Hidden POI Ranking with Spatial Crowdsourcing

Yue Cui
School of Computer Science and Engineering, University of Electronic Science and Technology of China
Chengdu, China
yueming@std.uestc.edu.cn

Liwei Deng
School of Computer Science and Engineering, University of Electronic Science and Technology of China
Chengdu, China
deng_liwei@std.uestc.edu.cn

Yan Zhao
School of Computer Science and Technology, Soochow University
Suzhou, China
zhaoyan@suda.edu.cn

Bin Yao
Department of Computer Science and Engineering, Shanghai Jiao Tong University
Shanghai, China
yaobin@cs.sjtu.edu.cn

Vincent W. Zheng
WeBank
Shenzhen, China
vincentz@webank.com

Kai Zheng*
School of Computer Science and Engineering, University of Electronic Science and Technology of China
Chengdu, China
zhengkai@uestc.edu.cn

ABSTRACT

Exploring Hidden Points of Interest (H-POIs), which are rarely referred in online search and recommendation systems due to insufficient check-in records, benefits business and individuals. In this work, we investigate how to eliminate the hidden feature of H-POIs by enhancing conventional crowdsourced ranking aggregation framework with heterogeneous (i.e., H-POI and Popular Point of Interest (P-POI)) pairwise tasks. We propose a two-phase solution focusing on both effectiveness and efficiency. In offline phase, we substantially narrow down the search space by retrieving a set of geo-textual valid heterogeneous pairs as the initial candidates and develop two practical data-driven strategies to compute worker qualities. In the online phase, we minimize the cost of assessment by introducing an active learning algorithm to jointly select pairs and workers with worker quality, uncertainty of P-POI rankings and uncertainty of the model taken into account. In addition, a (Minimum Spanning) Tree-constrained Skip search strategy is proposed for the purpose of reducing search time cost. Empirical experiments based on real POI datasets verify that the ranking accuracy of H-POIs can be greatly improved with small number of query iterations.

CCS CONCEPTS

• **Information systems** → *Location based services*; **Crowdsourcing**; **Answer ranking**; *Spatial-temporal systems.*

KEYWORDS

Hidden POI, POI ranking, spatial crowdsourcing

*Kai Zheng is the corresponding author.

KDD '19, August 4–8, 2019, Anchorage, AK, USA

ACM ISBN 978-1-4503-6201-6/19/08…$15.00
https://doi.org/10.1145/3292500.3330844

ACM Reference Format:
Yue Cui, Liwei Deng, Yan Zhao, Bin Yao, Vincent W. Zheng, and Kai Zheng. 2019. Hidden POI Ranking with Spatial Crowdsourcing. In *The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3292500.3330844

1 INTRODUCTION

In Location Based Social Networks (LBSNs), people check-in, rate and share their experience with friends when visiting a Point of Interest (POI). They also use these systems to explore new potential destinations. These networks are growing at an unprecedented speed, which makes it difficult for a user to extract her favorable information. To deal with the huge volume of data in LBSNs and understand user behavior patterns, POI recommendation systems, aiming at recommending a user the POIs that she may be interested in but has never visited, have attracted increasing attention recently.

However, as people rely more and more on recommendation systems, some challenges have emerged. First, the diversity of a user's decision will surely deteriorate. A personalized POI preference modeling strategy concerning geo-spatial factors [2, 13, 24, 25], user preference [2, 13, 24] and social influence [24, 25], can recommend a POI/POIs that is/are highly preferred by the user. But meanwhile, she may not have equitable access to other information that allows her to make a choice different from her current preference. Second, the recommendation result is often biased for popular POIs that have a greater share of check-in records in the overall LBSNs database and thus aggregates inequality of business market. A simple example is that when you travel to Chengdu and seek for a local stylish hot-pot restaurant for dinner, the places recommendation system suggests you to go are most likely to be ones frequently visited by tourists but not the most authentic ones that hide in small alleys and visited by the locals as daily routine. Therefore, the effectiveness of existing POI recommendation systems is yet to be improved on recommending POIs with few online check-in records, which will be formally defined as Hidden POIs (H-POIs).

The recent rise of crowdsourcing marketplaces allows individuals and organizations to leverage brainpower of humans to operate on data in ways that are difficult for computers to perform [10].

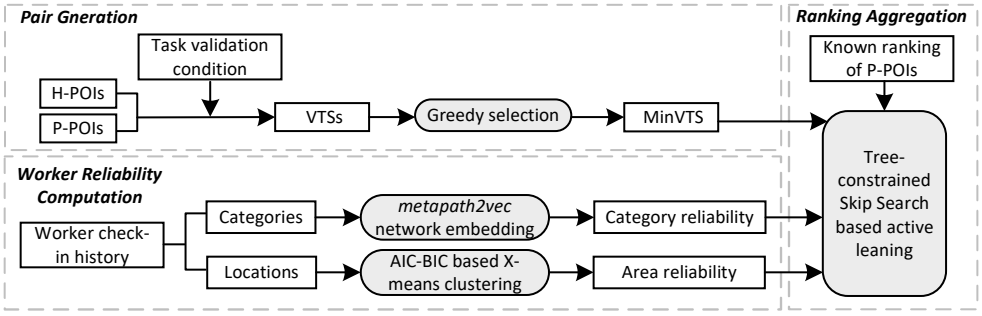


Figure 1: H-POI Ranking Framework Overview

Prominent platforms such as Amazon Mechanical Turk[1] and Zhihu[2] have made it easy for everyone to publish tasks or be a worker to perform tasks. Based on crowdsourcing platforms, ranking from binary comparisons becomes a ubiquitous problem in modern machine learning applications [21]. Given a set of n objects and a set of binary comparisons between pairs of objects, e.g., i is better than j (denoted as $i \succ j$), the task of ranking aggregation is to infer a total order over objects that aggregates the given measurements. Common settings for this problem allow binary comparisons to be measured either heuristically or based on machine learning approaches and extensive studies have been conducted [6, 12, 17, 18, 22, 23]. Inspired by these works, we put forward H-POI ranking problem with crowdsourcing, which takes advantage of crowd power to rank those POIs that are rarely mentioned in LBSNs but still of interest to people in their real lives.

It is worth mentioned that our works share some similarities with [23], which also studies the problem of ranking aggregation in crowdsourced settings. It shows in [23] that incorporation of worker quality and adaptively sampling the next assessment pair and worker can reduce the labeling cost while maintain accuracy. However, their methodologies can not be applied to our problem because of the following challenges.

First, due to the "hidden" feature of H-POIs, only special group of workers are qualified to perform a certain task. This is different from pairwise ranking aggregation on common objects, e.g., images classification, sentence comprehension [23, 26], where almost every worker on the Internet is capable (can recognize the objects and report an answer based on her experience) of a performing task. Thus, worker quality need to be estimated in a more accurate way. Second, it is impractical to ask a worker to compare two H-POIs, since it is highly likely that a worker has never been to one sides or even both sides of a H-POI pair. However, in a conventional crowdsourcing setting, in [23] for instance, the two sides of the pairwise tasks are comparable to any worker. Third, the amount of H-POIs to be ranked can be very large. Though the active learning strategy put forward in [23] performs well on reducing the number of tasks, the searching time cost of obtaining a next pair and worker can be incredibly high, especially when the candidate task pool is large.

In this work, we aim to address the above challenges by a series of novel strategies. First, instead of taking pure H-POIs as pairwise tasks, i.e., <H-POI, H-POI>, we generate heterogeneous pairs, i.e., <H-POI, P-POI>, (where P-POI is short for Popular POI and is formally defined in Definition 3). The validation of a heterogeneously pairwise task is defined considering both category (see Definition 4) and service region (see Definition 5) of POIs. More specifically, a pair is valid only when the H-POI and P-POI are of the same category and the service region of H-POI is included in that of the P-POI. We then evaluate worker qualification over certain task type and task location by taking workers' check-in records into account. As for category reliability, we define a check-in graph, design 3 meta-paths and then adapt a state-of-the-art *metapath2vec* network embedding approach to learn the representation vectors of workers and categories. Worker's qualification score can be computed based on the obtained vectors. Second, to identify each worker's most active areas, we adapt X-means [8] clustering algorithm with a modification of model quality criterion, to deal with the case when the amount of worker check-in records varies widely. Third, to maintain effectiveness and improve efficiency, we propose an optimized active learning strategy, which uses Tree-constrained Skip (TCS) search to reduce the search space, and further, we make use of P-POIs ranking to supervise the active learning process. The framework for this H-POI ranking strategy is shown in Figure 1.

