Learning Embeddings of Intersections on Road Networks

Meng-xiang Wang*, Wang-Chien Lee§, Tao-yang Fu§, Ge Yu*
* School of Computer Science and Engineering, Northeastern University, Shenyang, 110169, P.R. China
§Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA 16802, USA
wmx0425@gmail.com,wlee@cse.psu.edu,txf225@cse.psu.edu,yuge@mail.neu.edu.cn

ABSTRACT

Road network is a basic component of intelligent transportation systems (ITS) in smart city. Informative representation of road networks is important as it is essential to a wide variety of ITS applications. In this paper, we propose a neural network representation learning model, namely *Intersection of Road Network to Vector (IRN2Vec)*, to learn embeddings of road intersections that encode rich information in a road network by exploring geo-locality and intrinsic properties of intersections and moving behaviors of road users. In addition to model design, several issues unique to IRN2Vec, including data preparation for model training and various relationships among intersections, are examined. We evaluate the learned embeddings via extensive experiments on three real-world datasets using three downstream test cases, including prediction of traffic signals and crossings on intersections and travel time estimation. Experimental results show that the proposed IRN2Vec outperforms three existing methods, *DeepWalk*, *LINE* and *Node2vec*, in terms of *F1-score* in predicting traffic signals (22.21% to 23.84%) and crossings (8.65% to 11.65%), and *mean absolute error (MAE)* in travel time estimation (9.87% to 19.28%).

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**; **Learning latent representations**; *Model development and analysis.*

KEYWORDS

Road network, Representation learning, Neural network, Intelligent transportation systems

ACM Reference Format:
Meng-xiang Wang, Wang-Chien Lee, Tao-yang Fu, Ge Yu. 2019. Learning Embeddings of Intersections on Road Networks. In *27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL '19), November 5–8, 2019, Chicago, IL, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3347146.3359075

1 INTRODUCTION

Owing to population growth, urbanization, and technological advances in recent years, efforts for modernizing and smartening the various infrastructures of cities have been planned in many countries, aiming to transform urban areas economically by adopting new information and communication technologies. As an important branch of the smart city, *intelligent transportation systems* (ITS) [15], have received significant interests from academia, industry and governments, owing to the growing demands of solutions to address various transportation issues. The goal is to optimize the transportation systems to improve the efficiency and safety of transportation by addressing various issues, such as traffic congestion, pollution and accidents.

The core of ITS includes functionalities of analyses, mining, predictions, and management built upon data collected from various sources (e.g., road networks, vehicle trajectories, traffic signal control systems, etc.) Among them, road network is arguably the most basic due to the ubiquitous needs in ITS. For instance, road networks are essential to digital maps, (e.g., Google Map and OpenStreetMap [6]), online ride hailing services, (e.g., Uber and Lyft), self-driving cars, and various traffic analysis tasks.

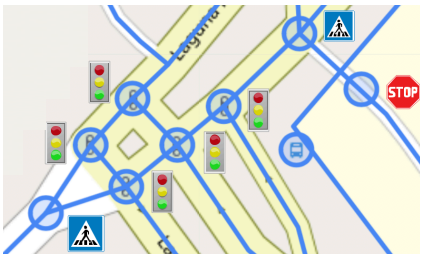

Figure 1: A portion of San Francisco Road Network

Typically, a real-world road network is modeled as a directed graph, which consists of *intersections* (or junctions) as nodes[1] and *road segments* as edges. Additionally, both the intersections (nodes) and road segments (edges) in a road network contain not only geospatial information (e.g., the coordinates), but also various information about road characteristics and transportation facilities, such as traffic signals, road types, number of lanes of road segments, and so on. Take OpenStreetMap (OSM) as an example. Figure 1 shows a small portion of the San Francisco road network where the blue lines are road segments and blue circles are intersections. Also displayed are information such as traffic lights, crossing and bus stops, tagged by volunteers. Indeed, as the main focus of ITS is on road transportation, the road network and associated information play a crucial role in various ITS applications, e.g., travel time estimation [2], destination prediction [17] and intelligent speed adaptation [9]. In fact, it is very important to characterize the intersections and road segments in road networks. For example, 40.1% of all traffic accidents in the United States happen at intersections

[1] In this paper, we use the terms "intersection", "junction" and "node" interchangeably.

SIGSPATIAL '19, November 5–8, 2019, Chicago, IL, USA

ACM ISBN 978-1-4503-6909-1/19/11…$15.00
https://doi.org/10.1145/3347146.3359075

and nearby areas [11]. Identifying intersections/areas with high accident rate may facilitate better understanding of the causes and adaptation of road conditions, e.g., setting a proper speed limit. Properly characterizing and representing intersections and road segments of road networks are important and useful for various ITS applications, especially those formulated as prediction tasks.

In order to achieve good performance in these ITS applications, high-quality features of intersections or road segments that capture intrinsic properties of road networks, such as the topological network structure, geo-locality and homogeneity amongst intersections and road segments are much needed, which conventionally require labor-intensive feature engineering effort by domain experts. Recently, representation learning techniques [27], aiming to automatically learn useful latent feature vectors (also called *embeddings*) of data objects as inputs to machine learning or data mining algorithms, have been developed and well received in the fields of speech recognition [23], computer vision [1], natural language processing (NLP) [7, 22], etc. In this paper, inspired by the successes in these fields, we investigate the issue of representation learning for real-world road networks, aiming to learn useful road network embeddings for general support of various ITS applications. As a first attempt to the aim, to the best knowledge of the authors, we focus on learning of intersection embeddings.[2]

To learn representations of road networks, it is natural to apply existing representation learning methods designed for general networks by treating a road network as a general network. However, simply applying the existing network representation learning methods for road network representation learning is impractical. Firstly, although network representation learning methods aim to capture the topological structure of networks, they do not consider the spatial properties of road networks, e.g., the geo-locality of intersections. Secondly, network representation learning methods usually apply random walk to sample relevant nodes, which fails to capture the moving behaviors of mobile road users who tend to take the shortest paths to move from their sources to destinations. Finally, network representation learning methods usually do not incorporate relationships among intersections, such as homogeneity amongst intersections.

To fill this gap between the conventional network representation learning methods and road networks, we propose a new neural network (NN) model, namely *Intersection of Road Network to Vector (IRN2Vec)*, aiming to embed a road network into a low-dimensional space where each intersection is represented as a latent feature vector. The IRN2Vec model encodes rich features of a road network by exploring the geo-locality and homogeneity of intersections, the moving behaviors of mobile users, and the topological network structure.

To train the *IRN2Vec* model, we design a new learning framework, also called *IRN2Vec* (as shown in Figure 2), which learns latent vectors of the intersections in a given road network. In specific, the IRN2Vec framework consists of two phases: (1) Training data preparation: A new data preparation approach samples the shortest paths by simulating user moving behaviors to prepare training data for learning the IRN2Vec model; (2) Intersection representation learning: The IRN2Vec model which learns intersection embeddings by jointly maximizing the likelihood of predicting various relationships existing among intersections. Furthermore, to be scalable for large-scale real-world datasets, we leverage asynchronous stochastic gradient descent in representation learning in parallel. Finally, we evaluate the effectiveness of the learned intersection embeddings by predictive ITS applications in digital road maps (i.e., missing tag prediction and travel time estimation).

