ST-SiameseNet: Spatio-Temporal Siamese Networks for Human Mobility Signature Identification

Huimin Ren[1,*], Menghai Pan[1,*], Yanhua Li[1], Xun Zhou[2], Jun Luo[3]
[1]Worcester Polytechnic Institute, [2]University of Iowa, [3]Lenovo Group Limited
[1]{hren,mpan,yli15}@wpi.edu, [2]xun-zhou@uiowa.edu, [3]jluo1@lenovo.com

ABSTRACT

Given the historical movement trajectories of a set of individual human agents (e.g., pedestrians, taxi drivers) and a set of new trajectories claimed to be generated by a specific agent, the Human Mobility Signature Identification (HuMID) problem aims at validating if the incoming trajectories were indeed generated by the claimed agent. This problem is important in many real-world applications such as driver verification in ride-sharing services, risk analysis for auto insurance companies, and criminal identification. Prior work on identifying human mobility behaviors requires additional data from other sources besides the trajectories, e.g., sensor readings in the vehicle for driving behavior identification. However, these data might not be universally available and is costly to obtain. To deal with this challenge, in this work, we make the first attempt to match identities of human agents only from the observed location trajectory data by proposing a novel and efficient framework named Spatio-temporal Siamese Networks (ST-SiameseNet). For each human agent, we extract a set of profile and online features from his/her trajectories. We train ST-SiameseNet to predict the mobility signature similarity between each pair of agents, where each agent is represented by his/her trajectories and the extracted features. Experimental results on a real-world taxi trajectory dataset show that our proposed ST-SiamesNet can achieve an F_1 score of 0.8508, which significantly outperforms the state-of-the-art techniques.

CCS CONCEPTS

• **Computing methodologies** → **Supervised learning by classification**; *Neural networks*; Anomaly detection.

KEYWORDS

Driver identification; Anomaly detection; Siamese networks; Few-shot learning

ACM Reference Format:
Huimin Ren[1,*], Menghai Pan[1,*], Yanhua Li[1], Xun Zhou[2], Jun Luo[3]. 2020. ST-SiameseNet: Spatio-Temporal Siamese Networks for Human Mobility Signature Identification. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '20), August 23–27, 2020, Virtual Event, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3394486.3403183

*Huimin Ren and Menghai Pan contributed equally to this work.

KDD '20, August 23–27, 2020, Virtual Event, CA, USA

ACM ISBN 978-1-4503-7998-4/20/08…$15.00
https://doi.org/10.1145/3394486.3403183

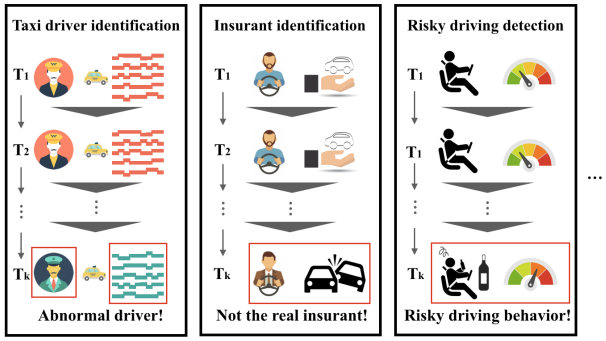


Figure 1: Applications of HuMID problem

1 INTRODUCTION

Given the historical movement trajectories of a set of individual human agents (e.g., pedestrians, taxi drivers) and a set of new trajectories claimed to be generated by a specific agent, the Human Mobility Signature Identification (HuMID) problem aims at validating if the incoming trajectory was indeed generated by the claimed agent. The HuMID problem has many real-world applications. Fig. 1 shows a few such examples. One of the major applications is automatic driver identification for taxi and rider-sharing services. According to the New York City Taxi and Limousine Commission (TLC) released statistics, there were on average 850, 000 trips taken by taxis and ride-sharing services per day in New York City in 2018 [2]. Meanwhile, the safety concerns have been raised by people recently. For example, some unauthorised drivers are reported to have taken the place of authorised drivers, and behave offensively towards passengers. Companies like Uber have taken actions to ensure the safety of passengers by enabling on-trip reporting from the APP [24, 30]. HuMID can help identify the above illegal driver substitutions as early as possible and help improve the safety of the passengers. Another example is insurer identification in the auto insurance industry. Insurance companies need to make sure that a vehicle was driven by the insured driver rather than others when the insurer filed a claim. All of these examples are downstream applications of and can benefit from solving HuMID problems.

Many prior works pay attention to the driving behavior identification problem, an instance of the HuMID problem. Hallac et al. [10] identified driver using automobile sensor data from a single turn. They monitored 12 sensors installed inside and outside the vehicle and implemented a hand-crafted rule-based classifier, which classifies up to 5 drivers. Chowdhury et al. [6] extracted 137 statistical features from smartphone sensors and used a random forest classifier to classify trajectories into small groups of 4 to 5 drivers. Kieu et al. [17] presented a multi-task learning model which

captured geographic features and driving behavior features of trajectories in 3D images as input to perform trajectory clustering and driver identification. Oh and Iyengar [25] used inverse reinforcement learning in sequential anomaly detection problem. They estimated reward function for each driver and evaluated 10 individuals from GeoLife-GPS dataset and aggregated normal behaviors of taxi drivers from Taxi Service Trajectory dataset.

Nonetheless, there exist significant limitations when implementing these methods in real-world applications. First, some of these works require additional data rather than the GPS records by installing sensors on the vehicles, for example, 12 sensors in Hallac et al.'s work [10]. However, few vehicles is equipped with these additional sensors, and it will be costly to install sensors to the vehicles. Second, most existing works can only deal with a small group of drivers because they employ classification or clustering approaches. For example, Chowdhury et al. [6] employed random forest to classify the trajectories of 4 to 5 drivers, and Oh et al. [25] estimated 10 reward functions for 10 drivers by using inverse reinforcement learning. In real-world cases, the pool of drivers is large. Thus, these methods are hard to be implemented. Third, a group of previous works require that all the drivers be known in advance [6, 10, 25]. Those methods are unsuitable for applications where only a subset of the drivers is known at training.

To address these limitations, in this paper we propose a Spatio-temporal Siamese networks (ST-SiameseNet) framework to identify the behavior of a large group of human agents (e.g., drivers) by using only their movement trajectory data. Since GPS devices are widely equipped on vehicles and smart phones nowadays, the data ST-SiameseNet requires can be easily collected. Also, ST-SiameseNet can deal with large groups of human agents in a single model and be used on new agents who are previously-unseen from training pool. To be more specific, we first extract different transit modes of the agents from the trajectory data. For example, there are **two** transit modes in taxi driving, i.e., the seeking trajectory where the vehicle has no passengers on-board, and the driving trajectory where the vehicle has passengers on-board. Besides, we extract different profile features and online features from the historical trajectories of each agent to augment the performance of ST-SiameseNet. Then, we input the trajectories together with the profile features to ST-SiameseNet pair-wisely and train the ST-SiameseNet to identify the similarity of each pair of inputs. Experiments on a real-world taxi trajectory dataset show that ST-SiameseNet outperforms all baselines in identification performances. *Our main contributions* are summarized as follows:

- We formulate the Human Mobility Signature Identification (HuMID) problem as a predictive analysis problem and, for the first time, employ the idea of the Siamese network to identify agents by their "mobility signatures" from solely their trajectory data.
- We design a novel ST-SiameseNet framework that can handle multimodal trajectory data. We also utilize both profile features and online features extracted from the agents' trajectory data to train ST-SiameseNet.
- We conduct substantial experiments using a real-world taxi trajectory dataset to evaluate the performance of our proposed ST-SiameseNet.

