# Regions, Periods, Activities: Uncovering Urban Dynamics via Cross-Modal Representation Learning

Chao Zhang[1]*, Keyang Zhang[1]*, Quan Yuan[1], Haoruo Peng[1], Yu Zheng[2], Tim Hanratty[3], Shaowen Wang[4], and Jiawei Han[1]
[1]Dept. of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, USA
[2]Microsoft Research, Beijing, China
[3]U.S. Army Research Laboratory, Adelphi, MD, USA
[4]Dept. of Geography & GIS, University of Illinois at Urbana-Champaign, Urbana, IL, USA
[1,4]{czhang82, kzhang53, qyuan, hpeng7, shaowen, hanj}@illinois.edu [2]yuzheng@microsoft.com
[3]timothy.p.hanratty@mail.mil

## ABSTRACT

With the ever-increasing urbanization process, systematically modeling people's activities in the urban space is being recognized as a crucial socioeconomic task. This task was nearly impossible years ago due to the lack of reliable data sources, yet the emergence of geo-tagged social media (GTSM) data sheds new light on it. Recently, there have been fruitful studies on discovering geographical topics from GTSM data. However, their high computational costs and strong distributional assumptions about the latent topics hinder them from fully unleashing the power of GTSM.

To bridge the gap, we present CrossMap, a novel cross-modal representation learning method that uncovers urban dynamics with massive GTSM data. CrossMap first employs an accelerated mode seeking procedure to detect spatiotemporal hotspots underlying people's activities. Those detected hotspots not only address spatiotemporal variations, but also largely alleviate the sparsity of the GTSM data. With the detected hotspots, CrossMap then jointly embeds all spatial, temporal, and textual units into the same space using two different strategies: one is reconstruction-based and the other is graph-based. Both strategies capture the correlations among the units by encoding their *co-occurrence* and *neighborhood* relationships, and learn low-dimensional representations to preserve such correlations. Our experiments demonstrate that CrossMap not only significantly outperforms state-of-the-art methods for activity recovery and classification, but also achieves much better efficiency.

## Keywords

Twitter; urban dynamics; activity; representation learning; social media; spatiotemporal data; geographical topic

*WWW 2017,* April 3–7, 2017, Perth, Australia.
ACM 978-1-4503-4913-0/17/04.
http://dx.doi.org/10.1145/3038912.3052601

## 1. INTRODUCTION

The rapid urbanization process [43, 9] has been nurturing big and complex cities worldwide. As of 2016, more than 54 percent of the world's population live in urban areas, and the percentage is expected to increase to 66 by 2050. 538 of today's cities have a population over one million[1], and 37 of them are megacities that have more than ten million inhabitants. With such substantial urbanization, urban activity modeling is widely recognized as a fundamental task [43, 42] for tackling urban challenges (*e.g.*, excessive energy consumption, air pollution, and traffic congestion). However, people's activities in the urban space are highly dynamic and vary greatly from one region to anther. Due to its intrinsic complexity and the lack of reliable data sources, systematically modeling urban activities was almost impossible years ago. Traditional approaches often rely on costly human surveys, yet the understanding is still coarse-grained and limited in geographical scope.

The recent emergence of geo-tagged social media (GTSM) sheds new light on this task because of its multi-dimensional nature and detailed coverage of urban activities. First, as exemplified in Table 1, every GTSM record consists of a location, a timestamp, and a text message. It provides a multi-dimensional view of the user's activity as he/she probes the urban space as a human sensor. Second, driven by the proliferation of GPS-enabled smartphones, every single day is witnessing millions of GTSM records on different platforms (Twitter, Instagram, Facebook, *etc.*). For instance, more than 10 million geo-tagged tweets are posted in the Twitterverse every day, and more than 10 billion checkins have been accumulated by Foursquare so far. The GTSM data provide an unprecedented coverage of today's major cities [22] and serve as a detailed proxy for understanding human activities [17, 5, 27, 18, 8, 26].

**Table 1: A real geo-tagged tweet created by a Los Angeles user who was watching the Lakers' basketball game at the Staples center.**

| | |
|---|---|
| Location | 34.0430, -118.2673 |
| Time | Apr 13, 2016, 6:50:00 PM |
| Message | Kobe's last game! @ Staples Center |

*Equal contribution.

[1] http://www.citypopulation.de/world/Agglomerations.html

Our goal is to harness the power of massive GTSM data for systematically characterizing people's activities in urban spaces. Although techniques [31, 35, 19] have been proposed for geographical topic modeling, they are inadequate for our goal because they do not meet the following requirements: (1) *Comprehensiveness.* There are three key factors involved in people's activities: location, time, and text. The above methods jointly analyze location and text to reveal the topics in different regions, but all ignore the time factor. As a result, they cannot capture the temporal dynamics in the same region (*e.g.*, a shopping area during the daytime might become a center for nightlife activities in the evening) [41]. (2) *Robustness.* They are all generative models that impose distributional assumptions for the latent topics (*e.g.*, defining the spatial distribution of each topic as Gaussian). Although such assumptions simplify model inference, they may not fit real-life data well and are sensitive to noise. (3) *Scalability.* For urban activity modeling, the power of GTSM data can be fully unlocked only when a sheer amount of them is used. Existing geographical topic modeling methods, which rely on probabilistic graphical models, cannot easily scale up to massive GTSM records.

We propose CrossMap, a novel method that models urban activities from massive GTSM data via cross-modal representation learning. CrossMap automatically extracts the spatiotemporal hotspots underlying people's activities, and maps all spatial, temporal, and textual units into the same space with their cross-modal correlations well preserved. With CrossMap, various questions can be answered regarding people's urban activities, such as: (1) how are different regions, time periods, and activities correlated? (2) given an activity, where and when does the activity usually happen? (3) what are the popular activities at a given location and time? Better still, CrossMap is highly useful for a wide spectrum of downstream applications. For example, it can empower mobile gadgets that allow tourists to explore an unfamiliar metropolis (*e.g.*, New York City) conveniently. At any time, a tourist just needs to take out her mobile phone, then the gadget can help discover the popular activities around her — based on the digital traces from millions of people in New York City. As another example, the embeddings learnt by CrossMap can empower a mobile personal assistant that continuously tracks the visited places of a user and infers her preferences. Based on her preferences along with current location and time, the assistant can smartly suggest to the user activities/venues of interest, achieving situational and immersive recommendation experience.

Technically, the contributions of CrossMap are highlighted by the following two modules:

1. A *hotspot detector* that detects spatial and temporal hotspots to address spatiotemporal continuity (Section 4). Unlike the text dimension where keywords are natural basic units for embedding, the space and time are continuous and it is infeasible to embed every location and timestamp. To address this problem, we introduce an accelerated mode seeking procedure that detects spatial and temporal hotspots based on kernel density estimation. Such hotspots, representing the geographical regions and time periods where people's activities burst, effectively address the spatiotemporal variations and alleviate data sparsity.
2. An *embedding module* that maps all the spatial, temporal, and textual units into a latent space to preserve their cross-modal correlations (Section 5). We design two different embedding strategies, both of which can capture the co-occurrence and neighborhood relationships among the units. The first is reconstruction-based. It directly considers each record as an observed relation, and learns the embeddings such that the observed relations can be reconstructed as much as possible. The second is graph-based. It uses a heterogeneous graph to encode the correlations among the units and then learns the low-dimensional representations of the nodes to preserve the graph structure.

We evaluate CrossMap on two large-scale GTSM data sets: one contains geo-tagged tweets in Los Angeles, and the other contains Foursquare checkins in New York City. Our experiments demonstrate that the urban activity model obtained by CrossMap is of high quality and outperforms state-of-the-art methods for recovering user activities significantly. In addition, CrossMap can easily process millions of GTSM records, making it suitable for handling big GTSM data. Finally, with activity classification as an example, we show that the embeddings learnt by CrossMap are highly useful for downstream applications.

