Throughput Prediction by Radio Environment Correlation Recognition Using Crowd Sensing and Federated Learning

Satoshi Nakaniida, Takeo Fujii
Advanced Wireless Communication Research Center (AWCC)
The University of Electro-Communications (UEC)
1–5–1, Chofugaoka, Chofu, Tokyo, 182–8585 Japan
E-mail: {nakaniida,fujii}@awcc.uec.ac.jp

***Abstract*— We propose an approach using federated learning for predicting Wi-Fi and LTE transmission control protocol (TCP) throughput to reduce the delay between the output of prediction results and the problem of security risks by sharing the datasets, which is a problem with conventional machine learning methods. The proposed method collects measurement datasets such as received signal strength index from distributed edge devices. Then, a shared learning model is created using the measured dataset on the server. The created model is retrieved by edge devices at any time and used to predict TCP throughput. To evaluate the effectiveness of the proposed method, we perform the emulation evaluation using measured datasets obtained in a real environment. The emulation results reveal that the proposed method can skillfully predict the TCP throughput in the realistic communications. Additionally, the prediction accuracy of the TCP throughput can be improved by creating a learning model for each network area.**

Keywords— Federated Learning, Deep-Neural-Network, TCP Throughput, Crowd Sensing, Android

I. INTRODUCTION

In recent years, methods that directly implement machine learning models on edge devices such as smartphones to perform everything from training to prediction have been attracting attention [1]-[5]. In conventional methods, the learning models are often placed in cloud services due to the advantages of scalable storage of large amounts of training data and the availability of high-performance processing functions [6][7]. However, if we try to complete the entire process from training to prediction in the cloud, it will take a long time for the edge devices to get the prediction results due to the delay caused by communication time. In addition, the exchange of data used for learning itself requires a large amount of communication, which poses a cost issue [8]. On the other hand, if learning and prediction are completed only by the edge device, prediction results can be obtained quickly, but the data used for learning is limited to the device itself. Federated Learning [1] is a method that leverages both the advantages of the cloud, where multiple devices can share their learning status, and the advantages of prediction on edge devices. In this paper, we focus on an approach to predict transmission control protocol (TCP) throughput using federated learning. In federated learning, learning and prediction is done by edge devices, while the learning status of the learning model is collected and shared on a server (we do not mention federated learning where edge devices share the learning model directly with each other). A device with a small amount of observation data can obtain highly accurate prediction results by downloading the weight parameters of the learning model learned by other devices. A feature of this system is that it does not share the training data, but only the weight parameters of the training model. This solves the problems of security, communication volume, and implementation cost that conventional machine learning methods have faced [9].

II. PROPOSED METHOD

A. Federated Learning

In the federated learning used in this study, the weight parameters of each device are shared with other devices by uploading the data to one dedicated servers for sharing. All the machine learning models used in this study are Deep-Neural-Network (DNN), and the network used for prediction is Wi-Fi. TABLE I shows the list of features used for training. The movement speed shown here is calculated from the temporal variation of location information, and is obtained using a library provided in advance for Android.

B. System Model

To build training models, we use Keras (in TensorFlow), a high-level neural network development library provided by Google [10]. We use TensorFlow because it can be easily converted into learning models optimized for mobile environments such as smartphones using existing APIs [11]. The hyper-parameters of the learning model are set to the values shown in TABLE II. To determine the hyper-parameters, we used a dataset obtained with the same equipment and methods as those used to create the dataset in "III. THROUGHPUT PREDICTION" described below. The dataset used for parameter tuning is not used for any other purpose than tuning. Since it is not practical to adjust the learning models of all the devices one by one, the values in TABLE II are fixed. When recording the learning status of the learning model to be used in this study, the file size for recording one machine learning model is always about 10kByte, regardless of the measurement time. This is smaller than the data size of about 2000kByte for 24 hours of measurement of the values in TABLE I at one per second. Keeping the file size small is an important consideration because it leads to a reduction in operational costs and communication time.

