Semi-Supervised Federated Learning for Travel Mode Identification From GPS Trajectories

Yuanshao Zhu, *Graduate Student Member, IEEE*, Yi Liu, *Student Member, IEEE*, James J. Q. Yu, *Senior Member, IEEE*, and Xingliang Yuan, *Member, IEEE*

***Abstract*— GPS trajectories serve as a significant data source for travel mode identification along with the development of various GPS-enabled smart devices. However, such data directly integrate user private information, thus hindering users from sharing data with third parties. On the other hand, existing identification methods heavily depend on the respective manual travel mode annotations, whose production is economically inefficient and error-prone. In this paper, we propose a Semi-supervised Federated Learning (SSFL) framework that can accurately identify travel modes without using users' raw trajectories data or relying on notable data labels. Specifically, we propose a new identification model named convolutional neural network-gated recurrent unit model in SSFL to accurately infer travel modes from GPS trajectories. Second, we design a pseudo-labeling method for the clients to set pseudo-labels on their local unlabeled dataset by using a small public dataset at the server. Furthermore, we adopt a grouping-based aggregation scheme and a data flipping augmentation scheme, which can boost the convergence and performance of the proposed framework. Comprehensive evaluations on a real-world dataset show that SSFL outperforms centralized semi-supervised baselines and is robust to the non-independent and identically distributed data commonly seen in practice.**

***Index Terms*— Travel mode identification, GPS trajectory, federated learning, deep learning, semi-supervised learning.**

I. Introduction

DATA is considered the new oil, as it is the fuel powering Artificial Intelligence (AI) and creates tremendous value for many domains from business activities to scientific research [1]. AI technologies have given new vitality to all walks of life and spawned a series of emerging applications, especially data-driven Intelligent Transportation System (ITS) applications [2]–[4]. For example, AI-based travel mode identification applications is an indispensable component of ITS [5], [6], which can enable governments, companies, and institutes to understand human behaviors and urban management better and planning [3], [7], [8]. Therefore, in this paper, we focus on studying how AI technology can empower travel mode identification applications in ITS.

For data-driven travel mode identification applications, the development of advanced data collection technologies in GPS-enabled smartphone devices has provided a new oil for research, i.e., a large number of user private trajectory data composed of GPS records [5], [9], [10]. These trajectories provide a path for the development of novel travel mode identification techniques [11], [12]. Generally, using GPS trajectories for travel mode identification comprises three steps. First, third-party entities directly collect private GPS trajectories data and upload them to a cloud server. Second, engineers use data mining and feature engineering methods to extract motion attributes (e.g., speed) of the collected data. Third, supervised centralized machine learning methods are adopted to perform the final identification [13]–[17]. Such a method has achieved great success in the research of travel mode identification.

However, in most cases, putting together a labeled dataset for a given travel mode identification task is a time-consuming, expensive, and complicated endeavor [18]. It generally requires a large amount of GPS trajectory as well as real ground-truth travel mode label information to train high-quality identification models [13], [14], [19] and may involve soliciting data from different entities [20], [21]. On the other hand, GPS trajectories information is closely related to user privacy, which makes it impractical for third-party entities to directly upload or share raw data to the server. In practice, the General Data Protection Regulation (GDPR) [22] does not allow entities to use data sharing to build powerful AI models. This means that the data sharing scheme cannot be directly used in the construction of the traffic travel mode identification model because it seriously violates the user's data privacy rights [23]–[25]. Therefore, we need to re-think about how to obtain the "right" data when much of the attention in AI has been focused on how to best leverage an existing data source and build a powerful model from that.

In a distributed learning community, the question of "how to get the right data" is solved by designing a privacy-preserving AI framework: different clients (or devices) jointly train a shared global model without sharing the raw data [1], [18].

Manuscript received January 26, 2021; revised April 22, 2021; accepted June 15, 2021. This work was supported in part by the Stable Support Plan Program of Shenzhen Natural Science Fund under Grant 20200925155105002, in part by the Guangdong Provincial Key Laboratory under Grant 2020B121201001, in part by the General Program of Guangdong Basic and Applied Basic Research Foundation under Grant 2019A1515011032,in part by the ARC Discovery Project under Grant DP200103308, and in part by the Data61-Monash Collaborative Research Project under Grant CRP43. The Associate Editor for this article was T.-H. Kim. *(Corresponding author: James J. Q. Yu.)*

Yuanshao Zhu and James J. Q. Yu are with the Guangdong Provincial Key Laboratory of Brain-inspired Intelligent Computation, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China (e-mail: yasozhu@gmail.com; yujq3@sustech.edu.cn).

Yi Liu was with the Guangdong Provincial Key Laboratory of Brain-inspired Intelligent Computation, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China. He is now with the Department of Computer Science, City University of Hong Kong, Hong Kong SAR 518057, China (e-mail: 97liuyi@ieee.org).

Xingliang Yuan is with the Faculty of Information Technology, Monash University, Melbourne, VIC 3800, Australia (e-mail: xingliang.yuan@monash.edu).

Digital Object Identifier 10.1109/TITS.2021.3092015

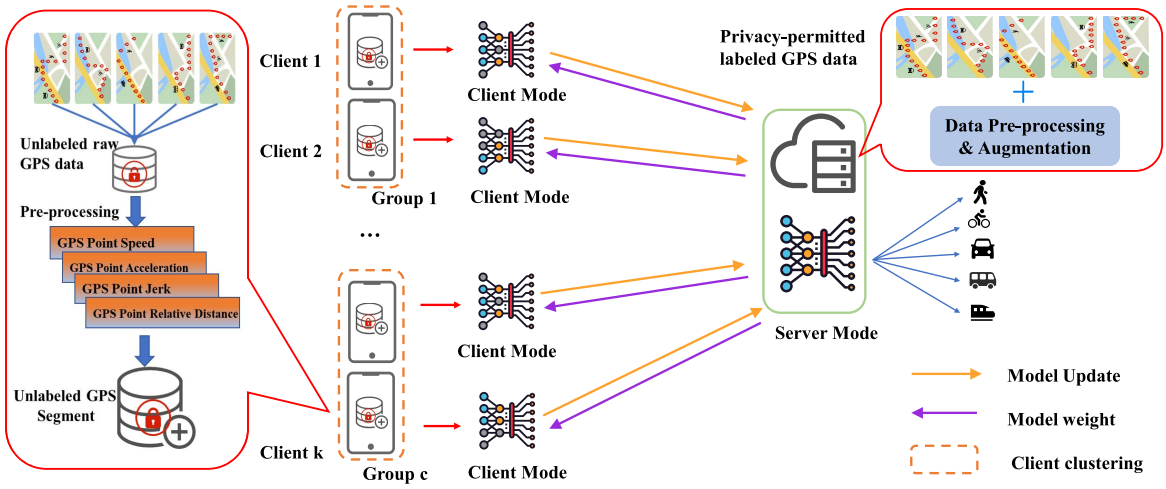


Fig. 1. Overview of the semi-supervised federated learning travel mode identification system. The clients collect raw GPS data without travel mode annotations and perform data processing. The server collects raw GPS data when privacy permits and annotate them, then preprocessing and augment data. Finally, the server and the client collaboratively train a DNN model for travel mode identification.

Furthermore, in the machine learning community, researchers generally use the semi-supervised learning paradigm [26] to respond to the lack of real ground-truth issues. Therefore, inspired by the above techniques, we design a novel Semi-supervised Federated Learning (SSFL) framework to bridge the above research gap. Specifically, we consider a more realistic scenario whose sketch is illustrated in Fig. 1, in which the cloud server holds a small privacy-permitted dataset with travel mode labels, and users distributedly and privately possess their unlabeled trajectories data with personal privacy. In our SSFL settings, travel model identification still faces the following challenges:

- ***Lack of Labeled data:*** Existing travel mode identification approaches and the applications it spawned are based on an impractical assumption: all clients hold a precisely labeled dataset. However, it is difficult to guarantee that each client holds such data in practice. While the clients can generate much data, they are typically unincentivized to also develop data annotations. Therefore, how to maximize the use of labeled data on the server and unlabeled data on the client to optimize the performance of the classification model is one of the challenges we face in this work.
- ***Privacy Protection:*** While the clients collect a large amount of data, due to privacy concerns and legal supervision, the server cannot directly access or obtain the client data. Hence, we need to find a good solution to construct a travel mode identification model without compromising client privacy.
- ***System and Statistical Heterogeneity:*** Clients may collect data by different devices (e.g., mobile phones, smartwatches), use different systems (e.g., IOS, Android) as well as collect local data in different environments [25], [27], [28]. On the other hand, the local dataset of each client may have different distribution and volume for each category. For example, some users may often walk, and others would prefer to drive, which violates the assumption of independent and identical distributed (i.i.d.) that is commonly used in machine learning. Furthermore, there is no guarantee that all clients keep connected with the server during model training. In summary, these systems and statistical heterogeneity bring challenges to modeling and optimization.

