You Are How You Drive: Peer and Temporal-Aware Representation Learning for Driving Behavior Analysis

Pengyang Wang*
Missouri Univ. of Sci. and Tech.
Missouri, USA
pwqt3@mst.edu

Yanjie Fu*†
Missouri Univ. of Sci. and Tech.
Missouri, USA
fuyan@mst.edu

Jiawei Zhang
Florida State University
Florida, USA
jzhang@cs.fsu.edu

Pengfei Wang
CNIC, Chinese Academy of Sciences
Beijing, China
wpf@cnic.cn

Yu Zheng
Urban Computing Business Unit,
JD Finance
Beijing, China
msyuzheng@hotmail.com

Charu Aggarwal
IBM Research AI
New York, USA
charu@us.ibm.com

ABSTRACT

Driving is a complex activity that requires multi-level skilled operations (e.g., acceleration, braking, turning). Analyzing driving behavior can help us assess driver performances, improve traffic safety, and, ultimately, promote the development of intelligent and resilient transportation systems. While some efforts have been made for analyzing driving behavior, existing methods can be improved via representation learning by jointly exploring the peer and temporal dependencies of driving behavior. To that end, in this paper, we develop a **P**eer and **T**emporal-**A**ware **R**epresentation **L**earning based framework (PTARL) for driving behavior analysis with GPS trajectory data. Specifically, we first detect the driving operations and states of each driver from GPS traces. Then, we derive a sequence of multi-view driving state transition graphs from the driving state sequences, in order to characterize a driver's driving behavior that varies over time. In addition, we develop a peer and temporal-aware representation learning method to learn a sequence of time-varying yet relational vectorized representations from the driving state transition graphs. The proposed method can simultaneously model both the graph-graph peer dependency and the current-past temporal dependency in a unified optimization framework. Also, we provide effective solutions for the optimization problem. Moreover, we exploit the learned representations of driving behavior to score driving performances and detect dangerous regions. Finally, extensive experimental results with big trajectory data demonstrate the enhanced performance of the proposed method for driving behavior analysis.

*These two authors contributed equally to this work.
†Corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
KDD '18, August 19–23, 2018, London, United Kingdom
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5552-0/18/08…$15.00
https://doi.org/10.1145/3219819.3219985

CCS CONCEPTS

• **Information systems** → **Data mining**; *Location based services*;

KEYWORDS

Driving Behavior Analysis, Spatio-temporal Graphs, Representation Learning

ACM Reference Format:
Pengyang Wang*, Yanjie Fu*†, Jiawei Zhang, Pengfei Wang, Yu Zheng, and Charu Aggarwal. 2018. You Are How You Drive: Peer and Temporal-Aware Representation Learning for Driving Behavior Analysis. In *Proceedings of The 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18).* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3219819.3219985

1 INTRODUCTION

Driving behavior is a complex activity that requires multi-level skilled operations, such as acceleration, deceleration, keeping constant speed, turning left, turning right, and moving straight. Analyzing driving behavior can help us assess driver performances, enhance traffic safety, and, ultimately, promote the development of intelligent and resilient transportation systems to enable many important applications, such as monitoring drivers, vehicles, and roads, providing early warning and driving assistance, and enhancing driving comfort and energy saving. In this paper, we study the problem of learning to profile driving behavior with applications to transportation safety.

Prior studies in driving behavior analysis can be categorized into: (i) descriptive analysis, in which transportation experts define measurements (e.g., harsh or frequent acceleration/braking, sharp turn, acceleration before turn) based on transportation theory to describe driving behavior [5]; (ii) predictive analysis, in which researchers mine the patterns from driving data and apply machine learning models (e.g., SVM, naive Bayesian, etc.) to predict risky scores [36]; (iii) causal analysis, in which researchers identify the causal factors of driving behavior and explain how these factors influence road safety [22]. Moreover, there are system-related studies that develop sensing systems to collect and analyze driving behavior [12]. However, previous studies have some limits. For example, some studies are based on biased and expensive data source, e.g., self-reported survey. Some studies empirically define descriptive variables with the help of domain experts, and, thereby, are lack of generalization capability when dealing with big and noisy driving data. Some studies mainly focus on analyzing coarse-grained (e.g., user-group-level or region-level) driving behavior, rather than individual-level driving behavior.

Indeed, the increasingly pervasiveness of GPS sensors has accumulated large-scale driving behavior data. And the emergence of representation learning provides great potential for automated behavior profiling. It is naturally promising to combine high-resolution widely-available GPS trajectories and representation learning for driving behavior analysis. However, two unique challenges arise in achieving this goal. First, the complex driving behavior needs to be effectively identified, quantified, and summarized in order to enrich the applicability of representation learning algorithms. Second, we need to jointly model the peer and temporal dependencies for learning effective representations of driving behavior. We outline how we tackle these two main challenges next.

The first challenge is that GPS traces (e.g., time, latitude, longitude) encode the driving operations, states, and styles in a semantically-implicit way, which jeopardizes the applicability of representation learning. Therefore, it highly necessitates a novel method to transform GPS traces into an appropriate structure that can effectively characterize driving activities and corresponding spatio-temporal dynamics. To address the challenge, we analogize driving behavior as a sequence of state transition graphs, and develop a three-step characterization method. To begin with, we identify two types of driving operations: (i) speed-related (i.e., acceleration, deceleration, constant speed) and (ii) direction-related (i.e., turning left, turning right, move straight) from a GPS trajectory, and generate a sequence of driving operations for each driver. Later, we define a driving state as a two-tuple combination that includes a speed operation status and a direction operation status, and extract a sequence of driving states. Lastly, to reduce the possible impacts of outliers, which might be generated by small sensor data errors, we derive multi-view driving state transition graphs (i.e., the transition probability and transition duration of driving states) to characterize driving behavior, and obtain a sequence of driving state transition graphs as the inputs of representation learning.

Second, after analyzing large-scale driving data, we identify two dependencies of driving state transition graphs: (i) *peer dependency*: if two driving state transition graphs are structurally similar, then the embeddings of the two graphs are similar in the latent feature space; (ii) *temporal dependency*: the embedding of a driving state transition graph not just depends on the driving operations at the current time period, but also has correlation with the previous ones. It thus is important to model the coupling of both peer and temporal dependencies in representation learning. Therefore, we develop a **P**eer and **T**emporal-**A**ware **R**epresentation **L**earning framework (PTARL) that can jointly model the graph-graph peer dependency across drivers, as well as the current-past temporal dependency within a driver, in representation learning. The proposed method can learn a sequence of time-varying yet relational vectorized representations from the driving state transition graphs, using a widely-available GPS data source and with very limited knowledge of surrounding conditions.

Along these lines, in this paper, we develop a peer and temporal-aware representation learning based analytic framework for driving behavior analysis using GPS traces. Specifically, we first construct a sequence of multi-view driving state transition graphs from GPS traces to characterize the dynamic driving behavior of each driver. Besides, we identify the graph-graph peer and current-past temporal dependencies of driving behavior, and incorporate the modeling of the peer and temporal dependencies into a unified Auto-Encoder based optimization framework. Also, we provide effective methods for the optimization problem. As applications, we exploit the learned representations of driving behavior for prediction and historical assessment, and risky region detection. Finally, we conduct extensive experiments to demonstrate the enhanced performance of the proposed method with real-world vehicle GPS traces.

2 PRELIMINARIES

We first introduce some important definitions and the problem statement, and then present an overview of the proposed method.

