's ability to more effectively navigate the loss landscape, resulting in a better-adapted model that more closely aligns with the client's specific data distribution.

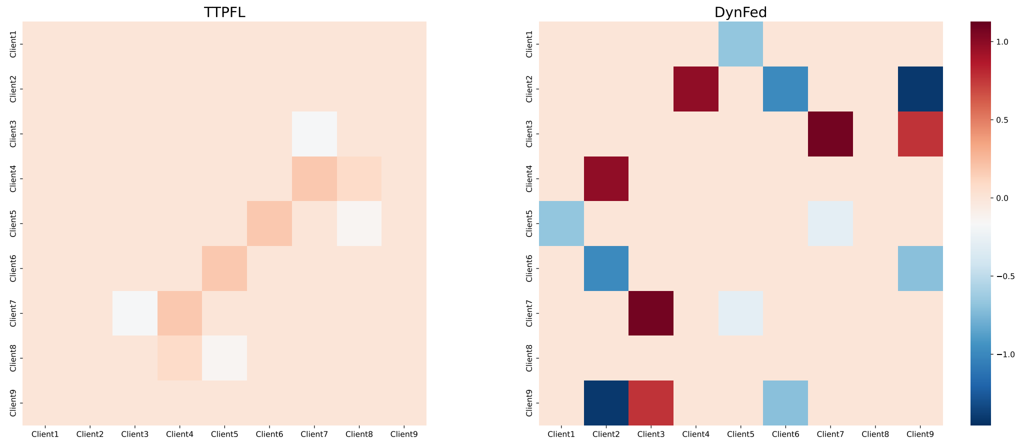


Figure 6: Heatmap comparison of adaptation rates between clients for TTPFL (left) and our proposed method (right).

Table 2: Accuracy (mean ± s.d. %) on target clients under hybrid shift on Digits-5 and PACS

Method	Digits-5					PACS			
	MNIST	SVHN	USPS	SynthD	MNIST-M	Art	Cartoon	Photo	Sketch
No adapt	95.5±0.2	52.3±1.5	89.6±0.4	79.8±0.7	55.6±0.8	71.6±1.2	74.7±0.7	90.3±0.8	74.2±0.7
BN-Adapt	94.9±0.3	57.6±0.5	89.5±0.4	75.3±0.5	59.7±0.4	73.6±0.5	71.5±0.6	92.1±0.3	70.9±0.5
SHOT	94.7±0.3	57.9±0.2	89.6±0.7	76.4±0.3	60.2±0.7	69.3±0.7	67.8±0.4	87.0±0.6	59.4±0.9
Tent	95.5±0.3	60.7±0.5	91.7±0.6	78.6±0.5	62.5±0.7	71.6±0.7	71.0±1.0	88.1±0.2	63.2±1.1
T3A	94.6±0.6	49.9±1.1	88.5±0.8	75.5±1.1	51.3±1.6	72.2±0.7	75.0±0.8	91.5±0.6	70.1±1.2
MEMO	95.9±0.2	52.9±1.1	89.8±0.4	80.1±0.9	55.5±1.1	71.5±1.3	75.6±1.0	90.7±0.9	76.3±0.7
EM	96.6±0.3	57.2±1.7	92.3±0.3	85.7±0.5	62.1±0.6	74.0±1.9	78.9±0.9	92.3±0.9	80.8±1.5
BBSE	94.5±0.6	57.3±1.5	91.3±0.4	85.5±0.5	61.6±0.9	74.3±1.8	78.7±1.0	91.8±0.7	80.2±1.4
Surgical	97.4±0.1	59.9±2.0	94.2±0.4	86.1±0.4	65.9±0.8	74.6±2.7	77.5±0.6	92.3±0.8	80.9±3.4
ATP	97.8±0.3	**62.2±1.7**	95.4±0.3	**87.9±0.5**	70.0±2.0	**82.9±1.0**	79.6±0.8	95.4±0.4	82.3±1.6
DynFed-B	97.7±0.2	61.9±1.7	**95.6±0.3**	87.8±0.5	68.9±1.8	81.9±1.1	**80.1±0.7**	94.9±0.4	82.0±1.6
DynFed-O	**97.9±0.2**	61.7±1.8	95.4±0.3	86.4±0.5	**71.1±2.1**	82.8±0.9	79.8±0.7	**95.7±0.4**	**82.6±1.5**