To address these challenges, we design a new deep neural network architecture and propose a novel clustering aggregation algorithm in our semi-supervised federated learning system for travel mode identification. In particular, the main contributions of this paper are summarized as follows:

- **A decentralized semi-supervised machine learning system for privacy-preserving travel mode identification:** We consider a more realistic assumption that the client data is unlabeled and we propose a novel semi-supervised federated learning system to accurately infer the travel mode without compromising privacy. Specifically, we use pseudo-labeling methods to label the client's data and combine federated learning (FL) to perform local model training, which allows users to share models instead of raw data to respect the privacy [23], [29].
- **A new deep neural network (DNN) architecture for travel mode identification:** We propose a novel convolutional neural network (CNN)-gated recurrent unit (GRU) model, which uses CNN to extract the trajectory changes and fine-grained features of GPS trajectories and use GRU to capture the time-correlation of GPS data. In this way, we can further capture the temporal and spatial correlation of GPS trajectories, thereby achieving fine-grained correlation modeling and accurate travel mode identification.
- **A novel group clustering aggregation algorithm and data augmentation methods:** We propose a group-clustering (GCL) algorithm to group client and aggregate local model parameters according to the distribution of the client dataset. The experimental results show that the GCL algorithm can boost the convergence speed of the model in a non-independent and identically distributed (non-i.i.d.) data distribution. Besides, The data

augmentation method we proposed can also be effective in improving the performance of the model.

- **Extensive experimentation and model validation:** We conducted a comprehensive set of experiments, showing that our system achieves over 90% accuracy with only 50% of labeled data, regardless of the data distribution. This indicates that our system does not only achieve accurate travel model identification on a real-world dataset but also efficient on non-i.i.d. data distribution.

To the rest of this paper is organized as follows. We first review the background of travel mode identification in Sec. II. Then, we present the problem definitions and challenges in Sec. III. Sec. IV elaborates on the proposed system. We conduct a series of experimental and analyses in Sec. V. Finally, we conclude this paper in Sec. VI.

II. Related Work

A. Travel Mode Identification Approaches

For the research on GPS data-driven travel mode identification, previous work has focused on developing methods that follow the aforementioned two-step identification paradigm proposed by Zheng *et al.* [13], [14]. In their pioneering study, the long trajectories are divided into multiple trip segments by using domain-specific commonsense information. Then hand-craft features (i.e., variance and mean of speed) of each segment are computed and used as inputs to the classification task to infer travel modes. These works provide a fundamental scheme for travel mode identification [15], [16], [30], [31], but there is still a need to perform high-dimensional feature extraction and follow this general approach to achieve fine-grained travel mode identification.

Therefore, some researchers have begun to use neural networks for travel mode identification. For example, Endo *et al.* designed a image and location retention information-based methods to map GPS trajectories into images and automatically extract features by DNN. Wang *et al.* adopted sparse autoencoder-based methods to capture features and perform travel mode identification. However, despite the use of DNN for feature extraction, their solution did not achieve good results, and the accuracy was even lower than that of handcraft feature extraction. Owing to the successful application of deep neural networks in research areas such as computer vision and natural language processing, many researchers begun to apply CNN [15], [31] and recurrent neural networks (RNN) [32] or their variants (e.g., long short-term memory (LSTM) [33], [34], convolutional bi-LSTM [34]) to capture the time dependence of GPS trajectories to obtain better classification results.

While these methods have led to long-lasting progress in travel mode identification, they still face two serious challenges when accessing GPS data: data privacy issues and data availability issues.

B. Privacy-Preserving Techniques for Travel Mode Identification

To response the first challenge, researchers have proposed many privacy-preserving techniques for travel mode identification. These techniques are mainly divided into two categories: encryption-based methods and privacy-preserving machine learning based methods.

1) Encryption-Based Methods: Such techniques generally use encryption algorithms (e.g., homomorphic encryption) [35] to encrypt data or use secure computing methods (e.g., secure multi-party computing) [36] to ensure the security of the computing results, thereby preventing malicious tampering and malicious attacks by attackers. However, these methods are not compatible with complex machine learning models due to the requirements of encryption algorithms, which hinders their own development.

2) Privacy-Preserving Machine Learning Based Methods: To be compatible with complex machine learning models, privacy-preserving machine learning technologies came into being. For example, Zhu *et al.* [37] utilize this technique to propose a privacy-preserving travel mode identification application. Despite the efforts on privacy preservation, the results cannot achieve a satisfactory trade-off between identification performance and privacy protection. To address this problem, Liu *et al.* [38] introduced federated learning to ITS, enabling them to conduct ITS research without compromising privacy or violating regulatory regulations.

Although the above methods have achieved great success in the field of privacy-protected traffic mode identification, they are still troubled by the availability of data.

C. Semi-Supervised Learning for Travel Mode Identification

To response the second challenge, researchers have made many efforts in the field of semi-supervised learning. Recall that, massive labeled data can provide great support in research on intelligent transportation systems, especially in travel mode identification. However, acquiring the manual labels has shown to be highly difficult in practice [30], [31], [39]. Therefore, researchers proposed some novel semi-supervised machine learning models, e.g., Autoencoder [31] and ensemble LSTM [30] to solve the problem of the limited amount of data. However, none of these researches considered both privacy-preserving and insufficient data issues.

Inspired by the previous work, this paper considers the problem of semi-supervised learning and privacy protection in travel mode identification. We propose a semi-supervised federated learning model to accomplish the task and address the aforementioned challenges, which can achieve high identification accuracy while protecting privacy.

III. Preliminaries

In this section, we give a brief introduction to the federated learning system and some related definitions in this work.

A. Federated Learning and Federated Averaging Algorithm

Federated learning system was proposed by McMahan *et al.* [29], whose idea of is to allow user build machine learning models while keeping user's data locally. Typically, FL systems employ the federated averaging (FedAvg) algorithm to train a shared global model

Algorithm 1 Federated Averaging (FedAvg) Algorithm

Server:

Initialize global model parameters ω^0
for each round $t = 1, 2, \ldots$ **do**
  $\mathcal{K}_t \leftarrow$ (Sample a subset client from $\mathcal{K}$)
  for each client $k \in \mathcal{K}_t$ **in parallel do**
    $\omega_k^{t+1} \leftarrow$ ClientModelUpdate(k, ω^t)
  end for
  $\omega^{t+1} \leftarrow \frac{1}{|\mathcal{K}_t|} \sum_{k \in \mathcal{K}_t} \omega_k^{t+1}$
end for

ClientModelUpdate (k,ω):

η is the learning rate,
for each epoch $e = 1, \ldots, E$ **do**
  $\omega \leftarrow \omega - \eta \nabla \ell(\omega)$
end for
return ω to server

collaboratively through multiple decentralized users without sharing the raw data [29]. Let $\mathcal{K}$ represent the set of clients, and $\mathcal{D} = \{D_1, D_2, \cdots, D_k\}$ denote the dataset owned by each client. The training steps of FedAvg are outlined as follows:

- ***Initialization:*** First, at each round of training t, the server randomly selects a subset of clients, i.e., $\mathcal{K}_t \subseteq \mathcal{K}$, from all participating clients to participate in the FL task (e.g., prediction tasks or classification tasks). Second, the server broadcasts the initialized global model parameters ω^t to each selected client $k, k \in \mathcal{K}_t$.
- ***Local Training:*** Each client trains the received global model by E epochs on its local dataset D_k. The goal of the client is to minimize the following objective function:

$$\arg\min_{\omega_k \in \mathbb{R}} L_k(\omega_k) = \frac{1}{|D_k|} \sum_{(x_i, y_i) \in D_k} \ell_i(y_i, f_k(x_i; \omega_k)), \tag{1}$$

  where D_k denotes the local dataset that contains input-output vector pairs (x_i, y_i), $x_i, y_i \in \mathbb{R}$, ω_k is the parameter of local model f_k, and $\ell_i(\cdot, \cdot)$ is a local loss function (i.e., $\ell_i(y_i, f_k(x_i; \omega_k)) = \frac{1}{2}({x_i}^T \omega - y_i)$). Then each client uploads its own model updates to the server.
- ***Aggregation:*** The server uses FedAvg [29] (as shown in Algorithm 1) algorithm to aggregate these model updates and obtains a new global model ω^{t+1} for the next iteration, i.e.,

$$\omega^{t+1} = \frac{1}{|\mathcal{K}_t|} \sum_{k \in \mathcal{K}_t} \omega_k^t, \tag{2}$$

  Note that the system repeats these training steps until the global model achieves convergence.

