Region Representation Learning via Mobility Flow

Hongjian Wang
College of Information Sciences and Technology
Pennsylvania State University
hxw186@ist.psu.edu

Zhenhui Li
College of Information Sciences and Technology
Pennsylvania State University
jessieli@ist.psu.edu

ABSTRACT

Increasing amount of urban data are being accumulated and released to public; this enables us to study the urban dynamics and address urban issues such as crime, traffic, and quality of living. In this paper, we are interested in learning vector representations for regions using the large-scale taxi flow data. These representations could help us better measure the relationship strengths between regions, and the relationships can be used to better model the region properties. Different from existing studies, we propose to consider both temporal dynamics and multi-hop transitions in learning the region representations. We propose to jointly learn the representations from a flow graph and a spatial graph. Such a combined graph could simulate individual movements and also addresses the data sparsity issue. We demonstrate the effectiveness of our method using three different real datasets.

KEYWORDS

mobility flow; graph embedding; spatial-temporal data

ACM Reference Format:
Hongjian Wang and Zhenhui Li. 2017. Region Representation Learning via Mobility Flow. In *Proceedings of CIKM'17 , Singapore, Singapore, November 6–10, 2017,* 10 pages.
https://doi.org/10.1145/3132847.3133006

1 INTRODUCTION

As of 2016, more than 54% of the world's population live in urban areas, and the percentage is expected to increase to 66 by 2050 [12]. In the meantime, increasing amount of urban data are being accumulated in the digital form, such as human traces, traffic, venues, local events, etc. Many cities (e.g., New York City, Chicago, and Los Angeles) have joined the open data initiative and created websites to release the city data to the public [4, 6]. Analyzing such data could provide us with valuable insights into our urban dynamics, and make the city smarter.

In this paper, we are interested in a typical inference problem, which aims to infer a regional property (i.e. crime rate, personal income, and real estate price) from observed auxiliary urban features. Such inference can help us better understand the correlations among urban properties. Recent work [23] has shown that the taxi volume could be used as a similarity measure between regions. Through the similarity measure from traffic flows, we are able to employ the target variable of relevant regions to improve the predication accuracy. The intuition is that a large volume of flow from region A to region B indicates that the properties of A and B should be more relevant, and thus we could use one to predict the other. Although the intuition seems straightforward, there are some issues with it. Consider the example in Figure 1. On the left, there is no significant flow between two solid blue circles. However, since they share a lot of common neighbors, it is reasonable to assume they are relevant. On the right, the solid blue circle is a hub with strong connections to other regions. However, the hub could be a downtown area and play a different role compared with other regions.

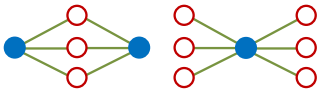


Figure 1: Each node is a region. The edge represents a significant amount of taxi flow between two regions.

The example above motivates us to account for the structural information of mobility flow graph. Graph embedding method [17, 20] can be one possible solution to model such structural information. A good region representation learned from mobility flow graph may help us better capture the relationships between regions and thus the correlations can be used to improve the inference model.

However, utilizing taxi flow data to learn the representations of regions is non-trivial. In literature, a transition matrix has been frequently used to represent mobility flow data [15, 18, 27, 28]. In this transition matrix, a region i can be represented by an n-dimensional vector, where the j-th element in the vector indicates the traffic volume from region i to j (out flow) or from region j to region i (in flow). However, such a representation does not consider the temporal dynamics. For example, the flow volume from region A to downtown might be the same as that from region B to downtown. Without considering the temporal dynamics, we cannot differentiate A and B w.r.t. downtown. However, the flow from A to downtown might mostly be morning transitions whereas the flow from B to downtown happens in the evenings. Downtown region might function as a working place for people living in Region A but function as an after-work entertainment region for people working in region B. So A and B should be different in their presentations since they have different relationships with downtown region.

To consider the temporal dynamics, we could construct a tensor by adding a time dimension in addition to the transition matrix [26]. However, such a tensor does not capture the multi-hop transitions between regions. For example, there could be strong flow from residential area A to working area B in the morning, and then from working area B to restaurant area C in the evening. This

CIKM'17 , November 6–10, 2017, Singapore, Singapore

ACM ISBN 978-1-4503-4918-5/17/11…$15.00
https://doi.org/10.1145/3132847.3133006

indirect transition relationship between the region A and C cannot be captured in the pairwise transition matrix.

We propose to learn representations of regions by adapting recent embedding techniques, which have demonstrated successful results in word embedding [9–11, 16] and graph embedding [8, 17, 20]. However, the mobility flow data input are quite different from those data. The key challenge lies in how to generate a meaningful context for a region using the mobility flow data, in a similar way to the sentence context for word embedding or the neighbor context for graph embedding. Another challenge lies in the data sparsity. Even though we have a huge mobility dataset, the data follow a long-tail distribution w.r.t. regions. We could still have no information for some remote regions at certain times such as midnight, and thus it is difficult to learn their corresponding representations.

We propose a region embedding method by considering both temporal dynamics and multi-hop transitions. We define a flow graph, where each vertex represents one region within a certain time interval and edges represent the transition flow between two regions at different time intervals. The structure encodes both temporal dynamics and multi-hop transitions. To further address the sparsity issue, we define a spatial graph, and learn the embedding jointly on the flow graph and the spatial graph. The spatial graph captures the geographical adjacency among regions and complements the flow graph.

In experiment, we evaluate our embedding methods on three prediction tasks. The proposed embedding method is shown to consistently outperform existing methods. In order to interpret the semantic meaning of learned embeddings, we conduct a quantitative analysis on taxi and POI data, and also give a case study on the embedding visualizations.

To summarize, the key contributions of this paper are:

- We study a generalized inference problem in the urban setting. We propose to learn region embedding from mobility flow data to enhance the inference model.
- In order to incorporate both temporal dynamics and multi-hop transition and also to address the sparsity issue, we propose to jointly learn the embedding from the flow graph and the spatial graph.
- We validate our method through extensive experiments on multiple datasets.

The rest of this paper is organized as follows. Section 2 gives the motivation of learning region embeddings. Section 3 presents the formal definition of our problem. Section 4 introduces the dynamic embedding method in detail. The experiment results are given in Section 5. Section 6 summarizes related work. Finally, we conclude in Section 7.

2 PRELIMINARY

In this section we first present the generalized inference model, and then introduce our empirical observations on the relationship strengths measure.

2.1 Generalized Inference Model

The generalized inference model is a typical regression task to study the urban dynamics from various data sources. Given a set of K non-overlapping regions $\mathcal{R} = \{r_1, r_2, \cdots, r_K\}$, we are interested in estimating the target variable for every region, denoted as y_i for region r_i. We only have observations of target variables on a subset of regions. However, we observe some auxiliary features for all of the regions, such as the demographics and average income. These auxiliary features are denoted as $X_i \in \mathcal{R}^d$ for region r_i, where d is the dimension of auxiliary features.

