Adversarial Learning to Compare: Self-Attentive Prospective Customer Recommendation in Location based Social Networks

Ruirui Li
UCLA
rrli@cs.ucla.edu

Xian Wu
University of Notre Dame
xwu9@nd.edu

Wei Wang
UCLA
weiwang@cs.ucla.edu

ABSTRACT

Recommendation systems tend to suffer severely from the sparse training data. A large portion of users and items usually have a very limited number of training instances. The data sparsity issue prevents us from accurately understanding users' preferences and items' characteristics and jeopardize the recommendation performance eventually. In addition, models, trained with sparse data, lack abundant training supports and tend to be vulnerable to adversarial perturbations, which implies possibly large errors in generalization.

In this work, we investigate the recommendation task in the context of prospective customer recommendation in location based social networks. To comprehensively utilize the training data, we explicitly learn to compare users' historical check-in businesses utilizing self-attention mechanisms. To enhance the robustness of a recommender system and improve its generalization performance, we perform adversarial training. Adversarial perturbations are dynamically constructed during training and models are trained to be tolerant of such nuisance perturbations. In a nutshell, we introduce a Self-Attentive prospective Customer RecommendAtion framework, SACRA, which learns to recommend by making comparisons among users' historical check-ins with adversarial training. To evaluate the proposed model, we conduct a series of experiments to extensively compare with 12 existing methods using two real-world datasets. The results demonstrate that SACRA significantly outperforms all baselines.

CCS CONCEPTS

• **Information systems** → **Location based services**; **Geographic information systems**.

KEYWORDS

Customer recommendation; self-attention; adversarial training; pairwise ranking

ACM Reference Format:
Ruirui Li, Xian Wu, and Wei Wang. 2020. Adversarial Learning to Compare: Self-Attentive Prospective Customer Recommendation in Location based Social Networks. In *The Thirteenth ACM International Conference on Web Search and Data Mining (WSDM '20), February 3–7, 2020, Houston, TX, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3336191.3371841

WSDM '20, February 3–7, 2020, Houston, TX, USA

ACM ISBN 978-1-4503-6822-3/20/02…$15.00
https://doi.org/10.1145/3336191.3371841

1 INTRODUCTION

Location-based social networks (LBSNs), such as Yelp, Foursquare, and Instagram, attract millions of users to share their locations, resulting in a huge amount of user check-in data. The availability of such abundant user check-ins brings in great opportunities to understand users' preferences and help businesses achieve profit growth by identifying prospective new customers. However, prospective customer predictions in LBSNs suffer severely from data sparsity. The sparse check-in data prevents us from comprehensively understand customers' preferences and makes prospective customer identification a daunting task.

Most existing works usually address the issues caused by data sparsity through incorporating various ancillary information, such as geographical influence, social correlations, temporal patterns, textual and visual contents [4, 9, 16, 18, 24, 29, 31]. Although the ancillary information, as extra knowledge, has been proven to be effective for improving recommendation performances, it is not always available, especially for those new businesses and customers, of which, we have limited information. Therefore, a good recommender system should not only rely on embracing ancillary information, but also seek more suitable techniques to enhance the recommendation.

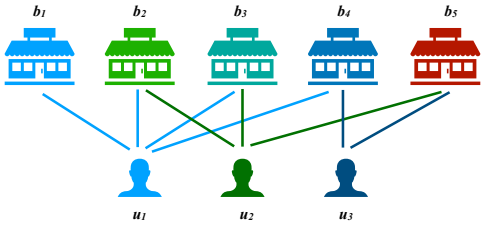


Figure 1: A motivating example

In contrast to previous methods, which focus on embracing ancillary information to improve recommendation performance, we explicitly refer to a user's historical check-in businesses with different attentions when inferring whether the user will check in a new business. Figure 1 shows a motivating example. There are five businesses b_1, b_2, b_3, b_4, and b_5 as well as three users u_1, u_2, and u_3. The check-ins are indicated by a line connection between a business and a user. Business b_1 has only one check-in user u_1. Therefore, it is relatively difficult to infer the characteristic of b_1 comparing to other businesses. However, u_1 has multiple check-in businesses. By referring b_1 to other check-in businesses of u_1, we could infer that the characteristic of b_1 is likely to be a mixture of the characteristics of b_2, b_3, and b_4 with different weights. Since b_2, b_3, and b_4 have relatively more check-in users, their characteristics learnt are relatively stable and reliable. By explicitly referring to

such related check-in businesses, it benefits the characteristic learning of businesses with fewer check-ins, such as b_1, and improves the recommendation performance, especially for those tail businesses.

Recent advances on adversarial machine learning [22] show that many state-of-the-art models are actually very fragile and vulnerable to adversarial examples, which are formed by applying small but intentional perturbations to inputs from the dataset. This is particularly true for tasks where training data is highly sparse [6, 11, 12]. Training models achieving excellent performance generally requires a sufficient amount of training data, especially for most neural network based models. However, as mentioned above, the training data are very sparse in recommendation tasks. Thus, the users and businesses, with low probabilities in the data distribution, lack abundant training supports. Therefore, there tends to be a great performance gap between training and test. The trained model can be very sensitive to unseen perturbations. In general, human perception and cognition are robust to a vast range of nuisance perturbations. However, neural networks are currently far from achieving the same level of tolerance of such adversarial perturbations [22]. This motivates us to also investigate the robustness property of modern recommender systems and propose an appropriate method to achieve high tolerance. To increase the robustness of the training model, we dynamically construct perturbations at the embedding level to form adversarial examples. The model, trained in the adversarial manner, not only learns from the original static training data, but also improves based on the dynamically constructed perturbed data. This allows the training model resistant of nuisance perturbations and achieve high generalizations in both training and test.

In this paper, we study the problem of prospective customer recommendation in LBSNs. In essence, we address the check-in sparsity issue by referring to users' historical check-ins. We enhance the model generalization and robustness by incorporating dynamically constructed adversarial examples. To be more specific, the main contributions of this work are as follows:

- Different from conventional methods, which directly rely on the representations of a user and a target business to infer the check-in tendency of the user to the target business, we explicitly refer to the user's historical check-in businesses as references. We investigate the differences between the target business and these reference businesses. The differences are then utilized to infer the check-in tendency.
- To differentiate the historical check-in businesses of a user, we introduce multiple self-attention mechanisms. Each check-in business of a user is explained with attention to all his/her historical check-in businesses. Therefore, business features get fused and comprehensively characterize the check-in behavior of a user, leading to the eventual improvement of the overall recommendation performance, especially for those tail businesses.
- The model proposed also focuses on improving the resistance to nuisance perturbations. This increases the generalization of the training data, reduces the performance gap between training and test, and provides favorable rankings of suggestions robustly.
- We present a comprehensive empirical evaluation of our approach against 12 recommendation methods on two real-world datasets. The results show that our approach, SACRA, outperforms all baseline methods in suggesting prospective customers for businesses in different cities.