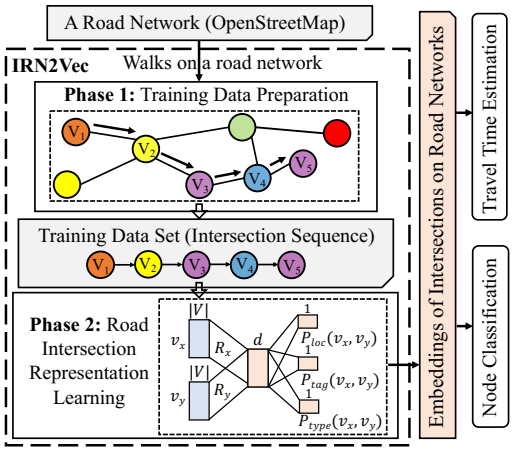

Figure 2: Overview of the IRN2Vec Framework

The main contributions of this paper are summarized as follows.

- **A new problem and novel ideas for intersection representation learning.** In this paper, we analyze the importance of intrinsic properties of the road networks. Accordingly, we propose novel ideas to learn meaningful intersection embeddings. They are able to generally support various ITS applications with better performance.
- **A new representation learning framework for road networks.** We propose a two-phase framework to learn representations of intersections in road networks by encoding various relationships among intersections. The training data preparation algorithm in Phase 1 samples shortest paths to prepare training data. The *IRN2Vec* model in Phase 2 works as a multi-task binary classifier that jointly captures various relationships among intersections.
- **Empirical study on multiple predictive applications using three real-world data.** We conduct extensive experiments to evaluate the performance of the proposed IRN2Vec model on two missing tag prediction tasks and a travel time estimation task using three real-world datasets. The Experimental results show that *IRN2Vec* outperforms a baseline and three general network representation learning methods in terms of *F1-score* in predicting traffic signals (22.21% to 23.84%) and crossings (8.65% to 11.65%), and *mean absolute error* (MAE) in travel time estimation (9.87% to 19.28%).

The remainder of this paper is organized as follows. Firstly, we review the related work in Section 2, and present our problem formulation in Section 3. Then, we present the novel *IRN2Vec* framework in Section 4 and report the evaluation result on real-world data in Section 5. Finally, we conclude the paper in Section 6.

[2]We leave road segments to the future work.

2 RELATED WORK

In past several years, the topic of representation learning has received significant interests in the fields of machine learning and data mining, owing to its advantages in reducing the labor-intensive effort in feature engineering [3]. The goal is to automatically transform raw data into low-dimensional latent vectors (i.e., embeddings), as input features to machine learning and data mining algorithms. Recently, neural network (NN) based representation learning models have achieved great success in various domains, including natural language processing (NLP) [26], speech recognition [5], and computer vision [1], etc.

Inspired by the aforementioned advances, research on representation learning has been extended to network data [8, 16, 19–21, 24]. Earlier works on network representation learning have attempted to partition a graph into different communities by exploring social dimensions [21]. By clustering nodes into different affiliations, the representation for each node is determined by its membership in different affiliations. These studies provide a cluster-centric view of the network, but do not manage to well catch valuable information embedded in the underlying topological network structure. In recent years, several works on network representation learning [8, 16, 20] have been proposed by exploring the network locality of nodes in networks. These works typically assume that nearby nodes (e.g., nodes within a specified *k-hop* neighborhood) are relevant and thus tend to place their learned representations close to each other in the latent feature space. *DeepWalk* [16] learns node representations by sampling nearby nodes via uniform random walk to traverse the network, and then applies *Skip-gram* model [13] to learn node representations by maximizing the likelihood of predicting whether two nodes are within *k-hop* to each other. *Node2vec* [8] also aims to learn node representations but focuses on the issue of sampling the network neighborhood relationship by applying parameterized random walk rather than uniform random walk. The parameterized random walk have two parameters: return parameter p in and out parameter q, which controls the walking process by the possibility of returning back to the previous node or selecting the next node away from the previous node to simulate *BFS* and *DFS* search strategies, respectively. Instead of employing random walk, *LINE* [20] basically samples node relevance in accordance with the frequencies of their 1-hop and 2-hop connectivity. It captures first-order similarity (similarity between adjacent nodes) and second-order similarity (similarity between nodes with common neighbors), separately, to learn two representations of nodes which are used by concatenation. The methods we have discussed above leverage only network structural information to obtain network embeddings. As nodes and edges in real-world networks are often associated with *attributes*. Therefore, it is desirable for network embedding methods to learn from the rich content in node attributes and edge attributes. Text-Associated DeepWalk (*TADW*) [24] is proposed to incorporate text features of nodes into network representation learning, and *CENE* [19] is a network embedding method which jointly models network structure and textual content in nodes.

Different from existing works, the proposed *IRN2Vec* explore the intrinsic geo-spatial properties of nodes and their relationships in road networks so as to learn an effective road network representation. By simultaneously learning the topological network structure, geo-closeness and node characteristics targeted in our model via a simple yet effective shortest-path sampling approach that resembles user moving behaviors, the embeddings learned by IRN2Vec can be generally employed to support multiple ITS applications.

3 PRELIMINARIES

In this section, we define important terms and concepts, formulate the tackled problem, and discuss the challenges.

3.1 Road Network and Intersection Sequence

Definition 1. *Road Network.* A road network is a directed graph $G = (V, E, \Psi)$, where V is a set of nodes (i.e., intersections);[3] $E \subseteq V \times V$ is a set of directed edges that denote road segments. $\Psi : V \rightarrow A$ is a characteristics function of nodes, where a node $v \in V$ can be described by a set of geo-spatial characteristics A, i.e., $\Psi(v) \in A$. Regarding the sources of geo-spatial characteristics set A, we further discuss them in Section 4.1.

Definition 2. *Intersection Sequence.* An intersection sequence $S_I = \{v_1, v_2, ..., v_{|S_I|}\}$ is a sequence of intersections (identified by unique IDs), where v_i is ith intersection in S_I generated by some sampling paths methods.

3.2 Problem Definition and Analysis

The goal of this work is to learn a representation of each node in a road network. Our idea is to explore the intrinsic geo-spatial properties of nodes in a road network to learn embeddings of intersections for use as input features to various predictive and analytical ITS applications. We formally define the problem below.

Definition 3. *Representation Learning on Road Networks.* Given a road network, denoted as a directed graph $G = (V, E, \Psi)$, road network representation learning learns a function $f: V \rightarrow \mathbb{R}^d$ that projects each intersection/node $v \in V$ to a vector in a d-dimensional latent space $\mathbb{R}^d$, where $d \ll |V|$.

In this work, we propose a neural network model, *IRN2Vec*, to tackle the representation learning problem on road networks. Our goal is to explore the rich information in the road network so as to learn an effective road network representation in support of various ITS applications with higher accuracy. To achieve this goal, we face the following challenges: (1) Model design. A well designed neural network model is essential for effective and efficient learning. (2) Geo-spatial characteristics. Different from general networks, a road network is characterized by specific geo-spatial information, e.g., traffic lights and stop signs on intersections, and various types of intersections, e.g., T-junction and dead ends. Thus, how to explore the characteristics and similarity between intersections to learn representations of intersections requires careful study. (3) Training data preparation. Training data need to be prepared and tailored based on the learning logic behind the proposed *IRN2Vec* model. A good sampling method is needed to capture the user moving behaviors in road networks.