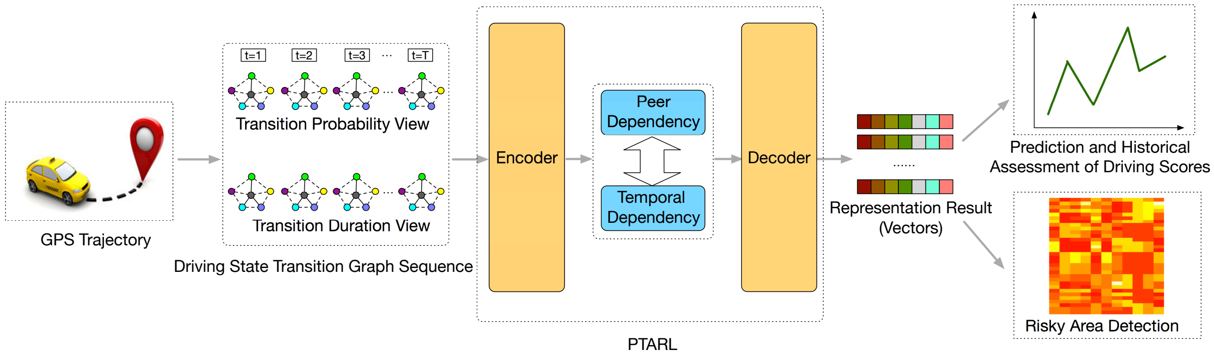
2.1 Definitions and Problem Statement

Definition 2.1. **Driving Operation.** Driving operations are defined as a set of activities and steps that a driver operates when driving a vehicle, according to the driver's personal judgment, experience and skills. Since a moving object can be characterized by speed and direction, we similarly categorize driving operations into (i) speed-related operations (i.e., acceleration, deceleration, constant speed) and (ii) direction-related operations (i.e., turning left, turning right, moving straight). The speed-related operations show how a driver operates the clutch pedal, gas pedal, and brake pedal of a vehicle. The direction-related operations show how a driver operates the steering wheel of a vehicle. The driving operations can be detected from GPS traces.

Definition 2.2. **Driving State.** A driving state concerns the way that a vehicle moves at a specific time point or in a small time window. In other words, a driving state of a vehicle contains both the speed status (i.e., acceleration, deceleration, constant speed) and the direction status (i.e., turning left, turning right, moving straight) of a vehicle. For instance, a driving state example of a car can be <constant speed, moving straight>.

Definition 2.3. **Driving State Transition Graph.** The driving states of a vehicle usually changes over time. For instance, a sequence of driving states for a vehicle can be: [<acceleration, moving straight>, <constant speed, moving straight>, $\cdots$, <deceleration, turning right>]. We propose to develop a driving state transition graph to summarize and characterize such time-varying sequence. In a driving state transition graph, nodes denote driving states, and the weights of edges can be the frequency of state changes or the duration of state changes between two driving states.

Definition 2.4. **Problem Statement.** In this paper, we study the problem of automated driving behavior profiling with GPS traces. Formally, given a driver (a vehicle) and corresponding GPS trajectories, we aim to find a mapping function $f : D \rightarrow V$ that takes the GPS trajectories $D = [< t, \varphi_t, \lambda_t >]_{t=1}^{T}$ as inputs, and outputs a sequence of time-varying yet relational vectorized representations $\mathbf{V} = [\mathbf{v}_n]_{n=1}^{N}$, in order to quantify the dynamics of the driver's



GPS Trajectory
t=1
t=2
t=3
t=T
Transition Probability View
Transition Duration View
Driving State Transition Graph Sequence
Encoder
Peer Dependency
Temporal Dependency
Decoder
PTARL
Representation Result (Vectors)
Prediction and Historical Assessment of Driving Scores
Risky Area Detection

Figure 1: Framework Overview.

driving behavior, where φ_t and λ_t respectively denote the latitude and longitude at the time t. We formulate this problem as a task of spatio-temporal representation learning. Essentially, we first construct a sequence of driving state transition graphs from GPS trajectories, and then learn the latent representations of driving behavior from the graphs.

2.2 Framework Overview

Figure 1 shows an overview of our proposed framework that includes the following essential tasks: (i) constructing multi-view driving state transition graphs; (ii) automated profiling of driving behavior via peer and temporal-aware representation learning; (iii) applications to transportation safety. Specifically, in the first task, we detect driving operations from GPS sequences, identify driving states, and construct driving state transition graphs from the perspectives of transition frequency and duration. In the second task, we incorporate the modeling of both graph-graph peer dependency and past-current temporal dependency into the Auto-Encoder based optimization framework to develop a representation learning model. The proposed method jointly adopts and adapts the ideas of gated recurrent unit and spatial autocorrelation regularization. In the third task, we exploit the learned representations of driving behavior graphs to enable important applications, including (i) prediction and historical assessment of driving scores, (ii) detecting risky regions.

3 CONSTRUCTION OF MULTI-VIEW DRIVING STATE TRANSITION GRAPHS

We propose a step-by-step testable analytic framework to transform GPS trajectories into driving state transition graphs. Specifically, for each driver, we first detect driving operations, identify driving states, and obtain a driving state sequence. Later, we segment the driving state sequence into small subsequences, each of which is reduced into a driving state transition graph. The extracted sequence of driving state transition graphs is used to characterize the time-varying driving behavior of a driver.

Detecting Driving Operations.
From GPS trajectories, we identify two categories of driving operations: (i) speed-related operations including "acceleration", "deceleration" and "constant speed"; (ii) direction-related operations including "turning right", "turning left" and "moving straight". Formally, given three consecutive GPS points $<\varphi_1,\lambda_1>$, $<\varphi_2,\lambda_2>$ and $<\varphi_3,\lambda_3>$ where φ_1, φ_2 and φ_3 respectively denote the three corresponding latitudes, λ_1, λ_2 and λ_3 respectively denote the three corresponding longitudes. We next show how to computationally detect the two types of driving operations.

(1) *Detection of driving-related operations.* Let $\Delta\varphi_{1,2}$ be the difference of φ_1 and φ_2, $\Delta\varphi_{2,3}$ be the difference of φ_2 and φ_3, $\Delta\lambda_{1,2}$ be the difference of λ_1 and λ_2, $\Delta\lambda_{2,3}$ be the difference of λ_2 and λ_3, and R be the radius of the earth. Then, the distance $d_{1,2}$ between the two GPS points $<\varphi_1,\lambda_1>$ and $<\varphi_2,\lambda_2>$ is given by Equation 1:

$$
\begin{aligned}
d_{1,2} =& 2R\cdot \text{atan2}(\sqrt{\sin^2(\Delta\varphi_{1,2}/2)+\cos\varphi_1\cdot\cos\varphi_2\cdot\sin^2(\Delta\lambda_{1,2}/2)}, \\
&\sqrt{1-\sin^2(\Delta\varphi_{1,2}/2)-\cos\varphi_1\cdot\cos\varphi_2\cdot\sin^2(\Delta\lambda_{1,2}/2)}),
\end{aligned} \tag{1}
$$

Similarly, the distance $d_{2,3}$ between the two GPS points $<\varphi_2,\lambda_2>$ and $<\varphi_3,\lambda_3>$ can also be calculated.

Then, given the time stamps of the three GPS points, denoted by t_1, t_2 and t_3, the speed s_2 at t_2 is given by $s_2 = d_{1,2}/(t_2 - t_1)$, the speed s_3 at t_3 is given by $s_3 = d_{2,3}/(t_3 - t_2)$. For t_3, if $s_3 > s_2$, the operation is detected as acceleration; if $s_3 < s_2$, the operation is deceleration; otherwise, the operation is "constant speed".

(2) *Detection of direction-related operations.* To detect the direction-related operations, we calculate the bearing $\theta_{1,2}$ between the two GPS points $<\varphi_1,\lambda_1>$ and $<\varphi_2,\lambda_2>$ by Equation 2:

$$
\theta_{1,2} = \text{atan2}(\sin\Delta\lambda_{1,2}\cdot\cos\varphi_2, \cos\phi_1\cdot\sin\varphi_2-\sin\varphi_1\cdot\cos\varphi_2\cdot\cos\Delta\lambda_{1,2}). \tag{2}
$$

Similarly, we can obtain the bearing $\theta_{2,3}$ between the two GPS points $<\varphi_2,\lambda_2>$ and $<\varphi_3,\lambda_3>$. Therefore, at t_3, if $\theta_{2,3} > \theta1,2$, then the operation is "turning right"; if $\theta_{2,3} < \theta_{1,2}$, then the operation is "turning left"; otherwise, the operation is "moving straight".