The remainder of the paper is organized as follows. Section 2 presents an overview of the key ideas of our research problem. Section 3 provides detailed methodology of our proposed model. We discuss the experimental results on different datasets and the related work in sections 4 and 5. Section 6 concludes the paper.

2 OVERVIEW

2.1 Problem Definition

In this section, we introduce some important definitions and formally define the problem.

Definition 2.1. Human-generated spatio-temporal trajectory tr. With the wide use of GPS sets, people can generate massive spatio-temporal data while they are using the devices equipped with GPS sets, e.g., the GPS records of vehicles, smartphones, smart watches, etc. Each GPS point p consists of a location in latitude lat and longitude lng, and a time stamp t, i.e. $p = \langle lat, lng, t \rangle$. A trajectory tr is a sequence of GPS points with a label of the agent a who generated the data, denoted as $tr = \{a, \langle p_1, p_2, ..., p_n \rangle\}$, where the set of trajectories is $\mathcal{T}$.

Definition 2.2. Transit mode. Transit modes are defined as a set of categories of trajectories, where each category is generated under a different mobility pattern. For example, taxi driving trajectories can be categorized into two modes, i.e., with and without any passenger on-board. Private car trajectories can be grouped into commute and recreational driving trajectories, etc. In this paper, we use taxi driving as the application. The seeking trajectory $\mathcal{T}_s$ is the sequence of GPS records while the vehicle is without any passengers on-board, and the driver is seeking for passengers to serve. The driving trajectory $\mathcal{T}_d$ is the sequence of GPS records while the vehicle is with passengers on-board, and the driver is taking the passengers to the destination.

Definition 2.3. Profile feature $\mathbf{f}_p$. Each agent has unique personal (or profile) characteristics which can be extracted from his/her trajectory data, such as frequent start/end locations, average trip time duration, and preferred geographic area, working as different dimensions $f_{p,i}$ of the profile features, where i is the i-th dimension of these features. The profile features of each agent can be extracted in different time period. Here we denote time period as T, where T can be one hour, one day or one week, etc.

Definition 2.4. Online feature $\mathbf{f}_o$. Online features represent agents' mobility patterns resulting from the agent's personal judgment, experience and skills, such as speed, acceleration, turning left, turning right of each grid cell, working as different dimensions $f_{o,i}$ of the online features, where i is the i-th dimension of these features. For each trajectory, we build the online features.

Problem definition. Given a set of historical trajectories $\mathcal{T}$ collected from a group of agents A in time periods $T_0, T_1, ..., T_t$, we aim to develop a framework to verify if the incoming trajectories $\mathcal{T}^{t+1}$ which are claimed being collected from an agent a's vehicle in T_{t+1} are indeed matching the agent a's behavior.

2.2 Data Description

The purpose of our framework is to recognize human mobility signatures with GPS records. In this paper, we **use taxi driving**

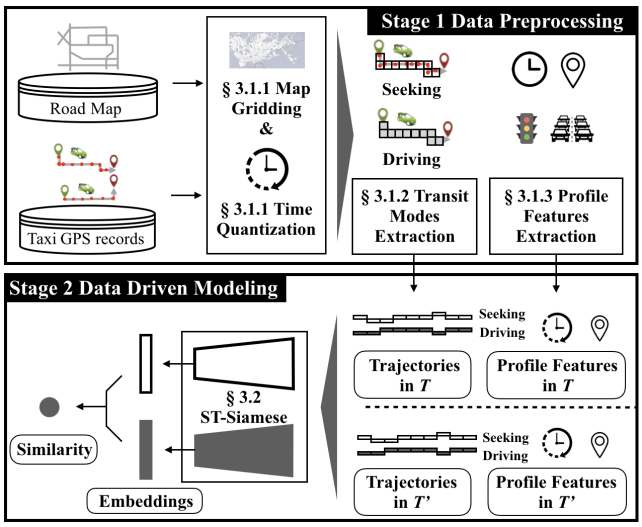


Figure 2: Solution framework

scenario as an example to demonstrate our techniques. However, the proposed solution can be easily generalized to other types of agents and trajectories. Our analytical framework takes two urban data sources as input, including (1) taxi trajectory data and (2) road map data. For consistency, both datasets are collected in Shenzhen, China in July 2016.

Taxi trajectory data contains GPS records collected from taxis in Shenzhen, China during July 2016. There were in total 17,877 taxis equipped with GPS sets, where each GPS set generates a GPS point every 40 seconds on average. Overall, a total of 51,485,760 GPS records are collected on each day, and each record contains five key data fields, including taxi ID, time stamp, passenger indicator, latitude and longitude. The passenger indicator field is a binary value, indicating if a passenger is aboard or not.

Road map data of Shenzhen covers the area defined between 22.44° to 22.87° in latitude and 113.75° to 114.63° in longitude. The data is from OpenStreetMap [1] and has 21,000 roads of six levels.

2.3 Solution Framework

Our proposed solution framework is outlined in Fig. 2, which takes two sources of urban data as inputs and contains two stages: (1) extracting trajectories and profile features in section 3.1, (2) identifying driving behavior in section 3.2.

3 METHODOLOGY

3.1 Data Preprocessing

In this stage, we employ the GPS trajectory data and the road map data to extract the seeking and driving trajectories and online features, together with the profile features of each human agent in each time period.

3.1.1 Map Gridding and Time Quantization. We use a standard quantization trick to reduce the size of the location space. Specifically, we divide the study area into equally-sized grid cells with a given side-length s in latitude and longitude. Our method has two advantages: (i) we have the flexibility to adjust the side-length to achieve different granularity, and (ii) it is easy to implement and highly scalable in practice [18, 19, 26]. Fig. 17b shows the actual grid in Shenzhen, China with a side-length $l = 0.01°$ in latitude and longitude. Eliminating cells in the ocean, those unreachable from the city, and other irrelevant cells gives a total of 1,934 valid cells. We denote each grid cell as g_i, with $1 \le g_i \le 1,934$, and the complete grid cell set as $\mathcal{G} = \{g_i\}$. We divide each day into five-minute intervals for a total of 288 intervals per day, denoted as $\mathcal{I} = \{\tilde{t}_j\}$, with $1 \le j \le 288$. A spatio-temporal region r is a pair of a grid cell s and a time interval $\tilde{t}$. Each GPS record $p = \langle lat, lng, t \rangle$ and be represented as an aggregated state $s = \langle g, \tilde{t} \rangle$, where the location $(lat, lng) \in g$, the time stamp $t \in \tilde{t}$. A trajectory of agent a then can be mapped to sequences of spatio-temporal regions, $tr = \{a, \langle s_1, s_2, ..., s_n \rangle\}$.

3.1.2 Transit Modes Extraction. Different transit modes can show different patterns of driving behavior. In the taxi driving scenario, seeking and driving trajectories reflect different characteristics for each human agent taxi driver. Thus, we split the trajectories into seeking $\mathcal{T}_s$ and driving trajectories $\mathcal{T}_d$ based on the status of the vehicle whether there are passengers on board. Fig. 3b and 3a illustrate the distribution of the number of driving trajectories and the length of each driving trajectory for each agent in each day, respectively. Here, $T = 1$ day. The distributions suggest that most agents have 20 seeking trajectories every day, and the average length of each seeking trajectory is around 14.03 km. The ratio between driving and seeking trips per day is approximately 1:1.

