2023 IEEE International Conference on Data Mining (ICDM) | 979-8-3503-0788-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICDM58522.2023.00024

Self-supervised Pre-training for Robust and Generic Spatial-Temporal Representations

Mingzhi Hu[1], Zhuoyun Zhong[1], Xin Zhang[2], Yanhua Li[1], Yiqun Xie[3], Xiaowei Jia[4],
Xun Zhou[5], Jun Luo[6]
Worcester Polytechnic Institute[1], San Diego State University[2], University of Maryland[3],
University of Pittsburgh[4], University of Iowa[5],
Logistics and Supply Chain MultiTech R&D Centre, Hong Kong[6]
mhu3@wpi.edu, zzhong3@wpi.edu, xzhang19@sdsu.edu, yli15@wpi.edu, xie@umd.edu,
xiaowei@pitt.edu, xun-zhou@uiowa.edu, jluo@lscm.hk

***Abstract*—Advancements in mobile sensing, data mining, and artificial intelligence have revolutionized the collection and analysis of Human-generated Spatial-Temporal Data (HSTD), paving the way for diverse applications across multiple domains. However, previous works have primarily focused on designing task-specific models for different problems, which lack transferability and generalizability when confronted with diverse HSTD. Additionally, these models often require a large amount of labeled data for optimal performance. While pre-trained models in Natural Language Processing (NLP) and Computer Vision (CV) domains have showcased impressive transferability and generalizability, similar efforts in the spatial-temporal data domain have been limited. In this paper, we take the lead and introduce the Spatial-Temporal Pre-Training model, *i.e.*, STPT, which is connected with a self-supervised learning task, to address these limitations. STPT enables the creation of robust and versatile representations of HSTD. We validate our framework using real-world data and demonstrate its efficacy through two downstream tasks, *i.e.*, trajectory classification and driving activity identification (*e.g.*, identifying seeking *vs*. serving behaviors in taxi trajectories). Our results achieve an accuracy of 83.125% (16.2% higher than the average baseline) for human mobility identification and an accuracy of 77.88% (13.0% higher than the average baseline) for the human activity identification task. These outcomes underscore the potential of our pre-trained model for diverse downstream applications within the spatial-temporal data domain.**

***Index Terms*—pre-training, self-supervised learning, spatial-temporal data mining, driver identification, human decision analysis**

I. Introduction

Advancements in mobile sensing and information technologies, coupled with the proliferation of data mining and artificial intelligence, have brought about a revolutionary shift in the collection and analysis of Human-generated spatial-temporal data (HSTD). This transformation has paved the way for various applications and benefits. One notable application is the human mobility identification models, which utilize human trajectory data, such as GPS traces from pedestrians, taxi drivers, and gig-workers, to identify individual users [1]–[5]. Prominent companies like Uber [6] and Lyft [7] have implemented these models to automatically identify drivers, ensuring authorized operations and enhancing service safety.

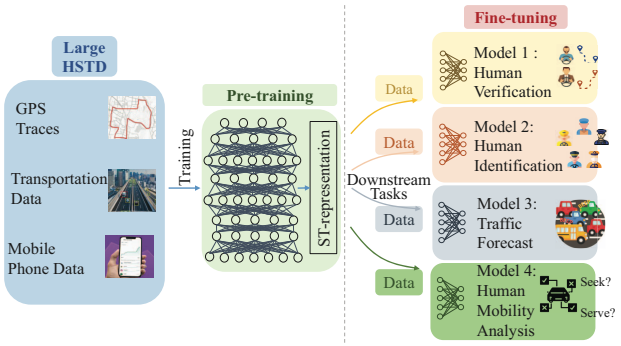


Fig. 1: Solving spatial-temporal tasks with pre-trained model.

Furthermore, numerous research works [8]–[15] rely on HSTD to investigate, monitor and forecast the dynamic urban status (*e.g.*, the traffic speed and volume). By leveraging HSTD, researchers can gain valuable insights into urban transit planning, resource allocation, traffic management, and public safety. Moreover, researchers delve into the study of human mobility patterns and decision-making based on mobility data [16]–[18]. By unveiling and analyzing the knowledge acquired from these datasets, valuable information can be derived to contribute to not only individual decision-makers but also public transportation efficiency and service quality.

While the aforementioned works have demonstrated effectiveness in addressing specific problems and applications, they typically rely on disparate models and datasets tailored to each scenario. These approaches introduce limitations concerning model generalizability and transferability. For example, a model designed for human identification might encounter challenges when repurposed for predicting traffic patterns. As a result, extra time and computation resources are needed to deal with various problems with diverse data sources. Moreover, task-specific models often necessitate a substantial labeled training dataset to ensure the reliability and robustness of the resulting model. For example, when designing an accident alarm model, it is often challenging to obtain a sufficient amount of traffic accident data, which can limit

979-8-3503-0788-7/23/$31.00 ©2023 IEEE
DOI 10.1109/ICDM58522.2023.00024

Authorized licensed use limited to: Gordon Library WPI. Downloaded on May 15,2024 at 14:16:59 UTC from IEEE Xplore. Restrictions apply.

the effectiveness of the model in accurately detecting and predicting accidents.

State-of-the-art pre-trained models. The emergent pre-trained models such as PaLM [19], LLAMA [20], GPT-3 [21], and ChatGPT [22], have shown superb generalizability and model transferability when pre-trained on a large dataset equipped with billions of parameters. During training, they leverage vast corpora and employ a self-supervised objective (*e.g.*, next token prediction in a sequence [23]) without human labeling to train the large model and empower the model with significant language understanding and generation capabilities. They have shown great performances in various tasks like natural language inference and paraphrasing [24]–[26], entity recognition, and question answering [21], [27]–[29], and *etc*. However, there is limited prior work [30] in designing a self-supervised pre-trained model for spatial-temporal tasks.

Our approach. To fill this gap, we are inspired by the success of the pre-trained language models to train a spatial-temporal pre-trained model using self-supervised learning. Such a model extracts knowledge from diverse HSTD (*e.g.*, GPS traces, transportation data, mobile phone data), and can be fine-tuned and applied to various downstream applications (*e.g.*, human verification and identification, traffic forecast, and human mobility analysis) given a small number of training data (for fine-tuning) as is shown in Fig. 1. We thus introduce the Spatial-Temporal Pre-Training model, *i.e.*, STPT, to generate robust and generic representations from HSTD using our novel self-supervised learning approach. *Our contributions* are summarized as follows:

- We present our STPT model which captures the distinct characteristics of HSTD through our innovative self-supervised learning training scheme. It effectively differentiates among various human decision-makers and identifies the interconnectedness within trajectories. By exploiting the inherent complex spatial and temporal patterns in the data, our STSP model offers comprehensive representation learning for HSTD. (See Section III-B).
- The spatial-temporal representations from our pre-trained model are robust and generic to support diverse spatial-temporal downstream tasks, such as trajectory classification, driving activity identification, etc. We design a spatial-temporal fine-tuning algorithm to transform the spatial-temporal pre-trained model for various downstream tasks (See Section III-C).
- We validate our framework using real-world HSTD on two downstream tasks, *i.e.*, trajectory classification and driving activity identification (*i.e.*, identifying taxi seeking *vs*. serving activities). Our result surpasses the average baseline accuracy by 16.2% and 13.0%, highlighting the effectiveness of our pre-trained STPT model in these tasks (See Section IV). *We made our code, unique dataset, and implementation appendix available to contribute to the research community via Github link.*[1].