Extracting Driving State Sequences.
Based on the two speed-related operations and the three direction-related operations, we can define the following driving states: (1) acceleration while turning right, (2) acceleration while turning left, (3) acceleration while straightforward, (4) deceleration while turning right, (5) deceleration while turning left, (6) deceleration while straightforward, (7) constant speed while turning right, (8) constant speed while turning left, (9) constant speed while straightforward.

With the above definitions, we can identify the driving state of a driver at each time stamp. In other words, each trajectory is associated with a driving state sequence, which is denoted by $\{(ID, t_n, S_n)\}_{n=1}^{N}$, where ID is the identity of the driver, N is the size of the driving state sequence, t_n is the n^{th} time stamp, and S_n is the driving state at t_n.

Constructing Multi-view Driving State Transition Graphs. We next construct driving state transition graphs from driving state sequences. We first segment the driving state sequence into a set of partitions: $\{sq_i\}_{i=1}^{I}$, where each partition corresponds to a small time window ΔT (we will discuss the parameter setup of ΔT in the experimental settings). Then, for each partition sq_i, we extract a feature vector v_i, where each entry is the number of the corresponding driving state, to represent the i^{th} partition. In these graphs, vertexes are regarded as driving states, and edges are regarded as transition relations. The transition relations can be formulated from two views: (i) transition probability, and (ii) transition duration.

(1) **Transition probability view.** The transition probability of driving states shows how likely (frequency) a driver changes from one driving state to another, and thus can be used to characterize driving habits from a frequency perspective. For example, an aggressive driver might easily transit from "acceleration while straightforward " to "acceleration while turning left/right". Quantitatively, the transition probability among driving states can be estimated by the frequencies of state transitions. We normalize the transition frequencies as the transition probability.

(2) **Transition duration view.** The transition duration of driving states shows how long (duration) it takes for a driver to response from one driving state to another, and thus can be used to characterize driving habits from a time perspective. For example, if the transition duration between "acceleration while straightforward" and "deceleration while straightforward" is small, this reflects that the driver is impatient and does not care about passengers' feelings. Quantitatively, we can calculate the average time interval of the transitions between two driving states.

After that, we can obtain two graph sequences of driving state transition probability and driving state transition duration for each driver respectively.

4 PEER AND TEMPORAL-AWARE REPRESENTATION LEARNING

We present a spatio-temporal representation learning method to model peer and temporal dependencies in representation learning.

4.1 Model Intuitions

There are peer and temporal dependencies among driving behavior. Therefore, in our approach, we model the representation of driving behavior based on the following intuition.

Intuition 1: Structural Reservation After reducing driving behavior into graphs, we needs a representation learning based method to transform graphs into vectors in a latent feature space for automated quantification and profiling. Consequently, the method should be able to project graphs into lower-dimensional vectors while reserving corresponding characteristics and structures.

Intuition 2: Peer Dependency. If two drivers exhibit similar driving habits, and the vehicle operation patterns of two corresponding trajectories are similar, then the driving state transition graphs of these two trajectories share a lot in terms of structures and characteristics. As a result, the learned representations of driving behavior should be close to each other. Consequently, the method should be able to model the graph-graph peer dependency in representation learning.

Intuition 3: Temporal Dependency. The driving operations of the current time slot have autocorrelation with previous driving states. For example, if a driver decelerates while straightforward at t, and if $\Delta(t, t+1)$ is small enough, then he is likely to accelerate at $t+1$. Consequently, the method should be able to model the current-past temporal dependency in representation learning.

4.2 Base Model

We utilize the deep Auto-Encoder model [2] as our base model. Auto-Encoder is an unsupervised neural network model, which projects the instances in original feature representations into a lower-dimensional feature space via a series of non-linear mappings. The Auto-Encoder model involves two steps: encode and decode. The encode part projects the original feature vector to the objective feature space, while the decode step recovers the latent feature representation to a reconstruction space. In the auto-encoder model, we need to ensure that the original feature representation of instances should be as similar to the reconstructed feature representation as possible.

Formally, let $\mathbf{x_i}$ be the original feature representation of the i^{th} driver, and $\mathbf{y}^1, \mathbf{y}^2, \cdots, \mathbf{y}^o$ be the latent feature representations of the diver at hidden layers $1, 2, \cdots, o$ in the encode step respectively, the encoding result in the objective lower-dimension feature space can be represented as $\mathbf{z_i} \in \mathbb{R}^d$ with dimension d. Formally, the relationship between these vector variables is denoted by:

$$\begin{cases} \mathbf{y}_i^1 &= \sigma(\mathbf{W}^1 \mathbf{x}_i + \mathbf{b}^1), \\ \mathbf{y}_i^k &= \sigma(\mathbf{W}^k \mathbf{y}_i^{k-1} + \mathbf{b}^k), \forall k \in \{2, 3, \cdots, o\}, \\ \mathbf{z_i} &= \sigma(\mathbf{W}^{o+1} \mathbf{y}_i^o + \mathbf{b}^{o+1}). \end{cases} \tag{3}$$

Meanwhile, in the decode step, the input will be the latent feature vector $\mathbf{z}_i$ (i.e., the output of the encode step), and the final output will be the reconstructed vector $\hat{\mathbf{x}}_i$. The latent feature vectors at each hidden layers can be represented as $\hat{\mathbf{y}}_i^o, \hat{\mathbf{y}}_i^{o-1}, \cdots, \hat{\mathbf{y}}_i^1$. The relationship between these vector variables is denoted by:

$$\begin{cases} \hat{\mathbf{y}}_i^o &= \sigma(\hat{\mathbf{W}}^{o+1} \mathbf{z}_i + \hat{\mathbf{b}}^{o+1}), \\ \hat{\mathbf{y}}_i^{k-1} &= \sigma(\hat{\mathbf{W}}^k \hat{\mathbf{y}}_i^k + \hat{\mathbf{b}}^k), \forall k \in \{2, 3, \cdots, o\}, \\ \hat{\mathbf{x}}_i &= \sigma(\hat{\mathbf{W}}^1 \hat{\mathbf{y}}_i^1 + \hat{\mathbf{b}}^1). \end{cases} \tag{4}$$

where $\mathbf{W}$s and $\mathbf{b}$s are the weight matrices and bias terms to be learned in the model.

The objective of the auto-encoder model is to minimize the loss between the original feature vector $\mathbf{x}$ and the reconstructed feature vector $\hat{\mathbf{x}}$. Formally, the loss function is

$$\mathcal{H}(\mathcal{U}) = \frac{1}{2} \sum_{u_i \in \mathcal{U}} \|(\mathbf{x}_i - \hat{\mathbf{x}}_i)\|_2^2 \tag{5}$$

where u_i denotes the i^{th} driver and $\mathcal{U}$ denotes the driver set.

4.3 Incorporating Temporal Dependency

The Auto-Encoder model is not able to capture the current-past temporal dependency. But, the Gated Recurrent Unit (GRU) is a variant of Long Short Term Memory networks (LSTMs), and can better connect previous information to the present task compared with the basic LSTMs [10]. To model the temporal dependency in representation learning, we incorporate GRU into the middle layer of auto-encoder, as shown in Figure 2.

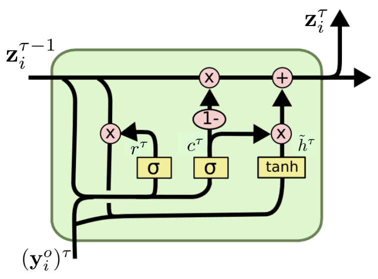


Figure 2: Incorporation of GRU with Auto-Encoder.