## 2. RELATED WORK

**Urban function modeling.** There has been considerable research that leverages GTSM data for understanding urban functions and discovering urban communities. Noulas *et al.* [28] use venue categories to generate fingerprints for users and areas, which facilitate discovering semantically coherent user/area clusters. Frias-Martinez *et al.* [11] use geo-tagged tweets to obtain the temporal patterns of land segments, and then cluster those segments to discover urban landscapes. Cranshaw *et al.* [8] extract urban communities with an affinity measure that considers both spatial distance and social proximity. Noulas *et al.* [26] treat venue categories as labels, and extract features to infer the popular activities around cell towers. Zhang *et al.* [41] demonstrate that the time factor plays an important role in revealing people's activities in different urban regions. The above studies either ignore the semantics of user activities, or simply use venue categories — which are coarse-grained and unavailable in many data sets (*e.g.*, tweets, Instagram posts). In contrast, we deal with the raw text to provide fine-grained understanding of urban regions, which makes our problem different.

**Geographica topic discovery.** Geographical topic discovery [31, 19, 35, 15, 36, 24, 34] aims at modeling the topics in different regions. Sizov *et al.* [31] extend LDA [2] by assuming each latent topic has a multinomial distribution over text, and two Gaussians over latitudes and longitudes. They later extend the model to find topics that have complex and non-Gaussian distributions [19]. Yin *et al.* [35] extend PLSA [14] by assuming each region has a normal distribution that generates locations, as well as a multinomial distribution over the latent topics that generates text. While the above models are designed to detect global geographical topics, Hong *et al.* [15] and Yuan *et al.* [36] introduce the user factor in the modeling process such that users' individual-level preferences can be inferred. Our work resembles the studies [31, 19, 35] more because we also model global-level urban activities instead of individual-level preferences. That said, there are two notable differences between our work and [31, 19, 35]. First, these studies have not considered the time factor during the modeling process, and thus cannot distinguish different activities happening in different time periods. Second, from the technical perspective, they all extend topic models. In comparison, we develop embedding techniques to capture cross-modal correlations in a more direct and scalable way.

**Local event detection.** A handful of studies [21, 4, 30, 10, 1, 40] use GTSM for local event detection. Sakaki *et al.* [30] train a classifier to judge whether an incoming tweet is related to earthquake or not, and release an alarm when the number of earthquake-related tweets is large. Krumm *et al.* [20] monitor the spatiotemporal distributions of tweet streams, and detect spikes in the signal as interesting events. Zhang *et al.* [40] propose a method that achieves real-time local event detection from geo-tagged tweet streams. There is a clear difference between local event detection and urban activity modeling. The former attempts to extract unusual activities bursted in local areas, whereas the latter aims at summarizing the typical activities at different locations and time.

**Activity pattern discovery.** There have also been studies that leverage GTSM data to extract the patterns underlying people's activities. In pioneering studies, Noulas *et al.* [27] analyze user activities with massive Foursquare checkins and find the checkin data can reveal meaningful spatiotemporal activity patterns; Cho *et al.* [6] collect large-scale checkin data and find that people's activities are usually centered around a few fixed locations, and exhibit strong periodicity. Later, Yuan *et al.* [37] propose a Bayesian non-parametric model to automatically extract the regions that a user visits periodically; Zhang *et al.* [38] apply sequential pattern mining techniques to extract frequent movement sequences from check-in data. Very recently, Zhang *et al.* [39] apply the Hidden Markov Model to GTSM data, observing that there are latent states underlying people's daily activities and people move between them with strong regularity; Hristova *et al.* [16] capture the social diversity of urban places based on joint analysis of the social network and the mobility patterns of the visitors. Our task is different from these studies, as none of them attempt to capture the inter-type correlations between location, time, and text.

**Embedding methods.** Various methods have been proposed for embedding different data types, such as words [25], graph nodes [29, 33, 32], and heterogeneous events [12]. Our study differs from the above methods because of the unique characteristics of GTSM data. First, due to spatiotemporal continuities, there are no natural basic spatiotemporal units, and representative hotspots must be first detected. Second, in addition to co-occurrences, the spatiotemporal continuities call for methods that are tailored to capture the similarities between nearby locations and timestamps.

# 3. PRELIMINARIES

## 3.1 Problem Description

Let $\mathcal{C}$ be a corpus of geo-tagged social media (GTSM) records. Each record $r \in \mathcal{C}$ is defined by a tuple $\langle t_r, l_r, m_r \rangle$ where: (1) $l_r$ is a two-dimensional vector that represents the user's location when $r$ is created; (2) $t_r$ is the creating time[2]; and (3) $m_r$ is a bag of keywords denoting the text message of $r$.

We aim to use a large amount of GTSM records to model people's activities in the urban space. As there are three different factors (*i.e.*, location, time, and text) that are intertwined, an effective urban activity model should accurately capture their cross-modal correlations. Given any two of the three factors, the activity model is expected to predict the remaining one. Specifically: (1) What are the typical activities occurring at a specific location and time? (2) Given an activity and time, where does this activity usually take place? and (3) Given an activity and a location, when does the activity usually happen?

## 3.2 Overview of CrossMap

Before presenting CrossMap, we identify two unique challenges of modeling urban activities with GTSM data, which motivate the design of CrossMap:

1) **Spatiotemporal variation.** The raw GTSM data exhibits considerable spatiotemporal variations. Consider a crowd of basketball fans. Even if they watched the same game at the same stadium, their created tweets may contain slightly different GPS coordinates and timestamps. To avoid data sparsity, it is important to tackle such spatiotemporal variations instead of simply considering every location or timestamp as independent. Nevertheless, how to effectively and efficiently address spatiotemporal variations without prior knowledge is challenging.

2) **Cross-modal correlation.** An effective urban activity model should accurately capture the cross-modal correlations between location, time, and text. For this purpose, existing models [31, 35, 19] assume latent states that generate multi-dimensional observations according to pre-defined distributions (*e.g.*, assuming the location follows Gaussian). Nevertheless, the distributional assumptions may not fit the real data well. For example, beach-related activities are usually distributed along coastlines that have complex shapes, and cannot be well modeled by a Gaussian distribution. Further, learning such generative models is usually time-consuming. Hence, can we capture the cross-modal correlations more directly?

To address the first challenge, our exploration into real-life GTSM data reveals that, people's activities in urban environments usually burst in certain geographical regions (*e.g.*, shopping area, airport) and time periods (*e.g.*, having lunch at noon). In other words, there are latent spatial and temporal *hotspots* that lead to the observed locations and time. Consider the above basketball game example: the stadium area serves as a spatial hotspot that leads to various location observations around it, and the game time is a temporal hotspot that generates multiple timestamps. We thus design an accelerated mode seeking module in CrossMap, which is a non-parametric procedure that fast detects such spatiotemporal hotspots without knowing the number of hotspots beforehand. The detected hotspots are used as basic units in later embedding process. In this way, the locations (timestamps) that lie around the same spatial (temporal) hotspot can be grouped to address spatiotemporal variations. Better still, as correlated locations and timestamps are grouped, the data sparsity can be largely

[2] We convert the raw time to the range of [0, 86400] by calculating its offset (in second) w.r.t. 12:00am.

reduced. It is worth mentioning that, once the embeddings of the spatial (temporal) hotspots are learnt, the embeddings of any ad-hoc locations (timestamps) can be easily obtained using interpolation.

Relying on the detected spatiomtemporal hotspots, we then develop a joint embedding module to effectively and efficiently capture the cross-modal correlations between location, time, and text. Different from existing generative models that use latent states to indirectly bridge different data types, our embedding procedure directly captures the cross-modal correlations by mapping all the units into a common latent space. We propose two strategies to learn the embeddings of different units: (1) The first is a reconstruction-based strategy, which considers every record as a multi-dimensional relation, and learns the embeddings to maximize the likelihood of observing the given records; and (2) The second is a graph-based strategy, which first constructs a heterogeneous graph to encode the proximities of different units, and then learns the embeddings of all the nodes to preserve the graph structure.