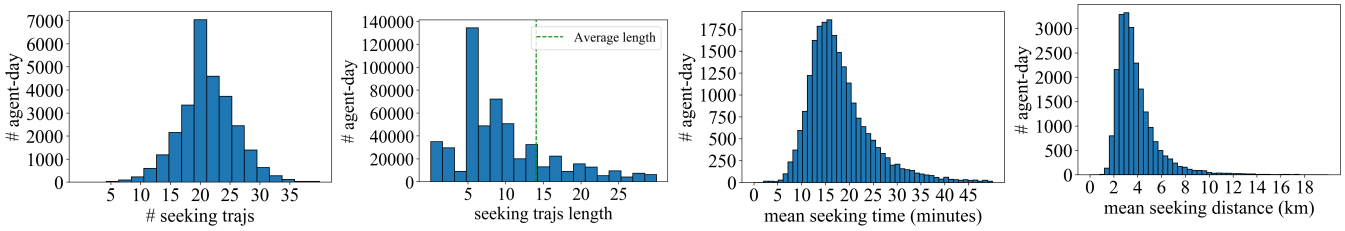
3.1.3 Features Extraction. Each agent has unique personal (or profile) characteristics, such as the location with the longest stay (possibly home location), daily working schedule (time duration), preferred geographic area, etc. These characteristics can be the statistical values extracted from their trajectory data. In this work, to augment the performance of driving behavior identification, we extracted the following 11 profile features for each agent in each time period of analysis. Moreover, we extract one online feature, i.e., speed, over time for each trajectory.

$f_{p,1}$ & $f_{p,2}$*: The coordinates (in longitude and latitude direction) of the longest-staying grid.* Each agent can have his/her own preference on **where** to take a break during work, thus we extract $f_{p,1}$ & $f_{p,2}$*: longest-staying grid* to represent the place where an agent takes a break. The longest staying grid is the grid where the GPS records remain unchanged for the longest time.

$f_{p,3}$ & $f_{p,4}$*: Break start & end time.* Similarly, each agent can have his/her own preference on **when** to take a break during work, thus we extract $f_{p,3}$ & $f_{p,4}$*: Break start & end time* to capture the schedule when an agent takes a break.

$f_{p,5}$ & $f_{p,6}$*: The coordinates of the most frequently visited grid.* Each agent has his/her own favorite region to go, which can help identify the agent. Thus, we extract $f_{p,5}$ & $f_{p,6}$*: most frequently visited grid* to capture the region that an agent visits the most frequently in T.

$f_{p,7}$ & $f_{p,8}$*: Average seeking trip distance & time.* Each agent has his/her own efficiency on looking for passengers. The experienced agents can find passengers quickly after serving a trip, while the new agents may take longer time and distance to find a new passenger. Thus, we extract $f_{p,7}$ & $f_{p,8}$*: Average seeking trip time & distance* to capture their efficiency on finding new passengers. The distribution of these two features are shown in Fig. 3c and 3d, respectively,



(a) No. of seeking trajectories (b) Length of seeking trajectories (c) Mean seek time (d) Mean seek distance

Figure 3: Profile features analysis

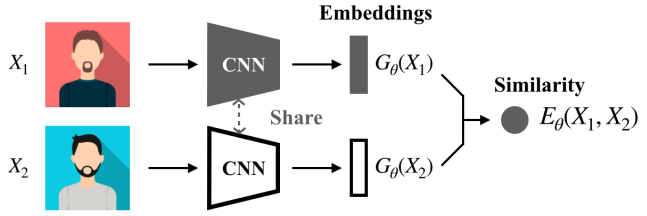


Figure 4: Siamese Network

where the x-axis the average seeking distance (in *km*) and the average seeking time (in min) for an agent in a day, and the y-axis is the number of driver-day's. Here $T = 1$ day. From the figures, we can see that averagely agents spend 15 minutes to seek passengers within 3 *km* from his/her current location.

$f_{p,9}$ & $f_{p,10}$: *Average driving trip time & distance.* Each of the agent can have his/her own preference on the length of trips that he/she serves. For example, some agents prefer to serve long trips, because they think they can earn much more at a time, and they may look for passengers near the airport or train station where the passengers have higher probabilities of asking for long trips. Some other agents prefer to look for short trips because they think they can earn money more efficiently by serving short trips. Thus, we calculate the average driving trip time and distance to capture each agent's unique preference on the length of driving trips.

$f_{p,11}$: *Number of trips served.* Each agent has his/ her own strategy on looking for passengers. The experienced agents may serve more trips in T than the new agents. Thus, we count the number of trips served of each agent in T to capture each agent's level of experience. This feature is just the number of driving trajectories in T.

$f_{o,1}$: *Speed.* Given a trajectory $tr = \{a, \langle(g_1, \tilde{t}_1)...(g_n, \tilde{t}_n)\rangle\}$, we also extract an online feature by calculating the speed information, denoted as v, for each data point in each trajectory to extract more information about driving behavior. The updated trajectory would be $\tau = \langle(g_1, \tilde{t}_1, v_1)...(g_n, \tilde{t}_n, v_n)\rangle$, where the set of trajectories is $\tilde{\mathcal{T}}$.

3.2 Data Driven Modeling

The increasingly pervasiveness of GPS sensors has accumulated large scale driving behavior data, which makes it possible to identify human mobility signature from trajectories. However, two challenges arise in achieving this goal. First, the pool of agents is large but the number of trajectories per agent is limited and a large number of new agents rise up every day, thus the data is sparse and maybe only subset of the data can be seen during training. Second, as a type of sequential data, trajectories has temporal dependencies which needs to be learned. We outline how we tackle these two main challenges next.

3.2.1 Siamese networks. To address the first challenge, we employ the siamese networks[5, 32]. Siamese networks train a metric to measure the similarity (or dissimilarity) from data, where the number of categories is very large or even not known during training, and where the size of training samples for a single category is very small. The key idea of the siamese networks is to find a function that maps the input patterns X into a lower-dimensional target space E_θ to approximate the "semantic" distance in the input space, where similar inputs are closer and dissimilar inputs are separated by a margin. Learning the dissimilarity metric is done by training a network, which consists of two identical sub-networks with shared weights. Fig. 4 shows an illustration of this structure. In particular, the end-to-end dissimilarity metric learning is replicated twice (one for each input) and the representations $G_\theta(X_1)$, $G_\theta(X_2)$ are used to predict whether the two inputs belong to the same category. A commonly used optimization function for siamese networks training[5] is :

$$\begin{aligned} \min_\theta & -((1-Y)L_s(E_\theta(X_1,X_2)^i) + YL_d(E_\theta(X_1,X_2)^i)) \\ & \text{s.t. } E_\theta(X_1,X_2) = \|G_\theta(X_1) - G_\theta(X_2)\|, \end{aligned} \tag{1}$$

where θ is the weights of the neural network, $Y = 0$ if the inputs X_1 and X_2 belong to the same category and $Y = 1$ otherwise, $E_\theta(X_1,X_2)^i$ is the i-th sample, which consists of a pair of inputs and a label (similar or dissimilar), L_s is the partial loss function for a similar pair, L_d is the partial loss function for an dissimilar pair.