The driving behavior transition graph sequence, denoted by $(G_i^0, G_i^1, \cdots, G_i^{\tau}, \cdots)$, represents the evolution of the time-varying driving behavior of the i-th driver u_i, at time slots $0, 1, \cdots, \tau, \cdots$, respectively. We flatten each graph into a vector to obtain the original vector representation of the driving behavior transition graph series, denoted by $(\mathbf{x}_i^0, \mathbf{x}_i^1, \cdots, \mathbf{x}_i^{\tau}, \cdots)$, where $\mathbf{x}_i^{\tau}$ is the original vector representation of G_i^{τ}.

In the evolution, the status of the driving behavior transition graph of the i^{th} driver (denoted by G_i^{τ}) at the time slot τ depends on the status of the graph at previous time slot τ (denoted by $G_i^{\tau-1}$). Formally, we can represent its status at these two time slots as $\mathbf{z}_i^{\tau-1}$ and $\mathbf{z}_i^{\tau}$ respectively. $\mathbf{z}_i^{\tau}$ evolves from $\mathbf{z}_i^{\tau-1}$, and the dependence relationship between them can be modeled with GRU. The temporal dependency between $\mathbf{z}_i^{\tau}$ and $\mathbf{z}_i^{\tau-1}$ can be denoted by

$$\mathbf{z}_i^{\tau} = (1-c^{\tau})\mathbf{z}_i^{\tau-1} + c^{\tau}\tilde{\mathbf{z}}_i^{\tau}, where \begin{cases} c^{\tau} & = \sigma(\mathbf{W}_c[\mathbf{z}^{\tau-1}, (\mathbf{y}_i^o)^{\tau}]) \\ r^{\tau} & = \sigma(\mathbf{W}_r[\mathbf{z}^{\tau-1}, (\mathbf{y}_i^o)^{\tau}]) \\ \tilde{\mathbf{z}}_i^{\tau} & = tanh(\mathbf{W}[r^{\tau}\mathbf{z}^{\tau-1}, (\mathbf{y}_i^o)^{\tau}]). \end{cases} \tag{6}$$

Therefore, the temporal-aware Auto-Encoder is denoted by:

$$\begin{cases} & \#Sequential \quad Encode \quad Step \\ (\mathbf{y}_i^1)^{\tau} & = \sigma(\mathbf{W}^1\mathbf{x}_i^{\tau} + \mathbf{b}^1), \\ (\mathbf{y}_i^k)^{\tau} & = \sigma(\mathbf{W}^k(\mathbf{y}_i^{k-1})^{\tau} + \mathbf{b}^k), \forall k \in \{2, 3, \cdots, o\}, \\ \mathbf{z}_i^{\tau} & = (1-c^{\tau})\mathbf{z}_i^{\tau-1} + c^{\tau}\tilde{\mathbf{z}}_i^{\tau}. \end{cases} \tag{7}$$

$$\begin{cases} & \#Sequential \quad Decode \quad Step \\ (\hat{\mathbf{y}}_i^o)^{\tau} & = \sigma(\hat{\mathbf{W}}^{o+1}\mathbf{z}_i^{\tau} + \hat{\mathbf{b}}^{o+1}), \\ (\hat{\mathbf{y}}_i^{k-1})^{\tau} & = \sigma(\hat{\mathbf{W}}^k(\hat{\mathbf{y}}_i^k)^{\tau} + \hat{\mathbf{b}}^k), \forall k \in \{2, 3, \cdots, o\}, \\ \hat{\mathbf{x}}_i^{\tau} & = \sigma(\hat{\mathbf{W}}^1(\hat{\mathbf{y}}_i^1)^{\tau} + \hat{\mathbf{b}}^1). \end{cases} \tag{8}$$

where all outputs of each layer are labeled superscript by corresponding time slot. Then the loss function is:

$$\mathcal{H}(\mathcal{U}) = \frac{1}{2}\sum_{\tau\in\mathcal{T}}\sum_{u_i\in\mathcal{U}} \|(\mathbf{x}_i^{\tau} - \hat{\mathbf{x}}_i^{\tau})\|_2^2 \tag{9}$$

4.4 Incorporating Peer Dependency

In the graph-graph peer dependency, trajectories that share similar driving behavior should have close representations in the learned representation feature space. Subject to such constraint, we introduce the loss function $\mathcal{H}_c(G^{\tau})$ to model the peer dependency.

$$\mathcal{H}_c(G^{\tau}) = \sum_{u_i\in\mathcal{U}}\sum_{u_j\in\mathcal{U}, u_i\neq u_j} s_{i,j}^{\tau} \cdot \|\mathbf{z}_i^{\tau} - \mathbf{z}_j^{\tau}\|_2^2 \tag{10}$$

where $s_{i,j}^{\tau}$ is the function to evaluate the similarity of driving behavior between the driver u_i and u_j at the time slot τ. We can define the function $s_{i,j}^{\tau}$ in many ways. For simplicity, in this paper, we define $s_{i,j}^{\tau}$ as the cosine similarity between the original representation vectors $\mathbf{x}_i^{\tau}$ and $\mathbf{x}_j^{\tau}$:

$$s_{i,j}^{\tau} = cos(\mathbf{x}_i^{\tau}, \mathbf{x}_j^{\tau}) \tag{11}$$

4.5 Solving the Optimization Problem

Formally, by incorporating temporal and peer dependency, we obtain the joint optimization objective function as follows:

$$\min \frac{1}{2}\sum_{\tau\in\mathcal{T}}\{\sum_{u_i\in\mathcal{U}(n)} \|(\mathbf{x}_i^{\tau} - \hat{\mathbf{x}}_i^{\tau})\|_2^2 + \alpha\cdot\mathcal{H}_c(G^{\tau})\} \tag{12}$$

where α is the hyperparameter to control the regularizer $\mathcal{H}_c(G^{\tau})$.

To minimize the objective function, we utilize Stochastic Gradient Descent (SGD) to infer parameters. For parameters of decoder layers of PTARL, the updating rule is:

$$\begin{aligned} \hat{\mathbf{W}}_{new}^k = \hat{\mathbf{W}}_{old}^k - \lambda\cdot(-\sum_{u_i\in\mathcal{U}(n)} (\mathbf{x}_i^{\tau} - \hat{\mathbf{x}}_i^{\tau})\cdot\hat{\mathbf{x}}_i^{\tau}(1-\hat{\mathbf{x}}_i^{\tau})\cdot \\ \prod_{s=1}^{k-1}(\hat{\mathbf{y}}_i^s)^{\tau}(1-(\hat{\mathbf{y}}_i^s)^{\tau})\hat{\mathbf{W}}_{old}^s\cdot(\hat{\mathbf{y}}_i^k)^{\tau}) \end{aligned} \tag{13}$$

$$\begin{aligned} \hat{\mathbf{b}}_{new}^k = \hat{\mathbf{b}}_{old}^k - \lambda\cdot(-\sum_{u_i\in\mathcal{U}(n)} (\mathbf{x}_i^{\tau} - \hat{\mathbf{x}}_i^{\tau})\cdot(\hat{\mathbf{x}}_i^{\tau})(1-(\hat{\mathbf{x}}_i^{\tau}))\cdot \\ \prod_{s=1}^{k-1}(\hat{\mathbf{y}}_i^s)^{\tau}(1-(\hat{\mathbf{y}}_i^s)^{\tau})\hat{\mathbf{W}}_{new}^s) \end{aligned} \tag{14}$$

For detailed parameter inferences, please refer to link [1].

5 APPLICATIONS

We demonstrate the two applications in transportation safety: (i) prediction and historical assessment of driving scores and (ii) risky area detection. To that end, we invite the domain experts from the Department of Transportation (DoT) to provide a driving score for each driver, by examining their driving operations across the entire time span.