Our contributions in this paper can be summarized as follows:

- We identify necessities and challenges of exploring H-POIs. Based on this, we define a two-phase solution to aggregate ranking of H-POIs.
- In the offline phase, we present novel pair generation strategies and develop worker reliability evaluation models.
- In the online phase, we present a novel active learning strategy called **T**ree-**c**onstrained **S**kip with **S**upervision (TCSS), aiming at reducing searching cost and selecting next pairs that can better improve our current knowledge of the ranking by taking worker quality and supervision of P-POIs ranking into account.
- Finally, we conduct extensive experiments based on real POI-review datasets. The favorable results verify our expectation that our proposed framework can reach a high accuracy with relatively small number of iterations.

This paper is organized as follows. Section 2 provides definitions and states the problem. In Section 3, we design a greedy strategy to generate pairwise tasks, present approaches to evaluate worker reliability on categories and locations respectively. Moreover, two novel active learning strategies are described in Section 3. We report the experimental results in Section 4, followed by related work in Section 5. Finally, we conclude the paper in Section 6.

[1] https://www.mturk.com/
[2] https://www.zhihu.com/signup?next=%2F

2 PROBLEM STATEMENT

In this section, we present a set of preliminary concepts and then state our problem.

2.1 Preliminary

Definition 1. (Point of Interest) *A Point of Interest (POI), denoted by p, is a place associated with positional coordinate l_p, a category, an overall score $s_p \in [0, S]$, where S is the upper bound of scores, and a list of check-in records c_p: $\{c_p^1, c_p^2, ...\}$. Each check-in record c_p^i includes a user ID, check-in time and the user's rating score $s_p^i \in [0, S]$.*

Definition 2. (Hidden Point of Interest) *Given a threshold θ, a Point of Interest (POI) is called a Hidden POI (H-POI), denoted by p_h, if its number of check-in records is less than θ, i.e., $|c_p| \leq \theta$.*

Definition 3. (Popular Point of Interest) *In contrast with H-POI, a POI is called a Popular POI (P-POI), denoted by p_p, if its number of check-in records is more than Θ, i.e., $|c_p| \geq \Theta$.*

We assume the overall score of a P-POI is valid and such of a H-POI is invalid for insufficient rating records.

Definition 4. (POI Category) *The category of POI p, denoted as C_p, indicates the functionality of it. A POI may fall into multiple categories.*

Definition 5. (POI Service Region) *Given a POI p, its service region is described as a circular area with radius r_p and center l_p.*

Since it is hard to obtain accurate value of r_p, we assume that r_p is a Gaussian distributed random variable, i.e., $r_p \sim N(\mu, \delta)$. μ and δ are identical for a certain POI type (hidden or popular). We assume the mean of normal distribution for P-POIs is k times larger than that of H-POIs, i.e, $\mu_{p_p} = k\mu_{p_h}$, where $k > 1$. μ_{p_h} and k are manually set parameters based on the geographical scale of the study area.

2.2 Problem Definition

Definition 6. (Spatial Task) *A spatial task, denoted by $s =< p_h, p_p >$, is a pairwise comparison task between H-POI p_h and P-POI p_p. A worker is supposed to report her preferred POI in the given spatial task.*

Definition 7. (Comparable Hidden Point of Interest) *A H-POI p_h is called a Comparable H-POI, if there exists at least κ P-POIs and each P-POI, i.e., p_p, should satisfy following conditions:*

(1) $C_{p_h} = C_{p_p}$, *and*
(2) $d(l_{p_h}, l_{p_p}) \leq r_{p_p} - r_{p_h}$

where $d(a, b)$ is the Euclidean distance between a and b.

We mainly take distance and category into consideration when generating heterogeneous comparable pairs. The second constraint ensures a H-POI's service region is inside that of the P-POI, which implies that if a worker ever had visited p_h, she is highly likely to know the p_p as well.

Definition 8. (Valid Spatial Task Set) *Given a spatial task set, it is called a Valid Spatial Task Set (VTS), denoted by S, if all comparable H-POIs are included and for each Comparable H-POI, the comparable conditions can be satisfied by P-POIs in VTS.*

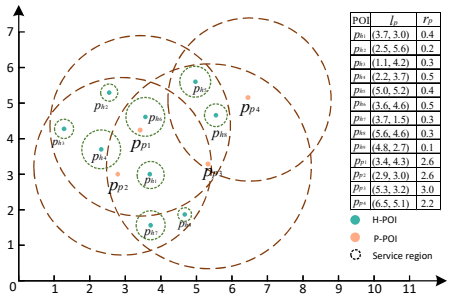


Figure 2: Running Example

Definition 9. (Minimum Valid Spatial Task Set) *A VTS is called a Minimum Valid Spatial Task Set (MinVTS), denoted as S_{min}, if none of its subset is a VTS.*

Problem Statement. Given a H-POI set P_h, a P-POI set P_p and a spatial crowdsourcing worker set W, we study the problem of how to generate candidate tasks and distribute proper tasks to proper workers with a goal of aggregating an accurate ranking for Comparable H-POIs in P_h in a small number of query iterations.

Some major notations used throughout the paper are listed in Appendix A

3 PROPOSED METHOD

3.1 Pair Generation

There usually exists a large number of P-POIs, H-POIs and valid <H-POI, P-POI> pairs in a VTS. Therefore, it is quite expensive to search for a best pair in each round of the subsequent active learning process. To solve this problem, we construct MinVTS, in which the number of P-POI is minimal, all H-POIs are comparable and the comparable conditions can be satisfied by P-POIs in MinVTS. Formally, the optimization goal of building MinVTS is to find minimum P-POIs that satisfy comparable constraints of all Comparable H-POIs.

However, finding an optimal MinVTS is time consuming, considering the large number of H-POIs and P-POIs. A greedy search algorithm is designed below.

MinVTS Greedy Search Algorithm. Algorithm 1 in Appendix B.1 illustrates the basic process of MinVTS greedy search. Input of the algorithm is H-POI set P_h, P-POI set P_p, and κ (the least number of P-POIs a that a H-POI should be matched with). First, MinVTS Greedy Search finds all Comparable H-POIs and constructs a VTS that contains all valid spatial tasks for each H-POI. It initializes zero arrays M and N to hold the "demand" and "supply" of H-POIs and P-POIs, e.g., $M_{p_h} = 0$ indicates there are κ tasks required by p_h and $N_{p_p} = 5$ means p_p can meet the demand of 5 H-POIs. In each iteration, it is prior to match H-POIs with the highest demand (the smallest entry in M). P-POI that contributes to the highest demanded H-POIs is greedily selected and added to the output set. If there are more than one P-POI candidates selected in above greedy process, then a P-POI that can meet demand of the greatest number of unsatisfied (number of matched P-POIs is less than κ) H-POIs will be selected as priority. Then M is refined. The above iteration continues until each H-POI is matched in κ tasks.

The time complexity of Algorithm 1 is $O(|P_h| \times |P_p| + \kappa \times |P_h| \times |P_p|) = O(|P_h| \times |P_p|)$, the first term of which is the cost of finding all

Table 1: Example of Pair Generation

iteration	M	min	I_{min}	N	S_{min}
1^{st}	{1:0,2:0,3:0,4:0,5:0,6:0,7:0,8:0,9:0}	0	{1,2,3,4,5,6,7,8,9}	{1:7,2:7,3:6,4:2}	{{1:1,2,3,4,5,6,8} }
2^{nd}	{1:1,2:1,3:1,4:1,5:1,6:1,7:0,8:1,9:0}	0	{7,9}	{2:2,3:2,4:0}	{{1:1,2,3,4,5,6,8},{2:1,2,3,4,6,7,9} }
3^{rd}	{1:2,2:2,3:2,4:2,5:1,6:2,7:1,8:1,9:1}	1	{5,7,8,9}	{3:4,4:2}	{{1:1,2,3,4,5,6,8},{2:1,2,3,4,6,7,9}, {3:5,7,8,9} }
4^{th}	{1:2,2:2,3:2,4:2,5:2,6:2,7:2,8:2,9:2}	2	{-}	{-}	{{1:1,2,3,4,5,6,8},{2:1,2,3,4,6,7,9}, {3:5,7,8,9} }

Comparable H-POIs and their corresponding P-POIs and the second term illustrates the worst case of MinVTS search, where all H-POIs are matched with κ different P-POIs. In the worst case of space cost, where in each iteration $|I_{min}| = |P_h|$, the space complexity is $O(|P_h| \times |P_p|)$.

