Federated Learning System for Transportation Mode Prediction based on Personal Mobility Data

Fuxun Yu[1], *Zirui Xu*[1], *Zhuwei Qin*[2], *Xinda Wang*[1], *Xiang Chen*[1]
[1]*George Mason University*
[2]*San Francisco State University*
{*fyu2, zxu21, xwang44, xchen26*}*@gmu.edu, zwqin@sfsu.edu*

1 Motivation

Personal mobility trajectories/traces could benefit a number of practical application scenarios, e.g., pandemic control, transportation system management, user analysis and product recommendation, etc. For example, Google COVID-19 Community Mobility Reports [1] demonstrated the daily people movement trend and have been utilized to accurately predict the influence of the community [2] like traveling agents, retail enterprises, etc. On the other hand, such mobility traces are also privacy-critical to the users, as they contain or can be used to infer highly private personal information, like home/work addresses, activity patterns, etc. Therefore, how to effectively utilize the data with high privacy-preserving degree, as well as to benefit the real-world applications remains challengeable.

In this project, we propose to apply the novel Federated Learning (FL) [3] framework to address the privacy preserving service-level requirement. As a deep-learning (DL) distributed training framework, Federated Learning could enable model training on the local devices without needs to upload the users' private data, thus greatly enhancing the privacy preserving capability as well as maintaining similar convergence accuracy of the model. Therefore, applying FL in the personal mobility data-related use scenarios have three major benefits: (1) High Privacy-Preserving Capability: The personal mobility data stays in the users' local devices and do not need to be sent to the central server, thus greatly reducing the risk of personal data leakage; (2) Implementation Efficiency: As there is no need to transmit the data to the central server, both the communication cost and the information transmission encryption efforts could be saved, thus achieving higher implementation efficiency; (3) Large-Scale User Participation: Meanwhile, the distributed training capability of FL enables salable amounts of users to flexibly participate in the training process, thus enhancing the overall application performance.

2 Application Scenario

Our application scenario is transportation mode prediction based on personal mobility traces, which is shown in Fig. 1. In practice, personal mobility traces can be of various types of transportation modes, like walking, car, bus, subway, train, etc. Understanding and analyzing the personal transportation mode is of great importance for public transportation system management, especially for metropolis cities like Beijing/Tokyo/New York, etc., which have large population and high traffic volumes. For example, analyzing users' traffic modes and aggregating the traffic pattern for major streets in a large scale could help

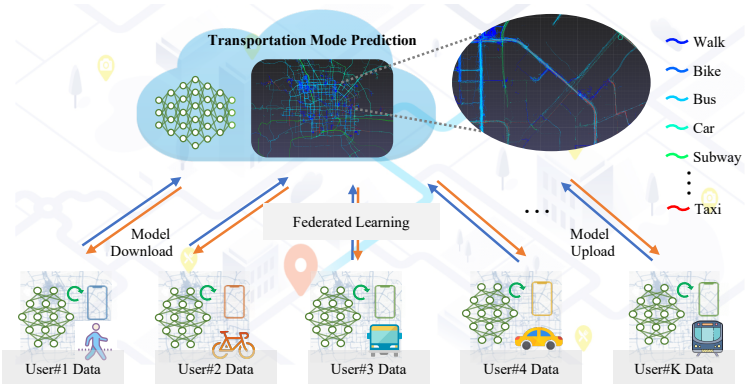

Figure 1: The overview of federated learning (FL) framework for collaborative transportation mode prediction task. One user' personal traces could be of different transportation modes, e.g., train, walk, car, etc. Leveraging the FL framework, we could collaboratively train a DNN-based classifier with strong privacy-preserving capability. The model can be then used to help automatically predict the traffic mode for intelligent transportation system management.

make wise decisions to allocate public transportation resource, e.g., adaptively allocate more resources in heavy time slots/streets to relieve the traffic burden.

However, the personal mobility data contributed by the community can be both labeled and unlabeled, of which unlabeled ones are mostly common. Therefore, automatically and accurately predicting the transportation mode for large amount of unlabeled user trajectories is one of the key tasks for the aforementioned intelligent transportation management system. In this work, we utilize deep neural network (DNN) as our base machine learning model for transportation mode prediction. Combined with privacy-preserving Federated Learning scheme, we show that we could collaboratively train a highly accurate DNN model for automatic transportation mode prediction.