5.1 Prediction and Historical Assessment of Driving Scores

Our proposed method can learn a series of vectorized representations for a driver at each time slot. Therefore, we can apply these representation vectors to train a regression model to predict and assess historical driving scores. To prepare the features of driving behavior, we apply the proposed method to learn a series of vectorized representations from a driver's trajectories. Then, we use the last vectorized representation paired with a driving score

[1]https://goo.gl/cnECP8

to train the Support Vector Regression (SVR) model [4]. Because the last learned vectorized representation has memorized the previous driving behavior and corresponding vectorized representations via the temporal dependency. The formulation of SVR is: $f(\mathbf{z}_i^\tau, w) = \sum_{j=1}^{m} w_j g_j(\mathbf{z}_i^\tau) + b$, where $\mathbf{z}_i^\tau$ is the learned representation vector of the i^{th} driver at the time slot τ, $g_j(\mathbf{z}_i^\tau)$ is a set of nonlinear transformation of $\mathbf{z}_i^\tau$, and b is a bias term. Specifically, we choose g_i as three-order polynomial transformation. Then the objective function of SVR is

$$\min \frac{1}{2}||w||^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^\star), s.t. \begin{cases} y_i - f(\mathbf{z}_i^\tau, w) \le \epsilon + \xi_i^\star \\ f(\mathbf{z}_i^\tau, w) - y_i^\tau \le \epsilon + \xi_i \\ \xi_i, \xi_i^\star \ge 0, i = 1, \cdots, n. \end{cases} \quad (15)$$

where $\xi_i, \xi_i^\star$ are the slack variables to measure the deviation of training samples outside ϵ-insensitive zone, and y_i^τ is the driving score of the i^{th} driver at the time τ. In addition, with the learned SVR model and the series of vectorized representations, we can assess the historical driving score of a driver at a specific previous time slot in a backward direction.

5.2 Risky Area Detection

It is important to understand how driving scores are distributed spatially and temporally. Therefore, we study the spatio-temporal dynamics of driving scores across all the areas, so as to detect risky areas. Specifically, if the vehicles in a given area are operated by low-score drivers, this area is likely to be risky. To detect risky areas, we first apply the trained driving scores predictive model to predict driving scores for a specific driver, at a specific time slot, and at a specific location. Moreover, we compute the average driving scores $\bar{y}_l^\tau$ of all the drivers for a specific location l and a specific time slot τ. In addition, since the occurrence of traffic accidents follows a Poisson distribution[11], we define the threshold γ^τ for detecting risky areas at the time slot τ as the lower bound of the 95% confidence interval: $\gamma^\tau = \mu^\tau - 1.96\sigma^\tau$, where μ^τ and σ^τ are the mean and the standard deviation of the average driving scores at time τ respectively. If $\bar{y}_l^\tau < \gamma^\tau$, then we detect the area l as a risky one; otherwise non-risky. We visualize the detection results using heat maps in the experiment.

6 EXPERIMENTAL RESULTS

This section details our empirical evaluation of the proposed method on real-world data.

6.1 Data Description

Table 1 shows the statistics of our real-world data sets *T-Drive* trajectory dataset [31, 32]. Each GPS point contains the information of corresponding driver ID, latitude, longitude, and time stamp. To prepare benchmark driving scores, we invite the domain experts from DoT to help us evaluate the visualizations of trajectories, and assign a driving performance score ranging from 0 to 1 to each driver. The higher the score is, the safer the driver is. Figure 3 shows the distribution of the driving scores: only a small number of the drivers have very high or very low scores, while most scores are moderate and range from 0.45 to 0.6.

Table 1: Statistics of the experimental data.

Properties	Statistics
Number of drivers	10,357
Time range	Feb.2 - Feb.8
City	Beijing

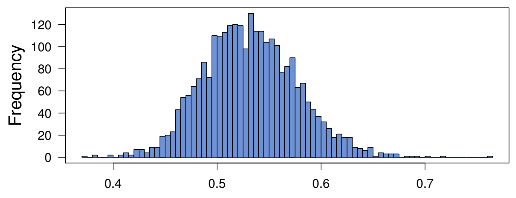


Figure 3: Driving score distribution.

6.2 Evaluation Metrics

Let us assume that each driver i is associated with a benchmark score y_i and a predicted score f_i. To show the effectiveness of the proposed model, we use the following metrics for evaluation.

Square Error. We utilize *Square Error* (SE) to measure regression errors. $SE = \sum_i (y_i - f_i)^2$, where N is the number of drivers. The lower the SE is, the better the learned representation is.

Coefficient of Determination. We utilize the *coefficient of determination* (or R^2 for short) to measure the regression accuracy. R^2 can be represented as $R^2 = 1 - \sum_i (y_i - f_i)^2 / \sum_i (y_i - \bar{y})^2$, where $\bar{y}$ is the mean of benchmark scores.

Kendall's Tau Coefficient. We utilize *Kendall's Tau Coefficient (Tau)* to measure the overall ranking accuracy. For a driver pair $< i, j >$, $< i, j >$ is said to be concordant, if both $y_i > y_j$ and $f_i > f_j$ or if both $y_i < y_j$ and $f_i < f_j$. Also, $< i, j >$ is said to be discordant, if both $y_i < y_j$ and $f_i > f_j$ or if both $y_i < y_j$ and $f_i > f_j$. Tau is given by Tau $= \frac{\#_{conc} - \#_{disc}}{\#_{conc} + \#_{disc}}$.

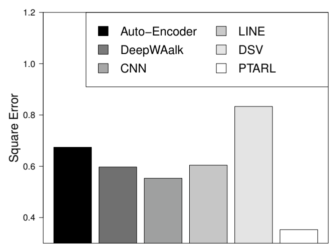
Normalized Discounted Cumulative Gain. We utilize *Normalized Discounted Cumulative Gain* (NDCG@N) to measure the ranking accuracy at the top-N cases. The discounted cumulative gain (DCG@N) is given by $NDCG[N] = \sum_{i'=1}^{N} \frac{y_{i'}}{log_2(1+i')} / \sum_{i=1}^{N} \frac{y_i}{log_2(1+i)}$ where i denotes the original ranking order of the benchmark and i' denotes the ranking order of the prediction. The larger NDCG@N is, the higher top-N ranking accuracy is.

6.3 Baseline Algorithms

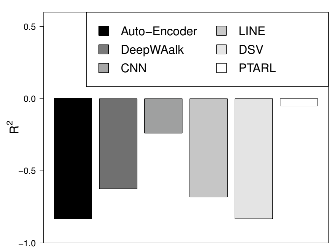
We compare the performances of our method against the following baseline algorithms.

(1) Auto-Encoder. The Auto-Encoder model [2] minimizes the loss between the original feature representations and reconstructed ones. In the experiments, we set the number of hidden layers = 4, the size of middle layer = 20.

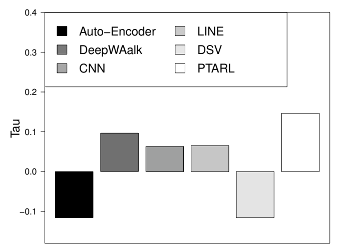
(2) DeepWalk. The DeepWalk model [21] extends the word2vec model [18] to the scenario of network embedding. DeepWalk uses local information obtained from truncated random walks to learn latent representations. In the experiments, we set the number of walks = 80, the size of representation = 20, the walk length = 40, and the window size = 10.