2 PROBLEM STATEMENT

The task of prospective customer recommendation is to rank candidate customers given a business. The goal is to rank the true incoming new customers higher than other candidates. The candidates here are all customers who have not checked in this business before.

Table 1: List of symbols

Symbol	Description
b	a business
u	a user
B	a set of businesses
$\boldsymbol{E}_b$	the embedding of business b
$\boldsymbol{E}_u$	the embedding of user u
$\tilde{\boldsymbol{E}}_B$	the fused embeddings of B
$\bar{\boldsymbol{E}}_B$	the representative embedding of B
$\boldsymbol{W}$ & $\boldsymbol{b}$	network weight and bias
α	importance weight in the attention mechanism
ℓ	loss function
η	learning rate
ϵ	perturbation bound
λ	regularizer weight
Θ	ranking model parameters
Δ	parameter perturbations
Ψ	Gaussian mixture model parameters
$\mathbf{l}$	business location vector
M	number of Gaussian components

To better explain the proposed method, we list the main notations we use in this paper in Table 1.

3 METHODOLOGY

Given a business b and a user u, we aim to derive a check-in likelihood $t(b, u)$ to estimate user u's tendency of checking in the business b. Figure 2 illustrates the process of such check-in tendency calculations. For example, to calculate a user u^+'s check-in tendency to a business b, we refer to a selection of u^+'s sampled historical check-in businesses, denoted as B^+. For each of the sampled businesses, we use a fixed-length embedding vector to represent its characteristic. To complement each check-in business with others, we feed the business embeddings into a multi-head fusion network, the output of which yields the fused business representations, denoted as $\tilde{\boldsymbol{E}}_B$. We further feed $\tilde{\boldsymbol{E}}_B$ into an aggregation attention network to derive the aggregated representative business embedding, denoted as $\bar{\boldsymbol{E}}_B$. We utilize this representative, together with the target business features $\boldsymbol{E}_b$, the user features $\boldsymbol{E}_{u^+}$, and the geographical features $\boldsymbol{E}_{g(b,u^+)}$ to calculate the check-in tendency by going though a check-in tendency network. As shown in Figure 2, instead of directly estimating the check-in tendency of a single user. We employ a pairwise network optimization strategy, where a pair of users u^+ and u^- is constructed regarding a business b. Here u^+ represents a user who did check in business b, while u^- represents a user who did not check in b, where u^- can be constructed by negative sampling. We calculate the check-in tendencies of both

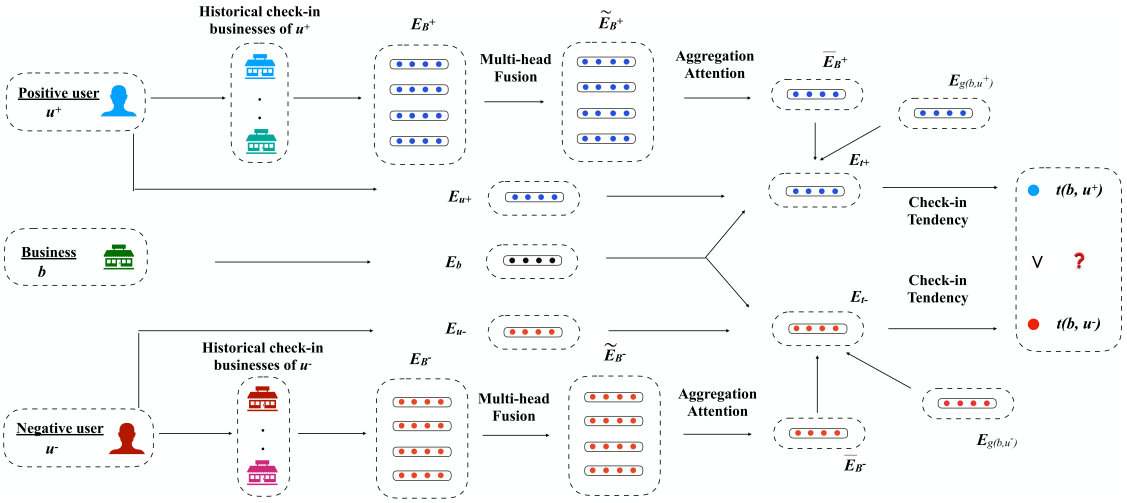


Figure 2: Framework of SACRA

users, the characteristic learning of business b gets optimized by correcting the check-in tendency orders of many such constructed user pairs. In this following paragraphs, we explain the multi-head fusion network, the aggregation attention network, the check-in tendency network, and the pairwise network optimization strategy in detail sequentially.

3.1 Multi-head Fusion Network

Each business in B is represented by a fused business embedding. The fused business embedding encodes the representation with attention to itself and the other businesses in B. In other words, we attempt to use the business itself and the other related businesses to explain each business in B. To achieve this, we develop a multi-head fusion module, which utilizes a self-attention mechanism. The scaled dot-product attention is defined as:

$$\text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}(\frac{\boldsymbol{Q} \cdot \boldsymbol{K}^T}{\sqrt{d_Q}})\boldsymbol{V}, \tag{1}$$

where $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$ represent the attention query, attention key, and attention value matrices, respectively. The scale factor $\sqrt{d_Q}$ is used to avoid overly large values of the inner product, where d_Q is the feature dimension of both $\boldsymbol{Q}$ and $\boldsymbol{K}$. We use multi-head attention with k heads, as introduced in [23],

$$\text{MultiHeadAttention}(\boldsymbol{H}) = \text{Concat}(\text{head}_1, \ldots, \text{head}_k)\boldsymbol{W}^O, \tag{2}$$

where

$$\text{head}_j = \text{Attention}(\boldsymbol{H} \cdot \boldsymbol{W}_j^Q, \boldsymbol{H} \cdot \boldsymbol{W}_j^K, \boldsymbol{H} \cdot \boldsymbol{W}_j^V), \tag{3}$$

with different projections using trainable parameter matrices $\boldsymbol{W}^Q$, $\boldsymbol{W}^K$, and $\boldsymbol{W}^V \in \mathbb{R}^{d \times d/k}$. $\boldsymbol{W}^O \in \mathbb{R}^{d \times d}$ is the transformation matrix to model the correlations among head-specific embeddings.