To predict the target variable, we use the following generalized regression model

$$y_i = \alpha \cdot X_i + \beta \sum_{j \in \mathcal{N}_i} sim(i,j) \cdot y_j + \gamma, \tag{1}$$

where α, β, and γ are parameters of the regression model. The term $\sum_{j \in \mathcal{N}_i} sim(i,j) \cdot y_j$ accounts for the propagation effect of neighboring regions of r_i, where $\mathcal{N}_i$ is a set of neighboring regions of r_i and $sim(i,j)$ measures the relevance of region pair $\langle r_i, r_j \rangle$. The relevance function $sim(i,j)$ is usually defined with extra information, such as the spatial information.

It is the relevance function $sim(i,j)$ that allows us to generalize the base model. For example, a straightforward definition with hard boundary is $sim(i,j) = \begin{cases} 1 & \text{if } r_i \in \mathcal{N}_j, \\ 0 & \text{otherwise,} \end{cases}$ where $\mathcal{N}_j$ is the set of k-nearest neighbors to region r_j. In order to provide a more flexible way to control the relevance of neighbors, a soft version of the relevance function could be defined with a spatial distance measure as $sim(i,j) = \frac{1}{d(i,j)}$, where $d(i,j)$ is the distance between the centroids of two regions. The intuition is that the closer two regions are, the more relevant they are.

As newer type of data is available, such as the taxi commuting data, the relevance measure can be defined accordingly as $sim(i,j) = \frac{f(j,i)}{\sum_{p \in \mathcal{N}_i} f(p,i)}$, where $f(j,i)$ is the amount of flow from r_j to r_i. Recent work from Wang et al. [23] employs this definition and brings taxi flow feature into the model. The prediction model becomes

$$y_i = \alpha \cdot X_i + \beta_1 \vec{w_g}^T \vec{y} + \beta_2 \vec{w_f}^T \vec{y} + \gamma, \tag{2}$$

where $\vec{w_g}$ is the reverse distance weighting vector, $\vec{w_f}$ is a weighted taxi flow vector, and $\vec{y}$ is the target variables of all other regions. Moreover, Wang el al. [23] verifies that negative binomial regression is preferable to linear regression for predicting non-negative y_i. In the rest of this paper, we use the negative binomial regression as our generalized model, i.e.

$$y_i = \exp(\alpha \cdot X_i + \beta \sum_{j \in \mathcal{N}_i} sim(i,j) \cdot y_j + \gamma). \tag{3}$$

2.2 Empirical Study with Urban Data

To verify the taxi flow can serve as a good relevance measure, we first make some observations with crime data and taxi data in Chicago. For every pair of regions $\langle r_i, r_j \rangle$, we plot their crime rate differences against the flow volume $f(i,j)$ in Figure 2.

Overall, the blue points in Figure 2 validate the intuition of adding taxi flow into prediction model in Equation (2). However, we notice that there are many red region pairs do not follow this intuition. The reason is that the downtown region is contained in those pairs. Chicago downtown region has the highest crime rate, and there is a significant amount of traffic between downtown and other regions.

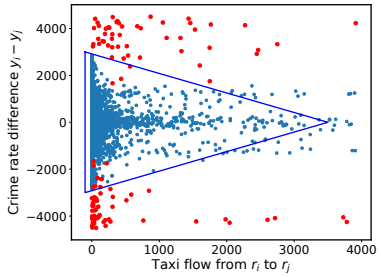


Figure 2: The crime rate difference vs. traffic flow volumes for every pair of regions $\langle r_i, r_j \rangle$. Points forming the blue triangle shape indicate that the larger the flow between region r_i and region r_j is, the difference between their crime rates is smaller. The red point denotes a pair of regions with one region being the downtown area.

The observation above motivates us to look beyond the traffic volume to determine the relevance measure. As shown in the Figure 1, if we account for the structural information of mobility flow, the downtown is a popular hub, which differentiates itself from most of other regions. The graph embedding method is therefore a sound solution to estimate region relevance by modeling such structural information.

3 PROBLEM DEFINITION

We define the region representation learning problem as a joint embedding learning problem on two different graphs — flow graph and spatial graph.

The input data consist of mobility data and spatial information of the city. The mobility dataset containing n trips is denoted as $\Gamma = \{\gamma^i\}$. Each trip γ has the format $\langle l_s, l_e, t_s, t_e \rangle$, where l_s and l_e are the starting and ending location coordinates (i.e., latitude and longitude), and t_s and t_e are the starting and ending time of the trip respectively. The spatial information of the city is a set of K non-overlapping regions, denoted as $\mathcal{R} = \{r_1, r_2, \cdots, r_K\}$. The regions could be defined as the administrative boundaries (e.g. tracts and community areas) or partitioned by the road network [26]. The spatial boundary of each region r_i is given as well. To simplify the temporal dynamics, we use relative timestamps within one day $\mathcal{T} = \{1, 2, \cdots T\}$ with fixed time intervals, such as 1 hour.

The same region at different timestamps bears different functions, and thus the embedding could be different. We use *time-enhanced vertices* to differentiate the same region at different timestamps in our heterogeneous dynamic graph.

Definition 3.1 (Time-enhanced vertex). Each vertex in our graph is denoted as v_i^t, which represents the region r_i at time t. We call this *time-enhanced vertex* and our method will learn embedding representation for each such vertex. The set of time-enhanced vertices is denoted as $\mathcal{V}$, which contains $K \cdot T$ vertices, where K is the number of regions, and T is the number of timestamps.

Given a set of vertices, there are two kinds of relations we want to capture. The first type of relation is derived from the mobility flow among those regions, which is formulated into the *flow graph* G_f. The second one is the spatial adjacency, which is defined by the *spatial graph* G_s. The intuitions and definitions of these two graphs will be introduced in detail in Section 4.

Our method learns the representations from both graphs simultaneously. The formal definition of our problem is as follows.

Definition 3.2 (Dynamic mobility graph embedding). Given the flow graph G_f and spatial graph G_s, we aim to learn a vector representation $\mathbf{u}_i^t \in \mathbb{R}^d$ in a low dimensional space for each vertex $v_i^t \in \mathcal{V}$, i.e. learning a mapping $f : \mathcal{V} \to \mathbb{R}^d$. In the d-dimensional embedding space, both the mobility flow relation and spatial adjacency are preserved.

With the region embeddings, we define the relevance measure by their dot product, i.e. $sim(i, j) = \mathbf{u}_i^T \mathbf{u}_j$.

4 METHOD

In this section, we give the design motivations and formal definitions of the flow graph and the spatial graph. Following the graph definitions, we describe the embedding learning objective. At last, we present the optimization techniques to learn the embedding.

4.1 Flow Graph

The same region at different time may carry different functions. Take the downtown area as an example, which has mixed point-of-interests distribution. In the morning, people go to downtown mostly for work. Therefore, in the morning the downtown area acts as a professional area. However, at night there are also a significant amount of people traveling to downtown for food and drink, and the downtown acts as an entertainment area.

