Atlas: Ensuring Accuracy for Privacy-Preserving Federated IoT Applications

Jiechao Gao[†]
University of Virginia
Charlottesville, VA, USA
jg5ycn@virginia.edu

Mingyue Tang[†]
University of Illinois
Urbana-Champaign
Champaign, IL, USA
mt55@illinois.edu

Wenpeng Wang
University of Virginia
Charlottesville, VA, USA
ww2cg@virginia.edu

Tushar Routh
University of Virginia
Charlottesville, VA, USA
tkr2w@virginia.edu

Bradford Campbell
University of Virginia
Charlottesville, VA, USA
bradjc@virginia.edu

Abstract

In smart Internet of Things (IoT) applications, edge devices often collect and store limited data, which is insufficient for training modern deep learning models. Collaborative training methods like cloud computing and federated learning enable robust models for IoT applications, yet introduce data privacy concerns due to central data collection and model inversion attacks. Remedies such as differential privacy can bring data privacy protection but dramatically degrade the accuracy performance of IoT applications. To safeguard user data privacy while maintaining application quality, it is imperative to establish a framework capable of preserving user privacy without compromising accuracy standards.

In this paper, we present `Atlas`, a private and accurate personalized federated local differential privacy (LDP) framework for IoT applications. We first design a layer-sharing mechanism called the layer importance mask to separate the local model into global and personalized layers. Second, we design a weighted LDP mechanism and add noise to the global layers before transmitting them to the federated learning framework for aggregation. Third, we combine local personalized layers and aggregated global layers to perform IoT tasks. Our experiments on five real-world IoT application datasets and the CIFAR-10 dataset show that our privacy-preserving approach only sacrifices 2% to 6% of accuracy compared to the state-of-the-art non-privacy preserving FL frameworks among various IoT applications and outperforms the current LDP-based FL framework by 8% to 13%.

CCS Concepts

• **Security and privacy** → **Privacy protections**; • **Computer systems organization** → *Sensor networks.*

Keywords

Federated Learning, IoT, Privacy-Preserving

ACM Reference Format:
Jiechao Gao[†], Mingyue Tang[†], Wenpeng Wang, Tushar Routh, and Bradford Campbell. 2025. Atlas: Ensuring Accuracy for Privacy-Preserving Federated IoT Applications. In *ACM/IEEE 16th International Conference on Cyber-Physical Systems (with CPS-IoT Week 2025) (ICCPS '25), May 6–9, 2025, Irvine, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3716550.3722017

1 Introduction

The Internet of Things (IoT) technologies enable a large number of applications that involve a rich set of sensors, such as energy consumption prediction [2], traffic forecasting [13], health monitoring [9], and human activity recognition (HAR) [4]. The key to enabling these IoT applications is machine learning or deep learning models trained on a massive amount of data. However, the edge devices of IoT applications often have access to only limited data, which is not sufficient for modern deep learning model training. The common solution is to aggregate all clients' data in a central server and train a global model using all available data, which raises problems with data privacy, sub-optimal accuracy performance and communication latency. Given the privacy-sensitive, data heterogeneity and latency-sensitive nature of IoT sensing data (e.g., medical data for health monitoring, activity data for HAR), we need to have a privacy-preserving, personalized and communication efficiency framework for smart and reliable IoT applications.

To address the intertwined challenges of data scarcity, data privacy, and communication latency in IoT applications, federated learning (FL) provides a popular method to achieve distributed machine learning among numerous devices without sharing raw data with a central server for collaborative training [22], FL has been widely applied in the IoT domain [25]. Most state-of-the-art FL frameworks share and aggregate the weights of the models trained on sensitive client data instead of the raw data directly [18, 22]. Unfortunately, FL still cannot guarantee sufficient privacy for sensitive data. Recent works have demonstrated that shared weights also can leak sensitive information by model inversion during the process of model updating [7, 24].

Status Quo and their Limitations.

[†] Both authors contributed equally to this paper.

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.
ICCPS '25, Irvine, CA, USA
© 2025 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1498-6/2025/05
https://doi.org/10.1145/3716550.3722017

(1) **Local differential privacy mechanisms.** As the practice of utilizing sensitive user data to train deep learning models has amplified privacy worries across various sectors, recent research [30] has shown that using FL alone can not meet the requirements of data privacy due to model inversion. Incorporating local differential privacy into federated learning serves as an effective method to ensure stringent privacy protections. However, existing LDP methods [20, 26, 39] add the same LDP noise to all the model layers for simplicity which is not practical, such methods ignore the fact that the model weights from different layers vary significantly. Presuming that the weights across all layers fall within a constant range can lead to significant fluctuations in the estimated model weights, ultimately resulting in suboptimal model performance.

(2) **Federated learning frameworks.** There are many existing personalized FL models using different technologies like adding user context [21], transfer learning [21], multi-task learning [32], meta-learning [14]. All the sets of personalized methods aim to solve the problem of data heterogeneity and achieve higher performance for individual users. The common personalized FL can be divided into two main categories: (1) model enhancement based federated learning personalization models [16, 29]. (2) transfer learning based federated learning personalization models [5, 14, 21]. However, the existing federated personalization solutions are not the perfect partner for the differential privacy setting. For model enhancement based federated learning personalization models, such as ClusterFL [29], the main concept is to enhance model training accuracy by identifying data relationships across nodes by clustering the clients in the central server. It expedites convergence and maintains accuracy by efficiently removing slower or less correlated nodes within each cluster. Yet, such a method requires high-quality model information (e.g., model gradient) for clustering based centralized aggregation. Directly adding LDP noise such as the Gaussian mechanism to such models will cause a dramatic performance reduction [10].

Transfer learning-based personalization methods aim to divide the model into two parts, the sharable layers for global aggregation and the personalized layers for local updates. Such a setup is more compatible with the FL-LDP setting since the differential privacy noises are added into each layer which does not require high-quality gradients to train the whole model. But existing transfer learning types of FL personalization methods, such as FedPer [5], and Reptile [14], cannot optimize the model performance since their sharable layers are simply fixed. Practically, the layers' importance for global aggregation and local updates are dynamically changing over time. Models for each client do not know which part of the layers can contribute more to the global model and which part of the layers is more helpful to the local models for each iteration [11].

(3) **Uniqueness of IoT data.** While Federated Learning (FL) holds substantial importance in solving data privacy for IoT application collaborative training, the majority of FL research primarily utilizes well-established datasets like CIFAR-10, MNIST and CIFAR-100 for evaluation. However, these datasets don't reflect the authentic characteristics and complexities of real-world IoT data, which fails to prove their practicality of IoT applications.

Overview of Proposed Approach. Motivated by the limitations of existing solutions, we present `Atlas`, a privacy-first practical personalized federated learning framework with dynamic local differential privacy designed specifically to resolve the above challenges simultaneously. `Atlas` aims to provide a unified privacy-preserving FL framework for IoT applications to (1) collaborative training deep learning models with different clients with privacy protection, (2) learn a personalized model for each participating client with optimal performance under the privacy protection. (3) achieve reasonable communication costs for IoT applications.

The core idea of `Atlas` can be split into two parts. First, we design a dynamic layer importance selection and sharing framework based on a self-learned layer importance mask mechanism for each client. The layer importance mask is refined along with the model parameters during the FL training process, which customizes the global and local layers according to the characteristics of each layer in every client. During the training process, the layer importance mask is randomly initialized and interacts with model layer weights to calculate the global importance of each layer during updating, the layers with higher importance scores will contribute to the global model training while the layers with lower importance scores will be pruned and contribute to local model training [35]. Note that since we only update selected layers to the central server, the communication cost can also be minimized during this process. Second, we add LDP to globally important layers. Instead of adding the same LDP noise to all selected model layers, we utilize the layer importance mask from the first step to add different noise for different selected layers and propose a more dynamic and practical LDP mechanism to demonstrate how the range influences the variance in the model weights.