4 THE IRN2VEC FRAMEWORK

In this section, we first introduce the *data preprocessing* and then the proposed *IRN2Vec* framework. The data preprocessing includes

[3] Without loss of generality, we also consider terminal/end of a road as an intersection.

road network extraction and intersection information extraction (i.e., coordinates, types and tages) (Section 4.1). Then, we introduce the *IRN2Vec* framework which consists of two phases: 1) *Training data preparation* and 2) *Representation learning* (see Figure 2). We first introduce the proposed *IRN2Vec* model, the training objective and technical issues. According the model, we then detail the preparation of the training data. To collect training data that reflect moving behaviors of mobile road users in road networks, we propose to sample the network by exploring *shortest paths* in road networks. Through the sampling process, we prepare positive and negative samples for training the model in Phase 2.

4.1 Data Preprocessing

In this work, we extract road networks and user-generated tags associated with of intersections in road networks from the OpenStreetMap (OSM), which is a free editable and downloadable map website [6]. We download the raw OSM data of three cities (i.e., San Francisco, Porto and Tokyo) from OSM website to generate the road networks.

Moreover, *intersection tags* are extracted from the "Highway" tag in OSM data, which is the primary tag used for any kind of streets or ways. As we argued previously, intersections share some common characteristics among them. Note that intersections may be similar due to the same traffic controls or *intersection types*. As intersection types are not handily available as tags in OSM, we derive the *N-way types* of intersections in terms of the number of ways being intersected.

4.1.1 Intersection Tags. The OpenStreetMap (OSM) mainly includes three fundamental element types: *Nodes*, *Ways*, and *Relations*. Among them, *Nodes* defines the position of points in the geo-space, *Ways* denote the roads or areas, while *Relations* describe the relationship between elements. Since the focus of this paper is on intersections in road networks, we pay attention to *Nodes* in the OSM data. A node consists of a unique ID, coordinate (i.e., longitude and latitude) and several key-value pairs of tag attributes, e.g., <node id="25496583" lat="51.5173639" lon="-0.140043"> <tag k="highway" v="traffic_signal"/> </node>. In this paper, we select 10 most frequent tags of nodes, including "give_way", "mini_roundabout", "bus_stop", "turning_circle", "crossing", "traffic_signal", "stop_sign", "speed_camera", "motorway_junction" and "turning_loop". Figure 3 shows some of these tags from the OSM data.

(a) Traffic signal (b) Crossing sign (c) Turn circle (d) Stop sign

Figure 3: Samples of Intersection Tags in OSM

4.1.2 N-way Types. In the real world, different types of intersections, such as T-juction and X-junction, exist. Thus, one natural way to characterize intersections is by the number of road segments (or road ways) that are intersected at it. However, in the OSM data, there is no tag associated with nodes to describe the junction types. In this paper, we calculate the number of road segments passing through an intersection as the *N-way* type of the intersection (N=2,3,..,6). For example, a T-junction between three road segments, as shown in Figure 4(b). In the real world, 5-way and 6-way intersections are less common but still exist, especially in suburban areas with non-rectangular blocks.

(a) X-junction (b) T-junction (c) 5-way junction

Figure 4: Samples of Road Crossing Types

4.2 Representation Learning

As discussed, our idea to learn latent intersection vectors in a road network lies in capturing shared inherent characteristics between two intersections, e.g., same intersection tag or N-way type. The sharing of common characteristics can be considered as a *relationship*. In addition, the geo-spatial locality can be also considered as a relationship, i.e., the distance between two intersections is within certain meters. Thus, we propose to develop an NN model that jointly predicts a set of targeted relationships between any given pair of intersections for intersection representation learning. More specifically, as shown in Figure 5, our *IRN2Vec* model is a multi-task binary classifier that takes a pair of intersections $v_x, v_y \subseteq V$ as the inputs to predict the three aforementioned relationships between them, i.e., geo-spatial locality, same intersection tag and same N-way type. In this model, the input layer takes in two one-hot vectors $\vec{v_x}$ and $\vec{v_y}$ of length $|V|$, representing node v_x and v_y, respectively. In the latent layer, $\vec{v_x}$ and $\vec{v_y}$ are transformed into latent vectors ${R_x}'\vec{v_x}$ and ${R_y}'\vec{v_y}$, where R_x and R_y are two $|V| \times d$ matrices representing the transformation, ${R_x}'$ and ${R_y}'$ are their transpose matrices, and d is the dimensionality of the hidden space. Next, we use inner product followed by a *Sigmoid* function to predict whether two intersections v_x and v_y have a specific relationship. More specifically, to present these operations in a neural network, shown in Figure 5, we apply *Hadamard* function, i.e., element-wise multiplication, to aggregate the two vectors which is denoted by ${R_x}'\vec{v_x} \odot {R_y}'\vec{v_y}$, and then we apply the Identity function for activation. Finally, the output layer, taking *Summation* as the input function and *Sigmoid* function for activation, computes $Sigmoid({R_x}'\vec{v_x} \cdot {R_y}'\vec{v_y})$ respectively to predict three relationships, correspondingly measured by (1) the probability $P_{loc}(v_x, v_y)$ for v_x and v_y to be located within a neighborhood of certain distance, (2) the probability $P_{tag}(v_x, v_y)$ for v_x and v_y to have the same intersection tag, and (3) the probability $P_{type}(v_x, v_y)$ for v_x and v_y to have the same N-way type. The three joint probabilities are derived as follows.

$$P_{loc}(v_x, v_y) = \sigma(\sum {R_x}'\vec{v_x} \cdot {R_y}'\vec{v_y}) \quad (1)$$

$$P_{tag}(v_x, v_y) = \sigma(\sum {R_x}'\vec{v_x} \cdot {R_y}'\vec{v_y}) \quad (2)$$

$$P_{type}(v_x, v_y) = \sigma(\sum {R_x}'\vec{v_x} \cdot {R_y}'\vec{v_y}) \quad (3)$$

where $\sigma(z) = \frac{1}{1+e^{-z}}$ is the *Sigmoid* function. In our framework, we make R_x and R_y to be the same matrix, which consists of all intersections' vectors, i.e., each row of this matrix denotes the vector for an intersection. In the training process, if intersections v_x and v_y are observed in the training data to satisfy one of the targeted

prediction tasks, $R_x{}'\vec{v_x}$ and $R_y{}'\vec{v_y}$ are moved closer in the latent space. Otherwise, they are moved away in the latent space.

