ReAct: Online Multimodal Embedding for Recency-Aware Spatiotemporal Activity Modeling

Chao Zhang[1], Keyang Zhang[1], Quan Yuan[1], Fangbo Tao[1], Luming Zhang[2], Tim Hanratty[3], and Jiawei Han[1]
[1]Computer Science Department, University of Illinois at Urbana-Champaign, Urbana, IL, USA
[2]Department of CSIE, Hefei University of Technology, China
[3]U.S. Army Research Laboratory, Adelphi, MD, USA
[1]{czhang82, kzhang53, qyuan, ftao2, hanj}@illinois.edu
[2]zglumg@gmail.com [3]timothy.p.hanratty@mail.mil

ABSTRACT

Spatiotemporal activity modeling is an important task for applications like tour recommendation and place search. The recently developed geographical topic models have demonstrated compelling results in using geo-tagged social media (GTSM) for spatiotemporal activity modeling. Nevertheless, they all operate in batch and cannot dynamically accommodate the latest information in the GTSM stream to reveal up-to-date spatiotemporal activities. We propose ReAct, a method that processes continuous GTSM streams and obtains *recency-aware* spatiotemporal activity models on the fly. Distinguished from existing topic-based methods, ReAct embeds all the regions, hours, and keywords into the same latent space to capture their correlations. To generate high-quality embeddings, it adopts a novel semi-supervised multimodal embedding paradigm that leverages the activity category information to guide the embedding process. Furthermore, as new records arrive continuously, it employs strategies to effectively incorporate the new information while preserving the knowledge encoded in previous embeddings. Our experiments on the geo-tagged tweet streams in two major cities have shown that ReAct significantly outperforms existing methods for location and activity retrieval tasks.

ACM Reference format:
Chao Zhang[1], Keyang Zhang[1], Quan Yuan[1], Fangbo Tao[1], Luming Zhang[2], Tim Hanratty[3], and Jiawei Han[1] . 2017. ReAct: Online Multimodal Embedding for Recency-Aware Spatiotemporal Activity Modeling. In *Proceedings of SIGIR '17, August 07-11, 2017, Shinjuku, Tokyo, Japan, ,* 10 pages.
DOI: 10.1145/3077136.3080814

1 INTRODUCTION

Today's big cities pose big challenges when people try to find desired resources like places and activities. Consider a tourist in a metropolis like New York City. What are the popular activities in her neighborhood at the time being? Which regions should she stay if she cares much about quality food and accessible transportation around her hotel? With hundreds of thousands places in the city and the complex spatiotemporal dynamics, answering such questions is challenging not only for tourists, but even for local citizens who have lived in New York City for many years.

Years ago, developing data-driven approaches to model people's activities in the physical world was almost impossible due to the lack of reliable sources. Traditional approaches have resorted to human surveys to unveil land uses [31, 32], yet such knowledge is still too limited to support downstream activity retrieval applications. The recent proliferation of geo-tagged social media (abbreviated as GTSM onwards), however, sheds light on this problem. As every GTSM record (*e.g.*, geo-tagged tweet, Foursquare checkin) contains a location, a timestamp, and a text message, the massive GTSM data generated from various platforms serve as a result of crowd sensing — they contain rich information about spatiotemporal dynamics as people probe different regions as human sensors. With tens of millions of GTSM records being collected every day, it becomes possible to perform data-driven spatiotemporal activity modeling to address people's information needs.

The recent studies [13, 16, 24, 28, 34, 35] have demonstrated the potential of GTSM for modeling spatiotemporal activities. By incorporating the spatial information into classical topic models like LDA [5], their proposed geographical topic models can detect the activities in different regions, and vice versa. While compelling results have been obtained by those geographical topic models, they all assume the GTSM data is given as static. In practice, however, people's activities in the physical world are dynamically evolving in nature, and the end users often demand the *latest* knowledge. For example, consider the user who queries for popular activities around her. What she needs are the activities popular in recent months or even weeks, rather than the ones that were popular years ago but no longer popular now. As another example, when users seek for places for outdoor activities in summer time, beach-area attractions could be good answers to such queries. But when it comes to the fall time, beach-related activities become too chilly for most people, while other places like parks with hiking trails are more appropriate. Such information is readily available in GTSM streams. Unfortunately, existing geographical topic models all operate in batch and cannot capture the latest information about spatiotemporal activities.

In this paper, we learn *recency-aware* spatiotemporal activity models from dynamic GTSM streams. This task is challenging on account of the following issues. First, *integrating diverse data types is nontrivial.* GTSM involves three different data types: location, time, and text. Considering the totally different representations

SIGIR '17, August 07-11, 2017, Shinjuku, Tokyo, Japan

DOI: 10.1145/3077136.3080814

of these data types and their complicated inter-type interactions, effectively fusing them into a unified model is a challenge. Second, *it is hard to capture activity semantics from short text.* In GTSM, the semantics of people's activities are expressed through short text messages instead of long documents. To build high-quality spatiotemporal activity models, it is important yet difficult to capture the semantics of such short text messages. Third, *recency-aware processing of massive streaming data is required.* We have to handle the online GTSM stream to build recency-aware spatiotemporal activity models. Nevertheless, it is challenging to effectively accommodate the new records without overfitting them. In addition, it is key to build an update-efficient model to process massive GTSM streams.

We propose a recency-aware spatiotemporal activity model named ReAct. The main technical contributions of ReAct are highlighted as follows:

(1) In ReAct, we directly embed all the regions, hours, and keywords into the same latent space to preserve their inter-type interactions. If two elements are highly correlated (*e.g.*, the JFK airport region and the keyword 'flight'), their representations in the latent space tend be close. Compared with geographical topic modeling methods, the multimodal embedding does not impose any distributional assumptions on people's activities, and incurs much lower computational cost in the learning process.
(2) We employ a novel semi-supervised paradigm in ReAct to generate high-quality embeddings. In a considerable number of records, the users explicitly specify the point-of-interest (POI) to indicate their activity categories (*e.g.*, outdoor, shop). The category information can serve as clean and well-structured knowledge, which allows us to better separate the elements with different semantics in the latent space. Hence, we adopt a semi-supervised paradigm that leverages the clean category information to guide the embedding process and generates better-quality embeddings.
(3) As new GTSM records arrive continuously, we design two strategies to update the embeddings and obtain recency-aware spatiotemporal activity models. The first imposes life-decaying weights on the records such that recent records are emphasized. The second treats previous embeddings as prior knowledge, and employs a constrained optimization procedure to obtain updated embeddings. With either strategy, ReAct effectively incorporates the information in the new records, while largely preserving the knowledge encoded in the previous embeddings to avoid overfitting.

To the best of our knowledge, ReAct represents the first work that can learn recency-aware spatiotemporal activity models from dynamic GTSM streams on the fly. We have performed extensive experiments on two million-scale geo-tagged tweet data sets collected from Los Angeles and New York City. We find that ReAct can generate spatiotemporal activity models in which the latest patterns of spatiotemporal activities are well captured. Across a variety of activity retrieval tasks, ReAct outperforms state-of-the-art methods significantly.