System Implementation and Performance, we developed the `Atlas` system and performed extensive experiments using real IoT-related application data. The model performance is evaluated specifically on five IoT tasks, We compare `Atlas` with 8 popular baseline functions in the personalized FL and LDP domain and achieved comparable model performance with non-privacy-preserving methods with privacy protection. Compared to other state-of-the-art FL methods, `Atlas` demonstrates the personalization power of our proposed dynamic layer selection strategy with LDP, which effectively resolves the limitations of the existing approaches. More specifically, Table 1 shows the comparison between `Atlas` and status quo methods. `Atlas` improves the model utility compared to LDP-FL by using a novel layer selection personalization method, while other personalized FL frameworks like FedMask, are hard to train along with differential privacy. Compared to FedPer and other local fine-tuning methods, our proposed global layer selection technique optimized the model structure for both global training and local fine-tuning. In addition, since we only update selected layer parameters to the central server in `Atlas`, the communication cost can be reduced compared to traditional FL frameworks like FedAvg and ClusterFL.

2 Related Work

Federated Learning and Personalization Numerous studies from a wide range of research have explored the usage of FL methods from various perspectives. Some recent works in the FL domain have focused on the usage of IoT devices. Li et al., [17] presented an overview of challenges, open problems, and issues associated with FL by considering the heterogeneity of devices while assuming

Table 1: Comparing Atlas with existing FL approaches in IoT scenario.

Approach	Personalization	Privacy-Preserving	Communication
FedAvg [22]	✗	✗	✗
ClusterFL [29]	✔	✗	✗
FedPer [5]	✔	✗	✔
FedDL [35]	✔	✗	✗
FedMask [16]	✔	✗	✔
LDP-FL [33]	✗	✔	✗
DP-rec-FL [34]	✗	✔	✔
Atlas	✔	✔	✔

all clients are resource-boundless. Yang et al., [38] focused on the FL settings categorization, Niknam et al., [27] elaborated on the issues of FL setup in a wireless environment. Adapting a global model can help achieve personalization. Existing works in this domain mainly use two separate steps that result in extra overhead. With first having the global model learned in a federated fashion, then fine-tuning the global model for each client using its local data. The common personalized FL can be divided into two main categories: (1) model enhancement based federated learning personalization models [16, 29]. (2) transfer learning based federated learning personalization models [5, 14, 21].

Differential Privacy Differential privacy has been widely used for IoT and distributed learning systems. Shokri et al., [31] presents a distributed learning system using local differential privacy without a central trusted party. However, it is only able to work on models with a small number of parameters due to its DP guarantee being per parameter. Wei et al., [36] proposed a new framework of differential privacy adding artificial noise on the client side, and discussed several key properties for differential privacy. Similar to most of the existing LDP methods [20, 26, 39] They add the same differential noise to all model layers, which fail to consider the range difference of model weights thus resulting in suboptimal model performance.

3 System Design

3.1 System Overview

In this paper, we present Atlas, a privacy-preserving, personalized and communication-efficient federated local differential privacy framework to achieve private and accurate IoT applications. Figure 1 depicts the overview of our Atlas framework.

In the initial phase, the central server sets up a uniform model structure for every client involved in the system. Throughout the local training iterations, we integrate a network layer importance mask to cultivate a personalized model, taking into account the significance of various layers in the deep learning model explicitly. The self-learned layer importance mask selects the layers with higher importance scores that are labeled as global layers and will be transmitted to the central server for aggregation ①. For all the selected layers, we add the weight-enhanced Gaussian Local differential privacy (GLDP) noise for each local client, taking into consideration the varying impacts of weights in each layer on the

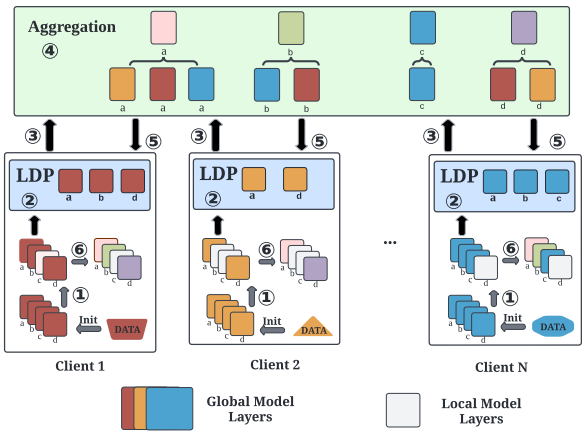


Figure 1: The Overview of Atlas Framework

model's performance ②. The selected layers with LDP noise together with the layers' importance score will be transmitted to the central server for aggregation within a random time slot T to to break the linkage among the model weight updates from the same clients and to mix them among updates from other clients ③. The central server then aggregates each layer based on the available layer number and sends back the updated layers to each client ④ ⑤. Each client generates their own personalized local model for IoT applications ⑥. By evaluating the impact and significance of weights in different model layers on the model's own performance, Atlas is able to achieve optimal model personalization. Even after incorporating local differential privacy (LDP), it still manages to deliver performance comparable to non-private methods. Additionally, by filtering the importance of model layers, we only propagate a certain number of layers in the model, which can reduce communication costs.

3.2 Model Layer Selection and Layer Importance Mask

What distinguishes Atlas from traditional approaches is that Atlas focuses on evaluating the significance of each layer in contributing to the model's overall effectiveness, and using this evaluation to determine which layers should be transmitted to the central server. An intuitive thought on the various ranges of weights of different deep learning model layers is that *the larger the weight is, the more important the layer is.* Practically, it's not always the truth. The relationship between the magnitude of a weight and its importance may not always be direct. In some cases, a larger weight might mean that it has a greater impact on the model's output, which intuitively could be interpreted as the weight or the corresponding feature being more "important." However, the true importance of each model layer depends on multifarious factors and the actual situation could be much more complex. For example, the importance of weight also depends on its location within the network structure. In some situations [6], smaller weights might be on a critical path and have a significant impact on the output, while larger weights might be on a less important path. Therefore, it's

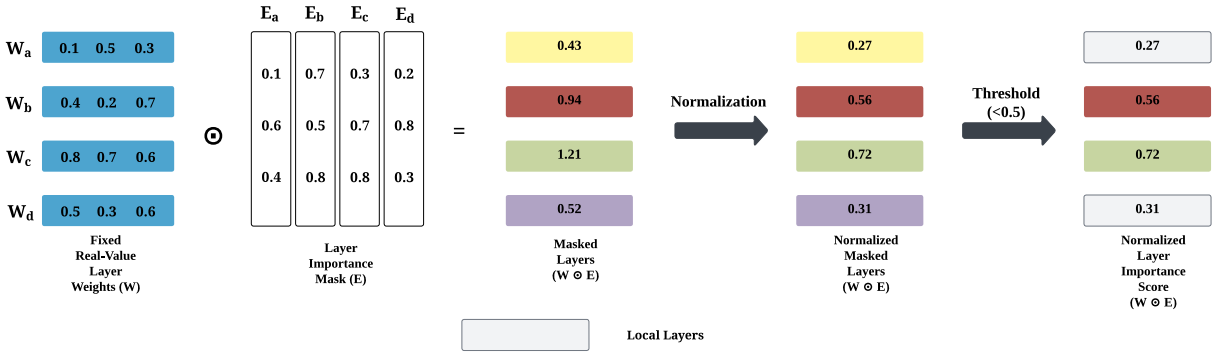


Figure 2: Personalization Layer Selection via Layer Importance Mask

necessary to dynamically select the importance of each layer to perform optimal model personalization in FL. Guided by the insights of deep learning model pruning [41] and Fedmask [16], we design a layer importance mask to capture the inner differences among each layer in the deep learning model. Figure 2 shows the detail of our model personalization layer selection via layer importance mask.

