GridTuner: Reinvestigate Grid Size Selection for Spatiotemporal Prediction Models

Jiabao Jin *, Peng Cheng *, Lei Chen †, Xuemin Lin #,*, Wenjie Zhang #
*East China Normal University, Shanghai, China
jiabaojin@163.com, pcheng@sei.ecnu.edu.cn
†The Hong Kong University of Science and Technology, Hong Kong, China
leichen@cse.ust.hk
#The University of New South Wales, Australia
lxue@cse.unsw.edu.au, wenjie.zhang@unsw.edu.au

***Abstract*—With the development of traffic prediction technology, spatiotemporal prediction models have attracted more and more attention from academia communities and industry. However, most existing researches focus on reducing model's prediction error but ignore the error caused by the uneven distribution of spatial events within a region. In this paper, we study a region partitioning problem, namely optimal grid size selection problem (OGSS), which aims to minimize the real error of spatiotemporal prediction models by selecting the optimal grid size. In order to solve OGSS, we analyze the upper bound of real error of spatiotemporal prediction models and minimize the real error by minimizing its upper bound. Through in-depth analysis, we find that the upper bound of real error will decrease then increase when the number of model grids increase from 1 to the maximum allowed value. Then, we propose two algorithms, namely Ternary Search and Iterative Method, to automatically find the optimal grid size. Finally, the experiments verify that the error of prediction has the same trend as its upper bound, and the change trend of the upper bound of real error with respect to the increase of the number of model grids will decrease then increase. Meanwhile, in a case study, by selecting the optimal grid size, the order dispatching results of a state-of-the-art prediction-based algorithm can be improved up to 13.6%, which shows the effectiveness of our methods on tuning the region partition for spatiotemporal prediction models.**

I. Introduction

Recently, many spatiotemporal prediction models are proposed to predict the number of events (e.g., online car-hailing requests [1], [2], [3], or street crimes [4], [5]) within a region (e.g., a gird of 1km×1km) in a period (e.g., next 30 minutes) [6], [7], [8]. With the help of predicted information of events, we can improve the platform revenue of online car-hailing systems (e.g., Uber [9]), or reduce crimes effectively through optimizing patrol route of police [5].

One common assumption of spatiotemporal prediction models is that the distribution of spatial events within a region is uniform [1], [6], [2], which is in fact almost never true in real scenarios. In addition, the selection of region size is mostly decided by experts' experience or simple experimental tests without detailed analysis in many existing research studies:

- "We use 20×30 = 600 grids to cover the cities and one grid represents a 0.01 (longitude)×0.01 (latitude) square" [1]
- The authors use 32×32 grids to cover Beijing area and 16×8 grids for New York City area [6].

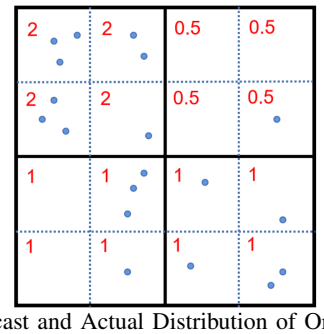


Fig. 1: Forecast and Actual Distribution of Orders in Grids

- "For the prediction model, DeepST, we set the default grid size as 2km×2km ..." [2]

Under the uniform region assumption, the spatiotemporal prediction models are optimized to minimize the model error in model grids (i.e., the difference between the predicted and actual number of spatial events in grids used in the prediction model), which may lead to dramatic real errors in some smaller regions (i.e., the difference between the predicted and the actual number of spatial events in smaller and homogeneous grids). Then, the overall performance of frameworks utilizing spatiotemporal prediction models will not be optimized for real applications. For example, with careful grid size selection, the overall performance of a state-of-the-art prediction based online spatial crowdsourcing framework [1] can be increased up to 13.6% (shown in a case study in Section V).