The goal of FedAvg is to perform multiple gradient descent optimizations on the client and reduce the communication overhead with the server.

B. Related Definitions

In this section, we introduce the definition and terminologies that are employed throughout this paper:

Definition 1 (GPS Raw Record): In this paper, the proposed method is applied to raw GPS data comprised of GPS records, which can be formally represented as follows:

$$GPS^{raw} = \{(\text{latitude, longitude, timestamp, mode})\} \tag{3}$$

Considering that raw GPS record cannot guarantee a consistent sampling rate and the accuracy of each location record, it cannot be directly applied to our proposed system. Hence, we follow the data segmentation method [13] and the feature extraction method [15] to process the unlabeled raw GPS record into a data structure that can better represent the data characteristics (refer to Sec. IV-A). Furthermore, since each client may collect GPS records with different travel modes (e.g., walk, bus, etc.), this may cause the data distribution between clients is non-i.i.d. Here, we need to define the degree of non-i.i.d. as follows:

Definition 2 (Metric R for non-i.i.d. Degree): We define the class distribution of the data D_k at the k-th client as $P_k = [l_1, l_2, \ldots, l_d]$, where l_i denotes the proportion of the i-th category in D_k, d is the total number of categories of travel modes. For all $1 \leq k \leq K$, $\sum_{i=1}^{c} P_k[i] = 1$. Therefore, the non-i.i.d. degree R_k to measure the class distribution skewness of the data D_k is defined by

$$R_k = \frac{\max(P_k) - \min(P_k)}{\sum(P_k)}. \tag{4}$$

For example, $R_k = 0$ when D_k has a uniform class distribution $P_k = [1/d, \ldots, 1/d]$. When D_k only has one class, $R_k = 1$.

Combining the above definitions, the problems targeted in this paper can be formalized as follows:

Definition 3 (Semi-Supervised Federated Learning-Based Travel Mode Prediction Problem): In our SSFL system, we use $F(\omega)$ to denote a global travel mode predictor that can infer the travel mode $\hat{y}$ on the local unlabeled GPS dataset D_k at the client. The formal definition of the problem studied in this paper is as follows:

$$\min_{\omega \in \mathbb{R}} \sum_{i=1}^{k} f_k(\omega) \underset{aggregated}{\overset{D_k}{\rightarrow}} F(\omega) \underset{predict}{\overset{D_s}{\rightarrow}} \hat{y}, \tag{5}$$

where D_s is the privacy-permitted labeled dataset at the server, and $f_k(\omega)$ is the local identification model at the client, and D_k is the unlabeled dataset at the client. The training objective of this problem is to find an optimal $F^*(\omega)$ to predict the travel mode, which is aggregated by the client's local predictor through an aggregation algorithm.

IV. Methodology

In this section, we present our solutions to address the challenges that mentioned in Sec. I. We first demonstrate the data pre-processing, data augmentation, and feature extraction methods. Second, we elaborate on the architecture of our proposed travel mode identification model. Third, we give the design of the semi-supervised federated learning system. Finally, to deal with the non-i.i.d. data issue, we propose a group clustering-based aggregation algorithm to achieve robust learning.

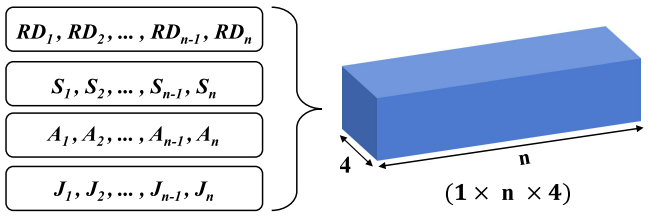


Fig. 2. The four channel structure of a GPS segment.

A. GPS Record Processing

The GPS record processing steps comprise data motion feature calculation, data augmentation, and time-domain feature extraction.

1) Capturing Motion Feature: A user's GPS trajectories with length n can be represented as a sequence of raw GPS record $g \in G$, and $G = \{g_1, g_2, \ldots, g_n\}$. Each record $g = [\text{lat}, \text{lng}, t]$ is a triple of longitude, latitude, and timestamp, which indicates the private location information of device at time t. We can capture several motion features for every GPS point based on the geographic coordinates and timestamps. The relative distance (RD) between two consecutive GPS record can be computed by Vincenty Formula [40]:

$$\text{RD}_i = \text{Vincenty}\left(\text{lat}_i, \text{lng}_i; \text{lat}_{i+1}, \text{lng}_{i+1}\right), \tag{6}$$

The time interval between two successive records can also be computed by timestamp:

$$\Delta t_i = t_{i+1} - t_i. \tag{7}$$

Based on the RD_i and time interval (Δt_i), we can calculate the speed S_i, acceleration A_i, and jerk J_i of the R_i location. These motion features can be calculated by using the following equations:

$$S_i = \frac{\text{RD}_i}{\Delta t_i}, \quad 1 \le i \le n, \ S_n = S_{n-1}, \tag{8}$$

$$A_i = \frac{S_{i+1} - S_i}{\Delta t_i}, \quad 1 \le i \le n, \ A_n = 0, \tag{9}$$

$$J_i = \frac{A_{i+1} - A_i}{\Delta t_i}, \quad 1 \le i \le n, \ J_n = 0. \tag{10}$$

2) Training Data Setup and Augmentation: After capturing the motion features of each GPS trajectories, we can get four motion feature vectors (RD, S, A, J) with fixed length n and we combine these vectors to form a tensor with 4 channels $X = \{x_1, x_2, \ldots, x_n\}$ ($x_i = [\text{RD}_i, S_i, A_i, J_i]$, i represents the i-th time step). As shown in Fig. 2, each channel represents the relative distance, speed, acceleration, and jerk, respectively. So far, we have obtained the training data X that can be used for travel mode inference through processing the raw GPS record. Considering that the amount of data X with complete annotations is too small, we define an augmentation function $AUG(\cdot)$, which flips X according to the time step arrangements of X. The implementation of this augmentation function is given by $X' = AUG(X) = \{x_n, x_{n-1}, \ldots, x_1\}$.

3) Capturing Time-Domain Feature: In addition to following previous literature on travel mode identification to design GPS data preprocessing algorithm [13], [15], [31], we believe that providing more feature information for trajectory data is beneficial for model classification. As described in Sec. II, there are many methods to further provide auxiliary features to the classification model (e.g., [19], [30], [41]), but we consider that the motion changes are discriminative with respect to various travel modes in the frequency domain.

While there are other methods for frequency-domain feature extraction, such as the discrete Fourier transform (DFT). Both DWT and DFT are widely recognized in time-series and signal processing tasks, especially for signal compression and frequency domain feature extraction. However, DWT achieves the same quality of trajectory reconstruction with slightly better performance than DFT. Furthermore, DWT provides higher spatial resolution and lower frequency resolution, which can capture the motion trends with a small number of coefficients [42]. Therefore, we deploy DWT to extract frequency domain features. Given a time sequence signal $x(t)$, DWT adopts discrete wavelets $\psi_{a,b}(t)$ to transform the input signals into the following signal:

$$d_{a,b} = \int_{-\infty}^{+\infty} x(t)\psi^*_{a,b}(t)dt = \left\langle x(t), \psi_{a,b}(t)\right\rangle, \tag{11}$$

where

$$\psi_{a,b}(t) = \frac{1}{\sqrt{2^a}}\psi\left(\frac{t}{2^a} - b\right), \quad a, b \in \mathbb{Z}. \tag{12}$$

Here, $\psi^*_{a,b}(t)$ is the complex conjugate of $\psi_{a,b}(t)$, a and b are oscillatory frequency and the shifted position of the discrete wavelet respectively.