The aforementioned example motivates us to learn different embeddings for the same region at different times. Follow this intuition, we design the flow graph G_f as a layered graph, shown on the left of Figure 3, and formally defined as follows.

Definition 4.1 (Flow graph). The *flow graph* is a layered graph defined as $G_f = (\mathcal{V}, E_f)$. The vertices $\mathcal{V} = \{v_i^t\}$ is the set of time-enhanced vertices. Vertices with the same timestamp are grouped into one *layer*, and there are T layers in total. The edge set E_f only contains one type of edges $\{e_{ij}^t\}$, where e_{ij}^t connects vertices v_i^t and v_j^{t+1} from two consecutive layers. The edge weight f_{ij}^t is the volume of mobility flow.

The flow graph models the mobility pattern of crowd in the city. More specifically, we can sample a lot of paths from the flow graph to represent human trajectories. Each path consists of a sequence of regions, whose timestamps are monotonically increasing with a fixed step of 1. The length of each path is bounded by the number of timestamps in the graph. And each path semantically refers to one trajectory of a individual. For example, one possible path is $\langle$home, 8:00 am$\rangle \to \langle$office, 9:00 am$\rangle \to \cdots \to \langle$office, 6:00 pm$\rangle \to \langle$bar, 7:00 pm$\rangle$.

However, there are three issues with a path sampled from the flow graph. First, the flow graph does not deal with the fact that people travel to a region and stay there. For example, in the left-most graph in Figure 3, the edge between v_1^1 (node of region r_1 at time $t = 1$) and v_1^2 (node of region r_1 at time $t = 2$) is missing, which means there is no trip observed transiting within the same region, but there could be people staying in that region. Second, the flow

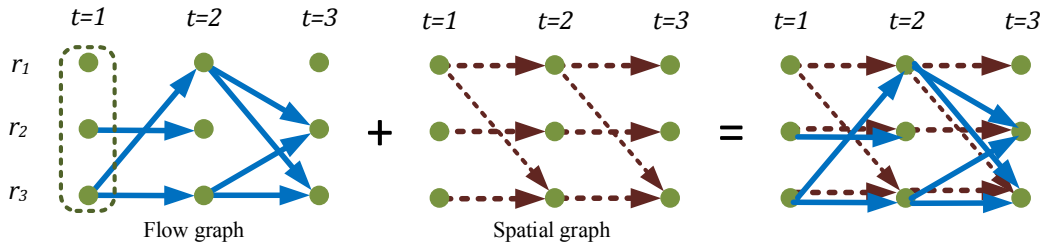


Figure 3: The layered structure of a flow graph (left), a spatial graph (middle), and the combined graph (right). Each row r_k represents one region. Each column t is the timestamp, and all vertices within at the same timestamp (the dotted rectangle) form one *layer* of the graph. Each vertex v_i^t is a *time-enhanced vertex* refers to region r_i at time t. On the left, the solid blue edge refers to the taxi flow, and edge weight is number of taxi trips. In the middle, the dotted red edge refers to the spatial adjacency, and the edge weight is inversely correlated with the distance between region centroids. From the flow graph, vertices v_1^2 and v_3^2 have similar embeddings because they have similar in-flow from v_3^1 and similar out-flow to v_2^3 and v_3^3. However, with flow graph alone, we are not able to learn the embeddings for v_1^1 and v_1^3, due to lack of traffic flow. The spatial graph provides spatial information, which makes it possible to learn an embeddings for v_1^1 and v_1^3.

graph suffers data sparsity issue. If there is no traffic flow going in/out certain region during a time interval, then it is impossible to learn the embedding of this region at that time. Third, the flow graph cannot recognize the same or nearby region across different time slots. More specifically, the flow graph treats all $K \cdot T$ vertices as independent regions. However, it is very likely that the same region in different time slots are strongly correlated. Recall the residential area example, where the large volume of in-coming flow at night is caused by the large volume of out-going flow in the morning.

4.2 Spatial graph

To address the issues with the flow graph, we propose a spatial graph, which is defined as:

Definition 4.2 (Spatial graph). The *spatial graph* is a layered graph as well, denoted as $G_s = (\mathcal{V}, E_s)$. The vertices set $\mathcal{V}$ is exactly the same as that of flow graph. The edge set E_s also only contains edges connecting two vertices from consecutive layers. The edge weight g_{ij}^t represent the spatial similarity of two regions, which is inversely correlated with distance.

The spatial graph shares the same structure and exactly the same vertices with the flow graph. The only difference is that the edges in spatial graph are constructed differently. The basic assumption behind the spatial graph is that human mobility are bounded by space. When there is no transition observed, the probability that people appeared at a different region is inversely correlated to the distance they need to travel. Therefore, two regions that are close in space should have stronger correlation in their embeddings. In spatial graph, the edges e_{ij} refers to the spatial similarity between regions r_i and r_j. The edge weight g_{ij} is inversely correlated with the distance, formally defined with exponential decay function [13] as follows

$$g_{ij} = \exp(-C \cdot d_{ij}), \tag{4}$$

where d_{ij} is the spatial distance between the centroids of two regions. We should notice that the spatial graph is static over time, therefore, all edges between any two consecutive layers are actually the same. C is a parameter controls the exponential decay rate of the distance. Larger C means faster decay, which makes regions far away have little correlation with current region.

The design of spatial graph, shown in the middle of Figure 3, naturally incorporates the spatial adjacency. This spatial adjacency could be regarded as a transition cost, which helps us to estimate the stay probability. Even more, the spatial adjacency enables the embedding learning for regions without any taxi flow, which solves the sparsity issue. Lastly, the spatial adjacency identifies the same region across different timestamps, because the edge between a pair of time-enhanced vertices representing the same region always has the maximum weight.

4.3 Heterogeneous Graph Property

Combining the flow graph and spatial graph together, we get a heterogeneous graph that represents the crowd mobility pattern on the right of Figure 3. In this heterogeneous graph, one path conveys more information about crowd mobility pattern than that from the flow graph. Now it is possible for a path to capture both transition and stay, such as $\langle$home, 8:00 am$\rangle \rightarrow \langle$office, 9:00 am$\rangle \xrightarrow{\textbf{stay}} \langle\textbf{office, 10:00am}\rangle \rightarrow \cdots \rightarrow \langle$office, 6:00 pm$\rangle \rightarrow \langle$bar, 7:00 pm$\rangle$.

This heterogeneous graph has two properties that meet the requirements of our problem. (1) The graph is still a temporal graph, which enables us to learn dynamic embeddings for each region. (2) In the heterogeneous graph, the multi-hop temporal dependency is captured within each path. The multi-hop temporal dependency is important to differentiate region functions. For example, at 6:00 pm we observe same amount of flow going into region A and B, which makes it difficult to differentiate the function of A and B. But if we know that in the morning, there is a large amount of flow going out of A, while almost no flow going out of B, then A is more likely to be a residential area, whereas B is more likely to be an after-work entertainment region.