# 4. THE HOTSPOT DETECTOR

## 4.1 Spatial and Temporal Hotspots

Our definitions of spatial and temporal hotspots are based on kernel density estimation, which is a non-parametric way to estimate the density function from a finite set of samples. The nice property of kernel density estimation is that, it does not assume any prior knowledge about the underlying data distribution, and thus can flexibly discover arbitrary modes in a complex data space. Given $n$ data points $\mathbf{x}_i$ ($i = 1, \dots, n$) in the $d$-dimensional space $R^d$, the kernel density at any point $\mathbf{x}$ is given by

$$f(\mathbf{x}) = \frac{1}{nh^d} \sum_{i=1}^{n} K(\frac{\mathbf{x} - \mathbf{x}_i}{h}),$$

where $K(\cdot)$ is a kernel function and $h$ is the kernel bandwidth. While various kernel functions can be chosen, we use the Epanechnikov kernel [7] due to its simplicity and optimality in terms of bias-variance tradeoff. We now define spatial and temporal hotspots as kernel density maxima in the two-dimensional and one-dimensional spaces, respectively.

Definition 1 (Spatial and Temporal Hotspots). *Given a GTSM corpus $\mathcal{C}$, let $\mathcal{L}$ be the collection of locations in $\mathcal{C}$, a spatial hotspot is a local maximum of the kernel density function estimated from $\mathcal{L}$. Similarly, let $\mathcal{T}$ be the collection of timestamps in $\mathcal{C}$, a temporal hotspot is a local maximum of the kernel density in $\mathcal{T}$.*

## 4.2 Hotspot Detection

We detect spatial and temporal hotspots by adapting a popular mode seeking method, the mean shift [7] algorithm. For a $d$-dimensional point, mean shift finds its corresponding mode by iteratively shifting a radius-$h$ window towards a local density maxima. The window is called the kernel window and the radius is called the bandwidth. In each iteration, let $\mathbf{y}^{(k)}$ be the center of current window, and $\mathcal{N} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m\}$ be the $m$ data points inside the window, then the kernel window is shifted towards the maximum increase of density for $\mathbf{y}^{(k)}$. Using the Epanechnikov kernel, the mean shift vector for $\mathbf{y}^{(k)}$ is $\mathbf{m}(\mathbf{y}^{(k)}) = (\sum_{i=1}^{m} \mathbf{x}_i - \mathbf{y}^{(k)})/m$. Then $\mathbf{y}^{(k)}$ is shifted by $\mathbf{m}(\mathbf{y}^{(k)})$, resulting in a new kernel window located at the mean of $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m\}$, namely

$$\mathbf{y}^{(k+1)} = \mathbf{y}^{(k)} + \mathbf{m}(\mathbf{y}^{(k)}) = \frac{1}{m} \sum_{i=1}^{m} \mathbf{x}_i. \tag{1}$$

Consider detecting spatial hotspots as an example. As shown in Figure 1, given a location $\mathbf{x}$, we start with an initial window centered at $\mathbf{y}^{(0)} = \mathbf{x}$, and iteratively shift the window according to Equation 1. The sequence $\{\mathbf{y}^{(k)}\}$ will converge to the hotspot $\mathbf{x}$ belongs to. After performing the mean shift procedure from every data point, all the hotspots can be detected.

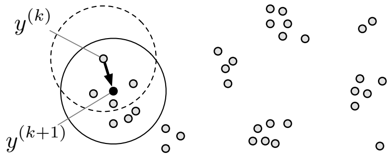


**Figure 1: Spatial hotspot detection with mean shift.**

Nevertheless, the standard mean shift algorithm has a time complexity of $O(KN^2)$, where $N$ is the total number of points, and $K$ is the average number of shifting steps for each point. Such a time complexity renders it inefficient for large data sets with millions of points. To make the mean shift algorithm capable of handling massive GTSM records, we adopt the space discretization strategy [3], which partitions the whole space into small equal-size cells and treat each cell as a basic shifting unit. As such, all the points that fall in the same cell will converge to the same hotspot.

Built upon space partitioning, we further propose a simple yet effective acceleration strategy. The key idea is to accelerate the shifting operation based on pre-computation. Consider spatial hotspot detection as an example. As shown in Figure 2, after partitioning the space into small cells, we build an index to record the points inside each cell. Meanwhile, for each cell $c$, we maintain two statistics: 1) $N_c$ is the number of points inside $c$; and 2) $S_c$ is the vector sum of the points in $c$. Such an indexing operation has $O(N)$ time complexity and only needs to be performed once.

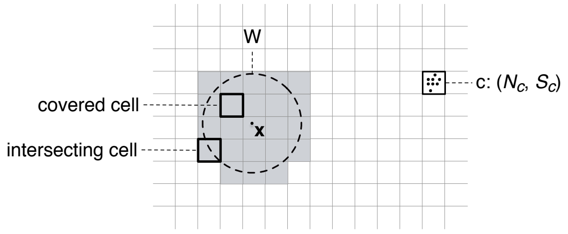


**Figure 2: Accelerated mean shift based on pre-computation. For each cell $c$, we maintain 1) $N_c$: the number of points in $c$; and 2) $S_c$: the vector sum of the points in $c$.**

The index can largely reduce the overhead for kernel center shifting. Specifically, let $W$ be a kernel window centered at $\mathbf{x}$. To shift the kernel center, we do not need to scan all the input points and find the ones inside $W$. Instead, we use the index to retrieve all the cells that overlap with

$W$ in constant time. Such cells contain all the points that contribute to computing the new kernel center, and can be categorized into two classes: 1) The first contains the cells that are entirely covered by $W$. For a covered cell $c$, there is no need to access the points inside $c$. Rather, we just use the pre-computed $N_c$ and $S_c$ to account for $c$'s contributions to the new center; and 2) The second contains the cells that intersect with $W$. For an intersecting cell $c$, we access the points inside it, compute their distances to $\mathbf{x}$, and preserve the points inside $W$. By combining the contributions from the covered and the intersecting cells, the new kernel center can be fast obtained. As a large amount of points that are guaranteed to be outside $W$ are never accessed, the time cost of the shifting operation is reduced significantly.

# 5. THE JOINT EMBEDDING MODULE

In this section, we describe the joint embedding module that maps all spatial, temporal, and textual units into a common latent space. Here, a spatial unit is a spatial hotspot, a temporal unit is a temporal hotspot, and a textual unit is a keyword. To learn high-quality embeddings of those units, we claim that two important relationships among them should be captured: the co-occurrence and neighborhood relationships.

The *co-occurrence relationship* exists between two units when they co-occur in the same record. Consider the tweet in Table 1 as an example. As shown, a spatial unit (34.0430, -118.2673), a temporal unit (6:50pm), and several textual units (*e.g.*, Kobe, game) occur in the same tweet. Their co-occurrences reflect their intrinsic correlations — they are all related to the Lakers' game.

The *neighborhood relationship* stems from spatial and temporal continuities. Take the spatial continuity as an example. According to the first law of geography: *everything is related to everything else, but near things are more related than distant things.* To achieve spatial smoothness, two spatial units that are close to each other should be considered correlated instead of independent. We thus introduce the *neighborhood relationship* to capture spatiotemporal proximities based on Gaussian kernels.