We illustrate this challenge with the following example:

Example 1. *As shown in Figure 1, the solid black lines divide the space into four model grids to be predicted. The blue dot lines further divide each model grid into four smaller grids. We can use spatiotemporal prediction models to predict the number of events in each model grid. In the absence of prior knowledge of the distribution of events within a model grid, the models generally assume that the distribution of events within a model grid is uniform, which means that the number of events in each smaller grid within a same model grid is equal to each other. Thus, we can estimate the predicted number of each smaller grid through averaging the predicted result of the corresponding model grid. The red number shown in Figure 1 denotes the predicted number of events for each smaller grid.*

The predicted result for each model grid is the summation of the number of events for all its smaller grids. We can directly calculate the model error of the prediction model on large grids is 3 *(=* $|8-9|+|2-1|+|4-4|+|4-5|$*). Nevertheless, if the model error is calculated based on smaller grids, it will increase to* 10 *(=* $|2-3|+|2-2|+|0.5-0|+|0.5-0|+|2-3|+|2-1|+|0.5-0|+|0.5-1|+|1-0|+|1-3|+|1-1|+|1-1|+|1-0|+|1-1|+|1-1|+|1-2|$*). The reason is that the distribution of events in each large region is uneven, which is ignored in almost all existing studies.*

Why not directly predict the spatial events for each smaller grids? The reason is twofold. Firstly, due to the uncertainty of spatial events, it is too hard to accurately predict the number of spatial events in a very small grid (e.g., 100m×100m). When the size of grid is too small, there will be no enough historical data for prediction model to learn the distribution of the spatial events in each small area. In addition, the number of spatial events in a small grid is also small, then the accuracy (relative error) of prediction models will be dramatically affected by the randomness of the spatial events, since the uncertainty of spatial events is inevitable [10], [1], [11]. Secondly, the computation complexity of prediction models will increase remarkably when the number of grids increase [10], [12]. Thus, almost all spatiotemporal prediction models still use relatively large grids (e.g., grids of 2km×2km) as the prediction units.

Can we have an automatic and theoretic-guaranteed optimal grid size selection method to minimize the overall real error of spatiotemporal prediction models? To overcome this long-standing challenge, in this paper we study the *optimal grid size selection* (OGSS) problem to guide the configuration of grid size such that the real errors are minimized for spatiotemporal prediction models in real applications.

In this paper, we reinvestigate the grid size selection problem in spatiotemporal prediction models in detail. We assume that the distribution of the spatial events in a small enough grid (e.g., 100m×100m) can be considered homogeneous. Then, the real error of a spatiotemporal prediction model is evaluated through the total difference between the predicted number and real number of spatial events among all small grids. Specifically, we decompose the real error of the prediction models into the model error and the expression error. Here, the model error indicates the inherent error of the prediction models, and the expression error stands for the error of using the predicted number of events in a large grid to express the future events in its inner smaller grids (as illustrated in Example 1). We prove that the summation of model error and expression error of a spatiotemporal prediction model is an upper bound of its real error. We also verify that for any spatiotemporal prediction model, with the increase of the size of regions, the upper bound of its real error will first decrease then increase. Based on our theoretical analysis, we propose two algorithms, namely Ternary Search algorithm and Iterative algorithm, to quickly find the optimal size of model grids for a given spatiotemporal prediction model.

To summarize, we make the following contributions:

- We formally define a novel metric, real error, to measure the deviation between the model's forecast and the actual number of events for homogeneous grids. Then, we propose a new problem, namely optimal grid size selection (OGSS), to automatically find the optimal grid size for a given spatiotemporal prediction model in Section II
- We analyze the upper bound of the real error and the relationship between the number of model grids and the upper bound in Section III. Then, we propose two algorithms to find the optimal grid size to minimize the upper bound of real error in Section IV.
- We conduct sufficient experiments on different spatiotemporal prediction models to explore the influencing factors of real errors in Section V.

We review the related studies in Section VI and conclude this paper in Section VII.

II. Preliminaries

In this section, we will introduce some basic concepts and present the formal definition of model grid, homogeneous grid, model error, expression error and real error. We prove that the summation of model error and expression error is an upper bound of real error.

A. Basic Concepts

Without loss of generality, in this paper, we consider that the spatiotemporal prediction models will first divide the whole space into n rectangular model grids, then predict the number of spatial events for each model grid in a given future time period. When the size of a grid is small enough (e.g., 100m×100m), the distribution of spatial events can be considered uniform for most application scenarios (e.g., online car-hailing systems). Under this assumption, each model grid will be further divided into m smaller homogeneous grids, where predicted spatial events is considered uniformly distributed in each homogeneous grid. To ensure HGrids are small enough, it is required that the number of HGrids is larger than N (i.e., $mn > N$). We note N as the minimum number that makes the distribution of spatial events in each HGrid itself is uniform. We propose a method to select a proper value of N in Section III. We formally define the model grids and the homogeneous grids as follows:

Definition 1 (Model Grid, MGrid)**.** The whole space is divided into n same-sized model grids $\{r_1, r_2, \cdots, r_n\}$. The number of actual spatial events happening in r_i in the next period is noted as λ_i. A spatiotemporal prediction model will predict the number, $\hat{\lambda}_i$, of spatial events that happening in each model grid r_i in the next period.