2 OVERVIEW

Problem Description. Let $\mathcal{R} = \{r_1, r_2, \ldots, r_N, \ldots\}$ be a stream of geo-tagged social media (GTSM) records. Each record $r \in \mathcal{R}$ 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 timestamp; and (3) m_r is a bag of keywords that represent the text message of r.

Our goal is to use the stream $\mathcal{R}$ to learn *recency-aware* spatiotemporal activity models. As there are three different attributes (*i.e.*, location, time, and text) that are intertwined, an effective spatiotemporal activity model should carefully capture their inter-type correlations. Further, to continuously incorporate the *latest* information in $\mathcal{R}$, the model should keep updating as new records arrive. At any time, the learnt model should be able to answer two types of queries: 1) *Spatial query.* Given a spatial location, what are the popular activities around it? and 2) *Textual query.* Given an activity query represented by a bunch of keywords, what are the suitable regions or places for such an activity? For both queries, the users can also specify a timestamp in their query. As such, the model can return time-specific results in response, *e.g.*, returning brunch places for the query 'food' issued in the morning and dining places for the same query issued in the afternoon.

Overview of ReAct. To model people's spatiotemporal activities, ReAct embeds all the spatial, temporal, and textual elements into the same latent space. While the keywords can serve as natural embedding elements for the textual part, it is infeasible to embed every location and timestamp as the space and time are continuous. We thus map each timestamp to some hour in a day and use the mapped hour as a basic temporal element, and hence have 24 possible temporal elements in total. Similarly, we partition the geographical space into equal-size regions and consider each region as a basic spatial element.

Meanwhile, we observe that a considerable number of records explicitly specify the points-of-interests (POIs), *e.g.*, many Foursquare users link their accounts with Twitter to *checkin* at different venues. The category information (*e.g.*, outdoor, shop) of those records, which is clean and well-structured, can serve as useful signals to understand people's activities. We thus regard those categories as labels, and design a semi-supervised paradigm to guide the learning of quality embeddings. At a high level, ReAct aims to learn the embeddings L, T, W, and C where: (1) L is the embeddings for regions; (2) T is the embeddings for hours; (3) W is the embeddings for keywords; and (4) C is the embeddings for categories. Take L as an example. Each element $\mathbf{v}_l \in L$ is a D-dimensional ($D > 0$) vector, which represents the embedding for region l.

Figure 1 shows the framework of ReAct. As shown, it adopts a semi-supervised paradigm for multimodal embedding. 1) For an unlabeled record r_u, we optimize the embeddings L, T, W for the task of recovering the attributes in r_u; and 2) For a labeled record r_l, we optimize the embeddings L, T, W, C for not only attribute recovery but also activity classification. In such a process, the embeddings of the regions, hours, and keywords are shared across the two tasks, while the category embeddings are specific to the activity classification task. In this way, the semantics of activity categories are propagated from labeled records to unlabeled ones,

thereby better separating the elements with different semantics in the latent space.

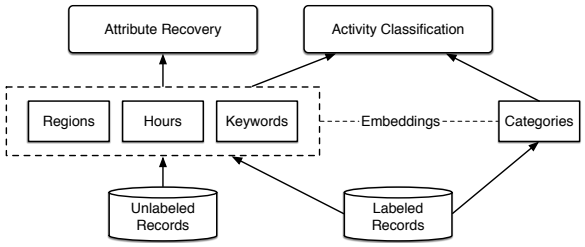


Figure 1: The semi-supervised multimodal embedding framework of ReAct.

The entire learning proceeds in an online manner. When a collection $\mathcal{R}_\Delta$ of new records arrive, our goal is to update the embeddings (L, T, W, C) to accommodate the information contained in $\mathcal{R}_\Delta$. While it is tempting to use $\mathcal{R}_\Delta$ to learn the embeddings from scratch, such an idea not only incurs unnecessary computational overhead, but also leads to overfitting of the new data. To address this issue, we propose an online learning procedure in Section 3, which effectively incorporates the new records while largely preserving the information encoded in the previous embeddings.

3 LEARNING THE REACT MODEL

In this section, we describe the learning procedures for ReAct. As aforementioned, given a collection of records $\mathcal{R}_\Delta$, for any record $r \in \mathcal{R}_\Delta$, either labeled or unlabeled, ReAct first takes the attribute recovery task, and updates the embeddings L, T, and W to correctly recover the attributes of r. If r is labeled, ReAct further leverages the category information as supervision, and updates L, T, W, and C such that r can be classified into the correct category.

The key problem in the above online learning framework is, how to update the embeddings with the goal of effectively incorporating the information in $\mathcal{R}_\Delta$ without overfitting it? We develop two different strategies for this problem: one is life-decaying learning, and the other is constraint-based learning. In what follows, we first describe the details of those two strategies in Section 3.1 and 3.2. Then we analyze their space and time complexities in Section 3.3.

3.1 Life-Decaying Learning

Our first strategy, called life-decaying learning, assigns different weights to the records in the GTSM stream such that more recent records receive higher weights. Specifically, for any record r that has appeared in the stream, we set its weight as:

$$w_r = e^{-\tau a_r},$$

where $\tau > 0$ is a decaying parameter, and a_r is r's age with regard to the current time. The general philosophy of such a weighing scheme is to emphasize the recent records and highlight the up-to-date observations of urban activities. On the other hand, the old records in the stream are not completely ignored, they have smaller weights but are still involved in model training to prevent overfitting.

Practically, it is infeasible to store all the records seen so far on account of the massive size of the GTSM stream. For tackling this issue, we maintain a continuously updating buffer $\mathcal{B}$, as shown in Figure 2. The buffer $\mathcal{B}$ consists of m buckets $B_0, B_1, \ldots, B_{m-1}$, where all the buckets have the same time span ΔT. For each bucket $B_i (0 \le i < m)$, we assign an exponentially decaying weight $e^{-\tau i}$ to it, where the weight represents the percentage of samples that we preserve for the respective time span. In other words, the most recent bucket B_0 holds the complete set of records within its time span, the next bucket B_1 holds $e^{-\tau}$ of the corresponding records, and so on. When a new collection of records $\mathcal{R}_\Delta$ arrive, the buffer $\mathcal{B}$ is updated to accommodate $\mathcal{R}_\Delta$. The new records $\mathcal{R}_\Delta$ are fully stored in the most recent bucket B_0. For each other bucket $B_i (i > 0)$, the records in its predecessor B_{i-1} are downsampled with rate $e^{-\tau}$ and then moved into B_i.

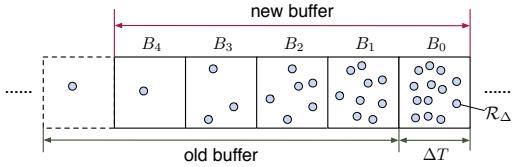


Figure 2: Maintaining a buffer $\mathcal{B}$ for life-decaying learning. For any bucket B_i, $e^{-\tau i}$ of the records falling in B_i's time span are preserved for model updating. When new records arrive, $\mathcal{B}$ is updated based on downsampling and shifting.