DEFINITION 2 (SPATIAL AND TEMPORAL KERNEL). *For two spatial hotspots* $\mathbf{x}$ *and* $\mathbf{y}$*, the kernel strength between* $\mathbf{x}$ *and* $\mathbf{y}$ *is:*

$$w(\mathbf{x},\mathbf{y}) = \begin{cases} \exp(-\|\mathbf{x}-\mathbf{y}\|^2/(2\sigma_s^2))/(2\pi\sigma_s^2) & if \|\mathbf{x}-\mathbf{y}\| \le \sigma_s, \\ 0 & otherwise, \end{cases}$$

*where* $\sigma_s$ *is a kernel bandwidth for spatial continuity. Similarly, for two temporal hotspots* $x$ *and* $y$*, their kernel strength is:*

$$w(x,y) = \begin{cases} \exp(-(x-y)^2/(2\sigma_t^2))/(\sqrt{2\pi}\sigma_t) & if |x-y| \le \sigma_t, \\ 0 & otherwise, \end{cases}$$

*where* $\sigma_t$ *is a kernel bandwidth for temporal continuity.*

DEFINITION 3 (SPATIAL AND TEMPORAL NEIGHBORHOOD). *For a spatial (temporal) hotspot* $\mathbf{x}$*, its spatial (temporal) neighborhood* $\mathcal{N}_\mathbf{x}$ *is the set of spatial (temporal) hotspots whose kernel strengths to* $\mathbf{x}$ *are non-zero.*

In what follows, we design two algorithms to encode the co-occurrence and neighborhood relationships and learn high-quality cross-modal embeddings. Our first embedding algorithm is reconstruction-based, named RECONEMBED; and our second algorithm is graph-based, named GRAPHEMBED.

## 5.1 The ReconEmbed Algorithm

RECONEMBED learns the embeddings of different units such that the observed records can be reconstructed as much as possible. Given a record $r$, for any unit $i \in r$ with type $X$ (could be location, time, or text), we model the likelihood of observing $i$ as

$$p(i|r_{-i}) = \exp(s(i,r_{-i}))/\sum_{j\in X}\exp(s(j,r_{-i})),$$

where $r_{-i}$ is the set of all the units in $r$ except $i$, and $s(i,r_{-i})$ is the similarity score between $i$ and $r_{-i}$. The key is how to define $s(i,r_{-i})$. A natural idea is to average the embedding vectors of the units in $r_{-i}$, and compute $s(i,r_{-i})$ as $s(i,r_{-i}) = \mathbf{v}_i^{\mathrm{T}}\sum_{j\in r_{-i}}\mathbf{v}_j/|r_{-i}|$. Nevertheless, such a simple definition fails to encode the neighborhood relationships among the spatial and temporal units.

To tackle the above issue, we propose a kernel-smoothed version of $s(i,r_{-i})$. Suppose the target unit $i$ is a word, and letting $l \in r_{-i}$ be the spatial hotspot in $r_{-i}$, then word $i$ is related to not only $l$, but also other hotspots in $l$'s neighborhood. Hence, we define a pseudo spatial hotspot $\hat{l}$, whose embedding is the weighted average of the hotspots in $l$'s neighborhood, namely $\mathbf{v}_{\hat{l}} = \sum_{l'\in\mathcal{N}_l} w(l,l')\mathbf{v}_{l'}/\sum_{l'\in\mathcal{N}_l} w(l,l')$. Similarly, for the temporal hotspot $t \in r_{-i}$, we define a pseudo temporal hotspot $\hat{t}$, whose embedding is defined as $\mathbf{v}_{\hat{t}} = \sum_{t'\in\mathcal{N}_t} w(t,t')\mathbf{v}_{t'}/\sum_{t'\in\mathcal{N}_t} w(t,t')$. Meanwhile, we define a pseudo word embedding: $\mathbf{v}_{\hat{w}} = \sum_{w\in r_{-i}}\mathbf{v}_w/N_w(r_{-i})$, where $N_w(r_{-i})$ is word count in $r_{-i}$. With those pseudo embeddings, we define the kernel-smoothed $s(i,r_{-i})$ as

$$s(i,r_{-i}) = \mathbf{v}_i^{\mathrm{T}}\mathbf{h}_i,$$

where $\mathbf{h}_i$ is the average of the pseudo spatial, temporal, and textual embeddings for the elements in $r_{-i}$.

Finally, the loss function of RECONEMBED is simply the total negative log-likelihood of observing all the units of the given records:

$$O = -\sum_{r\in\mathcal{C}}\sum_{i\in r}\log p(i|r_{-i}). \tag{2}$$

To efficiently optimize the above objective, we use stochastic gradient descent (SGD) and negative sampling [25]. At each time, we use SGD to sample a record $r$ and a unit $i \in r$. With negative sampling, we randomly select $K$ negative units that have the same type with $i$ but do not appear in $r$, then the loss function for the selected samples becomes:

$$L = -\log\sigma(s(i,r_{-i})) - \sum_{k=1}^{K}\log\sigma(-s(k,r_{-i})),$$

where $\sigma(\cdot)$ is the sigmoid function. Using SGD, the updating rules for different variables can be easily obtained by taking the derivatives of the above objective function.

## 5.2 The GraphEmbed Algorithm

GRAPHEMBED uses a heterogeneous graph to encode the proximities of different units and learns low-dimensional representations of the nodes to preserve the graph structure. Below, we first describe how to construct the heterogeneous graph, and then present the learning procedure.

### *5.2.1 Heterogeneous Graph Construction*

We use a heterogeneous graph to encode both the co-occurrence and neighborhood relationships. In the graph, there are three different node types: location, time, and text. The edges among the nodes are constructed as below. First, as each record consists of one spatial unit, one temporal unit, and multiple textual units, the co-occurrence relationship induces four edge types: (1) *word-word* edge; (2) *word-time* edge; (3) *word-location* edge; and (4) *time-location* edge. Within each edge type, we set the edge weight to the normalized co-occurrence count. Second, the neighborhood relationship induces two edge types: (1) *location-location* edge; and (2) *time-time* edge. For any spatial (temporal) hotspot $\mathbf{x}$, we connect it with its spatial (temporal) neighbors, and set the edge weights to the kernel strengths.

### 5.2.2 Learning Graph Embeddings

Now the question is how to learn quality embeddings of the graph nodes to preserve their proximities. Our idea is to model the emission probability distribution of each node based on the latent embeddings, and make such distributions close to the true observed distributions. In this way, the latent embeddings can capture not only explicit co-occurrence information, but also implicit interactions among the units — two nodes sharing many common neighbors tend to have similar embeddings even if they are not directly connected.

Consider a node $i$ with node type $X$, and a node $j$ with node type $Y$. Based on the latent embeddings, we model the likelihood of generating node $j$ given node $i$ as

$$p(j|i) = \exp(-\mathbf{v}_j'^{\mathrm{T}} \cdot \mathbf{v}_i) / \sum_{k \in Y} \exp(\mathbf{v}_k'^{\mathrm{T}} \cdot \mathbf{v}_i). \qquad (3)$$

Note that each node $i$ has two different embedding vectors: $\mathbf{v}_i$ is the vector when $i$ acts the given center node, while $\mathbf{v}_i'$ is the vector when $i$ is the emitted context node. Equation 3 gives the embedding-based distribution for node $i$.

Meanwhile, the true observed distribution of $i$ is defined as follows: let $w_{ij}$ be the weight of the edge $e_{ij}$, and $d_i = \sum_{j' \in Y} w_{ij'}$ be the total out-degree of node $i$ for node type $Y$. The true emission distribution of node $i$ is given by $\hat{p}(j|i) = w_{ij}/d_i$. To make the embedding-based distributions close to the observed distributions, for any two node types $X$ and $Y$, we define the loss function for the subgraph $G_{XY}$ as

$$O_{XY} = \sum_{i \in X} d_i \mathrm{KL}(p'(\cdot|i)||p(\cdot|i)) + \sum_{j \in Y} d_j \mathrm{KL}(p'(\cdot|j)||p(\cdot|j)),$$

where $\mathrm{KL}(\cdot)$ is the KL-divergence measure. As there are three different node types in the graph, the overall loss functions is

$$O = O_{WW} + O_{LL} + O_{TT} + O_{WL} + O_{WT} + O_{LT}. \qquad (4)$$

We use SGD to optimize the objective by alternating between those edge types with negative sampling. For a directed edge $e_{ij}$, we randomly select $K$ nodes that do not connect to node $i$. We consider node $j$ as a positive example, and the $K$ nodes as negative examples, then minimize the following function:

$$L = -\log \sigma(\mathbf{v}_j'^{\mathrm{T}} \cdot \mathbf{v}_i) - \sum_{k=1}^{K} \log \sigma(-\mathbf{v}_k'^{\mathrm{T}} \cdot \mathbf{v}_i).$$

The updating rules for different variables can be easily derived by taking the derivatives of the above objective, we omit the details to save space.