Definition 2 (Homogeneous Grid, HGrid)**.** Each model grid r_i can be further evenly divided into m homogeneous grids $\{r_{i1}, r_{i2}, \cdots, r_{im}\}$. For a homogeneous grid r_{ij}, the number of actual spatial events happening in it in the next period is marked as λ_{ij} (i.e., $\lambda_i = \sum_{j=1}^{m} \lambda_{ij}$).

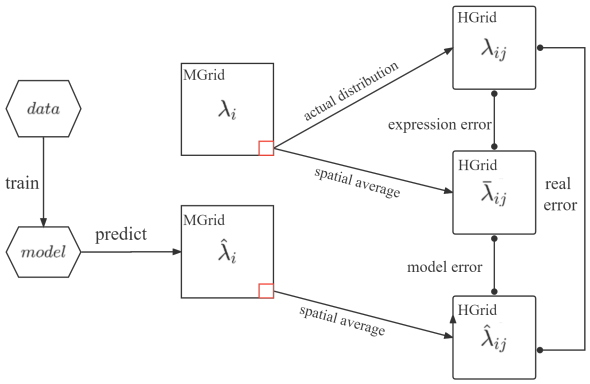


Fig. 2: An Illustration of Relationships between Basic Concepts.

In the absence of any prior knowledge of the distribution of the spatial events in a model, we assume that the number of spatial events of each HGrid in the MGrid is same to each other according to the principle of maximum entropy [13]. Thus, with the actual number, λ_i, of spatial events in MGrid r_i, the estimated number of spatial events of HGrid r_{ij} is denoted as $\bar{\lambda}_{ij} = \frac{\lambda_i}{m} = \frac{\sum_{j=1}^{m} \lambda_{ij}}{m}$. Similarly, with the predicted number, $\hat{\lambda}_i$, of spatial events of MGrid r_i, we can have the predicted number of spatial events of HGrid r_{ij} as $\hat{\lambda}_{ij} = \frac{\hat{\lambda}_i}{m}$.

The differences between λ_{ij}, $\bar{\lambda}_{ij}$ and $\hat{\lambda}_{ij}$ lead to three types of errors: model error, expression error and real error. Figure 2 illustrates the three types of errors. The frameworks utilizing spatiotemporal prediction models is hard to get the information about the real distribution of spatial events from the models. Thus, the prediction result $\hat{\lambda}_i$ of a MGrid r_i will be divided equally into HGrids without any prior information. The real error describes the difference between the actual number of spatial events λ_{ij} and the predicted result $\hat{\lambda}_{ij}$ of HGrid r_{ij}. However, it is difficult to calculate the real error directly. Therefore, $\bar{\lambda}_{ij}$ is introduced to decompose the real error into expression error and model error. We formally define the real error model error and expression error as follows:

Definition 3 (Real Error)**.** For a HGrid r_{ij}, its real error $E_r(i,j)$ is defined as the mean/average of difference between its predicted and actual numbers of spatial events in the corresponding same time periods of the historical days for the next period. It means $E_r(i,j) = \mathbb{E}_{\lambda_{ij}\sim P}(|\hat{\lambda}_{ij} - \lambda_{ij}|)$, where λ_{ij} follows a given distribution P.

In practice, it is difficult to calculate the difference $|\hat{\lambda}_{ij} - \lambda_{ij}|$ without the information about the number λ_{ij} of events in next period. Thus, we define the real error as the mean of the difference $|\hat{\lambda}_{ij} - \lambda_{ij}|$. However, due to the lack of a sufficient number of samples, it is difficult for us to calculate $E_r(i,j)$ accurately because λ_{ij} does not follow the same distribution for different time periods. Another factor is that the environment is prone to change over a long period, so the number of events does not follow the same distribution either. As a result, we use the number of events in the same time period on each day of the previous one month to estimate real error. Suppose that we have a set Λ_{ij} of the actual events number, λ_{ij}, and its corresponding prediction number, $\hat{\lambda}_{ij}$, we can estimate $E_r(i,j)$ as follows:

$$E_r(i,j) = \mathbb{E}_{\lambda_{ij}\sim P}(|\hat{\lambda}_{ij} - \lambda_{ij}|) = \frac{1}{|\Lambda_{ij}|} \sum_{(\hat{\lambda}_{ij},\lambda_{ij})\in\Lambda_{ij}} |\hat{\lambda}_{ij} - \lambda_{ij}|$$

Definition 4 (Model Error)**.** For a HGrid r_{ij}, its model error $E_m(i,j)$ is the mean/average of difference between its predicted and estimated numbers of spatial events in the corresponding same time periods of the historical days for the next period. It means $E_m(i,j) = \mathbb{E}_{\lambda_{ij}\sim P}(|\hat{\lambda}_{ij} - \bar{\lambda}_{ij}|)$, where λ_{ij} follows a given distribution P.

