

Northumbria Research Link

Citation: Luo, Yifan, Xu, Jindan, Xu, Wei and Wang, Kezhi (2021) Sliding Differential Evolution Scheduling for Federated Learning in Bandwidth-Limited Networks. IEEE Communications Letters, 25 (2). pp. 503-507. ISSN 1089-7798

Published by: IEEE

URL: <https://doi.org/10.1109/LCOMM.2020.3032517>
<<https://doi.org/10.1109/LCOMM.2020.3032517>>

This version was downloaded from Northumbria Research Link:
<http://nrl.northumbria.ac.uk/id/eprint/44873/>

Northumbria University has developed Northumbria Research Link (NRL) to enable users to access the University's research output. Copyright © and moral rights for items on NRL are retained by the individual author(s) and/or other copyright owners. Single copies of full items can be reproduced, displayed or performed, and given to third parties in any format or medium for personal research or study, educational, or not-for-profit purposes without prior permission or charge, provided the authors, title and full bibliographic details are given, as well as a hyperlink and/or URL to the original metadata page. The content must not be changed in any way. Full items must not be sold commercially in any format or medium without formal permission of the copyright holder. The full policy is available online: <http://nrl.northumbria.ac.uk/policies.html>

This document may differ from the final, published version of the research and has been made available online in accordance with publisher policies. To read and/or cite from the published version of the research, please visit the publisher's website (a subscription may be required.)



**Northumbria
University**
NEWCASTLE



UniversityLibrary

Sliding Differential Evolution Scheduling for Federated Learning in Bandwidth-Limited Networks

Yifan Luo, Jindan Xu, *Student Member, IEEE*, Wei Xu, *Senior Member, IEEE*,
Kezhi Wang, *Senior Member, IEEE*

Abstract

Federated learning (FL) in a bandwidth-limited network with energy-limited user equipments (UEs) is under-explored. In this paper, to jointly save energy consumed by the battery-limited UEs and accelerate the convergence of the global model in FL for the bandwidth-limited network, we propose the sliding differential evolution-based scheduling (SDES) policy. To this end, we first formulate an optimization that aims to minimize a weighted sum of energy consumption and model training convergence. Then, we apply the SDES with parallel differential evolution (DE) operations in several small-scale windows, to address the above proposed problem effectively. Compared with existing scheduling policies, the proposed SDES performs well in reducing energy consumption and the model convergence with lower computational complexity.

Index Terms

Federated learning (FL), sliding window, differential evolution (DE), scheduling policy, bandwidth-limited networks.

I. INTRODUCTION

IN future wireless networks, by building and utilizing the computation capability in edge nodes, e.g., access points (APs), edge networks can be established and are able to conduct complex task via intelligent scheduling and processing [1]. Several works utilizing machine learning have been proposed for future communications, e.g., C-RAN [2], MIMO channel information feedback system [3] and multi-antenna quantization [4]. However, massive raw data generated by user devices triggers two key problems, i.e., privacy disclosure and high cost from data transmission, making the above-mentioned intelligent applications, difficult to process in wireless networks. Federated learning (FL) has been proposed by Google, as a promising machine learning (ML) technology to solve the above problems [5].

Specifically, there are two key challenges for the deployment of FL in wireless networks. On one hand, local data samples in UEs are diversely distributed, i.e., non-independent and identically distributed (non-IID) and unbalanced [5], [6]. To strike a balance between computational efficiency and convergence, Google proposed a novel FL architecture, referred to as FedAvg [5]. Additionally, other researchers, e.g. [7], tried to accelerate the convergence by converting the optimization problem into sub-problems. Another challenge is that the limited capacity of communications, e.g., limited bandwidth. To address it, scheduling policy-aided FL architectures were utilized in [8], [9], [10]. The authors in [10] adopted the age of update (AoU) as the scheduling policy to accelerate model convergence in mobile edge networks. Also, the authors in [9] proposed a bandwidth resource scheduling policy for FL in wireless networks. Their scheduling policies only take FL model convergence into consideration [10] or just adopt randomly selection [8].

However, there is little literature related to joint energy efficiency and model convergence for federated learning in wireless communication. The authors in [7] carefully analysed the trade-off between energy consumed by UEs and FL convergence with no bandwidth-limited constrains. The authors in [9] proposed an energy-efficient bandwidth resource scheduling policy for FL in wireless networks, and it only considered the communication energy cost.

Against the above background, in this paper, we aim to save the energy consumption of the UEs and improve the convergence performance of FL in a bandwidth-limited network. We propose an efficient sliding differential evolution-based scheduling (SDES) with lower computational complexity compared with the existing methods. To the best of our knowledge, it is the first time to solve the above problem well. In detail, we introduce a convergence reference (CR) of the overall training model and propose the SDES policy to reduce the energy consumption and accelerating model convergence, by choosing the optimal subgroup of the UEs. Compared to

Y. Luo, J. Xu are with the National Mobile Communications Research Laboratory (NCRL), Southeast University, Nanjing 210096, China (email: {213161316, jdxu}@seu.edu.cn).