C. Implementation of the Federated Learning System

We implemented a federated learning system on a computer that operates using the measured data. In the following, the device that manages the learning model (shared model) shared by multiple terminals is called the "global node," and the terminal that collects data and performs learning and prediction, corresponding to an edge device, is called the "local node."

TABLE I. INPUT/OUTPUT PARAMETERS TO THE MACHINE LEARNING MODEL

Feature	*Description*	*Type*
Network ID	*Wi-Fi router identification number*	*Input*
RSSI	*Received Signal Strength Indicator*	
Latitude	*Location info. measured by smartphone*	
Longitude	*Location info. measured by smartphone*	
Week	*Parameters assigned to days of the week from 0 to 6.*	
Hour	*Hour is expressed as a number from 0 to 23.*	
Speed	*Smartphone movement speed*	
Throughput	*Downlink TCP Throughput*	*Output*

TABLE II. CONFIGURATION PARAMETERS OF THE MACHINE LEARNING MODEL

Variable	*Parameter*
Epoch number	500 *(Introduce Early-Stopping with patience=10)*
Neurons count of Hidden layer	32
Number of hidden layers	1
Initial learning rate	0.001
Normalization rate	0.01 *(L2 Normalization)*
Dropout rate	*(Not used)*
Data ratio for cross-validation	20%
Loss function	*Mean Square Error (MSE)*
Activation function	*ReLU*

Fig.1 shows the system model of the implemented federated learning. In this study, we do not consider the communication time and communication errors between nodes, and local nodes do not communicate directly with each other. In addition, we did not use any public libraries to implement the federated learning itself, but implemented it ourselves.

D. Updating of Shared Model

The basic flow of federated learning is to iterate the process of updating the shared model until the learning converges. First, the global node sends a request to the local node to send the learning results to the global node. Next, the local node that responds to the request learns with its own data and shares only the weight parameters that are the results of learning. Finally, the shared weight parameters are processed based on the Federated Averaging algorithm to update the weight parameters of the shared model. This algorithm is expressed in Equation (1) [1].

$$\mathbf{W} = \sum_{k=1}^{K} \frac{n_k}{n} \mathbf{W_k} \quad (1)$$

Where $\mathbf{W}$ is the weight matrix of the shared model, K is the number of local nodes, n_k is the number of records of training data used by local node k, n is the number of records of training data used by all local nodes, and $\mathbf{W}_k$ is the weight matrix of the training model at local node k.

TABLE III. THE EXPERIMENT SPECIFICATIONS

Number	*Type*	*Overview*
(1)	Device	Android11, Sharp Aquos R3
(2)	Device	Android6, Huawei MediaPad
(3)	Device	Android11, GalaxyS10
(4)	Router	Buffalo, WSR-1166DHP6 (Router A)
		NEC, AtermWR8370N (Router B)
(5)	PC	Lenovo, ThinkCenterM715qTiny Router A is connected with a wired LAN cable.
		Lenovo, ThinkCenterM715qTiny Router B is connected with a wired LAN cable.

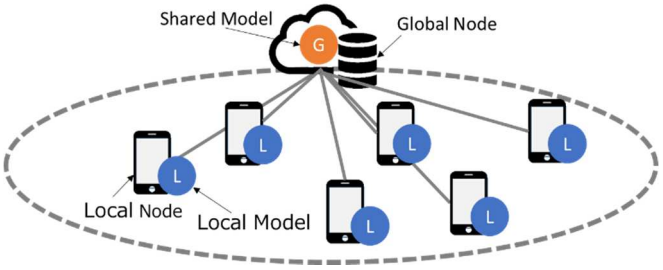

Fig. 1. System model (The global node is assumed to be operated by a serverless cloud service, and the local node is considered to be a smartphone with the Android OS. In this verification, the nodes are pseudo-split on the computer.)

Each local node reacquires the shared model when it is updated. At this stage, it is not yet possible to derive an appropriate value for the frequency with which the global node updates the shared model. In the literature [1], it is said that it is desirable to update the model at noon when many terminals are gathered and at night when terminals are not operated.

