A Joint Communication and Federated Learning Framework for Internet of Things Networks

Zhaohui Yang, Guangyu Jia, Mingzhe Chen, Hak-Keung Lam, *Fellow, IEEE*, Kai-Kit Wong, *Fellow, IEEE*, Shuguang Cui, *Fellow, IEEE*, and H. Vincent Poor, *Life Fellow, IEEE*

***Abstract*—In this paper, a communication efficient Federated learning (FL) framework is proposed for Internet of things networks. To reduce the FL transmission delay, a joint learning and resource allocation problem is formulated via optimizing the transmit power of each device, time allocation, and user selection. To solve this problem, its objective function is first converted into a tractable form. Next, a successive convex optimization method is used to solve the simplified optimization problem. Two simulations are conducted using the electromyographic signals for finger movement detection, in which the results demonstrate that the proposed FL framework with the personalized training process is able to provide high performance in detecting single and combined finger movements for distributed users. Specifically, over 98% overall test accuracy is achieved using the proposed FL framework on two benchmark datasets, which surpasses the conventional learning framework by 1.6% and 0.5% on average.**

***Index Terms*—Federated learning, deep learning, Electromyography (EMG) signals**

I. Introduction

The accurate classification of finger movements based on Electromyographic (EMG) signals plays an important role in a number of industrial and healthcare applications in Internet of things networks, such as movement intention detection [1], grasp recognition [2], dexterous prostheses control [3], etc. EMG signals are generally collected through wearable skin sensors to record and evaluate the electrical activity of muscle. As the data can be collected individually by users in different places and at different times, it is in essence a form of isolated data islands. Meanwhile, EMG signals of different subjects vary greatly due to the difference of muscle density, muscular habits, etc. For this type of highly personalized data, the static learning framework struggles to reach the demanding requirements of personalized healthcare and privacy protection.

Z. Yang, G. Jia, and H.K. Lam are with the Department of Engineering, King's College London, WC2R 2LS, UK. Emails: yang.zhaohui@kcl.ac.uk, guangyu.jia@kcl.ac.uk, hak-keung.lam@kcl.ac.uk.

M. Chen and H. V. Poor are with the Department of Electrical and Computer Engineering, Princeton University, NJ, 08544, USA. Email: mingzhec@princeton.edu, poor@princeton.edu.

K. Wong is with the Department of Electronic and Electrical Engineering, University College London, London, United Kingdom. Email: kai-kit.wong@ucl.ac.uk.

S. Cui is with Shenzhen Research Institute of Big Data and School of Science and Engineering, the Chinese University of Hong Kong, Shenzhen, 518172, China. Email: robert.cui@gmail.com.

This work was partly supported by King's College London, the China Scholarship Council, the U.S. National Science Foundation under Grant CCF-1908308, and has been performed using resources provided by the Cambridge Tier-2 system operated by the University of Cambridge Research Computing Service funded by EPSRC Tier-2 capital grant EP/P020259/1.

To satisfy the requirements mentioned above, one can use federated learning (FL) to train personalized local models based on each user's data. However, using conventional FL methods that require each local device to share all parameters of the local model with the server leads to several issues. Firstly, conventional FL methods perform poorly on the EMG datasets as the data of various subjects have distinct probability distributions. Secondly, classification models with a large number of parameters will significantly increase the FL model parameter transmission and computational delay. In order to solve these problems and keep the local models personalized, a novel FL framework must be developed.

The authors in [4] designed a type of federated transfer learning framework which combined FL and transfer learning. The basic idea was to obtain the parameters of the cloud model and perform transfer learning on the user side. More specifically, all the convolutional blocks obtained from the cloud model were kept frozen and only the parameters of the fully connected layers were updated with a stochastic gradient descent algorithm.

However, the designed FL in [4] requires pre-training of the cloud model based on a large amount of user data. Therefore, it is not cost-effective and feasible in many applications where the data is insufficient for cloud model pre-training and when a quick training process is desired in emergent situations. In light of these issues, this paper designs an FL framework which does not require the pre-training of the cloud model. Meanwhile, in the designed FL, each local model is trained using its own data. The users and the cloud only need to exchange the FL model parameters in high-level layers of deep learning models. In addition, through properly optimizing the resource allocation for wireless communication, the performance of the FL framework can be further improved.

The rest of this paper is arranged as follows. Section II illustrates the system model and problem formulation. The algorithm design is shown in Section III. The application of the proposed FL for finger movement detection is illustrated in Section IV. Section V draws the conclusion.