W. Xu is with the National Mobile Communications Research Lab, Southeast University, Nanjing 210096, China, and also with Purple Mountain Laboratories, Nanjing 211111, China (wxu@seu.edu.cn).

K. Wang is with Department of Computer and Information Sciences, Northumbria University, Newcastle upon Tyne NE1 8ST, U.K. (e-mail: kezhi.wang@northumbria.ac.uk).

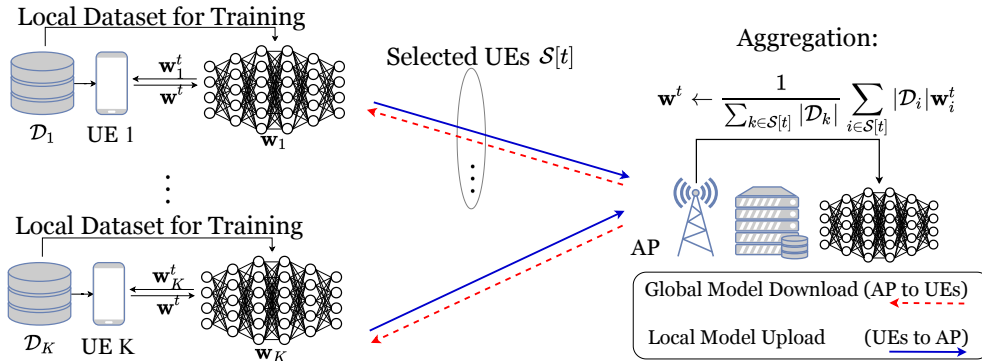


Fig. 1: Federated learning in wireless networks

conventional mathematical iterative tools, the proposed SDES can process the computational tasks in a parallel model. Experiment verifies the effectiveness of the proposed solution, in terms of both energy saving and convergence acceleration in the bandwidth-limited network.

II. SYSTEM MODEL

We consider a FL system as shown in Fig. 1, where a set \mathcal{K} of K UEs are connected to one AP. Each UE k stores a local dataset \mathcal{D}_k , with its size denoted by $D_k = |\mathcal{D}_k|$. Thus, the whole data size equals to $D = \sum_{k=1}^K D_k$. Dataset \mathcal{D}_k denotes the collection of data samples in the form of input-output pairs as $\{\mathbf{x}_i^{(k)}, y_i^{(k)}\}_{i=1}^{D_k}$, where $\mathbf{x}_i^{(k)} \in \mathbb{R}^d$ is an input sample vector with d features, and $y_i^{(k)} \in \mathbb{R}$ is the labeled output value for sample $\mathbf{x}_i^{(k)}$. Considering the user preference, different \mathcal{D}_k 's are non-IID and their corresponding data size D_k varies.

A. Model Convergence

The goal of AP is to learn a statistical model over the data that resides on the K associated UEs. Mathematically, AP needs to fit the model parameter $\mathbf{w}^t \in \mathbb{R}^d$ which characterizes the output y_i , by minimizing a particular loss function $f_i(\mathbf{w}^t) = \ell(\mathbf{x}_i^{(k)}, y_i^{(k)}; \mathbf{w}^t)$ in the t -th communication round. Formally, the loss function on the dataset of UE k is

$$F_k(\mathbf{w}^t) := \frac{1}{D_k} \sum_{i \in \mathcal{D}_k} f_i(\mathbf{w}^t). \quad (1)$$

Then, the global loss function minimization problem in AP can be expressed as

$$\min_{\mathbf{w}^t} F(\mathbf{w}^t) := \sum_{k=1}^K \frac{D_k}{D} F_k(\mathbf{w}^t). \quad (2)$$

To protect the user privacy, UE k only exchanges its model parameters \mathbf{w}_k^t with AP.

B. Energy Consumption

In bandwidth-limited systems, the number of UEs, K , far exceeds the number of subchannels, N . Only a small portion of UEs, referred to as the updating set $\mathcal{S}[t]$, are selected for participating in the t -th communication round. $\mathcal{S}[t] = \{k \mid S_k[t] = 1, k = 1, 2, \dots, K\}$, where $S_k[t] = 1$ implies that UE k is in the updating set $\mathcal{S}[t]$, otherwise $S_k[t] = 0$. The energy consumed by UEs in the updating set $\mathcal{S}[t]$ consists of two components, i.e., transmitting energy consumption and computing energy consumption.