We demonstrate the algorithm with the running example in Figure 2. Table 1 shows the iteration progress, where κ is set as 2. The first line of Table 1 can be interpreted as: in the first iteration, number of P-POIs that have been matched for H-POIs with index $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ are all 0 (see "M" column), thus, "min" is set to be 0 (see "min" column), the indexes of H-POIs whose entry is "min" in M are $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ (see "I_{min}" column), the indexes of P-POIs can be matched to H-POIs in I_{min} are $\{1, 2, 3, 4\}$ and their corresponding contribution to H-POIs in I_{min} are $\{7, 7, 6, 2\}$ (see "N" column), thus, p_{p_1} is picked, and $\{\underline{1} : 1, 2, 3, 4, 5, 6, 8\}$ denotes p_{p_1} is added to S_{min} in this iteration with related H-POIs $\{p_{h_1}, p_{h_2}, p_{h_3}, p_{h_4}, p_{h_5}, p_{h_6}, p_{h_8}\}$ (see "S_{min}" column). Similarly, we continue the loop until "min" becomes 2 in the last iteration.

3.2 Category and Area Aware Worker Reliability Calculation

It is essential to understand how well a worker can perform on tasks of certain types and at certain locations, before assigning her a task. In this section, we discuss how to evaluate worker reliability for POI categories and areas by using her historical check-ins.

We first design meta-paths, employ a network embedding approach to obtain workers' representation vectors and category representation vectors, and next compute the corresponding confidence score. Then we adapt a modified X-means clustering algorithm to identify each worker's active areas.

3.2.1 ***Category Reliability.*** In spatial crowdsourcing, a worker's mobility behavior patterns in the real physical world can be modeled as a heterogeneous information network, $G = (V, E)$, in which there are three type of nodes in V and two types of edges in E. More detailed descriptions of nodes and edges are as follows.

Definition 10. (Worker-based Node)*: A worker-based node represents a specific worker, denoted as W.*

Definition 11. (POI-based Node)*: A POI-based node represents a specific POI, denoted as P.*

Definition 12. (Category-based Node)*: A category-based node represents a POI category, denoted as C.*

Definition 13. (POI Relevance Edge)*: A POI relevance edge exists only between node W and node P, denoted as $e(W, P)$. It represents the POI Relevance Relation (PRR) of a worker and a POI. The edge weight $w(W, P)$ is defined as the times worker W visited P.*

Definition 14. (Category Relevance Edge)*: A category relevance edge exists only between node P and node C, denoted as $e(P, C)$, representing the Category Relevance Relation (CRR) of a POI and a POI category. The edge weight $w(P, C)$ is 1 if $e(P, C)$ exists.*

The prime goal in heterogeneous network representation learning is to learn low-dimensional latent representations $\mathbf{X} \in \mathbb{R}^{|V| \times d}$, $d \ll |V|$ that are able to capture the structural and semantic relations among them. DeepWalk [16] is an algorithm in network representation learning based on *word2vec* [14], which applies random walks to obtain node sequences. In a general DeepWalk setting, random walks are leveraged regardless of node type and edge type. However, the ignorance of node type may lead to poor information-extracting paths. The meta-path-based random walk has been proved to be effective dealing with heterogeneous information networks that consist of multi-typed and interconnected objects [5]. Inspired by *metapath2vec* model, we design a weighted meta-path-based random walk strategy and then propose a skip-gram model to achieve node embedding. Three types of meta-paths are designed as follows.

(1) *CPC path*: a CPC contains C nodes, P nodes and category relevance edge and is denoted in the form of $C_1 \xrightarrow{CRR} P_1 \xrightarrow{CRR} C_2 \ldots \xrightarrow{CRR} C_i \xrightarrow{CRR} P_i \ldots \xrightarrow{CRR} C_l$. It represents the relationships between a POI (P) and two categories (C).

(2) *PWP path*: a PWP path is denoted in the form of $P_1 \xrightarrow{PRR} W_1 \xrightarrow{PRR} P_2 \ldots \xrightarrow{PRR} P_i \xrightarrow{PRR} W_i \ldots \xrightarrow{PRR} P_l$, which contains W nodes, P nodes and POI relevance edges. It bridges the relationship between two POIs (P) and a worker (W).

(3) *CPWPC path*: a CPWPC path is denoted in the form of $C_1 \xrightarrow{CRR} P_1 \xrightarrow{PRR} W_1 \xrightarrow{PRR} P_2 \ldots \xrightarrow{CRR} C_i \xrightarrow{CRR} P_i \ldots \xrightarrow{CRR} C_l$, which contains W nodes, P nodes, C nodes, POI relevance edge and category relevance edges. It represents the two categories (C) and POIs (P) visited by a worker (W).

The above meta-path-based random walk strategy ensures that the semantic relationships between different types of nodes can be properly incorporated into skip-gram. The optimization function is defined as

$$arg\max_{\theta} \sum_{v \in V} \sum_{t \in T_V} \sum_{c_t \in N_t(v)} \log p(c_t|v;\theta) \quad (1)$$

where v is a node in V, T_V denotes set of node type in V, $N_t(v)$ is $v's$ neighborhood of type t and $p(c_t|v;\theta)$ denotes the conditional probability of having a context node c_t given a node v. As a common practice, $p(c_t|v;\theta)$ is defined as a softmax function [8], in which $p(c_t|v;\theta) = \frac{e^{X_{c_t} X_v}}{\sum_{u \in V_t} e^{X_u X_v}}$, X_v is the v^{th} row of $\mathbf{X}$ and V_t is the node set of type t in the network. Transition probability is defined as the weight between the current node and target node over the sum of weight between the current node and its neighbors that are of the same type as the target node. Getting the representation vector of each node, we are then enabled to estimate worker reliability over categories. The reliability score of worker w over category k can

be described as

$$\alpha_w^k = \frac{sim(X_w, X_{C_k})}{\sum_{j=1}^{|C|} sim(X_w, X_{C_j})} \quad (2)$$

where $sim(a, b)$ is the dot product of (a, b).

3.2.2 ***Area Reliability.*** To evaluate workers' reliability in a spatial-aware way, it is meaningful to understand her active area. Given a check-in data set of worker w, it is easy to obtain the check-in location set, $P^w = \{l_{p_1}, l_{p_2}, ..., l_{p_n}\}$, in the forms of latitude-longitude coordinate. We implement clustering algorithms to partition the n check-in locations into $k(\leq n)$ sets $P^w = \{P_1, P_2, ..., P_k\}$ and take focus point(s) of each set as the centroid(s) of the worker's active area(s). We denote P_i as the subset of L and its centroid is denoted as μ_i. Pelleg et al. [8] proposed X-means, a parameter independent clustering algorithm that can quickly estimates the number of partitions in k-means algorithm.

In X-means, the clustering centroids in each model is assumed with x Gaussians, each with identical variance and its own mean μ_i. Given a data set and candidate models that demonstrate clustering solutions with various values of x, the aim of X-means is to select the best model with Bayesian Information Criterion (BIC) [8]. However, as an approximation for a Bayes factor, BIC only valid when the number of samples to be considered is large [1], but the amount of check-in records for each worker can vary widely in a dataset. Inspired by X-means and to better fit our problem, we combine BIC and Akaike Information Criterion (AIC) [1], to score clustering models. AIC is another estimator of the relative quality of statistical models that can be applied to datasets with small number of samples.

Given a location set P, the BIC and AIC scores for model M_j can be calculated as follows:

$$BIC(M_j) = ln(\hat{L}) - \frac{m_j}{2} ln(R) \quad (3)$$

$$AIC(M_j) = 2m_j - 2ln(\hat{L}) \quad (4)$$

where $R = |P|$ describes the number of samples, m_j is the number of parameters in model M_j and $\hat{L}$ is the likelihood of data according to M_j and taken at the maximum likelihood point. $\hat{L}$ can be calculated as stated in Section 3.2 of [8].