[1]STPT page: https://github.com/mhu3/STPT

TABLE I: Notations.

Notations	Descriptions
$p = \langle lat, lng, t, sta \rangle$	GPS record with status.
$q = \langle lat, lng, t \rangle$	GPS record without status.
$\tau = \{a, (p_1, p_2, \cdots, p_n)\}$	Trajectory.
$\mathcal{T}$	Trajectory set.
f^{pre}	Self-supervised pre-training model.
g, G	GPS record embedding, and projection.
$E_{pos} = \langle e_1, e_2, ..., e_n \rangle$	Positional embedding
L	Transformer block layer number.
H	The hidden size.
A	Attention weights matrix.
W^q, W^k, W^v	Query, key and value matrices.
q_i, k_i, v_i	Query, key and value of input.
n_{head}	Self-attention head number.

II. Overview

In this section, we introduce the spatial-temporal pre-training problem and outline associated challenges in research. To facilitate understanding and clarity, we provide a summary of the notations used in this paper in Table I.

A. Human-Generated Spatial-Temporal Data

HSTD encapsulates sequential human decisions during mobility. For instance, freight tracking (as GPS traces) and automatic fare collection data (as transaction records) reveal choices made in delivery routes or daily commutes. Hence, HSTD can be interpreted as a series of trajectories, with humans navigating through spatial-temporal states. We formally define these concepts subsequently.

Definition 1. (A trajectory τ). With the wide use of GPS devices on vehicles, smartphones, smartwatches, *etc*., people can generate massive spatial-temporal data anywhere anytime. Each GPS point q consists of a location in latitude lat and longitude lng, and a time stamp t, *i.e.*, $q = \langle lat, lng, t \rangle$. A trajectory τ is a sequence of GPS points generated by the human agent a, denoted as $\tau = \{a, (q_1, q_2, ..., q_n)\}$ and we denote the set of trajectories as $\mathcal{T}$.

Definition 2. (A driving status sta). Driving status categorizes the mobility pattern of a trajectory. For instance, taxi trajectories can be classified into two status, *i.e.*, driving with a passenger on board and without. Private car trajectories can also be grouped based on purpose as commute or recreation. Given different transit modes (*e.g.*, private vehicles, taxis, trains, buses, *etc*.), the driving status sta varies. In this paper, we use the driving status sta to represent broad driving conditions and scenarios. Therefore, we denote p as the GPS record that carries the driving status information sta, *i.e.*, $p = \langle lat, lng, t, sta \rangle$. In some downstream tasks, the driving status sta serves as the label to be predicted.

B. Spatial-Temporal Pre-training

The spatial-temporal pre-training model, denoted as f^{pre}, leverages the spatial-temporal information in HSTD to enable downstream applications including but not limited to human behavior analysis, trajectory classification, and human activity identification.

Authorized licensed use limited to: Gordon Library WPI. Downloaded on May 15,2024 at 14:16:59 UTC from IEEE Xplore. Restrictions apply.

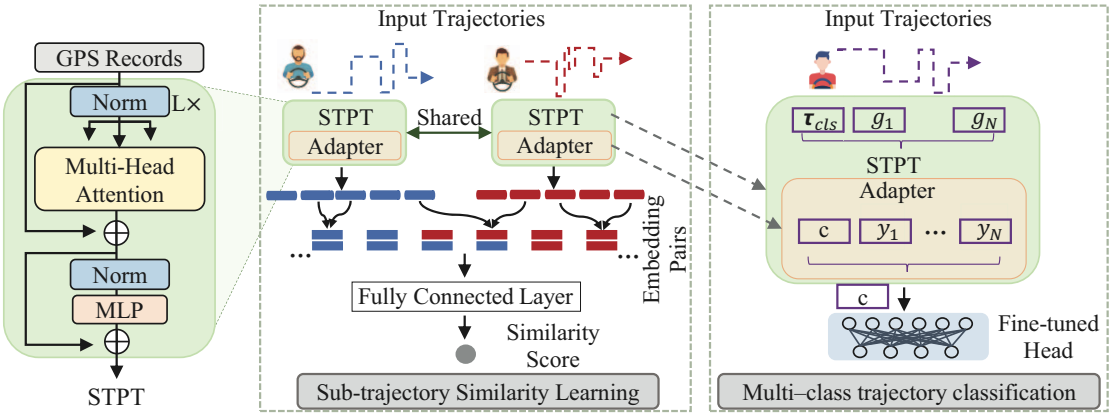


Fig. 2: STPT implementation framework which includes pre-training and fine-tuning. Utilizing identical architectural structures, except for the output layers, the model undergoes fine-tuning on all parameters to adapt to specific tasks. Adapters enable selective fine-tuning of a few top layers based on task requirements. Each input trajectory is marked with a unique identifier [cls] (*i.e.*, "classification") at the beginning.

Definition 3. (A spatial-temporal pre-training model f^{pre}). A spatial-temporal pre-training model f^{pre} is designed to understand and internalize the spatial-temporal dependencies inherent in HSTD. By sequentially ingesting sequences of GPS records, it learns to map these records to outputs that carry meaningful and task-related information. A spatial-temporal pre-training model f^{pre}, undergoes pre-training on a large-scale corpus of HSTD, encompassing diverse spatial-temporal patterns observed across various contexts and individuals.

The input to a spatial-temporal pre-training model f^{pre}, is a trajectory τ consisting of a sequence of GPS points, each associated with a driving status mode sta, *i.e.*, $\tau = a, (p_1, p_2, ..., p_n)$. To adapt to various downstream tasks, f^{pre} generates a comprehensive and generic representation y that effectively captures the fundamental features and essential characteristics inherent in the input trajectory.

C. Spatial-Temporal Pre-training Problem

Problem Definition. Given a set of trajectories $\mathcal{T}$, the objective is to learn a spatial-temporal pre-training model f^{pre} that captures the spatial-temporal patterns and human decision-making strategies reflected in the trajectories. The model f^{pre} should be capable of enhancing performance on various downstream tasks when fine-tuned on task-specific data.

Challenges. The proposed self-supervised pre-training for robust and generic spatial-temporal representations presents two unique research challenges: (**C1**) How to design a self-supervised learning scheme to train a spatial-temporal pre-training model f^{pre} that is able to effectively capture the intricate details embedded in a trajectory, and take into account the complex spatial-temporal scenarios and human decision-making patterns? (See Section III-B) (**C2**) How to effectively leverage the spatial-temporal pre-training model to enhance performance on various downstream tasks, such as the human mobility identification problem and the human activity distinction problem? (See Section III-C)

III. Methodology

To tackle the above challenges, we introduce our Spatial-Temporal Pre-Training model, *i.e.*, STPT, and illustrate its implementation in this section. The STPT training framework contains two steps, *i.e.*, model pre-training and task fine-tuning. During pre-training, the STPT model is trained via self-supervised learning to learn the representations of the trajectories and the dependency of the sub-trajectories to handle the challenge **C1**. For fine-tuning, the STPT model is first initialized with the pre-trained parameters and fine-tuned using labeled data for downstream tasks to handle the challenge **C2**. STPT's overall pre-training and fine-tuning framework are shown in Fig. 2.