The attention operation calculates a weighted sum of all values, where the weight between query i and value j relates to the interaction between query i and key j. The multi-head attention allows the model jointly attend to information from different representation spaces.

In our case, the multi-head attention operation takes the business embeddings $\boldsymbol{E}_B \in \mathbb{R}^{c \times d}$ as the inputs and feeds them into different head attention layers, the results of which are further concatenated as the final output:

$$\tilde{\boldsymbol{E}}_B = \text{MultiHeadAttention}(\boldsymbol{E}_B). \tag{4}$$

The multi-head self-attention result $\tilde{\boldsymbol{E}}_B$ learns the embeddings of businesses by comparing the pairwise closeness between businesses. Each fused business embedding $\tilde{\boldsymbol{E}}_{B_i}$ is a weighted sum of business embedding of itself and other related businesses, where each weight gauges the similarity between one business and another one in B. In this way, $\tilde{\boldsymbol{E}}_B$ encodes the fused business information, with each one explained by itself and others. This allows the information among a user's check-in businesses get fused and complement each other.

To increase the non-linearity of the self-attention mechanism, we further feed the fused business embeddings into a feed-forward neural network:

$$\tilde{\boldsymbol{E}}_B^f = \boldsymbol{W}_2^f \cdot \text{ReLU}(\boldsymbol{W}_1^f \cdot \tilde{\boldsymbol{E}}_B + \boldsymbol{b}_1^f) + \boldsymbol{b}_2^f, \tag{5}$$

where $\boldsymbol{W}_1^f$, $\boldsymbol{W}_2^f$, and $\boldsymbol{b}_2^f$, $\boldsymbol{b}_2^f$ are the weight matrices and bias in the feed-forward layer. To comprehensively fuse the business information among a user's historical check-ins, we perform the multi-head attention operations twice.

3.2 Aggregation Attention Network

As shown in Figure 2, we aim to compare the target business b with a user's historical check-in businesses B. To further summarize the characteristic relevance between compared businesses, we develop an aggregation attention network to learn the importance weights across compared businesses. Formally, the fusion attention module can be represented as follows:

$$\alpha_i = \text{softmax}(\boldsymbol{c} \cdot \text{Tanh}(\boldsymbol{W}^a \cdot \tilde{\boldsymbol{E}}_{B_i}^f + \boldsymbol{b}^a)). \tag{6}$$

$$\bar{\boldsymbol{E}}_B = \sum_i \alpha_i \tilde{\boldsymbol{E}}_{B_i}^f. \tag{7}$$

Each fused embedding $\tilde{\boldsymbol{E}}_{B_i}^f$ is first fed into a one-layer neural network. Its output, together with the context vector $\boldsymbol{c}$, are further utilized to generate the importance weight α_i for each fused business embedding $\tilde{\boldsymbol{E}}_{B_i}^f$ through a softmax function. The aggregated

embedding $\bar{E}_B$ is calculated as a weighted sum of the fused business embeddings based on the learned importance weights.

3.3 Check-in Tendency Network

To model the check-in tendency between user u and business b, we compare the similarity between the embedding $\boldsymbol{E}_b$ of the investigated business b and the representative fused business embedding $\bar{\boldsymbol{E}}_B$.

The similarity is represented by $\boldsymbol{E}_b \bullet \bar{\boldsymbol{E}}_B$, where $\bullet$ represents the element-wise multiplication. The similarity is then concatenated with the user embedding $\boldsymbol{E}_u$, the geographical features $\boldsymbol{E}_{g(b,u)}$, and fed into a one-layer neutral network to calculate the check-in tendency. Formally,

$$\boldsymbol{E}_t = (\boldsymbol{E}_b \bullet \bar{\boldsymbol{E}}_B)||\boldsymbol{E}_u||\boldsymbol{E}_{g(b,u)}, \tag{8}$$

where $||$ represents the embedding concatenation operation and $\boldsymbol{E}_{g(b,u)}$ gives the geographical influence features, which will be discussed in section 3.5.

$$t_{b,u}(\Theta) = \text{Sigmoid}(\boldsymbol{W}^t \cdot \boldsymbol{E}_t + \boldsymbol{b}^t), \tag{9}$$

where $\boldsymbol{W}^t$ and $\boldsymbol{b}^t$ are the weight and bias in the check-in tendency module, respectively. Θ is a set of model parameters.

Pairwise Network Optimization Strategy. To achieve good ranking performance, we apply a pairwise ranking scheme to measure the ranking error defined on a business b, as well as two users u^+ and u^-:

$$\ell(b, u^+, u^-|\Theta) = \log(1 + \exp(t_{b,u^-}(\Theta) - t_{b,u^+}(\Theta))), \tag{10}$$

where u^+ represents a positive user who did check in b, u^- represents a negative user who did not check in b. The loss of the model is further defined as a sum over all training instances:

$$L(D|\Theta) = \sum_{(b,u^+,u^-)\in D} \ell(b, u^+, u^-|\Theta), \tag{11}$$

where D denotes the set of pairwise training instances.

3.4 Adversarial Training

The goal of applying adversarial training is to make the recommender system not only suitable for ranking, but also robust to adversarial perturbations. To ensure high-quality ranking, we utilize pairwise ranking scheme, shown in Equation 10, as the building block. To enhance the robustness, we enforce the model to perform consistently well even when the adversarial perturbations are presented. To achieve this goal, we additionally optimize the model to minimize the objective function with the perturbed parameters. Formally, we define the objective function with adversarial examples incorporated as:

$$\begin{aligned} L_{adv}(D|\Theta) &= L(D|\Theta) + \lambda L(D|\Theta + \Delta_{adv}), \\ \text{where } \Delta_{adv} &= \arg\max_{\Delta, \|\Delta\| \le \epsilon} L(D|\hat{\Theta} + \Delta), \end{aligned} \tag{12}$$

where Δ denotes the perturbations on model parameters, $\epsilon \ge 0$ controls the magnitude of the perturbations, and $\hat{\Theta}$ denotes the current model parameters. In this formulation, the adversarial term $L(D|\Theta+\Delta_{adv})$ can be treated as a model regularizer, which stabilizes the ranking performance. λ is introduced to control the strength of the adversarial regularizer, where the intermediate variable Δ maximizes the objective function to be minimized by Θ. The training process can be summarized as playing a minimax game:

$$\Theta_{opt}, \Delta_{opt} = \arg\min_{\Theta} \max_{\Delta, \|\Delta\| \le \epsilon} L(D|\Theta) + \lambda L(D|\Theta + \Delta), \tag{13}$$

where the optimizer for the model parameters Θ acts as the minimizing player while the procedure to derive dynamic perturbations Δ acts as the maximizing player. The maximizing player strives to construct the worst-case perturbations against the current model. The two players alternately play the minmax game until convergence.

