Nested LSTM: Modeling Taxonomy and Temporal Dynamics in Location-Based Social Network

Xuan-An Tseng
National Tsing Hua University
killerjack003@gmail.com

Da-Cheng Juan*
Carnegie Mellon University
x@dacheng.info

Chun-Hao Liu
National Tsing Hua University
newgod1992@gapp.nthu.edu.tw

Wei Wei*
Carnegie Mellon University
weiwei@cs.cmu.edu

Yu-Ting Chen*
University of California, Los Angeles
ytchen@cs.ucla.edu

Jia-Yu Pan*
Carnegie Mellon University
jypan@cs.cmu.edu

Shih-Chieh Chang
National Tsing Hua University
scchang@cs.nthu.edu.tw

ABSTRACT

"Is there any pattern in location-based, mobile check-in activities?" "If yes, is it possible to accurately forecast the intention of a user's next check-in, given his/her check-in history?" To answer these questions, we study and analyze probably the largest mobile check-in datasets, containing 20 million check-in activities from 0.4 million users. We provide two observations—"*work-n-relax*" and "*diurnal-n-nocturnal*"—showing that the intentions of users' check-ins are strongly associated with time. Furthermore, the category of each check-in venue, which reveals users' intentions, has structure and forms taxonomy. In this paper, we propose **Nested LSTM** that takes both (a) time and (b) taxonomy structure from check-in sequences into consideration, providing accurate predictions on the category of a user's next check-in location. Nested LSTM also projects each category into an embedding space, providing a new representation with strong semantic meanings. Experimental results are poised to demonstrate the effectiveness of the proposed Nested LSTM: (a) Nested LSTM improves *Accuracy@5* by 4.22% on average, and (b) Nested LSTM learns a better taxonomy embedding for clustering categories, which improves *Silhouette Coefficient* by 1.5X. Both results (a)(b) are compared with LSTM-based, state-of-the-art approaches.

KEYWORDS

Long Short-Term Memory; Location-Based Social Network; Point of Interest; Behavior Model

1 INTRODUCTION

"Is there any pattern in location-based, mobile check-in activities?" "If yes, is it possible to accurately forecast the intention of a user's next check-in, given his/her check-in history?" These questions serve as the motivations for this work.

Location-Based Social Networks (LBSN) are rising due to the ubiquity of GPS-equipped smartphones. In LBSN, users bridge the gap between the physical world and the online social networks by checking in their footprints on the visited venues, referred as Point Of Interests (POIs). The category of a POI is often associated with certain activities related to a user's intention [10, 12, 15, 21]. Furthermore, a user's next intention and activity can be modeled and even predicted by analyzing these temporal check-in sequences on POIs. For example, if a person checks in at the office during the daytime, after work he/she may check in at a bar or a restaurant, and eventually checks in at home. Similarly, if a traveler frequently checks in at sightseeing spots, he/she may later on checking in at a metro station or hotel. Understanding and modeling these temporal dynamics of users' intentions or behaviors enable many

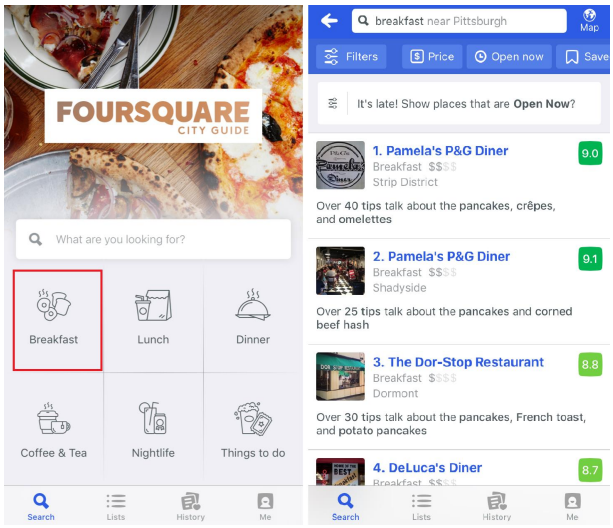


Figure 1: User interface of Foursquare application on a smartphone. When a user presses the button "Breakfast", Foursquare application will show a list of POIs which falls into "breakfast" nearby based on GPS.

*Recently working at Google, Mountain View, CA, USA.

MiLeTS '18, August 2018, London, United Kingdom

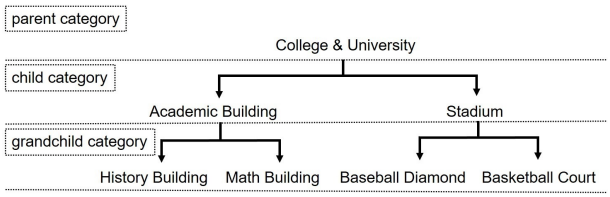


Figure 2: Example taxonomy of hierarchical category in Foursquare and Jiepang datasets. One parent category in a taxonomy contains several child categories. Similarly, one child category contains several grandchild categories.

useful applications, such as recommendation systems which are widely deployed in many products [24]. Given the history of his/her check-ins, POI recommender automatically suggests a venue that matches his/her next intention and activity, such as a relaxing bar after work or a nearby sightseeing spot after getting out from a metro.

One challenge in LBSN is how to accurately forecast the next intention of a user. Figure 1 demonstrates a mobile application called "Foursquare" [1], which provides location-based search services and recommendations. First, a user can select a category which he/she is interested in, and then the app will show a list of POIs which falls into that category nearby based on GPS. If a user's intention or preference can be predicted, such a model can be integrated into a recommender for more better suggestions, which in turn improves user experience. To better understand users' intentions and preferences, we study the public check-in logs from Foursquare and Jiepang [2] containing 20 million check-in activities from 0.4 million users in total. The categories of venues from these datasets are hierarchical and form a taxonomy shown in Figure 2. For example, parent category "*College & University*" includes child categories "*Academic Building*" and "*Stadium*" while child category "*Academic Building*" also includes "*History Building*" and "*Math Building*".

A challenge here is how to model both the taxonomy and long sequences of LBSN data at the same time. For modeling sequences, one popular and effective approach is Long Short-Term Memory (LSTM) [8]—famous for its superior ability to preserve sequence information over time. The order of check-in sequences is the key to modeling the subtle intention from a user; for example, a person takes metro to company in the morning, eats lunch at a restaurant, has a teatime in a coffee shop, and chats with friends at a bar after work; therefore, the expansion of these logs is exactly fitting to LSTM's characteristic of sequence modeling. We aim at predicting the next category of POI which a user is interested in by expanding user check-in logs as input sequences of LSTM to model users' activity preferences.