A. STPT Architecture & Input Design.

To efficiently capture the local and global information embedded in HSTD, our STPT employs a multi-layer transformer encoder based on the BERT model [23] and the Vision Transformer (ViT) [31].

The input of the model is a 1D sequence of GPS record embedding, and the encoder extracts the latent vector z_l in each layer l. We denote the number of layers (*i.e.*, transformer blocks) as L, and the hidden size as H through all the layers. We train on two model sizes: $\text{STPT}_{\text{BASE}}$ ($H = 128$, $n_{heads} = 8$, $L = 1$) and $\text{STPT}_{\text{LARGE}}$ ($H = 128$, $n_{heads} = 16$, $L = 8$) to compare the impact of the parameter number in the pre-trained model.

To create the input layer to the STPT (*i.e.*, z_0), we first embed the sequence of GPS records in a trajectory τ through a linear projection G to get embedding $g_1, g_2, \ldots, g_N$. Then, we add position embedding to the GPS record embedding to retain positional information, denoted as E_{pos}. Here, a

Authorized licensed use limited to: Gordon Library WPI. Downloaded on May 15,2024 at 14:16:59 UTC from IEEE Xplore. Restrictions apply.

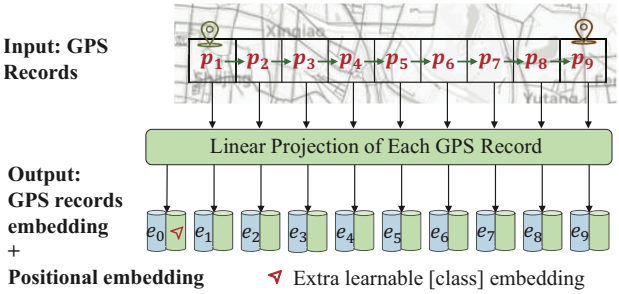


Fig. 3: STPT input embedding as the sum of the GPS records embedding and the position embedding.

standard learnable 1D position embedding is used. We append a learnable embedding to the sequence of embedded GPS records ($z_0^0 = \epsilon_{\text{class}}$), where the state at the output of STPT (z_L^0) serves as the trajectory representation y (described later in Eq. 1). Formally, the input layer z_0 can be represented as:

$$z_0 = [\epsilon_{\text{class}}; g_1; g_2; \ldots; g_N] + E_{\text{pos}},$$
$$G \in \mathbb{R}^{N \times H}, \quad E_{\text{pos}} \in \mathbb{R}^{(N+1) \times H}.$$

The transformer encoder, as described in [32], uses alternating layers of multi-head self-attention (MSA) and multilayer perceptron (MLP) blocks (Eq. 1). The MSA operation involves running n_{head} parallel self-attention heads and concatenating their outputs. Each self-attention head applies the self-attention (SA) mechanism with its own learned projection matrices W^q, W^k, W^v. The input sequence $z \in \mathbb{R}^{N \times H}$ is transformed into query (q), key (k), and value (v) representations as:

$$[q_i, k_i, v_i] = z \cdot W_i^q, z \cdot W_i^k, z \cdot W_i^v.$$

For each self-attention head, the attention weights $A_i \in \mathbb{R}^{N \times N}$ are computed by applying the softmax function to the pairwise dot product of the query (q_i) and key (k_i) representations, divided by $\sqrt{H}$, *i.e.*,

$$A_i = \text{softmax}\left(\frac{q_i \cdot k_i^T}{\sqrt{H}}\right).$$

The self-attention output $SA_i(z)$ for each head is calculated by multiplying the attention weights A_i with the corresponding value representation v_i, *i.e.*,

$$SA_i(z) = A_i \cdot v_i.$$

Finally, the outputs of all n_{head} attention heads are concatenated and projected using a learnable matrix $W_{\text{MSA}} \in \mathbb{R}^{n_{head}H \times H}$:

$$MSA(z) = \text{Concat}(SA_1(z), SA_2(z), \cdots, SA_{n_{head}}(z)) \cdot W_{\text{MSA}}.$$

Layer normalization (LN) [33] is applied before every block, and residual connections are applied after every block as in [34], [35]. The MLP contains two layers with a GELU non-linearity [36]. During both pre-training and fine-tuning, a classification head is attached to the embedding of the first element in the last layer, *i.e.*, z_L^0. The computation process of the transformer encoder can be expressed as,

$$\begin{aligned} z'_l &= \text{MSA}(\text{LN}(z_{l-1})) + z_{l-1}, \quad \text{for } l = 1, \ldots, L, \\ z_l &= \text{MLP}(\text{LN}(z'_l)) + z'_l, \quad \text{for } l = 1, \ldots, L, \\ y &= \text{LN}(z_L^0). \end{aligned} \tag{1}$$

B. STPT Pre-training

In this part, we pre-train our STPT model using self-supervised learning as is shown in Fig. 2(a). To accomplish this, we propose a novel self-supervised sub-trajectory similarity learning task. This task aims to unravel hidden relationships and commonalities among sub-trajectories nested within an entire trajectory. To accomplish this, we utilize the STPT model, which embeds sequences of trajectories. Post-STPT processing involves fragmenting trajectories into sub-trajectories, facilitating their comparison and analysis, thus enabling the learning and quantification of their similarity. In the pre-training phase, our objective is to enhance the model's ability to predict the source of sub-trajectories. To this end, given two distinct trajectories τ_i and τ_j from different sources i and j, we extract sub-trajectories from each of them and and get their representations denoted as γ_i and γ_j respectively. We then pair the extracted sub-trajectories and generate a label y_{real} based on if they are from the same source or not. Specifically, if a pair of sub-trajectory representations (γ_i, γ'_i) (or (γ_j, γ'_j)) are from the same source i (or j), the label y_{real} is 1; if otherwise (*i.e.*, (γ_i, γ_j)), the label y_{real} is 0. We thus introduce the objective function to pre-train our STPT as:

$$\min_{\theta} \ell(f^{pre}(\gamma_s, \gamma_{s'}, \theta), y_{real}), \tag{2}$$

where γ_s and $\gamma_{s'}$ represent pairs of sub-trajectory representations obtained from either distinct whole-day trajectories or consistent whole-day trajectories. With this training objective, our STPT model is able to learn representations that capture the characteristics and patterns of specific sub-trajectories within a full day's trajectory.

To achieve the pre-training objective, we introduce our STPT pre-training algorithm in Algorithm 1, where we iterate over the training data $\mathcal{T}$ for E training epochs and a batch size B. Within each epoch e and batch b, we select pairs of distinct trajectories τ_i and τ_j from $\mathcal{T}$ (line 3). These trajectories are fed into the STPT model to generate embeddings g_{ik} and g_{jk} for each GPS record p_k in τ_i and τ_j, respectively (line 4). We then flatten the embeddings g and order them based on sub-trajectory length, resulting in the creation of sub-trajectory representation sets $\{\gamma_i\}$ and $\{\gamma_j\}$ (line 5). Pairs of sub-trajectory representations, including both from distinct sources and from the same sources, *i.e.*, $(\gamma_s, \gamma_{s'})$, are constructed (line 6). By optimizing the pre-training loss function in Eq. 2 concerning the model parameters θ_b^e, we refine the model's ability to predict whether two sub-trajectories originate from the same source or not (line 7). Essentially, the model is trained to determine whether the second sub-trajectory in a pair originates from the same source as the first or if it belongs to a different source. By training on this task, our model becomes

Authorized licensed use limited to: Gordon Library WPI. Downloaded on May 15,2024 at 14:16:59 UTC from IEEE Xplore. Restrictions apply.