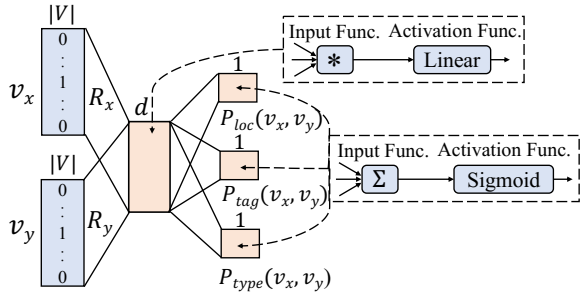

Figure 5: The IRN2Vec Model

The learning of *IRN2Vec* model parameters (i.e., intersection vectors) is realized by setting multiple optimization objectives and thus is critical to set the objective functions properly. To train *IRN2Vec*, a training data set D, which contains training data entries in the form of $\langle v_x, v_y, S_{loc}(v_x, v_y), S_{tag}(v_x, v_y), S_{type}(v_x, v_y)\rangle$ is extracted from the road network in the training data preparation phase (to be discussed in the next section). In a training data entry, $S_{loc}(v_x, v_y)$, $S_{tag}(v_x, v_y)$ and $S_{type}(v_x, v_y)$ are boolean values, indicating whether intersections v_x and v_y are located within k-meter in the road network, and whether v_x and v_y have the same intersection tag and N-way type, respectively. With the training data set D, the *IRN2Vec* model is trained by the backpropagation training algorithm in conjunction with asynchronous stochastic gradient descent in parallel. It goes backwards to adjust the weights in R_x and R_y for each entry s in D, attempting to maximize the objective function O, which is the production of $O_{loc}(v_x, v_y)$, $O_{tag}(v_x, v_y)$ and $O_{type}(v_x, v_y)$ which are derived as follows.

$$O_{loc}(v_x, v_y) = \begin{cases} P_{loc}(v_x, v_y) & if\ S_{loc}(v_x, v_y) = 1 \\ 1 - P_{loc}(v_x, v_y) & if\ S_{loc}(v_x, v_y) = 0 \end{cases} \quad (4)$$

$$O_{tag}(v_x, v_y) = \begin{cases} P_{tag}(v_x, v_y) & if\ S_{tag}(v_x, v_y) = 1 \\ 1 - P_{tag}(v_x, v_y) & if\ S_{tag}(v_x, v_y) = 0 \end{cases} \quad (5)$$

$$O_{type}(v_x, v_y) = \begin{cases} P_{type}(v_x, v_y) & if\ S_{type}(v_x, v_y) = 1 \\ 1 - P_{type}(v_x, v_y) & if\ S_{type}(v_x, v_y) = 0 \end{cases} \quad (6)$$

In Equation (4), (5) and (6), the functions $O_{loc}(v_x, v_y)$, $O_{tag}(v_x, v_y)$ and $O_{type}(v_x, v_y)$ quantify how *IRN2Vec* correctly predicts $S_{loc}(v_x, v_y)$, $S_{tag}(v_x, v_y)$ and $S_{type}(v_x, v_y)$ for a data entry s, respectively. In specific, for a training data entry $s = \langle v_x, v_y, S_{loc}(v_x, v_y), S_{tag}(v_x, v_y), S_{type}(v_x, v_y)\rangle$, $O_{loc}(v_x, v_y)$ aims to maximize $P_{loc}(v_x, v_y)$, when $S_{loc}(v_x, v_y)$ is 1, and minimize $P_{loc}(v_x, v_y)$, otherwise. Similarly, $O_{tag}(v_x, v_y)$ and $O_{type}(v_x, v_y)$ aim to maximize $P_{tag}(v_x, v_y)$ and $P_{type}(v_x, v_y)$, when $S_{tag}(v_x, v_y)$ and $S_{type}(v_x, v_y)$ is 1, respectively, and minimize $P_{tag}(v_x, v_y)$ and $P_{type}(v_x, v_y)$, otherwise.

To ease the computation in the optimization process, we maximize $\log O_{loc}(v_x, v_y)$, $\log O_{tag}(v_x, v_y)$ and $\log O_{type}(v_x, v_y)$ rather than $O_{loc}(v_x, v_y)$, $O_{tag}(v_x, v_y)$ and $O_{type}(v_x, v_y)$. The objective functions are shown below.

$$\log O_{loc}(v_x, v_y) = S_{loc}(v_x, v_y) \log P_{loc}(v_x, v_y) + [1 - S_{loc}(v_x, v_y)] \log[1 - P_{loc}(v_x, v_y)] \quad (7)$$

$$\log O_{tag}(v_x, v_y) = S_{tag}(v_x, v_y) \log P_{tag}(v_x, v_y) + [1 - S_{tag}(v_x, v_y)] \log[1 - P_{tag}(v_x, v_y)] \quad (8)$$

$$\log O_{type}(v_x, v_y) = S_{type}(v_x, v_y) \log P_{type}(v_x, v_y) + [1 - S_{type}(v_x, v_y)] \log[1 - P_{type}(v_x, v_y)] \quad (9)$$

The overall objective function O is defined as follows.

$$O = \sum_{s \subseteq D} \{\alpha \log O_{loc}(v_x, v_y) + \beta \log O_{tag}(v_x, v_y) + (1 - \alpha - \beta) \log O_{type}(v_x, v_y)\} \quad (10)$$

where α and β are weighing parameters.

We apply asynchronous stochastic gradient descent in parallel to maximize the objective function O. Specifically, for each training data entry, it goes backwards to adjust the weights of intersections v_x and v_y in $R_x{}'\vec{v_x}$ and $R_y{}'\vec{v_y}$ based on the gradients, respectively, as shown below.

$$R_x{}'\vec{v_x} := R_x{}'\vec{v_x} + [\alpha d \log O_{loc}(v_x, v_y) + \beta d \log O_{tag}(v_x, v_y) + (1 - \alpha - \beta) d \log O_{type}(v_x, v_y)]/dR_x{}'\vec{v_x} \quad (11)$$

$$R_y{}'\vec{v_y} := R_y{}'\vec{v_y} + [\alpha d \log O_{loc}(v_x, v_y) + \beta d \log O_{tag}(v_x, v_y) + (1 - \alpha - \beta) d \log O_{type}(v_x, v_y)]/dR_y{}'\vec{v_y} \quad (12)$$

4.3 Training Data Preparation

According to the need of model training, in Phase 1, we sample pairs of intersections in the road network to prepare training data entries for the *IRN2Vec* model in the form of $s = \langle v_x, v_y, S_{loc}(v_x, v_y), S_{tag}(v_x, v_y), S_{type}(v_x, v_y)\rangle$. Our design decision represents a trade-off in data collection between the computational efficiency (i.e., sampling instead of enumeration) and the quality (i.e., the training data should cover as many intersections as possible). Conventionally, network representation learning techniques adopt random walk to sample the network structure. However, we argue that random walk based sampling is not suitable for road networks because mobile road users do not move randomly. Therefore, we adopt shortest path, which is more aligned to the moving behaviors of mobile road users, to sample the road networks. More specifically, we apply *Dijkstra* (using a Min-heap as priority queue) to generate shortest paths (i.e., intersection sequences) between randomly selected intersection pairs. Then, we generate training data by using the generated intersection sequences. We empirically show that shortest path sampling approach soundly outperforms random walk sampling on multiple applications.