We modify X-means algorithm simply by adaptively implementing AIC or BIC as splitting criterion. If the number of samples is larger than a threshold H, then the criterion used to measure model quality is BIC, otherwise AIC. We name the algorithm proposed as AIC-BIC based X-means (A-B X-means). The pseudo code of A-B X-means algorithm is presented in Appendix B.2

The A-B X-means algorithm described so far can identify proper x centroids for a worker. Given a task pair $s =< p_h, p_p >$, based on the obtained centroids $O = \{o_1, o_2, ...o_x\}$ of worker w, now we introduce how to calculate worker reliability on task locations by an internal based approach. First, we introduce a set of cut-off values $0 \equiv \gamma_0 < \gamma_1 < ... < \gamma_{m-1} \equiv \inf$. Assume the observed distance between a centroid o and a POI p is $d(o, p)$. If $\gamma_{i-1} < d(o, p) \leq \gamma_i$, then $D(o, p) = i$. Then, reliability of worker w on locations of POIs in task s, denoted as l_s, can be calculated as,

$$\beta_w^{l_s} = (1-\lambda) \max_{n\in[1,x]} \{\frac{1}{D(o_n, p_h)}\} + \lambda \max_{n\in[1,x]} \{\frac{1}{D(o_n, p_p)}\} \quad (5)$$

where $\lambda \in [0, 1]$ is a tuning parameter concerning the trade-off between worker reliability importance on H-POIs and P-POIs.

3.3 Ranking Aggregation

The main idea of ranking is to assign each object with a score s_i and obtain the rank by sorting the scores. In this section, we discuss how to assign pairs to proper workers and aggregate workers' feedbacks by presenting a novel active learning strategy called Tree-constrained Skip with Supervision (TCSS). Specifically, we first review a Bradley-Terry based active learning model, then discuss how to compute worker quality in the current spatial crowdsourced setting and finally present the TCSS approach.

3.3.1 ***Crowed-BT.*** Each pairwise comparison result is obtained from the crowd at a certain cost. It is desirable to properly select the next pair of candidates that can better improve our current knowledge of score distributions. Chen et al. extend Bradley-Terry model to crowdsourcing setting, namely Crowd-BT, and incorporate it with Bayesian framework to facilitate an active learning strategy [23].

In Crowed-BT, the score of each object, s_i, is modeled by a Gaussian distribution $N(\mu_i, \sigma_i)$, and the quality of worker η_w is modeled by a Beta distribution, $\mathbf{Beta}(a_w, b_w)$. It computes the probability of worker w giving the answer $o_i \succ o_j$, which is denoted as $o_i \succ_w o_j$, as

$$Pr(o_i \succ_w o_j) = \eta_w \frac{e^{s_i}}{e^{s_i} + e^{s_j}} + (1 - \eta_w) \frac{e^{s_j}}{e^{s_i} + e^{s_j}} \quad (6)$$

Then, a pure greedy strategy has been implemented to globally select the triple (o_i, o_j, w_k) that maximizes information gain defined in Equation 7.

$$\begin{aligned} Pr(o_i \succ_w o_j) \Big(& KL\Big(N(\mu_i^{i\succ wj}, \sigma_i^{i\succ wj}) || N(\mu_i, \sigma_i)\Big) \\ & + KL\Big(N(\mu_j^{i\succ wj}, \sigma_j^{i\succ wj}) || N(\mu_j, \sigma_j)\Big) + \\ & \varphi KL\Big(Beta(a_k^{i\succ wj}, b_k^{i\succ wj}) || Beta(a_k, b_k)\Big)\Big) \\ + Pr(o_i \prec_w o_j) \Big(& KL\Big(N(\mu_i^{i\prec wj}, \sigma_i^{i\prec wj}) || N(\mu_i, \sigma_i)\Big) \\ & + KL\Big(N(\mu_j^{i\prec wj}, \sigma_j^{i\prec wj}) || N(\mu_j, \sigma_j)\Big) + \\ & \varphi KL\Big(Beta(a_k^{i\prec wj}, b_k^{i\prec wj}) || Beta(a_k, b_k)\Big)\Big) \end{aligned} \quad (7)$$

where $KL(\cdot)$ denotes the Kullback-Leibler (KL) divergence, φ is a tuning parameter about comparison result and worker quality.

We choose to extend CrowdBT for that it well suit the preference and can be optimized in terms of geo-spatial and categorical worker quality and search space.

3.3.2 ***Computation of Overall Worker Quality.*** With information extracted from LBSNs, worker quality score η_w can be computed more accurately than be estimated with statistical tools. There are two indexes discussed in previous sections: category reliability α and location reliability β. Thus, the overall quality of worker w on task s can be computed as,

$$\eta_w^s = \epsilon \alpha_w^{C_{(s)}} + (1 - \epsilon) \beta_w^{l_s} \quad (8)$$

where $\epsilon \in [0, 1]$ is a manually set parameter tuning the trade-off between category and location reliability importance and $C_{(s)}$ denotes the category of task s.

Therefore, information gain can be redefined as a modification of Equation 7 with Beta terms excluded as follow.

$$\begin{aligned} & Pr(p_h \succ_w p_p)\Big(KL\Big(N(\mu_{p_h}^{p_h \succ_w p_p}, \sigma_{p_h}^{p_h \succ_w p_p})||N(\mu_{p_h}, \sigma_{p_h})\Big) \\ & + KL\Big(N(\mu_{p_p}^{p_h \succ_w p_p}, \sigma_{p_p}^{p_h \succ_w p_p})||N(\mu_{p_p}, \sigma_{p_p})\Big)\Big) \\ & + Pr(p_h \prec_w p_p)\Big(KL\Big(N(\mu_{p_h}^{p_h \prec_w p_p}, \sigma_{p_h}^{p_h \prec_w p_p})||N(\mu_{p_h}, \sigma_{p_h})\Big) \\ & + KL\Big(N(\mu_{p_p}^{p_h \prec_w p_p}, \sigma_{p_p}^{p_h \prec_w p_p})||N(\mu_{p_p}, \sigma_{p_p})\Big)\Big) \end{aligned} \tag{9}$$

3.3.3 ***Tree-Constrained Skip Search.*** Despite its popularity for the definition of expected information gain function, KL divergence and its current extensions, the one presented in [23] for instance, have two major shortcomings: 1) only one (pair of) object will be selected and updated in each iteration and 2) in each iteration, the searching time cost for a global optimal (pair of) object can be very high. To deal with these problems, we propose a **T**ree-**c**onstrained **S**kip (TCS) Search algorithm that finds a local optimal pair in a Minimum Spanning Tree's (MST) nodes and skip to two sub-optimal nodes from the selected pair.

Algorithm 3 in Appendix B.3 illustrates TCS algorithm. A MinVTS can be further described in a graph. In each connected component, if an edge (p_h, p_p) exists, the weight is defined as

$$w(p_h, p_p) = -(d_{p_h} + d_{p_p} - 1) \tag{10}$$

where d_p is the degree of node p. The lower the weight is, the more likely the current nodes of the edge can compare with other nodes, and thus, the pair is more informative to skip from. We perform classic Kruskal Algorithm [11] on the weighted graph, purpose of which is to select the most informative $(|P_h| + |P_p| - 1)$ pairs. The TCS algorithm can be described in a three-step iteration.

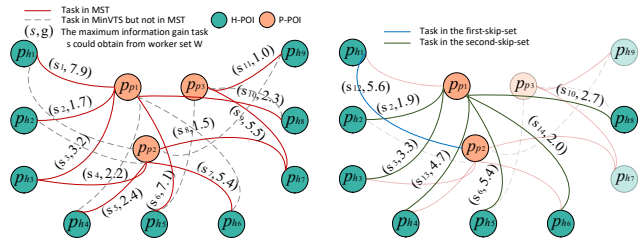
1) Select a pair among all MSTs' edges and a worker that maximize Equation 9. Distribute the task to the selected worker. Update scores of POIs.

2) Select a pair among all the neighbors of the H-POI of pair selected in step (1) and a worker that maximize Equation 9. Assign the task to the worker and update scores of POIs.

3) Select a pair among all the neighbors of the P-POI of pair selected in step (1) and a worker that maximize Equation 9. Assign the task to the worker and update scores of POIs. Then go to step (1).

We describe one iteration of TCS in the running example. As depicted in Figure 3, $s_1 =< p_{h_1}, p_{p_1} >$ from MST is selected with a proper worker in the first step (iteration). Then we update scores for POIs. Next, we select the next pair, i.e., $\{s_{12}\}$, from the adjacent nodes of p_{h_1} with a proper worker that maximizes information gain defined in Equation 9, and update the distribution of selected POIs. Then we go back to P-POI in s_1, select a next pair from $\{s_2, s_3, s_6, s_{10}, s_{13}, s_{14}\}$ with a proper worker and update POIs scores. In this way, all pairs have the possibility of being queried with its expected information gain considered.