Algorithm 1 sketches the learning procedure of ReAct with the life-decaying strategy. As shown, when a collection $\mathcal{R}_\Delta$ of new records arrive, we first shift the records from B_{i-1} to B_i by downsampling (lines 1-2), and store $\mathcal{R}_\Delta$ into B_0 in full (lines 3). Once the buffer $\mathcal{B}$ is updated, we randomly sample records from $\mathcal{B}$ (line 4-7) to update the embeddings. First, for any record r, we consider the attribute recovery task and update the embeddings L, T, and W such that the attributes of r can be correctly recovered. Second, if r is labeled, we further update L, T, W, and C such that r can be classified into the correct activity category. Such a process is repeated over $\mathcal{R}_\cup$ for a number of epochs before the updated embeddings of L, T, W, and C are output. In the following, we detail the updating rules for attribute recovery (line 8) and activity classification (line 9), respectively.

3.1.1 The Attribute Recovery Task. For attribute recovery, we optimize the embeddings L, T, W such that each attribute of a record r can be maximally recovered, assuming the other attributes of r are already observed. Given a record r, for any attribute $i \in r$ with type X (region, hour, or keyword), 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 attributes in r except i, and $s(i, r_{-i})$ is the similarity score between i and r_{-i}.

In the above, the key is how to define $s(i, r_{-i})$. A natural idea is to average the embeddings of the attributes in r_{-i}, and compute $s(i, r_{-i})$ as $s(i, r_{-i}) = \mathbf{v}_i^{\mathrm{T}} \sum_{j \in r_{-i}} \mathbf{v}_j / |r_{-i}|$, where $\mathbf{v}_i$ is the embedding for attribute i. Nevertheless, such a simple definition fails to consider the continuities of the space and time. Take the spatial continuity as an example. Because of spatial locality, two regions

Algorithm 1: Life-decaying learning of ReAct.

Input: The previous embeddings L, T, W, and C.
A buffer of m buckets $\mathcal{B} = \{B_0, B_1, \ldots, B_{m-1}\}$.
A collection $\mathcal{R}_\Delta$ of new records.
Output: The updated buffer $\mathcal{B}$ and embeddings L, T, W, and C.

```
// Downsampling with rate e^{-τ}.
for i from 1 to n do
    B_i ← e^{-τ}-downsampled records from B_{i-1};
B_0 ← R_Δ ;
R_∪ ← B_{m-1} ∪ B_{m-2} ... ∪ B_0;
for epoch from 1 to N do
    for i from 1 to |R_Δ| do
        r ← Randomly sample a record from R_∪;
        // for labeled and unlabeled records
        Update L, T, and W for recovering r's attributes;
        // for only labeled records
        if r is labeled then
            Update L, T, W, and C for classifying r's activity;
Return B, L, T, W, and C;
```

that are close to each other should be considered correlated instead of independent. We thus introduce *spatial smoothing* and *temporal smoothing* to capture the spatiotemporal continuities. With the smoothing technique, ReAct not only maintains local consistency of neighboring regions and hours, but also alleviates data sparsity.

Figure 3 illustrates the spatial and temporal smoothing processes. As shown, for each region l, we introduce a pseudo region $\hat{l}$. The embedding of $\hat{l}$ is the weighted average of the embeddings of l and l's neighboring regions, namely

$$\mathbf{v}_{\hat{l}} = (\mathbf{v}_l + \alpha \sum_{l_n \in \mathcal{N}_l} \mathbf{v}_{l_n})/(1 + \alpha|\mathcal{N}_l|),$$

where $\mathcal{N}_l$ is the set of l's neighboring regions, and α is a constant for spatial smoothing. Similarly, for each hour t, we introduce a pseudo hour $\hat{t}$, whose embedding is the weighted average of the embeddings of t and t's neighboring hours:

$$\mathbf{v}_{\hat{t}} = (\mathbf{v}_t + \beta \sum_{t_n \in \mathcal{N}_t} \mathbf{v}_{t_n})/(1 + \beta|\mathcal{N}_t|),$$

where $\mathcal{N}_t$ is the set of t's neighboring hours, and β is a temporal smoothing constant. In practice, we find that setting $\alpha = 0.1$ and $\beta = 0.1$ usually leads to satisfactory performance of the model.

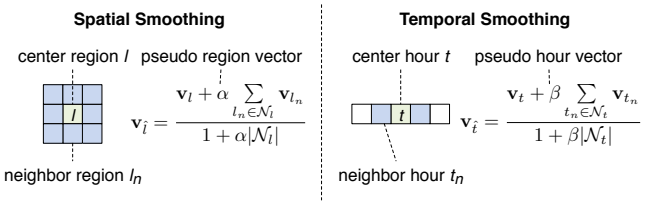


Figure 3: Spatial and temporal smoothing. For each region (hour), we combine it with its neighboring regions (hours) to generate a pseudo region (hour).

In addition to the above pseudo region and hour embeddings, we also introduce pseudo keyword embeddings for notational ease. Given r_{-i}, its pseudo keyword embedding is defined as:

$$\mathbf{v}_{\hat{w}} = \sum_{w \in \mathcal{N}_w} \mathbf{v}_w/|\mathcal{N}_w|,$$

where $\mathcal{N}_w$ is the set of keywords in r_{-i}. With these pseudo embeddings, we define a smoothed version of $s(i, r_{-i})$ as $s(i, r_{-i}) = \mathbf{v}_i^{\mathrm{T}}\mathbf{h}_i$, where

$$\mathbf{h}_i = \begin{cases} (\mathbf{v}_{\hat{l}} + \mathbf{v}_{\hat{t}} + \mathbf{v}_{\hat{w}})/3, & \text{if } i \text{ is a keyword,} \\ (\mathbf{v}_{\hat{t}} + \mathbf{v}_{\hat{w}})/2, & \text{if } i \text{ is a region,} \\ (\mathbf{v}_{\hat{l}} + \mathbf{v}_{\hat{w}})/2, & \text{if } i \text{ is an hour.} \end{cases}$$

Finally, the loss function for the attribute recovery task is simply the negative log-likelihood of observing all the attributes of the records in $\mathcal{R}_\cup$:

$$J_{\mathcal{R}_\cup} = -\sum_{r \in \mathcal{R}_\cup} \sum_{i \in r} \log p(i|r_{-i}). \tag{1}$$

To efficiently optimize the above objective, we use stochastic gradient descent (SGD) and negative sampling [22]. At each time, we use SGD to sample a record r and an attribute $i \in r$. With negative sampling, we randomly select K negative attributes that have the same type with i but do not appear in r, then the loss function for the selected samples becomes:

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

where $\sigma(\cdot)$ is the sigmoid function. The updating rules for $\mathbf{v}_i$, $\mathbf{v}_k$ and $\mathbf{h}_i$ can be obtained by taking the derivatives of J_r:

$$\frac{\partial J_r}{\partial \mathbf{v}_i} = (\sigma(s(i, r_{-i})) - 1)\mathbf{h}_i,$$
$$\frac{\partial J_r}{\partial \mathbf{v}_k} = \sigma(s(i, r_{-i}))\mathbf{h}_i,$$
$$\frac{\partial J_r}{\partial \mathbf{h}_i} = (\sigma(s(i, r_{-i})) - 1)\mathbf{v}_i + \sum_{k=1}^{K} \sigma(s(k, r_{-i}))\mathbf{v}_k.$$

For any attribute j in $\mathbf{h}_i$, we have $\partial L/\partial \mathbf{v}_j = \partial L/\partial \mathbf{h}_i \cdot \partial \mathbf{h}_i/\partial \mathbf{v}_j$, as $\mathbf{h}_i$ is linear in j, the term $\partial \mathbf{h}_i/\partial \mathbf{v}_j$ is convenient to calculate.