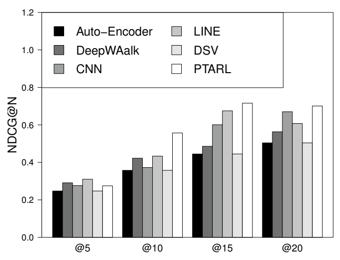
(a) SE



(b) R^2

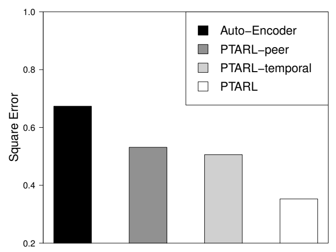


(c) Tau

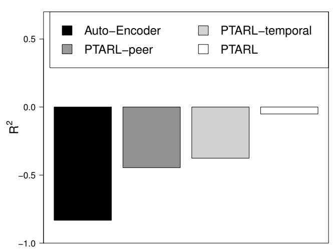


(d) NDCG@N

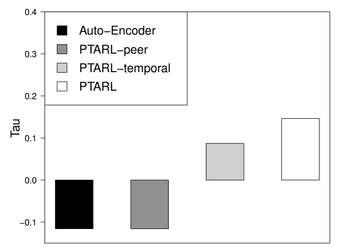
Figure 4: Overall comparison.



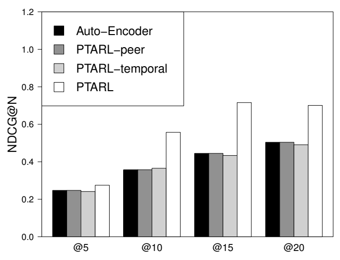
(a) SE



(b) R^2

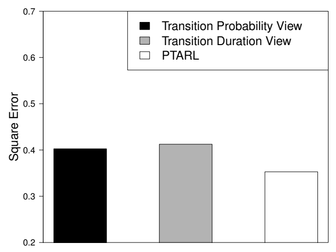


(c) Tau

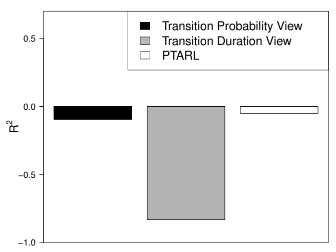


(d) NDCG@N

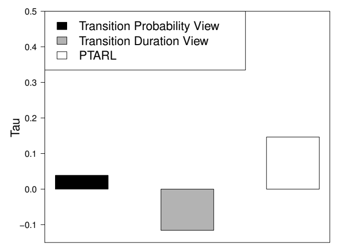
Figure 5: Performance with different dependencies.



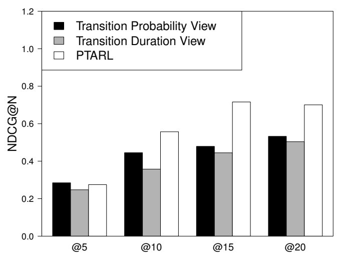
(a) SE



(b) R^2



(c) Tau



(d) NDCG@N

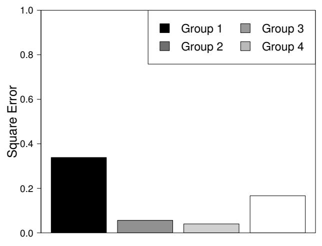
Figure 6: Performance in different views.

(3) LINE. The LINE model optimizes the objective function that preserves both the local and global network structures with an edge-sampling algorithm [24]. In the experiments, we set the size of representation = 20, the number of negative samples = 5, and the starting value of the learning rate = 0.025.
(4) CNN. The CNN model refers to Convolutional Neural Network, which projects original feature space into a new space via a variation of multilayer perceptrons [15]. In the experiments, we set the number of convolutional layer = 2.
(5) Driving State Vector (DSV). In addition, we also compare our model with the traditional transportation approach. We adopt the driving states defined in [5, 36] to profile driving behavior, including (1)acceleration while turning, (2)acceleration while straightforward, (3) deceleration while turning, (4) deceleration while straightforward, (5) constant speed while turning, and (6) constant speed while straightforward. We formulate a driving state vector (DSV) for each driver, where each entry in the vector is the percentage of the corresponding driving state. We feed DSVs and corresponding driving scores into SVR to train the regression model.

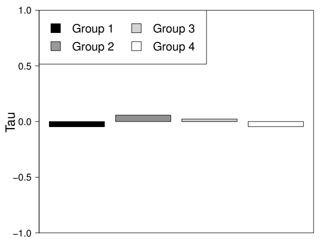
For the construction of multi-view driving state transition graphs, we set the time window $\Delta = 30 minutes$. For the structure of PTARL, we set the number of hidden layers of encoder = 1, the number of hidden layers of decoder = 2, the output size of vectors = 20. the penalty parameter $\alpha = 0.01$ for regularizer $\mathcal{H}_c(G^\tau)$, the learning rate = 0.0001. The source code of PTARL is available at link [2].

For SVR, we set the penalty parameter $C = 0.1$, $\epsilon = 0.0993$. All the evaluations are performed on a x64 machine with Intel E5-1680 3.40GHz CPU (with 14 cores) and 128GB RAM. The operation system is CentOS 7.4.

[2]https://github.com/Merlin55/PTARL

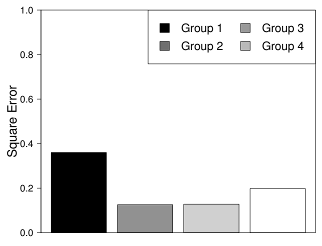


(a) SE

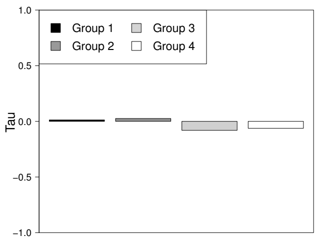


(b) Tau

Figure 7: Robustness check in the score-based group.



(a) SE



(b) Tau

Figure 8: Robustness check in the driving-state-based group.

6.4 Overall Performance

We compare our method with the baseline methods in terms of SE, R^2, Tau and NDCG@N. Figure 4 shows our model achieves the best performance. Auto-Encoder, DeepWalk, CNN, and LINE are not able to model sequential inputs. Therefore, in the experiment, we aggregate all the driving state transitions across all the time slots into one transition graph for these baselines. Since the aggregation will lose the peer and temporal dependencies of driving behavior, the performances of these baselines are severely disrupted.

6.5 Robustness Check

We apply the learned representations of driving behavior and the SVR model to different subgroups of the drivers, to examine the robustness of our method in these subgroups. We use two grouping methods: (i) driving score based grouping, (ii) driving state based grouping. For (i), we segment drivers into four subgroups by the driving score y_i, i.e. $y_i < 0.45$, $0.45 \leq y_i < 0.55$, $0.55 \leq y_i < 0.65$, and $y_i \geq 0.65$. For (ii), we generate driving state vectors (introduced in Section 6.3) for each driver; we then apply K-Means [14] to cluster drivers into four groups based on their driving vectors.

Figure 7 shows that for (i), our model can achieve a relative stable performance, especially in terms of Tau. For (ii) the results validate the assumption that if two drivers show similar driving behavior, their predicted driving scores are similar as well.

6.6 Study of Peer and Temporal Dependencies

We study the effects of peer and temporal dependencies on the model, by comparing our PTARL with Auto-Encoder and two other variants of PTARL, i.e. (i) **PTARL-peer** that only considers the peer dependency, and (ii) **PTARL-temporal** that only considers the temporal dependency.

From Figure 5, we can observe that PTARL outperforms Auto-Encoder, PTARL-peer and PTARL-temporal significantly. Because of ignoring both peer and temporal dependencies, Auto-Encoder performs worst in the comparison. Also, the results show that PTARL-temporal performs better than PTARL-peer, which means that the temporal dependency is more significant in profiling driving behavior than the peer dependency.

6.7 Study of Performance in Different Views

We introduce two views in driving state transition graphs: transition probability view and transition duration view. Therefore, we investigate the performance of our model under these two views.

