FedWindow: A Sliding Window-based Federated Learning Model Aaggregation Approach

1st Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

2nd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

3rd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

4th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

5th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

6th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

***Abstract*—Federated Learning (FL) has emerged as a transformative approach that enables multiple participants to collaboratively train a shared global model while preserving data privacy. However, concurrent updates from a large number of clients can lead to network congestion, which adversely affects the efficiency and stability of model training. To mitigate these issues and enhance the robustness and generalization of the global model, we introduce the FedWindow sliding window scheduling algorithm. This algorithm implements a sliding window congestion control mechanism between clients and the central server, dynamically adjusting the timing and sequence of client model uploads to alleviate network congestion. Through theoretical analysis and empirical validation, we have demonstrated the convergence of FedWindow in non-convex optimization scenarios and its advantages in reducing training delays and improving model accuracy. The experimental results show that compared to traditional synchronous methods, FedWindow significantly reduces delays and upload failures caused by congestion while maintaining model performance.**

***Index Terms*—Federated Learning, Sliding window.**

I. Introduction

With the proliferation of mobile devices and edge computing nodes, Federated Learning (FL) has emerged as a machine learning paradigm that allows multiple participants to collaboratively train a shared global model without sharing their private data. This decentralized learning approach not only preserves user privacy but also reduces the burden on central servers by minimizing data transmission. However, network congestion becomes a critical issue when a large number of users (also referred to as clients) attempt to upload their local model updates simultaneously during the synchronous FL process. This congestion can delay the aggregation of model updates and, in extreme cases, may lead to server overload, severely impacting the efficiency and stability of global model training.

To address this issue, we propose a novel method that integrates FL with a sliding window mechanism, designed to alleviate congestion by dynamically adjusting client participation. Our method is inspired by congestion control techniques in computer networks, particularly the sliding window protocol, which controls the flow of data packets by dynamically adjusting the window size to prevent network congestion. In this paper, we introduce the concept of the sliding window to FL and propose a Sliding Window Scheduling Algorithm (SWSA) to optimize the timing and sequence of client uploads. This algorithm aims to enhance the process of model aggregation in FL without incurring additional communication or computational burdens on clients and can be easily applied atop any existing FL algorithm. By dynamically adjusting the sliding window, our approach flexibly responds to varying network conditions and client statuses, reducing upload requests during peak periods, avoiding congestion, and enhancing the efficiency of global model aggregation. Our experimental results demonstrate that SWSA significantly reduces delays and failed upload attempts due to congestion while maintaining model performance, compared to traditional synchronous FL methods.Our key contributions are as follows:

1) We have designed a new federated learning architecture, *FedWindow*, which incorporates a sliding window congestion control mechanism between clients and the central server to dynamically adjust the frequency of global model parameter updates.
2) We provide a theoretical demonstration of the convergence analysis of *FedWindow* in a smooth, non-convex setting. The server can fine-tune the length of the sliding window to reduce the waiting delay caused by stragglers during training, thereby enhancing the accuracy of the model.
3) We have evaluated the performance of our proposed method under various sliding window lengths and under conditions of client heterogeneity, demonstrating its advantages in improving the efficiency and robustness

of federated learning.

The structure of this paper is as follows: Section II introduces the federated learning framework based on the sliding window; Section III elaborates on the training process and convergence analysis of our framework; Section IV presents the experimental results; and finally, Section V concludes the paper and discusses future research directions.

II. Sysetem Model

A. A federated learning wireless multi-user system

We consider a federated learning wireless multi-user system consisting of a server for model aggregation and a client set $\mathcal{U} = \{u_0, u_1, \cdots, u_{N-1}\}$ of N clients for federated learning training. Each participating user u_i stores a local data set $\mathcal{D}_i$, whose size is represented by D_i.

In FL, the training objective is to learn a global model $J(w) : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{X}$ is the input space (such as images), $\mathcal{Y}$ is the output space (such as labels), and $w \in \mathbb{R}^d$ represents the model parameters. Training is conducted over T rounds of communication and distributed across a set of devices S, which can access local private datasets $\mathcal{D}_i = \{(x_j, y_j)|x_j \in \mathcal{X}, y_j \in \mathcal{Y}, j \in D_i\}$, for $i \in S$.

