Vanilla Split Learning for Transportation Mode Detection using Diverse Smartphone Sensors

Umer Majeed, Sheikh Salman Hassan, Choong Seon Hong
Department of Computer Science and Engineering,
Kyung Hee University, Yongin, Korea
Email: {umermajeed, salman0335, cshong}@khu.ac.kr

Abstract

Transportation mode detection (TMD) by analyzing smartphones embedded sensors' data is an emerging application for mobility-awareness services at the government or individual level. Split Learning is a relatively new privacy-preserving distributed machine learning technique. In vanilla split learning, a neural network is vertically distributed between the client and the server. In this paper, we performed vanilla split learning for Transportation mode detection on smartphones sensors' based dataset. We showed that the Split neural network (SplitNN) has the same performance as the baseline typical deep neural network. However, split learning requires fewer computation resources from the client and also preserves the privacy of raw data.

I. Introduction

Identifying a user's mode of transportation based on observations of the users' environment is a trending topic of research with applications in the Internet of things, big data [1], wireless networks [2] and artificial intelligence. Such research can help in intelligent & environmentally sustainable transport management systems, and smart government monitoring [3] services in smart cities [4].

Transportation mode detection (TMD) can provide background information that can be used to provide suitable services based on the needs of the customer. TMD is an application of intelligent context-awareness, where wearable devices can be used to detect the traveling mode of users. TMD is currently approached using features such as speed, acceleration, and direction; either on their own or in combination with geographic information system (GIS) data.

Federated learning (FL) [5], [6], [7], [8], [9] is a method to train a global model at a server such that the privacy of raw data of users remains preserved. However, the local models sent by users' devices are still subject to inference attacks. Split Learning is a relatively new distributed learning technique in which a neural network (NN) is usually split into two parts. The first part is trained on the client's end, and the second part is trained on the server's end. Initially, in vanilla split learning, the client sends the label to the server. For each epoch of the training phase, the output of the client's NN portion is sent to the server. This intermediate output is fed to the server's portion of NN. The back-propagation is also vertically distributed between client and server by sharing gradients of corresponding specific layers. Split learning requires less computation from the client as well as is more robust to inference attacks than FL.

The contribution of this study is summarized as follows:

- We employed split learning for transport mode detection. For this, we set up a split-NN (splitNN) between a client and a server. Firstly, the labels of the data-set are sent to the server by the client. Afterward, the split learning is performed over a peer-to-peer (P2P) connection.
- The performance of the splitNN is found the same as of baseline typical deep learning NN.

The rest of the paper is organized as follows: Section II illustrates the system model for vanilla split learning for TMD. Section III formulates the split learning problem for TMD. Section IV briefly describes the dataset used. Section V gives the simulation results, and Section VI concludes our work.

II. System Model

The system model for split learning comprises a client C and a server S communicating through a P2P connection. A splitNN W^{SP} is created which has two parts. The first and second part of W^{SP} are denoted by subnet W^{C} and subnet W^{S} respectively. W^{C} is assigned to the client and W^{S} is assigned to the server.

The client owns the dataset D_C $(X_C ; Y_C)$, where X_C is feature space, and Y_C is label space. Client dataset D_C

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. 2020R1A4A1018607). *Dr. CS Hong is the corresponding author.

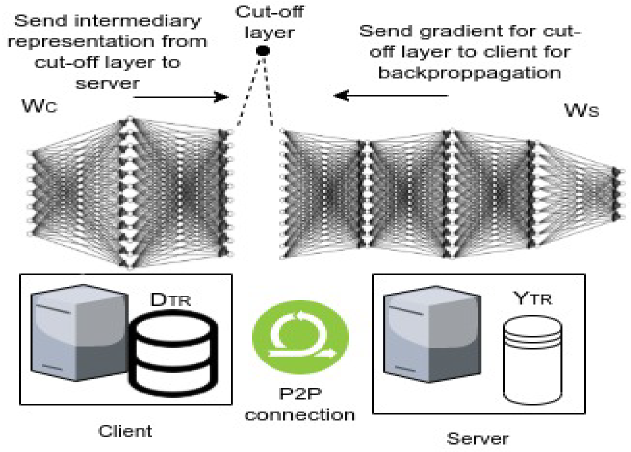
Send intermediary representation from cut-off layer to server
Cut-off layer
Send gradient for cut-off layer to client for backproppagation
Wc
Ws
DTR
YTR
P2P connection
Client
Server

Fig. 1. System model - Split Learning - Training phase

is divided into training dataset D_{TR} $(X_{TR}; Y_{TR})$ and test dataset D_{TE} $(X_{TE}; Y_{TE})$. The labels in label space Y_{TR} and Y_{TE} are sent to the server for each batch during the training phase of split learning as shown in Fig. 1.