Definition 5 (Expression Error)**.** For a HGrid r_{ij}, its expression error $E_e(i,j)$ is the mean/average of difference between its estimated and actual numbers of spatial events in the corresponding same time periods of the historical days for the next period. It means $E_e(i,j) = \mathbb{E}_{\lambda_{ij}\sim P}(|\bar{\lambda}_{ij} - \lambda_{ij}|)$, where λ_{ij} follows a given distribution P.

In the previous analysis, we explained that the grid size selection would significantly affect the real error. This paper aims to find an optimal size that minimizes the summation of real errors in all HGrids. We formally define the problem as follows:

Definition 6 (Optimal Grid Size Selection Problem, OGSS)**.** For a given number of HGrids m in each MGrid, and a given model to predict the number of spatial events for MGrids in next period, the optimal grids size selection problem is to find the optimal n to minimize the summation of real error of all HGrids under the constraint $nm > N$, which is:

$$\begin{aligned} \min_{n} \quad & \sum_{i=1}^{n}\sum_{j=1}^{m} E_r(i,j) \\ \text{s.t.} \quad & nm > N \end{aligned} \tag{1}$$

where N represents the minimum required number of HGrids in the whole space.

B. Upper Bound of Real Error

We denote the summation of model error and expression error as $E_u(i,j)$ $(= E_m(i,j) + E_e(i,j))$. We can prove that $E_u(i,j)$ is an upper bound on $E_r(i,j)$ by the theorem II.1.

Theorem II.1 (Upper Bound of Real Error)**.** *$E_u(i,j)$ is an upper bound of real error $E_r(i,j)$*

Proof. We prove it through the following inequalities:

$$\begin{aligned} E_r(i,j) &= \mathbb{E}\left(|\hat{\lambda}_{ij} - \lambda_{ij}|\right) = \mathbb{E}\left(|\hat{\lambda}_{ij} - \bar{\lambda}_{ij} + \bar{\lambda}_{ij} - \lambda_{ij}|\right) \\ &\le \mathbb{E}\left(|\hat{\lambda}_{ij} - \bar{\lambda}_{ij}| + |\bar{\lambda}_{ij} - \lambda_{ij}|\right) \\ &= \mathbb{E}\left(|\hat{\lambda}_{ij} - \bar{\lambda}_{ij}|\right) + \mathbb{E}\left(|\bar{\lambda}_{ij} - \lambda_{ij}|\right) \\ &= E_m(i,j) + E_e(i,j) = E_u(i,j) \end{aligned}$$

□

Meanwhile, we obtain the upper bound of the difference between $E_u(i,j)$ and $E_r(i,j)$ by the following scaling:

$$\begin{aligned} E_u(i,j) - E_r(i,j) &\le \mathbb{E}\left(2\min\left(|\hat{\lambda}_{ij} - \bar{\lambda}_{ij}|, |\bar{\lambda}_{ij} - \lambda_{ij}|\right)\right) \\ &= 2\min\left(\mathbb{E}\left(|\bar{\lambda}_{ij} - \lambda_{ij}|\right), \mathbb{E}\left(|\hat{\lambda}_{ij} - \bar{\lambda}_{ij}|\right)\right) \\ &= 2\min\left(E_e(i,j), E_m(i,j)\right) \end{aligned}$$

This indicates that we can ensure that the $E_r(i,j)$ is small when $E_u(i,j)$ is minimized. Therefore, we will minimize $E_u(i,j)$ as much as possible to optimize OGSS in the

TABLE I: Symbols and Descriptions.