For a sample data (x_j, y_j),with input x_j, the task is to find the model parameter w that characterizes the output y_j with the loss function $f_j(w)$, the loss function on the data set of u_i is defined as

$$F_i(w) := \frac{1}{D_n} \sum_{j \in \mathcal{D}_n} f_j(w). \tag{1}$$

Then, We consider the following global loss function minimization problem

$$\min_{w \in \mathbb{R}^d} F(w) := \sum_{i=1}^{N} \frac{D_i}{D} F_i(w), \tag{2}$$

where we can define the total data size by $D = \sum_{n=1}^{N} D_i$.

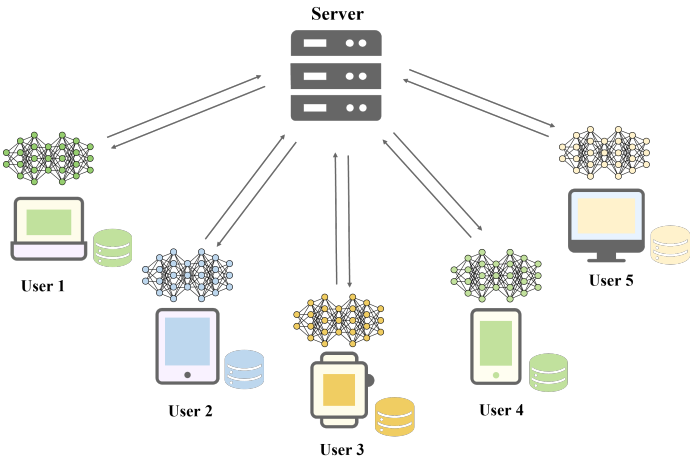

Fig. 1. A federated learning wireless multi-user system architecture

B. FedWindow: Federated Learning with sliding window Aggregation

To overcome the straggler problem in the aggregation process of federated learning in a wireless multi-user scenario, we introduce a sliding window-based model aggregation mechanism. We define the window size as W, which means selecting users entering the sliding window to perform local training. The trained local models are transmitted to the server. The server aggregates the models of the other users within the window into a global model, and then sends this global model to the users who are about to leave the window. The fundamental principle behind this approach is to enhance the robustness of the global model to distribution shifts across different rounds.

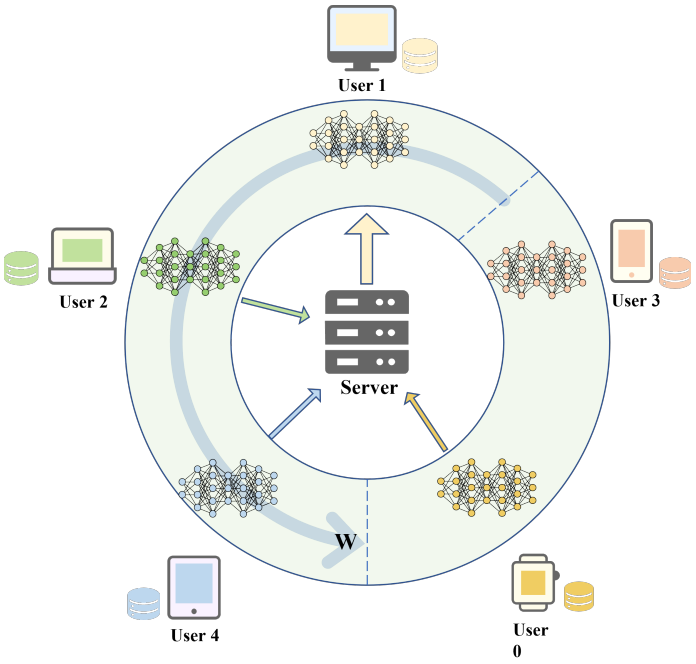

Fig. 2. Proposed FedWindow training architecture.