Algorithm 1 STPT Pre-training

Require: Training data $\mathcal{T} = \{\tau\}$; training epochs E; batch size B.
Ensure: Pre-trained $f_{\theta'}^{pre}$model parameters θ.
1: **for** $e = 1, 2, \ldots, E$ **do**
2: **for** $b = 1, 2, \ldots, /B$ **do**
3: Sample a pair of distinct trajectories τ_i and τ_j of length N from $\mathcal{T}$.
4: Feed τ_i and τ_j into STPT, to obtain the embedding $g_{ik} = G(p_{ik})$ and $g_{jk} = G(p_{jk})$ for each GPS record p_k in τ_i and τ_j, respectively, where $k = 1, 2, \cdots, N$.
5: Flatten the embedding g_k of each GPS record and order the set of the embeddings based on the sub-trajectory length to generate sub-trajectory representation sets $\{\gamma_i\}$ and $\{\gamma_j\}$.
6: Construct pairs of sub-trajectory representations $(\gamma_s, \gamma_{s'})$, where $\gamma_s \in \{\gamma_i\} \cup \{\gamma_j\}$ and $\gamma_{s'} \in \{\gamma_i\} \cup \{\gamma_j\}$.
7: Update the model parameters θ_b^e by optimizing Eq. 2 where γ_s and $\gamma_{s'}$ are pairs of sub-trajectory.
8: **end for**
9: **end for**
10: Return the pre-trained model f_θ^{pre} with updated parameter $\theta' = \theta_{E,|\mathcal{T}|/B}$.

adept at capturing the dependency and similarity between sub-trajectories within an entire day's trajectory. This enables us to gain deeper insights into the spatial-temporal patterns and relationships present in the trajectory data, leading to enhanced trajectory analysis and understanding.

C. STPT Fine-tuning

After pre-training our STPT model on HSTD, we extend its functionality by appending task-specific layers to adapt to various downstream tasks. STPT fine-tuning enables an effective transfer of knowledge from the STPT model, thereby enhancing the performance of subsequent spatial-temporal tasks. To fine-tune the models, we optimize a task-specific loss function in the general form as,

$$\min_{\theta,\phi} \mathbb{E}_{y,d\in\mathcal{D}}[\ell(f(d,\theta,\phi),y)]. \tag{3}$$

Here, θ and ϕ are parameters of our pre-trained STPT model and task-specific layers respectively. We use $f(\cdot,\theta,\phi)$ to denote the task-specific model with both the pre-trained STPT part and task-specific layers. $d \in \mathcal{D}$ is a training data instance from the fine-tuning training data set $\mathcal{D}$, and y is the label.

The generic fine-tuning algorithm, applicable across diverse tasks, is detailed in Algorithm 2. The algorithm iterates over the training data $\mathcal{D}$ for E epochs given a batch size B. Within each epoch, it samples a batch of B data samples as $\mathcal{D}_b^e$ from $\mathcal{D}$ (line 4). The sampled batch $\mathcal{D}_b^e$ is then fed into the STPT model to generate spatial-temporal representations. These representations then go through the task-specific layers to produce predictions (line 5). The parameters θ_b^e and ϕ_b^e, representing

Algorithm 2 STPT Fine-tuning

Require: Training data $\mathcal{D} = \{d\}$; training epochs E; batch size B.
Ensure: Task-specific fine-tuned θ and ϕ.
1: Initialize the STPT model in fine-tuning with pre-trained parameters θ and task-specific classifier parameters ϕ.
2: **for** $e = 1, 2, \ldots, E$ **do**
3: **for** $b = 1, 2, \ldots, |\mathcal{D}|/B$ **do**
4: Sample a batch of B training data $\mathcal{D}_b^e$ from $\mathcal{D}$.
5: Feed the sampled batch $\mathcal{D}_b^e$ to STPT and the task-specific layers.
6: Update θ_b^e and ϕ_b^e based on the objective in Eq. 3.
7: **end for**
8: **end for**

the pre-trained and task-specific parameters, respectively, are updated by minimizing the task-specific loss function in Eq 3 (line 6). The algorithm iterates through all the instances in the training data for E epochs, updating the parameters to optimize the task-specific prediction.

Next, we take two tasks, *i.e.*, trajectory classification and driving activity identification, as examples and illustrate how STPT adapts to downstream tasks.

Trajectory Classification. Trajectory classification refers to the task of categorizing trajectories into different predefined classes or categories based on their underlying characteristics or patterns. In our approach, we leverage the robust capabilities of the STPT model to effectively handle the task of trajectory classification. More specifically, a multi-class classifier is attached to the top of the STPT model's output embedding $\boldsymbol{r}$. Given a trajectory τ, we calculate the probability of it belonging to class c following:

$$P(c|\tau) = \text{softmax}(U^c \cdot \boldsymbol{r} + b^c).$$

In this equation, U^c and b^c represent the parameters of the classifier for class c. During fine-tuning, we borrow the parameters from the pre-trained spatial-temporal model as is shown in Fig. 2 (b). This approach allows us to effectively classify trajectories and gain insights into the underlying patterns and behaviors present in the trajectory data.

Driving Activity Identification. Another downstream task we consider is the distinction between 'seeking' and 'serving' segments within a taxi driver's trajectory. 'Seeking' segments correspond to the portions of the trajectory where a taxi driver is looking for a customer or heading towards a customer's location; whereas 'serving' segments correspond to when the driver is transporting a customer. This task is critical for ride-hailing services as it directly impacts the efficiency of taxi operations. To adapt the STPT model to this task, we attach a binary classifier to the STPT's output embedding $\boldsymbol{r}$. Given a trajectory segment s, the probability of it being a 'serving' segment or a 'seeking' segment is calculated as,

$$P(\text{'serving'}|s) = \sigma(U^s \cdot \boldsymbol{r} + b^s),$$
$$P(\text{'seeking'}|s) = 1 - P(\text{'serving'}|s).$$

Authorized licensed use limited to: Gordon Library WPI. Downloaded on May 15,2024 at 14:16:59 UTC from IEEE Xplore. Restrictions apply.

where U^s and b^s are the parameters of the classifier, and $\sigma(\cdot)$ denotes the sigmoid function. During fine-tuning, we optimize the binary cross-entropy loss between the predicted and true labels ('seeking' or 'serving'). Utilizing STPT as the backbone model, we add a task-specific binary classifier during fine-tuning, designed to address the seeking or serving task and customized for the task requirements.

IV. Experiments

In this section, we evaluate the performances of the pre-trained model STPT using the taxi GPS dataset collected in Shenzhen, China in July 2016, and two downstream tasks i.e., multi-class classification task and seeking *vs*. serving trajectory distinction task. We compare with other baselines to demonstrate that (i) our proposed pre-trained model STPT is able to improve the prediction accuracy of downstream tasks, and (ii) it is able to enhance efficiency in fine-tuning tasks, resulting in accelerated processing speed.

A. Data Description and Preparation

Our work takes two urban data sources as input, including (1) taxi GPS trajectory data and (2) road map data. Both datasets were collected in Shenzhen, China in 2016.