# 6. EXPERIMENTS

## 6.1 Experimental Setup

**Data Sets.** Our experiments are based on two real-life GTSM data sets: 1) Tweet is a data set collected from Twitter. It consists of around 1.1 million geo-tagged tweets published in Los Angeles during the time period 2014.08.01 - 2014.11.30; and 2) 4SQ is collected from Foursquare, consisting of around 0.6 million Foursquare checkins posted in New York during 2010.08 - 2011.10. For both data sets, we stem the text and remove stopwords as well as the keywords that appear less than 100 times in the entire corpus.

**Baselines.** We compare CrossMap with the following baseline methods:

- LGTA [35] is a geographical topic model that assumes a number of latent spatial regions — each described by a Gaussian. Meanwhile, each region has a multinomial distribution over the latent topics that generate text.
- MGTM [19] is a state-of-the-art geographical topic model based on the multi-Dirichlet process. It is capable of finding geographical topics with non-Gaussian distributions.
- TF-IDF constructs the co-occurrence matrices between each pair of the three data types (location, time, and text); and then transforms every element in each matrix to its td-idf weight by treating each row as a document and each column as a word.
- SVD first constructs the co-occurrence matrices between each pair of location, time, and text, and then performs Singular Value Decomposition on the constructed matrices.
- Tensor [13] builds a tensor to encode the co-occurrences among location, time, and text. It then factorizes the tensor to obtain low-dimensional representations of all the units.

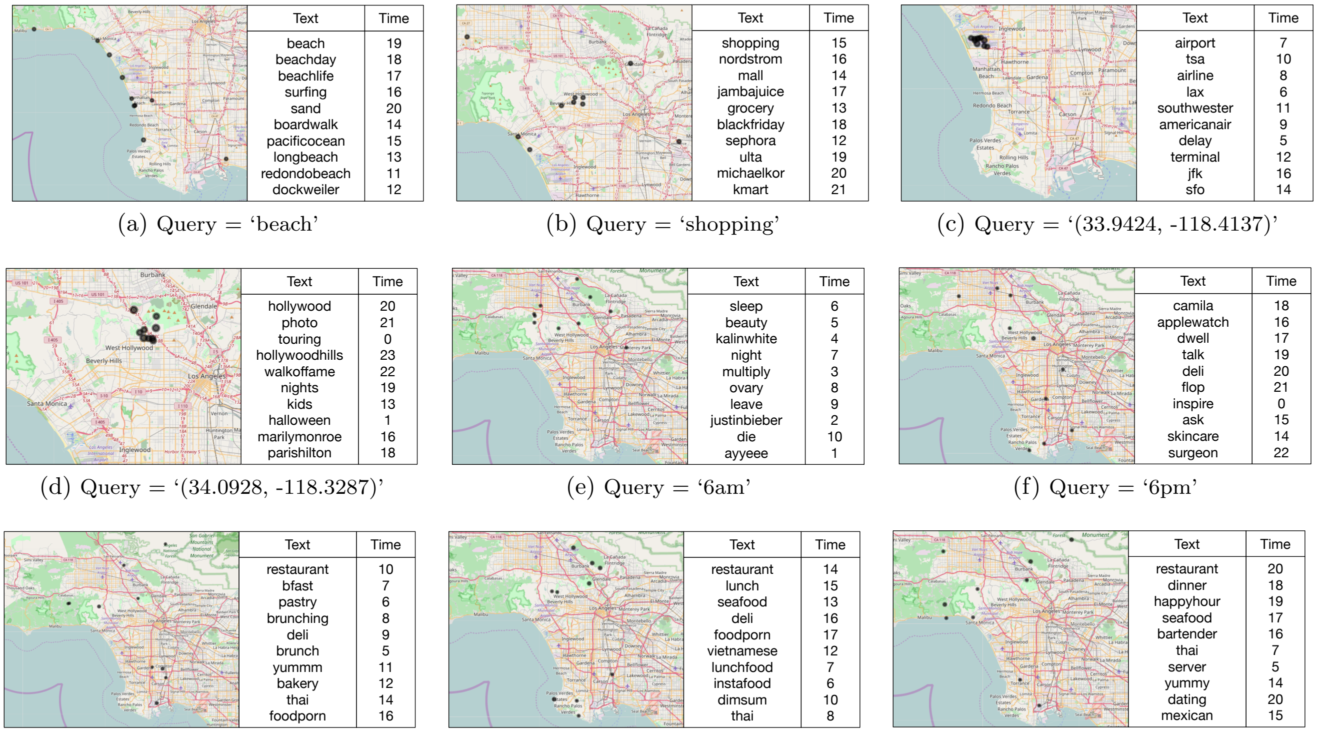
It is worth mentioning that TF-IDF, SVD and Tensor are all built upon the spatiotemporal hotspots detected by CrossMap for fair comparisons.

**Parameter Settings.** The major parameters of CrossMap include: (1) the spatial hotspot detection bandwidth $h_s$; (2) the temporal hotspot detection bandwidth $h_t$; (3) the spatial proximity bandwidth $\delta_s$; (4) the temporal proximity bandwidth $\delta_t$; and (5) the latent embedding dimension $D$. By default, we set $h_s = 0.002$, $h_t = 1000$, $\delta_s = 0.005$, $\delta_t = 4000$, and $D = 300$. With such a setting, the hotspot detector obtains around 10000 spatial hotspots and 22 temporal hotspots, which we believe are reasonable numbers for a metropolis (like Los Angeles and New York City) and a time range of one day. In LGTA, there are two major parameters, the number of regions $R$, and the number of latent topics $Z$. We set $R = 300$ and $Z = 10$. MGTM is a nonparametric method but involves several hyper-parameters. We set the hyper-parameters following the original paper [19]. For SVD and Tensor, we set the latent dimension as $D = 300$ to ensure fair comparisons with CrossMap.

## 6.2 Experimental Results

### 6.2.1 Illustrative Cases

We first use examples to verify whether the learnt embeddings of CrossMap indeed capture the cross-modal correlations between location, time, and text. To this end, we launch some sample queries on Tweet, and retrieve the top



| Text | Time |
|---|---|
| beach | 19 |
| beachday | 18 |
| beachlife | 17 |
| surfing | 16 |
| sand | 20 |
| boardwalk | 14 |
| pacificocean | 15 |
| longbeach | 13 |
| redondobeach | 11 |
| dockweiler | 12 |

(a) Query = ‘beach’

| Text | Time |
|---|---|
| shopping | 15 |
| nordstrom | 16 |
| mall | 14 |
| jambajuice | 17 |
| grocery | 13 |
| blackfriday | 18 |
| sephora | 12 |
| ulta | 19 |
| michaelkor | 20 |
| kmart | 21 |

(b) Query = ‘shopping’

| Text | Time |
|---|---|
| airport | 7 |
| tsa | 10 |
| airline | 8 |
| lax | 6 |
| southwester | 11 |
| americanair | 9 |
| delay | 5 |
| terminal | 12 |
| jfk | 16 |
| sfo | 14 |

(c) Query = ‘(33.9424, -118.4137)’

| Text | Time |
|---|---|
| hollywood | 20 |
| photo | 21 |
| touring | 0 |
| hollywoodhills | 23 |
| walkoffame | 22 |
| nights | 19 |
| kids | 13 |
| halloween | 1 |
| marilymonroe | 16 |
| parishilton | 18 |