We use a fully connected layer as a representative example to demonstrate our proposed mechanism for choosing layers. It is important to mention that this approach is versatile and can be readily adapted to various other layer types. We define a fully connected layer as $y = W \cdot x$, where $y \in \mathbb{R}^m$ is the output, $x \in \mathbb{R}^n$ is the input, and $W \in \mathbb{R}^{m \times n}$ denotes the weight matrix of the fully connected layer. Instead of using a binary mask in deep learning model pruning, we present a self-learned real-value layer importance mask. We apply the layer importance mask as the same size as the model W, and express the mask-based fully connected layer as $y = (W \odot E) \cdot x$, where $\odot$ denotes the elementwise multiplication and E denotes the real-value mask, where $E \in \mathbb{R}^{m \times n}$. In the feed-forward step, we set E^B as a binary mask under the threshold function as Equation 1.

$$E^B_{ij} = \begin{cases} 1, & E_{ij} \geq \tau \\ 0, & E_{ij} < \tau \end{cases} \tag{1}$$

where E^B_{ij} represents the element located in i^{th} row and j^{th} column of E^B. τ denotes the threshold. Given the non-differentiable nature of the threshold function, the current method [40] directly applies the gradients of E to the binary mask E^B as Equation 2.

$$\frac{\partial L}{\partial E} = \frac{\partial L}{\partial E^B} \tag{2}$$

While this approach can optimize E and E^B, it might lead to significant variations in gradient magnitude, potentially hindering the optimization of E. In order to mitigate the variance in the gradient, we adopt the method in [16] and integrate a sigmoid function. By replacing the hard threshold function with the differentiable sigmoid function, the back-propagation step from E to E^B can be defined as Equation 3.

$$\frac{\partial L}{\partial E} = \beta \odot \frac{\partial L}{\partial E^B} \tag{3}$$

where β denotes the gradient matrix of the sigmoid function.

For each training iteration, we set τ as the threshold to determine local personalization layers and global layers. For example in Figure 2, as we set $\tau = 0.5$, we only transmit layer 2 and layer 3 to the central server for global aggregation. Since the layer importance mask is self-learned and updated with the deep learning model during the training step, we eliminate the assumption that larger weights are more important for the model. Since the pruned local contains more model information for the local user, we keep all pruned layer information for future local updates and personalized model generation, which can be beneficial for each client. On the other hand, thanks to the layer importance mask, `Atlas` only needs to transmit a partial number of model layer information to the central server, which helps to achieve better communication efficiency.

3.3 Dynamic Local Differential Privacy

The local differential privacy module requires clipping the computed parameter gradients and adding noise (e.g., Gaussian noise). However, the depth of the layers in a deep learning model influences the features that its parameters extract. Features of different scales correspond to different layers, resulting in gradients and variances of varying magnitudes. For example, lower layers in deep convolutional networks can extract texture features, while deeper layers focus on semantic features. In most of the current local differential privacy mechanisms, there has been a lack of explicit consideration for the variance in weight ranges across different layers of deep learning models. Recent works [23, 33] have discussed the possibilities of considering the impact of range difference of model weights in LDP, however, these methods set a fixed model weight range by default in their LDP mechanism, which is not practical. With `Atlas` pinpointing the significance of different layers in a deep learning model, we can now dynamically account for the variations in the range of model weights across these layers when implementing the local differential privacy mechanism. The core difference about `Atlas` is that `Atlas` adds different LDP noise to each selected layer based on their layer importance mask. We can also dynamically add

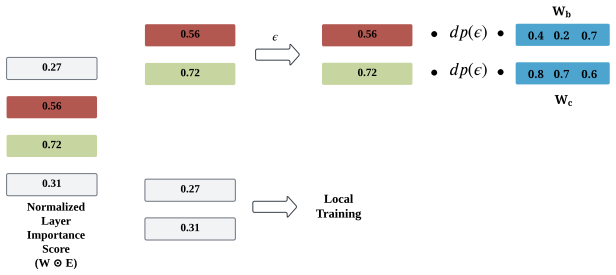


Figure 3: Dynamically adjust model weights to LDP mechanism.

the LDP based on the layer importance mask update from each FL global aggregation, which is optimal for the deep learning model performance while ensuring privacy protection. Figure 3 shows an example of our practical and dynamic LDP mechanism.

We first discuss the preliminary method which considers the range difference of model weights in the LDP mechanism. The main concept is to add noise separately to the gradients of each group using a unique pair of (C_g, z_g), where C is the clipping range, g is the the model gradient. First, we divide the model weights w into M groups. In the training process, the model gradients can be represented as $G = (g_1, g_2, ..., g_M)$. We denote that function of the clipping range C as $\pi_C(\mathbf{g}) = \mathbf{g} \cdot \min(1, \frac{C}{\|\mathbf{g}_t\|_2})$, for number L of training data in one training *lots*, we denote that gradient of the i_{th}, where $1 \le i \le L$ training data as $G^i = (g_1^i, g_2^i, ..., g_M^i)$. For all gradient G, we utilize (C_g, z_g) and perform differential privacy stochastic gradient descent (DP-SGD) as in Equation 4.

$$\begin{aligned} \tilde{\mathbf{G}} &= \frac{1}{L}[\sum_{i=1}^{L} \pi_C(\mathbf{G^i}) + \mathcal{N}(0; z^2C^2I)] \\ &= [\frac{1}{L}\sum_{i=1}^{L} \pi_C(\mathbf{G^i})] + \mathcal{N}(0; \frac{z^2C^2}{L^2}I) \end{aligned} \tag{4}$$

The previous research [23] proved that the above mechanism satisfied the (ϵ, δ)−DP, so we will not dive deep into the theoretical proof. Based on this mechanism, we observe that the differential privacy mechanism sums up the final aggregated gradients and then adds normalized noise with a variance of $\sigma = zC$. By averaging the obtained gradients, we acquire a gradient $\tilde{G}$ that satisfies the requirements of differential privacy. In this case, we can reformat Equation 4 that for each g_m, we use independent (C_m, z_m) as the differential privacy parameter. Since we have the layer importance mask E which is So we have Equation 5.

$$\begin{aligned} \tilde{\mathbf{g}}_m &= \frac{E}{L}[\sum_{i=1}^{L} \pi_{C_m}(\mathbf{g^i}_m) + \mathcal{N}(0; z_m^2C_m^2I)] \\ &= [\frac{E}{L}\sum_{i=1}^{L} \pi_{C_m}(\mathbf{g^i}_m)] + \mathcal{N}(0; \frac{z_m^2C_m^2}{L^2}I) \end{aligned} \tag{5}$$

where E is the layer importance score from the previous step. Since we require $\sum_{m=1}^{M} C_m^2 = C^2$, so we have $\|\mathbf{G}\|_2 \le C$, which satisfy the gradient clipping requirement of DP.