As illustrated in Figure 2, in each communication round, the server determines the length of the sliding window W based on its network conditions. Consider a scenario with five clients, represented as u_0, u_1, u_2, u_3, and u_4. The client about to enter the sliding window, u_0, performs local training, and the locally trained model is transmitted to the server. The local models of u_2 and u_4 are also uploaded to the server to be aggregated into the global model, which is then sent to the outgoing client u_1. In the subsequent round, the incoming client u_3 trains locally, and its model, along with the previously uploaded models of u_4 and u_0, is aggregated by the server into the global model, which is then dispatched to the outgoing client u_2. This process continues in a similar fashion for the other clients, with each client undergoing a cycle of sending and receiving once per round, maintaining the communication volume consistent with vanilla federated learning. This process is repeated until the model converges, as detailed in Algorithm 1.

III. Unveiling the sliding window

In this section, we will detail the FedWindow training process in a cross-device setting and introduce the convergence analysis of FedWindow.

A. FedWindow Training Process

1) Local Model Update: Each client u_i performs E epochs of local training to update its model parameters at each round t:

$$w_i^{(t+1)} = w_i^{(t)} - \eta \sum_{e=1}^{E} g_i(w_i^{(t,e)}), \tag{3}$$

where $w_i^{(t,e)}$ denotes the model parameters of client i after the e-th epoch in round t.

2) Global Model Update: At each step t, the model of the incoming client i is aggregated with the models of the remaining clients within the window, excluding the client j that is about to leave the window:

$$w^{(t+1)} = \frac{1}{W}\left(\sum_{k \in W_t \setminus \{j\}} w_k^{(t)} + w_i^{(t+1)}\right), \tag{4}$$

Algorithm 1: FedWindow Training Process

```
Server initializes model parameters W^0;
Clients upload state information (D_i: dataset size, C_i:
  computational power) to the server;
Server calculate sliding window length W;
for each round t from 0 to T − 1 do
    Sliding window slides one step;
    for About to enter sliding window W u_inwindow do
        for each epoch from 0 to E − 1 do
            for each batch (x_i, y_i) in D_i do
                BatchUpdate(w_i^t, (x_i, y_i));
            end
        end
        Upload training results w_i^{t+1};
    end
    Server aggregates parameters:
      w^{(t+1)} = (1/W) ( Σ_{k∈W_t\{j}} w_k^{(t)} + w_inwindow^{(t+1)} );
    Server sends w^{t+1} to u_outwindow;
end
```

B. Convergence Analysis

In this section, we provide convergence guarantees for FedWindow in a smooth, non-convex setting. In our federated learning framework, we operate under the following assumptions:

1) **Unbiased Stochastic Gradients:** For each client i, the stochastic gradient $g_i(w)$ is unbiased, i.e., $\mathbb{E}[g_i(w)] = \nabla F_i(w)$.
2) **Bounded Local Variance:** For each client i, the local variance is bounded above by $\mathbb{E}[\|g_i(w) - \nabla F_i(w)\|^2] \leq \sigma_l^2$.
3) **Bounded Global Variance:** The global variance is bounded above by $\mathbb{E}\left[\left\|\frac{1}{W}\sum_{i \in W_t \setminus \{j\}}(g_i(w) - \nabla F_i(w))\right\|^2\right] \leq \sigma_g^2$, where $W_t \setminus \{j\}$ denotes the set of all clients in the window excluding the client j that is about to leave.
4) **Bounded Gradients:** There exists a constant G such that $\|\nabla F_i(w)\| \leq G$ for all w and i.
5) **Lipschitz Continuous Gradients:** There exists a constant L such that $\|\nabla F_i(\mathbf{x}) - \nabla F_i(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|$ for all $\mathbf{x}$ and $\mathbf{y}$.

Utilizing the L-smooth property, for the global objective function F, we have:

$$\begin{aligned} F(w^{(t+1)}) \leq & F(w^{(t)}) + \left\langle \nabla F(w^{(t)}), w^{(t+1)} - w^{(t)} \right\rangle \\ & + \frac{L}{2}\|w^{(t+1)} - w^{(t)}\|^2. \end{aligned} \tag{5}$$

Due to the unbiasedness of the stochastic gradients, we can substitute $\nabla F(w^{(t)})$ with $\mathbb{E}[g_i(w^{(t)})]$. Subsequently, we can rewrite the inequality as:

$$\begin{aligned} \mathbb{E}[F(w^{(t+1)})] \leq & \mathbb{E}[F(w^{(t)})] + \mathbb{E}\left[\left\langle g_i(w^{(t)}), w^{(t+1)} - w^{(t)} \right\rangle\right] \\ & + \frac{L}{2}\mathbb{E}[\|w^{(t+1)} - w^{(t)}\|^2]. \end{aligned} \tag{6}$$

Now, we need to estimate the expected squared norm of the global gradient. Since $w^{(t+1)}$ is obtained by aggregating $w_i^{(t+1)}$ with the other clients' models within the window, we can bound this term using the local variance σ_l^2 and the global variance σ_g^2.