Constructing Adversarial Perturbations. Given a training instance (b, u^+, u^-), the problem of constructing adversarial perturbations Δ_{adv} is formulated as maximizing

$$\ell_{adv}(b, u^+, u^-|\Delta) = \log(1 + \exp(t_{b,u^-}(\hat{\Theta} + \Delta) - t_{b,u^+}(\hat{\Theta} + \Delta))), \tag{14}$$

where $\hat{\Theta}$ denotes a set of current model parameters. As it is difficult to derive the exact optimal solutions of Δ_{adv}, we apply the fast gradient method proposed in [5] to estimate Δ_{adv}, where we approximate the objective function around Δ as a linear function. To maximize the approximated linear function, we move towards the gradient direction of the objective function with respect the Δ. With the max-norm constraint $\|\Delta\| \le \epsilon$, we approximate Δ_{adv} as:

$$\Delta_{adv} = \epsilon \frac{\tau}{\|\tau\|}, \text{ where } \tau = \frac{\partial \ell_{adv}(b, u^+, u^-|\Delta)}{\partial \Delta}. \tag{15}$$

Learning Model Parameters. We now explain how to learn model parameters Θ. The local objective function to minimize for a training instance (b, u^+, u^-) is as follows:

$$\begin{aligned} \ell_{adv}(b, u^+, u^-|\Theta) &= \log(1 + \exp(t_{b,u^-}(\Theta) - t_{b,u^+}(\Theta))) \\ &+\lambda \log(1 + \exp(t_{b,u^-}(\Theta + \Delta_{adv}) - t_{b,u^+}(\Theta + \Delta_{adv}))), \end{aligned} \tag{16}$$

where Δ_{adv} is obtained from Equation 15. We can obtain the SGD update rule for Θ:

$$\Theta = \Theta - \eta \frac{\partial \ell_{adv}(b, u^+, u^-|\Theta)}{\partial \Theta}, \tag{17}$$

where η denotes the learning rate.

Algorithm 1: Parameter optimizations

Input: Training instances D, max iteration $iter_{\max}$;
Output: Model parameters Θ
Initialization: initialize Θ with Normal distribution $N(0,0.01)$, $iter = 0$, $\Theta_{opt} = \Theta$, $L_{opt} = L_{vali}$;
repeat
  foreach *training instance* $(b, u^+, u^-) \in D$ **do**
    // Constructing adversarial perturbations;
    $\Delta_{adv} \leftarrow$ Equation 15;
    // Updating model parameters;
    $\Theta \leftarrow$ Equation 17;
  if $L_{vali} < L_{opt}$ **then**
    $L_{opt} = L_{vali}$;
    $\Theta_{opt} = \Theta$;
  $iter++$;
until $iter > iter_{max}$;
Return Θ_{opt};

Algorithm 1 summarizes the training process. In each training step, we first randomly draw an instance (b, u^+, u^-). We then construct adversarial perturbations and optimize model parameters in a sequential order. The training involves multiple training steps and stops until reaching a certain number of training epochs. The parameters achieving the best performance on the validation dataset are utilized for evaluations.

3.5 Geographical Influence

In this section, we follow [10] and explain how to derive the geographical features $\boldsymbol{E}_{g(b,u)}$ given a business b and a user u.

We employ the Gaussian mixture model (GMM) [20] to generate the geographical features. A Gaussian mixture model is a weighted sum of M component Gaussian densities:

$$p(\mathbf{l}|\Psi) = \sum_{m=1}^{M} \beta_m f(\mathbf{l}|\mu_m, \Sigma_m), \tag{18}$$

where $\mathbf{l}$ is a 2-dimensional vector representing the latitude and longitude of a business, β_m is the mixture weight, and $f(\mathbf{l}|\mu_m, \Sigma_m)$ are the component Gaussian densities. Each component density is a 2-variate Gaussian function of the form,

$$f(\mathbf{l}|\mu_m, \Sigma_m) = \frac{1}{2\pi|\Sigma_m|^{1/2}} e^{-\frac{1}{2}(\mathbf{l}-\mu_m)'\Sigma_m^{-1}(\mathbf{l}-\mu_m)},$$

where μ_m and Σ_m are mean location vector and covariance matrix, respectively. The complete Gaussian mixture model is parameterized by the mean location vectors, covariance matrices, and mixture weights from all component densities. These parameters are further collectively notated by Ψ. For a particular customer, given all his/her check-in locations, represented by N location vectors $L = \{\mathbf{l}_1, .., \mathbf{l}_N\}$, the GMM likelihood is written as:

$$p(L|\Psi) = \prod_{n=1}^{N} p(\mathbf{l}_n|\Psi).$$

We use the Expectation-Maximization [3] algorithm to optimize the parameters. Due to the space limit, we skip the optimization details. After the GMM construction for a customer u, given the geographical location l_b of a business b, as shown in Equation 18, $p(\mathbf{l}_b|\Psi)$ gives the geographical features for a pair of business b and user u.

4 EXPERIMENTS

In this section, we conduct extensive experiments on two real-world datasets to evaluate the performance of SACRA.

4.1 Datasets and Experimental Settings

The experiments are conducted on two datasets. One is Yelp challenge dataset and the other is the Foursquare dataset. For the Yelp dataset, we investigate the recommendation tasks in six large cities. The Foursquare dataset contains interactions between customers and businesses in Los Angeles and New York. Table 2 shows the statistics for the eight cities in the two datasets.

We follow the data cleaning strategy in [7, 10] and filter out businesses and customers whose check-ins are less than 20 for the Yelp dataset. For the Foursquare dataset, we follow the cleaning steps in [1] and remove business and customers that have less than 8 check-ins. For each business, its check-ins are sorted in chronological order based on the timestamps. The first 50% of the check-ins are used as the training data. The following 20% are used for validation and the remaining 30% are used as the test data for evaluation. Table 3 shows the main parameters and their default values to tune in the experiments.

4.2 Baselines

To compare our approach with others, the following 12 methods are adopted as baselines.