Along a sampled shortest path, which can be seen as an intersection sequence S_I , we extract coordinates, tags and N-way types of intersections to prepare the training data for our exploration of geo-spatial locality, same-tag, and same-type relationships, based on the ground truth. For locality, conventional graph embedding techniques typically adopt a notion of *hop-based neighborhood*. Given an example of intersection sequence $S_I = \{v_1, v_2, v_3, v_4, v_5\}$,

where a sliding window of 2-hop is used, we have three neighborhood windows $w_1 = \{v_1, v_2, v_3\}$, $w_2 = \{v_2, v_3, v_4\}$ and $w_3 = \{v_3, v_4, v_5\}$. However, this definition of window size does not capture the geo-spatial characteristics. As shown in Figure 6, intersections v_3 and v_5 are distant but they are treated as close by a 2-hop window. Generally speaking, near things are more related than distant things. Therefore, in *IRN2Vec*, we explore *distance-based neighborhood*. For the same intersection sequence in Figure 6, where a 50m window is used to sample along the shortest path, resulting in $w_1 = \{v_1, v_2, v_3\}$, $w_2 = \{v_2, v_3\}$, $w_3 = \{v_3, v_4\}$, and $w_4 = \{v_4, v_5\}$.

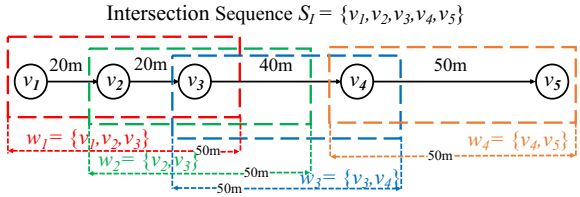

Figure 6: An Example of Distance-based Neighborhood

Accordingly, for each pair of intersections v_x and v_y within a window, we create a positive sample reflecting their geo-spatial locality. Then, we also consider that whether v_x and v_y share the same intersection tag and type. For instance, in the first window $\{v_1, v_2, v_3\}$, we create positive training samples $\langle v_1, v_2, 1, same_{tag}(v_1, v_2), same_{type}(v_1, v_2)\rangle$, $\langle v_1, v_3, 1, same_{tag}(v_1, v_3), same_{type}(v_1, v_3)\rangle$ and $\langle v_2, v_3, 1, same_{tag}(v_2, v_3), same_{type}(v_2, v_3)\rangle$, and so on, where $same_{tag}(v_x, v_y)$ and $same_{type}(v_x, v_y)$ are determined by checking whether v_x and v_y have the same intersection tag and the same N-way type, respectively.

In addition to positive data, the model also needs negative data for learning. Thus, while generating positive samples via shortest paths, we also generate negative data following the idea of *Negative Sampling* in Word2Vec [14]. For each sampled positive data entry, we generate *ns* negative training data entries by randomly replacing one of the two values with either $\tilde{v_x}$ or $\tilde{v_y}$, where $\tilde{v_x}$ or $\tilde{v_y}$ are randomly selected intersections.

5 EXPERIMENT

In this section, we conduct extensive experiments to evaluate IRN2Vec using three real-world road network data, against several models for network representation learning, including one baseline and three existing works, for comparison. We also perform sensitivity tests on parameters of IRN2Vec and examine several issues in IRN2Vec. Finally, we evaluate IRN2Vec and those models by three downstream applications: *traffic signal classification* and *crossing classification* on intersections, and *travel time estimation* for moving paths in road networks.

5.1 Datasets and Compared Methods

Our evaluation involves two types of real-world datasets, i.e., road network datasets and trajectory datasets. Trajectory datasets are used to provide the moving paths and their corresponding travel times (i.e., the groundtruths) for travel time estimation.

5.1.1 Road Network Data. As mentioned above, we extract road networks (with corresponding geo-spatial characteristics of intersections) from the downloadable raw data from OpenStreetMap website [6]. Our evaluation involves three road networks, including San Francisco, Porto and Tokyo. Some statistics of the road networks extracted are summarized in Table 1.

Table 1: Statistics of Road Network Datasets

Name	#Intersections	#Road segments	#Tags
San Francisco	58,404	76,744	14,832
Porto	119,769	154,128	8,878
Tokyo	217,117	305,874	24,215

5.1.2 Trajectory Data. We collect three publicly accessible trajectory datasets, that have GPS sample points in trajectories. Some statistics of these trajectory datasets are summarized in Table 2.

Table 2: Statistics of Trajectory Datasets

Name	#GPS Points	#Trajectories	Avg. Time Gap
San Francisco	11,219,955	443,406	14.12sec.
Porto	74,269,739	1,233,766	15.11sec.
Tokyo	68,275,641	273,046	15.00sec.

San Francisco [18] San Francisco taxi data collects 11 million GPS points from 536 taxis running in San Francisco for a 30 days period, and every taxis have two status (occupied or not). We select the sequences of GPS points by occupied taxis to form trajectories and remove trajectories with less than 5 sample points, which yields 0.4 million trajectories.

Porto [4] Porto taxi data collects 1.7 million taxi trajectories (containing 74 million GPS sample points) from 442 taxis running in Porto City over a complete year. Each taxi reports its location every 15 second. We remove trajectories with less than 10 sample points, which yields 1.23 million trajectories.

Tokyo [25] OpenPFLOW data collects 68 million GPS sample points from 617,040 users in Tokyo taking different vehicles, such as bike, train and car. We consider sequences of GPS sample points by users taking bicycle and bus and segment them into trajectories when there is no sample point for 45 seconds or more (which is longer than about 99% time gaps). Then, we remove trajectories with less than 5 sample points, which yields 0.27 million trajectories.

We evaluate the performance of *IRN2Vec* against one baseline and three well known network representation learning methods.

Unique ID (UID) directly uses the unique IDs of intersections in a network to represent each intersection as an one-hot vector. Without learning embeddings, this method serves as the baseline.

DeepWalk [16] learns d-dimensional node vectors by capturing node pairs within w-hop neighborhood via uniform random walks in the network.

LINE [20] learns node vectors by considering first and second order proximities of nodes in a network separately. We use $d/2$ dimensions to capture the first-order information, and another $d/2$ dimensions to capture the second-order information. They together form a d-dimensional node vector.

Node2vec [8] is generalized from DeepWalk. It learns d-dimensional node vectors by capturing node pairs within w-hop neighborhood via parameterized random walks in the network.

IRN2Vec is the new intersection embedding model proposed in this paper for road networks. To validate our idea of adopting

shortest paths in training data preparation, we also make a comparison with a variant of IRN2Vec based on random walks (denoted by **IRN2Vec$_R$**.)

5.2 Traffic Signal Classification

In this section, we evaluate the models by a binary classification task which infers whether an intersection in a road network has a traffic signal on it. We first introduce the experimental setup, including the process of classification, the preparation of labeled datasets and the default settings of parameters in the compared models. Then, we perform sensitivity tests on parameters of IRN2Vec to determine their default settings and study the unique issues in IRN2Vec. Finally, we show the experimental results.

5.2.1 Experimental Setup. After learning the intersection embeddings in a road network,[4] we select 2,582, 1,646 and 8,470 intersections, which have the "traffic signal" tag, from San Francisco, Porto and Tokyo datasets, respectively, as positive samples, and then randomly select equal number of other intersections (without the "traffic signal" tag) as negative samples to form balanced labeled datasets. In each experiment, for a labeled dataset, we randomly split it to 90% and 10% as the training set and test set, respectively. We use the training set to train a classifier using the linear SVM model and evaluate the performance using the test set. We report the *F1-score* as the metric for evaluation.