4.4 Embedding Learning Objective

In order to capture two properties mentioned above, we propose to use the embedding technique to learn the representation of each region. The reason is that graph embedding explicitly captures the multi-hop dependency. Meanwhile, the baseline method for graph

representation learning, such as directly using the in/out flow as vector representation or matrix factorization, is not able to capture the multi-hop correlation.

4.4.1 On Single Graph. The embedding learning process on the flow graph and the spatial graph are exactly the same, due to the fact that both graphs have similar structure. Without loss of generality, we explain the learning process on the flow graph.

First we define a path as $P_i = v_{i_1} v_{i_2} \cdots v_{i_m}$, whose starting and ending vertices are v_{i_1} and v_{i_m} respectively. We omit the time superscript, because the time slots for the vertices of path P must be monotonically increasing with fixed step size 1. And we denote the relation that a path contains a vertex v_i^t as $v_i^t \in P$. With the definition of path, we further define the set of paths containing v_i^t as $\mathcal{P}(v_i^t) = \{P_i | v_i^t \in P_i\}$. The context of one vertex v_i^t, which refers to all the other vertices that are multi-hop neighbors of v_i^t, is defined as $C(v_i^t) = \{v_c | \exists P_i \in \mathcal{P}(v_i^t), v_c \in P_i\} \setminus \{v_i^t\}$.

We adopt the skip-gram model [9] to learn the embedding $\mathbf{u}_i^t$ for each node v_i^t. Formally, we estimate

$$p_f(v_c|v_i^t) = \frac{\exp({\mathbf{u}_i^t}^T \mathbf{u}_c)}{\sum_{v_{i^*} \in C(v_i^t)} \exp({\mathbf{u}_i^t}^T \mathbf{u}_{i^*})}, \tag{5}$$

where v_c is one vertex in v_i^t's context $C(v_i^t)$, $\mathbf{u}_c$ and $\mathbf{u}_i^t$ are the embeddings of v_c and v_i^t respectively.

The empirical conditional probability $\hat{p_f}(v_c|v_i^t)$ is estimated by the volume of mobility flow in the flow graph. More specifically, if v_c is within the context of v_i^t, there must be at least one path from v_c to v_i^t or a path from v_i^t to v_c. Without loss of generality, we assume one of the path is from v_i^t to v_c with m vertices, denoted as P_i, where $v_{i_1} = v_i^t$ and $v_{i_m} = v_c$. First we estimate the transition probability of two adjacent vertices. Then the empirical probability $\hat{p_f}(v_c|v_i^t)$ is estimated from this transition probability.

The transition probability between two directly connected vertices $v_{i_k}^t$ and $v_{i_{k+1}}^{t+1}$ is given by

$$p(v_{i_{k+1}}^{t+1}|v_{i_k}^t) = \frac{f_{i_k i_{k+1}}^t}{\sum_{v_{j^*} \in N(v_i^t)} f_{i_k j^*}^t}, \tag{6}$$

where $N(v_i^t)$ refers to the direct next-hop neighbors of vertex v_i^t, and $f_{i_k i_{k+1}}^t$ refers to the weight of edge $e_{i_k i_{k+1}}^t$ in the flow graph. Therefore, the transition probability from v_{i_1} to v_{i_m} through P_i is

$$\begin{aligned} p(P|v_{i_1}) &= p(v_{i_m}, v_{i_{m-1}}, \cdots, v_{i_2}|v_{i_1}) \\ &= \prod_k^m p(v_{i_k}|v_{i_{k-1}}, v_{i_{k-2}}, \cdots, v_{i_1}) \end{aligned} \tag{7}$$

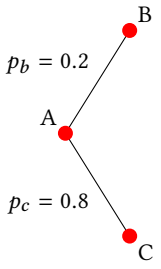
Due to the Markov property, Equation (7) becomes

$$p(P|v_{i_1}) = \prod_k^m p(v_{i_k}|v_{i_{k-1}}), \tag{8}$$

The empirical conditional probability $\hat{p_f}(v_c|v_i^t)$ is

$$\hat{p_f}(v_c|v_i^t) = \sum_{P_i \in \mathbb{P}} p(P_i|v_i^t), \tag{9}$$

where $\mathbb{P}$ is the set of all paths starting at v_i^t and ending at v_c. Finally, we can learn the embedding by minimizing the difference between



Vertex	Index	Probability p_i	Alias K_i
B	0	0.4	C
C	1	1.0	-

Constant time sampling process:
1. Draw uniform random number $x \in [0, 1)$.
2. Identify the index of row $i = \lfloor nx \rfloor$.
3. If $x > p_i$, return K_i. Otherwise, return i.
Example 1: $x = 0.45$, $i = 0$. Since $x > p_0$, return C.
Example 2: $x = 0.35$, $i = 0$. Since $x < p_0$, return B.

Figure 4: The alias method explanation. On the left, we want to draw the next vertices of A. The probability table and alias table are created on the top right. The bottom right shows the constant time sampling process from the alias method.

two distributions $p_f(v_c|v_i^t)$ and $\hat{p_f}(v_c|v_i^t)$. The objective is

$$O_f = D(p_f(\cdot|\cdot), \hat{p_f}(\cdot|\cdot)), \tag{10}$$

where D is the distance function for two distributions, and one commonly used function could be the KL divergence.

The embedding learning objective of spatial graph is similar to the flow graph. We minimize the difference between the embedding distribution and empirical distribution, which is

$$O_s = D(p_s(\cdot|\cdot), \hat{p_s}(\cdot|\cdot)). \tag{11}$$

4.4.2 On Heterogeneous Graph. In order to learn our embedding on two graphs simultaneously, we combine Equation (10) and Equation (11), and the joint learning objective is

$$O = O_f + O_s = D(p_f(\cdot|\cdot), \hat{p_f}(\cdot|\cdot)) + D(p_s(\cdot|\cdot), \hat{p_s}(\cdot|\cdot)). \tag{12}$$

4.5 Embedding Learning Optimization

4.5.1 On Single Graph. Directly optimizing the objective in Equation (10) and Equation (11) is computationally expensive, due to two reasons.

(1) To calculate the conditional probability $p_f(\cdot|v_i^t)$ in Equation (5), for each v_i^t it requires the summation over the entire set of vertices. Therefore, the overall complexity is $O(K^2 \cdot T^2)$, where $K \cdot T$ is number of vertices.
(2) To estimate the empirical conditional probability $\hat{p_f}(v_c|v_i^t)$ in Equation (9), for every pair of vertices we have to sum over all paths P_i among them, which is exponential to the number of vertices in one layer.