From Figure 6, it is interesting to observe that the performances of the transition probability view or transition duration view are worse than the combination of these two views. And, the difference of errors between two views are subtle. The possible explanation is that only one view, no matter probability view or time view, can not capture the complete information of driving behavior. By combining these two views, our model can systematically make up the deficiency of each single view.

6.8 Historical Assessment of Driving Scores

We select the driver with the highest driving score and the driver with the lowest driving score. Specifically, we name the driver with the highest driving score as "Safer Driver", and the other one with lowest driving score as "Riskier Driver". We utilize their learned representation sequences to assess their historical driving sores. We report the experimental results in Figure 10.

There is an interesting observation that a "Safer Driver" is not always safe and a "Riskier Driver" is not always risky. The driving scores are varying over time. Sometimes, the driving score of the "Riskier Driver" is higher than the "Safer Driver". There is a pattern that scores of the "Safer Driver" are relatively higher at most time, while the scores of the "Riskier Driver" are relatively lower at most time. This observation is consistent with our common sense that driving behavior is affected by random factors, like weather, road condition and mind status, while the driving habit has a relatively stable pattern.

6.9 Risky Area Detection

Figure 9 shows the visualization results of risky area detection. We can observe the dynamic evolution of the distribution of risky areas over time. Behind the evolution, there are two "varying" lines: "varying" driving behavior and "varying" locations for drivers. Figure 10 shows that driving behavior varying over time. Meanwhile locations of drivers are also changing, because drivers are always moving. Therefore, the mixture of two "varying" leads to the evolution of risky areas.

7 RELATED WORK

Spatio-temporal Representation learning. Our work is relevant to spatio-temporal representation learning. Spatio-temporal representation learning is the elevation of graph representation learning in the spatio-temporal contexts. Graph representation learning, also known as graph/network embedding, aims to learn a low-dimensional vector to represent vertexes or graphs[1, 3, 19,

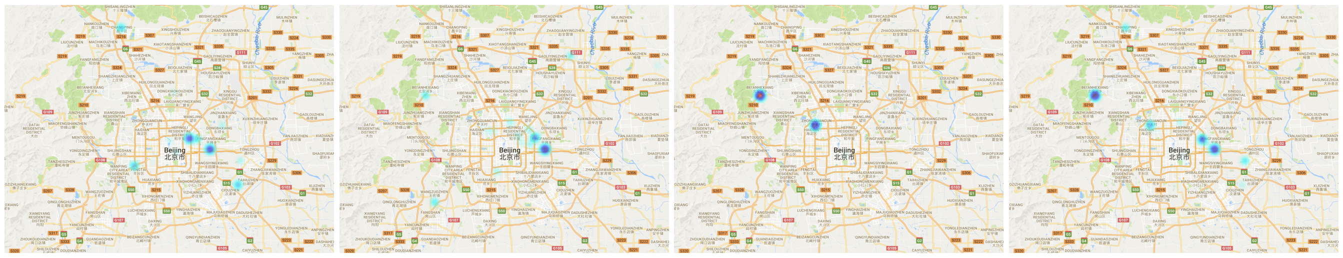

(a) t=1 (b) t=2 (c) t=3 (d) t=4

Figure 9: Risky area detection.

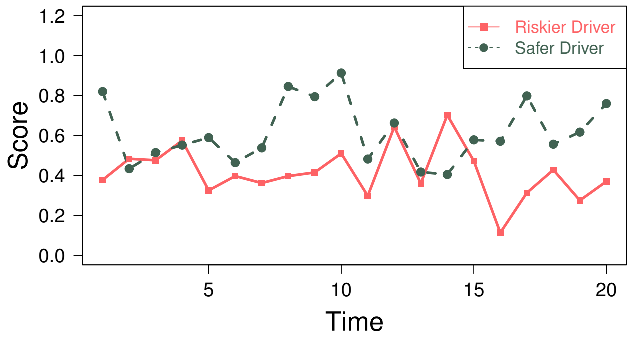


Figure 10: An example of historical assessment of driving scores over time for a safer driver and a riskier driver.

25, 28, 30]. For spatio-temporal representation learning, *Wang et al.* developed a geographical learning method to find proper representations of communities to mimic the spatial structure. *Wang et al.* proposed a collective embedding framework to learn the community structure from multiple periodic spatial-temporal graphs of human mobility [27]. *Yao et al.* developed a embedding method to learn the urban functions by exploring human mobility patterns.

Human Mobility modeling. Our work is related to the research of human mobility modeling. There are existing studies on human mobility modeling by exploiting mobility patterns to enable various applications[35]. *Wang et al.* encoded the dynamic mobility flow into vector representations of regions through a embedding method [26]. *Yuan et al.* analogized human mobility patterns as words, and exploited both topic modeling and spectrum analysis to analyze the urban functions of regions [33]. *Fu et al.* developed a geographic method named ClusRanking to exploite the geographic dependencies of the value of an estate with online user reviews and offline moving behaviors [6]. *Lu Lin et al.* presented a unified probabilistic framework, called Topic-Enhanced Gaussian Process Aggregation Model (TEGPAM), consisting of three components, i.e. location disaggregation model, traffic topic model and trafiňĄc speed Gaussian Process model, which integrate new-type data with traditional data [16]. *Sun et al.* proposed a RNN-based model to simulate the influence of dynamic social network over the human interests [23]. *Fu et al.* learned latent community functions and the corresponding portfolios of estates from human mobility data and Point of Interest (POI) data to incorporate the functional diversity of communities into real estate appraisal [7]. *Fu et al.* proposed a geographic method, named ClusRanking, for real estate appraisal by leveraging the mutual enforcement of ranking and clustering power and collecting estate-related mobile data [9]. *Yuan et al.* proposed a Bayesian non-parametric model, named Periodic Region Detection (PRED), to discover periodic mobility patterns by jointly modeling the geographical and temporal information [34]. *Lin et al.* proposed a new passive verification method that requires minimal imposition of users through modeling users subtle mobility patterns [17]. *Fu et al.* developed a general collective learning approach to model large-scale Heterogeneous Human Mobility Data (HHMD) at an individual level towards identifying and quantifying the urban forms of residential communities [8]. *Wang et al.* proposed a general probabilistic framework based on Howkees for spotting trip purposes from massive taxi GPS trajectories [29].

Driving behavior analysis. Finally, our work has a connection with driving behavior analysis. Prior driving behavior analysis research can be summarized as non-contextual and contextual approaches, according to whether driving information is context relevant. For non-contextual methods, they solely applied vehicle kinematic information like speed and acceleration/deceleration, to model driving behavior. For example, *Ellison et al.* applied Temporal and Spatial Identifiers to provide a common measurement of driving behavior, using vehicle motion information [5]. *Paefgen et al.* performed driver risk profiling by constructing features from real-world GPS data that included accident and accident-free driver [20]. For contextual approaches, they added contextual information like weather and road quality to models, except vehicle motion data. For example, *Zhu et al.* proposed a Bayesian Network model which combined GPS driving observations, individual driving behavior and individual driving risks with contextual features such as road conditions and dynamic traffic flow information [36]. *Jun et al.* evaluated driving exposure and performance differences between who were involved in crashes or not, with the detailed exposure data of individual drivers (travel by time of day and by roadway type) alongside driving performance data (speed, throttle, braking, and acceleration) [13].

8 CONCLUSION REMARKS

Driving behavior analysis has been important for assessing driver performances, improving traffic safety, and developing the intelligent and resilient transportation systems. In this paper, we investigated driving behavior analysis from the perspective of representation learning. We formulated the problem of driving behavior profiling and scoring as a task of spatial and temporal embedding and labeling with driving state transition graphs. We studied large-scale driving behavior data, and identified the peer and temporal dependencies. To improve the performance of automated behavior profiling, we developed an analytic framework that jointly modeled the peer and temporal dependencies. Specifically, we first construct multi-view driving state transition graphs from GPS traces to characterize driving behavior. Besides, we incorporated the idea of gated recurrent unit to model both the graph-graph peer dependency and integrate graph-graph peer penalties to capture the current-past temporal dependency in a unified optimization framework. In addition, we applied our proposed method to enable the applications of driving score prediction and risky area detection. The empirical experiments on real-world data demonstrated the effectiveness of spatio-temporal representation learning for profiling driving behavior.