II. System Model and Problem Formulation

Consider a cellular network that consists of one base station (BS), N access points (APs), and K users, as shown in Fig. 1. Denote the sets of APs and users as $\mathcal{N} = \{1, \cdots, N\}$ and $\mathcal{M} = \{1, \cdots, M\}$, respectively. Every AP can serve as a relay for some users. Denote $\mathcal{J}_i$ as the specific set of users served by AP $i \in \mathcal{N}$, and then $\bigcup_{i\in\mathcal{N}} \mathcal{J}_i = \mathcal{M}$.

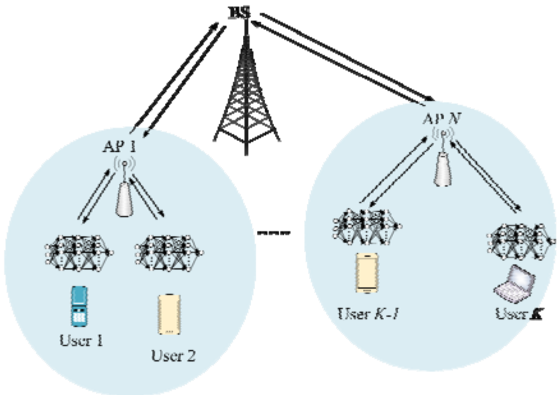

Fig. 1: Illustration of the considered model for the FL framework

A. Machine Learning Model

FL enables the BS and users to collaboratively learn a shared model while keeping all the training data at the device of each user. In an FL algorithm, each user will use its collected training data to train an FL model. Hereinafter, the FL model that is trained on the device of each user (using the data collected by the user itself) is called the *local FL model*. The BS aims to integrate the local FL models and generate a shared FL model. This shared FL model is used to improve the local FL model of each user so as to enable the users to collaboratively perform a learning task without sharing private data. The FL model, generated by the BS using the local FL models of the users, refers to the *global FL model*. The *uplink* from the users to the APs and from the APs to the BS are used to transmit the parameters related to the local FL model while the *downlink* is used to transmit the parameters related to the global FL model.

In our model, each user j collects a matrix $\boldsymbol{X}_j = \left[\boldsymbol{x}_{j1}, \ldots, \boldsymbol{x}_{jK_j}\right]$ of input data, where K_j is the number of the samples collected by each user j and each element $\boldsymbol{x}_{jk}$ is an input vector of the FL algorithm. The size of $\boldsymbol{x}_{jk}$ depends on the specific FL task. Let y_{jk} be the output of $\boldsymbol{x}_{jk}$. The output data vector for training the FL algorithm of user i is $\boldsymbol{y}_j = [y_{j1}, \ldots, y_{jK_i}]$. We define a vector $\boldsymbol{w}_j$ to capture the parameters related to the local FL model that is trained by $\boldsymbol{x}_j$ and $\boldsymbol{y}_j$. In particular, $\boldsymbol{w}_j$ determines the local FL model of each user j. For example, in a linear regression learning algorithm, $\boldsymbol{x}_{jk}^{\mathrm{T}}\boldsymbol{w}_j$ represents the predicted output and $\boldsymbol{w}_j$ determines the performance of the linear regression learning algorithm. The training process of a FL algorithm is conducted in a way to solve the following problem:

$$\min_{\boldsymbol{w}_1,\ldots,\boldsymbol{w}_M} \frac{1}{M}\sum_{j=1}^{M}\sum_{k=1}^{K_j} f\left(\boldsymbol{w}_j, \boldsymbol{x}_{jk}, y_{jk}\right), \tag{1}$$

$$\text{s. t. } \boldsymbol{w}_1 = \boldsymbol{w}_2 = \ldots = \boldsymbol{w}_M = \boldsymbol{g}, \quad \forall i \in \mathcal{U}, \tag{1a}$$

where $K = \sum\limits_{j=1}^{M} K_j$ is the total number of training samples of all users, $\boldsymbol{g}$ is the global FL model and $f\left(\boldsymbol{w}_j, \boldsymbol{x}_{jk}, y_{jk}\right)$ is the loss function.

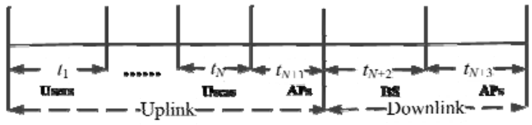

Fig. 2: Time sharing scheme during one transmission period.