From another perspective, DWT can also be considered as a multi-resolution decomposition of the input signal [43]:

$$\begin{aligned} s(t) &= \sum_b A_{M,b} 2^{-M/2}\varphi\left(\frac{t}{2^M} - b\right) \\ &\quad + \sum_a^M \sum_b d_{a,b}(x(t), \psi(t)) 2^{-a/2}\psi\left(\frac{t}{2^a} - b\right) \\ &\triangleq A_M(t) + \sum_a^M D_a(t), \end{aligned} \tag{13}$$

where $A_{M,b} = \left\langle x(t), \varphi_{M,b}(t)\right\rangle$ is the approximation coefficient at the decomposition M, $\varphi(t)$ is a companion scaling function. Using the Eq. (13), the signal $x(t)$ is decomposed into an approximation signal $A_M(t)$ and detail signal $D_a(t)$.

Since we need to pay more attention to the attribute trends of GPS trajectories than details when identifying travel modes. We only utilize the approximation signal $A_M(t)$ of the pre-processed attributes (RD,S,A,J) and db (daubechies) is adopted as mother wavelets to perform the decomposition. After the discrete wavelet transform, we can get 4 features with length $1/n$. This feature can be combined into a vector $\mathcal{W}$ with length n, which can then be used as the input of the identification model.

B. Travel Mode Identification Model

In this work, we design a novel DNN for travel mode identification. As shown in Fig. 3, this network is constructed by three main components, i.e., CNN, GRU, and fully connected layer (FCL).

1) Convolutional Layer: CNN is an outstanding DNN structure for feature extraction, which can also be used to acquire local data trends. Furthermore, the local connectivity in CNN can reduce the number of weights, speed up the training process, and mitigate the curse of dimensionality problem [44]. Therefore, we apply CNN to extract the motion change and high-level spatial features of GPS trajectories. In this architecture, we use three convolutional layers to extract features from the input data, and a pooling layer following each convolutional layer. Specifically, we set each convolutional layer's kernel size to (1×3) and let their strides equal to 1, then we employ a max-pooling layer with kernel size is (1×2). Finally, we use batch-normalization to normalized the model parameters.

The proposed model can obtain high-level spatial features about the changing trend of GPS trajectories after performing a series of convolution operations on the input data. However, we cannot achieve good results if we directly use these features for identification. Thanks to GRU's design based on time state propagation [45], [46], we can use GRU to learn potential time-correlations from the information extracted by CNN.

2) GRU Layer: GRU is an effective variant of the LSTM network, which has a more straightforward structure than LSTM and can also address the long dependency problem in conventional RNN models [46]. Typically, the GRU cell structure has two gates, i.e., reset gate r and update gate z. Intuitively, the reset gate determines how to combine the new input information with the previous memory. The update gate defines the amount of memory saved to the current time step.

Let x_t and h_t denote the input time series and intermediate states, $\tilde{h}_t$ denoted the candidate state respectively. At time t, the reset gate r_t and update gate z_t can be expressed as:

$$r_t = \sigma\left(W_z \cdot \left[h_{t-1}, x_t\right]\right), \tag{14}$$

$$z_t = \sigma\left(W_z \cdot \left[h_{t-1}, x_t\right]\right), \tag{15}$$

where W_z is the network weights, σ represents the sigmoid activation function, and h_{t-1} is the previous state at time step $t-1$. The reset gate can determine how much information from the previous state is passed to the current candidate state $\tilde{h}_t$:

$$\tilde{h}_t = \tanh\left(W \cdot \left[r_t \odot h_{t-1} x_t\right]\right), \tag{16}$$

where $\odot$ represents the Hadamard product. And it use tanh as an activation function can reduce the number of calculations and prevent gradient explosions. Finally, the current memory at the current time step as follows:

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t. \tag{17}$$

3) FCL Layer: As shown in Fig. 3, we use a GRU layer with 16 hidden states to extract time-correction in this model. Then, we connect the extracted high-dimensional feature attributes with the time-frequency domain features previously obtained by DWT. This combined vector is used as the input of the two consecutive FCL. Specifically, we employ Rectified Linear Units (ReLU) as the first layers due to its efficient gradient descent and backpropagation performance. Since we need to infer the travel mode through GPS trajectories, we use the softmax function as the activation function of the last layer, which can be mathematically expressed as follows:

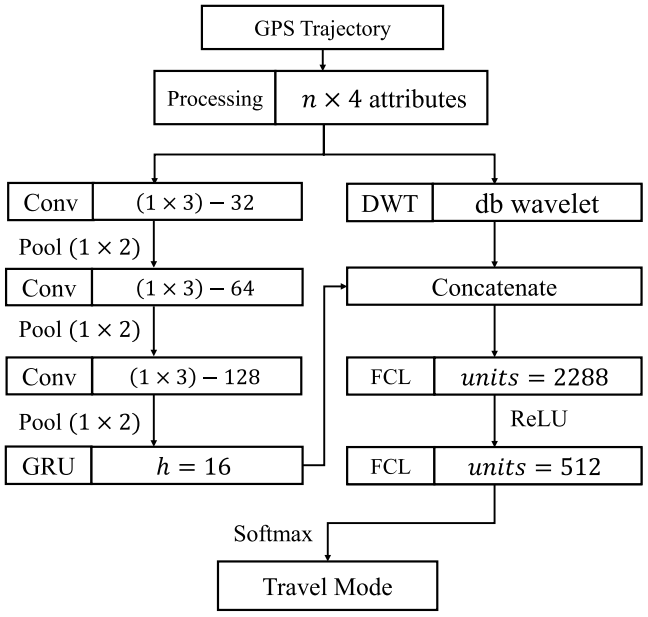


Fig. 3. The architecture of the proposed network.

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_{l=1}^{d} e^{z_l}}, \tag{18}$$

where d represents the total modes, and $\text{Softmax}(z_i)$ can be used to calculate the probability that z belongs to a certain modes.

C. Semi-Supervised Federated Learning

1) Pseudo-Labeling Training Method: As demonstrated in Fig. 1, the server holds a labeled dataset, and the clients have an unlabeled dataset in SSFL. In this context, we adopt a pseudo-labeling-based training method to achieve model training on the unlabeled dataset at the client. Since the identification model applied the softmax function on the last FCL layer, the model eventually predicts the probability that the GPS trajectories belong to a specific travel mode. Based on these characteristics, we set a threshold of τ to give a pseudo label to a training sample in the local unlabeled dataset. When the model's predicted probability of an input belonging to a specific travel mode is greater than τ, we consider this expected model has high confidence and set it as a pseudo label for the input data.

2) Training Objective and Model Averaging: Based on the pseudo-labeling training method, in SSFL, the objective function Eq. (1) mentioned in Sec. III-A can be rewritten as follows:

$$L_k = \frac{1}{|D_k|} \sum_{x_i \in D_k} \mathbf{sgn}\left(\max\left(p(\hat{y}_i)\right) \geq \tau\right) \times \ell\left(\arg\max\left(\hat{y}_i\right), f_k\left(x_i; \omega_k\right)\right), \tag{19}$$

where **sgn**$(\cdot)$ is the indicator function, $p(\hat{y}_i)$ is the prediction of the model f_k on the sample data x_i, and w_k are the weights of k-th client model f_k. Similarly, we can define the loss function on the server as follows:

$$L_s = \frac{1}{|D_s|} \sum_{(x_i, y_i) \in D_s} \ell\left(y_i, f_s\left(x_i; \omega_s\right)\right), \tag{20}$$

where y_i is the true label of x_i. The weight parameters in server model f_s and client model f_k are aggregated by global model averaging.

Typically, FedAvg [29] can be used in the SSFL system to aggregate model updates:

$$\omega_{glob} = \frac{1}{|\mathcal{K}| + 1}(\omega_s + \sum_{k \in \mathcal{K}} \omega_k). \tag{21}$$

However, if we direct use the vanilla FedAvg given in the Algorithm 1, the model diversity between different clients brings great challenges to training and aggregation [47]. To address this issue, we propose a client group clustering-based model to update the aggregation algorithm, which divides client $\mathcal{K}$ into c groups and then performs the model averaging. In particular, after completing the SSFL training of all clients' models, the server divides them into c groups ${C_i}_{i=1}^{c}$ and updates w_{glob} according to the following formula:

$$\begin{cases} \omega_{\{glob,i\}} = \dfrac{1}{|C_i| + 1}(\omega_s + \sum_{k \in C_i} \omega_k), \quad i \in \{1, 2, \ldots, e\} \\ \omega_{glob} = \dfrac{1}{c} \sum_{i=1}^{c} \omega_{\{glob,i\}} \end{cases} \tag{22}$$

The above model averaging algorithm's principle is to reduce diversity among the aggregating models, thereby speeding up the training process. The details of this algorithm will be introduced in Sec. IV-D.