III. THROUGHPUT PREDICTION

A. Dataset Preparation

Using the actual measured data, we evaluate the prediction accuracy of the throughput.

To obtain measured datasets, we used our own Android application and the open-source-software "iPerf3". iPerf3 is a popular software for measuring the maximum throughput. The reason why we chose to use a home-grown application is that there is no application that can measure the values that we put into the input features of the training model. However, by creating a home-made application, we can be flexible when we want to change the features we want to measure. In addition, application development will be necessary when we try to train and predict machine learning models on Android in the future, so we decided to create our own application. The data measured by the two software are merged in a time series and treated as a single dataset.

Table III shows the experiment specifications. To create the free Wi-Fi environment in the city, the two routers were placed on the second floor of a building. The building exists in a suburban area of Tokyo. We obtained measured dataset while walking around the communication area. In the measurement, the smartphones were held in our hands.

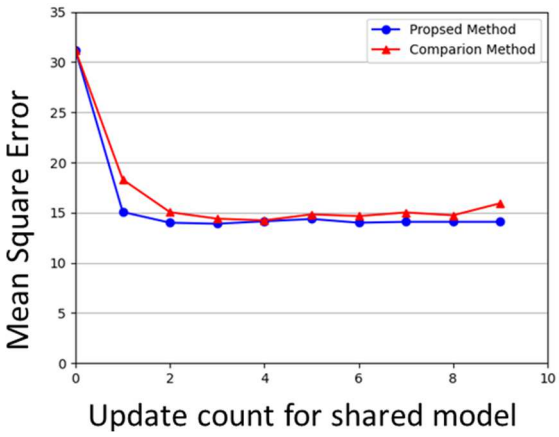

Fig. 2. Predicted mean square error of the proposed method using federated learning and comparison method (comparison method does not use federated learning)

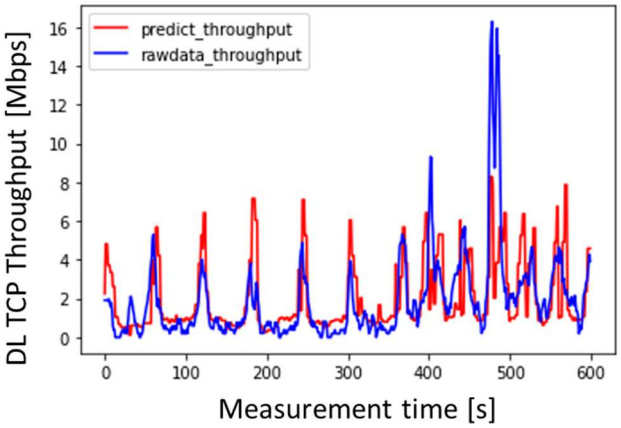

Fig. 3. Comparison of measured and predicted maximum TCP throughput values for DownLink (DL) (the prediction was made using a shared model that was updated seven times with minimal error)

The results presented here have been validated on three datasets. All datasets were created with three smartphones, and one dataset contains only the data measured by one smartphone.

The three smartphones started and ended their measurements almost simultaneously. The datasets used were also pre-processed. Among the features used for training, "week" and "time" were calculated from time stamps. The instantaneous values of throughput were then averaged every 5[s] in order to build an accurate learning model. The data was shuffled in uniform distribution so that the time series would be randomized. This preprocessing is used in many machine learning methods to construct an accurate learning model.

B. Details

In the performance evaluation, the two local nodes and one global node are used. In the evaluation, the two datasets are used as the training data in the two local nodes. Another dataset is utilized as the test data. The validation results are recorded with different combinations of datasets, and the average prediction error is calculated. The assumption is that the combination pattern will not be changed until the shared model converges. To speed up the training of the shared model, only one dataset is assigned to each local node, and the same data is always used for training.

This paper uses DNN as a comparison method. This method inputs all the data into one DNN training model for one update. The structure and hyper-parameters of the learning model of the comparison method are the same as those of the proposed method.