To solve (1), the BS will transmit the parameters $\boldsymbol{g}$ of the global FL model to the APs, which relay the global FL model to its served users so that the users can train their local FL models. Then, the users will transmit their trained local FL models to the BS to update the global FL model. The objective of training an FL algorithm is to minimize the loss function in (1). In FL, the update of each user j's local FL model $\boldsymbol{w}_j$ depends on the global model $\boldsymbol{g}$ while the update of the global model $\boldsymbol{g}$ depends on all of the users' local FL models. The update of the local FL model $\boldsymbol{w}_i$ is achieved through a learning algorithm. For example, one can use gradient descent, stochastic gradient descent, or randomized coordinate descent to update the local FL model. After receiving the results from the served users in $\mathcal{J}_i$, the update of the FL model at AP i is given by

$$\boldsymbol{g}_i = \frac{\sum\limits_{j\in\mathcal{J}_i} K_j \boldsymbol{w}_j}{\sum\limits_{j\in\mathcal{J}_i} K_j}. \tag{2}$$

At the BS, the update of the global model $\boldsymbol{g}$ is given by

$$\boldsymbol{g} = \frac{\sum_{i=1}^{N}\sum\limits_{j\in\mathcal{J}_i} K_j \boldsymbol{g}_i}{K} = \frac{\sum\limits_{i=1}^{U} K_i \boldsymbol{w}_i}{K}. \tag{3}$$

B. Transmission Model

As depicted in Fig. 2, the FL training procedure per learning iteration contains $N+3$ steps. Since users are classified into N groups, N steps are required for all users to transmit FL model parameters to their served AP. In the uplink, one step is needed for the APs to transmit model to the BS. In the downlink, there are two steps: 1) the BS broadcasts the global FL model to all APs, and 2) all APs relay the global FL model to all users. As a result, there is a total of $N+3$ steps. In the uplink, the transmission time for users in $\mathcal{J}_i$ is t_i, $\forall i \in \mathcal{N}$. We assume that the decode-and-forward protocol is adopted at each AP. Then, the amount of time t_{N+1} is assigned to all APs to transmit the decoded data from the served users. In the downlink, the BS broadcasts the global FL model parameters to all APs within time t_{N+2}, while all APs simultaneously relay the global FL model parameters to all users within time t_{N+3}. Obviously, to implement the FL algorithm, we have

$$\sum_{i=1}^{N+3} t_i \leq T, \tag{4}$$

where T is the delay requirement.

1) Uplink transmission: During the uplink transmission phase for users in $\mathcal{J}_i$, some of the users in $\mathcal{J}_i$ are selected to simultaneously transmit their local FL models to AP i. Let binary variable a_j denote whether user j is selected to transmit data, i.e., $a_j = 1$ implies that user j is selected to transmit data; otherwise $a_j = 0$. Using the non-orthogonal multiple access (NOMA) technique, the received signal of AP i is

$$y_i = \sum_{j=J_{i-1}+1}^{J_i} h_{ij}\sqrt{p_j}a_j s_j + n_i, \tag{5}$$

where $J_0 = 0$, $J_i = \sum_{l=1}^{i-1} |\mathcal{J}_l|$, $|\cdot|$ is the cardinality of a set, h_{ij} is the channel between user $j \in \mathcal{J}_i$ and AP i, p_j and s_j denote the transmit power and message of user j, respectively, and n_i represents the additive zero-mean Gaussian noise with variance σ^2. Without loss of generality, the channels are sorted as $|h_{i(J_{i-1}+1)}|^2 \geq \cdots \geq |h_{iJ_i}|^2$. Applying the successive interference cancellation in NOMA [5], the achievable data throughput for user $j \in \mathcal{J}_i$ in an uplink transmission period is given by

$$r_{ij}^{\mathrm{U}} = Bt_i \log_2\left(1 + \frac{|h_{ij}|^2 a_j p_j}{\sum_{l=j+1}^{J_i} |h_{il}|^2 a_l p_l + \sigma^2}\right), \tag{6}$$

where B is the bandwidth of the system, and we define $\sum_{l=J_i+1}^{J_i} p_l = 0$ for $j = J_i$.

After all APs successfully decode messages from the served users, all APs simultaneously transmit data to the BS using the NOMA technique. Denote h_i as the channel between AP i and BS. These channels are also sorted in decreasing order, i.e., $|h_1|^2 \geq \cdots \geq |h_N|^2$. As a result, the achievable data throughput of AP i can be written as

$$r_i^{\mathrm{U}} = Bt_{N+1} \log_2\left(1 + \frac{|h_i|^2 q_i}{\sum_{l=i+1}^{N} |h_l|^2 q_l + \sigma^2}\right), \tag{7}$$

where q_i is the transmission power of AP i.