To dissect the update term $w^{(t+1)} - w^{(t)}$, it is imperative to consider the influence of individual client model updates on the global model parameters. This influence can be estimated by aggregating the updates from all participating clients. The global model parameter update difference can be decomposed as:

$$\begin{aligned} w^{(t+1)} - w^{(t)} = & \frac{1}{W} \sum_{k \in W_t \setminus \{j\}} (w_k^{(t+1)} - w_k^{(t)}) \\ & + \frac{1}{W}(w_i^{(t+1)} - w_i^{(t)}). \end{aligned} \tag{7}$$

We assume that $w_k^{(t)} = w^{(t)}$ for all clients at time t, meaning that the model parameters across clients are synchronized with the global parameters at this timestep. This is a simplifying assumption, as discrepancies may exist in practice.

Given the L-smooth property, we can relate the upper bound of the squared norm of the model parameter updates to the local and global variances. Specifically, we estimate the expected squared norm of the global model update difference as:

$$\mathbb{E}[\|w^{(t+1)} - w^{(t)}\|^2] = \mathbb{E}\left[\left\|\frac{1}{W}\sum_{k\in W_t\setminus\{j\}}(w_k^{(t+1)} - w_k^{(t)}) + \frac{1}{W}(w_i^{(t+1)} - w_i^{(t)})\right\|^2\right]. \tag{8}$$

With the assumption that each client starts with the same model parameters $w^{(t)}$, we replace $w_i^{(t+1)} - w_i^{(t)}$ with the previously defined $\Delta w_i^{(t)}$ to obtain:

$$\|w^{(t+1)} - w^{(t)}\|^2 = \left\|\frac{1}{W}\sum_{k\in W_t\setminus\{j\}}\Delta w_k^{(t)} + \frac{1}{W}\Delta w_i^{(t+1)}\right\|^2. \tag{9}$$

We now compute the expected squared norm of this difference. Utilizing the linearity of expectation and the unbiased nature of the stochastic gradients, that is $\mathbb{E}[g_i(w)] = \nabla F_i(w)$, we have:

$$\mathbb{E}[\|w^{(t+1)} - w^{(t)}\|^2] = \mathbb{E}\left[\left\|\frac{1}{W}\sum_{k\in W_t\setminus\{j\}}\Delta w_k^{(t)} + \frac{1}{W}\Delta w_i^{(t+1)}\right\|^2\right]. \tag{10}$$

Given that $\Delta w_k^{(t)}$ and $\Delta w_i^{(t+1)}$ are composed of stochastic gradients, we can decompose the expected squared norm into two parts: the average variance of local updates and the global variance. Employing the definition and properties of variance, we arrive at:

$$\mathbb{E}[\|w^{(t+1)} - w^{(t)}\|^2] = \frac{1}{W^2}\mathbb{E}\left[\left\|\sum_{k\in W_t\setminus\{j\}}\Delta w_k^{(t)}\right\|^2\right] + \frac{1}{W^2}\mathbb{E}[\|\Delta w_i^{(t+1)}\|^2]. \tag{11}$$

We now apply the local variance σ_l^2 and the global variance σ_g^2 to the above expressions: **Local Variance**: For each client i, $\mathbb{E}[\|\Delta w_i^{(t)}\|^2]$ can be constrained by the local variance upper bound σ_l^2, as $\Delta w_i^{(t)}$ is the sum of E independent stochastic gradients from client i. **Global Variance**: The global variance σ_g^2 bounds the average variance of all clients' updates.