Taxi trajectory data is generated from 20,000 unique taxis in Shenzhen, China, from July to December, half-year 2016. Each taxi is equipped with a GPS unit that records one GPS point approximately every 40 seconds. The dataset collects about 51 million GPS records each day, with each record consisting of five primary data fields: latitude, longitude, a unique taxi identifier, a timestamp, and a driving status mode. The driving status mode is to signify whether a passenger is aboard, with 1 representing a passenger on board and 0 indicating otherwise.

Road map data provides the layout of Shenzhen, covering the area between 22.44° to 22.87° latitude and 113.75° to 114.63° longitude. The data is sourced from OpenStreetMap [37], comprising approximately 21,000 roads across six levels.

Map gridding and time quantization. To ensure the anonymity of the data and mitigate the potential for re-identification, we discretize the trajectories using data anonymization techniques. Specially, we adopt a standard quantization method, partitioning the Shenzhen area into grid cells with equal side-lengths of $0.01°$ in both latitude and longitude [5], [38], [39]. This approach safeguards individual identities while maintaining the usefulness of the data. Fig. 4 shows the gridding result in Shenzhen, China, where grids colored in dark grey are inaccessible as they are on the sea.

After filtering out cells located in the ocean, unreachable from the city, and other irrelevant cells, we have a total of 1,934 valid cells. We further divide each day into five-minute intervals for a total of 288 intervals per day. These intervals are denoted as $I = \{\tilde{t}_k\}$, with $1 \leqslant k \leqslant 288$. A spatial-temporal region r is a pair of a grid cell g and a time interval $\tilde{t}_k$ with the status sta, which is set to 0 if there is no passenger onboard, and to 1 if the driver is serving a passenger. Each GPS record denoted as $p = \langle lat, lng, t, sta \rangle$, can be mapped to an aggregated state $S = \langle g, \tilde{t}_k, sta \rangle$. Consequently, a trajectory of agent a can be transformed into sequences of spatial-temporal regions, represented as $\tau = \{a, \langle r_1, r_2, ..., r_n \rangle\}$.

Fig. 4: Map gridding demonstration.

B. Experiment Setups

We implement our deep neural network on Python 3.10.9 with Pytorch version 1.13.1. Our experiments run on a virtual machine running Linux-ubuntu $20.04 - x86\_64$ with 3 GPUs, NVIDIA A100-SXM4-80GB. We implement standard back-propagation on feed-forward networks using the adaptive moment estimation (Adam) method with $\beta_1 = 0.9$, $\beta_2 = 0.999$. Our mini-batch size is 16, learning rate is 0.0001 for both pre-traing task and fine-tuning tasks.

1) Pre-training: In the pre-training phase, we focus on a sub-trajectory similarity learning task using the STPT model. We randomly sample 500 drivers from the dataset, which spanned a period of half a year from July 1st to December 31st. Each driver's full-day trajectories are employed for pre-training. More specifically, we randomly pair the entire day's trajectories of two different drivers, denoted as τ^1 and τ^2, culminating in the generation of 100,000 pairs of trajectories from the selected 500 drivers.

2) Fine-tuning: Following the pre-training phase, we employ the learned representations for fine-tuning on labeled data. We supplement the STPT model with a randomly initialized output layer to facilitate character predictions. It's important to note that the dataset utilized for this fine-tuning process remains distinct and unseen during the pre-training phase of the STPT model. For the multi-classification trajectory classification model where the input is identical to that of the pre-trained model (i.e., whole-day trajectories), we incorporate the transformer layers and the positional embedding representation. However, for the seek-and-serve distinction model, given the typically shorter span of seeking-and-serving trajectories (between 10 to 60 steps), we decide against utilizing the positional embedding from the pre-trained model.

For multi-class trajectory classification model, we select 8 distinct drivers from the dataset and employed their full-day trajectories, τ^1, in our model. In total, we gather 90 days of trajectories for these drivers. To facilitate model development, we split the trajectory dataset into three sets: 60 days for training, 10 days for validation, and 20 days for testing.

For seeking *vs*. serving activity identification model, we focus on a specific sub-dataset by selecting 20 drivers from July

Authorized licensed use limited to: Gordon Library WPI. Downloaded on May 15,2024 at 14:16:59 UTC from IEEE Xplore. Restrictions apply.

2016. We extract their seeking and serving trajectories, denoted as τ^s and τ^d respectively, Each of them is a sequence of GPS records without status, $\tau_s, \tau_d = \{a, \langle r_1, r_2, ..., r_n \rangle\}$. The dataset includes 2,787 seeking and 2,787 serving trajectories, split into three subsets: 40% for training, 30% for validation, and 30% for testing. The aim is to train the model for effective seeking vs. serving behavior differentiation.

C. Baseline Methods

In this section, we describe the baselines for comparison. These baselines are configured with comparable parameters for a fair evaluation.

- **Transformer Encoder [32]** is a powerful deep-learning model commonly used for a wide range of tasks, mirrors the architecture described in Section III. The Transformer introduces an architecture leveraging self-attention mechanisms. By efficiently capturing dependencies among elements of input sequences, Transformers excel at modeling long-range dependencies and have consistently achieved state-of-the-art performance in various NLP and CV tasks.
- **Convolutional Neural Network (CNN) [40]** is a highly effective deep learning model extensively utilized in various tasks. In our approach, we employ a CNN to process the trajectories. By applying convolutional layers, CNN captures local patterns and relationships within the trajectory data. This enables the model to leverage spatial and temporal information for accurate predictions.
- **Long short-term memory networks (LSTM) [41]** are a type of recurrent neural network (RNN) architecture commonly used as baselines in various sequential data analysis tasks. They can handle context and dependencies crucial for accurate predictions by preserving and selectively updating information over extended periods, offering enhanced capabilities in understanding sequential data.

D. Evaluation Results

In this section, we present the evaluation results for our two downstream tasks. We compare the performance of $\text{STPT}_{\text{BASE}}$ and $\text{STPT}_{\text{LARGE}}$ models with other comparable baseline models using average F1-score, recall, precision, and accuracy metrics of each class, ensuring a fair comparison. The results are presented in two separate tables, one for the $\text{STPT}_{\text{BASE}}$ model and the other one for the $\text{STPT}_{\text{LARGE}}$ model, considering their respective parameter sizes.

For multi-class trajectory classification models, we conduct an evaluation of the performance of both $\text{STPT}_{\text{BASE}}$ and $\text{STPT}_{\text{LARGE}}$ models, comparing them to baselines with comparable parameters. The evaluation results, presented in Table IIa and Table IIb, demonstrate the superior performance of our models across multiple metrics, including average F1-score, recall, precision, and accuracy. In the base model configuration, $\text{STPT}_{\text{BASE}}$ attains an accuracy, precision, recall, and F1-score of 0.8125, 0.8148, 0.8125, and 0.7971 respectively, significantly outpacing other base models such as $\text{Transformer}_{\text{BASE}}$, CNN_{BASE}, and $\text{LSTM}_{\text{BASE}}$. The large model $\text{STPT}_{\text{LARGE}}$ consistently outperforms others with an accuracy of 0.83125, recall of 0.8312, and an F1-score of 0.8298. While $\text{CNN}_{\text{LARGE}}$ had the highest precision of 0.8861, $\text{STPT}_{\text{LARGE}}$ held its ground with a competitive 0.8326. Conversely, both large and base LSTM models fell short across all metrics, indicating their limitations in this specific task. Notably, their increased complexity does not enhance but instead seemed to diminish predictive performance. In terms of efficiency, LSTM models trailed behind, taking longer to train than all other methods, further highlighting the superior effectiveness and efficiency of our STPT models.