To address the first problem, we adopt the negative sampling approach proposed in [10], which samples multiple negative pairs from a noise distribution to estimate one true pair. The objective is given by

$$\log \sigma({\mathbf{u}_i^t}^T \mathbf{u}_c) + \sum_q^s \mathbb{E}_{v_q \sim P_n(v_i^t)} \left[\log \sigma(-\mathbf{u}_q^T \mathbf{u}_c) \right], \tag{13}$$

where $\sigma(x) = \frac{1}{1+\exp(-x)}$ is the sigmoid function, $P_n(v_i^t)$ is the noise distribution, and s is the number of negative samples. The Equation (13) is used to replace every $\log p(v_c|v_i^t)$ term during the skip-gram optimization.

To address the second problem, we use the graph sampling method to estimate the empirical probability $\hat{p}(v_c|v_i^t)$. More specifically, we generate m paths from the graph via random walk. Due

to the special structure of our layered graph, the time index of the sequence must be monotonically increasing with fixed step size 1. Given that m is large enough, we could use the co-occur frequency count from those random walks to estimate $\hat{p}(v_c|v_i^t)$ with sufficient accuracy.

The random walk boils down to next-vertex sampling according to the edge weights. Since the random walk is conducted on a weighted graph, at each vertex, we sample the next vertex according to the out-degree distribution, which could be expensive. The straightforward method is to convert each weighted edge into an interval within the range of $[0, w_{sum})$, where the w_{sum} is the sum of out degree at current vertex. The sampling process is that first generate a uniform random number $x \in [0, w_{sum})$, and then find the interval that x maps into. Therefore, this next-vertex sampling method takes $O(K)$ time, where K is the number of vertices in one layer, which is also the upper bound of number of outgoing edges from current vertex. Since we are generating m paths, we have to conduct $m \cdot T$ next-vertex sample process, where T is the upper-bound of the path length. The overall complexity would be $O(m \cdot T \cdot K)$.

We further boost the next-vertex sampling process with alias method [22]. The advantage of alias method is that it is possible to repeatedly sample next edge with constant time, after preprocessing outgoing edges and save the information. More specifically, the alias method creates two tables for the next edges as shown in Figure (4). The alias method makes the path sampling significantly faster, because in our path sampling process we repeatedly sample on each vertex. For each vertex, the initialization of alias tables take $O(K)$, where K is the upper bound for the number of next vertex. Therefore, the overall initialization takes $O(K \cdot T \cdot K) = O(K^2 \cdot T)$, where $K \cdot T$ is the number of alias tables need to create. The overall sampling process takes $O(m \cdot T)$. Since $m \gg K$, it is safe to assume that $m > K^2$, and then the overall complexity is $O(K^2 \cdot T + m \cdot T) = O(m \cdot T)$. The experimental comparison of alias method with the simple method is described in Section 5.2.3.

4.5.2 On Heterogeneous Graph. The sampling-based method above can be easily applied on the heterogeneous graphs to learn the joint embedding. We conduct random walk on both graphs to generate path simultaneously. Then we feed all the paths to the skip-gram neural networks model to learn a joint embedding for each vertex.

4.6 Discussion: Path Sampling

Here we draw a connection between our graph sampling-based optimization technique and word2vec [9] in language modeling method. The goal of word2vec is to build vector representations of words using probabilistic neural networks. This idea could be re-purposed to model the graph structure as well [17], due to the power law property in both the degree distribution in a graph and the word frequency distribution in natural language.

We regard the set of vertices in the graph as a special corpus, and each vertex is a word. The path sampled from the weighted graph via random walk can be thought of sentences. The multi-hop neighbors of a vertex in the path is similar to the word context. Therefore, estimating the neighboring vertices of a given vertex is analogy to the skip-gram language model [8].

5 EXPERIMENT

In this section, we first describe datasets and experiment settings. Then we evaluate the effectiveness and efficiency of proposed embedding method with several prediction tasks. Finally, we interpret the semantic meaning of the learned embedding with both quantitative analysis and case study.

5.1 Settings

5.1.1 Data description. We study the urban dynamics at community area (CA) level. A community area is a predefined administrative area in the city of Chicago. The geographic boundary information is available through US census survey [3]. The following urban data are collected and used in our evaluation.

Demographics data at community area level is made public by the US census bureau [3]. The demographic features mainly cover the following aspects of a community area: total population, population density, poverty, residential stability, and ethnic diversity.

Point-Of-Interest (POI) data is obtained through Foursquare API [19]. It contains more than 112,000 POI records for Chicago. Each POI record provides venue name, category, number of check-ins, and number of unique visitors. We use the POI category distribution information of each region to measure the region functions. There are 10 major POI categories including arts & entertainment, education, event, food, nightlife, outdoor & recreation, professional, residence, shops and travel.

Taxi data [6] in Chicago from 2013 to 2015 are used to construct the mobility flow graph. There are over 86 million taxi trips recorded over the three years, which is roughly 2.4 million trips per month. For each trip, we have the following information available: pick-up and drop-off dates and locations. Due to privacy concern, in this dataset, all timestamps are rounded to closest 15 minute marks, and all locations are mapped to the center of census tracts.

Crime data is publicly available on Chicago Data Portal [14], which contains more than 5 million crime incidents from 2001 to current day. The incident date, location, and primary type of each crime incident are recorded.

House price data is obtained from Zillow real estate website [29]. We collect the sale price, floor size, latitude, and longitude information for over 45,000 real estates that were sold within 2 years in the city of Chicago.

In order to evaluate and interpret our embedding results, we predict the following three target variables for each community area.

- Crime rate, which is crime incidents count per 10,000 population.
- Average personal income in dollar.
- Average house price with a unit of dollar per square foot.

5.1.2 Methods for comparison. For each prediction task, we follow the generalized regression framework in Equation (3). We use the state-of-the-art method in [23] as a base model, which does not employ the embedding technique to calculate relevances. Since the base model directly employs the traffic volume and inverse spatial distance as relevance measure, we denote it *RAW* in the rest of experiments.

We name our embedding method as **heterogeneous dynamic graph embedding (***HDGE***)**. This proposed dynamic embedding

technique also applies to single flow graph or spatial graph, which are called DGE_{flow} and $DGE_{spatial}$ respectively. We set the embedding dimension as 8 for all methods. We compare *HDGE* with two alternative embedding methods. First, we introduce a straightforward baseline approach for flow graph modeling, called *slotted graph*. Similar to flow graph, the slotted graph also accounts for the temporal dynamics. However, the slotted graph models the mobility flow for each time slot independently.

- **Matrix factorization (*MF*)** is a conventional method for dimension reduction. In order to get dynamic vector representations, the matrix factorization method is used to decompose the adjacency matrices of slotted graphs.
- **LINE** [20] is a graph embedding method that learns embedding on a weighted graph to encode both first and second order proximity. Applying LINE on the slotted flow graph also leads to an alternative temporal embedding.