3.2.2 Long short-term memory (LSTM) networks. The second challenge is that trajectory data has temporal dependencies. Therefore, we employ LSTM networks[11], which are capable of learning long-term dependencies for sequential data $(x_1, x_2, ..., x_T)$. LSTM sequentially updates a hidden-state representation by introducing a memory state C_t and input gate i_t, output gate o_t and forget gate f_t to control the flow of information through the time steps. At each time step $t \in \{1, 2, ..., T\}$, the hidden-state vector h_t as:

$$\begin{aligned} i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\ f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\ o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\ \tilde{C}_t &= \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \\ C_t &= \sigma(f_t * C_{t-1} + i_t * \tilde{C}_t) \\ h_t &= \tanh(C_t) * o_t, \end{aligned} \tag{2}$$

where W_x represents the weights for the respective gate(x) neurons and b_x is the bias for the respective gate(x). Since the trajectory data is a sequence of $(g, \tilde{t}, v)$, we employ LSTM to learn the embeddings of trajectories in the framework of siamese networks.

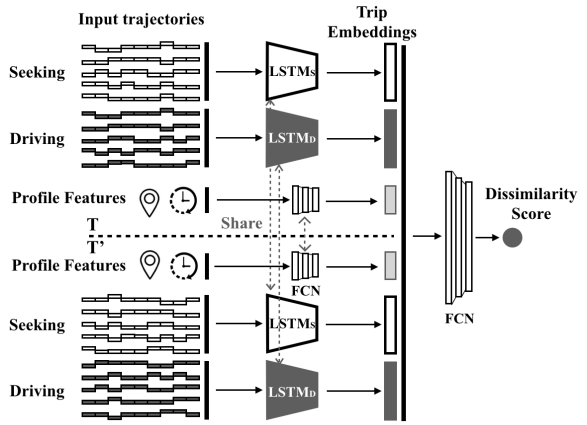


Figure 5: ST-SiameseNet framework

3.2.3 ST-SiameseNet. Given those two challenges, we present a spatio-temporal representation learning method to verify human mobility signature identity by implementing siamese networks with LSTM, named as ST-SiameseNet. The purpose of this paper is to learn the representation of trajectory with limited data and to use representation of trajectory data to compare or match new samples from previously-unseen categories (e.g. trajectories from agents not seen during training). To represent the feature of trajectory and learn the dissimilarity of driving behavior, we incorporate LSTM and fully-connected networks (FCN) into the middle layer of ST-SiameseNet, which is briefly depicted in Fig. 5.

We first extract profile features for each agent and the online feature for each data point in the trajectory. And then randomly select a pair of seeking trajectories $\tilde{\mathcal{T}}_{s,1}$ and $\tilde{\mathcal{T}}_{s,2}$, a pair of driving trajectories $\tilde{\mathcal{T}}_{d,1}$ and $\tilde{\mathcal{T}}_{d,2}$ (all the trajectories contain the online feature) and a pair of profile features $\mathbf{f}_{p,1}$ and $\mathbf{f}_{p,2}$ in each time period. In the original siamese networks, there are two sub-networks with identical weights. To a further step, we introduce six sub-networks where each two identical sub-networks share the set of weights since we have three types of inputs, i.e., $\tilde{\mathcal{T}}_{s,1}$ and $\tilde{\mathcal{T}}_{s,2}$ would share the weights of $LSTM_S$, while $\tilde{\mathcal{T}}_{d,1}$ and $\tilde{\mathcal{T}}_{d,2}$ use the same $LSTM_D$ to learn the representation. Since each agent has a vectorized profile features in each time period, here we implement fully-connected layers as a *profile-learner* to project the profile features. ST-SiameseNet learns the driving behavior from seeking, driving trajectories and profile features respectively and aggregates the embedding layers with a sequence of fully-connected layers, i.e. *dissimilarity-learner* as the dissimilarity metric. Differing from previous works [5] that use the L_1 norm to approximate the "semantic" distance, we utilize neural networks (as a more powerful function) to learn the dissimilarity.

The learning process minimizes the binary cross entropy loss that drives the dissimilarity metric to be small for pairs of trajectories from the same agent, and large for those from different agents. To achieve this property, we pose the following ST-SiameseNet optimization problem:

$$\begin{aligned} &\min_{\theta} -(y \log(D_\theta(X_1, X_2)) + (1-y)\log(1 - D_\theta(X_1, X_2))), \\ &\text{s.t. } X_1 = (\tilde{\mathcal{T}}_{s,1}, \tilde{\mathcal{T}}_{d,1}, \mathbf{f}_{p,1}), X_2 = (\tilde{\mathcal{T}}_{s,2}, \tilde{\mathcal{T}}_{d,2}, \mathbf{f}_{p,2}), \end{aligned} \tag{3}$$

where $y = 0$ if the trajectories belong to the same agent and $y = 1$ if the trajectories come from two different agents, $D_\theta(X_1, X_2)$ is the prediction probability of how likely the trajectories are from two different agents.

Algorithm 1 shows the training process of the ST-SiameseNet model. During the training process, we apply the gradient descent approach to update parameters $\boldsymbol{\theta}$, with learning rate α and a predefined i_{max}, (i.e. the total number of iterations). We first extract profile features $\mathbf{f}_p$ from trajectories $\mathcal{T}$ for each agent. And then compute the online feature $f_{o,1}$, i.e. speed information in each grid cell and update trajectories with the online feature from $\mathcal{T}$ to $\tilde{\mathcal{T}}$. Moreover, we split trajectories into seeking trajectories $\tilde{\mathcal{T}}_s$ and driving trajectories $\tilde{\mathcal{T}}_d$ for each agent. Since ST-SiameseNet pair-wisely trains the data, we randomly select a pair of trajectories, either from the same agent or from different agents in two time periods, with equal probability in each iteration. Next, we update ST-SiameseNet parameters θ by using Eq 3, with α as the step size (Line 7).

Algorithm 1 ST-SiameseNet Training

Require: Trajectories $\mathcal{T}$, initialized parameters $\boldsymbol{\theta}$, learning rate α, max iteration i_{max}.
Ensure: A well trained ST-SiameseNet with parameters $\boldsymbol{\theta}$.
1: Extract profile feature expectation vector $\mathbf{f}_p$.
2: Calculate the online feature $f_{o,1}$ and update trajectories from $\mathcal{T}$ to $\tilde{\mathcal{T}}$.
3: Split seeking $\tilde{\mathcal{T}}_s$ and driving $\tilde{\mathcal{T}}_d$ trajectories.
4: Sample a pair of trajectories with the online feature $\tilde{\mathcal{T}}_{s,i}$, $\tilde{\mathcal{T}}_{d,i}$ and a pair of profile features $\mathbf{f}_{p,i}$.
5: **while** iter $< i_{max}$ **do**
6: Calculate gradient $\nabla g(\boldsymbol{\theta})$ using Eq 3.
7: Update $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \nabla g(\boldsymbol{\theta})$.
8: **end while**

4 EXPERIMENTAL EVALUATION

In this section, we demonstrate the effectiveness of our proposed method by utilizing GPS records collected 10 workdays from 2197 taxis in Shenzhen, China in July 2016. We compare our model with other baseline methods, analyze the generalization of our approach and evaluate the importance of transit modes and profile features for each agent. To support the reproducibility of the results in this paper, we have released our code at Github [1].

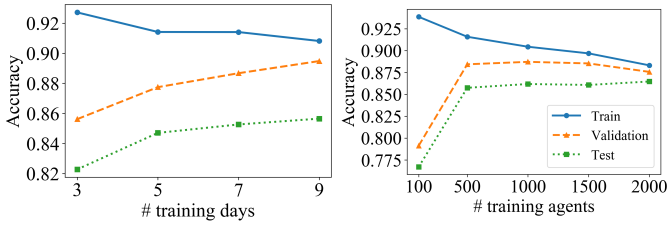
4.1 Evaluation Metrics

To evaluate the performance of our proposed model and baseline methods, we measure accuracy, precision, recall and F_1 score against the ground truth among labels. In our implementation, the dissimilarity score threshold is set to 0.5. If it is less than 0.5, we consider that the trajectories belong to the same agent. Otherwise they are from different agents. Note the threshold can be tuned on different datasets. In particular, precision is intuitively the ability of the classifier not to confuse different agents. Recall shows the ability of the classifier not to miss pairs of different drivers. The F_1 score is a weighted average of the precision and recall.