This paper brings the following contributions:

- We analyze two large-scale Location-Based Social Networks datasets from Foursquare and Jiepang, and provide two observations: "*work-n-relax*" and "*diurnal-n-nocturnal*" showing that the intentions of users' check-ins are strongly associated with time.
- We propose a novel and effective model: **Nested LSTM** that accurately forecasts the next POI category. Nested LSTM also captures the hierarchical structure of POI categories.
- To understand the effectiveness of Nested LSTM, we also perform a comprehensive study and comparison with state-of-the-art approaches. Experimental results show that, on average, Nested LSTM outperforms state-of-the-art approaches by 4.22% on *Accuracy@5* metric.
- Furthermore, Nested LSTM learns a more effective taxonomy embedding for clustering categories. Experimental results show that Nested LSTM improves *Silhouette Coefficient* by 1.5X, compared with the embedding learned by vanilla LSTM.
- Finally, we provide a practitioners' guide for deploying Nested LSTM to forecast the next POI category. Nested LSTM can be seamlessly integrated into recommendation system for improving user experience.

The remainder of this paper is organized as follows: Section 2 provides the problem definition and specifications of Foursquare and Jiepang datasets. Section 3 provides a crash course to LSTM and the details of the proposed Nested LSTM. Section 4 presents the experimental results and the analysis of taxonomy embedding. Section 5 provides a practitioners' guide, and Section 6 provides previous works. Finally, Section 7 concludes this paper.

2 PROBLEM DEFINITION

In this section, we provide an overview of Foursquare and Jiepang datasets, and two observations: "*work-n-relax*" and "*diurnal-n-nocturnal*" showing that the intentions of users' check-in are strongly associated with time. Next, we formulate the definition of our problem. Finally, we detail the illustration of data preprocessing.

2.1 Datasets: Foursquare & Jiepang

We analyze the public check-in posts from Foursquare and Jiepang websites. The specifications of both datasets are as follows: Foursquare dataset contains over 11 million check-in activities at 560 thousand venues collected from 56 thousand users in the United States from February 2010 to January 2011; Jiepang contains over 8 million check-in activities at 87 thousand venues collected from 382 thousand users in China from December 2010 to March 2013. Each check-in post contains an unique user ID, an unique venue ID indicating the POI and a time stamp of the check-in happened. We remove the abnormal check-in logs (*e.g.*, the latitude and longitude of a venue are all 999.0), which account for less than 0.001% of both datasets. All the venues in the datasets are marked with the hierarchical categories, as illustrated in Figure 2. There are 312 child categories within 12 parent categories in the Foursquare dataset and 51 child categories within 7 parent categories in the Jiepang dataset. We treat singular and plural forms as the same category. We only implement two-level categories (parent and child category) of our taxonomy in the following experiments because some information of the grandchild categories are incompleted.

D. Yang et al. [20] address that users' activities are often associated with the time called *"Temporal Correlation"*. For example, D. Yang et al. observe that people usually go to a coffee shop or a burger joint between 13:00 to 14:00 on a weekday, stay at a bar between 21:00 to 22:00 on Friday and go to the gym or outdoor places

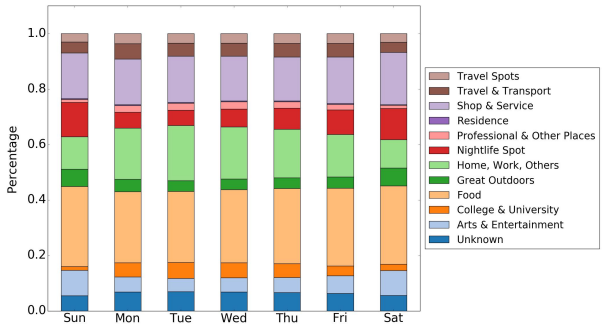


Figure 3: Percentage of each parent category in a week after we normalize the actual check-in count in Foursquare dataset. The percentage of category associated with "work" in weekdays is larger than the ones on weekends. On the contrary, the percentage of category associated with "relax" on weekends is larger than the ones in weekdays.

between 16:00 and 17:00 on weekend. The order of these check-in sequences causes a strong weekly pattern for a user. Therefore, we analyze the correlation of time and category in our datasets and get two observations as follows.

Observation 1. *"Work-n-relax" pattern: weekday check-ins can usually be associated with "work," and weekend check-ins are usually related to entertainment & relax.*

Figure 3 shows the breakdown of each parent category over a week after we normalize the actual check-in count in Foursquare dataset. The number of check-in count increases from Monday to Saturday and starts to decrease until Monday. On weekdays (i.e., from Monday to Friday), the percentage of category "*College & University*", "*Professional & Other places*" and "*Home, Work, Others*" are much larger than the ones on weekends, said that people usually work and study on weekdays. On weekends, the percentage of category "*Nightlife*", "*Shop & Service*", "*Arts & Entertainment*" and "*Great Outdoors*" are much larger than the ones on weekdays, said that people spend their time on relaxing and entertainment on weekends. Although "*Travel & Transport*", "*Travel Spots*" and "*Food*" have almost the same percentage on weekdays or weekends, we speculate for two reasons: 1) Commuters often check in at transportations on weekdays; however, travelers occasionally check in at travel spots on weekends. Therefore, the difference between the percentage of category "*Travel & Transport*" and "*Travel Spots*" on weekdays and weekends are smoothed and almost the same. 2) People go to restaurants and shops for food everyday because of life necessities.