5.2.5 Performance on Domain Generalization Benchmarks

Table 2 presents evaluation results on two challenging domain generalization datasets with hybrid shifts. Digits-5 comprises five domains (MNIST, SVHN, USPS, SynthDigits, MNIST-M), while PACS includes four domains (Art, Cartoon, Photo, Sketch). Following standard protocol, we use leave-one-domain-out evaluation, where one domain serves as the testing target while the remaining form the source domains. Our DynFed method demonstrates strong performance across all domains, with DynFed-B achieving the best accuracy on USPS and Cartoon domains, while DynFed-O excels on MNIST, MNIST-M, Photo, and Sketch domains. These results highlight DynFed's effectiveness in adapting to complex real-world domain shifts, where it either matches or outperforms state-of-the-art methods. The consistent improvement of DynFed-O over DynFed-B across multiple domains validates the efficacy of our proposed cumulative averaged adaptation mechanism for online test-time adaptation in challenging federated scenarios.

5.2.6 Adaptive flexibility in client-specific batch data handling

In order to verify that our method can be efficiently processed for different types of data, we conducted experiments in multi-type shift distributions. The front 9 clients were selected for experiment. Our experimental results, as illustrated in Figure 6, demonstrate the superior adaptability of our proposed method compared to the traditional TTPFL approach. The heatmaps depict the adaptation rates between different clients. The TTPFL method (left) shows minimal variation in adaptation rates across clients, indicated by the predominantly uniform coloration. This uniformity suggests a lack of client-specific customization, potentially limiting its effectiveness in heterogeneous federated learning environments. In contrast, our method (right) exhibits a more diverse range of adaptation rates, as evidenced by the varied color intensities across the heatmap. This diversity indicates a higher degree of flexibility in adjusting to each client's unique batch data characteristics. The increased number of non-zero elements in our method's heatmap further underscores its enhanced capability to capture and respond to inter-client differences. Such adaptive behavior is crucial for effectively handling the diverse data distributions commonly encountered in real-world federated learning scenarios, enabling our method to achieve more personalized and efficient learning across varied client datasets. To further validate our method's effectiveness across different clients, we conducted an accuracy comparison using the random sampling 10 clients from cifar10 on the multi-type shift distribution. Figure 7 illustrates the performance of our framework compared to TTPFL for each of these clients. As shown in the bar chart, DynFed consistently outperforms TTPFL across all individual clients. The blue bars, representing DynFed, are consistently higher than the orange bars of TTPFL for each client. Moreover, the average accuracy of DynFed, indicated by the blue dashed line, is notably higher than that of TTPFL, represented by the orange dashed line. This comprehensive superiority, both in individual client performance and overall average accuracy, underscores the adaptability of our method in diverse federated learning scenarios.