While there are many methods for parameter aggregation and optimization in the Federated Learning [48]–[51], these methods have been proposed mainly to solve specific problems (e.g., resource allocation, model optimization). From a practical perspective, FedAvg is still the most dominant and efficient method [23], [25]. Therefore, we adopt the FedAvg-based grouping clustering algorithm as the main aggregation approach in this paper.

D. Group-Clustering Aggregation Algorithm

In practice, each client's data may not satisfy the assumption of i.i.d. data distribution, which leads to difficulties in the optimization and convergence of the model. To address this issue, we propose a group-clustering (GCL) aggregation algorithm to robustly aggregate the clients' updates. Specifically, we use the class distribution P_k as a measure of clustering to divide client updates into c groups. The main steps of the GCL algorithm are as follows:

- **Step 1:** At each round of training t, the client receives the global model from the server. Then each client k uses the global model to make a travel mode prediction (i.e., pseudo label) on its own local unlabeled dataset D_k and calculate the class distribution P_k.

Algorithm 2 Group-Clustering (GCL) Aggregation Algorithm for non-i.i.d. Data

Input:
Client set $\mathcal{K} = \{k_1, k_2, \ldots k_m\}$,
the number of groups c, $m > c$
Output:
Clustering result $\mathcal{C} = \{C_1, C_2, \ldots C_c\}$.
for each client $k \in \mathcal{K}$ **do**
  Calculate the class distribution P_k
end for
Random select c clients as the initial cluster centroids.
repeat
  for each client $k \in \mathcal{K}$ **do**
    Select the cluster C_c has the smallest distance to the client, and assign k to the C_c
  end for
  for each cluster $C_i \in \mathcal{C}$ **do**
    Update the cluster centroids and make the centroids has the smallest distance to each client in the cluster.
  end for
until convergence
return $\mathcal{C}$ to server.

- **Step 2:** The server uses the GCL algorithm to calculate the similarity of the class distribution P_k. Specifically, the server divides the client into c groups according to the Euclidean distance of the class distribution P_k and uses the K-means clustering algorithm. Next, the Euclidean distance of different P_k is defined as follows:

$$dist(P_i, P_j) = \sqrt{\sum_{l=1}^{d} (P_i^l - P_j^l)^2} \tag{23}$$

- **Step 3:** We consider $\mathcal{K}$ as the client set. After multiple iterations of clustering, we finally divide the client set into c groups. The P_k distance in the same group is the smallest, which means that the clients in each group have similar class distributions.

The detailed steps of the GCL algorithm are shown in the Algorithm 2.

Remark: When the client data distribution is i.i.d., the categories distribution is essentially the same between different clients. It causes the GCL algorithm to tend to randomly select clients for grouping. Therefore, our algorithm can be mainly applied to non-i.i.d. data distribution.

V. Experimental Evaluation

In this section, we comprehensively evaluate our proposed system's performance in identifying travel mode using a real-world dataset. We first investigate the performance of our proposed system and compare it with other approaches. Second, its performance under a non-i.i.d. data distribution and evaluate the effectiveness of the proposed GCL algorithm and data augmentation method. Third, we analyze the impact of other parameters, which will be introduced in Sec. V-A.

TABLE I
COMPARISON OF TRAVEL MODE IDENTIFICATION APPROACHES

Approach	Accuracy (%)				Privacy-preserving
	$\alpha = 5\%$	$\alpha = 20\%$	$\alpha = 50\%$	Supervised	
SECA [32]	56.1	69.3	73.2	76.8	No
Semi-Two-step [32]	51.4	56.2	58.8	60.5	No
Semi-pseudo-label [32]	53.6	66.3	70.7	75.4	No
CNN [16]	61.1	74.9	83.6	84.3	No
Ensemble-LSTM [31]	**87.7**	**90.0**	90.8	91.5	No
Proposed	84.3	89.2	**91.3**	**92.2**	**Yes**

Finally, we will analyze the running efficiency and communication cost of the proposed system.

A. Dataset and Evaluation Setup

In the subsequent case studies, we adopt raw GPS data from the GeoLife project [14] for investigation, including the travel trajectories of 182 users over 5 years. 69 users recorded their travel modes in this dataset. We use their labeled information as ground-truth facts and to regulate them by the methods mentioned in Sec. IV-A. Following the dataset authors' recommendation, we select 5 main transportation modes for identification, i.e., walk, bus, bike, drive, and train. After preprocessing all the GPS trajectories, we get a total of 24, 743 samples. For the convenience of partitioning, we use 2743 samples from the total samples as the test set D_t, and the remaining samples are split into server dataset D_s and client dataset $D_{\mathcal{K}}$ according to the proportion of labeled samples. This data partitioning method is widely used in semi-supervised learning [30], [31], which enables better evaluation of the model with various proportions of labeled data.

We set the default settings for some common parameters before starting the simulation experiments as follows:

- Client number K: We set $K = 20$ by default, which means our unlabeled data are distributed equally to 20 clients.
- Percentage of labeled data α: α indicates the proportion of the labeled data in the server. The default alpha setting is 50%.
- Communication Round T: T denotes the number of training steps of the local model during two consecutive communications. The default setting of T is 5.
- Non-i.i.d. level R: We assume that each client contains at most two travel modes to simulate the distribution of non-i.i.d. data in the real world as much as possible, which means $R = 0.5$.
- The number of groups c: In each communication round, we use the GCL algorithm to divide the clients who participate in the communication into c groups. Considering that a large c renders increasing communication load, in our experiments, we set $c = 3$.

In each training round, we randomly select 50% of all clients to participate in the communication, and each client model adopts the Adam [52] optimizer on its dataset with local batch sizes $B = 30$, which performs a parameter aggregation with the server after 5 rounds of local training. All case studies are developed by Python and Pytorch [53], and all simulation experiments were performed on a computing server with an Nvidia GeForce RTX2080 Ti GPU and Intel(R) Xeon(R) Silver CPU.

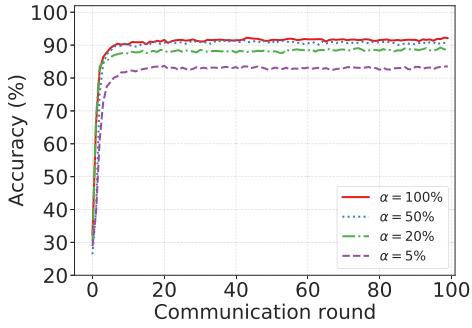


Fig. 4. The system convergence behaviors under different amounts of labeled data.

B. System Performance

In this section, we first evaluate the proposed system's effectiveness by comparing it with different percentages of labeled data (i.e., $\alpha = 5\%, 20\%, , 50\%, 100\%$). As shown in Table I, the proposed semi-supervised learning system only requires 5% labeled data for training can achieve 84.3% identification accuracy. When using 50% of all labeled data, the accuracy exceeded 91%, which is only 1% less than when training with all labeled data. Besides, as we can see from Fig. 4, there is a slight decrease in performance when using fewer labeled samples in training, but our model still converges quickly. Such results show that our scheme can converge and achieve better accuracy with a few labeled data. Furthermore, there is a trade-off between the privacy protection of our system and the local computational overhead. Using less labeled data (small α) means that our system discloses fewer data and can better protect user privacy, but correspondingly leads to more local computations and accuracy loss. For example, less labeled data may cause the client to increase the time for semi-supervised computing operations during local training. Notably, the privacy preservation for the system does not affect the communication overhead since only model parameters are passed between clients and the server.

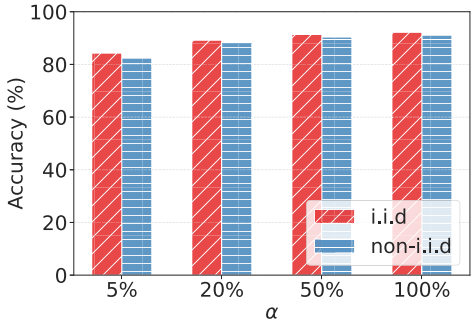


Fig. 5. System performance between i.i.d. and non-i.i.d. distribution.