Observation 2. *"Diurnal-n-nocturnal" pattern: check-in venues during the day can be very different from the ones during the night.*

Figure 4 are three pie charts of the check-in count of parent categories after we normalize the actual check-in count during a day in Foursquare dataset, we select the time 3:00 to 3:59, 10:00 to 10:59 and 20:00 to 20:59 which have a strong contrasting pattern to each other. During 3:00 to 3:59, most people enjoy their nightlife results in category "*Nightlife Spot*" accounting over 23%, which is much

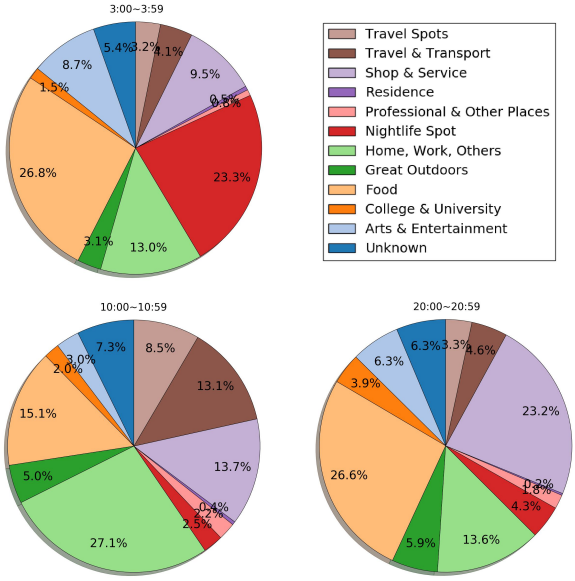


Figure 4: Pie charts of each parent category within 3:00 to 3:59, 10:00 to 10:59 and 20:00 to 20:59 after normalizing the actual check-in count during a day in Foursquare dataset. The percentage of checked-in category by "diurnal" users in daytime is larger than the ones at night. On the contrary, the percentage of checked-in category by "nocturnal" users at night is larger than the ones in daytime.

larger than daytime. At 10 a.m., commuters take transportations to work place or school so the check-in count of category "*Travel & Transport*" and "*Home, Work, Others*" increase. Until 20:00, people who after work or school go to find some food and shop to soothe the tiredness of a day, that's why the category "*Shop & Service*" and "*Food*" occupy over 50% of the pie chart.

2.2 Problem Formulation

The problem formulation can be described as: *"Given each user's check-in sequence, forecast the child category of POI in his/her next check-in."*

Specifically, from each user's check-in sequence, we partition them into several instances with length of τ, *e.g.*, instance 1 is 1^{st} to τ^{th} check-in, instance 2 is 2^{nd} to $(\tau+1)^{th}$ check-in, instance i is i^{th} to $(i+\tau-1)^{th}$ check-in. We use Figure 5 to better illustrate the instance partition from a user check-in sequence. The feature notations of an instance as follows:

- Parent category at j^{th} check-in of an instance, denoted as $x_{P,j}$.
- Child category at j^{th} check-in of an instance, denoted as $x_{C,j}$.
- Check-in time at j^{th} check-in of an instance, denoted as $x_{T,j}$.

where j from 1 to τ. Nested LSTM forecasts the child category of POI in the next check-in, *i.e.*, the $(\tau+1)^{th}$ check-in as label y. Mathematically, this problem can be expressed as:

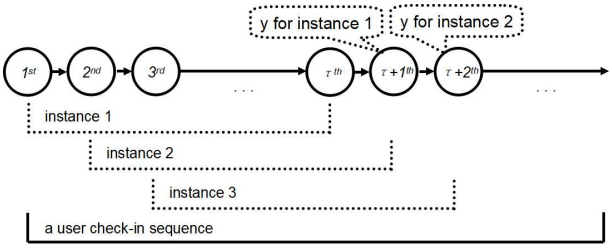


Figure 5: One user check-in sequence is partitioned into several instances. Each instance is with length of τ: instance 1 contains the 1^{st} to the τ^{th} check-in, instance 2 is the 2^{nd} to $(\tau+1)^{th}$ check-in, and so on. Given an instance, the goal is to predict the child category of POI in the next check-in—for example, given instance 1, the goal is to forecast the child category of POI in the $(\tau+1)^{th}$ check-in.

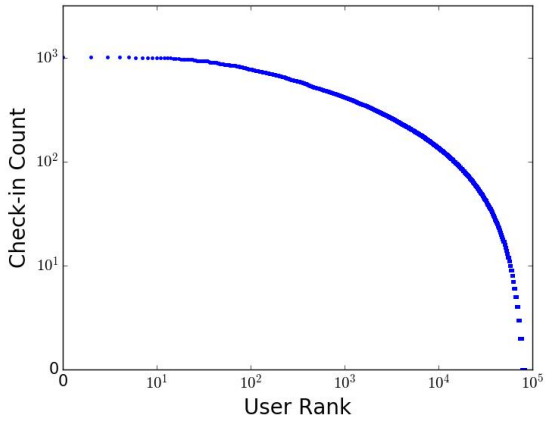


Figure 6: The ranking of user check-in count from the most frequent user to the last correspond to their count. Both of X-axis and Y-axis take log scale. This distribution matches *power-law-like*, which means the check-in count decrease progressively from most frequent user to the last.

$$y = f(x_{P,j}, x_{C,j}, x_{T,j}), \text{ where } j = 1 \text{ to } \tau \quad (1)$$

The goal here is to find a function f that takes $x_{P,j}$, $x_{C,j}$ and $x_{T,j}$ ($j = 1$ to τ) as inputs to predict y. Note that we predict the child category instead of the parent category. As Section 2.1 mentioned, the species of child categories are much more than parent categories. In simpler words, predicting a child category provides the "fine-grained" intention or preference of a user. For example, predicting the next POI category as "*Chinese Food Restaurant*" reveals more intention or preference of a user, compared to its parent category "*Food*," which is very important for designing a recommendation system.

2.3 Data Preprocessing

We model the prediction of the next POI category which a user is interested in from Eq (1) as a multi-class classification and construct the training datasets accordingly. Each instance of features consists of label y that represents the child category of $(\tau+1)^{th}$ check-in and a set of predictive features x represents an input sequence from 1^{st} to τ^{th} check-in extracted from an instance. The period of each check-in log between each neighbor is no longer than 24 hours to ensure the tight relation of each check-in and to filter out inactive users who seldom check in during a week.

The type of label y is determined based on the child category of $(\tau+1)^{th}$ check-in of an instance formulated in Section 2.2. We give each child category an index for y, *e.g.*, the number between 0 to 311 to represent 312 kinds of child category in Foursquare dataset.

For the predictive features x, we leverage the past check-in sequences of parent categories, child categories and the check-in time. We give each $x_{P,j}$ and $x_{C,j}$ an index to represent each parent category and child category respectively. Then we define $x_{T,j}$ as features of check-in time in Eq (1) which represent the weekday in a week and the hour in a day.