Traditional matrix factorization (MF) methods:

- **WRMF**, weighted regularized matrix factorization [8] minimizes the square error loss by assigning both observed and fake check-ins with different weights based on matrix factorization.
- **MMMF**, maximum margin matrix factorization [26] minimizes the hinge loss based on matrix factorization.
- **BPRMF**, Bayesian personalized ranking matrix factorization [19] optimizes Area Under the Curve based on pairs of observed and sampled fake check-ins.
- **CofiRank**, [25] optimizes the estimation of a ranking loss based on Normalized Discounted Cumulative Gain.
- **CLiMF**, [21] optimizes a different ranking-oriented loss, i.e., Mean Reciprocal Rank loss.

Classical Point-of-Interest (POI) recommendation methods:

- **USG**, [32] is a collaborative filtering method. It utilizes social and geographical information to improve recommendations.
- **GeoMF**, [14] is a geographically weighted matrix factorization model.
- **Rank-GeoFM**, ranking-based geographical factorization [13] incorporates geographical and temporal information to provide recommendations.
- **ASMF**, [9] utilizes geographical information, social information, and attributes of businesses to enhance the accuracy of recommendations.
- **ARMF**, [9] extends ASMF by applying ranking losses.
- **CORALS**, [10] incorporates geographical influence and textual contents to enhance recommendations.

Deep learning-based methods:

- **SAE-NAD**, self-attentive encoder and neighbor-aware decoder [18] applies auto-encoders to make recommendations.

Among these 12 baseline methods, WRMF is a point-wise matrix factorization method while MMMF and BPRMF are pair-wise based. CofiRank, CLiMF focus on optimizing top ranked positions. USG, GeoMF, Rank-GeoFM, ASMF, ARMF, and CORALS utilize additional information, such as check-in locations, social relationship, businesses' attributes, online reviews and temporal information to improve recommendation performance in LBSNs. SAE-NAD utilizes auto-encoders, with business neighborhood information considered, to make recommendations. All parameters in baselines are best tuned based on their guidelines.

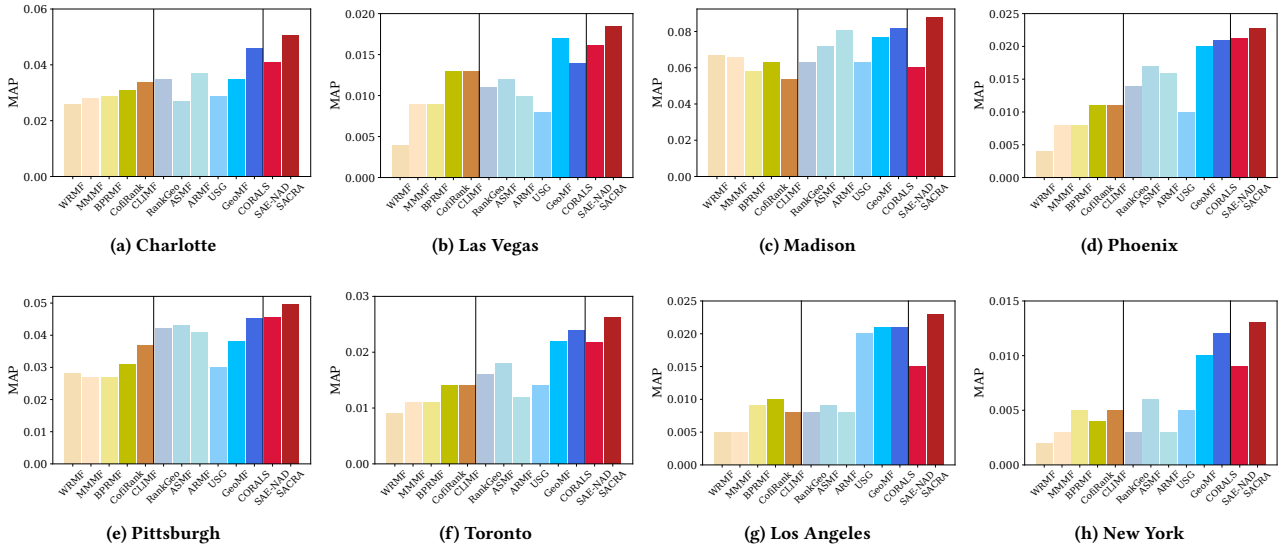
4.3 Recommendation Performance

In this section, we evaluate the performances of SACRA against the 12 baseline methods. Mean Average Precision (MAP) is adopted as the evaluation metric, which is also used in [9, 10].

Figure 3 shows the recommendation performances of different methods on the eight cities from the two datasets. Figures from 4a to 4f show the performances based on the six cities in the Yelp dataset, while the last two Figures 4g and 4h show the performances based on the two cities in the Foursquare dataset. To distinguish different types of methods, we further draw two vertical lines. The conventional MF based methods are shown in the left part, the classical POI based methods are in the middle, and the neural network based models are shown in the right segment.

Table 2: Business and customer statistics

Dataset	Yelp						Foursquare	
City	Charlotte	Las Vegas	Madison	Phoenix	Pittsburgh	Toronto	Los Angeles	New York
# of Customers	69,005	432,399	26,083	314,610	51,422	58,377	501,940	717,382
# of Businesses	10,652	282,204	3,895	43,482	8,037	20,849	215,614	206,416



(a) Charlotte (b) Las Vegas (c) Madison (d) Phoenix

(e) Pittsburgh (f) Toronto (g) Los Angeles (h) New York

Figure 3: MAP performances of different methods over eight cities

Table 3: Main Parameters

Parameters	Value	Parameters	Value
Learning rate η	0.001	Number of epochs	20
Batch size	64	Perturbation bound ϵ	0.5
User feature dimension	64	Regularizer weight λ	1
Business feature dimension	64	Number of attention heads	2
Number of businesses to compare	20	Context vector dimension	10