Regarding default parameters, the dimensionality of intersection embeddings, d, is set to 128 for all methods, which generally achieves the best performance. The negative (vs positive) sampling rate ns is set to 5, and the learning rate is 0.025 in all representation learning models. The number of sampled shortest paths per intersection wn in IRN2Vec is set to 1280. The window size ws in IRN2Vec is set to 500m. The window size w in DeepWalk and Node2vec is set to 5 because they achieve good performance. For Node2vec, the two parameters p and q for parameterized random walk are set to 1 and 4, respectively. The number of training data sampling or the length of random walk for generating training data varies for each model and for each dataset, in order to obtain converged results.

IRN2Vec is implemented in C. All experiments are run on the Ubuntu 18.04 operating system with an Intel Core i5-8400 CPU and an NVIDIA GTX 1080 GPU.

5.2.2 Parameter Analysis in IRN2Vec. Parameters settings in our model affect the representation learning and the application performance. To decide the default settings, we vary the values of important parameters to observe how the performance, i.e., F1-score, changes in traffic signal classification. The results are shown in Figure 7.

Dimensionality (d). Generally speaking, a small dimensionality is not sufficient to capture the information embedded in the various relationships between intersections, but a large dimensionality may lead to noises and easily cause over fitting. Figure 7(a) shows that setting the dimensionality d at 128 is reasonable. In these three datasets, their improvements are relative small (about 0.79% to 1.56%) when the dimension rises from 128 to 256.

[4]We remove all "traffic signal" tags while learning intersection embeddings.

No. of Negative Samples (ns). We test various number of negative samples per positive sample and observe that, as shown in Figure 7(b), the best performance is achieved when ns is set to 5 in the three datasets.

No. of Shortest Paths Per Node (wn). Generally speaking, a larger number of shortest paths sampled generates more training data for representation learning which usually lead to better performance in applications. Figure 7(c) suggests that when the wn is increased, the performance continues to improve and then converges when wn is set to 1280.

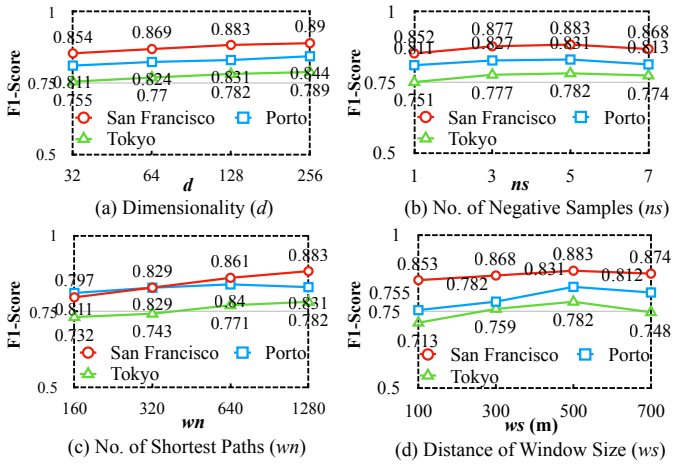

Figure 7: Parameter Analysis

Distance of Window Size (ws). Figure 7(d) shows that the performance improves while the ws increases from 100m to 700m (with a step of 200m). The best performance is achieved when ws is 500m.

Weights of α and β. Finally, we evaluate the weights in the overall objective function in *IRN2Vec*: α, β and $1-\alpha-\beta$ for weighing $O_{loc}(v_x, v_y)$, $O_{tag}(v_x, v_y)$ and $O_{type}(v_x, v_y)$, respectively, as derived in Equation (10). We perform grid search to tune the performance. As shown in Figure 8, the performance of these three datasets do not change much when α is set between 0.75 to 0.8 and β is set to between 0.05 to 0.1. Consistently, the best performance is achieved when α= 0.8 and β= 0.1. Note that when $\alpha = 0.85$ and $\beta = 0.15$ (i.e., does not consider the same-type relationship between intersections in IRN2Vec), the experiments have the worst results, which suggests that the same-type relationship between intersections is usually useful for representation learning and traffic signal classification.

5.2.3 Study of Unique Issues in IRN2Vec. As discussed, several unique issues arise in the design of IRN2Vec. In this section, we examine the following issues: 1) shortest paths vs. random walks in training data preparation; 2) distance-based vs. hop-based neighborhoods; and 3) ablation study on tags and N-way types.

To validate our argument that shortest paths better reflect the moving behaviors of mobile road users than random walks, we compare IRN2Vec with its random walk variant IRN2Vec$_R$, i.e., all the parameter settings in both are kept the same except for the sampling paths. As shown in Table 3, IRN2Vec is better than IRN2Vec$_R$ by 8.88% to 16.72% in traffic signal classification, which suggests that using shortest paths for training data preparation

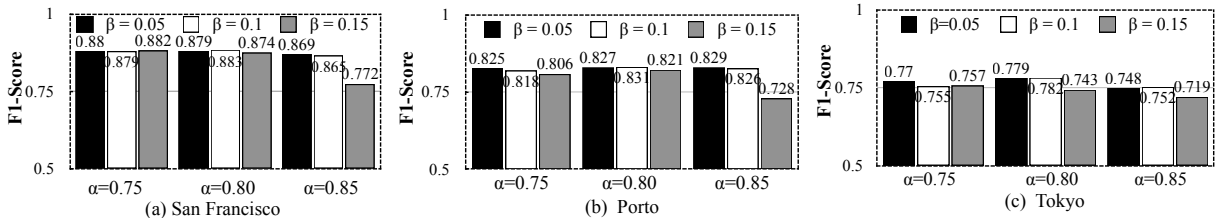

Figure 8: Evaluation of α and β

is significantly more effective than using random walks because mobile users naturally follow the more economic shortest paths while moving on roads rather than randomly walks. Thus, capturing samples by following user moving behaviors and geomatric distance embedded in shortest paths, IRN2Vec achieves better performance in traffic signal classification, while IRN2Vec$_R$ only captures the contextual information of nodes.

Table 3: Performance of Traffic Signal Classification

	San Francisco	Porto	Tokyo
UID	0.532	0.511	0.505
DeepWalk	0.711	0.661	0.639*
LINE	0.573	0.553	0.557
Node2vec	0.713*	0.680*	0.624
IRN2Vec$_R$	0.811	0.768	0.670
IRN2Vec	**0.883(23.84%)**	**0.831(22.21%)**	**0.782(22.38%)**

To validate our idea of adopting distance to capture geo-locality among intersections, we compare the proposed `distance-based neighborhood` (with ws=500m) for road networks representation learning against the `hop-based neighborhood` (with hop number k=5), which is widely used in conventional network representation learning. Figure 9(a) shows that `distance-based neighborhood` outperforms `hop-based neighborhood` in all three datasets (improving by 4.13% to 8.16%), which suggests that `distance-based neighborhood`, capturing the geo-spatial characteristics of the road network, is a more natural choice for road network representation learning.

Finally, we perform an ablation study to find the impact of same-tag and same-type relationships between intersections upon the performance of the proposed IRN2Vec. [5] In this study, we consider three vaiants: 1) `Complete` is the complete version of IRN2Vec with all factors considered; 2) `NoTag` is an IRN2Vec variant without incorporating the same-tag relationship in representation learning; and 3) `NoType` is an IRN2Vec variant without considering the same-type relationship in representation learning. Figure 9(b) shows that `Complete` outperforms `NoTag` and `NoType` for about 3.31% to 14.38% in the three datasets, which suggests that taking both tags and N-way types of intersections into account are beneficial for IRN2Vec to learn better intersection representations for traffic signal classification.