III. Problem Formulation

The problem is formulated in algorithmic form in Algorithm 1 and Algorithm 2 [10].

```
Algorithm 1 : Split Learning - Client
initialize $W_t^C$
for t   1; :::; $Epoch_{max}$ do
    for b   1; :::; $Tbatch_{max}$ do
        $f_{W^C}(X_{TR;b}; W_t^C)$   $A_{t;b}$   . forward-pass
        ServerSplitNN($A_{t;b}$, $Y_b$)   . in Algorithm 2
        procedure CLIENTBACKPROP($dA_{t;b}$)
            Model updates $W_{t \ 1}^C \leftarrow W_t^C - \eta \, dA_{t;b}$
        end procedure
    end for
end for
```

```
Algorithm 2 : Split Learning - Server
initialize $W_t^S$
procedure SERVERSPLITNN($A_{t;b}$, $Y_b$)
    $f_{W^S}(A_{t;b}; W_t^S)$   $Y^t$   . forward-pass
    L   $NLL_{loss}^{batch}$   $- P_{k \ 1}^{n} P_{j \ 1}^{m} (y_j)$   ; $y_j \in Y^t$
    . loss function for batch b containing n instances
    $W_{t \ 1}^S \leftarrow W_t^S - \eta$ O $(W_t^S; A_{t;b})$   .
    Back-propagation and model updates with learning rate $\eta$
    CLIENTBACKPROP($dA_{t;b}$   O $(A_{t;b}; W_t^S)$)   . in
    Algorithm 1
end procedure
```

IV. Dataset

A. Dataset Details

We used the TMD dataset as given in [11] for transport mode detection. The dataset consists of 37 features and a label for an instance. The description of the dataset is given in Table. I.

B. Preprocessing

We first normalized the dataset using the standard score using sci-kit learn library.

C. Splitting

We divided the client dataset D_C to D_{TR} and D_{TE} using Scikit-learn's train_test_split function with test_size set to 0.2.

TABLE I
List of features and target label space [12]

Features	Features
time	accelerometer [a]
game_rotation_vector [a]	gyroscope [a]
gyroscope_uncalibrated [a]	linear_acceleration [a]
orientation	rotation_vector [a]
sound [a]	speed [a]
Label Description	**Label Space**
Target Transport mode	f Bus, Car, Still, Train, Walkingg

[a] (min, max, mean, std)

V. Simulation Results

Table. II shows the layered architecture for splitNN W^{SP} and base-NN W^B. The Table. II also indicates the W^C, W^S, and the cut-off layer. The Dropout is set with p   0:2 and learning rate is lr   0:003. We used stochastic gradient descent (SGD) optimizer for the training of all models. We used PyTorch [13] for the training of both splitNN W^{SP} and baseline-NN W^B.

We trained the splitNN W^{SP} and base-NN W^B for 500 epochs. We used the weights of NN with maximum validation accuracy for further processing. Fig. 2 shows the training accuracy and training loss for splitNN W^{SP} and base-NN W^B. The training accuracy and training loss of W^{SP} is relatively better than the training accuracy and training loss of W^B because gradient for W^{SP} are calculated twice due to split learning.

Fig. 3 shows the test accuracy and test loss for splitNN W^{SP} and base-NN W^B.

Table. III shows the corresponding performance metrics such as precision, recall, F1-score and accuracy on test dataset

TABLE II
LAYERED ARCHITECTURE FOR SPLITNN W^{SP} AND BASE-NN W^{B}

		base-NN W^{B}	splitNN W^{SP}
Layer	Activation	Value	W^{C}/ W^{S} / cutoff-layer
Input	-	(37,)	W^{C}
Dense	Relu & Dropout	800	W^{C}
Dense	Relu & Dropout	512	cutoff-layer
Dense	Relu & Dropout	800	W^{S}
Dense	Relu & Dropout	512	W^{S}
Dense	Relu & Dropout	800	W^{S}
Dense	LogSoftmax	5	W^{S}