Symbol	Description
r_i	a MGrid
r_{ij}	a HGrid in MGrid r_i
n	the number of MGrids
m	the number of HGrids for each MGrid
$\bar{\lambda}_{ij}$	the estimated number of spatial events for HGrid r_{ij}
λ_{ij}	the actual number of spatial events in HGrid r_{ij}
λ_i	the actual number of spatial events in MGrid r_i
$\hat{\lambda}_i$	the prediction of λ_i
α_{ij}	the temporal mean of λ_{ij}
$E_r(i,j)$	the real error of HGrid r_{ij}
$E_e(i,j)$	the expression error of HGrid r_{ij}
$E_m(i,j)$	the model error of HGrid r_{ij}

following sections of this paper. Finally, Table I shows some important notations used in this paper.

III. Error Analysis

In this section, we first explain how to select a proper N such that each HGrid is small enough and can be considered uniform. Then, we discuss the property of expression error and propose two algorithms to quickly calculate expression error. Finally, we analyze the model error.

A. Select A Suitable N

We explain how to choose a suitable N in this section. Most of spatiotemporal prediction models are based on experience to divide the whole space into many same-sized MGrids (e.g., 2km×2km grid). However, these methods ignore the uneven distribution of spatial events within a MGrid.

We divide the whole space into $\sqrt{N} \times \sqrt{N}$ (i.e., $nm = N$) same-sized HGrids. Let α_{ij} be the mean number of events for HGrid r_{ij} in the next period, which can be estimated as the average number of the historical records (i.e., nearest one month's data) of r_{ij}.

Here, we give the definition of the uniformly distribution for a grid as follows:

Definition 7 (Uniformly Distributed Grid)**.** Given a grid r_{ij} with the expected spatial events number α_{ij} and a positive integer $K \in \mathbb{Z}^+$, we divide the grid into K smaller grids with the expected spatial events number $\alpha_{ijk}, k = 1, 2, ..., K$. Grid r_{ij} is **uniformly distributed** if and only if $\alpha_{ijk} = \frac{\alpha_{ij}}{K}$ for any $1 \le k \le K$.

Then, we introduce a metric to measure the degree of the uneven distribution for spatial events in HGrids, which is defined as the following formula:

$$D_\alpha(N) = \sum_{i=1}^{n} \sum_{j=1}^{m} |\alpha_{ij} - \bar{\alpha}_N|, \tag{2}$$

where $\bar{\alpha}_N = \frac{1}{N} \sum_{i=1}^{n} \sum_{j=1}^{m} \alpha_{ij}$. We notice that when N increases, $D_\alpha(N)$ will also increase. However, when N is large enough (i.e., spatial events can be considered evenly distributed in each HGrid), $D_\alpha(N)$ will not significantly increase any more. We prove this with the following theorem:

Theorem III.1. *Assume that N is suitable (sufficiently large) such that the spatial events are uniformly distributed in each HGrid, then $D_\alpha(N) = D_\alpha(NK)$, for any $K \in \mathbb{Z}^+$.*

Proof. We divide each HGrid r_{ij} into smaller grids denoted as r_{ijk} $(k = 1, 2, \ldots, K)$, where the mean number of events for each smaller grid is denoted as α_{ijk}. Due to the uniformity of each HGrid, we have $\alpha_{ijk} = \frac{\alpha_{ij}}{K}$. Thus, we have

$$\begin{aligned} D_\alpha(NK) &= \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{K} |\alpha_{ijk} - \bar{\alpha}_{NK}| \\ &= \sum_{i=1}^{n} \sum_{j=1}^{m} K \left| \frac{\alpha_{ij}}{K} - \frac{1}{K} \bar{\alpha}_N \right| = D_\alpha(N) \end{aligned} \tag{3}$$

□

The increase of N will not contribute to the increase of $D_\alpha(N)$ when N is large enough, which means that $D_\alpha(N)$ can be an indicator to help us to select N. In other words, we should choose a sufficiently large N so that $D_\alpha(N)$ is maximized in practice.

B. Analysis and Calculation of Expression Error

We assume that the number λ_{ij} of events in a HGrid r_{ij} follows a poisson distribution $Pois$ with parameter of α_{ij} (α_{ij} is the mean number of events for HGrid r_{ij}), which is verified in our previous work [3], [14].