C. Results

Fig. 2 shows the error (Mean Squared Error, MSE) between the prediction results of the federated learning and comparison methods and the measured throughput values when the shared model is updated a total of nine times. The results show that federated learning, which updates only the weights, provides the same level of prediction accuracy as the comparison method. Here, the convergence of learning using coalitional learning is a little faster. However, we do not compare the convergence speed this time because the assumption that the same data is always used for training the local model was introduced to make the learning converge faster. Fig. 3 shows the prediction results of the throughput in federated learning when the prediction error is the smallest for a single data pattern. A shared model that has been updated seven times is used to output the predictions shown in Fig. 3. This result shows that it can be seen that once the learning converges, the throughput can be predicted approximately even if parameters other than throughput are given as input. However, there is room for improvement in the prediction accuracy.

D. Discussion

From the results of the verification, it can be seen that the prediction of the points where the throughput increases or decreases drastically is almost complete, but it does not fully follow the scale. For example, we switched the router to which we connected 360 seconds after the start of the measurement, and although the average throughput tends to be lower for the router connected first than for the router connected later, the prediction result shows that the average throughput is constant regardless of the access point connected. This result shows a trend that is not good for prediction. The learning model has not learned that the throughput value may change largely even if the input features change little. However, this unfavorable trend was expected before the verification. Therefore, we included location information and base station IDs (Network IDs) as input parameters so that the learning model could determine the differences in the characteristics of each access point. However, as the result shows, the expected results were not obtained.

In the next chapter, we propose a solution to this problem by narrowing down the target of the learning model.

IV. Additional Verification of Throughput Prediction

This paper has only considered one shared model. However, the prediction error remains in the proposed method as described in Sect. III-C. Thus, we attempt to improve the prediction accuracy of throughput by creating a learning model for each area.

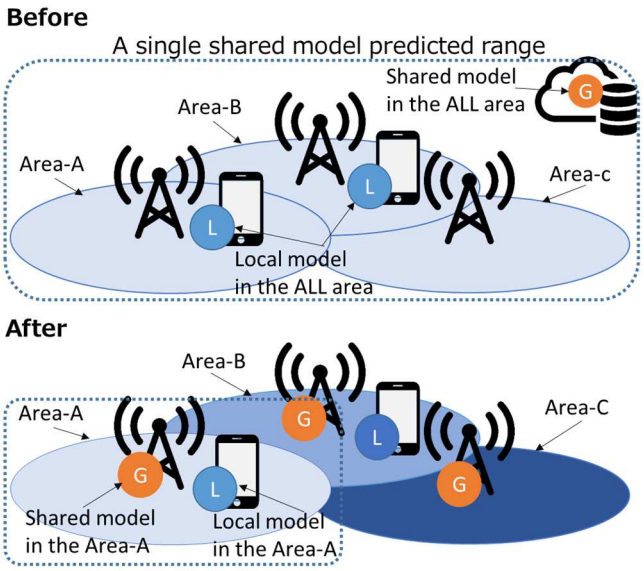

Fig. 4. Before/after comparison when changing the shared model's prediction target area

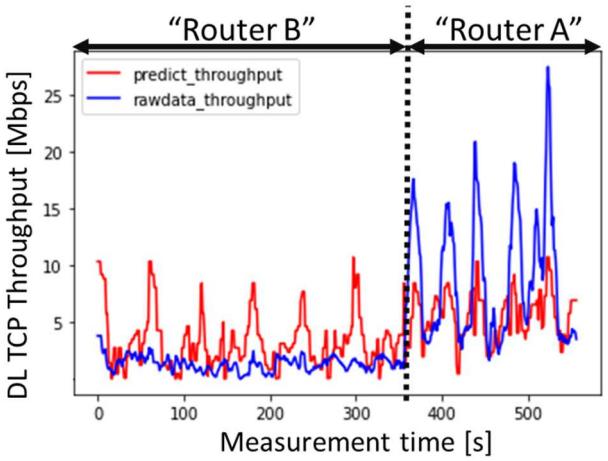

Fig. 5. Prediction results for measured values with large scale change in throughput using the shared model created before narrowing the area for prediction.