TABLE II: Comparison of Multi-class Classification Results

(a) Average F1, recall, precision, and accuracy on the real-world dataset and comparison of $\text{STPT}_{\text{BASE}}$ with baselines with comparable parameters for trajectory classification

Methods	Accuracy	Precision	Recall	F_1 Score
$\text{STPT}_{\text{BASE}}$	**0.8125**	**0.8148**	**0.8125**	**0.7971**
$\text{Transformer}_{\text{BASE}}$	0.7375	0.6979	0.7375	0.7091
CNN_{BASE}	0.74375	0.7696	0.7437	0.7102
$\text{LSTM}_{\text{BASE}}$	0.63125	0.5592	0.6312	0.5675

(b) Average F1, recall, precision, and accuracy on the real-world dataset and comparison of $\text{STPT}_{\text{LARGE}}$ with baselines with comparable parameters for trajectory classification

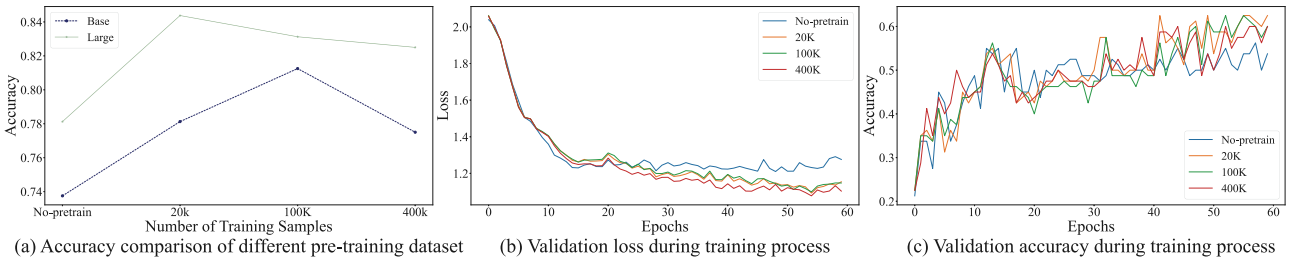
Methods	Accuracy	Precision	Recall	F_1 Score
$\text{STPT}_{\text{LARGE}}$	**0.83125**	0.8326	**0.8312**	**0.8298**
$\text{Transformer}_{\text{LARGE}}$	0.78125	0.8564	0.7812	0.7519
$\text{CNN}_{\text{LARGE}}$	0.78125	**0.8861**	0.7812	0.7726
$\text{LSTM}_{\text{LARGE}}$	0.44375	0.2994	0.4437	0.3355

For seeking *vs*. serving activity identification model, same as the multi-class classification task, we evaluate the performance of our models across multiple metrics. Based on the results presented in Tables IIIa and IIIb, it can be concluded that the $\text{STPT}_{\text{BASE}}$ and $\text{STPT}_{\text{LARGE}}$ models outperform the baselines in the seek and serve distinction task. The $\text{STPT}_{\text{BASE}}$ model achieves an accuracy of 0.7711, precision of 0.7849, recall of 0.7464, and an F1 score of 0.7652, while the $\text{STPT}_{\text{LARGE}}$ model achieves an accuracy of 0.7788, precision of 0.7920, recall of 0.7560, and an F1 score of 0.7736. Comparing these results with the baselines, both $\text{STPT}_{\text{BASE}}$ and $\text{STPT}_{\text{LARGE}}$ consistently demonstrate superior performance in terms of accuracy, precision, and F1 score. Although the LSTM baseline achieves a higher recall, its accuracy precision and F1 score are all lower than 0.7, indicating poor overall performance, so the overall performance of the STPT models showcases their effectiveness in distinguishing between seeking and serving actions. Therefore, it can be concluded that the utilization of pre-trained $\text{STPT}_{\text{BASE}}$ and $\text{STPT}_{\text{LARGE}}$ models prove to be highly beneficial for the downstream task of seeking *vs*. serving activity identification model.

E. Ablation Study

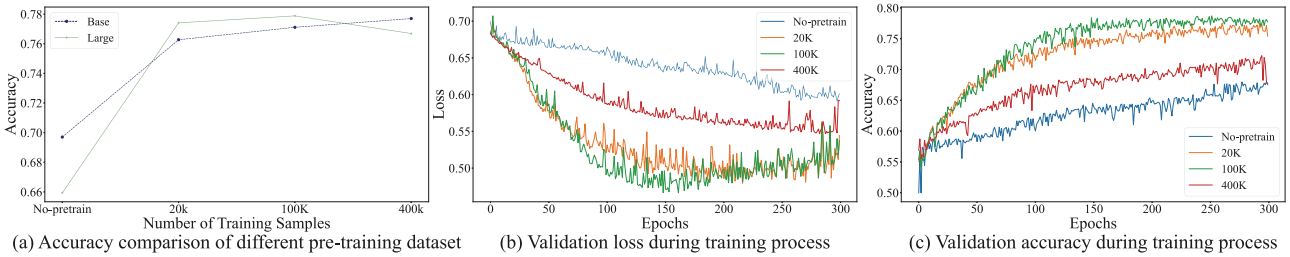
In the ensuing section, we delineate an ablation study centered around three significant components. To begin, we train our model utilizing random subsets of trajectory pairs extracted from authentic data to compare if the dataset size

Authorized licensed use limited to: Gordon Library WPI. Downloaded on May 15,2024 at 14:16:59 UTC from IEEE Xplore. Restrictions apply.



(a) Accuracy comparison of different pre-training dataset (b) Validation loss during training process (c) Validation accuracy during training process

Fig. 5: Comparison of multi-class trajectory classification performance: without Pre-training *vs*. different pre-training sizes.



(a) Accuracy comparison of different pre-training dataset (b) Validation loss during training process (c) Validation accuracy during training process

Fig. 6: Comparison of seeking *vs*. serving identification model without pre-training *vs*. with different pre-training sizes.

TABLE III: Comparison of Seek and Serve Distinction Results

(a) Average F1, recall, precision, and accuracy on the real-world dataset and comparison of $\text{STPT}_{\text{BASE}}$ with baselines with comparable parameters for seek and serve distinction model

Methods	Accuracy	Precision	Recall	F_1 Score
$\text{STPT}_{\text{BASE}}$	**0.7711**	**0.7849**	0.7464	**0.7652**
$\text{Transformer}_{\text{BASE}}$	0.6970	0.7018	0.6842	0.6929
CNN_{BASE}	0.7436	0.7375	0.7560	0.7466
$\text{LSTM}_{\text{BASE}}$	0.5852	0.5590	**0.8050**	0.6598

(b) Average F1, recall, precision, and accuracy on the real-world dataset and comparison of $\text{STPT}_{\text{LARGE}}$ with baselines with comparable parameters for seek and serve distinction model

Methods	Accuracy	Precision	Recall	F_1 Score
$\text{STPT}_{\text{LARGE}}$	**0.7788**	**0.7920**	0.7560	**0.7736**
$\text{Transformer}_{\text{LARGE}}$	0.6593	0.6377	0.7368	0.6837
$\text{CNN}_{\text{LARGE}}$	0.7029	0.7070	0.6926	0.6997
$\text{LSTM}_{\text{LARGE}}$	0.5852	0.5511	**0.9163**	0.6882

of the pre-trained model impacts the performance results. In particular, we employ three diverse dataset sizes: 20,000 pairs of comprehensive daily trajectories from 100 unique drivers, 100,000 pairs from 500 drivers, and a comprehensive set of 400,000 pairs from 1,000 drivers, which allow us to scrutinize the impact of successively increasing the dataset size during the pre-training phase. Moving to another aspect, we undertake a comparative study of the influence of model parameters on the results. Our final focus is on determining whether pre-training could accelerate the convergence in downstream tasks, thereby diminishing the computational overhead. Our results are shown in Fig. 5 and Fig. 6.