Consequently, we derive:

$$\mathbb{E}[\|w^{(t+1)} - w^{(t)}\|^2] \leq \frac{\eta^2E^2\sigma_l^2}{W} + \frac{\eta^2E^2\sigma_g^2}{W^2}, \tag{12}$$

where we have utilized the variance upper bounds of the stochastic gradients and the aggregation of client updates to constrain the expected squared norm of the global parameter update difference.

We next address the inner product term in the midst. As the global model update is the aggregate of all clients' local updates, we exploit the unbiased nature of the stochastic gradients, i.e., $\mathbb{E}[g_i(w)] = \nabla F_i(w)$, to handle this term:

$$\mathbb{E}[\langle\nabla F(w^{(t)}), w^{(t+1)} - w^{(t)}\rangle] = \mathbb{E}[\langle\nabla F(w^{(t)}), -\eta\sum_{e=1}^{E}\bar{g}(w^{(t)})\rangle], \tag{13}$$

where $\bar{g}(w^{(t)})$ is the average global stochastic gradient at time t. This expectation can be further simplified to:

$$-\eta E\mathbb{E}[\|\nabla F(w^{(t)})\|^2], \tag{14}$$

since $\mathbb{E}[\bar{g}(w^{(t)})] = \nabla F(w^{(t)})$.

Incorporating these terms into the inequality provided by the L-smooth property, we obtain:

$$\mathbb{E}[F(w^{(t+1)})] \leq \mathbb{E}[F(w^{(t)})] - \eta E\mathbb{E}[\|\nabla F(w^{(t)})\|^2] + \frac{L}{2}\left(\frac{\eta^2E^2\sigma_l^2}{W} + \frac{\eta^2E^2\sigma_g^2}{W^2}\right). \tag{15}$$

Rewriting this inequality in terms of the gradient norm and summing over all rounds $t = 0$ to $T-1$, we obtain:

$$\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla F(w^{(t)})\|^2] \leq \frac{1}{\eta E}\sum_{t=0}^{T-1}\left(\mathbb{E}[F(w^{(t)})] - \mathbb{E}[F(w^{(t+1)})]\right) + \frac{TL\eta E\sigma_l^2}{2W} + \frac{TL\eta E\sigma_g^2}{2W^2}. \tag{16}$$

Thus, we arrive at:

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla F(w^{(t)})\|^2] \leq \frac{F(w^{(0)}) - F^*}{\eta ET} + \frac{L\eta E\sigma_l^2}{2W} + \frac{L\eta E\sigma_g^2}{2W^2}, \tag{17}$$

where F^* is assumed to be the minimal possible value of the objective function, denoted as $\mathbb{E}[F(w^{(T)})]$. This culminates in our final convergence result, which indicates that the average of the squared norms of the global gradients is bounded and decreases over time, signifying the convergence of our algorithm on average.

C. Conclusion

From the above analysis, we conclude that under the given assumptions, our proposed federated learning framework ensures that the average squared norm of the gradients on the global objective function is bounded. This implies that the algorithm will converge to a first-order stationary point as the number of iterations T increases. The convergence rate is influenced by the initialization, local and global variances, and the length of the sliding window W.

IV. Experiments

In this section, we evaluate the effectiveness of the proposed FedWindow framework and analyze the performance improvements. We commence by delineating the experimental simulation setup, followed by presenting simulation results under various network architectures and datasets to substantiate the efficacy of the proposed FedWindow framework. Subsequently, we delve into a comprehensive investigation of FedWindow.

A. Simulation Setup

Herein, we elucidate the simulation employed to assess the convergence of the proposed algorithm. The simulations were conducted on a computer equipped with an NVIDIA GeForce RTX 3060 Ti GPU and an Intel Core i5-12400F CPU @ 2.5 GHz. The software utilized for the simulation was written in Python 3.8.17 and employed PyTorch 1.12.1 for model construction and training. In the simulation, a ring topology comprising five clients was established, which then engaged in FedWindow with a server. The model performance of RingSFL was evaluated using different models and both IID and non-IID datasets.