In fact, all UEs share the same model with their local parameters, and we use constant J to denote its size of \mathbf{w}^t . We assume in the t -th communication round, UE k is assigned with the n -th subchannel with channel gain $h_{k,n}$ and bandwidth B_n . Then, the achievable data rate of UE k can be

$$r_k = B_n \ln \left(1 + \frac{h_{k,n}^2 P_{k,n}}{N_0} \right), \quad (3)$$

where $P_{k,n}$ denotes the corresponding power allocation, and N_0 denotes the variance of the white Gaussian noise.

To obtain minimal transmit power, we assume the achievable rate r_k equals to the threshold transmission rate R_k , and the energy for signal transmission for UE k is formulated as

$$E_{k,TP} = \tau_k \cdot P_{k,n} = \frac{J}{R_k} \cdot \frac{N_0}{h_{k,n}^2} \left(e^{\frac{R_k}{B_n}} - 1 \right), \quad (4)$$

where τ_k is time duration of the signal transmission process.

On the other hand, the computing energy consumed by UE k to train its local model can be written as

$$E_{k,\text{CP}} = \sum_{i=1}^{c_k D_k} \frac{\alpha_k}{2} f_k^2 = \frac{\alpha_k}{2} c_k D_k f_k^2, \quad (5)$$

where c_k denotes the number of CPU cycles for executing one sample of data; $c_k D_k$ denotes the number of CPU cycles in one local round; and f_k is the CPU-cycle frequency. Then, the total energy consumption of UEs in the t -th communication round is

$$E_P[t] = \sum_{k=1}^K S_k[t](E_{k,\text{TP}} + \kappa E_{k,\text{CP}}), \quad S_k[t] \in \{0, 1\} \quad (6)$$

where κ is the number of local rounds for local model training.

C. Problem Formulation

We aim to minimize the weighted sum of global loss function in (2) and the total energy consumption in (6) in the t -th communication round as:

$$\min_{S[t]} F(\mathbf{w}^t) + \zeta E_P[t], \quad t \in \{0, 1, \dots, T\} \quad (7a)$$

$$\text{s.t. } S_k[t] \in \{0, 1\}, \quad \forall k \in \{1, 2, \dots, K\} \quad (7b)$$

$$\sum_{k=1}^K S_k[t] = N, \quad (7c)$$

where ζ is the factor to balance the loss function and the energy consumption; T denotes the number of communication rounds between AP and UEs; constraint (7c) shows that the number of the available sub-channels is N .

III. SLIDING DIFFERENTIAL EVOLUTION BASED SCHEDULING

There are two challenges for solving the optimization problem in (7). On one hand, due to the limited bandwidth, only a small portion of UEs' training loss and model parameters can be updated to AP. This makes it impossible to calculate the global loss $F(\mathbf{w}^t)$ in (7a) accurately. Also, it is difficult to get the relationship between $S[t]$ and \mathbf{w}^t in (7), where \mathbf{w}^t relies on model training. On the other hand, this optimization problem is a combinatorial problem which does not normally have low complexity solutions. For instance, in the case of 100 UEs and 25 available sub-channels, $C(100, 25) \approx 2.4 \times 10^{23}$ searches are needed in the exhaustive searching, where $C(n, k) = \frac{n!}{k!(n-k)!}$ is the searching number of k -combinations from a given set of n elements.

In this section, we first introduce the convergence reference (CR) function to replace $F(\mathbf{w}^t)$ for model convergence above, where two parameters in (7a) are unified into one set, $S[t]$. Based on CR value and energy consumption expression, we propose the SDES policy, for solving (7) efficiently.

A. Convergence Reference (CR) Function

To solve the aforementioned two challenges, we propose the concept of convergence reference (CR). In CR function, we collect the useful data from the updated UEs for model convergence, and utilize CR function to improve the model convergence based on these data. The CR function transforms the convergence problem into finding the optimal updated set $S[t]$, and it is consistent with the energy problem in (7a).

We introduce CR function based on staleness-loss (SL) measure for convergence, where CR value is utilized to select the optimal subgroup of users. We re-formulate the convergence performance of local models into SL measure based on two existing methods, i.e., staleness and training loss method.

The staleness method, also referred to as AoU [10] can transform convergence value into model training times. It records the duration $T_k[t]$ which the UE k uploading its model in the t -th round, written as $T_k[t] = (T_k[t-1] + 1)(1 - S_k[t-1])$. Staleness method leverages $T_k[t]$ to avoid over-training and under-training [11].

Moreover, the training loss method records the training loss for all the UEs, where the training loss for UE k is $\mathcal{L}_k[t] = F_k(\mathbf{w}^t)$ in the t -th communication round in (1).

Based the above two methods, we introduce staleness-loss (SL) measures, by combining the staleness and training loss. The set of SL value for all the UEs in the t -th communication round can be written as:

$$\mathcal{C}[t] = \{C_1[t], C_2[t], \dots, C_k[t], \dots, C_K[t]\}$$

where $C_k[t] = T_k[t]\mathcal{L}_k[t]$.