(d) Query = ‘(34.0928, -118.3287)’

| Text | Time |
|---|---|
| sleep | 6 |
| beauty | 5 |
| kalinwhite | 4 |
| night | 7 |
| multiply | 3 |
| ovary | 8 |
| leave | 9 |
| justinbieber | 2 |
| die | 10 |
| ayyeee | 1 |

(e) Query = ‘6am’

| Text | Time |
|---|---|
| camila | 18 |
| applewatch | 16 |
| dwell | 17 |
| talk | 19 |
| deli | 20 |
| flop | 21 |
| inspire | 0 |
| ask | 15 |
| skincare | 14 |
| surgeon | 22 |

(f) Query = ‘6pm’

| Text | Time |
|---|---|
| restaurant | 10 |
| bfast | 7 |
| pastry | 6 |
| brunching | 8 |
| deli | 9 |
| brunch | 5 |
| yummm | 11 |
| bakery | 12 |
| thai | 14 |
| foodporn | 16 |

(g) Query = ‘restaurant’ + ‘10am’

| Text | Time |
|---|---|
| restaurant | 14 |
| lunch | 15 |
| seafood | 13 |
| deli | 16 |
| foodporn | 17 |
| vietnamese | 12 |
| lunchfood | 7 |
| instafood | 6 |
| dimsum | 10 |
| thai | 8 |

(h) Query = ‘restaurant’ + ‘2pm’

| Text | Time |
|---|---|
| restaurant | 20 |
| dinner | 18 |
| happyhour | 19 |
| seafood | 17 |
| bartender | 16 |
| thai | 7 |
| server | 5 |
| yummy | 14 |
| dating | 20 |
| mexican | 15 |

(i) Query = ‘restaurant’ + ‘8pm’

**Figure 3: Illustrative queries for CrossMap. Figure 3(a) and 3(b) are textual queries; Figure 3(c) and 3(d) are spatial queries; Figure 3(e) and 3(f) are temporal queries; Figure 3(g), 3(h) and 3(i) are textual + temporal queries.**

ten most similar units in each data type, with vector cosine distance as the similarity measure.

**Textual Queries.** Figure 3(a) and 3(b) show the results when we query with the keywords ‘beach’ and ‘shopping’. One can see the retrieved units in each type are quite meaningful: (1) For the query ‘beach’, the top locations mostly fall in famous beach areas in Los Angeles; the top keywords reflect people’s activities on the beach, such as ‘sand’ and ‘boardwalk’; the top time slots are in the late afternoon, which are indeed good time to enjoy the beach life. (2) For the query ‘shopping’, the retrieved locations are at popular malls and outlets in Los Angeles; the keywords (*e.g.*, ‘nordstrom’, ‘mall’, ‘blackfriday’) are either brand names or shopping-related nouns; and the time slots are mostly around 3pm in the afternoon, matching people’s real-life shopping patterns intuitively.

**Spatial Queries.** Figure 3(c) and 3(d) show the results for two spatial queries: (1) the location of the LAX airport; and (2) the location of Hollywood. Again, we can see the retrieved top spatial, temporal, and textual units are closely related to airport and Hollywood, respectively. For instance, given the query at LAX, the top keywords are all meaningful concepts that reflect flight-related activities, such as ‘airport’, ‘tsa’, and ‘airline’.

**Temporal Queries.** Figure 3(e) and 3(f) show the results when we query with two timestamps: 6am and 6pm. We find the results in each list make practical sense (*e.g.*, keywords like ‘sleep’ are ranked high for the query ‘6am’), but are less coherent compared with those of spatial and textual queries. This phenomenon is reasonable. As later we will verify with quantitative studies, people’s activities in the same time slot could vary greatly. For instance, it is common that people have different activities at 6pm, ranging from having food to shopping and working. Therefore, the temporal signal alone cannot easily determine people’s activities or locations.

**Temporal-Textual Queries.** Figure 3(g), 3(h), and 3(i) show some temporal-textual queries to demonstrate the temporal dynamics of urban activities. As we fix the query keyword as ‘restaurant’ and vary the time, the retrieved units change obviously. Examining the top keywords, we can see the query ‘10am’ leads to many breakfast-related keywords in the list, such as ‘bfast’ and ‘brunch’. In contrast, the query ‘2pm’ retrieves many lunch-related ones while ‘8pm’ retrieves dinner-related ones. Also, the top locations for ‘10am’ and ‘2pm’ mostly fall in working areas, while the ones for ‘8pm’ distribute more in residential areas. Those results clearly show that the time factor plays an important role in determining people’s activities, and CrossMap effectively captures such subtle dynamics.

### 6.2.2 *Quantitative Evaluation*

We use the task of activity recovery to quantitatively evaluate different urban activity models. Recall that a record $r$ reflects the user’s activity with three attributes: a location $l_r$, a timestamp $t_r$, and a text message $m_r$. For each of the three attributes, say $l_r$, we mark it off and mix it with $M$ randomly chosen negative locations. With the observed timestamp $t_r$ and message $m_r$, the task aims at pinpointing the ground-truth location by ranking all the candidates. Intuitively, the better an activity model captures the cross-modal correlations, the more likely it ranks the ground truth location to top positions. Hence, we use the mean reciprocal rank (MRR) to quantify the performance of a model. Given a set $Q$ of queries, the MRR is defined as: $\text{MRR} = (\sum_{i=1}^{|Q|} 1/\text{rank}_i)/|Q|$, where $\text{rank}_i$ is the ranking of the ground truth for the $i$-th query.

We describe the ranking procedures of different methods as follows. Consider location prediction as an example. For CrossMap, we compute the average cosine similarity of each candidate location to the observed timestamp and keywords, and rank them in the descending order of the similarity; for LGTA and MGTM, we compute the likelihood of observing each candidate location given the keywords, and rank the candidates by likelihood; for TF-IDF, we rank the candidates by computing the tf-idf similarities; for SVD and Tensor, we use the decompositions to reconstruct densified co-occurrence matrices and tensor, and then retrieve the matrix/tensor elements to rank the candidates.

For each data set, we randomly choose 80% data for training, and the remaining 20% for testing the methods, with the number of candidates set as $M = 10$. We repeat such a process for five times and report the average performance in Table 2. Note that, for LGTA and MGTM, since they ignore the time factor in the modeling process, they do not support the time prediction subtask.

**Table 2: Mean reciprocal rank for activity recovery. There are three subtasks: (1) predicting text given location and time; (2) predicting location given text and time; and (3) predicting time given location and text. Recon is short for ReconEmbed, and Graph is short for GraphEmbed.**

| | Text | | Location | | Time | |
|---|---|---|---|---|---|---|
| Method | Tweet | 4SQ | Tweet | 4SQ | Tweet | 4SQ |
| LGTA | 0.376 | 0.6107 | 0.3792 | 0.6083 | - | - |
| MGTM | 0.3874 | 0.5974 | 0.4474 | 0.5753 | - | - |
| TF-IDF | 0.62 | 0.8505 | 0.4298 | 0.7097 | 0.3197 | 0.3431 |
| SVD | 0.4475 | 0.7137 | 0.3953 | 0.646 | 0.3256 | 0.3187 |
| Tensor | 0.4382 | 0.6826 | 0.3871 | 0.6251 | 0.3179 | 0.2983 |
| Recon | 0.6877 | 0.9219 | 0.6526 | 0.9044 | 0.3582 | 0.3612 |
| Graph | **0.7011** | **0.9449** | **0.6758** | **0.9168** | **0.3895** | **0.3716** |

Among all the methods, CrossMap always achieves the best performance across different subtasks on both data sets. Take text prediction as an example, CrossMap outperforms geographical topic modeling methods (LGTA and MGTM) by as much as 86% on Tweet, and 55% on 4SQ. The reason is two-fold: (1) Neither LGTA nor MGTM considers the time factor in the modeling process, and thus fails to leverage the time information for prediction; and (2) Instead of imposing distributional assumptions, CrossMap directly maps different data types into a common space, which captures their correlations more directly and accurately. CrossMap outperforms TF-IDF, SVD, and Tensor by large margins as well. Compared with TF-IDF, the major advantage of CrossMap is that it does not consider spatial, temporal, textual units as independent items, but captures their correlations by modeling both the co-occurrence and neighborhood relationships. Interestingly, SVD and Tensor do not perform as well as the simple TF-IDF method. Our explorations into the results demonstrate that, SVD and Tensor can effectively recover the co-occurrence matrices and tensor by filling in the missing values. However, the raw co-occurence is a less effective relatedness measure for activity recovery compared with the tf-idf measure.