Next, we calculate the privacy bond for our dynamic LDP mechanism. We denote $\sigma_m = E_m z_m C_m$ and reformat Equation 5, we get Equation 6.

$$\widetilde{\mathbf{g}}_m = \frac{\sigma_m}{L}[\sum_{i=1}^{L} \pi_{C_m}(\mathbf{g^i_m})/\sigma_m + \mathcal{N}(0; I)] \tag{6}$$

We can get Equation 7 based on the properties of the gradient function.

$$\pi_{C_m}(\mathbf{g}_m)/\sigma_m = \pi_{C_m/\sigma_m}(\mathbf{g}_m/\sigma_m) \tag{7}$$

We can further scale the input vector to $\mathbf{G}^* = (\mathbf{g}_1/\sigma_1, \cdots, \mathbf{g}_M/\sigma_m)$, At this point, the differential privacy mechanism is equivalent to adding standard Gaussian noise to the scaled gradient vector $\mathbf{G}^*$, which can be expressed as Equation 8:

$$\widetilde{\mathbf{g}}_m = \sigma_m \cdot \frac{1}{L}[\sum_{i=1}^{L} \pi_{C_m/\sigma_m}(\mathbf{g^i_m}/\sigma_m) + \mathcal{N}(0; I)] \tag{8}$$

Then, we apply the clipping operation in groups with a threshold parameter of $C_m^* = C_m/\sigma_m, C^* = \sqrt{\sum_{m=1}^{M}(C_m^*)^2}$ LDP noise, where the noise multiplier z^* is as Equation 9.

$$z^* = \frac{1}{C^*} = \frac{1}{\sqrt{\frac{1}{z_1^2} + \cdots + \frac{1}{z_M^2}}} \tag{9}$$

We can obtain the privacy loss associated with our dynamic LDP mechanism by substituting the noise multiplier z^* into the calculations of moments accountant with Gaussian mechanism [1].

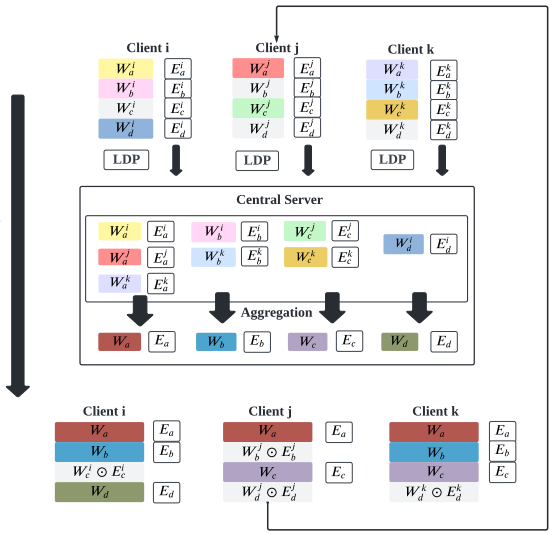


Figure 4: Atlas Model Aggregation

3.4 Central Server Aggregation

Figure 4 shows how we perform model aggregation in the central server for Atlas. After a certain round of local training, each client has the locally trained model weights for each layer, together with the layer importance mask. For each global aggregation round, after adding the dynamic LDP mechanism to the selected model weights and layer importance mask, the central server performs model aggregation. Note that for each client, the number of the selected layers is different, which means we can not directly perform an averaging strategy like FedAvg. To solve this problem, we perform layer-by-layer aggregation. As Figure 4 shows, client i selects model weight W_a^i, W_b^i and W_d^i for global aggregation, and keeps W_c^i locally for personalization. Client j selects model weight W_a^j and W_c^j for global aggregation and keeps W_b^j and W_d^j locally for personalization. Client k selects model weight W_a^k, W_b^k and W_c^k for global aggregation, and keeps W_d^k locally for personalization. Since each client has the default model structure, for each layer and client, we only aggregate their transmitted layers for global aggregation. More specifically in this example, we have:

$$\begin{aligned} W_{a\cdot t+1} &= \frac{W_{a,t}^i + W_{a,t}^j + W_{a,t}^k}{3} \\ W_{b\cdot t+1} &= \frac{W_{b,t}^i + W_{b,t}^k}{2} \\ W_{c\cdot t+1} &= \frac{W_{c,t}^j + W_{c,t}^k}{2} \\ W_{d\cdot t+1} &= W_{d,t}^i \end{aligned} \tag{10}$$

where W_t is the model weights for different layers before global aggregation, W_{t+1} is the model weights for different layers after global aggregation. We perform the same strategy to the layer importance mask as well.

3.5 Personalized Model Generation

As shown in Figure 1, after each global aggregation round, the central server sends back the updated layer weight for each client. Before the next local training round, each client generates their personalized modal locally based on the locally pruned layers and globally updated layers. Figure 5 shows an example of how Atlas achieves personalized model generation. We take client i as an example. Since the layer importance mask determined $W_{a,t}^i$, $W_{b,t}^i$ and $W_{d,t}^i$ for global aggregation, $W_{c_t}^i$ stayed locally waiting for the next local computational round. After our system updates all the selected layers, Client i receives the new weights for different layers, which is $W_{a,t+1}^i$, $W_{b,t+1}^i$ and $W_{d,t+1}^i$. The updated model weights then combine with the localized $W_{c_t}^i$ for the next local computational round for local training. Such a design takes into account the global aggregation as well as the localized personalized information, which ensures that the updated personalized model can achieve the best performance for each client in the system.

4 Evaluation

To demonstrate the generality, practicability and usefulness of Atlas across different IoT applications, we design our experiments

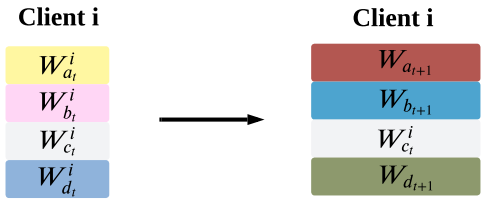


Figure 5: Atlas Personalized Model Generation

to evaluate Atlas on five real-world IoT application datasets focusing on the following research questions:

- **RQ1**: How does Atlas perform on real-world datasets compared to traditional methods, non-privacy-preserving FL methods, and LDP-based FL methods?
- **RQ2**: How does Atlas perform on variate of scalability compared to non-privacy-preserving FL methods and LDP-based FL methods?
- **RQ3**: How do different LDP privacy budgets in Atlas and LDP-based FL methods affect the accuracy of different IoT applications?
- **RQ4**: For IoT applications, it's practical to have new clients with heterogeneous data joining the system. How well can Atlas handle the new clients?
- **RQ5**: How is the communication efficiency?
- **RQ6**: How does the removal of different modules affect Atlas?

4.1 Applications and Datasets

Below are detailed descriptions of these five IoT applications and CIFAR-10, the statistics of the datasets that are used in the applications are concluded in table 2.

Table 2: Dataset Statistics.

Dataset	Statistics			
	IoT Platform	Data Modality	Dataset Size	Model
Pecan Street	Smart Home	Energy	3.2 GB	LSTM
UCI-HAR	Smartphone	Accelerometer, Gyroscope	58.17 MB	LSTM
MASS	PSG	PSG	1.2 GB	LSTM
WISDM-W	Smartwatch	Accelerometer, Gyroscope	294 MB	LSTM
CASAS	Smart Home	Motion Sensor, Door Sensor, Thermostat	233 MB	LSTM
CIFAR-10	-	Image	163 MB	Resnet 32

Application#1: Energy Prediction. Pecan Street [1] contains the energy consumption for every second of different home appliances in 1641 residents in Texas from 2013-01-01 to 2017-12-31. Since the recorded data has some missing data points, also the devices in each resident are not exactly the same, we randomly select 100 residents that have the full load records as our default experiment dataset. Recently, Pecan Street dataset also updated second-level data for New York and minute-level data for California. To test how Atlas can handle the new users and demonstrate the performance of **RQ4**, we will use the New York dataset together with Texas dataset for evaluation.