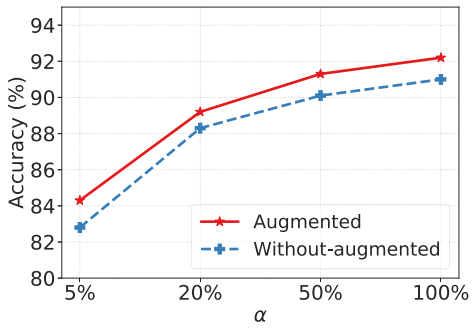


Fig. 6. Accuracy comparison with or without using data augmentation methods.

As one may concern that our model does not apply in the scenarios of non-i.i.d. data distributions, we consider the performance of the model under the distribution of non-i.i.d. data. The simulation results are shown in Fig. 5. We can find that although the performance of our model is degraded under the non-i.i.d. data distribution, it can still achieve a satisfactory identification accuracy.

Moreover, with data non-i.i.d. distributed and only 5% of all data labeled, our model still achieves over 82% accuracy. And the maximum difference in the accuracy of our model in different data distribution is not more than 2%, which fully demonstrates that the proposed system is also robust to the non-i.i.d. data distribution.

C. Model Performance Comparison

In this section, we investigate the performance of our proposed system with other approaches presented in the literature. To be specific, we conduct a series of experiments and compare their identification accuracy at various labeling rates, including semi-two-steps [31], semi-pseudo-label [31], semi-supervised convolutional autoencoder (SECA) [31], convolution neural network (CNN) [15], and semi-ensemble methods (Ensemble-LSTM) [30]. All these methods are applied to GeoLife dataset. It is worth noting that the proposed system achieves satisfactory accuracy with full privacy protection, while the rest do not have this property.

The results of the comparison are shown in Table I. The best performance is displayed in bold. We can find that our model outperforms all other methods being compared when using 50% of the total amount of labeled data from the experimental results. However, when the amount of label data used is too small, our model slightly underperforms the Ensemble-LSTM method. The reason is that the pseudo-labeling of the data causes some accumulation error, which affects the performance of the model. In addition, the model also has some drawbacks that make it not suitable for resource-constrained mobile scenes and FL systems: The first is Ensemble-LSTM uses a scheme that simultaneously trains multiple DNN models with stack LSTM layers [30], considering the client does not have the same computing power as a server, which is not applicable in an FL environment. Second, Ensemble-LSTM uses multiple DNN models, the smallest of which is approximately 8.8**Mb**, while the model we proposed is only 4.9**Mb**, which greatly increases the communication cost if we use Ensemble-LSTM. Most importantly, Ensemble-LSTM needs to collect users' data in a centralized server, leading to user privacy leakage. Therefore, our approach is effective and feasible both in terms of privacy protection and training efficiency of the model.

D. Performance of the Data Augmentation and Group Aggregation Operation

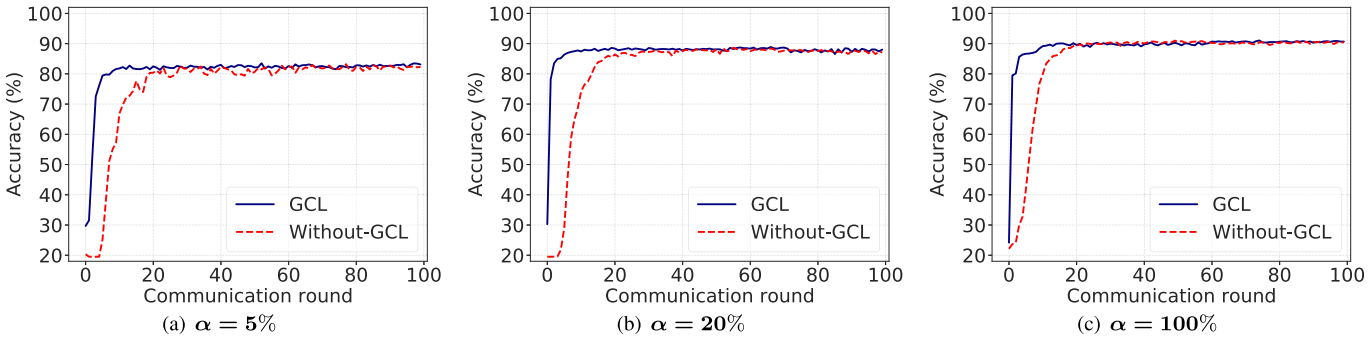
In this section, we evaluate the impact of the data augmentation methods and group aggregation algorithms (GCL) on system performance.

1) Data Augmentation Performance: Because the quantity of labeled data is too limited in semi-supervised learning, we use the data augmentation method proposed in Sec. IV-A.2 to increase the amount of labeled data. It should be noted that our augmentation method only applies to the labeled data (i.e., server data). From the simulation result presented in Fig. 6, it is quite clear that the method is effective in improving the accuracy of them. The reason is that our data augmentation method flips data through time series to maximize the simulation of real-world motion characteristics of different modes. These augmented data show the same motion characteristics (minimum and maximum values of speed) as the original data, albeit with different temporal dynamics. And the extra data can also be beneficial for model training.

2) Grouping Mechanism Performance: To verify the proposed GCL algorithm's effectiveness, we conduct a series of experiments with different amounts of labeled data. This simulation only considers the case of non-i.i.d. data distribution since the GCL algorithm degenerates into a random grouping algorithm under i.i.d. data distribution. We set $\alpha = 0.05/0.2/1, R = 0.5, C = 3$, respectively in non-i.i.d. data distribution, and all other parameters set to default. The performance are shown in Fig. 7. It can be seen that using the GCL algorithm can improve the convergence speed of the model. This is explained by the fact that we employed a group-clustering aggregation method based on data distributions, which is able to capture the main gradient features in different data distributions, thus minimizing the influence of the non-i.i.d. distribution on the global model.

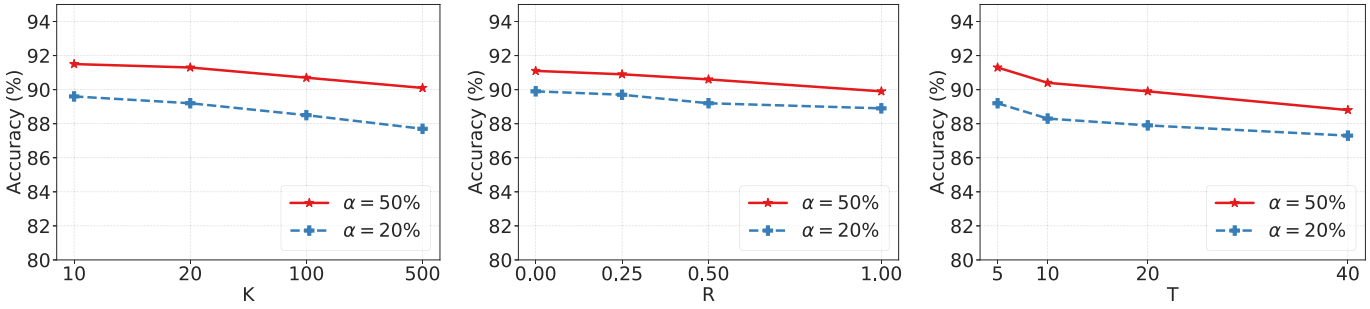
E. Parameter Sensitivity Test

Lastly, we perform a parameter sensitivity test to evaluate the impact of parameter selection in SSFL. We first assess the



(a) $\alpha = 5\%$ (b) $\alpha = 20\%$ (c) $\alpha = 100\%$

Fig. 7. Accuracy comparison with or without GCL algorithm.



(a) **Different number of clients.** (b) **Different non-i.i.d. level.** (c) **Different communication period.**

Fig. 8. Accuracy comparison under different environments.

impact of different numbers of users on system performance. We set $K = 10, 20, 100, 500$, and $\alpha = 0.2/0.5$ for simulation (Since the total amount of our data is too limited to distribute to 500 clients for deep learning modeling, we replicate the data by a multiple of 5 and then allocate it to 500 clients). As shown in Fig. 8(a), we can see that our system still works well for a large number of clients, but the increase in the number of users has a negative impact on the performance of the model. The reason is that the increase in the number of clients also increases the variability between models and makes model aggregation difficult. Furthermore, The grouping-based model averaging methods can address the problem of server-client communication in the FL framework since it is not guaranteed that every client participating in FL always has a network connection to the server.