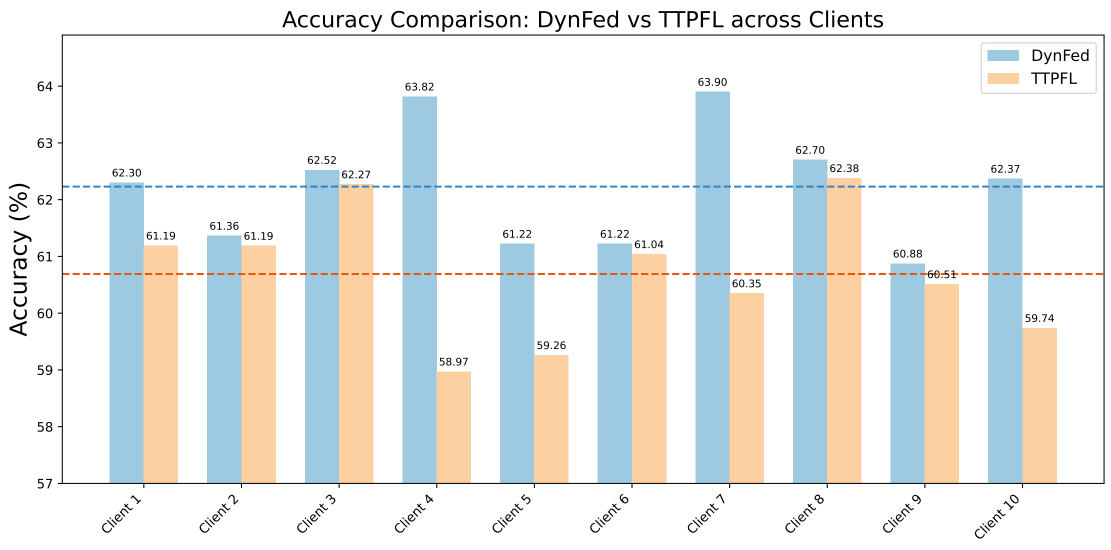


Figure 7: Accuracy comparison between clients for TTPFL and DynFed.

Method	Feature	Hybrid	MT
No adapt	64.36±0.13	63.68±0.24	57.71±0.20
DynFed w/o ARN	66.71±0.27	68.31±0.36	60.62±0.19
DynFed w/o opt. τ	65.23±0.23	65.69±0.37	57.75±0.21
DynFed-B	**68.12±0.18**	**69.56±0.31**	**62.23±0.17**

Table 3: Ablation study: accuracy (%) on target clients under various distribution shifts.

5.2.7 Ablation study

Ablation studies were conducted to evaluate the contribution of key components in our method, as shown in Table 3. Removing the ARN leads to a noticeable performance drop across all shift types, highlighting its importance in our approach. More strikingly, using ARN without the optimal τ value results in even lower performance than the variant without ARN, particularly in the multi-type distribution shift scenario where accuracy drops to 57.75%, barely surpassing the no-adaptation baseline. This underscores the critical role of τ in effectively leveraging ARN's capabilities. The full DynFed-B method, incorporating both ARN and optimal τ, consistently outperforms all variants, demonstrating the synergistic effect of these components in handling various unseen distribution shifts, with the most significant improvements observed in hybrid and multi-type distribution shift scenarios.

5.3 Limitations and Potential Failure Cases

While DynFed demonstrates superior performance across various distribution shift scenarios, it is important to acknowledge its limitations and potential failure cases:

5.3.1 Extreme Heterogeneity

In scenarios with extremely high degrees of heterogeneity where each client experiences a unique combination of multiple distribution shift types, the ARN may struggle to generate optimal adaptation rates. The adaptation strategy dichotomy we identified (Theorem 3.3) suggests that when faced with highly diverse shift combinations, the ARN might converge to compromise solutions that are suboptimal for individual clients.

5.3.2 Computational Overhead

DynFed introduces additional computational complexity through the ARN component. For resource-constrained edge devices with limited computational capabilities, the overhead of running the ARN might outweigh the benefits in adaptation quality. This limitation is particularly relevant for IoT applications where energy efficiency is paramount.

5.4 Conclusion

In this paper, we introduced DynFed, a novel approach for test-time adaptation in federated learning that effectively addresses the challenges of heterogeneous distribution shifts across clients. Our method leverages an ARN with an optimized τ parameter to generate client-specific adaptation rates, significantly outperforming existing methods across various shift scenarios, including one-type shift and multi-type shifts. Through extensive experiments and ablation studies, we demonstrated the crucial role of both ARN and the optimal τ in achieving superior performance, particularly in complex multi-type shift environments. Our work offers a robust and adaptive federated learning in real-world applications where distribution shifts are diverse and unpredictable.

References

Jinheon Baek, Wonyong Jeong, Jiongdao Jin, Jaehong Yoon, and Sung Ju Hwang. Personalized subgraph federated learning. In *International conference on machine learning*, pp. 1396–1415. PMLR, 2023.

Wenxuan Bao, Haohan Wang, Jun Wu, and Jingrui He. Optimizing the collaboration structure in cross-silo federated learning. In *International Conference on Machine Learning*, pp. 1718–1736. PMLR, 2023.