3.1.2 The Activity Classification Task. The objective of the activity classification task is to learn the embeddings such that the activity categories of those labeled records in $\mathcal{R}_\cup$ can be correctly predicted. Let r be a labeled record with category c. The basic intuition is to make c's embedding close to the constituent attributes in r. Based on this intuition, we model the probability of classifying r into category c as:

$$p(c|r) = \exp(s(c, r))/\sum_{c' \in C} \exp(s(c', r)).$$

For the similarity score $s(c, r)$, we define it in a smoothed way similar to the attribute recovery task. That is, $s(c, r) = \mathbf{v}_c^{\mathrm{T}}\mathbf{h}_r$, where $\mathbf{h}_r = (\mathbf{v}_{\hat{l}} + \mathbf{v}_{\hat{t}} + \mathbf{v}_{\hat{w}})/3$.

The objective function of the activity classification task is then the negative log-likelihood of predicting the activities categories

Algorithm 2: Constraint-based learning of ReAct.

Input: The previous embeddings L, T, W, and C.
A collection $\mathcal{R}_\Delta$ of new records.
Output: The updated embeddings L, T, W, and C.

```
for epoch from 1 to N do
    Randomly shuffle the records in R_Δ;
    foreach r ∈ R_Δ do
        Update L, T, and W for constrained attribute recovery;
        if r is labeled then
            Update L, T, W, and C for constrained activity classification;
Return L, T, W, and C;
```

for the new records in $\mathcal{R}_\cup$:

$$J_{\mathcal{R}_\cup} = -\sum_{r \in \mathcal{R}_\cup} \log p(c|r). \tag{2}$$

We again use SGD and negative sampling to optimize the objective efficiently. In specific, given the labeled record r with the positive category c, we randomly pick a negative category c' satisfying $c' \neq c$. Then the loss function for r in the activity classification task becomes:

$$J_r = -\log \sigma(s(c, r)) - \log \sigma(-s(c', r)).$$

Similar to the derivation in the attribute recovery task, the updating rules of the attributes and categories can be easily obtained by taking the derivatives of J_r and then applying SGD.

3.2 Constraint-Based Learning

The life-decaying strategy relies on the buffer $\mathcal{B}$ to keep old records besides $\mathcal{R}_\Delta$, thereby incorporating the information in $\mathcal{R}_\Delta$ without overfitting. However, maintaining $\mathcal{B}$ could incur additional space and time overhead. To avoid such overhead, we propose our second strategy named constraint-based learning. The key is to to accommodate the new records $\mathcal{R}_\Delta$ by fine-tuning the previous embeddings. During the fine-turning process, we impose the constraint that the updated embeddings do not deviate much from the previous ones. In this way, ReAct generates embeddings that are optimized for $\mathcal{R}_\Delta$ while respecting the prior knowledge encoded in previous embeddings. Algorithm 2 sketches the constraint-based learning procedure of ReAct. As shown, when a collection $\mathcal{R}_\Delta$ of new records arrive, we directly use them to update the embeddings for a number of epochs, where the updating for both attribute recovery and activity classification is performed under constraints.

Let us first examine the constraint-based attribute recovery task. Given the new records $\mathcal{R}_\Delta$ and their attributes, our goal is still to recover the attributes of $\mathcal{R}_\Delta$, but now we add a regularization term in the objective to ensure the result embeddings can retain the previous embeddings. In formal, we design the objective function for attribute recovery as:

$$J_{\mathcal{R}_\Delta} = -\sum_{r \in \mathcal{R}_\Delta} \sum_{i \in r} \log p(i|r_{-i}) + \lambda \sum_{i \in L, T, W, C} \|\mathbf{v}_i - \mathbf{v}'_i\|^2,$$

where $\mathbf{v}_i$ is the updated embedding of attribute i, and $\mathbf{v}'_i$ is i's previous embedding learnt before the arrival of $\mathcal{R}_\Delta$. In the above objective function, it is important to note the regularization term $\sum_{i \in L, T, W, C} \|\mathbf{v}_i - \mathbf{v}'_i\|^2$. It prevents the updated embeddings from deviating drastically from the previous embeddings. The value of λ ($\lambda \geq 0$) plays an important role in controlling the regularization strength. When $\lambda = 0$, the embeddings are purely optimized for fitting $\mathcal{R}_\Delta$; when $\lambda = \infty$, the learning process completely ignore the new records and all the embeddings remain unchanged.

We still combine SGD and negative sampling to optimize the above objective function. Consider a record r and an attribute $i \in r$. With negative sampling, we randomly select a set of K negative attributes N_i^-, then the objective for the selected samples is:

$$J_r = -\log \sigma(s(i, r_{-i})) - \sum_{k \in N_i^-} \log \sigma(-s(k, r_{-i})) + \lambda \sum_{i \in \{r\} \cup N_i^-} \|\mathbf{v}_i - \mathbf{v}'_i\|^2.$$

The updating rules for different attributes can be easily obtained by taking the derivatives of J_r. Taking attribute i as an example, the corresponding derivative and updating rule are given by

$$\frac{\partial J_r}{\partial \mathbf{v}_i} = (\sigma(s(i, r_{-i})) - 1)\mathbf{h}_i + 2\lambda(\mathbf{v}_i - \mathbf{v}'_i),$$
$$\mathbf{v}_i \leftarrow \mathbf{v}_i + \eta(1 - \sigma(s(i, r_{-i})))\mathbf{h}_i - 2\eta\lambda(\mathbf{v}_i - \mathbf{v}'_i),$$

where η is the learning rate for SGD.

By examining the updating rule for i, we can see the constraint-based strategy enjoys two attractive properties: 1) It tries to make i's embedding close to the average embedding (*i.e.*, $\mathbf{h}_i$) of the other attributes in r. Especially when the current embeddings do not produce high similarity score between i and r_i, *i.e.*, $s(i, r_{-i})$ is small, the updating takes an aggressive step to push $\mathbf{v}_i$ close to $\mathbf{h}_i$; and 2) With the term $-2\eta\lambda(\mathbf{v}_i - \mathbf{v}'_i)$, the learnt embeddings are constrained to preserve the information encoded in the previous embeddings. In specific, if the learnt embedding $\mathbf{v}_i$ deviates from the previous embedding $\mathbf{v}'_i$ too much, the updating rule would subtract the difference to some extent and drag $\mathbf{v}_i$ towards $\mathbf{v}'_i$.

We proceed to examine the activity classification task under the constraint-based strategy. The overall objective is to maximize the log-likelihood of predicting the activities categories for $\mathcal{R}_\Delta$ while minimizing the deviation from the previous embeddings. Using SGD, for any record r with activity category c, we generate a negative category c', and then define the objective as

$$J_r = -\log \sigma(s(c, r)) - \log \sigma(-s(c', r)) + \lambda \sum_{c \in \{c, c'\}} \|\mathbf{v}_c - \mathbf{v}'_c\|^2.$$

Again, the updating rules for the different variables in the above objective can be easily obtained by taking the derivatives of J_r, we omit the details here to save space.