Now, the question is: How to select τ from user check-in sequences that needed to be included in the datasets for an accurate prediction? Figure 6 illustrates the ranking of user check-in logs correspond to their count. This distribution matches *power-law-like*, which means the check-in count decreases progressively from the most frequent user to the last. When τ is smaller, we get a shorter instance, from both active and inactive users, and we can extract more training samples. When τ is larger, we can only get a longer instance from active users, but we can merely extract less training samples. Therefore, the selection of τ is the first important question we need to face. We provide experimental results in Section 4.2 for different values of τ we selected for the best results. Overall, x has 3 kinds of features ($x_{P,j}$, $x_{C,j}$ and $x_{T,j}$) $\times\ \tau$ (sequence length per instance) = 3τ predictive features in a training and testing instance.

3 METHODOLOGY

In this section, we give a crash course of one popular and effective approach in Neural Networks— Long Short-Term Memory. Then, we detail the structure and equations of Nested LSTM we proposed.

3.1 Crash Course to Long Short-Term Memory

Recurrent Neural Network (RNN) processes input sequences of arbitrary length by recursively activating transition function on a *hidden state vector* h_t. At each time step t, recurrent neurons receive an input vector x_t and their previous hidden state vector h_{t-1}, then feed into the transition function for the hidden state vector h_t as output. Basically, the transition function of RNN is a nonlinear transformation between input and output, *e.g.*, the hyperbolic tangent function in Eq (2). W_x, W_h are the weight matrices for input vector x_t and previous hidden state vector h_{t-1}; b is the bias of this function.

$$h_t = tanh(W_x x_t + W_h h_{t+1} + b) \quad (2)$$

During training RNN, the components of the gradient vector in this form of transition function can grow or decay exponentially over long sequences. This behavior of RNN causes a serious problem called "*exploding*" or "*vanishing gradients*" [3, 7], which makes RNN model difficult to learn long-distance correlations in a sequence.

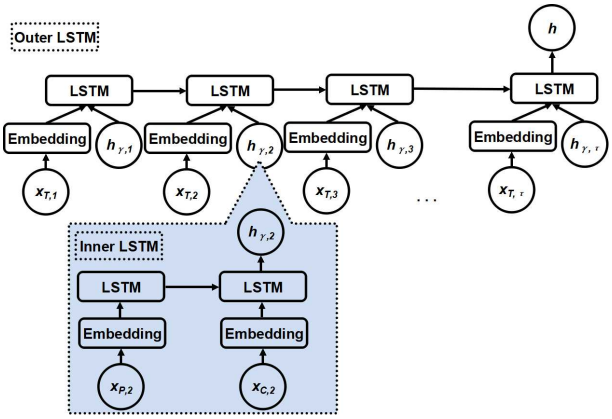


Figure 7: Structure of Nested LSTM. Two LSTM layers are used in this model: *Inner LSTM* and *Outer LSTM*. Inner LSTM outputs taxonomy embedding $h_{Y,j}$ as the input of Outer LSTM.

The LSTM architecture solves the problems of exploding and vanishing gradients by introducing a "*memory cell*" to preserve cell state over long periods of time. While numerous LSTM variants have been introduced, we describe the version used by O. Vinyals et al. [18] and K. Tai et al. [17].

We define d *memory dimensions* of LSTM units at each time step t to be a collection of vectors in $\mathbb{R}^d$. Each unit contains an *input gate* i_t, a *forget gate* f_t, an *output gate* o_t, a *memory cell state* c_t and a *hidden state* h_t. The value of these gating vectors i_t, f_t and o_t are in [0;1].

$$\begin{aligned} i_t &= \sigma(W_x^{(i)} x_t + W_h^{(i)} h_{t-1} + b) \\ f_t &= \sigma(W_x^{(f)} x_t + W_h^{(f)} h_{t-1} + b) \\ o_t &= \sigma(W_x^{(o)} x_t + W_h^{(o)} h_{t-1} + b) \\ u_t &= tanh(W_x^{(u)} x_t + W_h^{(u)} h_{t-1} + b) \\ c_t &= i_t \odot u_t + f_t \odot c_{t-1} \\ h_t &= o_t \odot tanh(c_t) \end{aligned} \tag{3}$$

Eq (3) are the transition equations of a single LSTM unit, where x_t is the input at time step t, h_{t-1} is the previous hidden state, c_t and c_{t-1} are the memory cell states from time step t and previous time step $t-1$, σ denotes the logistic sigmoid function and $\odot$ denotes elementwise multiplication. In general, the forget gate f_t controls the forgotten extent of previous memory cell state c_{t-1}, the input gate i_t controls the update extent of each unit, the memory cell state c_t decides the state of this LSTM unit and the output gate o_t controls the exposure of memory cell state c_t at time step t. Therefore, the hidden state vector h_t is a partial view of the state of an LSTM unit's memory cell state. Since the value of these gating variables vary for each vector element, the LSTM model can learn to preserve information after multiple time scales.

3.2 Nested LSTM

We illustrate the detailed framework of Nested LSTM with Figure 7. At first, we extract input features ($x_{P,j}$, $x_{C,j}$, $x_{T,j}$) where $j = 1$ to τ and label y from each instance which already described in Section 2.3.

In Eq (4), we feed $x_{P,j}$, $x_{C,j}$ and $x_{T,j}$ into different *embedding layers*, which turn indexes of category and check-in time into dense vectors of fixed size, to produce embedding vectors $e_{P,j}$, $e_{C,j}$, and $e_{T,j}$.

$$\begin{aligned} e_{P,j} &= Emb(x_{P,j}) \\ e_{C,j} &= Emb(x_{C,j}) \\ e_{T,j} &= Emb(x_{T,j}) \end{aligned} \tag{4}$$

In Eq (5), we pack up each sequence of $e_{P,j}$, $e_{C,j}$ as input of *Inner LSTM* to output a taxonomy embedding $h_{Y,j}$ which represents the hierarchical relationship from parent category to child category.

$$h_{Y,j} = LSTM(e_{P,j} \rightarrow e_{C,j}) \tag{5}$$

Then, we feed the taxonomy embedding $h_{Y,j}$ and the embedding vector $e_{T,j}$ into *Outer LSTM* to capture the sequence information in Eq (6) and get the last output as the internal vector h. Outer LSTM feed the internal vector h to *softmax layer* to make the final decision of next POI category form N child categories.

$$h = LSTM(h_{Y,j}, e_{T,j}) \tag{6}$$

Softmax function is added in the softmax layer to determine the final prediction of the next POI category. The output of the softmax function can be used to represent a categorical distribution, which is a probability distribution over N different possible outcomes. Eq (7) is the predicted probability for the k^{th} class from N child categories given an internal vector h in Eq (6) and weight matrix W_k where $k = 1$ to N.

$$P(y = k|h) = \frac{exp(W_k h)}{\sum_{l=1}^{N} exp(W_l h)} \tag{7}$$

We pick the output of the softmax layer with the highest probability for Accuracy@1 and top k probabilities for Accuracy@k. Note that Inner LSTM only feeds one sequence of $e_{P,j}$, $e_{C,j}$ as input to output one taxonomy embedding $h_{Y,j}$ but Outer LSTM feeds τ sequences of $h_{Y,j}$, $e_{T,j}$ as input to output one internal vector h.