Among the three subtasks, all methods perform the worst for time prediction. It is because the occurring time of an activity could be ambiguous in practice. For instance, we have examined an instance on which CrossMap did not perform well. The given text message is "Girls' shopping day! @ Sephora". Our top predicted time slots are around 3pm, but the ground truth is 9am. Clearly, the predictions make sense but just do not match the ground truth, as the temporal variation of the shopping activity is large. Meanwhile, all the methods perform considerably better on 4SQ than Tweet because Tweet is more noisy than 4SQ. We found that Tweet includes a considerable number of babbling tweets that are meaningless, as well as noisy tweets where the discussed activity and the referred location/time do not match.

On both data sets, GraphEmbed performs slightly better than ReconEmbed. The major advantage of GraphEmbed over ReconEmbed is that it captures not only the direct co-occurrence interactions between the units, but also their implicit interactions by modeling the conditional distributions of graph nodes.

### 6.2.3 *Efficiency.*

Figure 4 shows the running time of all the methods on Tweet. The core algorithms are implemented in C++ and the experiments are conducted on a machine with Intel Xeon E5-2680 2.80GHz using 10 threads. One can observe that the two variations of CrossMap both have excellent efficiency, while GraphEmbed is slightly slower than ReconEmbed.

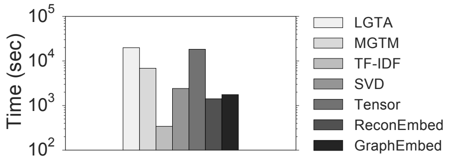
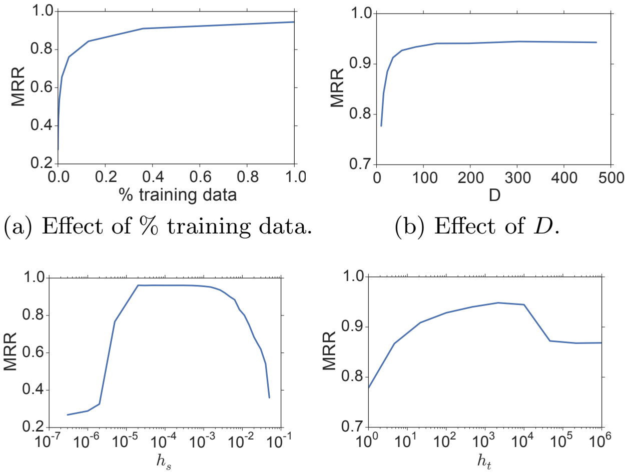


**Figure 4: Running time on Tweet.**

### 6.2.4 *Parameter Study.*

We choose GraphEmbed as a representative to study the effects of different parameters on CrossMap. Below, we report the location prediction MRRs on 4SQ as different parameters vary. Figure 5(a) shows the MRR as the training data percentage increases. In specific, with the test data set fixed, we choose subsets of the training data to learn CrossMap and measure its performance. We can see the MRR consistently increases as CrossMap sees more training checkins. Such a phenomenon shows that a sheer amount of GTSM data is indeed useful for learning quality activity models. Figure 5(b) reports the effect of the latent dimension $D$. As shown, the MRR keeps increasing as $D$ varies from 10 to 450. This phenomenon is expected. With enough training data, a larger $D$ leads to a more complex model that can capture the latent semantics more accurately. Figure 5(c) and 5(d) show the effects of the kernel bandwidths $h_s$ and $h_t$, respectively. We find that the kernel bandwidth should not be too small or too large:

1) a too small bandwidth fails to group correlated locations (timestamps), making the embedding module suffer from severe data sparsity; and 2) a too large bandwidth mistakenly groups uncorrelated locations (timestamps), and impairs the discriminative power of location (time). In the extreme case, $h_t = 10^6$ groups all timestamps into one hotspot, rendering CrossMap not leveraging the time information at all and causing 8.5% performance drop. Similarly, under the extreme case of $h_l = 0.05$, CrossMap cannot leverage the spatial information, and suffers from even larger performance drop. It is worth noting that CrossMap is quite robust as long as $h_l$ and $h_s$ are set to reasonable ranges (*e.g.*, note the plateau in Figure 5(c)).



(a) Effect of % training data. (b) Effect of $D$.

(c) Effect of $h_s$. (d) Effect of $h_t$.

**Figure 5: MRR versus different parameters on 4SQ.**

### 6.2.5 *Downstream Application.*

We choose activity classification as an example application to demonstrate the usefulness of the cross-modal embeddings learnt by CrossMap. In 4SQ, each checkin belongs to one of the following nine categories: Food, Shop & Service, Travel & Transport, College & University, Nightlife Spot, Residence, Outdoors & Recreation, Arts & Entertainment, Professional & Other Places. We use those categories as activity labels, and learn classifiers to predict the label for any given check-in. After random shuffling, we use 80% checkins for training, and the rest 20% for testing. Given a checkin $r$, any of the methods introduced in Section 6.1 (including CrossMap) can obtain three vector representations for the location, time, and text message; we concatenate the three vectors as the feature vector of the checkin $r$.

After feature transformation, we train a multi-class logistic regression for each method. We measure the classification performance of each method with the Micro-F1 metric and report the results in Figure 6. As shown, ReconEmbed and GraphEmbed outperform other methods significantly. Even with a simple linear classification model, the absolute F1 score can reach as high as 0.843. Such results show that the embeddings obtained by CrossMap can well distinguish the semantics of GTSM records. Figure 7 further verifies this fact. Therein, we choose three categories and use t-SNE [23] to visualize the feature vectors. One can observe that the learnt embeddings of CrossMap result in much clearer inter-class boundaries compared to LGTA.

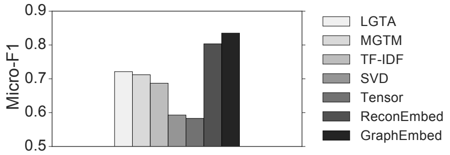
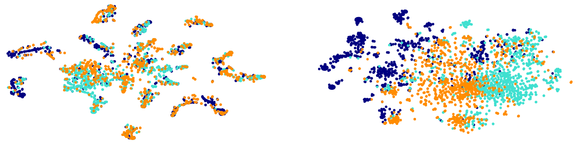


**Figure 6: Activity classification performance.**



(a) LGTA (b) CrossMap

**Figure 7: Visualizing the feature vectors generated by LGTA and CrossMap for three activity categories: 'Food' (cyan), 'Travel & Transport' (blue), and 'Residence' (orange). The feature of each 4SQ checkin is mapped to a 2D point with t-SNE [23].**

## 7. CONCLUSION

We have studied the problem of using geo-tagged social media to model people's activities in the urban space. Towards this end, we proposed CrossMap, an urban activity model based on cross-modal representation learning. It first detects representative spatial and temporal hotspots from massive GTMS data, addressing spatiotemporal variations and reducing data sparsity. It then jointly maps the spatial, temporal, and textual units into the common space, with their cross-modal correlations well preserved. The learnt embeddings of CrossMap can not only well recover urban activities, but also greatly benefit downstream applications like activity classification. Furthermore, CrossMap can easily process millions of GTSM records, making it suitable for handling large-scale GTSM data. As future work, it is interesting use CrossMap for other downstream applications such as tour recommendation and anomaly detection.