Application#2: Human Activity Recognition. Human Activity Recognition (HAR) is a task that distinguishes a physical activity

[1] https://www.pecanstreet.org/dataport/

completed by an individual subject. Physical activities, for example, *Walking*, *Standing*, and *Sitting*, etc. build our comprehensive actions in daily life. A well-known publicly available HAR dataset called *Human Activity Recognition Using Smartphones DataSet* [2], the HAR using smartphones dataset contains 30 participants age ranged from 19-48. Each person performed six activities (WALKING, WALKING UPSTAIRS, WALKING DOWNSTAIRS, SITTING, STANDING, LAYING) wearing a smartphone (Samsung Galaxy S II) on the waist [4]. Two types of 3-axial signal data are collected from the accelerometer sensor and gyroscope sensor in fixed-width sliding windows of 2.56 sec respectively.

Application#3: Sleep Stage Classification. Sleep stage classification is essential for the assessment of sleep quality and the diagnosis of sleep disorders. The *Montreal Archive of Sleep Studies (MASS)* [3] is an open-access and collaborative database of laboratory-based polysomnography (PSG) recordings. There are 97 males and 103 females of age ranging from 18 to 76 years participated in this study, and five sleep stages are recorded for classification [28].

Application#4: Daily Activity Recognition. The WISDM (Wireless Sensor Data Mining) dataset [19] stands as a prominent resource for daily activity recognition tasks. It utilizes accelerometer and gyroscope sensor data gathered from smartphones and smartwatches. The dataset encompasses data from 51 participants, each engaged in 18 distinct daily activities over sessions lasting 3 minutes. We keep the same data pre-processing mechanism as mentioned in [3]. More specifically, we have streamlined the dataset by consolidating activities like eating soup, chips, pasta, and sandwiches into a unified "eating" category, while eliminating rare activities associated with ball play, such as kicking, catching, and dribbling. Acknowledging that individuals might not simultaneously carry a smartphone and wear a smartwatch in real-world scenarios, we divided WISDM into two distinct datasets: WISDM-W, containing solely smartwatch data, and WISDM-P, comprising only smartphone data. The training and test sets for WISDM-W include 16,569 and 4,103 samples, respectively, while for WISDM-P, they contain 13,714 and 4,073 samples, respectively. In our experiment, we only focused on the WISDM-W subset of this dataset, mainly because the application#2 is a smartphone-based dataset.

Application#5: Smart Home Daily Living Sequence Recognition. The CASAS dataset, a product of the CASAS smart home project, serves as a resource for recognizing daily living activities (ADL) through sequences of sensor states over time, aiming to facilitate independent living applications. The data collection took place across three different apartments, each outfitted with a trio of sensor types: motion, temperature, and door sensors. We keep the same data pre-processing mechanism as mentioned in [3]. We have chosen five specific datasets named "Milan", "Cairo", "Kyoto2", "Kyoto3", and "Kyoto4", selected for the consistency in their sensor data representation. Within each dataset, we have streamlined the original ADL categories into 11 home activity-related categories such as "sleep", "eat", and "bath". Each data entry is a categorical time series with a length of 2,000, depicting sensor states across a certain time span. In total, the training and test sets comprise 12,190 and 3,048 samples, respectively.

[2]https://archive.ics.uci.edu/ml/datasets/
[3]http://ceams-carsm.ca/mass/

Application#6: Image Classification. Image classification is an important part of smart IoT applications. We use CIFAR-10 [8] to demonstrate the performance of `Atlas` in Image Classification. Although CIFAR-10 is not a typical IoT application dataset, it has been widely used to test the performance of federated learning frameworks. In this experiment, we use Resnet 32 [12] as the base model for training.

4.2 Experimental Setup and Evaluation Metrics

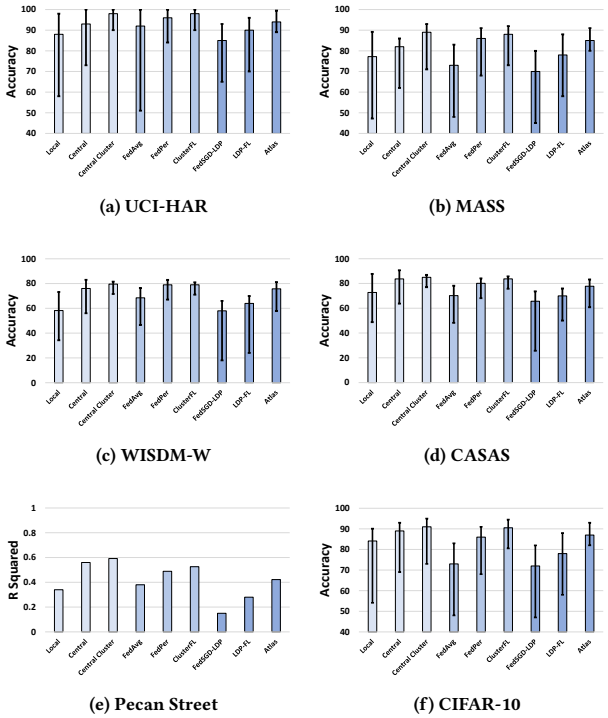
We implemented our experiments on five real-world IoT application datasets and CIFAR-10 using two NVIDIA RTX 3090 GPUs. We divided the data into training and test sets using an 80-20 split. We chose the base model with the same structure including 8 LSTM layers with three fully connected layers for each application (for CIFAR-10, we use Resnet 32 as the base model). By default, each participating client performs 10 local training rounds for one global aggregation round when evaluating the model performance of each application. We also evaluate the performance of different participating clients and the influence on different local training rounds/-global aggregation rounds as well.

We compare `Atlas` with 8 baselines in 3 categories. (1) Traditional methods include localized, centralized model, centralized clustering [37] and FedAvg [22], (2) Personalized FL approaches include FedPer [5] and Cluster FL [29], (3) Privacy-preserving FL methods include FedSGD-LDP [15] and LDP-FL [33]. For evaluation metrics, we use accuracy for the classification IoT applications and R^2 for regression applications. We evaluate the performance on each client's test data, and report the average accuracy over all clients for evaluations. For communication evaluation, we normalize the time cost as the ratio to the communication time of FedAVG as reported communication cost. The details of the baselines are as follows:

- **Localized model**, trains models on local clients by using only the local data, without the collaborations between different clients. Since there is no privacy issue nor communication cost when training data is completely local, we use this method only as a baseline to compare the model performance among different IoT applications.
- **Centralized model**, trains all data on a centralized global server as a unified model. This method has significant privacy issues such as malicious cloud servers, etc. We did not use any privacy protection mechanisms in this approach. We use this method only as a baseline to compare the model performance among different IoT applications.
- **Centralized clustering model [37]**, trains all data on a centralized global model with a centralized clustering algorithm. Similar to the Localized and centralized model, we use the centralized clustering model only as a baseline to compare the model performance among different IoT applications. We include this method to show the best performance that collaborative training among different users can get to compare with our method.
- **FedAvg** [22], is a classic federated learning model that clients exchange model information with the central server by sending updated local parameters and retrieving the aggregated global model, facilitating ongoing local training.