3.3.4 ***Tree-Constrained Skip Search with Supervision.*** As described, reliable scores of P-POIs can be obtained in prior from most LSBNs. We take advantage of this information to extend TCS to Tree-Constrained Skip with Supervision (TCSS) Search. Given a MinVTS, the P-POI set included in it can be described as $P_p = \{p_{p_1}, p_{p_2}, ..., p_{p_n}\}$ and its current rank position (obtained from



(a) Step 1 (b) Step 2 and 3

Figure 3: Example of Tree-Constrained Skip Search

sorting the current scores) $R = \{r_1, r_2, ..., r_n\}$. As a common metric [23, 26], the ranking accuracy can be described as,

$$ACC = \frac{\sum_{p_{p_i}, p_{p_j} \in P_p} \mathbb{I}(r_i < r_j \cap t_i < t_j)}{|P_p|(|P_p|+1)/2} \tag{11}$$

where t_i is the real ranking position of p_i, and $\mathbb{I}$ is a function which returns 1 if the condition is true; otherwise 0.

Given a candidate triple $< p_h, p_p, w >$, the change of ranking accuracy if obtaining $p_h \succ_w p_p$, which is denoted as $\Delta^{p_h \succ_w p_p}$, can be defined as

$$\begin{aligned} \Delta^{p_h \succ_w p_p} &= ACC^{p_h \succ_w p_p} - ACC \\ &= \frac{\sum_{p_{p_i}, p_{p_j} \in P_p} \mathbb{I}(r_i^{p_h \succ_w p_p} < r_j^{p_h \succ_w p_p} \cap t_i < t_j)}{|P_p|(|P_p|+1)/2} \\ &\quad - ACC \end{aligned} \tag{12}$$

where $r_i^{p_h \succ_w p_p}$ denotes the ranking position of p_i if $p_h \succ p_p$ is reported by worker w. Similarly, we could obtain $\Delta^{p_h \prec_w p_p}$.

Theorem 3.1. *Δ has a similar probability of being negative or positive in both cases of $p_h \succ_w p_p$ and $p_h \prec_w p_p$.*

We prove Theorem 3.1 in Appendix C.

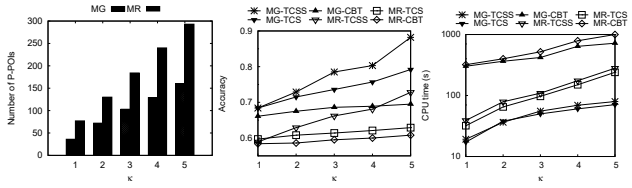
Theorem 3.1 indicates that Δ can well measure the contribution of pairs when workers answers are unknown in advance.

Now, we rewrite Equation 9 as

$$\begin{aligned} & Pr(p_h \succ_w p_p)\Big(KL\Big(N(\mu_{p_h}^{p_h \succ_w p_p}, \sigma_{p_h}^{p_h \succ_w p_p})||N(\mu_{p_h}, \sigma_{p_h})\Big) \\ & + (1 + \Delta^{p_h \succ_w p_p}) KL\Big(N(\mu_{p_p}^{p_h \succ_w p_p}, \sigma_{p_p}^{p_h \succ_w p_p})||N(\mu_{p_p}, \sigma_{p_p})\Big)\Big) \\ & + Pr(p_h \prec_w p_p)\Big(KL\Big(N(\mu_{p_h}^{p_h \prec_w p_p}, \sigma_{p_h}^{p_h \prec_w p_p})||N(\mu_{p_h}, \sigma_{p_h})\Big) \\ & + (1 + \Delta^{p_h \prec_w p_p}) KL\Big(N(\mu_{p_p}^{p_h \prec_w p_p}, \sigma_{p_p}^{p_h \prec_w p_p})||N(\mu_{p_p}, \sigma_{p_p})\Big)\Big) \end{aligned} \tag{13}$$

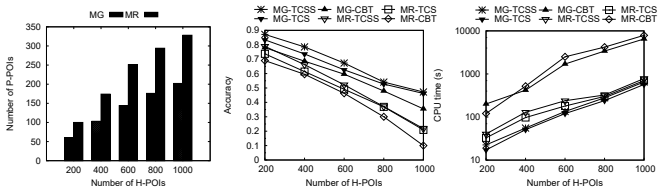
The above equation will degenerate to Equation 9 if all P-POIs rank correctly. Due to limited space, readers could find the parameter update strategy in Section 4.1 of [23].

We now theoretically explain why TCSS is more efficient. First, with the implementation of MinVTS Greedy Search, the comparison graph can be connected well since a P-POI is selected only if it can be matched with the largest number of matchable H-POIs. This promises the strong connected components of MinVTS is of relatively small amount. Search time complexity thus could reach an optimization of $O(|W| \times (|P_h| + |P_p|))$ while that of active learning in CrowdBT is $O(|W| \times |P_h| \times |P_p|)$ in the best and worst cases. Second, information gain defined in Equation 13 considers uncertainty of model, worker quality, uncertainty of P-POI ranking, which makes selections effective. Third, though in the first step, pairs are not



(a) (b) (c)

Figure 4: Effect of κ (LV)



(a) (b) (c)

Figure 5: Effect of H-POI Number (LV)

selected as a global optimal of Equation 13, the searching space includes all P-POIs and H-POIs. Moreover, as defined in Equation 10, candidate pairs in the first iteration have the most neighbors, and thus, in next skips (step 2 and 3) the candidate pool is large, which increases the probability of selecting an informative pair that might not be an edge of MSTs.

4 EXPERIMENTS AND RESULTS

All the algorithms are implemented on an Intel Xeon CPU E5-2690 v4 @ 2.60 GHz. The experimental setups are described in Appendix D.1.

4.1 Experimental Results on Las Vegas Dataset

For effectiveness and efficiency measurement, we evaluate the H-POI ranking accuracy and CPU time cost. Accuracy is computed as

$$ACC = \frac{\sum_{p_{h_i}, p_{h_j} \in P_h} \mathbb{I}(r_i < r_j \cap t_i < t_j)}{|P_h|(|P_h|+1)/2} \tag{14}$$

where r_i denotes the obtained ranking position of p_{h_i} and t_i denotes the ground-truth position of p_{h_i}. The CPU time cost only contains the next pair searching time in ranking aggregation phase, because this is the only online phase of all the proposed approaches. Meanwhile, we exclude the time cost from loading data from disk, which dominates the computational time cost. All the parameters and their default values in the experiments on LV dataset are summarized in Appendix D.2.

4.1.1 Experiments on Pair Generation. We then evaluate the number of selected P-POIs from pair generation strategy, and the influence on the ranking accuracy and CPU time on aggregation phase when parameter κ and number of sampled H-POIs varies.

Baseline. Since to the best of our knowledge, there is no existing pair generation algorithm, we devise a baseline method as follow. For each iteration in Algorithm 1, we randomly select a P-POI instead of the one best meets the needs of H-POIs. This strategy is denoted as MinVTS Random (MR). We compare the proposed MG based TCSS (MG-TCSS) with

(1) MR based TCSS (MR-TCSS), where the input MinVTS of TCSS is generated from MR;

(2) MR based TCS (MR-TCS);

(3) MR based CrowdBT [23] (MR-CBT), where CBT is the active learning strategy described in Section 3.3.1.

(4) MG based TCS (MG-TCS);

(5) MG based CBT (MG-CBT).

Effect of κ. κ directly affects the number of P-POIs in MinVTS. As demonstrated in Figure 4 (a), MinVTS Greedy (MG) significantly reduce the number of selected P-POIs. From Figure 4 (b), we observe that when κ increases, the accuracy of ranking based on TCS/TCSS related algorithms increases; MG-TCSS increases most notably and accuracy of CBT based aggregation remains relatively stable. This is because: 1) small number of P-POIs lead to a more densely connected graph that TCS and TCSS could effectively select pairs that can better improve our knowledge of rankings; 2) the strategy of P-POI supervision in TCSS performs better when the number of P-POIs and H-POIs are not different much. Nevertheless, greater number of P-POI will result in more candidate pairwise tasks and lead to a higher searching time cost as shown in Figure 4 (c). Moreover, the implementation of TCS and TCSS significantly saves CPU time.