4.2 Baseline Algorithms

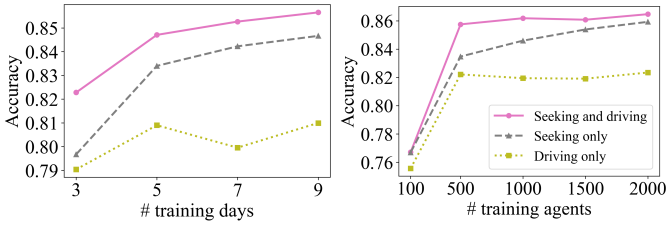
We compare the performances of our method against the following baseline algorithms.

[1] https://github.com/huiminren/ST-SiameseNet



(a) Models accuracy across days (b) Models accuracy across agents

Figure 6: Accuracy across days and agents



(a) Transit modes across days (b) Transit modes across agents

Figure 7: Model comparison among transit modes

(1) **Support Vector Machine (SVM).** Taigman et al. [32] tested the similarity between faces using a linear SVM. Here we utilize the set of profile features as input, described in section 3.1. We conduct absolute difference between profile feature vectors of two agents and then employ SVM to classify whether these two agents are same.

(2) **Fully-connected Neural Network (FNN).** Fully-connected neural network is a basic classification or regression model in deep learning. Here we concatenate all the trajectories in two time periods from two agents together as the inputs of the neural network. In addition, we compare the model accuracy with and without features.

(3) **Naive Siamese Network.** Chopra et al. [5] used a Siamese architecture for face verification. Here we train a network which consists of two identical fully connected networks that share the same set of weights. We concatenate all the trajectories in two time periods from each agent and evaluate the baseline with and without features.

(4) **ST-SiameseNet-L_1.** To evaluate the advantages of using FCN than a predefined function in learning dissimilarity, we replace FCN with the L_1-norm distance to approximate the "semantic" distance.

4.3 Results

4.3.1 Comparison results. We compare our ST-SiameseNet with the baseline models in terms of precision, recall and F_1 score. All the models train with trajectories from 500 agents in 5 days, validate with trajectories from the same agents as training set but in another 5 days and test with trajectories from 197 new agents in the latter 5 days. Similar to the training dataset, we uniformly sample two sets of trajectories from the same agent or different agents in two time periods during validation and testing. Table 1 shows the evaluation metrics from all the methods. It is clear that our approach achieves the best performance. SVM outperforms other baseline models using profile features but is worse than our model. This is because SVM is not able to model sequential inputs and the aggregation will lose information of driving behavior. With profile and basic features added, both FNN and Siamese FNN work better, indicating that features can provide useful information in both models. However, all of the deep learning and machine learning models perform worse than our model, since ST-SiameseNet has a more effective ability to capture the information of sequential inputs by using LSTM. In addition, ST-SiameseNet-L_1 performs worse than ST-SiameseNet with FCN to learn the dissimilarity, showing that L_1 norm has limited ability to learn the dissimilarity between two identities. In particular, the F_1 score of ST-SiameseNet is over 0.85, which is significantly higher than all baselines.

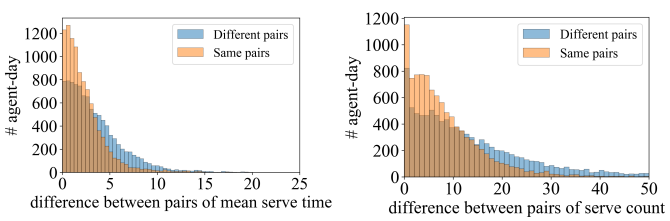
Table 1: Average F_1, recall and precision on real-world dataset and comparison with baselines

Methods	Precision	Recall	F_1 score
ST-SiameseNet	**0.8710**	**0.8317**	**0.8508**
SVM	0.8100	0.7661	0.7874
FNN (with features)	0.6112	0.6298	0.6195
FNN (without features)	0.5266	0.5470	0.5365
Naive Siamese (with features)	0.6137	0.6707	0.6407
Naive Siamese (without features)	0.5580	0.5657	0.5617
ST-SiameseNet-L_1	0.8052	0.7775	0.7910

4.3.2 Model generalization. We evaluate different design choices of our model on classification accuracy. Similar to the previous experiments, we use the trajectories of the first 500 agents in 10 consecutive days as training and validation sets and vary the split ratio.

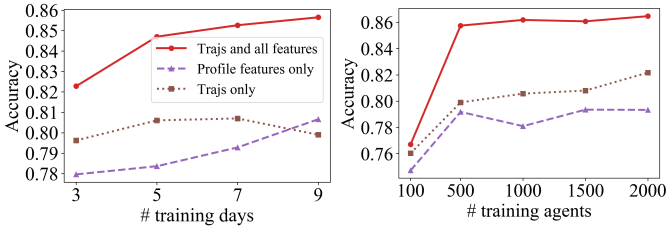
Impact of different number of days. First, we vary the number of days in the training set of the 500 agents to $N_{day} = 3, 5, 7$ and 9 respectively. We train trajectories of 500 agents from Day 1 to Day N_{day}, validate trajectories of the same 500 agents from Day $(N_{day} + 1)$ to Day 10, test trajectories of the new 197 agents from Day $(N_{day} + 1)$ to Day 10. Fig. 6a depicts the training, validation and test accuracy across different days. As more days are added to the training dataset, the training accuracy decreases slightly, while validation and test accuracy gradually increase, indicating that larger datasets can help with over-fitting problem. In addition, when the number of days extending from 3 to 5, both the validation and test accuracy have a dramatically increase, while the validation and test accuracy have a small increase after adding more days, indicating that trajectories of 500 agents from 5 days contain enough information to learn the similarity of agents.

Impact of different number of agents. we also vary the training dataset size by using a subset of the agents. Subsets of sizes $N_{agents} = 100, 500, 1000, 1500$ and 2000 agents in 5 days are used. We train trajectories of N_{agents} agents from Day 1 to Day 5, validate trajectories of the same N_{agents} agents from Day 6 to Day 10, test trajectories of new 197 agents from Day 6 to Day 10. Fig. 6b shows the training, validation and test accuracy across different number of agents. With more agents added to the training dataset, the neural networks have better generalizability and can have a better performance when seeing new data. There is an enormous increase of validation and test accuracy when the number of training agents growing from 100 to 500, indicating that the neural networks benefits from the increase of diversity of agents. However, similar to the impact of different number of days, the validation and test accuracy are flattening when adding more agents to the training pool,



(a) Mean serving time (b) Number of serves per day

Figure 8: Analysis of profile features



(a) Features across days (b) Features across agents

Figure 9: Model comparison among features

showing that trajectories of 500 agents from 5 days are sufficient to learn the mobility signatures of agents.

4.3.3 Importance of transit modes. Transit modes can show different driving habits among different agents. Seeking and driving trajectories are two typical transit modes, defined based on the situation of the vehicle whether any passenger on-board. Different agents would have different strategies to seek passengers [26]. In this section, we would test if each of the transit mode can contribute to identify the drivers' behavior. As can be seen from Fig. 7a and 7b, both seeking and driving trajectories have the ability of discriminating the same and different agents. All the accuracy of seeking trajectories are higher than driving trajectories, which is consistent with human intuition that different agents have different strategies when seeking passengers, while agents do not have the choice of destination when passengers on board. With both seeking and driving trajectories included, ST-SiameseNet performs the best by integrating the information of both seeking and driving trajectories. Fig. 7a and 7b also show the same trend as Fig. 6a and 6b that models of seeking trajectories only and driving trajectories only benefit from larger dataset.