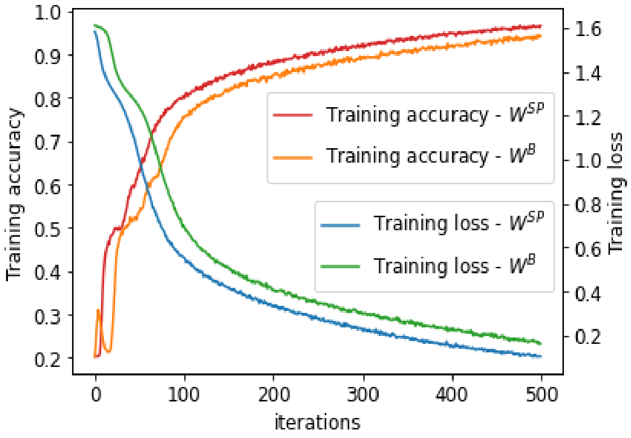

Fig. 2. Training accuracy and training loss for splitNN W^{SP} and base-NN W^{B}

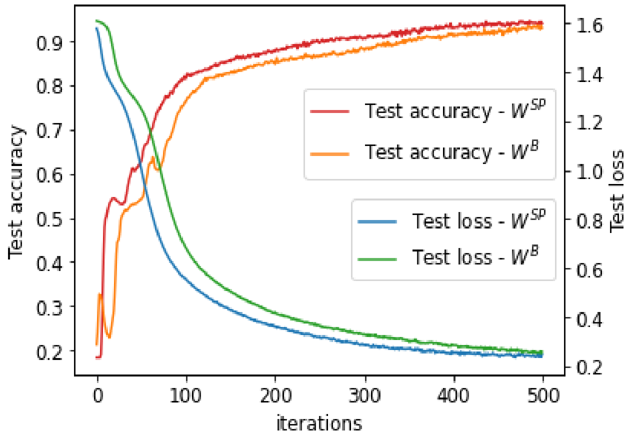

Fig. 3. Test accuracy and Test loss for splitNN W^{SP} and base-NN W^{B}

TABLE III
PERFORMANCE METRICS OF W^{B} AND W^{SP} ON TEST DATASET D_{TE}

	Precision	Recall	F-1	Accuracy
W^{B}	0.94	0.94	0.94	0.94
W^{SP}	0.95	0.94	0.94	0.94

D_{TE}. The performance metrics of splitNN W^{SP} and base-NN W^{B} on test dataset D_{TE} are almost same.

VI. Conclusion

TMD plays a significantly important role in human activity recognition. In this paper, we did split learning for TMD by establishing splitNN between a client and a server. The splitNN shows the same classification performance as typical deep NN. However, split learning is more robust to the inference attacks, thus preserving the privacy of the client's raw data. Moreover, split learning requires fewer computation resources from the client end.

References

[1] I. Yaqoob, U. Majeed, and C. S. Hong, "An Information Fusion based Framework for Social Big Data," Proc. of the KIISE Korea Computer Congress (KCC), pp. 1315–1317, 2019.

[2] L. U. Khan, W. Saad, D. Niyato, Z. Han, and C. S. Hong, "Digital-Twin-Enabled 6G: Vision, Architectural Trends, and Future Directions," 2021.

[3] U. Majeed and C. S. Hong, "Smart Patient Monitoring over a M2M Framework," Proc. of the KIISE Korea Computer Congress (KCC), pp. 430–432, 2017.

[4] U. Majeed, L. U. Khan, I. Yaqoob, S. A. Kazmi, K. Salah, and C. S. Hong, "Blockchain for IoT-based smart cities: Recent advances, requirements, and future challenges," Journal of Network and Computer Applications, vol. 181, p. 103007, 2021.

[5] U. Majeed and C. S. Hong, "FLchain: Federated Learning via MEC-enabled Blockchain Network," in *2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, 2019, pp. 1–4.

[6] U. Majeed, L. U. Khan, and C. S. Hong, "Cross-Silo Horizontal Federated Learning for Flow-based Time-related-Features Oriented Traffic Classification," in *2020 21st Asia-Pacific Network Operations and Management Symposium (APNOMS)*, 2020, pp. 389–392.

[7] L. U. Khan, W. Saad, Z. Han, E. Hossain, and C. S. Hong, "Federated Learning for Internet of Things: Recent Advances, Taxonomy, and Open Challenges," 2020.

[8] U. Majeed and C. S. Hong, "Blockchain-assisted Ensemble Federated Learning for Automatic Modulation Classification in Wireless Networks," Proc. of the KIISE Korea Computer Congress (KCC), pp. 756–758, 2019.

[9] U. Majeed, S. S. Hassan, and C. S. Hong, "Cross-Silo Model-Based Secure Federated Transfer Learning for Flow-Based Traffic Classification," in 2021 International Conference on Information Networking (ICOIN), 2021, pp. 588–593.

[10] C. Thapa, M. A. P. Chamikara, and S. Camtepe, "SplitFed: When Federated Learning Meets Split Learning," arXiv e-prints, p. arXiv:2004.12088, Apr. 2020.

[11] C. Carpineti, V. Lomonaco, L. Bedogni, M. D. Felice, and L. Bononi, "Custom Dual Transportation Mode Detection By Smartphone Devices Exploiting Sensor Diversity," in 2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), 2018, pp. 367–372.

[12] "TMD Dataset - 5 seconds sliding window, Version 1." [Online]. Available: https://www.kaggle.com/fschwartzer/tmd-dataset-5-seconds-sliding-window

[13] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in Advances in Neural Information Processing Systems, vol. 32, 2019.