- **FedPer** [5], is a federated learning model that each client contributes to share their lower layers and leave their upper layers locally for personalization. The method requires to pre-set the number of layers that are shared in the system. In our experiments, we set the number of layers to be 4.
- **Cluster FL** [29], introduces a unique clustered federated learning framework, enhancing the training accuracy while discerning the inherent clustering relationship among data across different nodes. Leveraging these cluster relationships, ClusterFL efficiently eliminates slower-converging or weakly-correlated nodes within each cluster, hastening convergence without sacrificing accuracy.
- **FedSGD-LDP [15]**, introduces Gaussian mechanisms to preserve local differential privacy (LDP) of user data in the FL model. The method directly adds LDP noise to all the model weights to achieve privacy guarantee for each client.
- **LDP-FL [33]**, makes the local weights update differentially private by adapting to the varying ranges at different layers of a deep neural network, which introduces a smaller variance of the estimated model weights. This method considers a variety of range differences in model weights to achieve a better performance.

4.3 Results

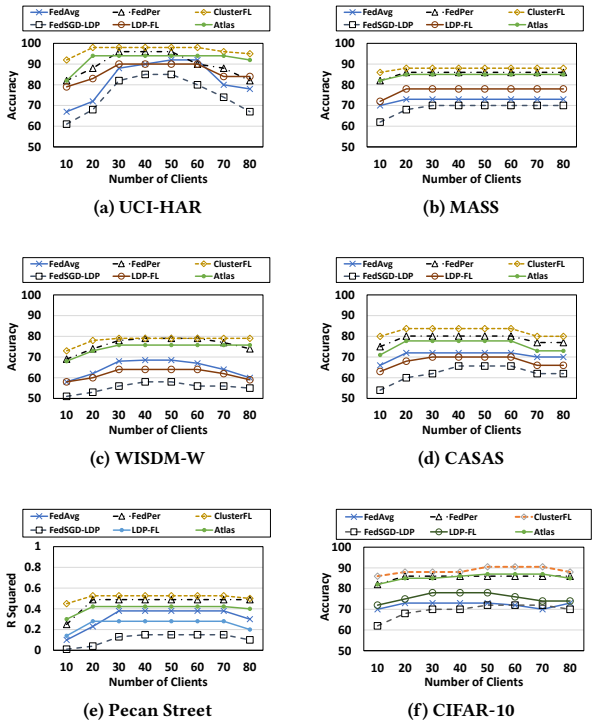


(a) UCI-HAR (b) MASS (c) WISDM-W (d) CASAS (e) Pecan Street (f) CIFAR-10

Figure 6: Performance comparison on accuracy and R^2 between traditional methods, non-privacy-preserving FL methods, LDP-based FL methods and `Atlas`

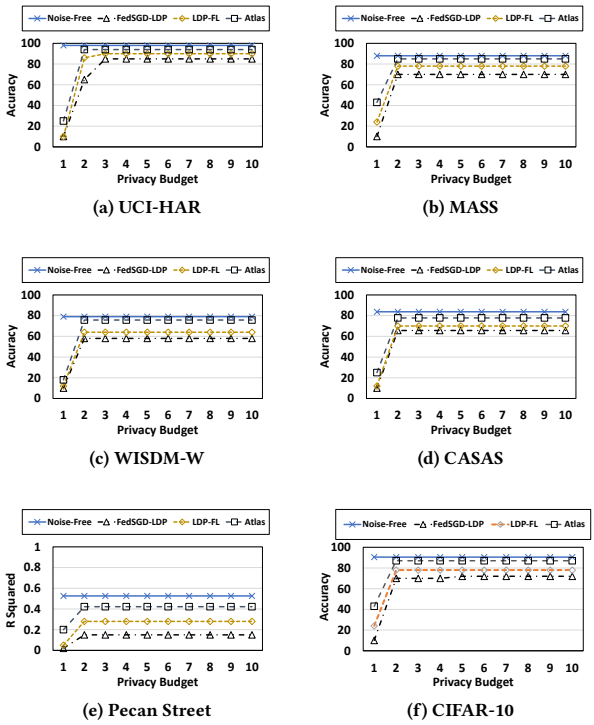
4.3.1 RQ1: Performance Analysis on real-world IoT data. In this section, we evaluate how `Atlas` performs compared to the other 8 baseline models. Figure 6 demonstrates that `Atlas` can achieve comparable accuracy performance compared to the other personalized model while providing privacy guarantee. Also, `Atlas` outperforms all other LDP-based FL frameworks. More specifically, we can observe that for UCI-HAR, `Atlas` can achieve 94% accuracy, which is similar level compared to central training, and FedPer. Compared to the best performance model, which is ClusterFL, `Atlas` sacrifices 4% accuracy performance in exchange for privacy-preserving, which is practical for real-world IoT applications. For MASS, `Atlas` achieves 85% accuracy which sacrifices 4% accuracy compared to the centralized clustering method and 3% compared to ClusterFL. `Atlas` also outperforms the LDP-FL-based model by 7%. For WISDM-W and CASAS, the results follow the same pattern as the classification tasks. We observe that `Atlas` generally sacrifices 2% to 6% of accuracy in exchange for privacy guarantee. One unique case is the Pecan Street data, we observe that `Atlas` sacrifices 0.104 in R^2, which is larger than other tasks. We also noticed that for all LDP-based FL methods, the performance dropped dramatically. The reason is that time-series data such as energy data, typically contains crucial temporal correlations and patterns. Adding noise to satisfy differential privacy requirements can distort these temporal relationships, leading to misinformation extracted from the data which makes the model performance drop higher. For all LDP base FL frameworks, `Atlas` outperforms LDP-FL by 0.142 and FedSGD-LDP by 0.272, which is still the best for privacy-guarantee methods. For image classification tasks like CIFAR-10, we observe the same trend as the previous tasks. We also notice that the accuracy performance of `Atlas` only drops 3.2% comparing to the best performed method, which is central cluster, and outperformed the LDP-based methods by 10%.

4.3.2 RQ2: Performance analysis on scalability. Figure 7 shows the model performance comparison on different numbers of clients participating in the system. We compare `Atlas` with non-privacy-preserving FL methods, LDP-based FL methods in this section. For each IoT application, we assign a different number of clients during the training process. We observe that the general performance of `Atlas` follows the same pattern as in RQ1. Interestingly to see that, for UCI-HAR, the performance changes dramatically based on the change in the number of clients compared to other datasets. We see that along with the growth of the participating clients, most of the methods first achieve a higher model performance, but fail to hold such a performance while the number of clients is growing higher. However, `Atlas` and ClusterFL remain steady with the growth of the number of clients. The reason is because of the model personalization. For other methods, since there is no model personalization mechanism, as more and more clients joined the system, the diversity of clients' data makes the global model vulnerable to providing an accurate global model for each client. However, `Atlas` utilizes the layer personalization mask for model personalization, which keeps the personalization layers locally and only updates the weights that are beneficial for the global model, which makes `Atlas` maintain a steady performance facing a larger amount of clients. From this study, we also think that it showcases the uniqueness of the IoT application. For different IoT applications,



(a) UCI-HAR (b) MASS (c) WISDM-W (d) CASAS (e) Pecan Street (f) CIFAR-10

Figure 7: Model performance comparison on variate of scalability between non-privacy-preserving FL methods, LDP-based FL methods and `Atlas`.