Due to the fact that only partial users are selected to transmit local FL model parameters, the global FL model in (3) can be given by

$$\boldsymbol{g}(\boldsymbol{a}) = \frac{\sum\limits_{j=1}^{M} K_j a_j \boldsymbol{w}_j}{\sum\limits_{j=1}^{M} K_j x_j}, \tag{8}$$

where $\boldsymbol{x} = [a_1, \cdots, a_M]^T$ is the vector of the user selection vector.

2) Downlink transmission: During time t_{N+2}, the BS broadcasts the global FL model to all APs with maximum transmit power Q_0. The downlink transmission rate of AP i is

$$r_i^{\mathrm{D}} = Bt_{N+2} \log_2\left(1 + \frac{|h_i|^2 Q_0}{\sigma^2}\right). \tag{9}$$

During the last transmission time t_{N+3}, all APs broadcast the same global FL model to all users. To maximize the received signal-to-noise ratio, each AP i transmits with maximum power Q_i and the downnlink transmission rate of user $j \in \mathcal{M}_i$ is given by

$$r_{ij}^{\mathrm{D}} = Bt_{N+3} \log_2\left(1 + \frac{\sum_{i=1}^{N} |h_{ij}|^2 Q_i}{\sigma^2}\right). \tag{10}$$

C. Problem Formulation

We formulate an optimization problem whose goal is to minimize the loss function of an FL algorithm. This minimization problem includes optimizing transmit power allocation as well as resource allocation for each user. The minimization problem can be given by

$$\min_{\boldsymbol{p},\boldsymbol{q},\boldsymbol{t},\boldsymbol{a}} \frac{1}{M} \sum_{j=1}^{M} \sum_{k=1}^{K_j} f\left(\boldsymbol{g}(\boldsymbol{a}), \boldsymbol{x}_{jk}, y_{jk}\right) \tag{11}$$

$$\text{s.t. } r_{ij}^{\mathrm{U}} \geq R_j, \quad \forall i \in \mathcal{N}, j \in \mathcal{J}_i, \tag{11a}$$

$$r_i^{\mathrm{U}} \geq R_0, \quad \forall i \in \mathcal{N}, \tag{11b}$$

$$r_{ij}^{\mathrm{D}} \geq R_0, \quad \forall i \in \mathcal{N}, j \in \mathcal{J}_i, \tag{11c}$$

$$r_i^{\mathrm{D}} \geq R_0, \quad \forall i \in \mathcal{N}, \tag{11d}$$

$$\sum_{i=1}^{N+1} t_i \leq T, \tag{11e}$$

$$0 \leq p_j \leq P_j, 0 \leq q_i \leq Q_i, \quad \forall i \in \mathcal{N}, j \in \mathcal{J}_i, \tag{11f}$$

$$0 \leq t_i, \quad \forall i \in \mathcal{N} \cup \{N+1, N+2, N+3\}, \tag{11g}$$

where $\boldsymbol{p} = [p_1, \cdots, p_M]^T$, $\boldsymbol{q} = [q_1, \cdots, q_N]^T$, $\boldsymbol{t} = [t_1, \cdots, t_{N+3}]^T$, R_j is the data size of local FL model $\boldsymbol{w}_j$ that user j has to upload within delay constraint T, R_0 is the data size of the global FL model, and P_j is the maximum transmit power of user j.

III. Algorithm Design

Problem (11) is hard to solve due to the following two difficulties. The first difficulty is the complex objective function and the second difficulty is the non-convexity as shown in constraints (11b)-(11c). To handle the first difficulty of non-convex objective function, we first approximate problem (11) with the following problem by using [6, Theorem 1]:

$$\min_{\boldsymbol{p},\boldsymbol{q},\boldsymbol{t},\boldsymbol{a}} \frac{1}{M} \sum_{j=1}^{M} (1 - a_j) K_j \tag{12}$$

$$\text{s.t. } r_{ij}^{\mathrm{U}} \geq R_j, \quad \forall i \in \mathcal{N}, j \in \mathcal{J}_i, \tag{12a}$$

$$r_i^{\mathrm{U}} \geq R_0, \quad \forall i \in \mathcal{N}, \tag{12b}$$

$$r_{ij}^{\mathrm{D}} \geq R_0, \quad \forall i \in \mathcal{N}, j \in \mathcal{J}_i, \tag{12c}$$

$$r_i^{\mathrm{D}} \geq R_0, \quad \forall i \in \mathcal{N}, \tag{12d}$$

$$\sum_{i=1}^{N+1} t_i \leq T, \tag{12e}$$

$$0 \leq p_j \leq P_j, 0 \leq q_i \leq Q_i, \quad \forall i \in \mathcal{N}, j \in \mathcal{J}_i, \tag{12f}$$

$$0 \leq t_i, \quad \forall i \in \mathcal{N} \cup \{N+1, N+2, N+3\}. \tag{12g}$$