A. *Details of Additional Validation*

Fig. 4 shows an improvement of the proposed method. In the above figure that is represented as 'Before', a same shared model is used in all areas. Meanwhile, in the below figure, different models are utilized among areas. The local node switches the machine learning model for each access point. Using different models according to the area, the prediction accuracy of throughput can be improved.

This section verifies how much the prediction accuracy improves when the target of the learning model is narrowed down. In the additional validation, we use the same dataset as in Sect. III so that the prediction accuracy can be easily compared. For the sake of explanation, we name the routers used as connection points as Router A and Router B, respectively. In order to prepare a local node learning model and a shared model for each area, the datasets used for prediction must be only those for the same area. For example, if we want to predict the throughput of Router A, we should use only the dataset for Router A. As it is, the dataset used in Sect. III is recorded in a single file regard-

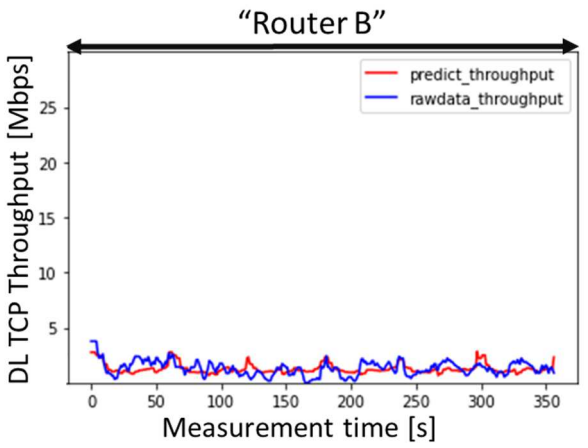

Fig. 6. Throughput prediction results in the same area by a learning model that predicts the communication-capable area of Router B.

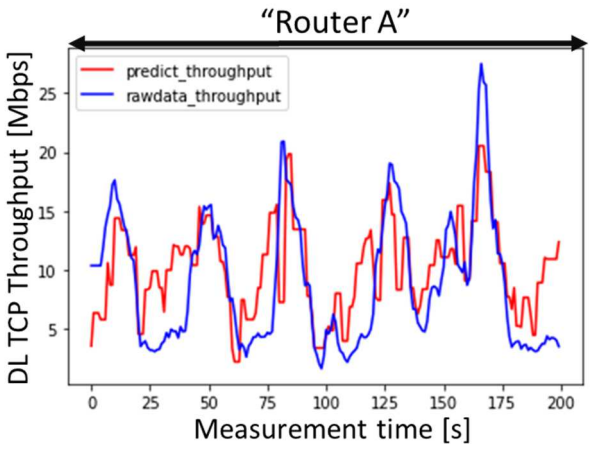

Fig. 7. Throughput prediction results in the same area by a learning model that predicts the communication-capable area of Router B.

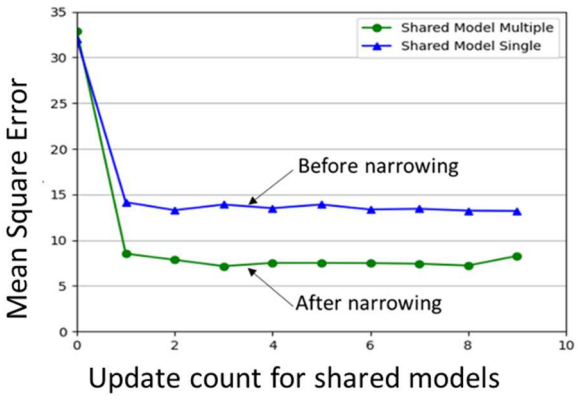

Fig. 8. Before/After narrowing the area for prediction

less of the number of access points to be connected. Therefore, the dataset was divided into separate files for each connection point before this verification. In other words, each of the three datasets was split into two files, one for Router A and one for Router B, and then the prediction accuracy was evaluated for each area. In other words, each of the three datasets will be split into one for Router A and one for Router B.