In Figure 9(b), we also observe that `NoType`, which uses tags but not intersection types, performs the worst in all datasets. This means the N-way intersection types we derived is more informative than the tags generated by OSM volunteers. This may be due to the incomplete/inaccurate tag information associated with intersections in the OSM data. We dig further into this issue and find significant amount of missing tags in open street maps. For example, there are only 243 stop signs in San Francisco road network, which is unreasonably few. To investigate this issue and to validate our idea of using tags in learning, we collect additional 3,727 stop signs from the official website of San Francisco government [10] as supplementary stop sign tags of intersections in the San Francisco road network for representation learning. The new result is improved for about 3.2% compared with the original one without the augmented stop sign tags. This supports our argument that the information of intersection tags, if available and more accurate, is indeed useful for representation learning. Thus, the IRN2Vec embeddings have room to get better as the volunteer-generated tags grow or official/proprietary geographical information describing the road networks is used.

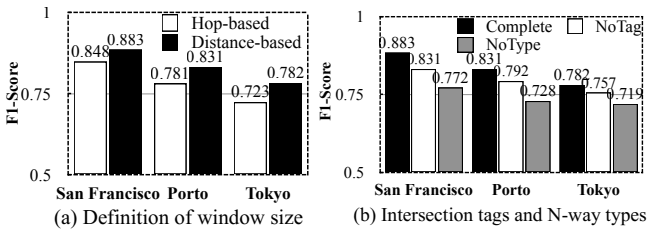

Figure 9: Unique Issues in Traffic Signal Classification

5.2.4 Evaluation of Model. The performance of all evaluated models on the task of traffic signal classification is reported in Table 3. First of all, we observe that all the embedding-based methods outperform the UID method, which indicates that the idea of using intersection embeddings in traffic signal classification is indeed effective, as the intersection embeddings capture the inherent information in road networks while the UID (node identifier) approach treats intersections are independent from each other and miss the meaningful relationships among them. Moreover, IRN2Vec soundly outperforms all the compared models. The improvement ratio (compared with the best of these existing models, marked by $*$) ranges from 22.21% to 23.84% in the three datasets. Moreover, the results also show that LINE consistently performs the worst, while DeepWalk and Node2vec have similar performance, closer but still inferior to IRN2Vec$_R$. It may suggest that LINE, only capturing the information of 1-hop or 2-hop neighborhood of nodes in networks, is not suitable for road network representation learning, which has a broader neighborhood and more general relationships. Moreover, the result also validates our claim that using shortest path sampling

[5]Note that the shortest path sampling and distance-based neighborhood are essential to IRN2Vec, so they are not included in this study.

is significantly better than random walk sampling approaches (i.e., DeepWalk, Node2vec and IRN2Vec$_R$) due to the inherent nature of movements on road networks.

5.3 Crossing Classification

To demonstrate the robustness of IRN2Vec, we evaluate the models by performing an alternative node classification task which infers whether an intersection in a road network has a crossing on it. The experimental setup is the same with traffic signal classification described previously, except for having the "crossing" tags as positive samples (5,021, 6,934 and 14,377 intersections in San Francisco, Porto and Tokyo datasets, respectively.) Again, all "crossing" tags in road networks are removed in learning. Note that, like in traffic signal classification, parameter settings have been tuned but the result are similar, so we choose the same settings as in the traffic signal classification. We skip the details of parameter tuning due to space constraint.

Table 4: Performance of Crossing Classification

	San Francisco	Porto	Tokyo
UID	0.527	0.516	0.509
DeepWalk	0.692	0.635	0.718
LINE	0.554	0.567	0.577
Node2vec	0.717*	0.644*	0.734*
IRN2Vec$_R$	0.726	0.667	0.719
IRN2Vec	**0.779(8.65%)**	**0.719(11.65%)**	**0.800(8.99%)**

The performance of all evaluated models on the task of crossing classification is reported in Table 4. As shown, IRN2Vec robustly outperforms all the compared models, with improvement ratios (compared with the best of these existing models, marked by $*$) ranging from 8.65% to 11.65% in the three datasets.

5.4 Travel Time Estimation

Next, we demonstrate that the intersection embeddings learned by IRN2Vec is also applicable for estimating travel time on given moving paths in road networks. In the following, we firstly introduce the experimental setup, including the experimental flow of travel time estimation, road network mapping, and then we show the experimental results of IRN2Vec in comparison with other methods.

5.4.1 Experimental Setup. Travel time estimation is a regression task to predict the travel time of a given moving path in road networks. Publicly available trajectory datasets, including San Francisco, Porto, and Tokyo, are processed for preparation of training and used as groundtruth for testing.[6] To avoid the regression model to be trained by simply counting the number of sample points in a trajectory to exploit the relatively fixed time gap between consecutive sample points for travel time estimation, we do not directly use raw trajectories for travel time estimation. Instead, we map raw trajectories onto their underlying road network to obtain sequences of intersections (which depict corresponding moving paths of the trajectories in the road network) by employing a state-of-the-art road network matching techniques (i.e., Barefoot [12], a Hidden Markov model based model). Thus, this task aims to estimate the travel time of a given moving path (i.e., a sequence of intersections). To achieve the goal, we apply a Long Short-Term Memory (LSTM) model followed by three fully-connected layers (with dimensions 128, 128 and 1, respectively) as a regression model. More specifically, given a moving path as the input, the LSTM model takes each intersection in the moving path (replaced by its corresponding learned embeddings) as a state to estimate its travel time. We use *mean absolute error (MAE)* between the predicted result and the ground truth extracted from raw trajectories (in seconds) as the metric to evaluate the travel time estimation. Regarding the parameter settings, we use the same parameter values used in the experiment of traffic signal classification.

Table 5: Performance of Travel Time Estimation

	San Francisco	Porto	Tokyo
UID	75.62	169.49	106.39
DeepWalk	64.97	163.17	95.74*
LINE	67.05	161.99	99.86
Node2vec	61.37*	160.65*	96.31
IRN2Vec$_R$	58.38	149.18	92.43
IRN2Vec	**49.54(-19.28%)**	**141.96(-11.63%)**	**86.29(-9.87%)**

5.4.2 Evaluation of models. The result of travel time estimation by all evaluated models is reported in Table 5. As shown, we observe that all the embedding-based methods outperform the UID method, which has yet certified that the idea of using intersection embeddings in travel time estimation is indeed effective. Morever, the results show that IRN2Vec outperforms all the compared methods. The improvement ratios of MAE (compared with the best of these existing models, marked by $*$) are ranging from 9.87% to 19.28% in the three datasets. The results demonstrate that the intersection embeddings learned by IRN2Vec model is able to capture the intrinsic properties of the road networks and thus improve the performance. Finally, IRN2Vec again performs better than IRN2Vec$_R$ (by about 4.85% to 15.13%) in the three datasets, demonstrating the importance of capturing the moving behaviors of mobile road users (i.e., by shortest path sampling) to achieve effective performance in road network applications.