To handle the second difficulty of non-convex constraints (12b)-(12c), we can utilise the difference of two convex functions (DC) programming [7]. In particular, the non-convex rate expressions in (12b) and (12c) can be expressed as a difference of two concave functions. Through the first-order explanation, the difference of two concave functions in (12b) and (12c) can be approximated as a concave function, which indicates that (12b) and (12c) can be approximated as convex constraints. Through DC programming, problem (12) is transformed into a convex function, which can be solved by using the standard convex optimization method, such as dual method.

IV. Applications of the Designed FL Framework for Finger Movement Detection

In this section, we apply the designed FL framework to the classification of two EMG datasets.

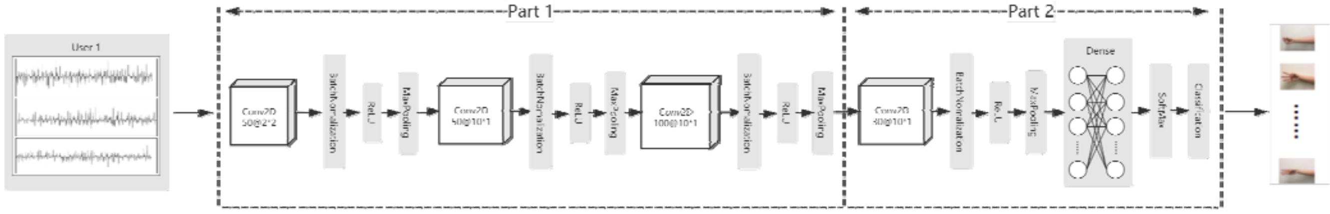

Fig. 3: Local model structure.

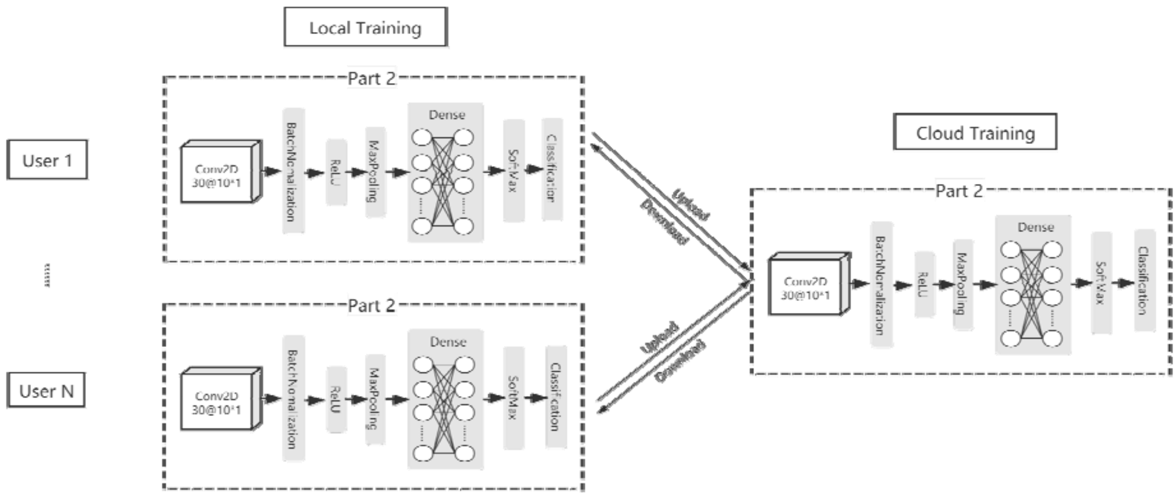

Fig. 4: The proposed federated learning process. Local models were trained using their own data, the parameters in the part two of local models were jointly trained with the cloud.

As shown in Fig. 3 and Fig. 4, we first train local FL models using the personalized data of each user. Then, the parameters of part one in local models (Fig. 3) remain frozen and the parameters of part two is transmitted to the BS for aggregation (Fig. 4).