Effect of H-POI number. Figure 5 shows the P-POI number, aggregation accuracy and time cost on different pair generation strategies by varying the number of sampled H-POI to be ranked. Not surprisingly, the number of P-POIs in MinVTS increases when H-POI number rises as there are more Comparable H-POIs need to be matched. An interesting observation is that the number of P-POIs picked in MG increases slower than that of MR. To explain this, recalling the greedy strategy of MG, in each iteration, a P-POI is selected first to meet the needs of the most demanding H-POI and then adjusted with next-most demanding one. When the number of iterations is small, the number of P-POIs matched with each H-POI is less than the chosen κ. So, this strategy could make sure the selected P-POIs can balance all H-POIs' current number of matched P-POIs and meanwhile match with its maximal matchable H-POIs. As shown in Figure 5 (b), the accuracy of ranking decreases when H-POI number becomes large. Observing the details, we find that the decrease of MG based algorithms is slower than that of MR. This is because the total number of POI is small and the connectivity of graph is stronger. The same explanation fits the trend of sub-graph (c) in Figure 5.

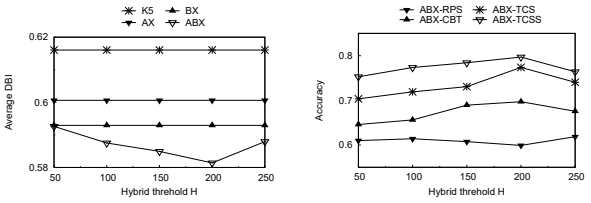
4.1.2 Experiments on Worker Qualification Evaluation Phase. Next we evaluate the performance of worker qualification methods proposed in Section 3.2. We first evaluate the clustering effectiveness using a popular metric Davies-Bouldin Index (DBI) [3], where a lower DBI indicates a better clustering algorithm. The effect of H on ranking aggregation accuracy is also evaluated. As for the *metapath2vec* algorithm adapted to compute category reliability, we test the effect of dataset size over ranking accuracy.

Baseline: The baselines to compare with AIC-BIC-based X-means (ABX) are as follows:

(1) AIC based X-means (AX), which only uses AIC as cluster splitting criterion;

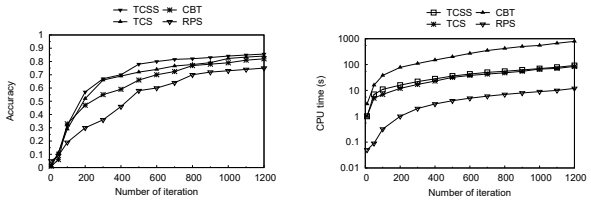
(2) BIC based X-means (BX), which only uses BIC as cluster splitting criterion;

(3) K-means with k=5 (K5), which splits the samples with k-means and set k as constant 5.



(a) (b)

Figure 6: Effect of H (LV)



(a) (b)

Figure 7: Effect of Iteration Number (LV)

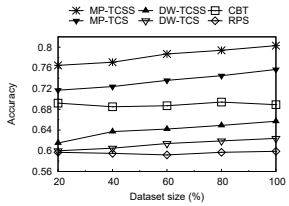


Figure 9: Effect of Dataset Size (LV)

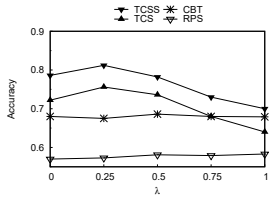


Figure 10: Effect of λ (LV)

The upper bound of cluster number for X-means algorithm is set as 10. Since H doesn't have influence on baseline clustering algorithms, we only report accuracy results on AIC-BIC based X-means when H varies. The baselines include 1) ABX based Random Pair Select (ABX-RPS), where RPS randomly selects a pair and a worker in each iteration of active learning phase; 2) ABX based CBT (ABX-CBT). As for the *metapath2vec* algorithm, experiments on Deepwalk (DW) [16] are conducted as a baseline. We implement *metapath2vec* (MP) and Deepwalk with the embedding size = 128, walks per vertex = 80 and walk length = 40.

Effect of H. As demonstrated in Figure 6, K-means (k=5) has a higher DBI score than all the X-means based algorithms, which proves that when the dataset is unbalance, ABX could lead to more effective clustering results. The performance of ABX decreases before H reaches a point around 200 and increases again after that. This implies that for the current dataset there might be at least one suitable H.

Effect of dataset size. Figure 9 shows the experimental results on various dataset sizes, where $x\%$ means for each worker, we only use $x\%$ of her total check-in records in the embedding algorithm. As expected, with dataset size increasing, the ranking accuracy of TCS/TCSS related algorithms increase. And the accuracy of *metapath2vec* outperforms Deepwalk significantly. This is because *metapath2vec* can capture the heterogeneity of check-in network more effectively.

With the experiments of varying H and dataset size, we draw a conclusion that an accurate evaluation of worker quality can result in a higher ranking accuracy.

4.1.3 Experiments on Ranking Aggregation Phase. The last set of experiments evaluate the effect of iteration number and λ on ranking accuracy. We evaluate the ranking accuracy and CPU time cost over iterations (one successful task assignment is considered as an iteration, i.e., selecting a proper task-worker triple) to show that how the TCSS and TCS algorithms perform on efficiency. Moreover, we also report how the supervision of P-POI ranking contributes to the active learning process. λ varies from 0 to 1 denotes there is more attention paid to worker reliability on H-POIs or P-POIs.

Baseline. The baseline methods include 1) CrowdBT (CBT), which is the active learning strategy described in Section 3.3.1; 2) Random Pair Selection (RPS), where RPS randomly selects a pair and a worker in each iteration.

Effect of iteration number. Figure 7 shows the accuracy and time cost with number of iteration increases. Pair selection strategies with TCS and TCSS achieve a higher accuracy than CBT and RPS. We observe that it takes CBT 1.67 times of iterations of TCSS to reach the same accuracy level before converge. From Figure 7 (b), it is evident that TCS and TCSS have a significant lower time cost than CBT. As mentioned in Section 4.3, the lower the number of tasks distributed to the crowd, the less cost there will be. Thus TCS and TCSS are more effective and economic.

The better performance of TCS and TCSS is probably resulted from 1) when the number of iteration is small (e.g. < 600), TCS and TCSS use worker quality data trained in Section 3.2 and TCSS leverages the supervision of P-POI ranking to properly select qualified worker, while CBT, could not estimate an accurate parameters on worker quality; 2) TCS and TCSS both implement a tree-constrained skip approach. In the Minimum Spanning Tree (MST), all to-be-ranked POIs (nodes) are connected with the most informative pairs (edges), which naturally reduces searching space. As for RPS, since there is no need to "select" a task-worker triple, it is less time consuming but it has the lowest accuracy as a side effect.

Effect of λ. We also evaluate the performance of ranking aggregation by varying λ. We only report λ's effect on ranking accuracy since λ won't influence time cost. As demonstrated in Figure 10, when λ increases, ranking accuracies of TCS and TCSS first increase and then decrease, while the performance of CBT and RPS remains relatively stable. An interpretation for this observation is that when λ is large, the emphasize on worker active area quality of P-POIs becomes strong. However, since the service region of P-POI is larger than that of a H-POI, a stress of "workers' familiarity on P-POI" would not contribute much to our goal for finding a pair and a worker as well as improving the current ranking. Therefore, we suggest λ to be set around 0.25 to balance the importance of worker reliability on H-POIs and P-POI.s

4.2 Experimental Results on Phoenix Dataset

We test the same set of parameters on PN dataset and observe similar results. The main differences with the LV dataset include that 1) in MinVTS, the number of P-POIs is larger and the number of P-POIs selected by MG is closer to that by RG. This is because PH is a smaller dataset and the distribution of POIs is more sparse; 2) the performance of TCSS excesses that of TCS more significantly on accuracy. This is due to the fact that when the connectivity of graph is not too strong, supervision of P-POI ranking can help to

select a proper next pair. Due to space limit, we omit the display of these results in this version.

5 RELATED WORK

5.1 Crowdsourced Top-k

Crowdsourced top-k has attracted increasing attention due to the popularity of crowdsourcing platforms. Given a set of objects, the major goal of crowdsourced top-k algorithm is to infer the top-k objects based on the crowdsourced comparison results. Ways to aggregate high-quality answer from the crowd have been studied independently by database community and machine learning community. The database community focuses more on heuristic-based solutions. For instance, [7, 17] construct digraphs to describe the pairwise results and compute the score by evaluating in-degree and out-degree of each vertex, which represents an object. [7] also proposes an approach to obtain the score through iteratively eliminating lower ranked objects until k objects left. [12] proposes approaches to hybrid two types of tasks, rating and ranking. Machine learning community implements learning-based techniques to estimate the scores. Chen et al. [23] proposes a CrowdBT model by taking worker quality into consideration. Pfeiffer et al. [18] propose CrowdGauss model that assumes the scores are in Gauss distribution to estimate the scores by maximizing the product of probability of $s_i > s_j$, where s_i denotes the score of object o_i, and the times of $o_i > o_j$ reported by the crowd.