B. Results of Additional Validation

Fig. 5 shows the results of predicting the average throughput of each access point with the shared model before narrowing down the prediction target. Fig. 6 and Fig. 7 show the results of prediction using the same measured values as in Fig. 5, but with different learning models for each router.

When comparing Fig. 8 shows the number of updates of the shared model and the error between the throughput prediction results of the shared model and the actual measured values. The errors in this Fig. are averaged over three patterns, each with different test data. From the results, we can expect a much better prediction accuracy.

C. Discussion of Additional Validation

We have confirmed that narrowing down the target of prediction has certain advantages in terms of improving prediction accuracy, but we will discuss the possible disadvantages. Among several possible problems, one that we should pay particular attention to is that it takes a long time for the learning of a particular shared model to converge. The assumption is that each shared model can only collect local models from users in its own area. By subdividing the prediction target into smaller and smaller areas, it will be easier for users not to gather in the target area, or for the observation time in the target area to be shorter than before. If a local model that has not been sufficiently trained is shared, and a shared model with low prediction accuracy is created, it may adversely affect the convergence speed of training and prediction accuracy of all terminals that receive the shared model.

V. Future Work

A. Shared Model Transfer

We believe that transfer of training models can be effective in dealing with the problem of local nodes not congregating in a particular area, and we show a simple example of transferring a training model in Fig. 9. We will not go into the details of the validation results here, but by comparing the results of training a trained shared model with an untrained shared model, and training the model with a dataset under the same conditions, we found that the accuracy and convergence time were better when the trained shared model was transferred. The verification we have done to date is limited to only those cases where the surrounding environment of the source and destination are similar. Therefore, depending on the surrounding environment and communication method, we may not always obtain good results. As a future work, we would like to conduct additional discussions along with the increase of the datasets.

B. Learning on Smartphones

Understanding the workload of smartphones is also an important issue. This is because smartphones have a limited drive time and can only process a limited number of tasks at a time. Even if the proposed method can reduce the security risk, it is difficult to share the weight parameters as a local node if the load on the smartphone is too high. Therefore, we are considering the use of TensorFlow Lite [11], which is expected to reduce the processing load of

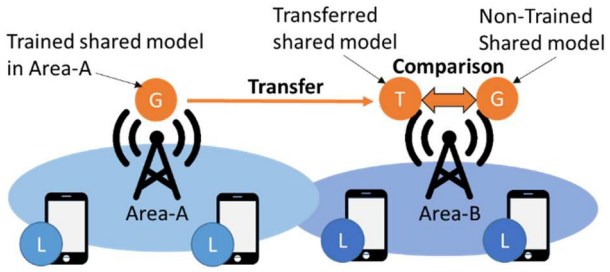

Fig. 9. A simple example of transferring a learned shared model to another area (comparing the method of learning from the transferred model with the method of learning from the unlearned model, and examining the difference in learning speed and prediction accuracy)

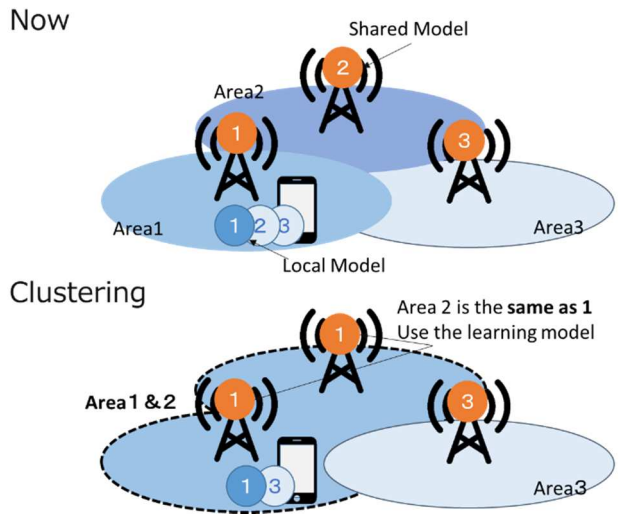

Fig. 10. Image of a clustering method that supports multiple areas with a single shared model.)

smartphones. This is expected to reduce the size of the model, thereby reducing not only the processing load but also the load on the device's storage. We plan to develop TensorFlow Lite models while confirming their performance in questionnaires to the extent that they do not impose a burden on users.