Subsequently, we conduct research on different non-i.i.d. levels. We set $K = 20, C = 3, \alpha = 0.2/0.5$, and $R = 0/0.25/0.5/1$, respectively, and all other parameters set to default. When $R = 0 (R = 1)$, the clients have a uniform class distribution (single class data). The simulation results are presented in Fig. 8(b). We can discover that no significant degradation is observed in the model performance as non-i.i.d. level increases, which can be attributed to the extraction capability of our model for high-dimensional data features and the adaptability of the proposed GCL algorithm. And the experimental result also indicate the robustness of our system to the distribution of non-i.i.d. data.

Finally, we consider the fact that the communication window between client and server may vary in different cases. Therefore, we also evaluate the performance of the system under different communication window length. As can be seen in Fig. 8(c), an increase in T implies a decrease in the communication frequency, which leads to worse performance of the model. This is because communicating with the server for a long period can lead to the local model's overfitting issue. Therefore, maintaining a continuous communication period is essential for model performance.

F. Discussion

Due to computational resource constraints, we do not have enough resources to simulate distributed training, so we use a single-core CPU to perform serial training on a server with an Intel(R) Xeon(R) Silver 4210 CPU and an Nvidia 2080Ti GPU. By default, there are 20 clients in total, and 10 clients are selected for training in each round. The total time to complete 100 training rounds is about 3060 seconds (ignoring the parameter communicate time). It means that each round in the simulation takes about 30.6 ($3060/100 = 30.6$) seconds, and each client takes about 3 ($3060/100/10 = 3.06$) seconds for training a round. In other words, the training time for 100 rounds is about 5 minutes (306 seconds) for each client. Although the users' device may not have such computing power as the server and considering the communication time between the server and the client, we can still optimistically estimate that the whole experiment can be completed within a few hours. In most cases, the FL training is conducted when the device is unused (e.g., at night after the user has slept). The whole process does not interfere with users' daily routine and is highly feasible in real-life situations.

In addition, we also examine the testing time of the system. For all 2743 trajectories, our system can infer all travel modes within 0.2 seconds. Although the simulation test time may slightly differ from the real environment considering different devices, such efficient inference capability fully illustrates the feasibility of our system in practical applications.

VI. Conclusion

This paper proposes a semi-supervised federated learning (SSFL) empowered travel mode identification system that can use a small privacy-permitted labeled data on the server to accurately infer travel modes without compromising user privacy. In the proposed system, we design a novel DNN architecture that integrates CNN and GRU's superior performance to capture the spatio-temporal high-dimensional features of GPS trajectories data in a fine-grained manner. To enhance this identification model's performance, we introduce a data augmentation method that uses the method of flipping time-series data to increase the amount of label data held by the servers. In particular, we set a threshold to determine the pseudo-labels' confidence and feed these pseudo-label data to the proposed system. For the problem of non-i.i.d. distribution of data collected in travel mode identification applications, we propose a group-clustering aggregation algorithm to the group and aggregate the heterogeneous data according to the client class distribution, thereby achieving robust aggregation to non-i.i.d. data.

To evaluate the performance of our proposed system, we conduct extensive experiments on the Geolife dataset. We first evaluate the performance of our system under different volumes of available information. Compared to existing methods, our system can be trained with 50% of the labeled data to achieve 91.3% identification accuracy while protecting privacy, which is impossible with all other methods. Subsequently, we evaluate our system's performance in different environments and parameter settings to demonstrate that it is also effective in different scenarios (i.e., massive clients, non-i.i.d. distribution, and long communication period). We consider that this work provides a promising way to use small amounts of data for travel mode identification without infringing on user privacy.

References

[1] R. Jia *et al.*, "Efficient task-specific data valuation for nearest neighbor algorithms," *Proc. VLDB Endowment*, vol. 12, no. 11, pp. 1610–1623, Jul. 2019.

[2] F.-Y. Wang, "Parallel control and management for intelligent transportation systems: Concepts, architectures, and applications," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 3, pp. 630–638, Sep. 2010.

[3] J. Zhang, F.-Y. Wang, K. Wang, W.-H. Lin, X. Xu, and C. Chen, "Data-driven intelligent transportation systems: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 1624–1639, Dec. 2011.

[4] J. J. Q. Yu and J. Gu, "Real-time traffic speed estimation with graph convolutional generative autoencoder," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 10, pp. 3940–3951, Oct. 2019.

[5] B. Wang, L. Gao, and Z. Juan, "Travel mode detection using GPS data and socioeconomic attributes based on a random forest classifier," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 5, pp. 1547–1558, May 2018.

[6] M. A. Mondal and Z. Rehena, "Intelligent traffic congestion classification system using artificial neural network," in *Proc. Companion World Wide Web Conf.*, New York, NY, USA: ACM, May 2019, pp. 110–116.

[7] E. Anagnostopoulou, B. Magoutas, E. Bothos, and G. Mentzas, "Persuasive technologies for sustainable smart cities: The case of urban mobility," in *Proc. Companion World Wide Web Conf.* New York, NY, USA: ACM, May 2019, pp. 73–82.

[8] G. Li *et al.*, "Public transportation mode detection from cellular data," in *Proc. ACM Conf. Inf. Knowl. Manage.*, Nov. 2017, pp. 2499–2502.

[9] X. Su, H. Caceres, H. Tong, and Q. He, "Online travel mode identification using smartphones with battery saving considerations," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 10, pp. 2921–2934, Oct. 2016.

[10] B. Assemi, H. Safi, M. Mesbah, and L. Ferreira, "Developing and validating a statistical model for travel mode identification on smartphones," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 7, pp. 1920–1931, Jul. 2016.

[11] L. Wu, B. Yang, and P. Jing, "Travel mode detection based on GPS raw data collected by smartphones: A systematic review of the existing methodologies," *Information*, vol. 7, no. 4, pp. 67–86, 2016.

[12] A. C. Prelipcean, G. Gidófalvi, and Y. O. Susilo, "Transportation mode detection—An in-depth review of applicability and reliability," *Transp. Rev.*, vol. 37, no. 4, pp. 442–464, Jul. 2017.

[13] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W.-Y. Ma, *Understanding Mobility Based on GPS Data*. New York, NY, USA: ACM, 2008, pp. 312–321.

[14] Y. Zheng, L. Liu, L. Wang, and X. Xie, "Learning transportation mode from raw gps data for geographic applications on the Web," in *Proc. 17th Int. Conf. World Wide Web (WWW)*. New York, NY, USA: ACM, 2008, pp. 247–256.

[15] S. Dabiri and K. Heaslip, "Inferring transportation modes from GPS trajectories using a convolutional neural network," *Transp. Res. C, Emerg. Technol.*, vol. 86, pp. 360–371, Jan. 2018.

[16] J. J. Q. Yu, "Travel mode identification with GPS trajectories using wavelet transform and deep learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 2, pp. 1093–1103, Feb. 2021.

[17] P. A. Gonzalez *et al.*, "Automating mode detection for travel behaviour analysis by using global positioning systems-enabled mobile phones and neural networks," *IET Intell. Transp. Syst.*, vol. 4, no. 1, pp. 37–49, 2010.

[18] R. Jia *et al.*, "Towards efficient data valuation based on the shapley value," in *Proc. 22nd Int. Conf. Artif. Intell. Statist.*, 2019, pp. 1167–1176.

[19] Y. Endo, H. Toda, K. Nishida, and A. Kawanobe, "Deep feature extraction from trajectories for transportation mode estimation," in *Proc. Pacific–Asia Conf. Knowl. Discovery Data Mining*. Cham, Switzerland: Springer, 2016, pp. 54–66.

[20] W. Jeong, J. Yoon, E. Yang, and S. J. Hwang, "Federated semi-supervised learning with inter-client consistency & disjoint learning," 2020, *arXiv:2006.12097*. [Online]. Available: http://arxiv.org/abs/2006.12097

[21] A. Albaseer, B. S. Ciftler, M. Abdallah, and A. Al-Fuqaha, "Exploiting unlabeled data in smart cities using federated edge learning," in *Proc. Int. Wireless Commun. Mobile Comput. (IWCMC)*. Byblos, Lebanon: IEEE, Jun. 2020, pp. 1666–1671.