4.3.4 Importance of features. In this section, we evaluate the importance of features by comparing our model with and without features. Each agent has unique personal characteristics which can be extracted from his/her trajectory data over a time period [26]. In addition, Speed information of each trajectory are able to capture agents' driving behavior [8]. Fig.8a and 8b describe the distribution of two profile features, mean serving time and number of service trips. The orange bar depicts the absolute difference of profile features between same agents, while the blue bar otherwise. There is an obvious difference between same agents and different agents, indicating the profile features are able to identify the similarity of driving behavior.

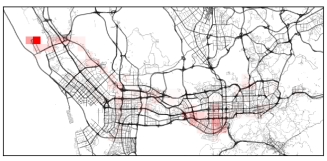
From Fig. 9a and 9b, we can see that our ST-SiameseNet with trajectories and features works the best comparing with the model with profile features only as well as with trajectories only in different number of training days and different number of training agents. In particular, the model with profile features only gets the worst performance, indicating that the aggregation may lose information of driving behavior. Besides, if we only use trajectories as inputs i.e. $tr = \langle s_1, s_2 ... s_n \rangle$ ($s = \langle g, \tilde{t} \rangle$), all the test accuracy across days and agents are also lower than our ST-SiameseNet with both trajectories and features included, probably because some statistical information, such as mean seeking distance, mean seeking time, number of serves, cannot be captured by LSTM. In addition, Fig. 9a and 9b shows the same trend as Fig. 6a and 6b that the neural networks benefit from larger datasets. Overall, with raw trajectory data and extracted features working together, we can learn the driving patterns from trajectory data more effectively and verify the agent more accurately. Note that we only extract 11 profile features and 1 online feature, which requires little human work of feature engineering comparing with [6], which extracted 137 statistical human-defined features.

4.4 Case Studies

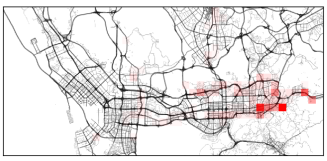
To further understand how ST-SiameseNet identifies agents' behaviors, we investigate individual agents' cases to show what factors ST-SiameseNet considers when identifying the agents. Four case studies are presented.

4.4.1 Cases of identifying different drivers. First, we show an example of two randomly selected human agent taxi driver, driver 1 and driver 2. We extract their trajectories and profile features on July 4^{th}, 2016, then, our proposed ST-SiameseNet consumes the trajectories and features and produces a dissimilarity score of **0.99**, which means ST-SiameseNet identifies that this pair of inputs is from two different agents. To figure out what factors that ST-SiameseNet consider to identify them, we visualize the heat map of their visitation frequency on that day to each grid of the city in Fig.10a&10b. The darker red color in the grid indicates higher visitation frequency. Fig.10a&10b illustrate that driver 1 and driver 2 have significantly different active regions. Driver 1 likes working in the west part of the city, especially near the airport, while driver 2 prefers to work in the east part near the downtown area. The difference in the active regions of driver 1 and driver 2 helps ST-SiameseNet identify them. Fig.12a shows the comparison of the profile features of driver 1 and driver 2, which illustrates that they have significantly different feature values on $f_{p,2}$: *the longest staying grid id in latitude direction* and $f_{p,6}$: *the most frequently visited grid id in latitude direction*. This is consistent with the finding from the heat map, i.e., the difference in their active regions.

ST-SiameseNet can also identify different drivers even if their active regions are similar to each other. We select another case of identifying different drivers from our data randomly. Fig.11a&11b show the heat map of the visitation frequency of driver 3 and driver 4, which indicate that driver 3 has similar active region as driver 4. They both like working near the downtown area. And our proposed ST-SiameseNet successfully identifies them with a dissimilarity score of **0.88**, which means they are significantly different. Fig.11a&11b show that driver 3 and driver 4 have similar active region near the downtown area, and the difference on $f_{p,2}$: *the longest staying grid id in longitude direction* and $f_{p,6}$: *the most frequently visited grid id in longitude direction* is small as shown in Fig.12b. However, they have significantly different profile feature

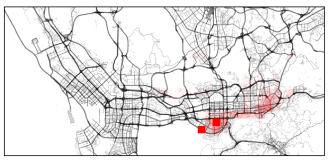


(a) Driver 1

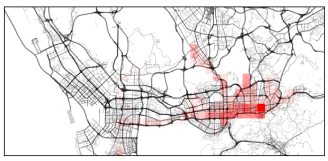


(b) Driver 2

Figure 10: Case 1 of identifying different drivers

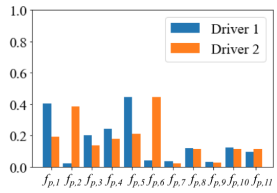


(a) Driver 3

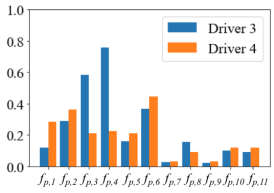


(b) Driver 4

Figure 11: Case 2 of identifying different drivers



(a) Driver 1 & driver 2



(b) Driver 3 & driver 4

Figure 12: Profile feature comparison

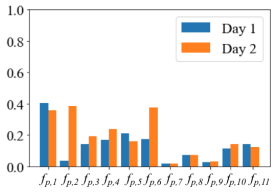


Figure 13: Case 3 profile features

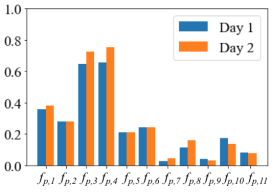


Figure 14: Case 4 profile features

values on $f_{p,3}$ & $f_{p,4}$: *Break start & end time.*, and this information has been discovered by ST-SiameseNet, thus driver 3 and driver 4 can be identified by ST-SiameseNet.

4.4.2 Case of identifying ***abnormal*** *behavior of one driver ID.* Apart from the case of identifying different drivers, ST-SiameseNet can deal with the case of identifying abnormal driving behavior of "one" driver. Our dataset contains only the licence plate of each vehicle. Therefore, abnormal behaviors of the same vehicle can might suggest a change of driver. Here, we study a driver's behavior in 2 days from July 5^{th} to July 6^{th} 2016. Let's call this driver "John". Our ST-SiameseNet produces a dissimilarity score of **0.84** for the trajectories in these 2 days, which indicates that John's behavior changes significantly from day 1 to day 2. To figure out how John's behavior changed significantly, we plot the heat map of his visitation frequency in Fig.15, from which, we find that his active region changes from the west part of the city in day 1 to the east part in day 2. Also, Fig.13 shows the profile features in these 2 days. Most of the profile features change significantly, e.g., $f_{p,2}$: *the longest staying grid id in longitude direction*, $f_{p,6}$: *the most frequently visited grid id in longitude direction*, $f_{p,3}$ & $f_{p,4}$: *Break start & end time.* We also study a few more days after day2, the behaviors are similar to that in day2, thus, this abnormal behavior after day 1 appears to be the result of a new driver operating the vehicle after day 1.