C. Clustering of Shared Models

It is a good approach to reduce the total number of shared models, because if we prepare training models for each area, there will be less information sharing from local nodes, and it may not be possible to perform sufficient training. The number of training models is a tradeoff between prediction accuracy and learning difficulty. If there are fewer training models, the number of local nodes that provide data to the shared model increases, making learning easier, but the prediction accuracy becomes worse because the area to be predicted is too large. On the other hand, if the number of training models is many, prediction accuracy can be expected to improve because only local areas need to be predicted, but the number of local nodes providing data to one shared model will decrease, and in the worst case, no one will provide data. These tradeoffs can be seen from the additional validation described earlier. Therefore, if the prediction accuracy can be maintained to some extent, we should cluster the areas. Specifically, if the performance of the surrounding environment and router devices are similar, and if there is no significant difference in prediction accuracy, the approach is to integrate them.

Fig. 10 shows an image of a shared model that is clustered. In the upper part of the figure, a training model is needed for each area before clustering, but if area 1 and area 2 are merged as shown in the lower part of the figure, the area that one training model is responsible for can be expanded. The clustering method will be an issue to be discussed carefully.

VI. Conclusion

We proposed an approach using coalition learning to solve the problems of security risk, communication volume, and operational cost in throughput prediction using conventional machine learning. Using real data measured by smartphones, coalition learning was conducted, and it was confirmed that the prediction accuracy was equivalent to that of conventional methods. In addition, the case where the prediction accuracy is improved by narrowing down the prediction target was also examined, and its merits and demerits were discussed.

This study was supported by JSPS Grants-in-Aid for Scientific Research JP18H01439 and 18KK0109.

Refarences

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson and B. Aguera, “Communication-efficient learning of deep networks from decentralized data,” *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 1273-1282, 2017.

[2] T. Nishio and R. Yonetani, “Client selection for federated learning with heterogeneous resources in mobile edge,” *Proc, IEEE ICC*, Shanghai, Cnina, May 2019.

[3] Q. Yang, Y. Liu, T. Chen, and Y. Tong, “Federated machine learning: Concept and applications,” *CoRR*, abs/1902.04885, 2019.

[4] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, “Analyzing federated learning through an adversarial lens,” *In Proceedings of the 36th International Conference on Machine Learning*, pp. 634–643, 2019.

[5] T. Yang, G. Andrew, H. Eichner, H. Sun, W. Li, N. Kong, D. Ramage, and F. ¸Beaufays, “Applied federated learning: Improving Google keyboard query suggestions,” *arXiv preprint 1812.02903*, 2018.

[6] D.Phanekham,S.Nair, N.Rao and M.Truty, "Predicting throughput of cloud network infra-structure Using Neural Networks," *IEEE INFOCOM 2021-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 1-6, 2021.

[7] H. Jha and V. Vijayarajan,"Mobile internet throughput prediction using machine learning techniques," *2020 International Conference on Smart Electronics and Communication (ICOSEC),* 2020, pp. 253-257, 2020.

[8] A. Panghal, K. Govindan and K. Subramaniam, "Transmission time estimator for social and cloud applications in smartphones," *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1-6, 2017.

[9] S. K. Das and S. Bebortta, "Heralding the future of federated learning framework: architecture, tools and future directions," *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence),* pp. 698-703, 2021.

[10] TensorFlow, URL https://www.tensorflow.org/

[11] TensorFlow Lite Converter, URL https://www.tensorflow.org/lite /convert/

[12] Android Studio, URL https://developer.android.com/