Then, we introduce the convergence reference (CR) function based on SL value as

$$T_L[t] = \frac{(\sum_{k=1}^K D_k V_k[t] S_k[t])^{1-\beta}}{1-\beta}, \quad (8)$$

where $\beta \in (0, 1)$ is a constant to adjust the sensitivity to the value change of $\sum_{k=1}^K D_k V_k[t] S_k[t]$, and D_k is the size of user data. $V_k[t] \in \{T_k[t], \mathcal{L}_k[t], C_k[t]\}$ denotes the method used in CR function. Then, in the t -th communication round, the objective in (7a) can be re-written as:

$$\min_{S[t]} -T_L[t] + \zeta E_P[t], \quad t \in \{0, 1, \dots, T\} \quad (9a)$$

(7b), (7c)

B. The Sliding Differential Evolution (SDE) Concept

The optimization problem in (9) is NP-hard. Differential evolution (DE) [12] is a common method to solve the above kind of problem, where DE generates M individuals with the chromosome scale, K , i.e., the dimensionality of (9a), for each individual. We assume there are G_{DE} generations in DE, and DE algorithm terminates after exceeding G_{DE} iterations. The execution time is proportional to the objective function evaluation $c(K)$ of (9) [13] with K dimensionality, and the number of elementary operations is proportional to the maximal iteration number G_{DE} and the population size, i.e., $\mathcal{O}(c(K) \cdot M \cdot G_{DE})$. However, traditional DE methods suffer from heavily computational complexity when the scale of (9) increases. Therefore, we propose the concept of sliding differential evolution (SDE) to decrease the computational complexity by reducing the scale of chromosomes from K to W , and the number of generations from G_{DE} to G_{SDES} with parallel computation, where W is the length of energy windows and G_{SDES} is the number of generations in SDE. Consequently, its complexity can be given by $\mathcal{O}(c(W) \cdot M \cdot G_{SDES})$.

C. Sliding Differential Evolution-based Scheduling (SDES)

The sliding differential evolution-based scheduling (SDES) algorithm is shown in Algorithm 1 and its process is summarized in Fig. 2. SDES takes the steps as follows:

- **a)** Energy windows generation: We first leverage W -length SW to generate $K - W + 1$ energy windows, where $N \leq W \leq K$. Specifically, we first sort all UEs according to their energy consumption from small to large, and then align the head of the SW with the first UE to select the first W UEs in one window. Similarly, we slide the window to the end of the queue for another $K - W$ energy windows.
- **b)** Alternative individuals evolution: In each window, we utilize DE to evolve one alternative individual, i.e., one scheduling scheme with minimal value of (9a) from the W UEs, and the scale of chromosome scale is W . We conduct DE operations in $K - W + 1$ energy windows parallelly, and generate $K - W + 1$ alternative individuals.
- **c)** Optimal solution selection: We select the optimal individual, i.e., the best solution of (9a) from the $K - W + 1$ alternative individuals.

For DE operations in each energy window in the above **Step b)**, we define the number of generations as $G_{SDES} = \min\{\lceil \frac{C(K,W)}{M} \rceil, G_{DE}\}$, where M is the number of individuals and G_{DE} is the number of evolution generations in the traditional DE algorithm. Each individual $\mathbf{x}_i^{(g,w)}$ represents one solution of (9a). For instance, in the w -th energy window, the agent first generates the initial population $P^{0,w} = \{\mathbf{x}_1^{(0,w)}, \mathbf{x}_2^{(0,w)}, \dots, \mathbf{x}_M^{(0,w)}\}$. $\mathbf{x}_i^{(0,w)} = \{x_{i,1}^{(0,w)}, x_{i,2}^{(0,w)}, \dots, x_{i,W}^{(0,w)}\}$ meets the constrains of $S_k[t]$ in (7), where (7c) is rewritten as $\sum_{j=1}^W x_{i,j}^{(0,w)} = N$. Any individual violating the constraint of (7) is abandoned. Then each individual $\mathbf{x}_i^{(g,w)}$ from the g -th generation in the w -th energy window generates the offspring with three process, given as

- **Mutation:** We choose three individuals from $P^{g,w}$ via roulette wheel selection (RWS) to generate $\mathbf{v}_i^{(g,w)}$.
- **Crossover:** We cross the current individual $\mathbf{x}_i^{(g,w)}$ with $\mathbf{v}_i^{(g,w)}$ and then generate $\mathbf{u}_i^{(g,w)}$.
- **Selection:** We choose the appropriate offspring between $\mathbf{u}_i^{(g,w)}$ and $\mathbf{v}_i^{(g,w)}$ by comparing their fitness value.

The RWS in the process of mutation associates the probability of selecting individual x with the fitness function, as $p(x) = \frac{Q(x)}{\sum_{j=1}^M Q(j)}$.