4.4.3 Case of identifying ***normal*** *behavior of one driver ID.* For the normal behavior of a driver, ST-SiameseNet can identify it correctly. Here, we study a driver's behavior in 2 days from July 5^{th} to July 6^{th} 2016. Let's call this driver "Mike". Our ST-SiameseNet produces a dissimilarity score of **0.03**, which indicates that Mike's behavior remains consistent from day 1 to day 2. The heat map of his visitation frequency in Fig.16 illustrates his active region does not change. Also, his profile features in these 2 days as shown in Fig.14 remain stable.

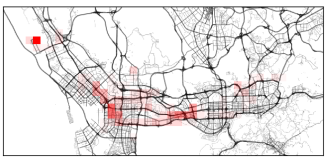
5 RELATED WORK

Human mobility signature identification has been extensively studied in recent years due to the emergence of the ride-sharing business model and urban intelligence[7, 18, 33, 38]. However, to the best of our knowledge, *we make the first attempt to employ siamese network to verify human mobility signature identification.* Related work are summarized below.

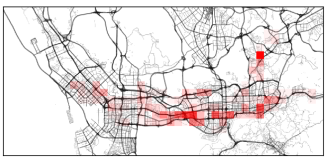
Urban computing. Urban computing is a general research area which integrates urban sensing, data management and data analytic together [16, 20, 22, 23, 37]. In particular, a group of work focus on taxi operation management, such as dispatching [14, 29] and passenger seeking [34–36]. They aim at finding an optimal actionable solution to improve the performance/revenue of individual taxi drivers or the entire fleet. Rong et al.[28] solved the passenger seeking problem by giving direction recommendations to drivers. However, all of these works focus on finding "what" are the best driving strategies (as an optimization problem), rather than considering the benefits of passengers. By contrast, our work focuses on driver identification, which can enhance the safety of passengers.

Driver behavior learning. Most existing literature on human driving behavior rely on human-defined driving style feature set. These handcrafted vehicle movement features derived from sensor data or constructed from real-world GPS data [6, 9, 10, 21, 38]. They used supervised classification, unsupervised clustering or reinforcement learning to solve problems as such driver identification, sequential anomaly detection,etc [8, 17, 21, 25, 38]. Ezzini et al. [9] addressed the driver identification problem using real driving datasets consisting of measurements taken from in-vehicle sensors, such as driver camera, smartphone are placed within the car and the driver is connected to electrodes and skin conductance response. However, such existing work require expensive sensor installed in the vehicle or excessively rely on human-defined features. Dong et al. [8] proposed a deep-learning framework to driving behavior analysis based on GPS data. They used CNN and RNN respectively to predict driver identity among 50 and 1000 drivers for a given trajectory. Such frameworks is generally require all the categories be known in advance as well as the training examples be available for all the categories, as opposed to our objective which only a subset of the categories is known at the time of training.

Siamese network. The siamese network [3] is an architecture for similarity learning of inputs, which has been widely used in multiple applications, namely but a few, vision area, unsupervised

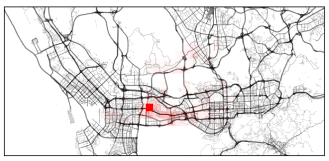


(a) day 1

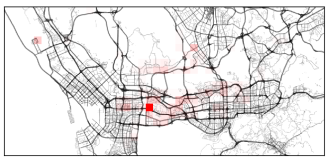


(b) day 2

Figure 15: Case 3: Abnormal driving behavior



(a) day 1



(b) day 2

Figure 16: Case 4: Normal driving behavior

acoustic modelling, natural language processing [5, 12, 13, 15, 31]. Chopra et al.[5] learned complex similarity metrics of face verification by introducing convolutional networks to siamese networks. Hoffer and Ailon [12] proposed a variant of siamese networks, triplet networks to learn an image similarity. Hu et al.[13] applied siamese networks with convolutional layers to match two sentences. However, to our best knowledge, we are the first one to employ siamese network to human-generated spatio-temporal data.

6 CONCLUSION

In this paper, we propose the Spatio-temporal Siamese Networks (ST-SiameseNet) to solve the Human Mobility Signature Identification (HuMID) problem. The HuMID problem aims at validating if an income set of trajectories belong to a certain agent based on historical trajectory data. ST-SiameseNet can deal with large group of agents in a single model. Also, we extract several effective profile features from the trajectories to augment the performance of the ST-SiameseNet. The experimental results illustrate that ST-SiameseNet outperforms state-of-the-art works and achieves an F_1 score of 0.8508 on a real-world taxi trajectory dataset. Our proposed ST-SiameseNet framework can be applied to many other real-world cases with human-generated spatio-temporal data other than the ride-sharing and taxi case. In the future, we will continue studying the driver identification problem with multiple inputs rather than pairwise inputs.

7 ACKNOWLEDGEMENTS

Huimin Ren, Menghai Pan and Yanhua Li were supported in part by NSF grants IIS-1942680, CNS-1657350 and CMMI-1831140. Xun Zhou was partially supported by a grant from the SAFER-SIM University Transportation Center.