3.3 Complexity Analysis

Space complexity. With either life-decaying learning or constraint-based learning, we need to maintain the embeddings of all the regions, hours, keywords, and categories. Let D be the dimension of the latent space. Then the space cost for maintaining those embeddings is $O(D(|L| + |T| + |W| + |C|))$, where $|L|$, $|T|$, $|W|$, and $|C|$ are the numbers of regions, hours, keywords, and categories, respectively. In addition, both strategies need to keep a collection of training records. For the constraint-based learning, the space cost of this part is $O(|\mathcal{R}_{max}|)$ where $|\mathcal{R}_{max}|$ is the maximum number of new records that arrive at one time. The life-decaying learning

strategy needs to keep the new records as well as some old ones. As it imposes exponentially decaying sampling rates on the buckets, the space cost for maintaining those records is

$$O(|\mathcal{R}_{max}|(1 + e^{-\tau} + \ldots + e^{-(m-1)\tau})) = O(|\mathcal{R}_{max}|\frac{1 - e^{-m\tau}}{1 - e^{-\tau}}).$$

Time complexity. We first analyze the time complexity of the constraint-based learning strategy. Examining Algorithm 2, one can see that the constraint-based strategy needs to go over $\mathcal{R}_\Delta$ for N epochs and process every record in $\mathcal{R}_\Delta$ exactly once in each epoch. Hence, the time complexity is $O(NDM^2|\mathcal{R}_{max}|)$, where M is the maximum number of attributes in any record. Since N and D are fixed beforehand, and M is usually sufficiently small, ReAct scales roughly linearly with $\mathcal{R}_\Delta$. Similarly, the time complexity of the life-decaying strategy is derived as $O(NDM^2|\mathcal{R}_{max}| + |\mathcal{R}_\cup|)$, where $|\mathcal{R}_\cup| = |\mathcal{R}_{max}|(1 - e^{-m\tau})/(1 - e^{-\tau})$.

4 EXPERIMENTS

4.1 Experimental Setup

4.1.1 Data Sets. Our experiments are based on two real-life geo-tagged tweet data sets: LA and NY. The LA data set contains ∼1.10 million geo-tagged tweets published in Los Angeles. We crawled the LA data set by monitoring the Twitter Streaming API[1] during 2014.08.01 – 2014.11.30 and continuously gathering the geo-tagged tweets in the bounding box of LA. In addition, we crawled all the POIs in LA through Foursquare's public API[2]. We are able to link ∼0.11 million of the crawled tweets to the POI database and assign them to one of the following categories: Food, Shop & Service, Travel & Transport, College & University, Nightlife Spot, Residence, Outdoors & Recreation, Arts & Entertainment, Professional & Other Places. We preprocessed the raw data as follows. For the text part, we removed user mentions, URLs, stopwords, and the words that appear less than 100 times in the corpus. For the space and time, we partitioned the LA area into small grids with size 300m*300m, and broke the one-day period into 24 one-hour windows. The NY data set is also collected from Twitter and then linked with Foursquare. It consists of ∼1.20 million geo-tagged tweets in New York City during 2014.08.01 - 2014.11.30, and we are able to link ∼0.10 million of them with Foursquare POIs. The preprocessing steps are the same as LA.

4.1.2 Baselines. We compare our proposed ReAct model with the following baseline methods:

- LGTA [34] 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 keywords.
- MGTM [16] 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, and does not require a pre-specified number of topics.
- Tensor [9] builds a 4-D tensor to encode the co-occurrences among location, time, text, and category. It then factorizes the tensor to obtain low-dimensional representations of all the elements.
- SVD first constructs the co-occurrence matrices between each pair of location, time, text, and category, and then performs Singular Value Decomposition on the matrices.
- TF-IDF constructs the co-occurrence matrices between each pair of location, time, text, and category. It then computes the tf-idf weight for each entry in the matrix by treating rows as documents and columns as words.

Similar to our ReAct method, Tensor, SVD, and TF-IDF also rely on space and time partitioning to obtain regions and time periods. We use the same partitioning granularity for those methods to ensure fair comparison. For the two different learning strategies of ReAct, we refer to the life-decaying one as RA-Decay, and the constraint-based one as RA-Cons. Besides them, we also implement two weakened variants of ReAct to validate the effectiveness of recency-ware learning and the semi-supervised paradigm: 1) RA-Base is a variant of ReAct that neither adopts recency-aware training, nor leverages the category information as supervision; and 2) RA-Semi is another variant, which uses the category information as supervision, but does not perform recency-aware training. Note that both RA-Base and RA-Semi perform training in batch instead of online, treating all the seen instances equally.

4.1.3 Parameter Settings. There are five common parameters for the life-decaying and constraint-based strategies: 1) the latent embedding dimension D; 2) the number of epochs N; 3) the SGD learning rate η; 4) the spatial smoothing constant α; and 5) the temporal smoothing constant β. By default, we set $D = 300$, $N = 50$, $\eta = 0.01$, and $\alpha = \beta = 0.1$. Meanwhile, the life-decaying strategy has its specific parameters, the decaying rate τ and the number of buckets m; and the constraint-based strategy also has its own parameter, the regularization strength λ. We set their default values to $\tau = 0.01$, $m = 500$, and $\lambda = 0.3$.

In LGTA, there are two major parameters, the number of regions R, and the number of latent topics Z. After careful tuning, we set $R = 300$ and $Z = 10$. MGTM is a non-parametric method that involves several hyper-parameters. We set the hyper-parameters following the original paper [16]. For Tensor and SVD, we set the latent dimension as $D = 300$ to compare with ReAct fairly.

4.1.4 Evaluation Tasks and Metrics. We use two types of activity retrieval tasks to evaluate the effectiveness of different models. The first is to *retrieve locations for a given activity.* Specifically, recall that each GTSM record reflects a user's activity with three attributes: a location l_r, a timestamp t_r, and a bag of keywords m_r. In the location retrieval task, the input is the timestamp t_r and the keywords m_r, and the goal is to accurately pinpoint the ground-truth location from a pool of candidates. We retrieve the location at two different granularities: 1) *coarse-grained region retrieval* is to retrieve the ground-truth region that r falls in; and 2) *fine-grained POI retrieval* is to retrieve the ground-truth POI that r corresponds to. Note that fine-grained POI retrieval is only evaluated on the tweets that have been linked with Foursquare. The second task is to *retrieve activities for a given location.* In this task, the input is the timestamp t_r and the location l_r, and the goal is to pinpoint the ground-truth activities at two different granularities: 1) *coarse-grained category retrieval* is to retrieve the ground-truth activity category of r. Again, such a coarse-grained activity retrieval is performed only on the

[1] https://dev.twitter.com/streaming/overview
[2] https://developer.foursquare.com/

tweets that have been linked with Foursquare; and 2) *fine-grained keyword retrieval* is to retrieve the ground-truth message m_r from a candidate pool of messages.