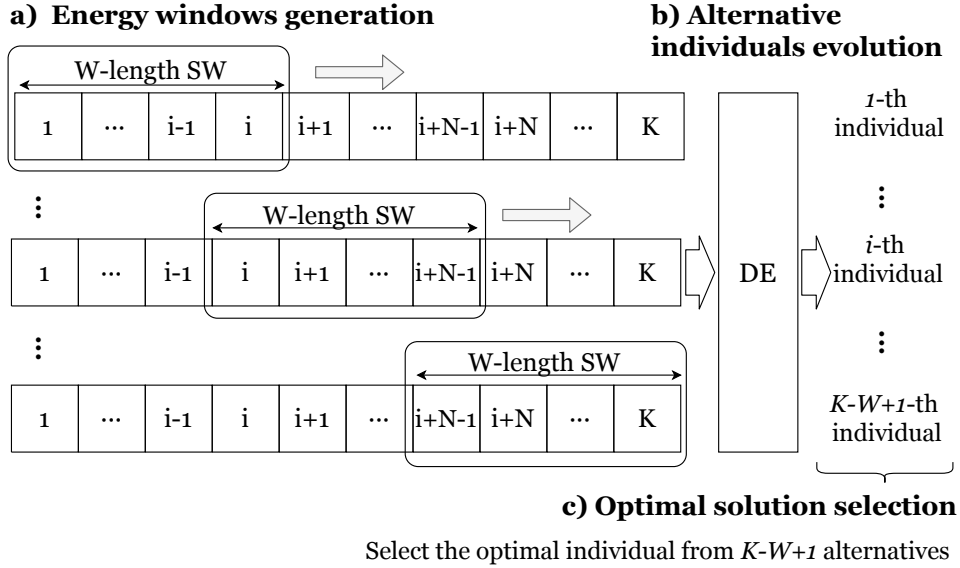


Fig. 2: Sliding differential evolution algorithm

Algorithm 1 SDES

Parameters: W : the length of SW of (9a); M : the size of populations; f_{CR} : the crossover rate; F : the selection weighting factor; G_{SDES} : the number of generations

Input: Optimization problem (9a); $K - W + 1$ energy windows

1: **for** $w = 1, 2, \dots, K - W + 1$ **do** %% Step a

 Initialization:

2: Generate Initial population $P^{0,w}$ with M individuals

3: **for** $g = 0, 1, \dots, G - 1$ **do** %% Step b

4: Calculate the fitness value $Q(\cdot)$ of Generation $p^{g,w}$

5: **for** each individual $\mathbf{x}_i^{(g,w)}$ in Generation $p^{g,w}$ **do**

 (a) Mutation:

6: Select three individuals $r1 \neq r2 \neq r3$ via RWS

7: $\mathbf{v}_i^{(g,w)} = \mathbf{x}_{r1}^{(g,w)} + F \cdot (\mathbf{x}_{r2}^{(g,w)} - \mathbf{x}_{r3}^{(g,w)})$

 (b) Crossover:

8: $\mathbf{u}_i^{(g,w)} = \mathbf{x}_i^{(g,w)}$

9: j randomly selected from $\{1, 2, \dots, D\}$, $L = 1$

10: **repeat**

11: $u_{i,j}^{(g,w)} = v_{i,j}^{(g,w)}$

12: $j = (j + 1)$ modulo K

13: $L = L + 1$

14: **until** $\text{rand}(0, 1) < f_{CR}$ and $L < D$

 (c) Selection:

15: **if** $Q(\mathbf{u}_i^{(g,w)}) \geq Q(\mathbf{x}_i^{(g,w)})$ **then**

16: add $\mathbf{u}_i^{(g,w)}$ in the next generation $P^{g+1,w}$

17: **else** add $\mathbf{x}_i^{(g,w)}$ in the next generation $P^{g+1,w}$

18: Add the best individual in the population $P^{G,w}$ into the alternative individual list

Output: The optimal individual from $K - W + 1$ alternative individuals

%%

Step c

The fitness function $Q(\cdot)$ is transformed from the optimization objective in (9a) via linear scaling as follows

$$Q(\mathbf{x}_i^{(g,w)}) = \alpha_1 \cdot O(\mathbf{x}_i^{(g,w)}) + \beta_1 \quad (10a)$$

$$\text{where } O(\mathbf{x}_i^{(g,w)}) = -(-T_L[t] + \zeta E_P[t]) \Big|_{S_k[t] \in \mathbf{x}_i^{(g,w)}} \quad (10b)$$

$$\alpha_1 = \frac{O_{\text{avg}}}{O_{\text{avg}} - O_{\text{min}}}, \quad O_{\text{avg}} = \frac{1}{M} \sum_{j=1}^M \mathbf{x}_j^{(g,w)} \quad (10c)$$

$$\beta_1 = \frac{-O_{\text{min}} O_{\text{avg}}}{O_{\text{avg}} - O_{\text{min}}}, \quad O_{\text{min}} = \min_{j=1, \dots, M} \mathbf{x}_j^{(g,w)} \quad (10d)$$