[22] C. of European Union. (2016). *Regulation (EU) 2016/679 of the European Parliament and of the Council on the Protection of Natural Persons With Regard to the Processing of Personal Data and on the Free Movement of Such Data, and Repealing Directive 95/46/EC (General Data Protection Regulation)*. European Commission. [Online]. Available: https://gdpr-info.eu/

[23] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 1–19, 2019.

[24] W. Y. B. Lim *et al.*, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 2031–2063, 3rd Quart., 2020.

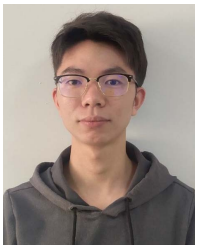
[25] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, May 2020.

[26] Y. Liu, X. Yuan, R. Zhao, Y. Zheng, and Y. Zheng, "RC-SSFL: Towards robust and communication-efficient semi-supervised federated learning system," 2020, *arXiv:2012.04432*. [Online]. Available: http://arxiv.org/abs/2012.04432

[27] F. Sattler, S. Wiedemann, K.-R. Muller, and W. Samek, "Robust and communication-efficient federated learning from non-i.I.d. Data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 9, pp. 3400–3413, Sep. 2020.

[28] A. Ghosh, J. Hong, D. Yin, and K. Ramchandran, "Robust federated learning in a heterogeneous environment," 2019, *arXiv:1906.06629*. [Online]. Available: http://arxiv.org/abs/1906.06629

[29] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*. 2017, pp. 1273–1282.
[30] J. J. Q. Yu, "Semi-supervised deep ensemble learning for travel mode identification," *Transp. Res. C, Emerg. Technol.*, vol. 112, pp. 120–135, Mar. 2020.
[31] S. Dabiri, C.-T. Lu, K. Heaslip, and C. K. Reddy, "Semi-supervised deep learning approach for transportation mode identification using GPS trajectory data," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 5, pp. 1010–1023, May 2020.
[32] T. H. Vu, L. Dung, and J.-C. Wang, "Transportation mode detection on mobile devices using recurrent nets," in *Proc. 24th ACM Int. Conf. Multimedia*. New York, NY, USA: ACM, Oct. 2016, pp. 392–396.
[33] J. V. Jeyakumar, E. S. Lee, Z. Xia, S. S. Sandha, N. Tausik, and M. Srivastava, "Deep convolutional bidirectional LSTM based transportation mode recognition," in *Proc. ACM Int. Joint Conf. Int. Symp. Pervas. Ubiquitous Comput. Wearable Comput.*, Oct. 2018, pp. 1606–1615.
[34] H. Liu and I. Lee, "End-to-end trajectory transportation mode classification using Bi-LSTM recurrent neural network," in *Proc. 12th Int. Conf. Intell. Syst. Knowl. Eng. (ISKE)*, Nanjing, China, Nov. 2017, pp. 1–5.
[35] M. Conti, L. V. Mancini, R. Spolaor, and N. V. Verde, "Analyzing Android encrypted network traffic to identify user actions," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 1, pp. 114–125, Jan. 2016.
[36] M. Liu, L. Cheng, Y. Gu, Y. Wang, Q. Liu, and N. E. O'Connor, "MPC-CSAS: Multi-party computation for real-time privacy-preserving speed advisory systems," *IEEE Trans. Intell. Transp. Syst.*, early access, Jan. 28, 2021, doi: 10.1109/TITS.2021.3052840.
[37] Y. Zhu, S. Zhang, Y. Liu, D. Niyato, and J. J. Q. Yu, "Robust federated learning approach for travel mode identification from non-IID GPS trajectories," in *Proc. IEEE 26th Int. Conf. Parallel Distrib. Syst. (ICPADS)*, Dec. 2020, pp. 585–592.
[38] Y. Liu, J. J. Q. Yu, J. Kang, D. Niyato, and S. Zhang, "Privacy-preserving traffic flow prediction: A federated learning approach," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7751–7763, Aug. 2020.
[39] J. J. Q. Yu, "Sybil attack identification for crowdsourced navigation: A self-supervised deep learning approach," *IEEE Trans. Intell. Transp. Syst.*, early access, Nov. 17, 2020, doi: 10.1109/TITS.2020.3036085.
[40] T. Vincenty, "Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations," *Surv. Rev.*, vol. 23, no. 176, pp. 88–93, Apr. 1975.
[41] H. Wang, G. Liu, J. Duan, and L. Zhang, "Detecting transportation modes using deep neural network," *IEICE Trans. Inf. Syst.*, vol. E100.D, no. 5, pp. 1132–1135, 2017.
[42] R. Annoni and C. H. Q. Forster, "Experimental comparison of DWT and DFT for trajectory representation," in *Proc. 13th Int. Conf. Intell. Data Eng. Automated Learn. (IDEAL)*. Berlin, Germany: Springer-Verlag, 2012, pp. 670–677.
[43] S. G. Mallat, "A theory for multiresolution signal decomposition: The wavelet representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 7, pp. 674–693, Jul. 1989.
[44] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015.
[45] K. Cho *et al.*, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," 2014, *arXiv:1406.1078*. [Online]. Available: http://arxiv.org/abs/1406.1078
[46] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, *arXiv:1412.3555*. [Online]. Available: http://arxiv.org/abs/1412.3555
[47] Z. Zhang, Y. Yang, Z. Yao, Y. Yan, J. E. Gonzalez, and M. W. Mahoney, "Improving semi-supervised federated learning by reducing the gradient diversity of models," 2020, *arXiv:2008.11364*. [Online]. Available: http://arxiv.org/abs/2008.11364
[48] W. Chen, K. Bhardwaj, and R. Marculescu, "FedMAX: Mitigating activation divergence for accurate and communication-efficient federated learning," 2020, *arXiv:2004.03657*. [Online]. Available: http://arxiv.org/abs/2004.03657
[49] X. Zhang, M. Hong, S. Dhople, W. Yin, and Y. Liu, "FedPD: A federated learning framework with optimal rates and adaptivity to non-IID data," 2020, *arXiv:2005.11418*. [Online]. Available: http://arxiv.org/abs/2005.11418
[50] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," 2018, *arXiv:1812.06127*. [Online]. Available: http://arxiv.org/abs/1812.06127
[51] S. Wang *et al.*, "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1205–1221, Jun. 2019.
[52] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, San Diego, CA, USA, May 2015, pp. 1–15.
[53] A. Paszke *et al.*, "Automatic differentiation in PyTorch," in *Proc. Adv. Neural Inf. Process. Syst.*, Long Beach, CA, USA, 2017, pp. 1–4.



Yuanshao Zhu (Graduate Student Member, IEEE) received the B.Eng. degree in telecommunication engineering from Shandong University, Weihai, China, in 2019. He is currently pursuing the master's degree with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China. His research interests include deep learning in smart city and edge computing, intelligent transportation systems, and federated learning.

Yi Liu (Student Member, IEEE) received the B.Eng. degree from Heilongjiang University, China, in 2019. His research interests include security and privacy in smart city and edge computing, deep learning, intelligent transportation systems, and federated learning.

James J. Q. Yu (Senior Member, IEEE) received the B.Eng. and Ph.D. degrees in electrical and electronic engineering from The University of Hong Kong, Hong Kong, in 2011 and 2015, respectively. From 2015 to 2018, he was a Post-Doctoral Fellow with The University of Hong Kong. He is currently an Assistant Professor with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China. He is an Honorary Assistant Professor with the Department of Electrical and Electronic Engineering, The University of Hong Kong. He is the Chief Research Consultant at GWGrid Inc., Zhuhai, and Fano Labs, Hong Kong. His research interests include smart city and urban computing, deep learning, intelligent transportation systems, and smart energy systems. He is an Editor of the *IET Smart Cities* journal.

Xingliang Yuan (Member, IEEE) received the B.S. degree in electrical engineering from the Nanjing University of Posts and Telecommunications in 2008, the M.S. degree in electrical engineering from the Illinois Institute of Technology in 2009, and the Ph.D. degree in computer science from the City University of Hong Kong in 2016. He is currently a Lecturer with the Faculty of Information Technology, Monash University, Australia. His research interests include cloud computing security, secure networked systems, and hardware security.