(a) UCI-HAR (b) MASS (c) WISDM-W (d) CASAS (e) Pecan Street (f) CIFAR-10

Figure 8: Model performance comparison on variate of privacy budget ϵ between LDP-based FL methods and `Atlas`.

we can observe different moving patterns with different numbers of users. As for the other applications, we can see that for MASS, Pecan Street, CASAS, WISDM-W and CIFAR-10, the accuracy and R square first move higher along with the increase of the client numbers by around 10%, then decrease by around 2-4% once the number of clients is getting larger, which is around 70. The main reason is that, the clients may have different data patterns, and since the number of clients is going up, the performance will slightly drop due to the heterogeneity of user data.

4.3.3 RQ3: Impact of Different LDP Parameter ϵ. Figure 8 shows the impact of privacy budget ϵ for LDP-based FL framework. In differential privacy, there's typically a trade-off between privacy and accuracy. A larger privacy budget allows for more accurate results, as it permits more noise to be added to the data, thereby preserving privacy while minimizing the impact on the accuracy of the analysis or query results. We assign the same ϵ to all LDP-based FL models to ensure that they are under the same privacy guarantee level. We can observe that for all IoT applications, the accuracy follows Noise-Free > `Atlas` > LDP-FL > FedSGD-LDP. We can also observe that the accuracy remains steady once the privacy budget increases to 2. `Atlas` outperforms all other baseline models and only sacrifices 2% to 6% compared to the noise-free method.

4.3.4 RQ4: Impact on New Users. In real-world settings, it's practical that there will be new users joining the system. Traditionally

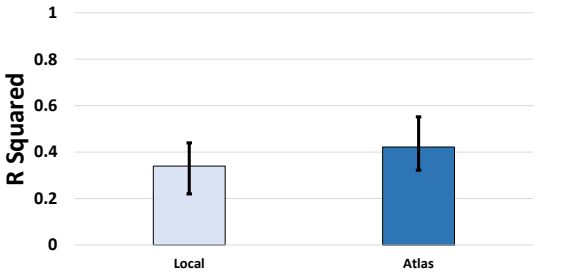


Figure 9: Model performance comparison on adding new users in `Atlas`.

researchers tend to use dataset separation to evaluate the model's ability to handle new clients in the system. However, it's not practical in real-world IoT settings. Because even with dataset separation, part of the separated dataset still contains the data pattern even if it's not used in either model training or testing. To this end, we want to evaluate how `Atlas` handles new users more practically. As we mentioned in the dataset selection section. We trained our model solely using the Pecan Street energy datasets from houses in Texas. To evaluate our assumption, here we use the trained model from Pecan Street Texas data and use purely Pecan Street New York

Table 3: Ablation study on the accuracy of Atlas w/o different modules across different datasets.

Model/Datasets	UCI-HAR	MASS	WISDM-W	CASAS	Pecan Street (R^2)	CIFAR-10
Atlas	94.32	85.24	75.77	77.78	0.422	85.12
Atlas w/o MLS and LIM	86.88	77.65	68.46	70.23	0.382	77.48
Atlas w/o DLDP	97.20	88.43	79.11	81.32	0.447	88.91
Atlas w/o layer-by-layer aggregation	91.66	82.36	73.28	75.42	0.409	82.23

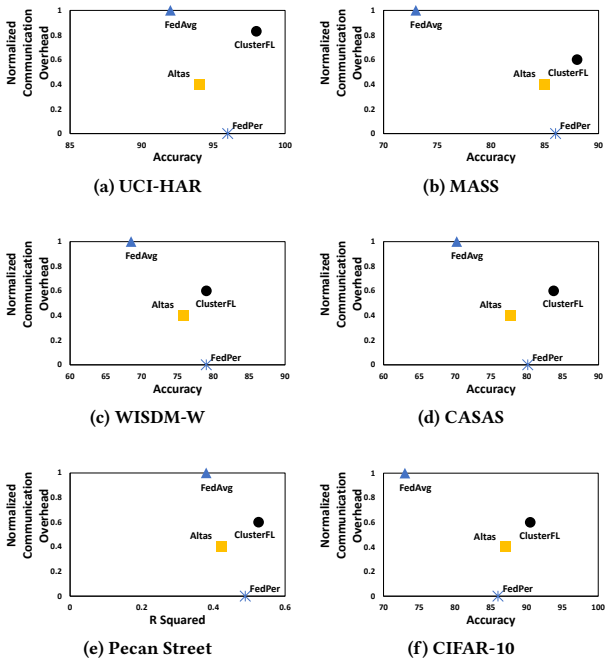


Figure 10: Model performance comparison on communication efficiency between FL methods and Atlas.

data as testing. Since there is a time difference between NY and TX, we adjusted the timeframe in the original dataset and performed the experiment. Figure 9 shows the result of the impact on new users joining the system. We can observe that, compared with the local training method, Atlas can achieve better performance for most of the new clients, which proves that Atlas can handle new users joining the system.

4.3.5 RQ5: Communication Efficiency. Figure 10 shows the performance of communication efficiency between FL methods and Atlas. We observe that for all IoT applications, the results have a similar pattern that Atlas outperforms FedAVG and ClusterFL, but not as well as FedPer. The reason is that, although both FedPer and Atlas pruned certain model layers locally and only transmit partial of the whole model, Atlas also needs to transmit the layer importance mask to dynamically the layer selection mechanism and LDP, which will incur more communication cost.

4.3.6 RQ6: Ablation Study. Table 3 presents the ablation study results, evaluating the impact of removing different modules from Atlas across various datasets. Removing **Model Layer Selection (MLS) and Layer Importance Mask (LIM)** leads to a significant performance drop of 5% to 10% across datasets (e.g., UCI-HAR drops to 86.88%), indicating their crucial role in selecting the most relevant layers and improving accuracy effectively. The **Dynamic Local Differential Privacy (DLDP)** module is designed to enhance privacy protection by introducing controlled and weighted noise, which naturally results in a slight accuracy reduction. As expected, removing DLDP leads to a 3% to 5% accuracy gain across datasets (e.g., CIFAR-10 improves from 85.12% to 88.91%). Despite this trade-off, our method still surpasses all existing LDP-based baselines, demonstrating a superior balance between privacy and model performance. Furthermore, removing **layer-by-layer aggregation** results in a moderate accuracy drop (3% to 5%), emphasizing its importance in maintaining model performance. Unlike traditional averaging methods, our layer-by-layer aggregation strategy only aggregates layers that are actually uploaded by each client, while excluding those that remain local. This approach ensures that personalization is retained for non-uploaded layers, allowing clients to maintain certain unique representations while still benefiting from global knowledge fusion. Overall, these results confirm that MLS and LIM are critical for optimal feature extraction, while our aggregation method preserves personalization by selectively fusing only the uploaded layers. Although D-LDP introduces a necessary trade-off for privacy preservation, our approach maintains competitive performance, outperforming all LDP-based baselines.

5 Conclusion

We present Atlas, a practical personalized federated learning framework with dynamic local differential privacy for IoT applications. Initially, we implemented a layer importance mask for sharing layers, distinguishing between global and personalized components within the local model. Following that, we introduce weight-enhanced Local Differential Privacy (LDP) noise to the global layers prior to their integration into the federated learning framework for aggregation purposes. Ultimately, we amalgamate the local personalized layers with the aggregated global layers to effectively execute IoT tasks. Atlas achieved comparable model performance with non-privacy-preserving methods with privacy guarantee.