5.5 Test on Generality and Robustness

As mentioned earlier, this work aims to learn useful road network embeddings for *general support* of various ITS applications. To validate this idea, we learn intersection common embeddings by IRN2Vec, denoted by `Common`, for common use in the three applications (i.e., traffic signal classification, crossing classification and travel time estimation), in contrast to the fine-tuned intersection embeddings optimized for individual applications. As the missing tag detection tasks (i.e., traffic signal and crossing classifications) heavily rely on tag information, we go with the extreme and simply do not use any tags in model training in order to show the robustness and generality of our intersection embeddings. We choose the same parameter settings used in the traffic signal classification, and set α=0.8 and β=0.2 which archieves good performance by parameter tuning. We use the obtained embedding as common embeddings in all applications, with the same parameter settings. The performance of Common and the best model among compared

[6] Note that the raw trajectory data contains the arrival time information of GPS sample points in a trajectory which is not available for testing.

Table 6: Evaluation of Common Embeddings in Applications

	Traffic Signal Classification (F1-score)		Crossing Classification (F1-score)		Travel Time Estimation (MAE)	
	Best Baseline	Common	Best Baseline	Common	Best Baseline	Common
San Francisco	0.713	**0.831(16.55%)**	0.717	**0.754(4.91%)**	61.37	**52.94(-13.74%)**
Porto	0.680	**0.792(16.47%)**	0.644	**0.701(8.85%)**	95.74	**89.37(-6.65%)**
Tokyo	0.639	**0.757(18.46%)**	0.734	**0.788(5.40%)**	160.65	**147.62(-8.11%)**

existing models, denoted by `Best Baseline`, is reported in Table 6. The results show that `Common` is robustly outperforms `Best Baseline` in all experiments. In specific, the improvement ratios in terms of F1-score are ranging from 16.55% to 18.46% in traffic signal classification, 4.91% to 8.85% in crossing classification, and the improvement ratio in terms of MAE are 6.65% to 13.74% in travel time estimation, respectively. These results demonstrate that, although the performance of `Common` is slightly worse than `IRN2Vec` which optimized for individual application as shown in previous experiments, the common embeddings (`Common`) are generally useful for various applications, and thus meeting the goal of road network representation learning. On the other hand, it also shows that the information of intersection tags between intersections is usually useful for representation learning if available and more complete.

6 CONCLUSION

In this paper, we focus on representation learning of intersections on road networks. To the best of our knowledge, this is the first attempt to capture intrinsic properties in a road network to tackle the problem by exploring the geo-locality and homogeneity of road network intersections and the moving behaviors of mobile road users. To achieve the goal, we propose a novel two-phase framework, namely *IRN2Vec*, to capture various geo-spatial characteristics among intersections. The training data preparation algorithm in Phase 1 samples shortest paths to prepare training data for IRN2Vec. In Phase 2, the proposed IRN2Vec model captures various geo-spatial characteristics, including geo-spatial locality, same intersection tags and same N-way types, for multi-objective learning. Empirically, we validate our ideas and show that the proposed IRN2Vec framework is able to automatically learn effective embeddings of intersections to support a varity of ITS applications, e.g., prediction of traffic signals and crossings on intersections and travel time estimation of given moving paths. Via extensive experiments on multiple large-scale real-world datasets, we demonstrate the superiority, robustness and generality of IRN2Vec to other methods.

ACKNOWLEDGMENTS

This research is supported in part by the National Science Foundation under Grant No. IIS-1717084. Meng-xiang Wang and Ge Yu are supported by the State Scholarship Fund of the China Scholarship Council (201806080042) and the National Natural Science Foundation of China (U1811261, 61872070).

REFERENCES

[1] Naveed Akhtar and Ajmal S. Mian. 2018. Threat of Adversarial Attacks on Deep Learning in Computer Vision: A Survey. *IEEE Access* 6 (2018), 14410–14430.
[2] Mansur As and Tsunenori Mine. 2018. Dynamic Bus Travel Time Prediction Using an ANN-based Model. In *Proceedings of the 12th International Conference on Ubiquitous Information Management and Communication, IMCOM, Langkawi, Malaysia, January 05-07, 2018.*
[3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence* 35, 8 (2013), 1798–1828.
[4] Taxi Service Trajectory (TST) Prediction Challenge. 2015. Taxi Trajectory Prediction. http://www.geolink.pt/ecmlpkdd2015-challenge/.
[5] William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. 2016. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 4960–4964.
[6] Steve Coast. 2004. OpenStreetMap. https://www.openstreetmap.org/.
[7] Yoav Goldberg. 2017. *Neural Network Methods for Natural Language Processing*. Morgan & Claypool Publishers.
[8] Aditya Grover and Jure Leskovec. 2016. Node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.
[9] Carlos Herranz Perdiguero and Roberto J López Sastre. 2018. ISA: Intelligent Speed Adaptation from Appearance. *The Computing Research Repository* (2018).
[10] Tania Jogesh. Jason Lally, Blake Valenta. 2019. Open Data in San Francisco. https://data.sfgov.org/Transportation/Stop-Signs/4542-gpa3.
[11] ZHAO Jinbao DENG Wei WANG Jian. 2012. Analysis of Urban Intersection Traffic Accidents Based on Bayesian Network [J]. *Journal of Transport Information and Safety* 2 (2012).
[12] Sebastian Mattheis. 2016. Barefoot. https://github.com/bmwcarit/barefoot/.
[13] 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.
[14] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. (2013), 3111–3119.
[15] U.S. Department of Transportation. 2017. Intelligent Transportations Systems. https://www.its.dot.gov.
[16] Al-Rfou Rami Perozzi Bryan and Skiena Steven. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining*. ACM, 701–710.
[17] Jean Michel Loubes Francois Royer Philippe C. Besse, Brendan Guillouet. 2018. Destination Prediction by Trajectory Distribution-Based Model. *IEEE Transactions on Intelligent Transportation Systems* 19, 8 (2018), 2470–2481.
[18] Michal Piorkowski and Matthias Grossglauser. 2009. Dataset of mobility traces of taxi cabs in San Francisco. https://crawdad.org/epfl/mobility/20090224/.
[19] X Sun, J Guo, X Ding, and T Liu. 2016. A General Framework for Content-enhanced Network Representation Learning. *arXiv preprint arXiv:1610.02906* (2016).
[20] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 1067–1077.
[21] Lei Tang and Huan Liu. 2011. Leveraging social media networks for classification. *Data Mining and Knowledge Discovery* 23, 3 (2011), 447–478.
[22] Soujanya Poria Tom Young, Devamanyu Hazarika and Erik Cambria. 2018. Recent Trends in Deep Learning Based Natural Language Processing. *IEEE Computational Intelligence Magazine* 13, 3 (2018), 55–75.
[23] Quoc V. Le William Chan, Navdeep Jaitly and Oriol Vinyals. 2016. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, Shanghai, China*. 4960–4964.
[24] Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Chang. 2015. Network representation learning with rich text information. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.
[25] Yanbo Pang. Yoshihide Sekimoto Takehiro Kashiyama. 2017. An open dataset for Tokyo trajectory. https://github.com/sekilab/OpenPFLOW.
[26] Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. 2018. Recent trends in deep learning based natural language processing. *ieee Computational intelligenCe magazine* 13, 3 (2018), 55–75.
[27] Zhu Xingquan Zhang Daokun, Yin Jie and Zhang Chengqi. 2018. Network representation learning: A survey. *IEEE transactions on Big Data* (2018).