To summarize, we study four sub-tasks in total: 1) region retrieval; 2) POI retrieval; 3) category retrieval; and 4) keyword retrieval. For each retrieval task, we generate a candidate pool by mixing the ground truth with a set of M negative samples. Take region retrieval as an example. Given the ground-truth region l_r, we mix l_r with M randomly chosen regions. Then we try to pinpoint the ground truth from the size-$(M+1)$ candidate pool by ranking all the candidates. Intuitively, the better a model captures the patterns underlying people's activities, the more likely it ranks the ground truth to top positions. We thus use Mean Reciprocal Rank (MRR) to quantify the effectiveness 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 rank_i is the ranking of the ground truth for the i-th query.

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

On each data set, we randomly generate 20 one-hour query windows in 2014.08.01 – 2014.11.30, and use all the tweets in those windows as test instances. Within those query windows, there are ~11 thousand test tweets in LA, and ~12 thousand test tweets in NY. We set the number of candidates to $M = 10$ for different retrieval tasks except for category retrieval (there are only nine categories in total). For each query window, we use the tweets that have arrived before the window start to train different models. All the methods except for ReAct work in a batch manner, hence we have to train them repeatedly for 20 times, with different training data for different query windows. As ReAct works online, we use a one-hour window to hold new records and update ReAct hourly. We ensure ReAct uses the same training data as other methods during the one-pass training process.

4.2 Effectiveness Comparison

4.2.1 Qualitative Results. We first use several examples to examine whether ReAct can capture the dynamic evolutions of spatiotemporal activities, as well as how well it models the correlations between location, time, and text. Specifically, we perform one-pass training of RA-Decay on LA and NY, and launch a bunch of queries at different stages. For each query, we retrieve the top-10 most similar elements with different types from the entire search space.

Textual Queries. Figure 4(a) shows the results when we query with the keyword 'beach' on LA. We find the results quite meaningful: the top regions fall along the coastline; the top POIs are famous beach attractions; and the top keywords well reflect people's activities on the beach (*e.g.*, 'sand', 'boardwalk'). Figure 4(b) and 4(c) show the results for the query 'outdoor + weekend' issued on NY for two different days. Interestingly, the results obtained for the two days both relate to 'outdoor', but exhibit clear evolutions. While the results for 2014.08.30 contain many swimming-related activities, those for 2014.10.30 are mostly fitness venues. Based on such phenomena, one can clearly see that ReAct captures not only the inter-type correlations between different attributes but also the temporal evolutions underlying spatiotemporal activities.

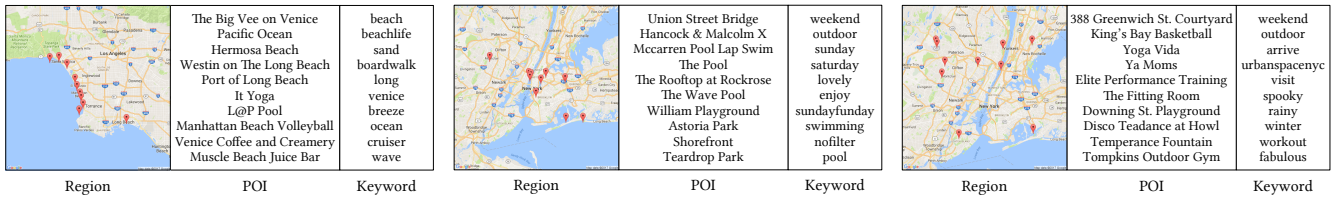
Spatial Queries. Figure 4(d) and 4(e) illustrate the evolutions of two spatial queries: 1) the Metlife Stadium; and 2) the Universal Studio. Again, we can see the results well match the query location and meanwhile reflect urban dynamics clearly. For the Metlife Stadium query, the top keywords evolve from concert-related ones to football-related ones. It is because the NFL season opens in early September, and people start visiting the stadium to watch the games of the Giants and the Jets. For the Universal Studio query, we intentionally include Halloween and Thanksgiving in the query days. In such a setting, we find the latter two lists contain holiday-specific keywords, verifying the capability of ReAct for capturing the most recent activity patterns.

Table 1: The MRRs of different methods for location retrieval. For each test tweet, we assume its timestamp and keywords are observed, and perform location retrieval at two granularities: 1) *region retrieval* retrieves the ground-truth region; and 2) *POI retrieval* retrieves the ground-truth POI (for Foursquare-linked tweets).

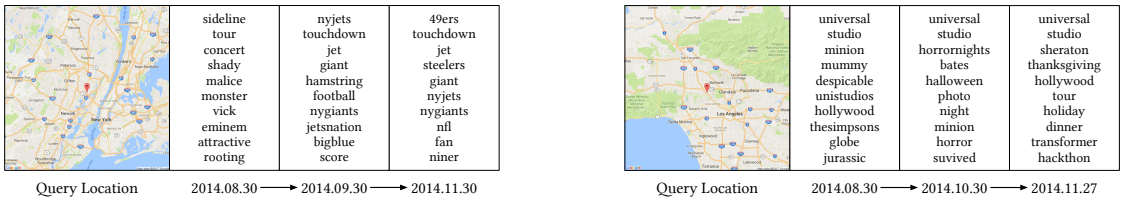
Method	Region Retrieval		POI Retrieval	
	LA	NY	LA	NY
LGTA	0.3583	0.3544	0.5889	0.5674
MGTM	0.4007	0.391	0.5811	0.553
Tensor	0.3592	0.3641	0.6672	0.7399
SVD	0.3699	0.3604	0.6705	0.7443
TF-IDF	0.4114	0.4605	0.719	0.776
RA-Base	0.5373	0.5597	0.7845	0.8508
RA-Semi	0.5586	0.5632	0.8155	0.8712
RA-Cons	0.5714	0.5864	0.8311	**0.8896**
RA-Decay	**0.5802**	**0.5898**	**0.8473**	0.885

Table 2: The MRRs of different methods for activity retrieval. For each test tweet, we assume its location and timestamp are observed, and retrieve activities at two granularities: 1) *category retrieval* retrieves the ground-truth category (for Foursquare-linked tweets); and 2) *keyword retrieval* retrieves the ground-truth message.

Method	Category Retrieval		Keyword Retrieval	
	LA	NY	LA	NY
LGTA	0.4409	0.4527	0.3392	0.3425
MGTM	0.4587	0.464	0.3501	0.343
Tensor	0.8635	0.7988	0.4004	0.3744
SVD	0.8556	0.7826	0.4098	0.3728
TF-IDF	0.9137	0.8259	0.5236	0.4864
RA-Base	0.6225	0.5874	0.5693	0.5538
RA-Semi	0.9056	0.8993	0.5832	0.5793
RA-Cons	0.92	0.8964	0.6097	0.5887
RA-Decay	**0.9272**	**0.9026**	**0.6174**	**0.5928**



(a) Query = 'beach' + 3pm (2014.08.30@LA) (b) Query = 'outdoor + weekend' (2014.08.30@NY) (c) Query = 'outdoor + weekend' (2014.10.30@NY)



(d) Query = '(40.8128, -74.0764)' (Metlife Stadium@NY) (e) Query = '(34.1381, -118.3534)' (Universal Studio@LA)

Figure 4: Illustrative cases. Figure 4(a), 4(b), and 4(c) are textual queries issued on different days (*i.e.*, the dates in bracket). For each query, we use the trained model on the issuing day to retrieve ten most similar regions (the markers in the map denote the region centers), POIs, and keywords, based on cosine similarities of the embeddings. Figure 4(d) shows two spatial queries at the Metlife Stadium and Universal Studio. For each query, we retrieve ten most similar keywords on different days.