4 EXPERIMENTAL RESULT

In this section, we first describe the prerequisites for conducting experiments, then present the experimental results from proposed Nested LSTM and also other state-of-the-art approaches. Finally, we analyze the taxonomy projected to an embedding space.

4.1 Experimental Setup

While training and testing models, *Cross Entropy* is used to measure the difference between the true class y and the distribution of predictive classes $\hat{y}$, which is described in Eq (8):

$$J = -\frac{1}{N} \sum_{k=1}^{N} P(y^{(k)}) log P(\hat{y}^{(k)}) + \frac{\lambda}{2} ||w||^2 \tag{8}$$

where N is the number of class of child category, P is the probability of softmax function in Eq (7), the superscript k indicates the k^{th} class of the child category, and λ is a L2 regularization hyperparameter.

To best train the proposed Nested LSTM and other state-of-the-art approaches, we manage to search for the best hyperparameters by using 10-fold cross validation. Then we evaluate the performance of each model on the test set.

For the evaluation metrics, we report *Precision, Recall, Accuracy@1* and *Accuracy@5* to provide a comprehensive study on the performance evaluation of different models. *Precision*, *Recall* and *Accuracy@k* are defined as:

$$\begin{aligned} Precision &= \frac{True\ Positives}{True\ Positives + False\ Positives} \\ Recall &= \frac{True\ Positives}{True\ Positives + False\ Negatives} \\ Accuracy@k &= \frac{\#\ correct\ prediction\ in\ top\ k\ candidates}{\#\ total\ prediction} \end{aligned} \tag{9}$$

We compare the performance evaluation with other state-of-the-art approaches, which are the variants of the basic LSTM model as follows:

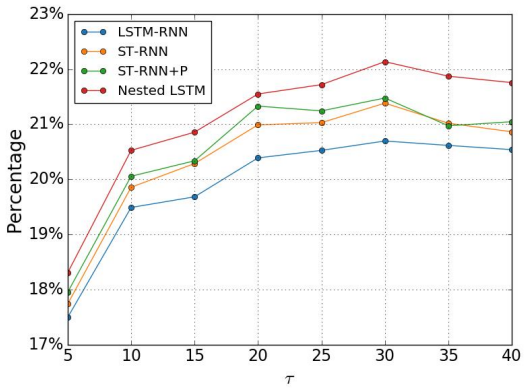
- **LSTM-RNN:** We reimplement LSTM-RNN proposed by [13] that feeds child category $x_{C,j}$ as input features through an embedding layer for $j = 1$ to τ.
- **ST-RNN:** We reimplement ST-RNN proposed by [11] that feeds child category $x_{C,j}$ and check-in time $x_{T,j}$ as input features through embedding layers for $j = 1$ to τ.
- **ST-RNN+P:** This model is an extension of ST-RNN. We reimplement the model which feeds parent category $x_{P,j}$, child category $x_{C,j}$ and check-in time $x_{T,j}$ as input features through embedding layers for $j = 1$ to τ.

We use the softmax layer as the last layer to decide the final prediction for these state-of-the-art models.

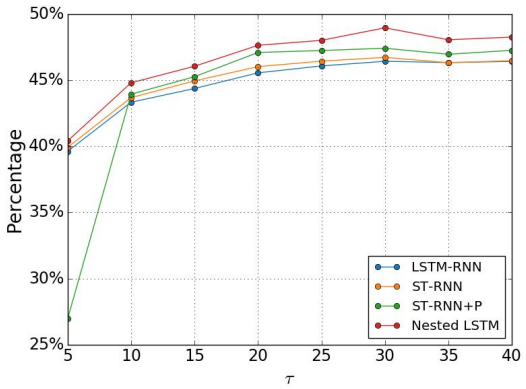
4.2 Result Summary

As the problem mentioned in Section 2.3, we try to find the best value of τ for Nested LSTM. Figure 8 shows the three different evaluation metrics with different values of τ on state-of-the-art models and Nested LSTM from Foursquare dataset, we also do the same searching method on Jiepang dataset. Furthermore, *Accuracy@1* and *Accuracy@5* increase when $\tau = 5$, arrive at the peak value when $\tau = 30$, and start to decrease until the end. *Loss* calculated by Eq (8) also reaches the lowest point when $\tau = 30$. We finally find the balance point of τ which is not the corner value.

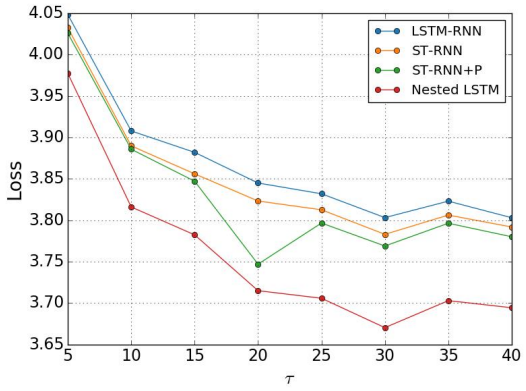
Table 1 shows the experimental results from different state-of-the-art models and Nested LSTM with three different τ values selected in both of our datasets. We calculate and report *Accuracy@1*, *Accuracy@5* and improvement from baseline for comprehensive comparisons. The results demonstrate that Nested LSTM outperforms state-of-the-art models on both Foursquare and Jiepang datasets. We observe that LSTM-RNN (the only approach here without using check-in time) has the worst performance, indicating that check-in time is an informative feature which should always be included for behavior and intention modeling. When using the



(a) *Accuracy@1* with different values of τ. *Accuracy@1* of all models increase when $\tau = 5$, arrive at the peak value when $\tau = 30$, and start to decrease until the end.



(b) *Accuracy@5* with different values of τ. *Accuracy@5* of all models increase when $\tau = 5$, arrive at the peak value when $\tau = 30$, and start to decrease until the end.