Next, we will first introduce the data information in Section IV-A. Then we will explain the proposed FL model in Section IV-B. Finally, the experimental results will be presented in Section IV-C.

A. Data Information

The surface EMG signals used in this paper are from the experiments in [8], [9]. Subjects aged between 20 and 35 years old without neurological or muscle disorders were recruited to execute the required order of the finger movements in the experiments. The summary of data information is shown in Table I. The datasets were collected respectively using two and eight EMG channels and processed by the Bagnoli Desktop EMG Systems from Delsys Inc. A 12-bit analog-to-digital converter was applied with a sampling rate of 4000 Hz in data collection [8]. More details of the datasets refer to [8], [9].

These two datasets are considered in this paper as they detected single and combined finger movements with a small amount of electrodes, which leads to less intrusion, lower computation load and more dexterous prosthesis design in practical [10], [11].

TABLE I: Data Information

Dataset	Sampling rate	Sampling time	Electrodes	Sample's size	Subjects	Classes
Dataset 1	4000 Hz	20 seconds	8	80000×8	8	15
Dataset 2	4000 Hz	5 seconds	2	20000×2	10	10

As raw data tends to contain noise caused by muscle fatigue or device shifts, data normalization was employed to rescale the data. In implementation, the input signals of each subject were rescaled to the range [0, 1] as follows:

$$y_N = \frac{y - y_{\min}}{y_{\max} - y_{\min}}, \tag{13}$$

where y is the signal to be normalized, y_N is the normalized signal, $y_{\min}$ and $y_{\max}$ denote the minimum and maximum elements in y, respectively.

B. Proposed Model

The proposed model is shown in Fig. 3. In the data processing stage, raw EMG signals of each movement is of size 20000×2 or 80000×8, we found that feeding the classifier raw EMG signals would lead to lower classification performance. To alleviate the difficulty of classification, we process the inputs through the windowing approach [12], [13] which uses a sliding window (size: 500×2) on the EMG signals with an increment (10 for dataset 1, 50 for dataset 2) to generate overlapping samples from the original samples. By doing so, more samples with shorter length are obtained and each newly generated sample contains part of information of the original samples.

In the classification stage, a CNN having 20 layers is adopted for the classification of finger movements. As shown in Fig. 3, four convolutional blocks – consisting of a convolutional, a batch normalization layer, a ReLU layer, and a MaxPooling layer [14] – are established which are followed by two fully-connected layers and a softmax layer. In the initial training stage of each local model, Adam optimizer [15] is used as the optimization algorithm, cross-entropy is the loss function and epoch is set to 5.

As shown in Fig. 4, after the initial training of each local model, the parameters of each model in high-level layers could be transmitted to the cloud model and interacted with the cloud model. Two scenarios are then considered: convex loss function and non-convex problems in terms of the loss function. As shown in Fig. 5, if the parameters before dense layers are frozen and we only train the dense layers in the FL framework, the loss function is convex since each device only needs to update dense layers which have a linear activation function. However, if more convolutional layers with non-linear activation functions are involved in the FL training, the convex property does not hold any more.

In order to observe the classification performance when the convex condition is not satisfied, we also conducted a comparative experiment. As shown in Fig. 5, in the convex scenario, we only trained the dense layers in the FL framework. The second case was to incorporate non-linear activation functions and convolutional layers in the shared model, where the objective function is non-convex.

To test the effectiveness of the proposed method, two datasets introduced in Section IV-A were employed for the classification of finger movements. As the number of APs (N) and users (K) varied in different tasks, we assume that there are two cases: $N < K$ and $N = K$. The first case happens when the data of several users are collected in an AP, for instance, each AP is a hospital which collects data from several patients. Dataset 1 is used for this purpose. We assume that $N = 3$ and $K = 8$ in this case.

The second situation is that each user has an AP which is able to collect and transmit data to the server. A real example is that each subject has a sensor for data collection and a device for model training and data transmission. Dataset 2 is employed in the case where we assume that $N = K = 10$. In the following sections, we will discuss the influence of involving a different number of users in the FL framework. The experiments in this paper were conducted using NVIDIA P100 GPUs. The results of the above experiments are as follows.

C. Results

1) $N < K$: In this case, we assume that 3 APs are used in the data collection and transmission for 8 users. We also use a conventional deep learning (DL) method for comparison. The conventional DL method gathers data from all users in advance and train the model off-line. The test data and hyper parameters of the proposed model and the comparative method are the same in all experiments.