3 Privacy Preserving Level

Our design could provide level-2 privacy preserving capability. Specifically, level-1 privacy preserving, i.e., unsafe data uploading channel, is not one concern in our framework since the users' private data remains in the local without uploading. Similarly, for the level-2 privacy preserving, even though the service provider and certain malicious users exist, they cannot access or are hard to recover other benign users' private data in the FL framework only based on malicious queries, which ensures the privacy in level-2 circumstances.

But for level-3 privacy preserving, if both service provider and users exist and can access the inter-service DNN model and issue different queries, certain reverse-engineering algorithms might be potentially used and cause data leakage. For example, FL-oriented attacks such as *Deep Leakage from Gradients* [5] could leverage the model gradients from central server and fake user's generated fake input queries to recover part of other users' private data.

4 System Design

System Overview: The overview of our project is shown in Fig. 2, which includes four major steps: (1) dataset collection, (2) data pre-processing, (3) federated training, and (4) model evaluation. For data collection, we build our project upon one large-scale personal mobility dataset, Geolife [4], which

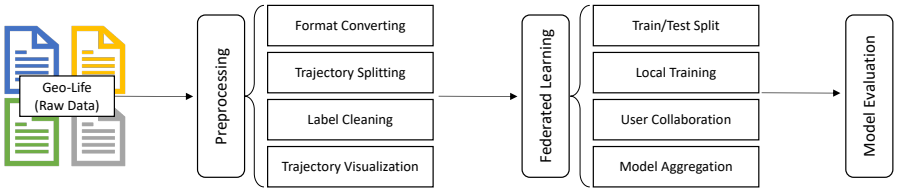

Figure 2: Implementation flow of the proposed framework. Three major steps include data cleaning, federated learning and model evaluation.

include 182 users' personal trajectories collected in Beijing, China within a 5-year range. The raw data (GPS-based traces) of Geolife are sequences of points sampled and annotated with long-, latitude and time stamps properties, etc. To enable the DNN-based prediction, we then conduct data cleaning and preprocessing including format converting, trajectories splitting and etc. After the data pre-processing, we could represent the personal mobility trajectories in the standardized format of images (1x32x32, 1x64x64, etc.) We then launch our learning system and conduct federated training upon the trajectory image dataset. In this step, we conduct federated learning with different settings to evaluate the robustness of our framework, like different number of participant users, different amount of training data, etc. Finally, we compared our methods with centralized training where all private data are sent to the central model. Without loss of generality, we show that we could achieve on-pair accuracy with centralized learning, as well as providing level-2 privacy preserving capability.

4.1 Geo-Life Dataset Description

GeoLife is a location-based dataset, which enables users to share life experiences and build connections among each other using human location history [4]. In this work, we specifically use the transportation mode labeling and conduct transportation mode prediction.

Dataset Overview: The raw data of Geolife contains 182 users, and each user's GPS trajectories are separatedly stored as .plt files with optionally a labels.txt file specifying the mode of transportation employed for a given time interval. One user's trajectory is a sequence of time-stamped points:

$$\text{Traj} = [p_0,\ p_1,\ ...,\ p_k], \quad \text{where} \quad p_i = (lat_i,\ long_i,\ t_i), \quad \forall i < k, \tag{1}$$

where $lat_i, long_i$ is the location coordinates represented by latitude and longitude, and t_i is the corresponding timestamp. The transportation mode of one trajectory can be one of 10 classes: 'walk', 'bike', 'bus', 'car', 'subway', 'train', 'airplane', 'boat', 'run', 'motorcycle' and 'taxi'. In Geolife dataset, many users' trajectories are unlabeled or partially labeled. Therefore, in our project, we utilize only the labeled trajectories for training purposes.

4.2 Data Pre-processing

To utilize the data for CNN-based transportation mode classification, we first conduct data pre-processing to clean the noisy data and also transform the excel-like sequence format into image format. After that, the data pre-processing includes two major steps: (1) Trajectories splitting and cleaning, and (2) Trajectories visualization.

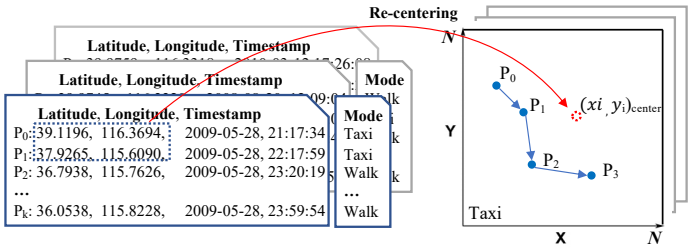

Figure 3: Trajectory Visualization Illustration.