1) Influence of Varying Pre-training Dataset Sizes on Model Performance and Training Efficiency.: In this part, we conduct an ablation study to examine the impact of pre-training dataset sizes on the prediction accuracy and training efficiency of both tasks. For the multi-class classification model, we evaluate the impact of varying pre-training dataset sizes, and the results are presented in Fig. 5 (a). Surprisingly, increasing the size of the pre-training dataset does not show a significant impact on the prediction performance of both the $\text{STPT}_{\text{BASE}}$ and $\text{STPT}_{\text{LARGE}}$ models. However, it is worth noting that pre-trained models outperform the results obtained without any pre-training. Fig. 5 (b) shows that pre-training does not necessarily accelerate the convergence of the downstream task training. For the seeking vs. serving identification model, as illustrated in Figure 6 (a), we observe that increasing the size of the pre-training dataset has a minimal impact on the prediction performance of both $\text{STPT}_{\text{BASE}}$ and $\text{STPT}_{\text{LARGE}}$ models. When examining training speed in Figures 6 (b) and (c), we note that the pre-trained model converges more rapidly, exhibiting lower loss and higher accuracy compared to the non-pre-trained model. This finding underscores the computational efficiency gained through pre-training. Analyzing the training and validation loss patterns reveals that the effect of pre-training on convergence speed varies depending on the specific downstream task. While pre-training may not expedite convergence in all cases, it consistently improves the validation accuracy.

2) Comparison of Different Parameters in the Model: As described in Section III, we conduct pre-training using $\text{STPT}_{\text{BASE}}$ and $\text{STPT}_{\text{LARGE}}$ models, which correspond to different sizes of the pre-trained model. Fig. 5 (a) presents the results for the multi-class classification model, we observe that models with a greater number of layers of transformer blocks (i.e., more parameters) outperform models with fewer param-

Authorized licensed use limited to: Gordon Library WPI. Downloaded on May 15,2024 at 14:16:59 UTC from IEEE Xplore. Restrictions apply.

eters. However, for the seek *vs*. serve depreciation model, as depicted in Fig. 6, we obtain similar results regardless of the parameter size. Evaluating these results, the impact of parameter size varies across the two models. In the multi-class classification model, increased parameters improved performance, indicating the benefits of added complexity. However, in the seek *vs*. serve model, parameter size does not significantly affect performance, suggesting a robustness to parameter variations.

V. Related Work

Large pre-trained models. Large pre-trained models, including transformer-based large language models, have had a significant impact on the field of NLP problems [23], [25]. Models like GPT-3 [21] and PaLM [19] have demonstrated remarkable performance on NLP benchmarks and natural language generation tasks, leading to their widespread adoption across various industries. Similarly, in automatic speech recognition (ASR) and speech synthesis, large-scale pre-trained models such as DeepSpeech2 [42] and wav2vec 2.0 [43] have pushed the boundaries of speech recognition, enabling applications in transcription services, voice assistants, and voice-controlled systems. Moreover, large pre-trained vision models like the VIT [31], DeiT [44], BEiT [45], and MAE [46] have revolutionized CV, achieving impressive results in image recognition and demonstrating innovative capabilities. The availability of these large-scale pre-trained models has opened up new possibilities for enhancing human-computer interaction, and various applications in speech and vision domains. However, the availability of comprehensive research specifically focused on spatial-temporal data is limited. [30] puts forth a pre-training model specifically designed to learn representations for individual locations, focusing on location-related tasks like next location prediction. However, this location-centered approach falls short in capturing the intrinsic information of mobility patterns and, thus is inadequate to address trajectory-level downstream tasks, *e.g*., trajectory classification, driving activity identification, and more. Therefore, we introduce the STPT model, a novel approach that leverages a distinctive self-supervised pre-training task aimed at capturing both the similarities and distinctions within HSTD. Our objective is to acquire comprehensive generic spatial-temporal representations in HSTD.

Urban computing. Urban computing is a broad research field that combines urban sensing, data management, and data analysis on urban data [10], [47]–[49]. A group of works focus on human decision analysis using human behavior data. Especially in taxi operation management, there has been significant research on two main areas: dispatching and passenger seeking. Several studies have focused on optimizing taxi dispatching strategies [50], [51], aiming to improve the efficiency of matching taxis with passenger requests. Additionally, research has been conducted on understanding taxi-seeking behavior [16], [18], [52], they seek to identify the best actionable solution for improving the performance of taxi drivers. Besides, a group considers the benefits of passengers. They [3], [5], [53], [54] focus on driver identification, they attempted to match the identities of human agents only from the observed trajectory data and detect the abnormal driver behaviors that enhance the safety of passengers. A group of works that focuses on estimating urban traffic to help reduce traffic congestion and provide insights for urban planning [8], [9], [55]. In this paper, our main focus is to develop a self-supervised pre-training for robust and generic spatial-temporal representations model that can be fine-tuned and applied to diverse downstream applications in urban computing.

VI. Conclusion

In this study, we address HSTD challenges by introducing a pre-training framework (STPT) that incorporates a novel self-supervised learning task. This task plays a crucial role in capturing the complex spatial and temporal patterns present in HSTD. By leveraging a spatial-temporal pre-trained model trained on HSTD, we demonstrate significant improvements in two fine-tuning tasks: trajectory classification and seeking *vs*. serving trajectory identification. The evaluation results on real-world datasets highlight the effectiveness of the pre-trained model in accurately representing HSTD and enhancing downstream tasks. Our proposed STPT, with its unique self-supervised learning task, offers a promising approach to overcoming the limitations of disparate models and datasets and providing a robust and generic representation of HSTD for diverse applications in urban computing and beyond.

VII. Acknowledgements

Mingzhi Hu and Yanhua Li were supported in part by NSF grants IIS-1942680 (CAREER), CNS-1952085 and DGE-2021871. Yiqun Xie was supported in part by NSF awards 2105133, 2126474, and 2147195. Xiaowei Jia was supported by NSF award IIS-2147195. We would also like to thank Andrew Chen for his valuable contributions to the experiments and for composing the initial draft of this paper.

References

[1] 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 *ITSC*, IEEE, 2016.

[2] 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.

[3] T. Kieu, B. Yang, C. Guo, and C. S. Jensen, "Distinguishing trajectories from different drivers using incompletely labeled trajectories," in *CIKM*, 2018.

[4] M.-h. Oh and G. Iyengar, "Sequential anomaly detection using inverse reinforcement learning," in *SIGKDD*, 2019.