5.2 Spatial Crowdsourcing

Spatial Crowdsourcing (SC) can be deemed as one of the main enablers of urban computing applications. According to the task publish mode, SC can be classified into two categories, namely Server Assigned Tasks (SAT) mode and Worker Selected Tasks (WST) mode. Most of the current research carried out so far has been focused on SAT mode, where the SC-server takes charge of the task assignment process. In SAT mode, the server assigns proper tasks to workers in order to achieve some optimization goals such as maximizing the number of assigned tasks after collecting all the locations of workers at each timestamp [4, 9, 19]. It is a natural problem in SC to distribute proper tasks to appropriate workers. Spatial and temporal information of tasks and workers is taken into consideration in most spatial crowdsourcing studies [15, 19, 20, 27, 28]. However, distinguishing from the traditional spatial crowdsourced task, our work mainly concerns on novel techniques to generate highly informative pair-wise tasks and estimate the scores of H-POIs based on the answers with different qualities. Moreover, in our work, spatial experience of a worker is also crucial to contribute reliability of results.

6 CONCLUSION

In this work we have analyzed the necessities of H-POI exploration and proposed a framework that can aggregate H-POI ranking from pairwise comparisons of the crowd. Our solution starts with two offline operations, first of which is finding the MinVTS as initial task candidates and the second is computing worker category and geo-spatial quality using her historical check-in data in LBSNs. In the subsequent, through an active learning strategy we assign proper tasks to proper workers to aggregate the ranking online. Extensive experiments based on real POI-review datasets are conducted and the favorable results confirm that the accuracy of ranking can be enhanced with even a small number of iterations.

ACKNOWLEDGMENTS

Kai Zheng was supported by NSFC (61836007, 61832017, 61532018) and Alibaba Innovation Research. Bin Yao was supported by NSFC (61872235, 61729202, U1636210), and the National Key Research and Development Program of China (2018YFC1504504).

REFERENCES

[1] Hirotogu Akaike. 1998. *Information Theory and an Extension of the Maximum Likelihood Principle.* Springer New York, 199–213.
[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*. 17–23.
[3] David L Davies and Donald W Bouldin. 1979. A Cluster Separation Measure. *IEEE Trans Pattern Anal Mach Intell* PAMI-1, 2 (1979), 224–227.
[4] Dingxiong Deng, Cyrus Shahabi, and Linhong Zhu. 2015. Task matching and scheduling for multiple workers in spatial crowdsourcing. In *SIGSPATIAL*. 21.
[5] Yuxiao Dong, Nitesh V. Chawla, and Ananthram Swami. 2017. Metapath2Vec: Scalable Representation Learning for Heterogeneous Networks. In *KDD*. 135–144.
[6] Brian Eriksson. 2013. Learning to Top-K Search using Pairwise Comparisons. In *AISTATS*, Vol. 31. 265–273.
[7] Stephen Guo, Aditya Parameswaran, and Hector Garcia-Molina. 2012. So Who Won?: Dynamic Max Discovery with the Crowd. In *SIGMOD*. 385–396.
[8] Tsunenori Ishioka. 2000. Extended K-means with an Efficient Estimation of the Number of Clusters. In *ICML*.
[9] Leyla Kazemi, Cyrus Shahabi, and Lei Chen. 2013. GeoTruCrowd: Trustworthy Query Answering with Spatial Crowdsourcing. In *SIGSPATIAL*. 314–323.
[10] Asif R. Khan and Hector Garcia-Molina. 2014. *Hybrid Strategies for Finding the Max with the Crowd: Technical Report.* Technical Report. Stanford University.
[11] Joseph B Kruskal. 1956. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proc. Amer. Math. Soc.* 7, 1 (1956), 48–50.
[12] Kaiyu Li, Xiaohang Zhang, and Guoliang Li. 2018. A Rating-Ranking Method for Crowdsourced Top-k Computation. In *SIGMOD*. 975–990.
[13] Defu Lian, Zhao Cong, Xie Xing, Guangzhong Sun, Enhong Chen, and Rui Yong. 2014. GeoMF: joint geographical modeling and matrix factorization for point-of-interest recommendation. In *KDD*.
[14] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *CoRR* abs/1301.3781 (2013).
[15] Cheng Peng, Lian Xiang, Chen Lei, and Cyrus Shahabi. 2017. Prediction-Based Task Assignment on Spatial Crowdsourcing. In *ICDE*. 997–1008.
[16] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. *CoRR* abs/1403.6652 (2014).
[17] J.-C. Pomerol and S. Barba-Romero. 2012. Sophisticated voting with information for two voting functions. *Springer Science and Business Media* 25 (2012).
[18] Y. Chen A. Mao T. Pfeiffer, X. A. Gao and D. G. Rand. 2012. Adaptive polling for information aggregation. In *AAAI*.
[19] Y. Tong, L. Chen, Z. Zhou, H. V. Jagadish, L. Shou, and W. Lv. 2018. SLADE: A Smart Large-Scale Task Decomposer in Crowdsourcing. *IEEE Transactions on Knowledge and Data Engineering* 30, 8 (2018), 1588–1601.
[20] Yongxin Tong, Jieying She, Bolin Ding, Libin Wang, and Chen Lei. 2016. Online mobile Micro-Task Allocation in spatial crowdsourcing. In *ICDE*. 49–60.
[21] F.L. Wauthier, Michael Jordan, and N Jojic. 2013. Efficient ranking from pairwise comparisons. *ICML* (01 2013), 1146–1154.
[22] Y. Yao X. Jiang, L. Lim and Y. Ye. 2011. Statistical ranking and combinatorial hodge theory. 127, 1 (2011), 203–244.
[23] Chen Xi, Paul N. Bennett, Kevyn Collins-Thompson, and Eric Horvitz. 2013. Pairwise ranking aggregation in a crowdsourced setting. In *WSDM*. 193–202.
[24] Mao Ye, Peifeng Yin, Wang-Chien Lee, and Dik-Lun Lee. 2011. Exploiting Geographical Influence for Collaborative Point-of-interest Recommendation. In *SIGIR*. ACM, New York, NY, USA, 325–334.
[25] J-D Zhang and C-Y Chow. 2015. GeoSoCa: Exploiting Geographical, Social and Categorical Correlations for Point-of-Interest Recommendations. In *SIGIR*.
[26] X. Zhang, G. Li, and J. Feng. 2016. Crowdsourced Top-k Algorithms: An Experimental Evaluation. *Proc. VLDB Endow.* 9, 8 (April 2016), 612–623.
[27] Yan Zhao, Yang Li, Yu Wang, Han Su, and Kai Zheng. 2017. Destination-aware Task Assignment in Spatial Crowdsourcing. In *CIKM*. 297–306.
[28] Yan Zhao, Jinfu Xia, Guanfeng Liu, Han Su, Defu Lian, Shuo Shang, and Kai Zheng. 2019. Preference-aware Task Assignment in Spatial Crowdsourcing. In *AAAI*.

A NOTATIONS

Table 2 summarizes the major notifications of this paper.

Table 2: Summary of Notations

Notation	Definition
p	A Point-of-Interest (POI)
p_h	A Hidden POI (H-POI)
p_p	A Popular POI (P-POI)
P_h	A H-POI set
P_p	A P-POI set
l_p	Location of POI p
C_p	Category of POI p
r_p	Service region of POI p
w	A worker
W	A Worker set
s	A spatial task
S	A Valid Spatial Task Set (VTS)
S_{min}	A Minimum Valid Spatial Task set (MinVTS)
$d(a, b)$	Euclidean distance between a and b
κ	Lower bound in Comparable H-POI conditions
H	Criterion determination threshold in A-B X-means algorithm
λ	Tuning parameter of the trade-off between worker's location reliability importance on H-POIs and P-POIs
α_w^C	Reliability of worker w on category C
$\beta_w^{l_s}$	Reliability of worker w on locations of POIs in task s
η_w^s	Overall reliability of worker w on task s

B PSEUDO CODE

B.1 MinVTS Greedy Search Algorithm

The pseudo code for MinVTS Greedy Search algorithm is displayed in Algorithm 1.