The test results are shown in Table II, from which we can see that the proposed FL method improves the classification performance of EMG signals compared to the conventional DL method. It is demonstrated in Table II that the proposed method not only achieves higher overall accuracy, but also has better performance in each AP.

TABLE II: Test accuracy of the proposed FL method and the conventional deep learning (DL) method in the case $N < K$. The number of users of AP 1, AP 2 and AP 3 are 3, 3, 2, respectively.

Method	Overall	AP 1	AP 2	AP 3
Proposed FL method	**0.983**	**0.978**	0.978	**1**
Conventional DL method	0.967	0.956	0.978	0.967

2) $N = K$: In this case, we assume that the number of APs and users is the same. Dataset 2 is used in this case. As there are 10 users in this dataset, we have $N = K$ = 10. The test accuracy of each local model as well as the comparative results with the conventional DL method are given in Table III. From Tables II and III, it can be concluded that the proposed method has superiority in improving classification performance due to its personalized training process.

Also, three cases are considered in this scenario. That is, we randomly selected 10 users, 8 users, and 6 users participating in the proposed FL. The results in Fig. 6 demonstrate that when 10 users participated in the learning process, the overall test accuracy was 98.5% at the 50-th iteration. When 8 users joined the FL, the test accuracy was 81%. Meanwhile, if 6 users took part in the training, the final test accuracy was 58%. It can be seen that as the number of active users decreased, the overall accuracy decreased. The statistical properties of the proposed method can be found in Table IV, from where we can see that the classes 2, 4, 6, 8, and 10 have good statistical properties while other classes were less capable of being detected using the proposed method.

3) Non-convex Scenario: As mentioned above, a non-convex case was also considered in this paper because it is important and common in practice. The corresponding results are shown in Fig. 6. In the non-convex scenario, the global model of 10 users reached 96% overall test accuracy, while the models of 8 and 6 users demonstrated test accuracies of

TABLE IV: Statistical properties of the proposed method on the 10-class finger movement detection.

Properties \ Classes	1	2	3	4	5	6	7	8	9	10
Sensitivity	1	1	0.90	1	0.90	1	1	1	0.90	1
Specificity	0.99	1	1	1	0.99	1	0.99	1	0.99	1
Precision	0.95	1	1	1	0.90	1	0.91	1	0.95	1
F1 Score	0.98	1	0.95	1	0.90	1	0.95	1	0.92	1
MCC	0.97	1	0.94	1	0.89	1	0.95	1	0.92	1

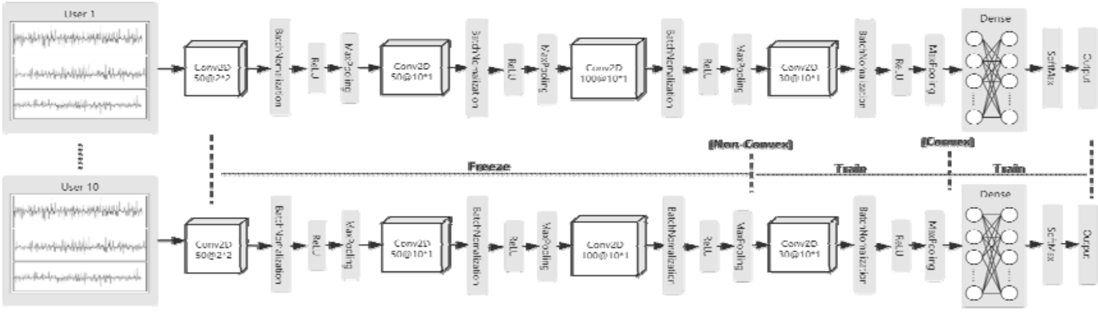

Fig. 5: The convex and non-convex scenarios.

TABLE III: Test accuracy of the proposed FL method and the conventional deep learning method when $N = K$.

Method	Overall	User 1	User 2	User 3	User 4	User 5	User 6	User 7	User 8	User 9	User 10
Proposed FL method	**0.985**	1	1	0.9	1	**1**	0.95	1	1	**1**	1
Conventional DL method	0.980	1	1	0.9	1	0.95	1	1	1	0.95	1

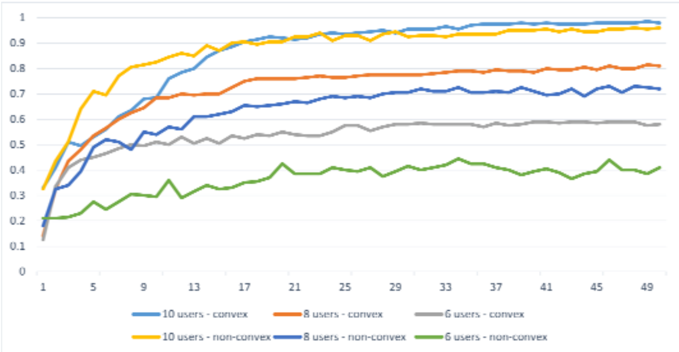

Fig. 6: The test accuracy of the global model as the number of users varies (10 users, 8 users and 6 users) in two scenarios (convex and non-convex). The horizontal axis represents the iterations and the vertical axis shows the test accuracy.