[5] H. Ren, M. Pan, Y. Li, X. Zhou, and J. Luo, "St-siamesenet: Spatio-temporal siamese networks for human mobility signature identification," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1306–1315, 2020.

[6] Uber, "Uber services," 2023.

[7] Lyft, "Lyft services," 2023.

[8] Y. Zhang, Y. Li, X. Zhou, X. Kong, and J. Luo, "Strans-gan: Spatially-transferable generative adversarial networks for urban traffic estimation," in *2022 IEEE International Conference on Data Mining (ICDM)*, pp. 743–752, IEEE, 2022.

[9] Y. Zhang, Y. Li, X. Zhou, and J. Luo, "Mest-gan: Cross-city urban traffic estimation with me ta s patial-t emporal g enerative a dversarial n etworks," in *2022 IEEE International Conference on Data Mining (ICDM)*, pp. 733–742, IEEE, 2022.

Authorized licensed use limited to: Gordon Library WPI. Downloaded on May 15,2024 at 14:16:59 UTC from IEEE Xplore. Restrictions apply.

[10] 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 International Conference on Knowledge Discovery & Data Mining*, pp. 984–992, 2018.

[11] A. Zonoozi, J.-j. Kim, X.-L. Li, and G. Cong, "Periodic-crn: A convolutional recurrent model for crowd density prediction with recurring periodic patterns.," in *IJCAI*, pp. 3732–3738, 2018.

[12] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," *arXiv preprint arXiv:1506.04214*, 2015.

[13] Z. Cui, R. Ke, Z. Pu, and Y. Wang, "Deep bidirectional and unidirectional lstm recurrent neural network for network-wide traffic speed prediction," *arXiv preprint arXiv:1801.02143*, 2018.

[14] H. Yu, Z. Wu, S. Wang, Y. Wang, and X. Ma, "Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks," *Sensors*, vol. 17, no. 7, p. 1501, 2017.

[15] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: a deep learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2014.

[16] X. Zhang, Y. Li, X. Zhou, and J. Luo, "Unveiling taxi drivers' strategies via cgail: Conditional generative adversarial imitation learning," in *2019 IEEE International Conference on Data Mining (ICDM)*, pp. 1480–1485, IEEE, 2019.

[17] X. Zhang, Y. Li, X. Zhou, Z. Zhang, and J. Luo, "Trajgail: Trajectory generative adversarial imitation learning for long-term decision analysis," in *2020 IEEE International Conference on Data Mining (ICDM)*, pp. 801–810, IEEE, 2020.

[18] 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 International Conference on Data Mining*, pp. 783–791, SIAM, 2019.

[19] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, *et al.*, "Palm: Scaling language modeling with pathways," *arXiv preprint arXiv:2204.02311*, 2022.

[20] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[21] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[22] OpenAI, "ChatGPT: Large-scale Language Model for Conversational AI." https://openai.com, 2021. Version GPT-3.5.

[23] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[24] A. M. Dai and Q. V. Le, "Semi-supervised sequence learning," *Advances in neural information processing systems*, vol. 28, 2015.

[25] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding with unsupervised learning," 2018.

[26] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," *arXiv preprint arXiv:1801.06146*, 2018.

[27] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "Squad: 100,000+ questions for machine comprehension of text," *arXiv preprint arXiv:1606.05250*, 2016.

[28] M. E. Peters, W. Ammar, C. Bhagavatula, and R. Power, "Semi-supervised sequence tagging with bidirectional language models," *arXiv preprint arXiv:1705.00108*, 2017.

[29] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[30] Y. Lin, H. Wan, S. Guo, and Y. Lin, "Pre-training context and time aware location embeddings from spatial-temporal trajectories for user next location prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pp. 4241–4248, 2021.

[31] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[33] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[34] Q. Wang, B. Li, T. Xiao, J. Zhu, C. Li, D. F. Wong, and L. S. Chao, "Learning deep transformer models for machine translation," *arXiv preprint arXiv:1906.01787*, 2019.

[35] A. Baevski and M. Auli, "Adaptive input representations for neural language modeling," *arXiv preprint arXiv:1809.10853*, 2018.

[36] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," *arXiv preprint arXiv:1606.08415*, 2016.

[37] OpenStreetMap contributors, "Planet dump retrieved from https://planet.osm.org ." https://www.openstreetmap.org, 2017.

[38] 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 *2015 IEEE 31st international conference on data engineering*, pp. 1376–1387, IEEE, 2015.

[39] Y. Li, M. Steiner, J. Bao, L. Wang, and T. Zhu, "Region sampling and estimation of geosocial data with dynamic range calibration," in *2014 IEEE 30th International Conference on Data Engineering*, pp. 1096–1107, IEEE, 2014.

[40] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[41] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[42] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen, *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International conference on machine learning*, PMLR, 2016.

[43] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Advances in neural information processing systems*, vol. 33, pp. 12449–12460, 2020.

[44] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *International conference on machine learning*, pp. 10347–10357, PMLR, 2021.

[45] H. Bao, L. Dong, S. Piao, and F. Wei, "Beit: Bert pre-training of image transformers," *arXiv preprint arXiv:2106.08254*, 2021.

[46] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[47] 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 Transactions on Intelligent Systems and Technology (TIST)*, vol. 8, no. 6, pp. 1–24, 2017.

[48] C. Liu, K. Deng, C. Li, J. Li, Y. Li, and J. Luo, "The optimal distribution of electric-vehicle chargers across a city," in *2016 IEEE 16th International Conference on Data Mining (ICDM)*, 2016.

[49] M. Qu, H. Zhu, J. Liu, G. Liu, and H. Xiong, "A cost-effective recommender system for taxi drivers," in *SIGKDD*, 2014.

[50] S. Ma, Y. Zheng, and O. Wolfson, "T-share: A large-scale dynamic taxi ridesharing service," in *2013 IEEE 29th International Conference on Data Engineering (ICDE)*, pp. 410–421, IEEE, 2013.

[51] N. J. Yuan, Y. Zheng, L. Zhang, and X. Xie, "T-finder: A recommender system for finding passengers and vacant taxis," *IEEE Transactions on knowledge and data engineering*, vol. 25, no. 10, pp. 2390–2403, 2012.

[52] Y. Zhang, Y. Li, X. Zhou, Z. Zhang, and J. Luo, "Stm-gail: Spatial-temporal meta-gail for learning diverse human driving strategies," in *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*, pp. 208–216, SIAM, 2023.

[53] E. Cheung, A. Bera, E. Kubin, K. Gray, and D. Manocha, "Identifying driver behaviors using trajectory features for vehicle navigation," in *2018 IEEE/RSJ IROS*, IEEE, 2018.

[54] M. Hu, X. Zhang, Y. Li, X. Zhou, and J. Luo, "St-ifgsm: Enhancing robustness of human mobility signature identification model via spatial-temporal iterative fgsm," in *the 29th SIGKDD conference on Knowledge Discovery and Data Mining (KDD 2023)*, 2023.

[55] E. Toto, E. A. Rundensteiner, Y. Li, R. Jordan, M. Ishutkina, K. Claypool, J. Luo, and F. Zhang, "Pulse: A real time system for crowd flow prediction at metropolitan subway stations," in *ECML PKDD*, pp. 112–128, Springer, 2016.

Authorized licensed use limited to: Gordon Library WPI. Downloaded on May 15,2024 at 14:16:59 UTC from IEEE Xplore. Restrictions apply.