REFERENCES

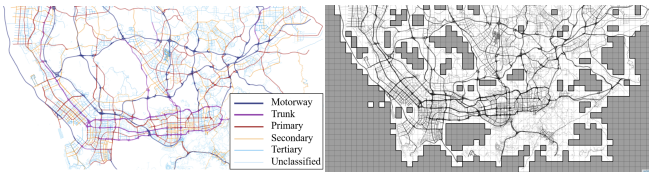
[1] OpenStreetMap. http://www.openstreetmap.org/.
[2] Taxi, Uber, and Lyft Usage in New York City. http://toddwschneider.com/posts/taxi-uber-lyft-usage-new-york-city/.
[3] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah. Signature verification using a" siamese" time delay neural network. In *NIPS*, pages 737–744, 1994.
[4] F. Chollet et al. Keras. https://keras.io, 2015.
[5] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, pages 539–546. IEEE, 2005.
[6] A. Chowdhury, T. Chakravarty, A. Ghose, T. Banerjee, and P. Balamuralidhar. Investigations on driver unique identification from smartphone's gps data alone. *Journal of Advanced Transportation*, 2018, 2018.
[7] Y. Ding, Y. Li, K. Deng, H. Tan, M. Yuan, and L. M. Ni. Detecting and analyzing urban regions with high impact of weather change on transport. *IEEE Transactions on Big Data*, 2016.
[8] W. Dong, J. Li, R. Yao, C. Li, T. Yuan, and L. Wang. Characterizing driving styles with deep learning. *arXiv preprint arXiv:1607.03611*, 2016.
[9] S. Ezzini, I. Berrada, and M. Ghogho. Who is behind the wheel? driver identification and fingerprinting. *Journal of Big Data*, 5(1):9, 2018.
[10] D. Hallac, A. Sharang, R. Stahlmann, A. Lamprecht, M. Huber, M. Roehder, J. Leskovec, et al. Driver identification using automobile sensor data from a single turn. In *2016 IEEE 19th ITSC*, pages 953–958. IEEE, 2016.
[11] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
[12] E. Hoffer and N. Ailon. Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*. Springer, 2015.
[13] B. Hu, Z. Lu, H. Li, and Q. Chen. Convolutional neural network architectures for matching natural language sentences. In *NIPS*, pages 2042–2050, 2014.
[14] J. Yuan, Y. Zheng, L. Zhang, X. Xie. T-Finder: A Recommender System for Finding Passengers and Vacant Taxis. *IEEE TKDE*, 25(10):2390–2403, 2013.
[15] H. Kamper, W. Wang, and K. Livescu. Deep convolutional acoustic word embeddings using word-pair side information. In *ICASSP*, pages 4950–4954. IEEE, 2016.
[16] A. V. Khezerlou, X. Zhou, L. Li, Z. Shafiq, A. X. Liu, and F. Zhang. A traffic flow approach to early detection of gathering events: Comprehensive results. *ACM TIST*, 8(6):74, 2017.
[17] T. Kieu, B. Yang, C. Guo, and C. S. Jensen. Distinguishing trajectories from different drivers using incompletely labeled trajectories. In *Proceedings of the 27th ACM International CIKM*, pages 863–872. ACM, 2018.
[18] Y. Li, J. Luo, C.-Y. Chow, K.-L. Chan, Y. Ding, and F. Zhang. Growing the charging station network for electric vehicles with trajectory data analytics. In *ICDE*, 2015.
[19] Y. Li, M. Steiner, J. Bao, L. Wang, and T. Zhu. Region sampling and estimation of geosocial data with dynamic range calibration. In *ICDE*, 2014.
[20] C. Liu, K. Deng, C. Li, J. Li, Y. Li, and J. Luo. The optimal distribution of electric-vehicle chargers across a city. In *ICDM*. IEEE, 2016.
[21] J. O. López, A. C. C. Pinilla, et al. Driver behavior classification model based on an intelligent driving diagnosis system. In *15th ITS*, pages 894–899. IEEE, 2012.
[22] B. Lyu, S. Li, Y. Li, J. Fu, A. C. Trapp, H. Xie, and Y. Liao. Scalable user assignment in power grids: a data driven approach. In *SIGSPATIAL GIS*. ACM, 2016.
[23] M. Qu, H. Zhu, J. Liu, G. Liu, H. Xiong. A Cost-Effective Recommender System for Taxi Drivers. In *The 20th SIGKDD'14*, pages 45–54, New York, NY, 2014. ACM.
[24] A. MARSHALL. Uber's New Features Put a Focus on Rider Safety . https://www.wired.com/story/ubers-new-features-focus-rider-safety/, 09 2019.
[25] M.-h. Oh and G. Iyengar. Sequential anomaly detection using inverse reinforcement learning. In *Proceedings of the 25th ACM SIGKDD*, pages 1480–1490, 2019.
[26] M. Pan, Y. Li, X. Zhou, Z. Liu, R. Song, H. Lu, and J. Luo. Dissecting the learning curve of taxi drivers: A data-driven approach. In *Proceedings of the 2019 SIAM SDM*, pages 783–791. SIAM, 2019.
[27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
[28] H. Rong, X. Zhou, C. Yang, Z. Shafiq, and A. Liu. The rich and the poor: A markov decision process approach to optimizing taxi driver revenue efficiency. In *Proceedings of the 25th CIKM*, pages 2329–2334. ACM, 2016.
[29] W. S. Ma, Y. Zheng. A large-scale dynamic taxi ridesharing service. In *The 29th ICDE'13*, pages 410–421, New York, NY, 2013. IEEE.
[30] F. Siddiqui. Uber makes changes amid swarm of criticism over rider safety. https://www.washingtonpost.com/technology/2019/09/26/uber-makes-safety-changes-amid-swarm-criticism-over-protection-riders/, 19.
[31] G. Synnaeve, T. Schatz, and E. Dupoux. Phonetics embedding learning with side information. In *2014 IEEE SLT*, pages 106–111. IEEE, 2014.
[32] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on CVPR*, pages 1701–1708, 2014.
[33] T. Xu, H. Zhu, X. Zhao, Q. Liu, H. Zhong, E. Chen, and H. Xiong. Taxi driving behavior analysis in latent vehicle-to-vehicle networks: A social influence perspective. In *Proceedings of the 22nd SIGKDD*, pages 1285–1294. ACM, 2016.
[34] Y. Ge and H. Xiong and A. Tuzhilin and K. Xiao and M. Gruteser. An energy-efficient mobile recommender system. In *The the 16th International Conference on KDD*, pages 899–908, New York, NY, 2010. ACM.
[35] Y. Ge, C. Liu, H. Xiong, J. Chen. A Taxi Business Intelligence System. In *The 17th International Conference on KDD*, pages 735–738, New York, NY, 2011. ACM.
[36] J. Yuan, Y. Zheng, L. Zhang, X. Xie, and G. Sun. Where to find my next passenger. In *Proceedings of the 13th Ubicomp*, pages 109–118, New York, NY, 2011. ACM.
[37] Z. Yuan, X. Zhou, and T. Yang. Hetero-convlstm: A deep learning approach to traffic accident prediction on heterogeneous spatio-temporal data. In *Proceedings of the 24th ACM SIGKDD*, pages 984–992. ACM, 2018.
[38] X. Zhang, X. Zhao, and J. Rong. A study of individual characteristics of driving behavior based on hidden markov model. *Sensors & Transducers*, 167(3):194, 2014.

A APPENDIX FOR REPRODUCIBILITY

In this section, we provide detailed information to support the reproducibility of the results in this paper. We implement deep neural network with machine learning library Keras, version 2.3.0 [4] and using scikit-learn, version 0.22 [27] to implement SVM. Our experiments run on a virtual machine running Red Hat Enterprise Linux 7.2 with 8 GPUs and 32 GB memory.

A.1 Preprocessing of Data

The road map data includes 21, 000 road segments with six levels as shown in Fig. 17a. In this paper, we utilize GPS records from 2197 taxis in Shenzhen, China from July 4^{th} to July 15^{th} (10 workdays) 2016. To keep enough information in each trajectory, we filter out the trajectory which length is less than 10 steps, where each step is a tuple of a grid and a time slot. After filtering out those grids that taxis cannot reach, there are 1934 valid grids as shown in Fig. 17b. For each driver, we randomly select 5 seeking and 5 driving trajectories in each work day as partial inputs of our ST-Siamese. 11 profile features and 1 online feature are extracted from the GPS records data.



(a) Shenzhen road map **(b) Map gridding**

Figure 17: Shenzhen map data

A.2 Settings of Baselines

- **SVM.** We use the default hyper-parameters of SVM in scikit-learn, i.e. $C = 1.0$ and radial basis function is used as kernel function.
- **FNN.** In the FNN experiments, we set 7 layers as [2000, 1500, 1000, 500, 100, 50, 10, 1]. We use ReLU activation functions for all hidden layers and sigmoid activation at the output layer.
- **Naive Siamese.** In the experiments, the neuron sizes in each layer are [1000, 500, 100, 50, 10, 1] with ReLU activation function and we also use sigmoid activation at the output layer.

A.3 Settings of ST-Siamese

We train our model on Shenzhen taxi dataset as a balanced binary class classification problem, since we randomly select a pair of trajectories with equal probability in each iteration. To predict whether the trajectories from two time periods belong to the same driver, we implement the standard back-propagation on feed-forward networks by adaptive moment estimation (Adam) with first momentum (set to 0.9) and second momentum (set to 0.999). Our mini-batch size is 1 since each trajectory has variable length sequences. The learning rate is 0.00006. We trained the network for 1000000 iterations which took 2 days.

The following is the structure of ST-SiameseNet for human mobility signature identification:

- $LSTM_S$. The trajectories are embedded by two hidden layers, which contains 200 and 100 units respectively.
- $LSTM_D$. The $LSTM_D$ has the same components of neurons as $LSTM_S$
- *Profile-learner.* The features are embedded by a three-layer fully-connected network with hidden units [64,32,8]. We use ReLU activation functions for all hidden layers.
- *Similarity-learner.* It's a three-layer fully-connected network with hidden units [64,32,8,1]. We use ReLU activation functions for all hidden layers and sigmoid activation at the output layer.