Wenxuan Bao, Tianxin Wei, Haohan Wang, and Jingrui He. Adaptive test-time personalization for federated learning. *Advances in Neural Information Processing Systems*, 36, 2024.

Mikhail Belkin, Siyuan Ma, and Soumik Mandal. To understand deep learning we need to understand kernel learning. In *International Conference on Machine Learning*, pp. 541–549. PMLR, 2018.

Dian Chen, Dequan Wang, Trevor Darrell, and Sayna Ebrahimi. Contrastive test-time adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 295–305, 2022.

Hong-You Chen and Wei-Lun Chao. Fedbe: Making bayesian model ensemble applicable to federated learning. *arXiv preprint arXiv:2009.01974*, 2020.

Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive personalized federated learning. *arXiv preprint arXiv:2003.13461*, 2020.

Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. *Advances in neural information processing systems*, 33:3557–3568, 2020.

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

Avishek Ghosh, Justin Hong, Dong Yin, and Kannan Ramchandran. Robust federated learning in a heterogeneous environment. *arXiv preprint arXiv:1906.06629*, 2019.

Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *International conference on machine learning*, pp. 1225–1234. PMLR, 2016.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

Yusuke Iwasawa and Yutaka Matsuo. Test-time classifier adjustment module for model-agnostic domain generalization. *Advances in Neural Information Processing Systems*, 34:2427–2440, 2021.

Liangze Jiang and Tao Lin. Test-time robust personalization for federated learning. In *International Conference on Learning Representations (ICLR)*, 2023.

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

Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, pp. 896. Atlanta, 2013.

Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International conference on machine learning*, pp. 6028–6039. PMLR, 2020.

Jian Liang, Ran He, and Tieniu Tan. A comprehensive survey on test-time adaptation under distribution shifts. *International Journal of Computer Vision*, pp. 1–34, 2024.

Mi Luo, Fei Chen, Dapeng Hu, Yifan Zhang, Jian Liang, and Jiashi Feng. No fear of heterogeneity: Classifier calibration for federated learning with non-iid data. *Advances in Neural Information Processing Systems*, 34:5972–5984, 2021.

Jie Ma, Guodong Long, Tianyi Zhou, Jing Jiang, and Chengqi Zhang. On the convergence of clustered federated learning. *arXiv preprint arXiv:2202.06187*, 2022.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pp. 1273–1282. PMLR, 2017.

Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. Agnostic federated learning. In *International conference on machine learning*, pp. 4615–4625. PMLR, 2019.

Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Zhiquan Wen, Yaofo Chen, Peilin Zhao, and Mingkui Tan. Towards stable test-time adaptation in dynamic wild world. *arXiv preprint arXiv:2302.12400*, 2023.

Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Improving robustness against common corruptions by covariate shift adaptation. *Advances in neural information processing systems*, 33:11539–11551, 2020.

Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. Towards personalized federated learning. *IEEE transactions on neural networks and learning systems*, 34(12):9587–9603, 2022a.

Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. Fedproto: Federated prototype learning across heterogeneous clients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 8432–8440, 2022b.

Yue Tan, Yixin Liu, Guodong Long, Jing Jiang, Qinghua Lu, and Chengqi Zhang. Federated learning on non-iid graphs via structural knowledge sharing. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 9953–9961, 2023.

Yue Tan, Chen Chen, Weiming Zhuang, Xin Dong, Lingjuan Lyu, and Guodong Long. Is heterogeneity notorious? taming heterogeneity to handle test-time shift in federated learning. *Advances in Neural Information Processing Systems*, 36, 2024.

Guancheng Wan, Wenke Huang, and Mang Ye. Federated graph learning under domain shift with generalizable prototypes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 15429–15437, 2024.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations*, 2021a. URL `https://openreview.net/forum?id=uXl3bZLkr3c`.

Haozhao Wang, Zhihao Qu, Song Guo, Ningqi Wang, Ruixuan Li, and Weihua Zhuang. Losp: Overlap synchronization parallel with local compensation for fast distributed training. *IEEE Journal on Selected Areas in Communications*, 39(8):2541–2557, 2021b.

Haozhao Wang, Yichen Li, Wenchao Xu, Ruixuan Li, Yufeng Zhan, and Zhigang Zeng. Dafkd: Domain-aware federated knowledge distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 20412–20421, 2023.