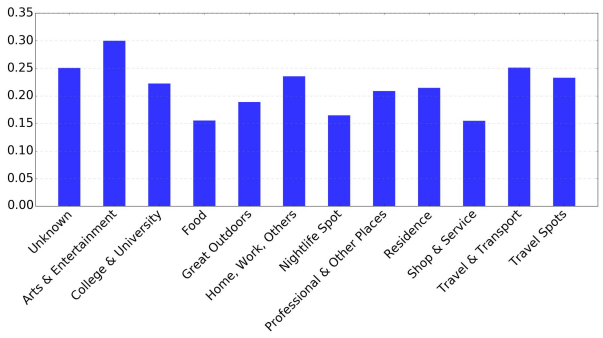


(c) *Loss* with different values of τ. *Loss* of all models reach the lowest point when $\tau = 30$, and start to increase until the end.

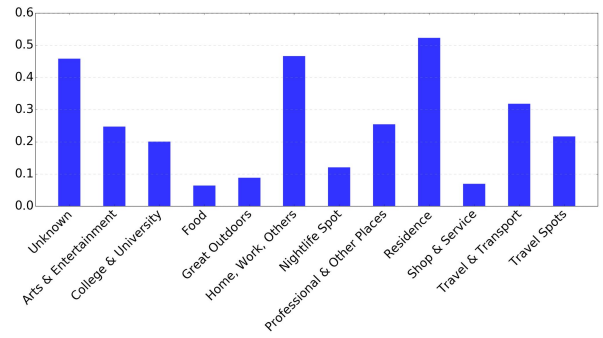
Figure 8: Calculate the evaluation metrics with different values of τ on state-of-the-art models and Nested LSTM from Foursquare dataset.

Table 1: Experimental results on both Foursquare and Jiepang datasets with various τ values. Notice that the proposed Nested LSTM consistently outperforms the previous state-of-the-art approaches over different τ values.

Foursquare	$\tau = 25$		$\tau = 30$		$\tau = 35$	
Model	Accuracy@1	Accuracy@5	Accuracy@1	Accuracy@5	Accuracy@1	Accuracy@5
LSTM-RNN [13]	20.53% (baseline)	46.08% (baseline)	20.70% (baseline)	46.43% (baseline)	20.62% (baseline)	46.33% (baseline)
ST-RNN [11]	21.01% (+2.34%)	46.44% (+0.78%)	21.38% (+3.29%)	46.73% (+0.65%)	21.02% (+1.94%)	46.33% (+0.00%)
ST-RNN+P	21.24% (+3.46%)	47.25% (+2.54%)	21.48% (+3.77%)	47.42% (+2.13%)	20.97% (+1.70%)	46.93% (+1.30%)
Nested LSTM	**21.72% (+5.80%)**	**48.02% (+4.21%)**	**22.14% (+6.96%)**	**48.96% (+5.45%)**	**21.87% (+6.06%)**	**48.06% (+3.73%)**
Jiepang	$\tau = 40$		$\tau = 45$		$\tau = 50$	
Model	Accuracy@1	Accuracy@5	Accuracy@1	Accuracy@5	Accuracy@1	Accuracy@5
LSTM-RNN [13]	35.66% (baseline)	64.78% (baseline)	36.09% (baseline)	65.25% (baseline)	34.29% (baseline)	64.60% (baseline)
ST-RNN [11]	35.38% (-0.79%)	64.52% (-0.04%)	36.00% (-0.25%)	65.38% (+0.02%)	34.45% (+0.47%)	64.34% (-0.04%)
ST-RNN+P	35.58% (-0.22%)	65.30% (+0.08%)	36.28% (+0.53%)	66.29% (+1.59%)	34.44% (+0.44%)	64.84% (+0.37%)
Nested LSTM	**37.55% (+5.30%)**	**66.98% (+3.40%)**	**37.99% (+5.27%)**	**68.20% (+4.52%)**	**37.00% (+7.90%)**	**67.19% (+4.00%)**



(a) *Precision* of each parent category predicted by Nested LSTM. Nested LSTM performs well on predicting category "*Arts & Entertainment*."



(b) *Recall* of each parent category predicted by Nested LSTM. Nested LSTM can accurately predict on category "*Home, work, Others*" and "*Residence*."

Figure 9: Calculate *Precision* and *Recall* by Eq (9) for Nested LSTM with $\tau = 30$ from Foursquare dataset.

same set of input features, Nested LSTM outperforms ST-RNN+P by every evaluation metric.

Furthermore, we provide *Precision* and *Recall* of each parent category predicted by Nested LSTM from Foursquare dataset in Figure 9 calculated by Eq (9). From Figure 9(a), Nested LSTM performs well on predicting category "*Arts & Entertainment*" but the difference of each category prediction is slight. From Figure 9(b), Nested LSTM can accurately predict on category "*Home, work, Others*" and "*Residence*" but incorrectly predicts on category "*Food*", "*Great Outdoors*", "*Nightlife Spot*" and "*Shop & Service*".

We further provide the confusion matrix of each parent category predicted by Nested LSTM from Foursquare dataset after we normalize the actual count in Figure 10. In general, category predictions from Nested LSTM match the ground truth very well, as most of the mass locates on the diagonal. Yet Nested LSTM seems to make more predictions on "*Home, work, Others*" (notice the corresponding column with slightly deeper color) than other categories. This modeling behavior can be attributed to label imbalance: many users in Foursquare dataset often check in at their own homes or workplaces, resulting in a larger number of check-ins with "*Home, work, Others*." This gives an intriguing opportunity to further improve Nested LSTM with techniques for handling label imbalance, *e.g.*, down-sampling; we leave this as an interesting and straight-line future work.

4.3 Taxonomy Embedding Analysis

To detail the better performance of Nested LSTM, we analyze the difference between taxonomy embedding $h_{Y,j}$ of Nested LSTM and embedding vectors $e_{C,j}$ of ST-RNN+P, which both represent the child categories. Figure 11 are the TSNE [14] graphs of $h_{Y,j}$ and $e_{C,j}$, which transfer high dimensions of vector into X-axis and Y-axis. The dots with the same color in the TSNE graph indicate the child categories from the same parent category in Foursquare dataset. Figure 11(a) is messy and Figure 11(b) is organized with the same color. Inner LSTM performs well on matching the hierarchical relationship of the path from the parent category to the child category on taxonomy.

In addition, we provide two evaluation metrics: *Silhouette Coefficient* [14] and *Euclidean Distance*. Both of them are used to measure the distance between clusters. The vectors of Nested LSTM and ST-RNN+P are normalized before we calculate. *Silhouette Coefficient* is calculated by using the mean of intra-cluster distance α and the mean of nearest-cluster distance β for each sample in Eq (10).