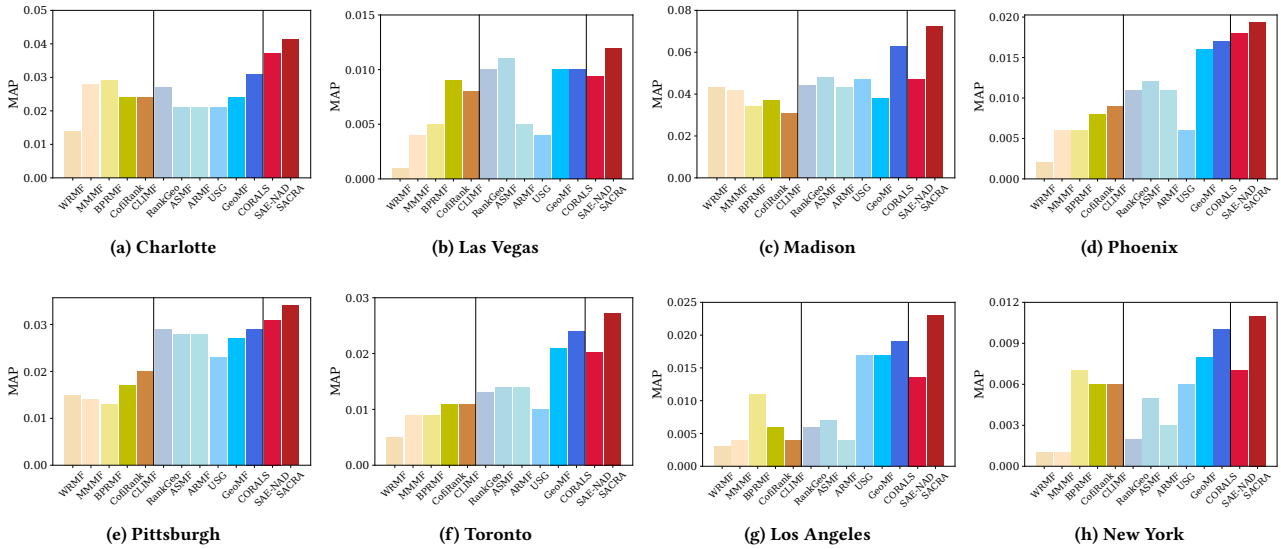
Among conventional MF based methods, MMMF, BPRMF, CofiRank, and CLiMF achieve better recommendation performances than WRMF in general. This demonstrates the advantage of employing ranking-based optimization strategies. In other words, directly optimizing the predicted check-ins may not always provide the best recommendation lists to businesses.

POI based and neutral network based methods, i.e., Rank-GeoMF, ASMF, ARMF, USG, GeoMF, CORALS, and SAE-NAD, with geographical features incorporated, outperform WRMF, MMF, BPRMF, CofiRank, and CLiMF in general. It proves that utilizing ancillary information compensates for the sparsity issue in location-based recommendation tasks. The performance of Rank-GeoFM is not as good as the one of GeoMF. This is because Rank-GeoFM, which incorporates temporal information, intends to predict the next POI to visit, while the task in this work is to predict new customers for POIs. GeoMF and CORALS achieve better MAP than ASMF and ARMF. This might be because ASMF and ARMF focus on utilizing social information, while learning geographical influence might be a better choice to improve recommendation performances in location-based tasks. SAE-NAD achieves relatively worse MAP performances than CORALS. This mainly results from the rich ancillary information incorporated in CORALS. SACRA further outperforms all baseline methods. SACRA learns through comparing and fusing users' historical check-ins and utilizes attention mechanisms to differentiate the importance of check-in businesses when making recommendations. These appropriate settings make SACRA a good fit for new user recommendations in LBSNs.

In addition, we further investigate the performance of different models on tail (10%) businesses over the eight cities. Businesses are sorted by their check-in numbers, and tail businesses are the ones that have fewer user check-ins. Figure 4 shows the corresponding performances. Compared with the performances shown in Figure 3, we observe that the performances on the tail businesses are worse than the performances on the overall businesses. This makes senses since the less check-in information we have for a business, the less guidance we have to learn the characteristic of the businesses. Among all the baseline methods, SACRA still outperforms all baseline methods. This contributes from the adoption of explicit comparisons among users' historical check-ins and the dynamic attention mechanisms.

4.4 Effectiveness of geographical features

In this work, we utilize GMM to model the geographical convenience of a visit from a user to a business. In this section, we conduct the ablation study by removing the GMM component. After removing the GMM component, SACRA does not utilize geographical



(a) Charlotte (b) Las Vegas (c) Madison (d) Phoenix

(e) Pittsburgh (f) Toronto (g) Los Angeles (h) New York

Figure 4: MAP performances of different methods on tail businesses over eight cities

features any more. We denote this variant of SACRA as SARCA$_g$. We compare the performance of SACRA with the one of SARCA$_g$ over the eight cities. Figure 5 shows the relative MAP improvements of SARCA against SARCA$_g$.

We observe that the performances of SARCA always outperform SARCA$_g$ by at least 30%. This demonstrates the effectiveness of incorporating geographical features via GMM in location based recommendation systems. The geographical convenience, modeled by GMM, well captures the relative effort of a visit from a user to a business from the perspective of geographical influence. The geographical convenience, different from conventional distance based metrics [2, 18, 32], not only captures the physical efforts of a visit, but also accurately distinguishes customers with different traveling flexibility. For example, the difference of a business can be the same to two customers, with one relying on walking and the other tending to drive, the actual efforts to travel there can be significantly different. The geographical convenience, modeled by GMM, exactly captures such relative efforts and contributes to the dramatic MAP improvements.

4.5 Effectiveness of explicit attentive comparisons

Learning from comparisons among users' check-ins is a key component in the proposed model. To evaluate the effectiveness of incorporating comparisons, we remove the multi-head fusion and the aggregation attention modules from SACRA. In other words, we directly use the business embedding, the user embedding, and the geographical features to estimate the user check-in tendencies. We denote this variant of SACRA as SACRA$_\alpha$. We perform both SACRA and SACRA$_\alpha$ on the two datasets and observe that SACRA always outperforms SACRA$_\alpha$ in terms of the MAP performance. Figures 6 shows the relative MAP improvements of SACRA against SACRA$_\alpha$ on the eight cities. We notice that there is at least 10% relative performance improvements after incorporating the multi-head fusion and the aggregation attention modules. The multi-head fusion module fuses the business features and characterizes related check-in businesses more comprehensively. The aggregation attention module adaptively yields the representative of a user's check-in businesses and enhances the dynamic comparisons between a target business and related historical check-in businesses. These two modules leads to the performance improvements when making recommendations in SACRA.

4.6 Effectiveness of adversarial learning

In this section, we show the advantage of adopting the adversarial training of SACRA. In SACRA, adversarial perturbations are dynamically generated and applied to business and user embeddings. To conduct the ablation study, we denote SACRA without adversarial training as SACRA$_a$.

Figure 7 shows the relative MAP improvements of SACRA against SACRA$_a$. We observe that with adversarial training, SACRA always outperforms SACRA$_a$ by at least 5%. This demonstrates that learning from unseen tough user and business interaction instances, generated by adversarial training, successfully shrinks the performance gap between training and test.