Haozhao Wang, Yabo Jia, Meng Zhang, Qinghao Hu, Hao Ren, Peng Sun, Yonggang Wen, and Tianwei Zhang. Feddse: Distribution-aware sub-model extraction for federated learning over resource-constrained devices. In *Proceedings of the ACM on Web Conference 2024*, pp. 2902–2913, 2024a.

Haozhao Wang, Haoran Xu, Yichen Li, Yuan Xu, Ruixuan Li, and Tianwei Zhang. Fedcda: Federated learning with cross-rounds divergence-aware aggregation. In *The Twelfth International Conference on Learning Representations*, 2024b.

Haozhao Wang, Peirong Zheng, Xingshuo Han, Wenchao Xu, Ruixuan Li, and Tianwei Zhang. Fednlr: Federated learning with neuron-wise learning rates. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 3069–3080, 2024c.

Qin Wang, Olga Fink, Luc Van Gool, and Dengxin Dai. Continual test-time domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7201–7211, 2022.

Honglin Yuan, Warren Morningstar, Lin Ning, and Karan Singhal. What do we mean by generalization in federated learning? *arXiv preprint arXiv:2110.14216*, 2021.

Dun Zeng, Zenglin Xu, Yu Pan, Qifan Wang, and Xiaoying Tang. Tackling hybrid heterogeneity on federated optimization via gradient diversity maximization. *arXiv preprint arXiv:2310.02702*, 2023.

Marvin Zhang, Sergey Levine, and Chelsea Finn. Memo: Test time robustness via adaptation and augmentation. *Advances in neural information processing systems*, 35:38629–38642, 2022.

Xinwei Zhang, Wotao Yin, Mingyi Hong, and Tianyi Chen. Hybrid federated learning for feature & sample heterogeneity: Algorithms and implementation. *Transactions on Machine Learning Research*.

Yifei Zhang, Dun Zeng, Jinglong Luo, Xinyu Fu, Guanzhong Chen, Zenglin Xu, and Irwin King. A survey of trustworthy federated learning: Issues, solutions, and challenges. *ACM Transactions on Intelligent Systems and Technology*, 2024.

Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

Appendix: Theoretical Analysis

In this section, we provide theoretical justification for DynFed by analyzing its convergence properties and generalization guarantees. We show that despite having client-specific adaptation mechanisms, DynFed maintains both convergence and generalization capabilities comparable to traditional federated learning methods. Moreover, we demonstrate why different adaptation strategies are necessary for different types of distribution shifts.

Convergence Analysis

We first analyze the convergence properties of DynFed to demonstrate that our dynamic adaptation approach converges despite the additional complexity of client-specific, adaptive rates. Similar to previous work in federated learning McMahan et al. (2017), we establish convergence guarantees under standard assumptions. In particular, we consider the objective function of DynFed as minimizing the expected loss over the client population after adaptation:

$$\mathcal{L}(\mathbf{w}_G, \{\alpha_j\}_{j=1}^M) = \frac{1}{M}\sum_{j=1}^{M} \mathcal{L}_{P_j}(\mathbf{w}_G, \alpha_j), \tag{17}$$

where $\mathbf{w}_G$ represents the global model parameters and $\{\alpha_j\}_{j=1}^M$ denotes the set of adaptation rates for M clients. Here, $\mathcal{L}_{P_j}(\mathbf{w}_G, \alpha_j)$ is the loss for client j after adaptation.

To analyze convergence, we define the following Lyapunov function that tracks the distance from the current parameters to the optimal ones:

$$V_t = \|\mathbf{w}_G^t - \mathbf{w}_G^*\|^2 + \gamma \sum_{j=1}^{M} \|\alpha_j^t - \alpha_j^*\|^2, \tag{18}$$

where $\mathbf{w}_G^*$ and α_j^* are the optimal parameters, and $\gamma > 0$ is a constant balancing the two terms.