5.1.3 Evaluation metrics. The dynamic embedding method learns different embeddings for different time slot. Within each time slot, we use leaned embeddings to calculate the relevance measures and evaluate the regression model with leave one out setting. The model performance is evaluated by mean relative error and mean absolute error:

$$MRE = \frac{1}{T}\sum_{t=1}^{T}\frac{\sum_i^n |y_{it} - \hat{y_{it}}|}{\sum_i^n y_{it}} \quad MAE = \frac{1}{T}\sum_{t=1}^{T}\sum_{i}^{n}|y_{it} - \hat{y_{it}}|,$$

where y_{it} is the ground truth value for target variable of region i at time slot t, and $\hat{y_{it}}$ is the estimate.

It is worthy mentioning that among all three target variables only crime rate presents daily periodicity. For average personal income and real estate price, the value of the same region does not change within one day, i.e. $\forall t \in \mathcal{T}, y_{it} = y_i$.

5.2 Evaluations

5.2.1 Feature Selection. For each predication task, we have four types of features available, which are demographic features (D), POI features (P), geographical feature (G), and taxi flow feature (T). In this section, we aim to identify the best feature combinations for each prediction task. We use the base model *RAW* for this purpose.

Table 1: Crime rate prediction with *RAW* from 2013 to 2015. The MAE unit is crime count per 10,000 population.

Year	2013		2014		2015	
Features[1]	MAE	MRE	MAE	MRE	MAE	MRE
D+P	15.03	0.318	13.26	0.317	7.31	0.335
D+P+G	15.54	0.329	13.75	0.326	7.46	0.337
D+P+T	14.52	0.308	12.79	0.307	7.15	0.322
D+P+G+T	14.92	0.316	13.15	0.316	7.35	0.332

[1] D – demographic features, G – geographical influence, P – POI features, T – taxi flow feature.

The crime rate prediction results of *RAW* method with different feature combinations are shown in Table 1. We only show the prediction results from year 2013 to 2015, because only in those years we have both taxi flow data and crime incident data. From Table 1, we observe that the best crime rate prediction is achieved by using only three types of features, i.e. demographics, POI, and taxi flow. Adding geographic features does not improve the prediction accuracy. This observation is actually consistent with previous work [23].

Table 2: Average personal income and house price prediction with *RAW*. The MAE unit of personal income is dollar. The MAE unit of house price is dollar per square foot.

Data	Income		House Price	
Features	MAE	MRE	MAE	MRE
D+P	15304	0.253	39.87	0.233
D+P+G	16905	0.279	41.40	0.242
D+P+T	15433	0.255	39.28	0.229
D+P+G+T	15127	0.250	40.728	0.238

The average personal income and house price prediction results of *RAW* are shown in Figure 2. When making income prediction, we eliminate related features from the demographics features. To make fair evaluation, we try our best to align the time window of features and target variables. More specifically, the income census data is collected in 2010, and we use taxi flow in the closest year as features. The house price data is from 2015 to 2017, and the taxi flow in 2015 is used to predict house price.

From Table 2, we observe that the best feature combination for income prediction is to involve all four types of features. Meanwhile, the best feature combination for house price prediction is demographics, POI, and taxi flow.

In all three prediction tasks, the taxi flow features are consistently proven to effectively improve the prediction accuracy.

5.2.2 Embedding Evaluation. In this section, we evaluate the embedding results by calculating the relevance measures with learned embeddings. Without loss of generality, we define the relevance measure in Equation (3) by their dot product, i.e. $sim(i, j) = \mathbf{u}_i^T \mathbf{u}_j$.

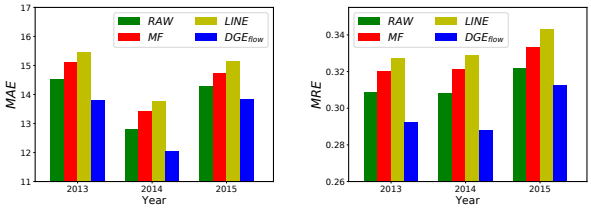


Figure 5: Crime rate prediction MAE (left) and MRE (right) with dynamic mobility flow embeddings.

The *MAE* and *MRE* of crime rate prediction in different years are shown in Figure 5. All methods use D+P+T feature combinations, and the MRE of *RAW* (green bar) is from the highlighted row in Table 1.

We could see that DGE_{flow} consistently has the best performance. There are two reasons that DGE_{flow} is able to outperform *RAW*. First, DGE_{flow} employs the multi-hop structural information, which potentially enables the crime to be propagated for more than one hop. Second, DGE_{flow} captures the temporal transition information as well. *LINE* and *MF* have worse performance than DGE_{flow}, mainly because embeddings are learned on the independent slotted graph, which does not account for the temporal transition information.

Table 3: Average personal income and house price prediction with embedding methods.

Data	Income		House Price	
Features	D+P+G+T		D+P+T	
Method	MAE	MRE	MAE	MRE
RAW	15127	0.250	39.28	0.229
MF	16674	0.2756	39.83	0.233
LINE	15534	0.2567	40.438	0.236
DGE_{flow}	-	-	38.95	0.226
HDGE	14740	0.2436	-	-

We show the embedding methods comparison of average income and house price prediction in Table 3. The income prediction uses the feature combination D+P+G+T, while the house price prediction uses the feature combination D+P+T. Similarly, we observe that the proposed *HDGE* and DGE_{flow} are able to learn a better relevance scores respectively, and thus improve the *RAW* method. The other embedding methods *MF* and *LINE*, however, lead to a worse performance. This verifies that the proposed flow graph design is necessary to account for the relevance among regions.

5.2.3 Running Time. We validate the performance gain of applying alias method for random walk sampling on weighted graphs.

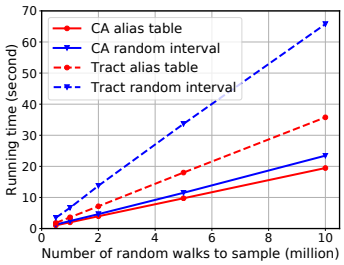


Figure 6: The running time of random walk sampling on weighted graphs.

In order to validate the efficiency of alias method, we conduct random walk sampling on two flow graphs. The first flow graph is generated at community area level, while the second flow graph is generated at tract level. The tract is a smaller administrative boundary used for the census survey. There are 801 tracts in Chicago, compared to 77 communities areas. The length of random walks for both graph are bounded by 24. The number of sampled random walks ranges from 500k to 10 million.

The running time is shown in Figure 6. The compared method is called random interval, which is described in Section 4.5.1. It is clear that the alias method consistently runs faster than the random interval method. The alias table method has better performance gain when the number of sampled random walks is large, comparing the solid blue line and solid red line. The reason is that the alias method has a fixed overhead to calculate the alias table for each vertex. Also, the performance gain of alias method on a large graph is bigger. The reason is that alias method reduce the next-vertex sampling complexity from $O(K)$ of the random interval method to $O(1)$, and a larger graph usually has larger K, and thus a larger performance gain.