4.7 Sensitivity Study

In this section, we examine how different choices of parameters influence the performance of SACRA. Except for the parameter being tested, we set other parameters at the default values (see in Table 3). Figure 8 shows the evaluation results as a function of one selected parameter when fixing others. Overall, we observe that SACRA is not strictly sensitive to these parameters, which demonstrates the robustness of SACRA.

Figure 8a shows that he MAP performances of SACRA when we change the learning rate. It may get stuck to local optimal and lead to sub-optimal performance when the learning rate is either

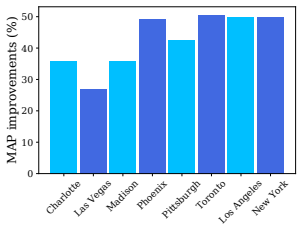


Figure 5: Relative MAP improvements after incorporating the geographical influence

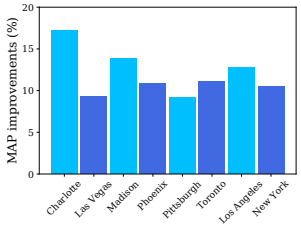


Figure 6: Relative MAP improvements after incorporating the fusion and aggregation modules

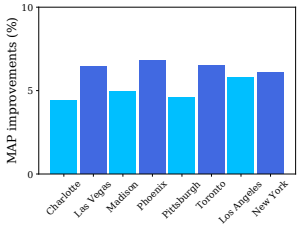


Figure 7: Relative MAP improvements after adopting adversarial training

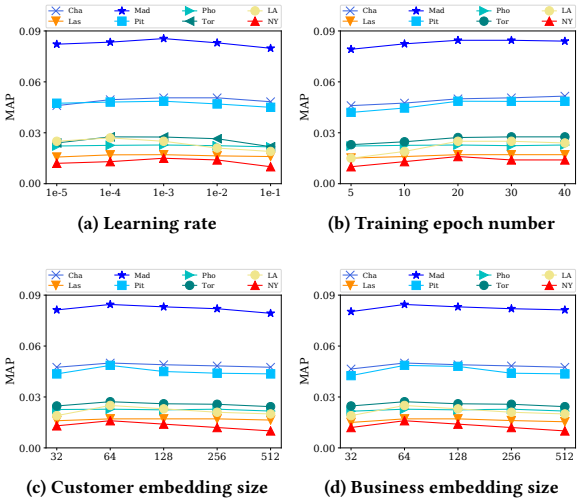


Figure 8: Parameter sensitivity studies

too small or too large. In this work, we set it as $1e-3$ with the consideration of the performance. Figure 8b shows the effect of the number of training epochs. SACRA is not well trained when the training epoch number is set to 5 and 10, leading to relatively poor MAP performances. As we increase the epoch number to 20 or larger, the performances increase correspondingly. In Figures 8c and 8d we observe that the increase of recommendation performance saturates as the representation dimensionality increases more than 64. This is because: at the beginning, a larger value of embedding dimension brings a stronger representation power to express customer preference and business characteristic, but the further increase of dimension size of latent representations might lead to the overfitting issue. In our experiments, we set the dimension size as 64 due to the consideration of the performance and computational cost.

5 RELATED WORK

Many recent studies [2, 14, 27, 28, 30, 32, 33] show that there is a strong correlation between customers' check-in activities and geographical distances. Thus leveraging geographical influences to improve the recommendation accuracy has been noticed by most of the current location-based recommendation work. For example, [2] first detect multiple centers for each customer based on their check-in histories. Then it recommends a business with the probability that is inversely proportional to the distance between the location of the business and the nearest customer center. In [32], geographical influence is modeled by a power-law distribution between the check-in probability and the pair-wise distance of two check-ins. [15, 34] utilize the kernel density estimation to study customers' check-ins and avoid employing a specific distribution. [17] exploits geographical neighborhood information by assuming that customers have similar preferences on neighboring POIs and POIs in the same region may share similar user preferences. PACE [29] explores the use of deep neural networks to learn location embeddings and user preferences over POIs. SAE-NAD [18] applies auto-encoder to learn POI recommendations. APOIR [35] employs a generative model to make POI recommendations. The generator suggests POIs based on the learned distribution by maximizing the probabilities over these POIs and the discriminator distinguishes the recommended POIs from the true check-ins.

The proposed method, SACRA, differs from the above work by not only focusing on incorporating ancillary information to compensate for the data sparsity, but also explicitly learning through comparing among users' historical check-ins to improve the recommendation performances. In addition, adversarial training is adopted to improve the generalization and robustness of recommendation system.

6 CONCLUSION

In this work, we study the problem of recommending prospective customers to businesses in LBSNs. To cope with the data sparsity issue, we utilize the geographical convenience to model the relative efforts of a visit from a user to a business. More over, we explicitly pay attention to user's historical check-in businesses as references. SACRA learns by comparing among the references and comparisons between the references and the target business to make recommendations. To distinguish the check-in businesses regarding a user, we utilize multiple attention mechanisms. Therefore, the user check-in behaviors are modeled more accurately and comprehensively. To further improve the generalization and robustness of the model, we employ adversarial training by dynamically constructing adversarial examples. Comprehensive experiments on two real-world datasets demonstrate the significant performance improvement of SACRA with comparisons to 12 baseline methods.

ACKNOWLEDGMENTS

This project was supported by NSF DGE-1829071. We also would like to thank the reviewers for their constructive feedback.

REFERENCES

[1] Jie Bao, Yu Zheng, and Mohamed F. Mokbel. 2012. Location-based and preference-aware recommendation using sparse geo-social networking data. In *SIGSPATIAL, Redondo Beach, CA, USA, November 7-9.*

[2] Chen Cheng, Haiqin Yang, Irwin King, and Michael R. Lyu. 2012. Fused Matrix Factorization with Geographical and Social Influence in Location-Based Social Networks. In *AAAI, July 22-26, 2012, Toronto, Ontario, Canada.*

[3] Arthur P Dempster, Nan M Laird, and Donald B Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the royal statistical society. Series B (methodological)* (1977).

[4] Huiji Gao, Jiliang Tang, Xia Hu, and Huan Liu. 2015. Content-Aware Point of Interest Recommendation on Location-Based Social Networks. In *AAAI, January 25-30, 2015, Austin, Texas, USA.* 1721–1727.

[5] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and Harnessing Adversarial Examples. In *ICLR.*

[6] Xiangnan He, Zhankui He, Xiaoyu Du, and Tat-Seng Chua. 2018. Adversarial Personalized Ranking for Recommendation. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018.* 355–364.