In this study, we experimented with the widely-utilized MNIST and CIFAR10 datasets. Specifically, MNIST consists of 70,000 square (28x28=784 pixels) grayscale images of handwritten digits, divided into 10 categories (60,000 for training, 10,000 for testing); CIFAR10 is comprised of 60,000 square (3x32x32=3072 pixels) 3-channel color images, categorized into 10 classes (50,000 for training, 10,000 for testing). To validate the performance of our scheme, we conducted evaluations using mainstream networks, including ResNet18 and LeNet-5.

In our experiments, we utilized vanilla FL as a benchmark. For each experiment, including five clients, a total of 100 rounds of training were carried out. In each round, every client performed two epochs of iteration on their local dataset.

B. Results and Discussion

We first evaluated the convergence performance of FedWindow in the simulation environment, as well as the impact of window length on model accuracy.

1) Convergence Performance: In the simulated environment, we assessed the convergence performance of the FedWindow framework as well as the impact of the sliding window's length on the model's accuracy. This experiment was conducted with the aim of validating the convergence capabilities of FedWindow, using vanilla FL as the benchmark for comparison. The window lengths were set to 2, 3, and 4. As depicted in Figure 3, FedWindow achieved the highest model accuracy on the CIFAR10 (IID) dataset after the same number of communication rounds when compared to the considered benchmark, with the window length (W=2). For window lengths W = 3 and W = 4, the convergence accuracy slightly decreased but still surpassed that of vanilla FL. Moreover, Figure 4 illustrates the influence of different data distributions on the convergence performance of FedWindow when trained on the CIFAR10 (Non-IID) dataset. In the Non-IID setting, the performance was comparable to vanilla FL at a window length of (W=4), but the accuracy was lower at W = 2 and W = 3, which can be attributed to the reduced global information contained within shorter window lengths in the Non-IID dataset.

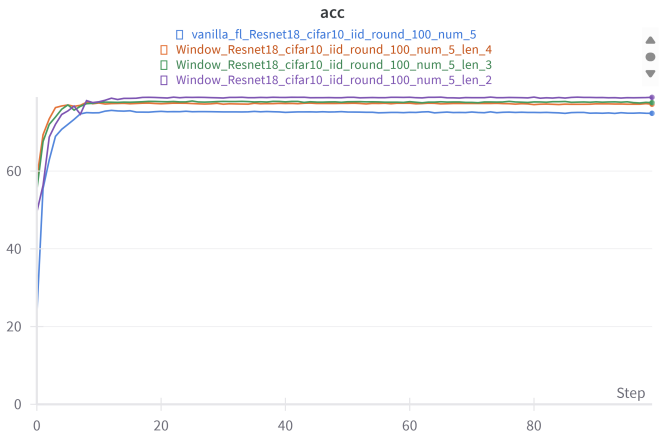

Fig. 3. Testing convergence of ResNet18 on CIFAR10 (IID) under different window length.

Fig. 4. Testing convergence of ResNet18 on CIFAR10 (Non-IID) under different window length.

2) Reduction in Convergence Time: Empirically, we have demonstrated that FedWindow exhibits a 1.5X increase in efficiency over competing synchronous FL algorithms, even without penalizing stragglers in synchronous FL setups. Furthermore, we have substantiated that FedBuff outperforms the closest asynchronous FL algorithm in the literature, FedAsync by 2.5X in terms of efficiency.

V. Conclusion

This paper presents FedWindow, a sliding window-based approach for model aggregation in Federated Learning, addressing the challenges of heterogeneous data distribution

TABLE I
AVERAGE (SPEEDUP) NUMBER OF CLIENT TRIPS.

Dataset	Accuracy	vanillafl	$W = 2$	$W = 3$	$W = 4$
MNIST	90% CIFAR-10	60%			

and network congestion. By incorporating dynamic client participation and a sliding window mechanism, FedWindow enhances the efficiency and stability of global model training. Our convergence analysis in a non-convex setting, along with extensive experimentation, demonstrates the robustness and effectiveness of FedWindow, outperforming traditional FL methods in terms of reduced delays and improved model performance. Future work will explore further optimization of the sliding window length and its application in more complex network environments.

ACKNOWLEDGMENT

This work was supported by the National Key Research and Development Program of China (2020YFB1807700), the National Natural Science Foundation of China (NSFC) under Grant No. 62071356, and the Fundamental Research Funds for the Central Universities under Grant ZYTS23175.

REFERENCES