5.3 Interpretations

In this section we give semantic interpretation of the learned dynamic graph embedding. First, we show that *HDGE* to some degree account for the POI similarity among regions. Next, we use a case study to intuitively explain the semantics captured by *HDGE*.

5.3.1 HDGE and POI. The POI data reflect different functions of urban areas [26], which is a candidate measure of similarity among regions. Our hypothesis is that to certain degree the *HDGE* accounts for the POI similarity among regions, even though the *HDGE* learning process does not involve any POI data at all.

Due to lack of ground truth, we conduct an unsupervised information retrieval experiment to compare different embedding methods. Each region is used as a query, and the goal is to rank other regions according to their similarities to the query region. The POI similarity ranking is used as the ground truth. The quality of various embedding methods are evaluated with the *nDCG* measures of corresponding rankings.

We use normalized discounted cumulative gain (nDCG) as evaluation measure. Formally, the discounted cumulative gain (DCG) is defined as $DCG@k = \sum_{i=1}^{k} \frac{rel_i}{\log_2(i+1)}$, where the relevance rel_i is derived from POI similarity. The nDCG is the DCG normalized by the idea DCG (iDCG), i.e. $nDCG@k = \frac{DCG@k}{iDCG@k}$, where *iDCG* is the DCG of the best ranking. Higher $nDCG@k$ value means better quality of the mobility flow embedding similarity.

We conduct this experiment at tract level, and there are 801 tracts in Chicago. We set the embedding dimension as 20 for all methods, and divide one day into 8 3-hour time slots. To make fair comparison, we sample a subset of tracts that all methods are able to learn embeddings, which results in a set of 419 tracts.

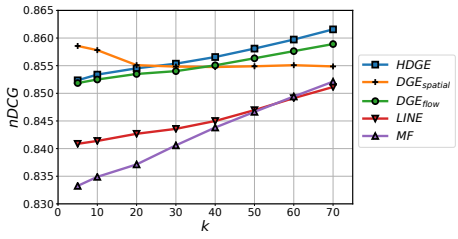
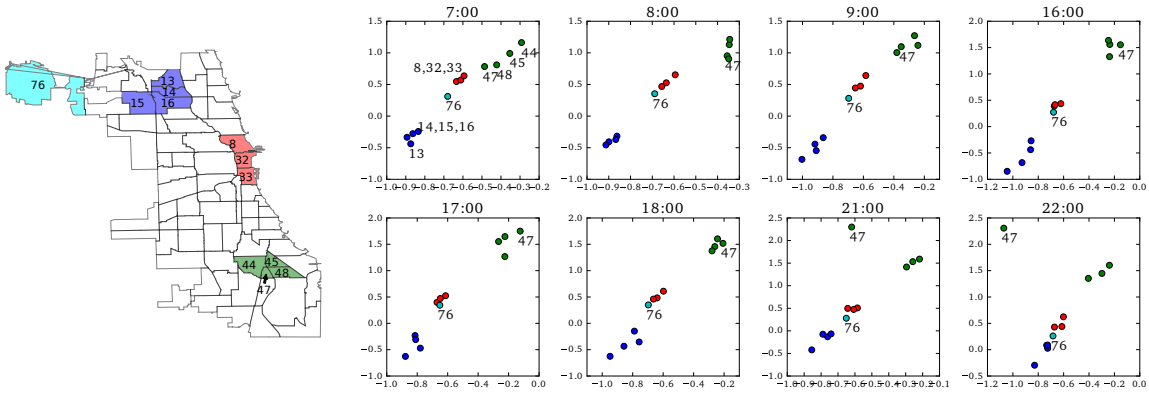


Figure 7: The $nDCG@k$ plot for various methods with the pairwise similarity evaluation. k is the number of regions to retrieve.

For each embedding method, we report the average $nDCG@k$ across all tracts over all timestamps. The results are shown in Figure 7. From the results we made the following several observations.

Overall, the *HDGE* method significantly outperform other embedding methods, such as *MF* and *LINE*. This verifies that the design of flow graph accounts for the POI similarity better than the other embedding methods. The reason is that our flow graph not only consider the temporal dynamics, but also draws connection across different timestamps, which is missing in the slotted graph.



(a) The positions of selected community areas. (b) The 2D embedding visualization of selected community areas during different hours.

Figure 8: Case study with 2D visualization. We pick 12 communities areas, whose positions in the city are shown in (a). The 2D embeddings from different time are visualized in (b). The 12 communities fall in 4 groups: downtown (red), airport (cyan), residential areas (blue), and residential areas with socio-economic issues (green).

It is interesting to notice that when k is small, the $DGE_{spatial}$ gives the best performance, and the performance decreases as the k increases. The reason is that spatially adjacent tracts usually share similar POI distributions. Therefore, given a query tract, a spatial-based method could easily find adjacent tracts as the results for the top 5 other tracts that has the most similar POIs. However, when k is larger than 5, the spatial distance based search does not dominate the results anymore, and thus the performance of $DGE_{spatial}$ decreases.

Although we cannot draw conclusion that *HDGE* is positively correlate with POI information. This experiment concludes that *HDGE* design is better than other embedding methods.

5.3.2 Case Study. To intuitively demonstrate the semantics of *HDGE*, we learn a 2-dimensional embedding with *HDGE* method, and visualize 12 hand-picked community areas that represent four different types of areas.

The locations of these 12 community areas are shown in Figure 8a. In Figure 8a, the blue CA 13, 14, 15, and 16 in the north side are densely populated residential areas of the city, where the resident demographics are mostly middle and upper-class. The red CA 8, 32, and 33 locate in downtown, with many commercial, cultural, and financial institutes. In the south of Chicago, the green CA 44, 45, 47 and 48 have different population demographics from the north side. We also plot the Chicago airport, i.e. CA 76, as cyan region. As shown in the map, the Chicago airport locates in the far northwest side of the city, however, it is noteworthy that there are a significant amount of taxi flow commuting between airports and the rest of the city.

In Figure 8b, we visualize the 2-dimensional embedding of these selected regions from different hours. Particularly, we pick three hours in the morning traffic peak, three hours in the afternoon peak, and two hours at night.

As expected, we observe that spatially adjacent community areas are close in the *HDGE* embedding space. Also, mobility flow helps to identify similar regions beyond spatial adjacency, which explains why the CA 76 is close to downtown area.

An interesting case is observed on CA 47. From the visualization we notice that region 47 has a dramatic change from day to night. During the day time, CA 47 is close to its geographical neighbors, i.e. 44, 45, 48, while at night the embedding of CA 47 is far away from most of the communities. After looking into the taxi trips, we found that there is almost no traffic trip going in or out of CA 47 at night. And the reason behind the extremely low taxi volume is that CA 47 suffers from serious gang violence, so that people are trying to avoid this area at night. In Table 4, we show the taxi flow and crime rate of CA 47 compared to its neighbors.

Table 4: CA 47 suffers from serious gang-related violence, and thus has much less traffic flows compared to its neighbors. The total number of taxi in/out trips are in 2013. The crime rate is gang-related crime count per 10,000 population in 2013.