[Proof of Theorem 4.4.1] By the smoothness assumption, we have

$$\mathbb{E}[\mathcal{L}(\mathbf{w}_G^{t+1}, \{\alpha_j^{t+1}\}_{j=1}^M)] \leq \mathbb{E}[\mathcal{L}(\mathbf{w}_G^t, \{\alpha_j^t\}_{j=1}^M)] - \frac{\eta_w}{2}\mathbb{E}[\|\nabla_{\mathbf{w}_G}\mathcal{L}(\mathbf{w}_G^t, \{\alpha_j^t\}_{j=1}^M)\|^2] \tag{19}$$

$$- \frac{\eta_\alpha}{2}\sum_{j=1}^{M}\mathbb{E}[\|\nabla_{\alpha_j}\mathcal{L}(\mathbf{w}_G^t, \{\alpha_j^t\}_{j=1}^M)\|^2] + \frac{\beta}{2}\eta_w^2\sigma_w^2 + \frac{\beta}{2}\eta_\alpha^2\sigma_\alpha^2 \tag{20}$$

Using this inequality and the definition of the Lyapunov function, we can show that

$$\mathbb{E}[V_{t+1}] - \mathbb{E}[V_t] \leq -\eta_w \mathbb{E}[\langle \nabla_{\mathbf{w}_G}\mathcal{L}(\mathbf{w}_G^t, \{\alpha_j^t\}_{j=1}^M), \mathbf{w}_G^t - \mathbf{w}_G^*\rangle] \tag{21}$$

$$- \gamma\eta_\alpha \sum_{j=1}^{M}\mathbb{E}[\langle \nabla_{\alpha_j}\mathcal{L}(\mathbf{w}_G^t, \{\alpha_j^t\}_{j=1}^M), \alpha_j^t - \alpha_j^*\rangle] \tag{22}$$

$$+ \frac{\eta_w^2}{2}\sigma_w^2 + \frac{\gamma\eta_\alpha^2}{2}\sigma_\alpha^2 \tag{23}$$

By convexity, we have

$$\langle \nabla f(x), x-y \rangle \geq f(x) - f(y) \geq \frac{1}{2}\|\nabla f(x)\|^2 \tag{24}$$

for any convex function f. Applying this to our case, we get

$$\mathbb{E}[V_{t+1}] - \mathbb{E}[V_t] \leq -\frac{\eta_w}{2}\|\nabla_{\mathbf{w}_G}\mathcal{L}(\mathbf{w}_G^t, \{\alpha_j^t\}_{j=1}^M)\|^2 - \frac{\gamma\eta_\alpha}{2}\sum_{j=1}^{M}\|\nabla_{\alpha_j}\mathcal{L}(\mathbf{w}_G^t, \{\alpha_j^t\}_{j=1}^M)\|^2 \tag{25}$$

$$+ \frac{\eta_w^2}{2}\sigma_w^2 + \frac{\gamma\eta_\alpha^2}{2}\sigma_\alpha^2 \tag{26}$$

Summing up from $t = 0$ to $T-1$, we get

$$\mathbb{E}[V_T] - \mathbb{E}[V_0] \leq -\sum_{t=0}^{T-1}\left(\frac{\eta_w}{2}\|\nabla_{\mathbf{w}_G}\mathcal{L}(\mathbf{w}_G^t, \{\alpha_j^t\}_{j=1}^M)\|^2 + \frac{\gamma\eta_\alpha}{2}\sum_{j=1}^{M}\|\nabla_{\alpha_j}\mathcal{L}(\mathbf{w}_G^t, \{\alpha_j^t\}_{j=1}^M)\|^2\right) \tag{27}$$

$$+ \frac{T\eta_w^2}{2}\sigma_w^2 + \frac{T\gamma\eta_\alpha^2}{2}\sigma_\alpha^2 \tag{28}$$

Setting $C = \frac{T\eta_w^2}{2}\sigma_w^2 + \frac{T\gamma\eta_\alpha^2}{2}\sigma_\alpha^2$ completes the proof.

Generalization Guarantees

Next, we analyze the generalization properties of DynFed. Unlike standard federated learning, our method introduces dynamically generated adaptation rates. However, we show that DynFed enjoys good generalization guarantees due to the controlled dimensionality of the adaptation parameters.