TABLE I: Parameter Settings

Symbol	Parameters	Value
N, K	Number of subchannels, UEs	25, 100
N_0, B, R	Noise, bandwidth, threshold transmission rate	10^{-8} W, 10Mbps, 500Kbps
η, J, D	Learning rate, model and data size	0.1, 86.6 KB, 47.04 MB
f, α, C	CPU frequency, capacitance coefficient, cycles to execute	2GHz, $2 * 10^{-28}$, 20 cycle/bit
$L(d_{k,n})$	Path loss of Rayleigh fading	$99.3 + 20 \log d_{k,n}$
$d_{k,n}$	Distribution of UE k	Uniform in [5,50] m

and (10b) takes the reverse direction of (9a) for minimization.

D. Two Cases of SDES: $W=K$ and $W=N$

The computational complexity of DE algorithm is $\mathcal{O}(c(K) \cdot M \cdot G_{\text{DE}})$, while the one for SDES is $\mathcal{O}(c(W) \cdot M \cdot G_{\text{SDES}})$. where $G_{\text{SDES}} = \min\{\lceil \frac{c(K,W)}{M} \rceil, G_{\text{DE}}\}$. Considering $W \leq K$ and $G_{\text{SDES}} \leq G_{\text{DE}}$ where two equations all reach only if $W = K$, SDES can decrease the computational complexity compared with DE. However, the performance of scheduling policy generated by SDES is decreased when W reduces.

To investigate the stability of SDES, we analyse two cases of SDES, i.e., $W=K$ and $W=N$. More specifically, when $W = K$, there is only one energy windows in SDES, and the SDES algorithm can generate the best solution of (9a) at the highest computational cost. When $W=N$, all the UEs in one energy window are selected as the scheduling policy, and there is no need to generate policies by DE, where SDES generates the worst solution with the lowest computational cost.

IV. SIMULATION RESULTS

In this simulation, we adopt the orthogonal frequency division multiple access (OFDMA) system, and the details of the system are summarized in Table I. We assume all N sub-channels share the same bandwidth of $\frac{B}{N}$.

In FL training, the task is to classify handwritten digits using the MNIST dataset. In detail, the dataset distribution over UEs are unbalanced and non-i.i.d, where the unbalanced feature means that the dataset size varies greatly between different UEs. The training model is a 6-layer convolutional neural network (CNN), consisting of two 5×5 convolution layers with rectified linear unit (ReLU) activation. The two convolution layers have 10 and 20 channels respectively, and each layer has 2×2 max pooling, a fully-connected layer with 50 units and ReLU activation, and a log-softmax output layer.

Next, we validate the overall performance of SDES with the respect of energy saving and model convergence, through CR function in (8), where SDES ($W=K$) and SDES ($W=N$) are examined. The measure $V_k[t]$ in CR function can be selected from $\{T_k[t], \mathcal{L}_k[t], C_k[t]\}$. We adopt the FedAvg from Google [5] as the benchmark and set $\zeta = 5$. In detail, when the weight factor $\zeta > 5$, SDES will focus more on energy saving, and consequently improve energy efficiency, however, at the expense of worse convergence performance. When $\zeta < 5$, the model convergence gets improved, and the performance of energy efficiency will deteriorate.

Fig. 3 shows the performance gain of the proposed $C_k[t]$ measure in (8), compared with staleness $T_k[t]$ and training loss $\mathcal{L}_k[t]$. One can see that in Fig. 3(a) and (b), SDES with $C_k[t]$ achieves good convergences similar to the optimal solution (i.e., FedAvg) which only considers the model convergence. FedAvg may often train the models of UEs with bad channel condition or with large dataset size, making it suffering from huge energy expense. Both of the cases converge fast at the beginning and also has good performance towards the end. In Fig. 3(c) and (d), we compare the cumulative energy consumption among three measures. SDES with $\mathcal{L}_k[t]$ has the lowest cumulative energy consumption but with poor convergence performance, as shown in Fig. 3(a). Moreover, SDES ($W=K$) and ($W=N$) with $C_k[t]$ have the first and second best performance in energy saving, as they consider both parameters of $\mathcal{L}_k[t]$ and $T_k[t]$. Fig. 3(e) and (f) shows that SDES achieves the best performance in energy conservation, and SDES with $C_k[t]$ and $W = K$ have the similar convergence performance as FedAvg in a more intuitive way.

Fig. 4 further analyses the instant performance of SDES ($W=K$) and SDES ($W=N$) in terms of energy saving, where $C_k[t]$ in CR function is applied. One can see that both cases have good performance in energy saving compared with FedAvg. Moreover, one sees that the performances of SDES ($W=K$) with respect to model convergence in Fig. 3 and energy saving in Fig. 4(a) are better than those of SDES ($W=N$). This is because high computational resource is required in case of ($W=K$), as explained in Section III.D.