B.2 AIC-BIC Based X-means Algorithm

The pseudo code for AIC-BIC Based X-means (A-B X-means) algorithm is displayed in Algorithm 2.

B.3 Tree-constrained Skip Search Algorithm

The pseudo code for Tree-constrained Skip (TCS) Search algorithm is displayed in Algorithm 3. Input parameter B describes total iteration number budget and is set as infinity in above experiments for evaluation purpose.

C PROOF OF THEOREM 3.1

proof: Assume there are n objects in total, the current ranking accuracy is Acc, we next analyze the Δ of accuracy after updating object i. Assume the ranking change of i is K, without generality, we assume $K > 0$, which means after updating, the ranking of i increases K. Denoting that in the K objects, there are k_1 objects having a higher ground-truth score than $i$and k_2 objects having a lower ground-truth score than i, where k_1 is of uniform distribution $U(0, K)$ and $k_1 + k_2 = K$. Then Δ can be calculated as $\frac{k_1 - k_2}{n(n+1)} =$

Algorithm 1: MinVTS Greedy Search

Input: H-POI set P_h, P-POI set P_p, threshold κ
Output: MinVTS $S_{min} = \{s_1, s_2, ..., s_i\}$
Find all Comparable H-POIs and append to a set P_{ch} and all their matchable P-POIs. Construct an adjacent matrix T, where $T_{i,j} = 1$ if pair < i, j > is matched in above process;
Initialize an empty set S_{min}={ };
Initialize $M_{1\times|P_{ch}|} = 0$, $N_{1\times|P_p|}$;
min ← the minimum entry of M;
I_{min} ← indexes of entries with value min in M;
while $min < \kappa$ **do**
  for ***each*** p_p ***in*** N **do**
    for ***each*** p_h ***in*** I_{min} **do**
      Add T_{p_h,p_p} to N_{p_p};
    end
  end
  $tmp\_P_{p_{max}}$ ← indexes of entries with the maximum value in N;
  if $|tmp\_P_{p_{max}}| > 0$ **then**
    A similar accumulation as above but with N ← $p_{p_{max}}$, and M ← M \ { p_h | with κ P-POIs matched }
  end
  $p_{p_{max}} \leftarrow arg\max_{p_p}\{N\}$;
  $S_{min} \leftarrow S_{min} \cup \{ < p_h, p_{p_{max}} > | < p_h, p_{p_{max}} > \text{is valid}\}$;
  $N \leftarrow N \setminus p_{p_{max}}$, $P_{ch} \leftarrow P_{ch} \setminus \{ p_h \mid M_{p_h} \geq \kappa\}$;
  Reinitialize N;
  Update M, min and I_{min};
end

Algorithm 2: A-B X-means

Input: Location set of a worker $P = \{l_i, l_2, ..., l_n\}$, upper bound of number of clusters: K, criterion choosing threshold H.
Output: Set of optimal x centroids $O = \{o_1, o_2, ..., o_x\}$
Initialize: k ← 1; O={ }; Clst ← P;
if $|P| > H$ **then**
  criterion metric is set as BIC;
end
else
  criterion metric is set as AIC;
end
while *Clst is not empty and* $k < K$ **do**
  for ***each*** *sub-Clst* ***in*** *Clst* **do**
    Run k-means on points in sub-Clst with $k' = 2$;
    Compute criterion score of the new splitting ;
    if *new splitting is better* **then**
      Update O;
      k ← k+1;
    end
  end
  Update Clst with split sub-Clsts;
end

Algorithm 3: TCS Algorithm

Input: MinVTS, worker set W, prior distribution parameters $\{\mu_p\}$, $\{\sigma_p\}$ of each POI in MinVTS, the total budget B
Output: POI Ranking by sorting $\{\mu_p\}$
Construct adjacent matrix M_A for MinVTS;
t← 0;
Find all connected components and run Kruskal algorithm on each component to obtain the MSTs;
while $t<B$ **do**
  Select a pair $(p_h, p_p) \in E(MSTs)$ and a worker $w \in W$ that maximize Equation 13 ;
  Ask the worker one her preference between p_h and p_p;
  Update $\{\mu_{p_h}\}$, $\{\sigma_{p_h}\}$ $\{\mu_{p_p}\}$, $\{\sigma_{p_p}\}$ based on her report;
  t←t+1;
  if $t<B$ **then**
    Select a pair $(p_h, p'_p) \in$ the adjacent set of p_h ,and a worker $w \in W$ that maximize Equation 13;
    Update scores according to the worker's report;
    t←t+1;
    if $t<B$ **then**
      Select a pair $(p'_h, p_b) \in$ the adjacent set of p_p and a worker $w \in W$ that maximize Equation 13;
      Update scores according to the worker's report;
      t←t+1;
    end
  end
end

$\frac{2k_1-K}{n(n+1)}$, which is a uniformly distributed random variable with $U(-\frac{K}{n(n+1)}, \frac{K}{n(n+1)})$. In a similar way, we can obtain Δ in the case of $K < 0$. Theorem is proved.

D EXPERIMENT PARAMETERS

D.1 Experiment Setup

We use a real public crowd-sourced local business reviews dataset generated by Yelp[3]. The Yelp dataset spanning over 11 metropolitan areas in four aspects: business, check-in, user and review. Each POI has a POI ID, a location (in the form of latitude and longitude), a city tag, category tags (with several sub-categories), a score (between 0 to 5) and other business (POI) related information. Each review is tagged with a review ID, a user ID, a POI ID, score (between 0 to 5) and other context or date related information. Thus, the reviews can be taken as check-in records. We extract the data in Las Vegas (LV) and Phoenix (PN), USA, which contains the most POIs reviewed. The dataset statistics are listed in Table 3. We set thresholds $\Theta = 50$ to identify P-POIs. And the rating score for P-POIs are assumed valid. As for H-POIs, in order to evaluate the ranking accuracy with ground-truth, we take POIs with number of check-ins in range (30,50] (different from the H-POI definition in Section 2 for evaluation purpose) as H-POIs and the ranking generated by sorting their scores is regarded as ground-truth. For the nature of LBSNs, every user can be regarded as a crowdsourcing worker. To simulate the situation that real H-POIs share poor information in the database, unless otherwise specified, all H-POIs related data are excluded from the dataset in following experiments.

We re-categorized all the POIs into the highest categories generated by Yelp[4]. Since the answer of a crowdsourcing task is a decision supposed to be based on the subjective quality of POIs, we generate workers' answers in a category-location aware way. For each worker and category, we first compute the worker's rating of all the target H-POIs in her reviewing records over the sum of corresponding H-POIs' overall scores. If it is larger than 0.5, then we consider the worker is reliable on this category. As for area reliability, if she ever had check-in records within the circles centered by P-POI and H-POI respectively and with a radius of 0.005 (measure by Euclidean distance of latitude and longitude), which means she is highly likely to be familiar with both POIs. If a worker satisfies both above conditions, we assume she will return a correct answer (determined by POIs's overall scores, i.e., if $s_{p_{h_i}} < s_{p_{p_j}}$ then the correct answer is $p_{h_i} \prec p_{p_j}$), otherwise a wrong one.

Table 3: Statistic of dataset

Attribute	Las Vegas	Phoenix
Total number of POIs	28865	18633
Number of H-POIs	5042	2865
Number of P-POIs	6277	2792
Total number of categories	1982	1865
Total number of users	86480	23871

Table 4: Parameter settings of LV dataset

Parameter	Values (default in bold)
Size of dataset	20%,40%,60%,80%,**100%**
Criterion determination threshold H	50,100,**150**,200,250
Number of P-POIs per H-POI κ	1,2,**3**,4,5
λ	0.00,0.25,**0.50**,0.75,1.00
Number of iterations(quries)	400,600,**800**,1000,1200
Number of sampled H-POIs	200,**400**,600,800,1000

D.2 Parameter settings of LV dataset

Table 4 displays the evaluated parameters of LV dataset. Considering the scale of Las Vegas, the mean of POI service region is set as: $\mu_{p_p} = 0.065$ and $\mu_{p_h} = 0.0065$. We assume worker reliability on task categories and locations are equally important. Thus, ϵ is set as 0.5. All values of accuracy and CPU time are based on 800 queried tasks.

[3]https://www.kaggle.com/yelp-dataset/yelp-dataset

[4]https://www.yelpblog.com