Calculation of Expression Error. We first analyze how to calculate expression error for a given HGrid r_{ij}. Due to $\lambda_{ij} \sim Pois(\alpha_{ij})$, we have

$$P(\lambda_{ij} = k_h) = e^{-\alpha_{ij}} \frac{{\alpha_{ij}}^{k_h}}{k_h!}, k_h \in \mathbb{N} \tag{4}$$

Then, we define the random variable $\lambda_{i,\neq j}$ as the mean of the number of events for all HGrids in MGrid r_i excluding the HGrid r_{ij} (i.e., $\lambda_{i,\neq j} = \sum_{g \neq j} \lambda_{ig}$), and have $\lambda_{i,\neq j} \sim Pois(\sum_{g \neq j} \alpha_{ig})$ because of the additivity of Poisson distribution. Let $\bar{\lambda}_{i,\neq j} = \frac{1}{m} \lambda_{i,\neq j}$, we have

$$P(\bar{\lambda}_{i,\neq j} = \frac{k_m}{m}) = e^{-\sum_{g \neq j} \alpha_{ig}} \frac{\left(\sum_{g \neq j} \alpha_{ig}\right)^{k_m}}{k_m!}, k_m \in \mathbb{N} \tag{5}$$

Since λ_{ij} and $\bar{\lambda}_{i,\neq j}$ is independent of each other, $P(|\bar{\lambda}_{ij} - \lambda_{ij}|)$ can be expressed by

$$\begin{aligned} P(|\bar{\lambda}_{ij} - \lambda_{ij}| = \frac{k_d}{m}) &= P(|\frac{m-1}{m} \lambda_{ij} - \bar{\lambda}_{i,\neq j}| = \frac{k_d}{m}) \\ &= \sum_{|\frac{m-1}{m} k_h - \frac{k_m}{m}| = \frac{k_d}{m}} P(\lambda_{ij} = k_h) P\left(\bar{\lambda}_{i,\neq j} = \frac{k_m}{m}\right) \\ &= \sum_{\frac{(m-1)k_h - k_m}{m} = \pm \frac{k_d}{m}} p(r_{ij}, k_h, k_m) \end{aligned} \tag{6}$$

where $p(r_{ij}, k_h, k_m) = e^{-\sum_{j=1}^{m} \alpha_{ij}} \frac{(\sum_{g \neq j} \alpha_{ig})^{k_m} (\alpha_{ij})^{k_h}}{k_m! k_h!}$ denoting the probability when the number of events in HGrid r_{ij} is k_h and the number of events in MGrid r_i is $k_h + k_m$. Then, we have:

$$\begin{aligned} E_e(i,j) &= \mathbb{E}(|\lambda_{ij} - \bar{\lambda}_{ij}|) = \sum_{k_d=0}^{\infty} \frac{k_d}{m} P(|\lambda_{ij} - \bar{\lambda}_{ij}| = \frac{k_d}{m}) \\ &= \sum_{k_d=0}^{\infty} \frac{k_d}{m} \sum_{\frac{(m-1)k_h - k_m}{m} = \pm \frac{k_d}{m}} p(r_{ij}, k_h, k_m) \\ &= \sum_{k_h=0}^{\infty} \sum_{k_m=0}^{\infty} b_{k_h k_m} \end{aligned} \tag{7}$$

where $b_{k_h k_m} = \left| \frac{(m-1)k_h - k_m}{m} \right| p(r_{ij}, k_h, k_m)$. Here, $p(r_{ij}, k_h, k_m)$ represents the probability when the number of events in MGrid r_i is $k_m + k_h$ and the number of events in HGrid r_{ij} is k_h, and the single expression error of HGrid r_{ij} in this situation is $\left| \frac{(m-1)k_h - k_m}{m} \right|$. Thus, Equation 7 can be regarded as a weighted average of the single expression error in all possible cases.

We can use Equation 7 to calculate the expression error of HGrid r_{ij}, which also indicates that the expression error is only related to the α_{ij} of each HGrid r_{ij} and m.

Properties of expression error. We show that the upper bound of expression error $E_e(i,j)$ is positively correlated to α_{ij} and m. In other words, when α_{ij} or m increases, the upper bound of expression error $E_e(i,j)$ will also increase. This relationship is presented with the following lemma:

Lemma III.1. *$\forall M_1, M_2 \in \mathbb{Z}^+$, we have*

$$\sum_{k_h=0}^{M_1}\sum_{k_m=0}^{M_2} b_{k_h k_m} < (1-\frac{2}{m})\alpha_{ij} + \frac{\sum_{k=1}^{m}\alpha_{ik}}{m}.$$

Proof.

$$\sum_{k_h=0}^{M_1}\sum_{k_m=0}^{M_2} b_{k_h k_m} = \sum_{k_h=0}^{M_1}\sum_{k_m=0}^{M_2}\left|\frac{(m-1)k_h - k_m}{m}\right| p(r_{ij},k_h,k_m)$$

$$\leq \sum_{k_h=0}^{M_1}\sum_{k_m=0}^{M_2}\left(\frac{(m-1)k_h}{m}+\frac{k_m}{m}\right) p(r_{ij},k_h,k_m) \quad (8)$$