CA	In	Out	Crime rate	Crime rank
44	4099	5300	124	7
45	857	1611	112	9
47	**221**	**287**	**185**	**1**
48	1935	2848	72	26

6 RELATED WORK

Mobility Data in Urban Problems. Mobility data has been used to solve a wide spectrum of urban problems, such as air quality inference [27], noise pollution estimation [28], real estate ranking [7], and region function detection [15, 18]. In these existing works, the transition matrix is the most frequently used to represent the mobility flow data. However, the transition matrix ignores the temporal information and the multi-hop transitions. To account for the temporal dynamics, Yuan et al. [26] propose a tensor-based framework to discover regions of different functions, which adds a temporal dimension to the transition matrix. Still, the mobility flow tensor can not capture the multi-hop transitions.

Our method differs from the research mentioned above in how we encode the mobility flow information. We try to encode the dynamic mobility flow into vector representations of regions through a embedding method. The advantage of an embedding method over the transition matrix is that the embedding method preserves the global structural information. More specifically, the transition matrix only preserves the pairwise similarity, while the graph embedding is able to make use of higher order proximity and encode such information into the region representations.

Embedding in Heterogeneous Network. Our method is related to the methods of graph embedding and dimension reduction in general. Some typical methods include multidimensional scaling (MDS) [5], IsoMap [21], Laplacian Eigenmap [2], and graph factorization [1]. These methods find the embedding of a graph by representing the graph as an affinity matrix and then applying matrix factorization. However, the objective of matrix factorization does not necessarily preserve the global network structure, because the matrix factorization only captures the pairwise first-order proximity.

Inspired by the word2vec method from the natural language processing field [9–11], which learns continuous vector representations for words, recent research established an analogy for networks by representing a network as a document [8, 17, 20]. One could sample network by random walk to get sequences of vertices and learn a continuous representations for each vertex in a low-dimensional space.

When there are multiple types of vertices and edges in the network, the graph embedding learning objective is different. Wang et al. [24] proposed a word embedding method for linked documents, which learns embedding for words, documents, and document labels. Xie et al. [25] apply the heterogeneous embedding technique in a location network to recommend locations.

Our embedding method is applied on a heterogeneous graph as well, but it is still different from most existing works in heterogeneous network embedding. In our problem, we consider a dynamic graph where the relations between the same pair of vertices are changing over time. This new property presents new challenges in embedding learning.

7 CONCLUSION

In this paper, a graph embedding method is proposed to uncover the urban dynamics using mobility flow data. We define a flow graph to incorporate both temporal dynamics and multi-hop transitions. We also define a spatial graph to address the sparsity issue within the flow graph. The dynamic region embeddings are jointly learned from two graphs. With three inference tasks, we demonstrate the effectiveness of our embedding method.

ACKNOWLEDGEMENTS

The work was supported in part by NSF awards #1544455, #1652525, #1618448, and #1639150. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agencies.

REFERENCES

[1] Amr Ahmed, Nino Shervashidze, Shravan Narayanamurthy, Vanja Josifovski, and Alexander J Smola. 2013. Distributed large-scale natural graph factorization. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 37–48.
[2] Mikhail Belkin and Partha Niyogi. 2001. Laplacian eigenmaps and spectral techniques for embedding and clustering.. In *NIPS*, Vol. 14. 585–591.
[3] United States Census Bureau. 2010. Demographics survey. http://www.census.gov. (2010).
[4] NYC Taxi & Limousine Commission. 2017. NYC Taxi Data. http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml. (2017). Accessed: February, 2017.
[5] Trevor F Cox and Michael AA Cox. 2000. *Multidimensional scaling*. CRC press.
[6] Chicago Digital. 2016. Chicago Taxi Data Released. http://digital.cityofchicago.org/index.php/chicago-taxi-data-released/. (2016). Accessed: November, 2016.
[7] 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.
[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*. ACM.
[9] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).
[10] 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.
[11] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic Regularities in Continuous Space Word Representations.. In *HLT-NAACL*, Vol. 13. 746–751.
[12] U Nations. 2014. World Urbanization Prospects: The 2014 Revision, Highlights. Department of Economic and Social Affairs. *Population Division, United Nations* (2014).
[13] Jeffrey C Nekola and Peter S White. 1999. The distance decay of similarity in biogeography and ecology. *Journal of Biogeography* 26, 4 (1999), 867–878.
[14] City of Chicago data portal. 2015. https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-present/ijzp-q8t2. (2015).
[15] Gang Pan, Guande Qi, Zhaohui Wu, Daqing Zhang, and Shijian Li. 2013. Land-use classification using taxi GPS traces. *IEEE Transactions on Intelligent Transportation Systems* 14, 1 (2013), 113–123.
[16] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global Vectors for Word Representation.. In *EMNLP*, Vol. 14. 1532–43.
[17] 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.
[18] Guande Qi, Xiaolong Li, Shijian Li, Gang Pan, Zonghui Wang, and Daqing Zhang. 2011. Measuring social functions of city regions from large-scale taxi behaviors. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2011 IEEE International Conference on*. IEEE, 384–388.
[19] Foursquare Venues Service. 2015. https://developer.foursquare.com/overview/venues.html. (2015).
[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*. ACM, 1067–1077.
[21] Joshua B Tenenbaum, Vin De Silva, and John C Langford. 2000. A global geometric framework for nonlinear dimensionality reduction. *science* 290, 5500 (2000), 2319–2323.
[22] Alastair J Walker. 1974. New fast method for generating discrete random numbers with arbitrary frequency distributions. *Electronics Letters* 10, 8 (1974), 127–128.
[23] Hongjian Wang, Daniel Kifer, Corina Graif, and Zhenhui Li. 2016. Crime Rate Inference with Big Data. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. ACM, New York, NY, USA, 635–644. https://doi.org/10.1145/2939672.2939736
[24] Suhang Wang, Jiliang Tang, Charu Aggarwal, and Huan Liu. 2016. Linked Document Embedding for Classification. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 115–124.
[25] Min Xie, Hongzhi Yin, Hao Wang, Fanjiang Xu, Weitong Chen, and Sen Wang. 2016. Learning Graph-based POI Embedding for Location-based Recommendation. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 15–24.
[26] Jing Yuan, Yu Zheng, and Xing Xie. 2012. Discovering regions of different functions in a city using human mobility and POIs. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 186–194.
[27] Yu Zheng, Furui Liu, and Hsun-Ping Hsieh. 2013. U-air: When urban air quality inference meets big data. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1436–1444.
[28] Yu Zheng, Tong Liu, Yilun Wang, Yanmin Zhu, Yanchi Liu, and Eric Chang. 2014. Diagnosing New York city's noises with ubiquitous data. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 715–725.
[29] Zillow.com. 2017. Real Estate Value in Chicago. https://www.zillow.com/. (2017).