72% and 41%, respectively. The classification accuracy of FL in the non-convex scenario is lower compared to the case in which the convex condition is satisfied.

V. Conclusion

In this paper, we have proposed a communication efficient FL framework for single and combined finger movement detection. We have formulated a joint learning and resource allocation problem to reduce the FL transmission delay via optimizing transmit power of each device, time allocation, and user selection. To solve the formulated problem, we have converted it to a tractable expression. Then, we have used a successive convex optimization method to solve the simplified optimization problem. To verify the performance of the proposed FL framework, two EMG datasets with respectively 10 and 15 finger movements were employed in our experiments. Simulation results demonstrate the effectiveness of the proposed framework.

References

[1] M. V. Arteaga, J. C. Castiblanco, I. F. Mondragon, J. D. Colorado, and C. Alvarado-Rojas, "EMG-driven hand model based on the classification of individual finger movements," *Biomedical Signal Processing and Control*, vol. 58, p. 101834, 2020.

[2] M. Zanghieri, S. Benatti, F. Conti, A. Burrello, and L. Benini, "Temporal variability analysis in sEMG hand grasp recognition using temporal convolutional networks," in *Proc. IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, Genova, Italy, Aug 2020, pp. 228–232.

[3] B. Paaßen, A. Schulz, J. Hahne, and B. Hammer, "Expectation maximization transfer learning and its application for bionic hand prostheses," *Neurocomputing*, vol. 298, pp. 122–133, 2018.

[4] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao, "Fedhealth: A federated transfer learning framework for wearable healthcare," *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 83–93, 2020.

[5] X. Chen, A. Benjebbour, A. Li, and A. Harada, "Multi-user proportional fair scheduling for uplink non-orthogonal multiple access (NOMA)," in *Proc. IEEE Vehicular Technology Conference (VTC Spring)*, Seoul, South Korea, May 2014, pp. 1–5.

[6] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 1, pp. 269–283, 2021.

[7] H. H. Kha, H. D. Tuan, and H. H. Nguyen, "Fast global optimal power allocation in wireless networks by local D.C. programming," *IEEE Trans. Wireless Commun.*, vol. 11, no. 2, pp. 510–515, 2012.

[8] R. N. Khushaba, S. Kodagoda, M. Takruri, and G. Dissanayake, "Toward improved control of prosthetic fingers using surface electromyogram (EMG) signals," *Expert Systems with Applications*, vol. 39, no. 12, pp. 10 731–10 738, 2012.

[9] R. N. Khushaba and S. Kodagoda, "Electromyogram (EMG) feature reduction using mutual components analysis for multifunction prosthetic fingers control," in *Proc. International Conference on Control Automation Robotics & Vision (ICARCV)*, Guangzhou, China, Dec 2012, pp. 1534–1539.

[10] J. J. V. Mayor, R. M. Costa, A. Frizera Neto, and T. F. Bastos, "Dexterous hand gestures recognition based on low-density semg signals for upper-limb forearm amputees," *Research on Biomedical Engineering*, vol. 33, no. 3, pp. 202–217, 2017.

[11] A. H. Al-Timemy, G. Bugmann, J. Escudero, and N. Outram, "Classification of finger movements for the dexterous hand prosthesis control with surface electromyography," *IEEE Journal of Biomedical and Health Informatics*, vol. 17, no. 3, pp. 608–618, 2013.

[12] G. Jia, H. K. Lam, J. Liao, and R. Wang, "Classification of electromyographic hand gesture signals using machine learning techniques," *Neurocomputing*, vol. 401, pp. 236–248, 2020.

[13] G. Jia, H. K. Lam, S. Ma, Z. Yang, Y. Xu, and B. Xiao, "Classification of electromyographic hand gesture signals using modified fuzzy c-means clustering and two-step machine learning approach," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 28, no. 6, pp. 1428–1435, 2020.

[14] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep Learning*. MIT Press Cambridge, 2016.

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