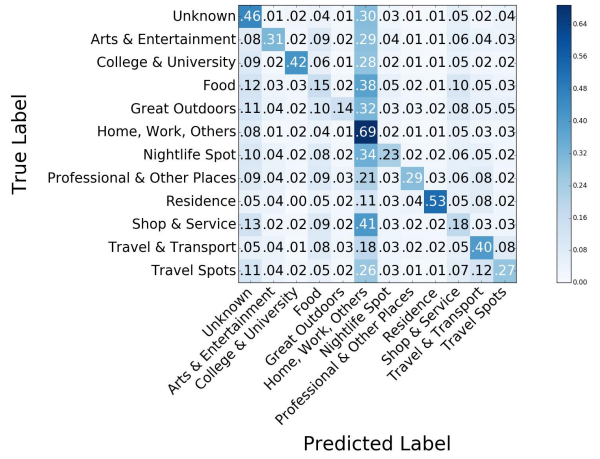


Figure 10: The confusion matrix of the parent category predicted by Nested LSTM from Foursquare dataset after we normalize the actual count. The depth of colors represents the proportion of each category.

$$\frac{\beta - \alpha}{max(\alpha, \beta)} \tag{10}$$

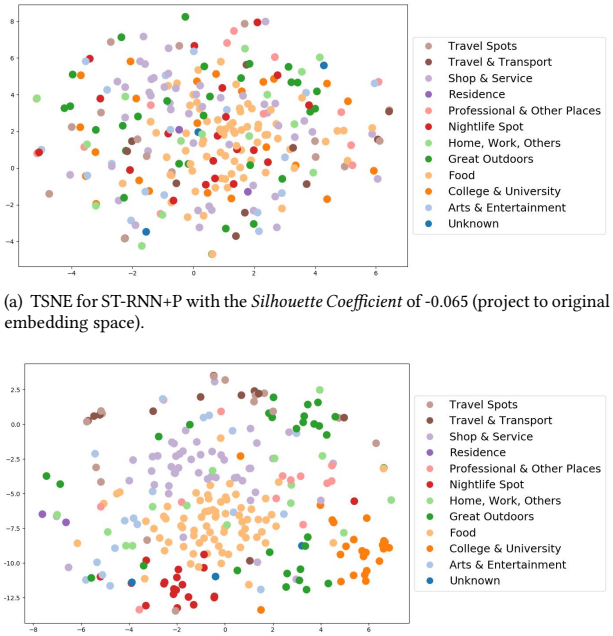
The value of *Silhouette Coefficient* is in $[-1; 1]$. *Silhouette Coefficient* of Nested LSTM is 0.034 while ST-RNN+P is only -0.065, which means Inner LSTM has the capability of clustering child categories.

We calculate the average of *Euclidean Distance* from each element to the mean value of its cluster, as the radius of the cluster. Figure 12 shows the calculated results of Nested LSTM and ST-RNN+P. In each parent category, Nested LSTM's *Euclidean Distance* is smaller than ST-RNN+P; on average, the mean of Nested LSTM is 0.812 while ST-RNN+P is 0.919. These results show that taxonomy embedding learned by Nested LSTM enables more distinct and clear clusters: smaller intra-cluster distance and larger inter-cluster distance (both effects captured by *Silhouette Coefficient*).

5 PRACTITIONERS' GUIDE

Here we provide the practitioner's guide to apply Nested LSTM for forecasting next check-in category of a user:

- **Construct training dataset:** Build a database by including the SQL files and partition user check-in sequences by their unique ID described in Section 2.1. Filter out the user whose total check-in count is less than $\tau + 1$ times and the period between each check-in is longer than a day. The remaining part of the sequence is the training set, which is partitioned as input features $x_{P,j}$, $x_{C,j}$, $x_{T,j}$ where $j = 1$ to τ of each instance and the child category of $(\tau + 1)^{th}$ check-in as y.
- **Construct Nested LSTM:** After constructing the training set of $x_{P,j}$, $x_{C,j}$, $x_{T,j}$ and y, build a neural network model with the structure described in Figure 7. The dimension of embedding layers is a fixed value based on the scale of the vocabulary size of category and check-in time. The dimension of softmax layer is the same as the size of the child category. The dimension of Inner LSTM and Outer LSTM are optional based on τ. We find the best hyperparameters via grid-search and cross-validation.



(a) TSNE for ST-RNN+P with the *Silhouette Coefficient* of -0.065 (project to original embedding space).

(b) TSNE for Nested LSTM with the *Silhouette Coefficient* of 0.034 (project to original embedding space).

Figure 11: TSNE graphs for embedding vectors $e_{C,j}$ of ST-RNN+P and taxonomy embedding $h_{Y,j}$ of Nested LSTM from Foursquare dataset. The dots with the same color in the graphs indicate the child categories from the same parent category.

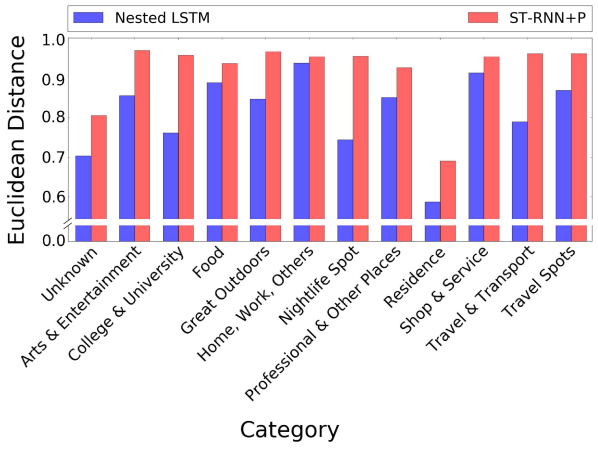


Figure 12: The average of *Euclidean Distance* from each element to the mean value of its cluster, as the radius of the cluster. On average, the mean of Nested LSTM is 0.812 while ST-RNN+P is 0.919.

After Nested LSTM are trained, each new incoming instance can generate a prediction of the child category. It can be used for both (a) offline analysis in other similar datacenters, and (b) serving as a recommend system integrated into a cloud/cluster and connected to mobile applications, with training via historical data offline.

6 RELATED WORK

We classify the previous work into two parts as follows.