The proposed SDES can be extended to more general cases, where the UEs are mobile with time-varying channels or several APs are deployed in FL. In the former case, the energy consumption of UEs is constantly changing. In the latter case, UEs send the trained models to the appropriate APs considering the channel condition, and APs then centralize all the data into one AP for the global model training. Compared with the benchmark solution of FedAvg, SDES bears acceptable computational complexity in the real-time application. Moreover, the choice of weight factor should be careful, since bad choice may lead to unacceptable model convergence performance.

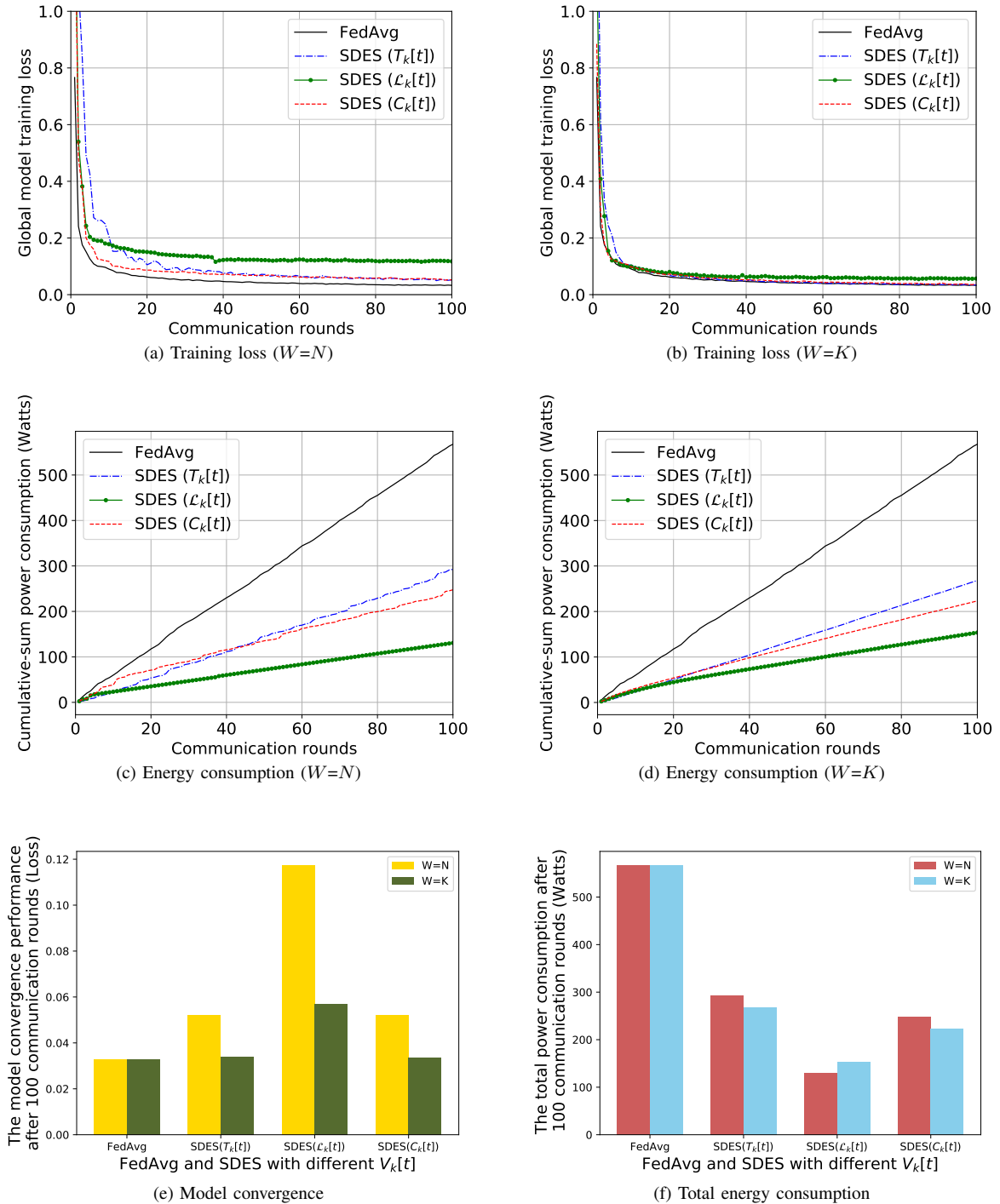


Fig. 3: SDES of $V_k[t] \in \{T_k[t], \mathcal{L}_k[t], C_k[t]\}$ ($\beta = 0.7, \zeta = 5$)

V. CONCLUSION

In this paper, we have proposed a novel energy-efficient scheduling policy, i.e., SDES for federated learning in bandwidth-limited systems with energy-limited UEs. We have utilized the CR function for model convergence and introduced the SDES algorithm, which can reduce the computational complexity with parallel computing architecture. Simulation shows that our proposed SDES performs well in model convergence, and it can save energy consumed by UEs significantly compared with the benchmark solution in bandwidth-limited networks. In the future, we will focus on the energy efficiency in the more practical federated learning cases in wireless communication, where the dataset contains complicated real information and the UE size is extended to thousands scale.