Acknowledgments

The authors gratefully acknowledge the support provided by the National Science Foundation under NSF #2217071 and the Commonwealth Cyber Initiative (CCI). We appreciate the guidance and resources that have been instrumental in advancing our work.

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *SIGSAC*.

[2] Tanveer Ahmad, Huanxin Chen, Ronggeng Huang, Guo Yabin, Jiangyu Wang, Jan Shair, Hafiz Muhammad Azeem Akram, Syed Agha Hassnain Mohsan, and Muhammad Kazim. 2018. Supervised based machine learning models for short, medium and long-term energy prediction in distinct building environment. *Energy* 158 (2018), 17–32.

[3] Samiul Alam, Tuo Zhang, Tiantian Feng, Hui Shen, Zhichao Cao, Dong Zhao, JeongGil Ko, Kiran Somasundaram, Shrikanth S Narayanan, Salman Avestimehr, et al. 2023. FedAIoT: A Federated Learning Benchmark for Artificial Intelligence of Things. *arXiv preprint arXiv:2310.00109* (2023).

[4] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra Perez, and Jorge Luis Reyes Ortiz. 2013. A public domain dataset for human activity recognition using smartphones. In *ESANN*.

[5] Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. 2019. Federated learning with personalization layers. *arXiv preprint arXiv:1912.00818* (2019).

[6] Alain Barrat, Marc Barthelemy, Romualdo Pastor-Satorras, and Alessandro Vespignani. 2004. The architecture of complex weighted networks. *Proceedings of the national academy of sciences* 101, 11 (2004), 3747–3752.

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

[8] CIFAR-10. [n. d.]. The CIFAR-10 dataset. https://www.cs.toronto.edu/~kriz/cifar.html.

[9] Jiechao Gao and Yuangang Li. 2024. FedMetaMed: Federated Meta-Learning for Personalized Medication in Distributed Healthcare Systems. In *2024 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 6384–6391.

[10] Jiechao Gao, Mingyue Tang, Tianhao Wang, and Bradford Campbell. 2022. PFed-LDP: A Personalized Federated Local Differential Privacy Framework for IoT Sensing Data. In *Sensys*.

[11] Jiechao Gao, Wenpeng Wang, Fateme Niksersht, Viswajith Govinda Rajan, and Bradford Campbell. 2023. PFDRL: Personalized Federated Deep Reinforcement Learning for Residential Energy Management. In *ICPP*.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[13] Weiwei Jiang and Jiayun Luo. 2022. Big data for traffic estimation and prediction: a survey of data and tools. *Applied System Innovation* 5, 1 (2022), 23.

[14] Yihan Jiang, Jakub Konečnỳ, Keith Rush, and Sreeram Kannan. 2019. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488* (2019).

[15] Muah Kim, Onur Günlü, and Rafael F Schaefer. 2021. Federated learning with local differential privacy: Trade-offs between privacy, utility, and communication. In *ICASSP*.

[16] Ang Li, Jingwei Sun, Xiao Zeng, Mi Zhang, Hai Li, and Yiran Chen. 2021. Fedmask: Joint computation and communication-efficient personalized federated learning via heterogeneous masking. In *Sensys*. 42–55.

[17] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. 2020. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine* (2020).

[18] Paul Pu Liang, Terrance Liu, Liu Ziyin, Nicholas B Allen, Randy P Auerbach, David Brent, Ruslan Salakhutdinov, and Louis-Philippe Morency. 2020. Think locally, act globally: Federated learning with local and global representations. *arXiv preprint arXiv:2001.01523* (2020).

[19] Jeffrey W Lockhart, Gary M Weiss, Jack C Xue, Shaun T Gallagher, Andrew B Grosner, and Tony T Pulickal. 2011. Design considerations for the WISDM smart phone-based sensor mining architecture. In *SensorKDD*.

[20] L Lyu, H Yu, X Ma, L Sun, J Zhao, Q Yang, and PS Yu. 2012. Privacy and robustness in federated learning: Attacks and defenses, 2020. *arXiv preprint arXiv:2012.06337* (2012).

[21] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. 2020. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619* (2020).

[22] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*.

[23] H Brendan McMahan, Galen Andrew, Ulfar Erlingsson, Steve Chien, Nicolas Papernot, and Peter Kairouz. 2018. A general approach to adding differential privacy to iterative training procedures. *arXiv preprint arXiv:1812.06210* (2018).

[24] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. 2019. Exploiting unintended feature leakage in collaborative learning. In *S & P*.

[25] Dinh C Nguyen, Ming Ding, Pubudu N Pathirana, Aruna Seneviratne, Jun Li, and H Vincent Poor. 2021. Federated learning for internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials* (2021).

[26] Thông T Nguyên, Xiaokui Xiao, Yin Yang, Siu Cheung Hui, Hyejin Shin, and Junbum Shin. 2016. Collecting and analyzing data from smart device users with local differential privacy. *arXiv preprint arXiv:1606.05053* (2016).

[27] Solmaz Niknam, Harpreet S Dhillon, and Jeffrey H Reed. 2020. Federated learning for wireless communications: Motivation, opportunities, and challenges. *IEEE Communications Magazine* (2020).

[28] Christian O'reilly, Nadia Gosselin, Julie Carrier, and Tore Nielsen. 2014. Montreal Archive of Sleep Studies: an open-access resource for instrument benchmarking and exploratory research. *Journal of sleep research* 23, 6 (2014), 628–635.

[29] Xiaomin Ouyang, Zhiyuan Xie, Jiayu Zhou, Jianwei Huang, and Guoliang Xing. 2021. Clusterfl: a similarity-aware federated learning system for human activity recognition. In *Mobisys*. 54–66.

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

[31] Reza Shokri and Vitaly Shmatikov. 2015. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*.

[32] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. 2017. Federated multi-task learning. *NeurIPS* (2017).

[33] Lichao Sun, Jianwei Qian, and Xun Chen. 2020. Ldp-fl: Practical private aggregation in federated learning with local differential privacy. *arXiv preprint arXiv:2007.15789* (2020).

[34] Aleksei Triastcyn, Matthias Reisser, and Christos Louizos. 2021. Dp-rec: Private & communication-efficient federated learning. *arXiv preprint arXiv:2111.05454* (2021).

[35] Linlin Tu, Xiaomin Ouyang, Jiayu Zhou, Yuze He, and Guoliang Xing. 2021. Feddl: Federated learning via dynamic layer sharing for human activity recognition. In *Sensys*.

[36] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. 2020. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE TIFS* (2020).

[37] Rui Xu and Donald Wunsch. 2005. Survey of clustering algorithms. *IEEE Transactions on neural networks* 16, 3 (2005), 645–678.

[38] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated machine learning: Concept and applications. *ACM (TIST)* (2019).

[39] Yang Zhao, Jun Zhao, Mengmeng Yang, Teng Wang, Ning Wang, Lingjuan Lyu, Dusit Niyato, and Kwok-Yan Lam. 2020. Local differential privacy-based federated learning for internet of things. *IEEE Internet of Things Journal* 8, 11 (2020), 8836–8853.

[40] Hattie Zhou, Janice Lan, Rosanne Liu, and Jason Yosinski. 2019. Deconstructing lottery tickets: Zeros, signs, and the supermask. *NeurIPS* (2019).

[41] Michael Zhu and Suyog Gupta. 2017. To prune, or not to prune: exploring the efficacy of pruning for model compression. *arXiv preprint arXiv:1710.01878* (2017).