4.2.2 Quantitative Results. Table 1 and 2 report the quantitative results of different methods for location and activity retrievals, respectively. As shown, on all of the four sub-tasks, ReAct and its variants achieve much higher MRRs than the baseline methods. Compared with the two geographical topic models (LGTA and MGTM), ReAct yields as much as 62% performance improvement for location retrieval, and 83% for activity retrieval. There are three factors for explaining the performance gap: (1) Neither LGTA nor MGTM models the time factor, and thus fails to leverage the time information for prediction; (2) ReAct emphasizes recent records to capture up-to-date spatiotemporal activities, while LGTA and MGTM work in batch and treat all training instances equally; and (3) Instead of using generative models, ReAct directly maps different data types into a common space to capture their correlations more directly.

Tensor, SVD, and TF-IDF have better performance than LGTA and MGTM by modeling time and category, yet ReAct still outperforms them by large margins. Interestingly, TF-IDF turns out to be a strong baseline, demonstrating the effectiveness of the tf-idf similarity for the retrieval tasks. SVD and Tensor can effectively recover the co-occurrence matrices and tensor by filling in the missing values. However, the raw co-occurrence seems a less effective relatedness measure for location and activity retrieval.

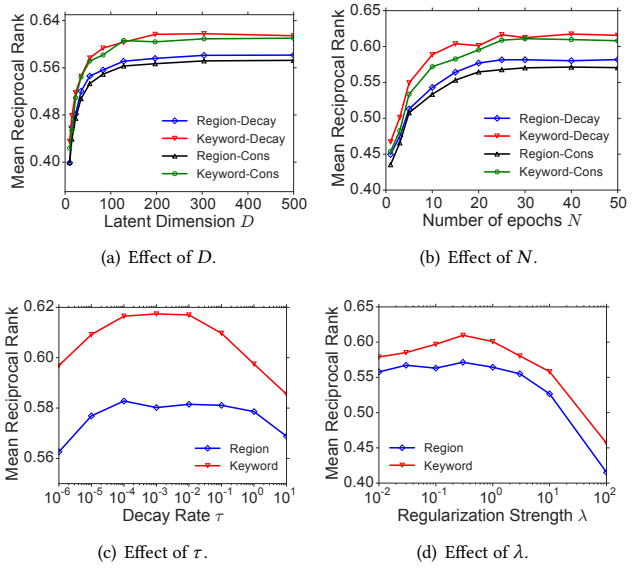
Comparing the variants of ReAct, we see clear performance gaps between RA-Base and RA-Semi, particularly for the category retrieval task. The major difference between RA-Base and RA-Semi is that, RA-Base just treats category descriptions as keywords, while RA-Semi uses activity categories as labels to guide embedding. This phenomenon shows the semi-supervised paradigm indeed helps propagate structured category information into the embedding process to generate high-quality embeddings.

RA-Semi is inferior to RA-Decay and RA-Cons considerably. Although the three variants all use semi-supervised training, RA-Semi treats all the training instances equally whereas the other two work online and emphasize recent instances more. This fact verifies that there are notable evolutions underlying people's activities in the four-month time period, and the recency-aware nature of ReAct effectively captures such evolutions to better suit users' retrieval needs. Finally, examining the performance of RA-Decay and RA-Cons, we find that the life-decaying learning strategy performs slightly better than the constraint-based one in practice, but at the cost of extra space and time overhead.

4.3 Effects of Parameters

Lastly, we study the effects of different parameters on the performance of ReAct. Figure 5(a) and 5(b) show the effects of the latent dimension D and the number of epochs N. Since the trends are very similar for fine-grained and coarse-grained retrieval tasks, we omit the results for POI retrieval and category retrieval for clarity. As shown in Figure 5(a), the MRRs of both methods keep increasing with D and gradually converge. This phenomenon is expected because a larger D leads to a more expressive model that can capture latent semantics more accurately. From Figure 5(b), one can see as N increases, the performance of ReAct also increases first and finally becomes stable: when N is small, the updated embeddings do not incorporate the new information sufficiently; when N is large, both the life-decaying and constraint-based strategies can effectively prevent ReAct from overfitting the new records.

Figure 5(c) and 5(d) depict the effects of τ and λ on the performance of the two learning strategies, respectively. As shown, for life-decaying learning, its performance first increases with τ, then becomes stable, and finally deteriorates. The reason is two-fold: 1) a too small τ makes the buffer contain too many old records in the history, thus diluting the most recent information; and 2) a too large τ leads to a buffer that contains only recent records,



(a) Effect of D. (b) Effect of N.

(c) Effect of τ. (d) Effect of λ.

Figure 5: Parameter study on LA. Figure 5(a) and 5(b) show the effects of the latent dimension D and the number of epochs N on RA-Decay and RA-Cons. Figure 5(c) shows the effect of the decaying rate τ on RA-Decay. Figure 5(d) shows the effect of the regularization strength λ on RA-Cons.

making the result model suffer from overfitting. The effect of λ on the constraint-based learning is similar. A too large λ causes underfitting of the new records, while a too small λ causes overfitting. Besides the above parameters, we have also studied the effects of the smoothing parameters α and β, and found that the performance of ReAct varied no more than 3% when α and β are set to the range $[0.05, 0.5]$, thus we omit the results to save space.

5 RELATED WORK

Spatiotemporal Activity Modeling. Previous approaches to spatiotemporal activity modeling mostly rely on geographical topic modeling, which extends classical topic models to discover representative topics for different regions. Specifically, Sizov *et al.* [28] extend LDA [5] by assuming that each latent topic has a multinomial distribution over text, and two Gaussians over latitudes and longitudes. They later upgrade the model to find topics that have complex and non-Gaussian distributions [16]. Yin *et al.* [34] extend PLSA [12] by assuming that each region has a normal distribution that generates locations, as well as a multinomial distribution over the latent topics that generate text. While the above models are designed to detect crowd-level geographical topics, Hong *et al.* [13] and Yuan *et al.* [35] introduce the user factor in the modeling process to infer individual-level user preferences. Our work resembles the studies [16, 28, 34] more because we also model crowd-level activities instead of individual-level preferences. That said, ReAct differs from these methods notably. Instead of using latent states to indirectly bridge different data modalities, it jointly maps location, time, and text into the same space, which not only frees us from imposing distribution assumptions for different modalities, but also makes it scalable.

Representation learning techniques have been proposed for learning distributed representations for words [22], graph nodes [25, 29, 30], user-item interactions [10], multimedia data [23], *etc.*. Recently, Zhang *et al.* [38] propose CrossMap, a cross-modal representation learning method for spatiotemporal activity modeling. It first detects the hotspot regions and time periods underlying people's activities, and then jointly maps different regions, periods, and keywords into the same latent space. The key difference between ReAct and CrossMap is two-fold: (1) Instead of handling static data, ReAct processes continuous GTSM streams and learns recency-aware models online; and (2) while CrossMap learns cross-modal embeddings in an unsupervised way, ReAct adopts a semi-supervised framework, which is capable of leveraging activity category information to learn better-quality embeddings.