Considering the first term of right hand side of Inequation 8, we have

$$\sum_{k_h=0}^{M_1}\sum_{k_m=0}^{M_2}\frac{(m-1)k_h}{m} p(r_{ij},k_h,k_m)$$

$$= \sum_{k_h=0}^{M_1}\sum_{k_m=0}^{M_2}\frac{(m-1)k_h}{m} e^{-\sum_{j=1}^{m}\alpha_{ij}}\frac{(\sum_{g\neq j}\alpha_{ig})^{k_m}(\alpha_{ij})^{k_h}}{k_m! k_h!}$$

$$= \frac{(m-1)}{m}\sum_{k_h=1}^{M_1}\frac{e^{-\alpha_{ij}}(\alpha_{ij})^{k_h}}{(k_h-1)!}\sum_{k_m=0}^{M_2} e^{\sum_{g\neq j}\alpha_{ig}}\frac{(\sum_{g\neq j}\alpha_{ig})^{k_m}}{k_m!} \quad (9)$$

$$< \frac{(m-1)}{m}\sum_{k_h=1}^{M_1}\frac{e^{-\alpha_{ij}}(\alpha_{ij})^{k_h}}{(k_h-1)!} \quad (10)$$

$$= \frac{(m-1)\alpha_{ij}}{m}\sum_{k_h=1}^{M_1}\frac{e^{-\alpha_{ij}}(\alpha_{ij})^{k_h-1}}{(k_h-1)!}$$

$$= \frac{(m-1)\alpha_{ij}}{m}\sum_{k_h=0}^{M_1-1}\frac{e^{-\alpha_{ij}}(\alpha_{ij})^{k_h}}{k_h!} \quad (11)$$

$$< \frac{(m-1)}{m}\alpha_{ij} \quad (12)$$

The item $e^{-\sum_{g\neq j}\alpha_{ig}}\frac{(\sum_{g\neq j}\alpha_{ig})^{k_m}}{k_m!}$ in Equation 9 can be regarded as the probability of $\tilde{P}(x=k_m)$ where $\tilde{P}$ is a Poisson with the mean of $\sum_{g\neq j}\alpha_{ig}$. Besides, the item $\frac{e^{-\alpha_{ij}}(\alpha_{ij})^{k_h}}{k_h!}$ in Equation 11 can also be regarded as the probability of $\tilde{P}(x=k_h)$ where $\tilde{P}$ is a Poisson with the mean of α_{ij}. inequalities 10 and 12 hold, because as the integral of a Poisson distribution on a part of the whole range is smaller than 1. Then, we can do the same thing to simplify the second term of right hand side of Inequality 8::

$$\sum_{k_h=0}^{M_1}\sum_{k_m=0}^{M_2}\frac{k_m}{m} p(r_{ij},k_h,k_m)$$

$$= \frac{1}{m}\sum_{k_h=0}^{M_1}\frac{e^{-\alpha_{ij}}(\alpha_{ij})^{k_h}}{k_h!}\sum_{k_m=1}^{M_2} e^{-\sum_{g\neq j}\alpha_{ig}}\frac{(\sum_{g\neq j}\alpha_{ig})^{k_m}}{(k_m-1)!}$$

$$< \frac{1}{m}\sum_{g\neq j}\alpha_{ig}$$

Thus, we have $\sum_{k_h=0}^{M_1}\sum_{k_m=0}^{M_2} b_{k_h k_m} < (1-\frac{2}{m})\alpha_{ij} + \frac{\sum_{k=1}^{m}\alpha_{ig}}{m}$ □

Then, we can get the upper bound of the summation of $E_e(i,j)$ for all HGrids is $\sum_{i=1}^{n}\sum_{j=1}^{m} E_e(i,j) \leq 2(1-\frac{1}{m})\sum_{i=1}^{n}\sum_{j=1}^{m}\alpha_{ij}$.

According to Theorem II.1, to minimize the overall real error, we need to minimize the overall expression error. With Lemma III.1, to minimize expression error $E_e(i,j)$, we can minimize α_{ij} or m. However, α_{ij} is an inner property of HGrid r_{ij} and not determined by the prediction algorithms. For example, α_{ij} on weekdays and weekends in the same HGrid r_{ij} is quite different. On the other hand, the mean number of events in the same grid can vary greatly in different time periods of the day. Then, *to minimize expression errors, we should select the appropriate n to minimize m under the constraint $nm > N$*. Since $nm > N$, in order to minimize m, we should maximize n.