Trajectories Splitting and Cleaning: We first convert the original raw .plt files to .csv files. One individual user's .plt file on one day can contain multiple separate trajectories and potentially in different transportation modes, e.g., from home to work, and then to the gym and then to home. Therefore, we then split each individual file into multiple separate trajectories and selected only the labeled trajectories based on the labeling information. Also, we remove the trajectories with less than 20 points to avoid including noisy or incomplete data. After the trajectories splitting and cleaning step, we finally got 2785 trajectories within 10 transportation modes.

Trajectories Visualization: As we mainly use CNN-based model, we then transform the sequential time-stamped data into the image format to better capture the locality patterns of the trajectories. The trajectories visualization is shown in Fig. 3, which includes two steps: re-centering and visualization.

Re-Centering: Original trajectories are represented in latitude and longitude in Beijing. To visualize all trajectories, we first need to re-center all the trajectories, for which we use the mean center coordinates, i.e., transforming the $(lat, long)$ to re-centered (x, y) coordinates:

$$(x_i, y_i) = (lat_i - \sum_{j}^{n} lat_j/n, \quad long_i - \sum_{j}^{n} long_j/n). \tag{2}$$

Visualization: After re-centering, we could then visualize the trajectories in the image format. Specifically, this requires setting a window size around the central point. Large window size will cause loosing the trajectories details while too small window size will lead to partial trajectories being outside the visualization window. Here we set the half of window size $(N/2)$ as the median length of all trajectories, which could capture most trajectories major part. After that, we visualize the trajectory in a matrix M_i of size $N \times N$ for visualization:

$$M_i[N,\ N] = \begin{cases} 1, & \text{If } (x_i, y_i) \in Traj, \\ 0, & \text{Else.} \end{cases}$$

The trajectory matrix M_i will be re-scaled to fixed resolutions (e.g., 32×32, 64×64, etc.) for our following model training and evaluation.

4.3 Federated DNN Training

After data pre-processing and visualization, we could then launch the federated DNN training upon the image-format trajectories dataset. The overview of federated DNN training for transportation mode classification is shown in Fig. 1. Federated learning (FL) [3] is first proposed by Google as an efficient distributed training algorithm with high data privacy preserving capability. In FL, users conduct model training locally using their private data and only upload the model weights periodically, thus greatly reducing the risks of data leakage .

DNN Model Structure: We select CNN structures as our base classification model due to their exceptional classification performance in common benchmarks. We adopt a VGG-based network structures but shrink the model depth and width considering the lower trajectory complexity compared to common object classification tasks. Specifically, the model follows a 11-layer model structure of $[64, M, 128, M, 256, M, 256, M, 256, M, FC]$, where numbers (64, 128, 256) indicate the convolutional layer with corresponding numbers of filters, M indicates the max-pooling layer, and FC indicates the fully connected layer.

Federated Training: We simulate the federated training by randomly splitting the visualized trajectories dataset into K users. During local training, each user holds a model weights replica $f(\omega)$ and conducts local loss minimization:

$$\text{Minimize}_{\omega} \quad \text{Loss}_{ce} = -(Y \log(f(\omega)) + (1 - Y) \log(1 - f(\omega))), \tag{3}$$

where Y is the one-hot label of the transportation mode and $f(\omega)$ is the prediction results of the model. During local training, all user's private local data remains locally without needs to be uploaded.

After each local training epoch, we then conduct model weights averaging to get the global model $F_g(\omega)$ following FedAvg [3]:

$$F_g(\omega) = \sum_{i}^{K} F_k(\omega) \quad \text{where} \quad F_k(\omega) = \frac{1}{K} f_k(\omega). \tag{4}$$

The weights averaging in this step requires all user to upload the model weights through the communication channel. After central server conducts model averaging, the averaged model weights will be downloaded and synchronized among all users. Usually, the FL training repeats the aboe two steps in multiple epochs until model convergence.

FL Privacy Analysis: Here we analyze the privacy preserving capability and potential risks in the aforementioned framework.

Level-1 and Level-2 Privacy: Under the aforementioned FL training scheme, even with unsafe communication channels and non-trustable users, the real user's private data remains locally without going through the communication channel, fulfilling the level-1 and level-2 privacy preserving requirement.

Level-3 Privacy: However, for level-3 privacy requirement where both users and service providers are non-trustable, certain FL attacks could be launched to steal other users' private data based on the collaboration of the central server and the fake user, such as deep leakage from gradients [5].