**Acknowledgements:** We would like to thank Shi Zhi, Yucheng Chen, and Meng Qu for constructive discussions. We also thank the reviewers for their insightful comments. This work was sponsored in part by the U.S. Army Research Lab. under Cooperative Agreement No. W911NF-09-2-0053 (NSCTA), National Science Foundation IIS-1017362, IIS-1320617, and IIS-1354329, HDTRA1-10-1-0120, and Grant 1U54GM114838 awarded by NIGMS through funds provided by the trans-NIH Big Data to Knowledge (BD2K) initiative (www.bd2k.nih.gov), and MIAS, a DHS-IDS Center for Multimodal Information Access and Synthesis at UIUC. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as representing the official policies of the U.S. Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

# 8. REFERENCES

[1] H. Abdelhaq, C. Sengstock, and M. Gertz. Eventweet: Online localized event detection from twitter. *PVLDB*, 6(12):1326–1329, 2013.

[2] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(1):993–1022, 2003.

[3] M. A. Carreira-Perpinan. Acceleration strategies for gaussian mean-shift image segmentation. In *CVPR*, pages 1160–1167, 2006.

[4] L. Chen and A. Roy. Event detection from flickr data through wavelet-based spatial analysis. In *CIKM*, pages 523–532, 2009.

[5] Z. Cheng, J. Caverlee, K. Lee, and D. Z. Sui. Exploring millions of footprints in location sharing services. In *ICWSM*, pages 81–88, 2011.

[6] E. Cho, S. A. Myers, and J. Leskovec. Friendship and mobility: user movement in location-based social networks. In *KDD*, pages 1082–1090, 2011.

[7] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(5):603–619, 2002.

[8] J. Cranshaw, R. Schwartz, J. I. Hong, and N. Sadeh. The livehoods project: Utilizing social media to understand the dynamics of a city. In *ICWSM*, pages 58 – 65, 2012.

[9] M. Daggitt, A. Noulas, B. Shaw, and C. Mascolo. Tracking urban activity growth globally with big location data. *CoRR*, abs/1512.05819, 2015.

[10] W. Feng, C. Zhang, W. Zhang, J. Han, J. Wang, C. Aggarwal, and J. Huang. Streamcube: hierarchical spatio-temporal hashtag clustering for event exploration over the twitter stream. In *ICDE*, pages 1561–1572, 2015.

[11] V. Frias-Martinez, V. Soto, H. Hohwald, and E. Frias-Martinez. Characterizing urban landscapes using geolocated tweets. In *SocialCom/PASSAT*, pages 239–248, 2012.

[12] H. Gui, J. Liu, F. Tao, M. Jiang, B. Norick, and J. Han. Large-scale embedding learning in heterogeneous event data. In *ICDM*, 2016.

[13] R. A. Harshman. Foundations of the PARAFAC procedure: Models and conditions for an “explanatory” multi-modal factor analysis. *UCLA Working Papers in Phonetics*, 16(1):84, 1970.

[14] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR*, pages 50–57, 1999.

[15] L. Hong, A. Ahmed, S. Gurumurthy, A. J. Smola, and K. Tsioutsiouliklis. Discovering geographical topics in the twitter stream. In *WWW*, pages 769–778, 2012.

[16] D. Hristova, M. J. Williams, M. Musolesi, P. Panzarasa, and C. Mascolo. Measuring urban social diversity using interconnected geo-social networks. In *WWW*, pages 21–30, 2016.

[17] R. Jurdak, K. Zhao, J. Liu, M. AbouJaoude, M. Cameron, and D. Newth. Understanding human mobility from twitter. *PloS one*, 10(7):e0131469, 2015.

[18] W. Kang, A. K. H. Tung, W. Chen, X. Li, Q. Song, C. Zhang, F. Zhao, and X. Zhou. Trendspedia: An internet observatory for analyzing and visualizing the evolving web. In *ICDE*, pages 1206–1209, 2014.

[19] C. C. Kling, J. Kunegis, S. Sizov, and S. Staab. Detecting non-gaussian geographical topics in tagged photo collections. In *WSDM*, pages 603–612, 2014.

[20] J. Krumm and E. Horvitz. Eyewitness: Identifying local events via space-time signals in twitter feeds. In *SIGSPATIAL*, 2015.

[21] R. Lee, S. Wakamiya, and K. Sumiya. Discovery of unusual regional social activities using geo-tagged microblogs. *World Wide Web*, 14(4):321–349, 2011.

[22] K. Leetaru, S. Wang, G. Cao, A. Padmanabhan, and E. Shook. Mapping the global twitter heartbeat: The geography of twitter. *First Monday*, 18(5), 2013.

[23] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(85):2579–2605, 2008.

[24] Q. Mei, C. Liu, H. Su, and C. Zhai. A probabilistic approach to spatiotemporal theme pattern mining on weblogs. In *WWW*, pages 533–542, 2006.

[25] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119, 2013.

[26] A. Noulas, C. Mascolo, and E. Frias-Martinez. Exploiting foursquare and cellular data to infer user activity in urban environments. In *MDM*, pages 167–176, 2013.

[27] A. Noulas, S. Scellato, C. Mascolo, and M. Pontil. An empirical study of geographic user activity patterns in foursquare. In *ICWSM*, pages 570–573, 2011.

[28] A. Noulas, S. Scellato, C. Mascolo, and M. Pontil. Exploiting semantic annotations for clustering geographic areas and users in location-based social networks. In *ICWSM*, 2011.

[29] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: Online learning of social representations. In *KDD*, pages 701–710, 2014.

[30] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *WWW*, pages 851–860, 2010.

[31] S. Sizov. Geofolk: latent spatial semantics in web 2.0 social media. In *WSDM*, pages 281–290, 2010.

[32] J. Tang, M. Qu, and Q. Mei. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *KDD*, pages 1165–1174, 2015.

[33] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei. Line: Large-scale information network embedding. In *WWW*, pages 1067–1077, 2015.

[34] C. Wang, J. Wang, X. Xie, and W.-Y. Ma. Mining geographic knowledge using location aware topic model. In *GIR*, pages 65–70, 2007.

[35] Z. Yin, L. Cao, J. Han, C. Zhai, and T. S. Huang. Geographical topic discovery and comparison. In *WWW*, pages 247–256, 2011.

[36] Q. Yuan, G. Cong, Z. Ma, A. Sun, and N. M. Thalmann. Who, where, when and what: discover spatio-temporal topics for twitter users. In *KDD*, pages 605–613, 2013.

[37] Q. Yuan, W. Zhang, C. Zhang, X. Geng, G. Cong, and J. Han. Pred: Periodic region detection for mobility modeling of social media users. In *WSDM*, 2017.

[38] C. Zhang, J. Han, L. Shou, J. Lu, and T. F. L. Porta. Splitter: Mining fine-grained sequential patterns in semantic trajectories. *PVLDB*, 7(9):769–780, 2014.

[39] C. Zhang, K. Zhang, Q. Yuan, L. Zhang, T. Hanratty, and J. Han. Gmove: Group-level mobility modeling using geo-tagged social media. In *KDD*, pages 1305–1314, 2016.

[40] C. Zhang, G. Zhou, Q. Yuan, H. Zhuang, Y. Zheng, L. Kaplan, S. Wang, and J. Han. Geoburst: Real-time local event detection in geo-tagged tweet streams. In *SIGIR*, pages 513–522, 2016.

[41] K. Zhang, Q. Jin, K. Pelechrinis, and T. Lappas. On the importance of temporal dynamics in modeling urban activity. In *UrbComp*, 2013.

[42] Y. Zheng. Methodologies for cross-domain data fusion: An overview. *IEEE transactions on big data*, 1(1):16–34, 2015.

[43] Y. Zheng, L. Capra, O. Wolfson, and H. Yang. Urban computing: Concepts, methodologies, and applications. *ACM TIST*, 5(3):38:1–38:55, 2014.