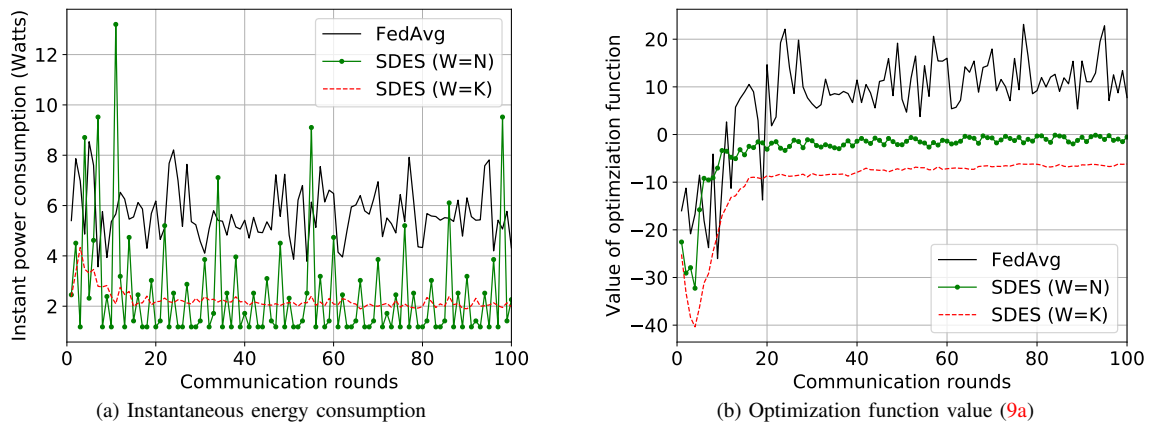


Fig. 4: SDES ($W \in \{K, N\}$) ($\beta = 0.7$, $\zeta = 5$, $V_k[t] = C_k[t]$)

REFERENCES

- [1] J. Park, S. Samarakoon, M. Bennis, and M. Debbah, "Wireless network intelligence at the edge," *Proceedings of the IEEE*, vol. 107, no. 11, pp. 2204–2239, 2019.
- [2] Y. Luo, J. Yang, W. Xu, K. Wang, and M. Di Renzo, "Power consumption optimization using gradient boosting aided deep q-network in c-rans," *IEEE Access*, vol. 8, pp. 46 811–46 823, 2020.
- [3] C. Lu, W. Xu, H. Shen, J. Zhu, and K. Wang, "MIMO channel information feedback using deep recurrent network," *IEEE Communications Letters*, vol. 23, no. 1, pp. 188–191, 2018.
- [4] C. Lu, W. Xu, S. Jin, and K. Wang, "Bit-level optimized neural network for multi-antenna channel quantization," *IEEE Wireless Communications Letters*, vol. 9, no. 1, pp. 87–90, 2020.
- [5] H. B. McMahan, E. Moore, D. Ramage, S. Hampson *et al.*, "Communication-efficient learning of deep networks from decentralized data," *arXiv preprint arXiv:1602.05629*, 2016.
- [6] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv preprint arXiv:1806.00582*, 2018.
- [7] C. Dinh, N. H. Tran, M. N. Nguyen, C. S. Hong, W. Bao, A. Zomaya, and V. Gramoli, "Federated learning over wireless networks: Convergence analysis and resource allocation," *arXiv preprint arXiv:1910.13067*, 2019.
- [8] H. H. Yang, Z. Liu, T. Q. Quek, and H. V. Poor, "Scheduling policies for federated learning in wireless networks," *IEEE Transactions on Communications*, 2019.
- [9] Q. Zeng, Y. Du, K. K. Leung, and K. Huang, "Energy-efficient radio resource allocation for federated edge learning," *arXiv preprint arXiv:1907.06040*, 2019.
- [10] H. H. Yang, A. Arafa, T. Q. Quek, and H. V. Poor, "Age-based scheduling policy for federated learning in mobile edge networks," *arXiv preprint arXiv:1910.14648*, 2019.
- [11] J. Chen and X. Ran, "Deep learning with edge computing: A review," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1655–1674, 2019.
- [12] R. Storn, "On the usage of differential evolution for function optimization," *Proceedings of North American Fuzzy Information Processing*, pp. 519–523, 1996.
- [13] K. R. Opara and J. Arabas, "Differential evolution: A survey of theoretical analyses," *Swarm and evolutionary computation*, vol. 44, pp. 546–558, 2019.