5 Experimental Evaluation

Experiment Setup: We build our project using PyTorch deep learning library. For trajectories visualization setting, we use $32 \times 32 \times 1$ as the image dimension. The trajectories dataset contains 2785 trajectories with 10 classes of transportation mode labels. For our evaluation, we randomly split the dataset into training set (2000 samples) and testing set (785 samples).

FL Performance vs Centralized Training: We first compare our FL training performance with regular centralized training. For our FL training, we simulate FL framework by randomly allocating data into 2 clients, 4 clients and 8 clients. And the clients then conduct FL training by FedAvg and only upload/download model weights for collaboration. For centralized training, all data is uploaded to the central server for model training.

The comparison results are shown in Table. 1. The FL training framework achieves on-pair classification accuracy (68.13% for 8-client FL) with centralized

Table 1: Accuracy of FL Training vs. Centralized Training.

Setting	Accuracy
FL (2 clients)	64.46%
FL (4 clients)	66.88%
FL (8 clients)	**67.13**%
Central Training	**67.52**%

Table 2: Accuracy of FL (4 clients) w.r.t Number of Training Samples.

Setting	Accuracy
250 Samples	60.76%
500 Samples	63.31%
1000 Samples	65.35%
2000 Samples	**66.62**%

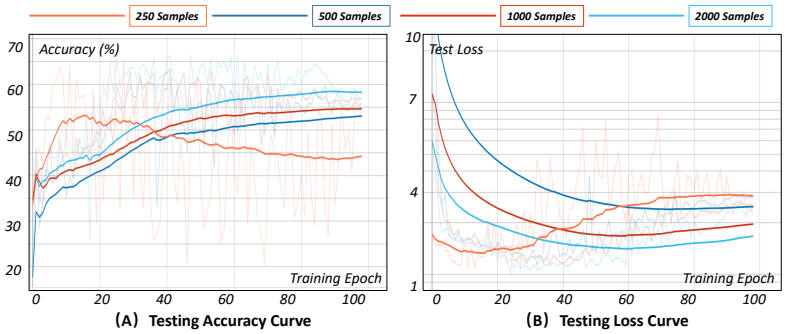

Figure 4: Test Accuracy and Test Loss Curve during FL Training.

training (67.52%). But FL avoids all data transmission cost to the central server, which provide much high privacy preserving capability.

FL Performance w.r.t Number of Training Samples: As we have mentioned before, trajectories uploaded by users can be mostly unlabeled. For example, by pre-processing the entire Geolife dataset from a 5-year time range, only 2785 trajectories are labeled and processed as training data. Therefore, we evaluate the classification accuracy influence with different number of available training trajectories to analyze the potential performance influence on FL.

Fig. 4 shows the test loss and accuracy curves during the FL training process. As we can see, with 250 samples as training data, FL training appears to be over-fitted to the limited training samples and the test accuracy decreases in the later epochs. With more data involved, the over-fitting phenomenon becomes less prominent, i.e., the testing loss gradually decreases and the test accuracy increases until convergence. Table 2 shows the final accuracy results: The accuracy of Fl training with 250 samples only achieves 60.76% accuracy, and gradually improves to 66.62% (+5.9%) with full 2000 training samples.

The above results demonstrate the importance of enough amounts of training data, and also indicate the current accuracy is highly limited to the number of training samples. With more samples collected, the performance could potentially be further improved.

6 Conclusion

In this work, we build an FL system for transportation mode prediction based on personal mobility data. Utilizing FL-based training scheme, we could yield highly accurate DNN models similar to the centralized training performance, but also provide level-2 privacy preserving capability. In practice, we hope the proposed transportation mode prediction system could serve as a prototype and potentially benefit the transportation data analysis and help make wise decisions to manage public transportation resources.

References

[1] Google covid-19 community mobility reports. https://www.google.com/covid19/mobility/. Accessed: 2020-12-18.

[2] Google mobility trends: How has the pandemic changed the movement of people around the world? https://ourworldindata.org/covid-mobility-trends. Hannah Ritchie, June 02, 2020.

[3] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017.

[4] Yu Zheng, Xing Xie, Wei-Ying Ma, et al. Geolife: A collaborative social networking service among user, location and trajectory. *IEEE Data Eng. Bull.*, 33(2):32–39, 2010.

[5] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. In *Advances in Neural Information Processing Systems*, pages 14774–14784, 2019.