[7] Bo Hu and Martin Ester. 2013. Spatial topic modeling in online social media for location recommendation. In *RecSys, Hong Kong, China, October 12-16.*

[8] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *ICDM, December 15-19, 2008, Pisa, Italy.*

[9] Huayu Li, Yong Ge, Richang Hong, and Hengshu Zhu. 2016. Point-of-Interest Recommendations: Learning Potential Check-ins from Friends. In *SIGKDD, San Francisco, CA, USA, August 13-17.*

[10] Ruirui Li, Jyunyu Jiang, Chelsea Ju, and Wei Wang. 2019. CORALS: Who are My Potential New Customers? Tapping into the Wisdom of Customers' Decisions. In *WSDM, Melbourne, Australia, February 11-15.*

[11] Ruirui Li, Jyun-Yu Jiang, Jiahao Liu, Chu-Cheng Hsieh, and Wei Wang. 2020. Automatic Speaker Recognition with Limited Data. In *Proceedings of WSDM, Houston, Texas, USA, February 3-7.*

[12] Ruirui Li, Liangda Li, Xian Wu, Yunhong Zhou, and Wei Wang. 2019. Click Feedback-Aware Query Recommendation Using Adversarial Examples. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019.* 2978–2984.

[13] Xutao Li, Gao Cong, Xiaoli Li, Tuan-Anh Nguyen Pham, and Shonali Krishnaswamy. 2015. Rank-GeoFM: A Ranking based Geographical Factorization Method for Point of Interest Recommendation. In *SIGIR, Santiago, Chile, August 9-13, 2015.* 433–442.

[14] Defu Lian, Cong Zhao, Xing Xie, Guangzhong Sun, Enhong Chen, and Yong Rui. 2014. GeoMF: joint geographical modeling and matrix factorization for point-of-interest recommendation. In *KDD, New York, USA - August 24 - 27.*

[15] Moshe Lichman and Padhraic Smyth. 2014. Modeling human location data with mixtures of kernel densities. In *KDD, New York, USA - August 24 - 27.*

[16] Yanchi Liu, Chuanren Liu, Bin Liu, Meng Qu, and Hui Xiong. 2016. Unified Point-of-Interest Recommendation with Temporal Interval Assessment. In *SIGKDD, San Francisco, CA, USA, August 13-17.*

[17] Yong Liu, Wei Wei, Aixin Sun, and Chunyan Miao. 2014. Exploiting Geographical Neighborhood Characteristics for Location Recommendation. In *CIKM, Shanghai, China, November 3-7, 2014.* 739–748.

[18] Chen Ma, Yingxue Zhang, Qinglong Wang, and Xue Liu. 2018. Point-of-Interest Recommendation: Exploiting Self-Attentive Autoencoders with Neighbor-Aware Influence. In *CIKM.* ACM, 697–706.

[19] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI '09, Montreal, QC, Canada, June 18-21.*

[20] Douglas A. Reynolds. 2009. Gaussian Mixture Models. In *Encyclopedia of Biometrics.*

[21] Yue Shi, Alexandros Karatzoglou, Linas Baltrunas, Martha Larson, Nuria Oliver, and Alan Hanjalic. 2012. CLiMF: learning to maximize reciprocal rank with collaborative less-is-more filtering. In *RecSys, Dublin, Ireland, September 9-13.*

[22] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *ICLR.*

[23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NIPS, 4-9 December 2017, Long Beach, CA, USA.* 6000–6010.

[24] Suhang Wang, Yilin Wang, Jiliang Tang, Kai Shu, Suhas Ranganath, and Huan Liu. 2017. What Your Images Reveal: Exploiting Visual Contents for Point-of-Interest Recommendation. In *WWW, Perth, Australia, April 3-7, 2017.* 391–400.

[25] Markus Weimer, Alexandros Karatzoglou, Quoc V. Le, and Alexander J. Smola. 2007. COFI RANK - Maximum Margin Matrix Factorization for Collaborative Ranking. In *NIPS, Vancouver, British Columbia, Canada, December 3-6.*

[26] Markus Weimer, Alexandros Karatzoglou, and Alexander J. Smola. 2015. Improving maximum margin matrix factorization. *Machine Learning* 72, 3 (2015).

[27] Xian Wu, Yuxiao Dong, Baoxu Shi, Ananthram Swami, and Nitesh V. Chawla. 2018. Who will Attend This Event Together? Event Attendance Prediction via Deep LSTM Networks. In *Proceedings of the 2018 SIAM International Conference on Data Mining, SDM 2018, May 3-5, 2018, San Diego, CA, USA.* 180–188.

[28] Xian Wu, Baoxu Shi, Yuxiao Dong, Chao Huang, Louis Faust, and Nitesh V. Chawla. 2018. RESTFul: Resolution-Aware Forecasting of Behavioral Time Series Data. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018.* 1073–1082.

[29] Carl Yang, Lanxiao Bai, Chao Zhang, Quan Yuan, and Jiawei Han. 2017. Bridging Collaborative Filtering and Semi-Supervised Learning: A Neural Approach for POI Recommendation. In *KDD, Halifax, NS, Canada, August 13 - 17, 2017.* 1245–1254.

[30] Huaxiu Yao, Yiding Liu, Ying Wei, Xianfeng Tang, and Zhenhui Li. 2019. Learning from Multiple Cities: A Meta-Learning Approach for Spatial-Temporal Prediction. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019.* 2181–2191.

[31] Zijun Yao, Yanjie Fu, Bin Liu, Yanchi Liu, and Hui Xiong. 2016. POI Recommendation: A Temporal Matching between POI Popularity and User Regularity. In *ICDM, December 12-15, 2016, Barcelona, Spain.*

[32] Mao Ye, Peifeng Yin, Wang-Chien Lee, and Dik Lun Lee. 2011. Exploiting geographical influence for collaborative point-of-interest recommendation. In *SIGIR, Beijing, China, July 25-29.*

[33] Hongzhi Yin, Bin Cui, Yizhou Sun, Zhiting Hu, and Ling Chen. 2014. LCARS: A Spatial Item Recommender System. *ACM Trans. Inf. Syst.* 32, 3 (2014).

[34] Jia-Dong Zhang and Chi-Yin Chow. 2013. iGSLR: personalized geo-social location recommendation: a kernel density estimation approach. In *SIGSPATIAL, Orlando, FL, USA, November 5-8.*

[35] Fan Zhou, Ruiyang Yin, Kunpeng Zhang, Goce Trajcevski, Ting Zhong, and Jin Wu. 2019. Adversarial Point-of-Interest Recommendation. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019.* 3462–34618.