Spatiotemporal event detection. Several studies [1, 8, 15, 19, 20, 27, 40] use GTSM for spatiotemporal event detection. Sakaki *et al.* [27] train a classifier to judge whether an incoming tweet is related to an earthquake or not, and release an alarm when the number of earthquake-related tweets is large. Krumm *et al.* [19] 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 spatiotemporal event detection and spatiotemporal 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.

Human mobility modeling. There have also been studies that leverage GTSM data to extract mobility patterns underlying people's activities. Cho *et al.* [7] collect large-scale checkin data and find that people's activities are usually centered around a few fixed locations, and exhibit strong periodicity. Yuan *et al.* [36] propose a Bayesian non-parametric model to automatically extract the regions that a user visits periodically. Zhang *et al.* [37] apply sequential pattern mining techniques to extract frequent movement sequences from check-in data. Later, 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. Although these mobility modeling methods also models the time factor, they aim to understand the temporal transitions of human movements in the physical world, whereas we focus on the temporal evolution of global-level activities.

Temporal evolution modeling. A handful of studies [17, 26, 41] have investigated the temporal evolutions of spatiotemporal activities. Noulas *et al.* [24] analyze user activities with Foursquare checkins and find that the checkin data reveal meaningful spatiotemporal patterns. Zhang *et al.* [41] find people's activities exhibit temporal dynamics via analyzing the number of Foursquare checkins in different areas. Kling *et al.* [17] apply LDA to checkin data by treating hourly-aggregated checkins as documents. They show notable topical evolutions across different hours. Pozdnoukhov *et al.* [26] first apply online LDA to the streaming tweet data, and then analyze the spatiotemporal distributions of the extracted topics in a post-processing step. While these pioneering studies have clearly

demonstrated that dynamic evolutions exist in urban activities, they do not address our problem of designing a unified model that dynamically captures the inter-type correlations among location, time, and text from GTSM streams.

Modeling the temporal evolutions in data streams has also been studied for classical tasks like classification [33], clustering [2, 6], topic modeling [3, 4, 14, 21], and recommendation [11, 18]. For instance, Blei *et al.* [4] propose the Dynamic Topic Model, which uses the Markovian assumption and state space models to capture dynamic topical evolutions. Distinguished from these studies, our problem is new in that we aim to capture the dynamic evolutions for spatiotemporal activity modeling. From the technical standpoint, we model the dynamic evolutions in a novel framework of semi-supervised embedding, rendering our method fundamentally different from existing techniques.

6 CONCLUSION

We proposed a recency-aware spatiotemporal activity model that learns from geo-tagged social media streams online. Our proposed model, ReAct, embeds all the regions, hours, and keywords into the same latent space with a novel semi-supervised multimodal embedding paradigm. ReAct employs two strategies to emphasize the information contained in newly arrive records without overfitting. Our empirical evaluations have shown that ReAct outperforms the existing spatiotemporal activity modeling methods significantly in location and activity retrieval tasks.

7 ACKNOWLEDGEMENTS

We would like to 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. Luming Zhang was supported in part by the National Natural Science Foundation of China (Grant no. 61572169, 61472266), the National University of Singapore (Suzhou) Research Institute, and the Fundamental Research Funds for the Central Universities. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing any funding agencies.

REFERENCES

[1] H. Abdelhaq, C. Sengstock, and M. Gertz. Eventweet: Online localized event detection from twitter. *PVLDB*, 6(12):1326–1329, 2013.
[2] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu. A framework for clustering evolving data streams. In *VLDB*, pages 81–92, 2003.
[3] L. AlSumait, D. Barbará, and C. Domeniconi. On-line LDA: adaptive topic models for mining text streams with applications to topic detection and tracking. In *ICDM*, pages 3–12, 2008.
[4] D. M. Blei and J. D. Lafferty. Dynamic topic models. In *ICML*, pages 113–120, 2006.
[5] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(1):993–1022, 2003.
[6] D. Chakrabarti, R. Kumar, and A. Tomkins. Evolutionary clustering. In *KDD*, pages 554–560, 2006.
[7] E. Cho, S. A. Myers, and J. Leskovec. Friendship and mobility: User movement in location-based social networks. In *KDD*, pages 1082–1090, 2011.
[8] 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.
[9] 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.
[10] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T. Chua. Neural collaborative filtering. In *WWW*, pages 173–182, 2017.
[11] X. He, H. Zhang, M. Kan, and T. Chua. Fast matrix factorization for online recommendation with implicit feedback. In *SIGIR*, pages 549–558, 2016.
[12] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR*, pages 50–57, 1999.
[13] 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.
[14] L. Hong, D. Yin, J. Guo, and B. D. Davison. Tracking trends: incorporating term volume into temporal topic models. In *KDD*, pages 484–492, 2011.
[15] 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.
[16] 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.
[17] F. Kling and A. Pozdnoukhov. When a city tells a story: urban topic analysis. In *SIGSPATIAL*, pages 482–485, 2012.
[18] Y. Koren. Collaborative filtering with temporal dynamics. In *KDD*, pages 447–456, 2009.
[19] J. Krumm and E. Horvitz. Eyewitness: Identifying local events via space-time signals in twitter feeds. In *SIGSPATIAL*, 2015.
[20] 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.
[21] Q. Mei and C. Zhai. Discovering evolutionary theme patterns from text: an exploration of temporal text mining. In *KDD*, pages 198–207, 2005.
[22] 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.
[23] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng. Multimodal deep learning. In *ICML*, pages 689–696, 2011.
[24] 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.
[25] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: Online learning of social representations. In *KDD*, pages 701–710, 2014.
[26] A. Pozdnoukhov and C. Kaiser. Space-time dynamics of topics in streaming text. In *LBSN*, pages 1–8, 2011.
[27] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *WWW*, pages 851–860, 2010.
[28] S. Sizov. Geofolk: latent spatial semantics in web 2.0 social media. In *WSDM*, pages 281–290, 2010.
[29] J. Tang, M. Qu, and Q. Mei. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *KDD*, pages 1165–1174, 2015.
[30] 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.
[31] P. H. Verburg, P. P. Schot, M. J. Dijst, and A. Veldkamp. Land use change modelling: current practice and research priorities. *GeoJournal*, 61(4):309–324, 2004.
[32] P. Waddell. Urbansim: Modeling urban development for land use, transportation, and environmental planning. *Journal of the American Planning Association*, 68(3):297–314, 2002.
[33] H. Wang, W. Fan, P. S. Yu, and J. Han. Mining concept-drifting data streams using ensemble classifiers. In *KDD*, pages 226–235, 2003.
[34] Z. Yin, L. Cao, J. Han, C. Zhai, and T. S. Huang. Geographical topic discovery and comparison. In *WWW*, pages 247–256, 2011.
[35] 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.
[36] 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.
[37] 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.
[38] C. Zhang, K. Zhang, Q. Yuan, H. Peng, Y. Zheng, T. Hanratty, S. Wang, and J. Han. Regions, periods, activities: Uncovering urban dynamics via cross-modal representation learning. In *WWW*, pages 361–370, 2017.
[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.