[Proof of Theorem 4.4.2] We use covering number techniques from learning theory. For any fixed global model $\mathbf{w}_G$, the generalization error of adaptation rates can be bounded using uniform convergence bounds. Let $\mathcal{B}_R = \{\alpha : \|\alpha\|_2 \leq R\}$ be the ball of radius R in the adaptation rate space.

The covering number of $\mathcal{B}_R$ with balls of radius $\epsilon/4LH$ is bounded by $(\frac{12LHR}{\epsilon})^d$, where d is the dimension of α.

Let $\alpha_1, \ldots, \alpha_N$ be the centers of these covering balls. For any $\alpha \in \mathcal{B}_R$, there exists an α_i such that $\|\alpha - \alpha_i\|_2 \leq \epsilon/4LH$. By Lipschitz continuity of the model and bounded update directions, this implies $|\varepsilon(\alpha) - \varepsilon(\alpha_i)| \leq \epsilon/4$ and $|\hat{\varepsilon}(\alpha) - \hat{\varepsilon}(\alpha_i)| \leq \epsilon/4$.

By Hoeffding's inequality and the fact that we have NK samples (across N clients, each with K batches), for any fixed α_i,

$$\Pr(|\varepsilon(\alpha_i) - \hat{\varepsilon}(\alpha_i)| \geq \epsilon/2) \leq 2\exp\left(-\frac{NK\epsilon^2}{2(b-a)^2}\right) \tag{29}$$

where $b-a$ is the range of the loss function, which we can bound by $\sqrt{K}+1$ due to the averaging of K batches.

Applying the union bound over all α_i, we get

$$\Pr\left(\max_i |\varepsilon(\alpha_i) - \hat{\varepsilon}(\alpha_i)| \geq \epsilon/2\right) \leq 2\left(\frac{12LHR}{\epsilon}\right)^d \exp\left(-\frac{NK\epsilon^2}{2(\sqrt{K}+1)^2}\right) \tag{30}$$

For any $\alpha \in \mathcal{B}_R$, using the triangle inequality,

$$|\varepsilon(\alpha) - \hat{\varepsilon}(\alpha)| \leq |\varepsilon(\alpha) - \varepsilon(\alpha_i)| + |\varepsilon(\alpha_i) - \hat{\varepsilon}(\alpha_i)| + |\hat{\varepsilon}(\alpha_i) - \hat{\varepsilon}(\alpha)| \tag{31}$$

$$\leq \epsilon/4 + \epsilon/2 + \epsilon/4 = \epsilon \tag{32}$$

with probability at least $1 - 2\left(\frac{12LHR}{\epsilon}\right)^d \exp\left(-\frac{NK\epsilon^2}{2(\sqrt{K}+1)^2}\right)$.

Simplifying, we get the desired bound.

Adaptive Mechanisms for Different Distribution Shifts

Here, we theoretically explain why different types of distribution shifts require different adaptation strategies, which forms the foundation for our dynamic adaptation approach. In federated learning, two common types of shifts are:

- **Covariate (Feature) Shift**: Clients have different input distributions $P(X)$ while the conditional distribution $P(Y|X)$ remains unchanged.
- **Label Shift**: Clients have different label distributions $P(Y)$ while the feature distribution given a label $P(X|Y)$ remains the same.

For covariate shift, the main challenge is adapting the feature extractor to the new input distribution. The optimal strategy is to modify the early layers (or Batch Normalization statistics) to realign the feature representations. Conversely, under label shift, the feature extractor remains valid, and only the final classifier requires adjustment (typically by calibrating the output layer bias). Hence, the optimal adaptation rates for these two scenarios differ significantly.