ACKNOWLEDGMENTS

This research was partially supported by the National Science Foundation (NSF) via the grant number: 1755946. This research was partially supported by the University of Missouri Research Board (UMRB) via the proposal number: 4991.

REFERENCES

[1] Yoshua Bengio and Olivier Delalleau. 2009. Justifying and generalizing contrastive divergence. *Neural computation* 21, 6 (2009), 1601–1621.

[2] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. 2007. Greedy layer-wise training of deep networks. In *Advances in neural information processing systems*. 153–160.

[3] Kamran Ghasedi Dizaji, Amirhossein Herandi, Cheng Deng, Weidong Cai, and Heng Huang. 2017. Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization. In *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, 5747–5756.

[4] Harris Drucker, Christopher JC Burges, Linda Kaufman, Alex J Smola, and Vladimir Vapnik. 1997. Support vector regression machines. In *Advances in neural information processing systems*. 155–161.

[5] Adrian B Ellison, Michiel CJ Bliemer, and Stephen P Greaves. 2015. Evaluating changes in driver behaviour: a risk profiling approach. *Accident Analysis & Prevention* 75 (2015), 298–309.

[6] Yanjie Fu, Yong Ge, Yu Zheng, Zijun Yao, Yanchi Liu, Hui Xiong, and Jing Yuan. 2014. Sparse real estate ranking with online user reviews and offline moving behaviors. In *Data Mining (ICDM), 2014 IEEE International Conference on*. IEEE, 120–129.

[7] Yanjie Fu, Guannan Liu, Spiros Papadimitriou, Hui Xiong, Yong Ge, Hengshu Zhu, and Chen Zhu. 2015. Real estate ranking via mixed land-use latent models. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 299–308.

[8] Yanjie Fu, Guannan Liu Liu, Yong Ge, Pengyang Wang, Hengshu Zhu, and Hui Xiong. 2018. Representing Urban Forms: A Collective Learning Model with Heterogeneous Human Mobility Data. *IEEE transactions on knowledge and data engineering* (2018).

[9] Yanjie Fu, Hui Xiong, Yong Ge, Yu Zheng, Zijun Yao, and Zhi-Hua Zhou. 2016. Modeling of Geographic Dependencies for Real Estate Ranking. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 11, 1 (2016), 11.

[10] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. 2017. LSTM: A search space odyssey. *IEEE transactions on neural networks and learning systems* (2017).

[11] Ismail Bulent Gundogdu. 2010. Applying linear analysis methods to GIS-supported procedures for preventing traffic accidents: Case study of Konya. *Safety Science* 48, 6 (2010), 763–769.

[12] Michael Robert James, Michael Edward Samples, and Steven F Kalik. 2013. Combining driver and environment sensing for vehicular safety systems. US Patent 8,384,534.

[13] Jungwook Jun, Jennifer Ogle, and Randall Guensler. 2007. Relationships between crash involvement and temporal-spatial driving behavior activity patterns: use of data for vehicles with global positioning systems. *Transportation Research Record: Journal of the Transportation Research Board* 2019 (2007), 246–255.

[14] Tapas Kanungo, David M Mount, Nathan S Netanyahu, Christine D Piatko, Ruth Silverman, and Angela Y Wu. 2002. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE transactions on pattern analysis and machine intelligence* 24, 7 (2002), 881–892.

[15] Steve Lawrence, C Lee Giles, Ah Chung Tsoi, and Andrew D Back. 1997. Face recognition: A convolutional neural-network approach. *IEEE transactions on neural networks* 8, 1 (1997), 98–113.

[16] Lu Lin, Jianxin Li, Feng Chen, Jieping Ye, and Jin-Peng Huai. 2017. Road traffic speed prediction: a probabilistic model fusing multi-source data. *IEEE Transactions on Knowledge and Data Engineering* (2017).

[17] Miao Lin, Hong Cao, Vincent W Zheng, Kevin Chen-Chuan Chang, and Shonali Krishnaswamy. 2015. Mobility Profiling for User Verification with Anonymized Location Data.. In *IJCAI*. 960–966.

[18] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[19] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. Asymmetric transitivity preserving graph embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1105–1114.

[20] Johannes Paefgen, Florian Michahelles, and Thorsten Staake. 2011. GPS trajectory feature extraction for driver risk profiling. In *Proceedings of the 2011 international workshop on Trajectory data mining and analysis*. ACM, 53–56.

[21] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 701–710.

[22] Sarah M Simmons, Anne Hicks, and Jeff K Caird. 2016. Safety-critical event risk associated with cell phone tasks as measured in naturalistic driving studies: A systematic review and meta-analysis. *Accident Analysis & Prevention* 87 (2016), 161–169.

[23] Peijie Sun Sun, Le Wu, and Meng Wang. 2018. Attentive Recurrent Social Recommendation. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[24] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding.. In *WWW*. ACM.

[25] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1225–1234.

[26] Hongjian Wang and Zhenhui Li. 2017. Region Representation Learning via Mobility Flow. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 237–246.

[27] Pengyang Wang, Yanjie Fu, Jiawei Zhang, Xiaolin Li Li, and Dan Lin. 2018. Learning Urban Community Structures: A Collective Embedding Perspective with Periodic Spatial- temporal Mobility Graphs. *ACM Transactions on Intelligent Systems and Technology (TIST)* (2018).

[28] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, and Xueqi Cheng. 2016. Multi-task representation learning for demographic prediction. In *European Conference on Information Retrieval*. Springer, 88–99.

[29] Pengfei Wang, Guannan Liu, Yanjie Fu, Yuanchun Zhou, and Jianhui Li. 2018. Spotting Trip Purposes from Taxi Trajectories: A General Probabilistic Model. *ACM Transactions on Intelligent Systems and Technology (TIST)* 9, 3 (2018), 29.

[30] Yue Wang, Dawei Yin, Luo Jie, Pengyuan Wang, Makoto Yamada, Yi Chang, and Qiaozhu Mei. 2016. Beyond ranking: Optimizing whole-page presentation. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. ACM, 103–112.

[31] Jing Yuan, Yu Zheng, Xing Xie, and Guangzhong Sun. 2011. Driving with knowledge from the physical world. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 316–324.

[32] Jing Yuan, Yu Zheng, Chengyang Zhang, Wenlei Xie, Xing Xie, Guangzhong Sun, and Yan Huang. 2010. T-drive: driving directions based on taxi trajectories. In *Proceedings of the 18th SIGSPATIAL International conference on advances in geographic information systems*. ACM, 99–108.

[33] Nicholas Jing Yuan, Yu Zheng, Xing Xie, Yingzi Wang, Kai Zheng, and Hui Xiong. 2015. Discovering urban functional zones using latent activity trajectories. *IEEE Transactions on Knowledge and Data Engineering* 27, 3 (2015), 712–725.

[34] Quan Yuan, Wei Zhang, Chao Zhang, Xinhe Geng, Gao Cong, and Jiawei Han. 2017. PRED: Periodic region detection for mobility modeling of social media users. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 263–272.

[35] Yu Zheng, Licia Capra, Ouri Wolfson, and Hai Yang. 2014. Urban computing: concepts, methodologies, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)* 5, 3 (2014), 38.

[36] Xiaoyu Zhu, Yifei Yuan, Xianbiao Hu, Yi-Chang Chiu, and Yu-Luen Ma. 2017. A Bayesian Network model for contextual versus non-contextual driving behavior assessment. *Transportation Research Part C: Emerging Technologies* 81 (2017), 172–187.