Conventional Approaches. The collaborative filtering (CF) techniques can be tailored for POI prediction and may collaborative filtering methods have been proposed [6, 9, 16]. Yuan et al. [22] find that POIs are time-dependent, so they develop a model which can capture temporal information to make the prediction more accurate.

Besides CF techniques, Chang et al. [4] incorporate different features in the LDA model for next POI prediction. Gao et al. [5] propose a social-historical model based on Hierarchical Pitman-Yor process for predicting the next check-in of a user. Ye et al. [21] apply a framework which uses a mixed hidden Markov model to predict the category of user activity at the next time.

None of these works apply Neural-based methods, *e.g.*, RNN or LSTM, to accurately predict the next users' behavior.

Neural-based Approaches. As we know, RNN or LSTM have been widely applied to model sequential data, and they also have been successfully adopted in the prediction problems. For the typical prediction task, e.g., movies, song, book, goods, Wu et al. [19] propose the Recurrent Recommender Networks (RRN) which are able to predict the movie which a user is interested in. Zhu et al. [25] propose a new variant of LSTM called Time-LSTM to predict what will a user buy next.

Contrast with typical prediction tasks, to predict a location is more complex. We have to consider a wide variety of contextual factors, e.g., temporal context, geographical influence, and sequential relations. Zhang et al. [23] implement the framework named NEXT which predicts the next POI a user will visit. Liu et al. [11] propose a novel method called Spatial Temporal Recurrent Neural Networks (ST-RNN) which models local temporal and spatial contexts to predict next location.

Compared with the approaches above, we not only consider the temporal and spatial contexts, but also take taxonomy from check-in sequences into consideration. Nested LSTM provides an accurate prediction on the category of next POI which a user is interested in.

7 CONCLUSION

In this paper, we first analyze two large-scale LBSN datasets and provide two observations: "*work-n-relax*" and "*diurnal-n-nocturnal*." Then we propose Nested LSTM to forecast the category of next POI where a user is interested in. Thanks to Nested LSTM, we now can answer the two motivational questions: "Is there any pattern in location-based, mobile check-in activities?" "If yes, is it possible to forecast a user's next check-in intention, given his/her check-in history?" Experimental results show that, Nested LSTM achieves *Accuracy@5* about 48.96% and 68.20% on Foursquare and Jiepang datasets, respectively. The taxonomy embedding learned by Nested LSTM achieves *Silhouette Coefficient* of 0.034 which outperforms other state-of-the-art approaches. Inner LSTM captures the hierarchical relationship of categories on taxonomy (parent category—child category), and has the capability of better clustering child category. Finally, a practitioners' guide is provided for deploying Nested LSTM to predict the next POI category.

REFERENCES

[1] 2018. Foursquare website. (2018). https://foursquare.com/.
[2] 2018. Jiepang website. (2018). https://jiepang.com/.
[3] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks* 5, 2 (1994), 157–166.
[4] Jonathan Chang and Eric Sun. 2011. Location3: How Users Share and Respond to Location-Based Data on Social. (2011).
[5] Huiji Gao, Jiliang Tang, and Huan Liu. 2012. Exploring Social-Historical Ties on Location-Based Social Networks. (2012).
[6] Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, and John Riedl. 1999. An Algorithmic Framework for Performing Collaborative Filtering. In *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '99)*. ACM, 230–237. https://doi.org/10.1145/312624.312682
[7] Sepp Hochreiter. 1998. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 6, 02 (1998), 107–116.
[8] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.
[9] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (Aug. 2009), 30–37. https://doi.org/10.1109/MC.2009.263
[10] Defu Lian and Xing Xie. 2011. Collaborative activity recognition via check-in history. In *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Location-Based Social Networks*. ACM, 45–48.
[11] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Predicting the Next Location: A Recurrent Model with Spatial and Temporal Contexts. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI'16)*. AAAI Press, 194–200.
[12] Anastasios Noulas, Cecilia Mascolo, and Enrique Frias-Martinez. 2013. Exploiting foursquare and cellular data to infer user activity in urban environments. In *Mobile Data Management (MDM), 2013 IEEE 14th International Conference on*, Vol. 1. IEEE, 167–176.
[13] Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. 2016. Deep Sentence Embedding Using Long Short-term Memory Networks: Analysis and Application to Information Retrieval. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.* 24, 4 (April 2016), 694–707.
[14] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.
[15] Fabio Pianese, Xueli An, Fahim Kawsar, and Hiroki Ishizuka. 2013. Discovering and predicting user routines by differential analysis of social network traces. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2013 IEEE 14th International Symposium and Workshops on a*. IEEE, 1–9.
[16] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based Collaborative Filtering Recommendation Algorithms. In *Proceedings of the 10th International Conference on World Wide Web (WWW '01)*. ACM, 285–295. https://doi.org/10.1145/371920.372071
[17] Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075* (2015).
[18] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3156–3164.
[19] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J. Smola, and How Jing. 2017. Recurrent Recommender Networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. 495–503. http://dl.acm.org/citation.cfm?id=3018689
[20] Dingqi Yang, Daqing Zhang, Vincent W Zheng, and Zhiyong Yu. 2015. Modeling user activity preference by leveraging user spatial temporal characteristics in LBSNs. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 45, 1 (2015), 129–142.
[21] Jihang Ye, Zhe Zhu, and Hong Cheng. 2013. What's your next move: User activity prediction in location-based social networks. In *Proceedings of the 2013 SIAM International Conference on Data Mining*. SIAM, 171–179.

[22] Quan Yuan, Gao Cong, Zongyang Ma, Aixin Sun, and Nadia Magnenat Thalmann. 2013. Time-aware Point-of-interest Recommendation. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '13)*. ACM, 363–372. https://doi.org/10.1145/2484028.2484030

[23] Zhiqian Zhang, Chenliang Li, Zhiyong Wu, Aixin Sun, Dengpan Ye, and Xiangyang Luo. 2017. NEXT: A Neural Network Framework for Next POI Recommendation. *CoRR* abs/1704.04576 (2017). arXiv:1704.04576

[24] Shenglin Zhao, Irwin King, and Michael R. Lyu. 2016. A Survey of Point-of-interest Recommendation in Location-based Social Networks. *CoRR* abs/1607.00647 (2016). arXiv:1607.00647 http://arxiv.org/abs/1607.00647

[25] Yu Zhu, Hao Li, Yikang Liao, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. 2017. What to Do Next: Modeling User Behaviors by time-LSTM. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17)*. AAAI Press, 3602–3608.