[Proof of Proposition 4.4.3] Assume without loss of generality that the last layer is linear. Let $g(\mathbf{x}; \mathbf{w}_g)$ denote the features extracted by the network, and let $\mathbf{w}_1, \ldots, \mathbf{w}_K$ be the weights with biases $b_1, \ldots, b_K$ for K classes. Then,

$$f(\mathbf{x}; \mathbf{w})_c = \frac{\exp(\mathbf{w}_c^\top g(\mathbf{x}; \mathbf{w}_g) + b_c)}{\sum_{c'=1}^{K} \exp(\mathbf{w}_{c'}^\top g(\mathbf{x}; \mathbf{w}_g) + b_{c'})}.$$

Since the network is calibrated on p, we have $f(\mathbf{x}; \mathbf{w})_c = p(\mathbf{y} = \mathbf{e}_c|\mathbf{x})$. By Bayes' theorem, for distribution q,

$$q(\mathbf{y}|\mathbf{x}) = \frac{p(\mathbf{y}|\mathbf{x}) \cdot \frac{q(\mathbf{y})}{p(\mathbf{y})}}{\sum_{\mathbf{y}} p(\mathbf{y}|\mathbf{x}) \cdot \frac{q(\mathbf{y})}{p(\mathbf{y})}}.$$

Thus, adding $\log \frac{q(\mathbf{y})}{p(\mathbf{y})}$ to each bias term effectively calibrates the network on q.

[Proof of Proposition 4.4.3] Let the source and target feature distributions be $p(\mathbf{x})$ and $q(\mathbf{x})$, respectively. Since the shift alters only the mean and variance, there exist constants Δ and $r > 0$ such that for any threshold z_t,

$$\Pr_q(z \geq z_t) = \Pr_p\left(z \geq \frac{z_t - \Delta}{r}\right).$$

Thus, updating the BN layer's running mean and variance as

$$\mu_q = r \cdot \mu_p + \Delta, \quad \sigma_q = r \cdot \sigma_p,$$

aligns the target feature distribution with the source distribution, effectively removing the shift.

Next, we analyze the theoretical foundation for our core insight: that optimal adaptation strategies for one-type and multi-type distribution shifts are fundamentally different.

[Proof of Theorem 3.3] Consider the loss landscapes corresponding to the two scenarios. For one-type shifts, since all clients experience the same type of bias, the gradients with respect to the adaptation rates,

$$\nabla_\alpha \mathcal{L}_{\mathcal{S}}(\alpha) = \frac{1}{N} \sum_{i=1}^{N} \nabla_\alpha \mathcal{L}_{\mathcal{S}_i}(\alpha),$$

tend to align, leading to an optimal adaptation rate $\alpha_{\mathcal{S}}^*$ in a common direction. For multi-type shifts, the clients experience conflicting adaptation needs; for instance, some require increasing the learning rate for early layers (to adjust feature extractors) while others require a decrease (to recalibrate the output layer). Thus, the aggregated gradient

$$\nabla_\alpha \mathcal{L}_{\mathcal{M}}(\alpha) = \frac{1}{N} \sum_{i=1}^{N} \nabla_\alpha \mathcal{L}_{\mathcal{M}_i}(\alpha)$$

points in a direction nearly opposite to $\nabla_\alpha \mathcal{L}_\mathcal{S}(\alpha)$. Consequently, by strong convexity of the loss functions, the optimal solutions satisfy

$$\alpha^*_\mathcal{S} \cdot \alpha^*_\mathcal{M} < 0.$$

This negative correlation confirms that different types of distribution shifts necessitate fundamentally different adaptation strategies.

This theorem underpins the design of our dynamic adaptation mechanism: the self-adaptive rate network (ARN) must generate different learning rate values (and even opposite adjustment directions) depending on whether the encountered shift is one-type or multi-type. Empirically, we observe that the optimal scaling factor τ for one-type shifts is typically higher than that for multi-type shifts, validating this theoretical insight.

Finally, we extend our discussion on the effect of learning rate selection on generalization. The choice of learning rates not only impacts convergence but also plays a critical role in the stability and generalization of the model. In classical optimization, a larger learning rate can speed up convergence but may harm stability, whereas a small learning rate tends to improve stability and thus generalization Hardt et al. (2016). In DynFed, the ARN dynamically adjusts the learning rate based on the observed gradient information, achieving an automatic balance between exploration (rapid adaptation) and convergence (stability in later iterations). Such dynamic scheduling effectively reduces the sensitivity of the final model to variations in the training data, thereby improving generalization performance. This insight is supported by theoretical results on algorithmic stability which show that a controlled update step leads to lower generalization error Hardt et al. (2016); Belkin et al. (2018).