Convergence of Expression Error. We notice that Equation 7 is a summation of an infinite series. For a HGrid r_{ij}, we explain that Equation 7 to calculate expression error $E_e(i,j)$ can converge:

Lemma III.2. *Equation 7 to calculate Expression error $E_e(i,j)$ can converge.*

Proof. Considering that $b_{k_h k_m}$ for any k_h, k_m is positive and $\sum_{k_h=0}^{M_1}\sum_{k_m=0}^{M_2} b_{k_h k_m}$ is bounded according to the Lemma III.1, we can prove that Equation 7 can converge. Let $S(M_2,M_1) = \sum_{k_m=0}^{M_2}\sum_{k_h=0}^{M_1} b_{k_h k_m}$. Lemma III.1 shows $\exists M > 0$ make the $S(M_2,M_1) \leq M$ hold for any $M_2 \in \mathbb{Z}$. Since $S(M_2,M_1)$ is monotonically increasing with respect to M_2, $\lim_{M_2\to\infty} S(M_2,M_1)$ can converge, and we have $\lim_{M_2\to\infty} S(M_2,M_1) \leq M$, that is:

$$\sum_{k_m=0}^{\infty}\sum_{k_h=0}^{M_1} b_{k_h k_m} \leq M \Leftrightarrow \sum_{k_h=0}^{M_1}\sum_{k_m=0}^{\infty} b_{k_h k_m} \leq M$$

Let $T(M_1) = \sum_{k_h=0}^{M_1}\sum_{k_m=0}^{\infty} b_{k_h k_m}$, and we have $\lim_{M_1\to\infty} T(M_1) = E_e(i,j)$. In the same way, we can prove that $E_e(i,j)$ can converge, which means $E_e(i,j) = \lim_{M_1\to\infty} T(M_1) \leq M$ because of the monotone increase with respect to M_1. □

Since Equation 7 can converge, we will introduce algorithms to calculate expression error.

Algorithm to Calculate Expression Error. Equation 7 shows how to calculate the expression error $E_e(i,j)$ for a HGrid r_{ij}. In fact, we cannot compute the expression error exactly, but we can prove that the expression error can be approximated to arbitrary precision through the below theorem.

Theorem III.2. *For any ε, there is a number K that makes the following inequality holds:*

$$\left|\sum_{k_h=0}^{K}\sum_{k_m=0}^{(m-1)K} b_{k_h k_m} - E_e(i,j)\right| < \varepsilon$$

Proof. Lemma III.2 shows that $E_e(i,j)$ can converge. We have

$$\lim_{\tilde{k}_1\to\infty}\sum_{k_h=0}^{\tilde{k}_1}\sum_{k_m}^{\infty} b_{k_h k_m} = E_e(i,j)$$

According to the definition of limit, there will be M_1 for any $\varepsilon > 0$ that we have

$$\frac{-\varepsilon}{2} < \sum_{k_h=0}^{\tilde{k}_1}\sum_{k_m}^{\infty} b_{k_h k_m} - E_e(i,j) < \frac{\varepsilon}{2}$$

when $\tilde{k}_1 > M_1$, which means

$$\frac{-\varepsilon}{2} + E_e(i,j) < \sum_{k_h=0}^{\tilde{k}_1}\sum_{k_m}^{\infty} b_{k_h k_m} < \frac{\varepsilon}{2} + E_e(i,j)$$

Algorithm 1: Expression Error Calculation

Input: the number m of HGrids per MGrid, α_{ij} for each HGrid r_{ij} in the MGrid r_i, a hyper-parameter K
Output: the expression error $E_e(i,j)$ of the HGrid r_{ij}

$E_e(i,j) \leftarrow 0$
$\alpha_{i,\neq j} \leftarrow \sum_{g\neq j}^{m} \alpha_{ig}$
$p_1 \leftarrow e^{-\alpha_{ij}}$
for $k_h = 1$ *to* K **do**
  $p_2 \leftarrow e^{-\alpha_{i,\neq j}}$
  for $k_m = 1$ *to* $(m-1)K$ **do**
    $\Delta \leftarrow \left|\frac{(m-1)k_h - k_m}{m}\right| p_1 p_2$
    $E_e(i,j) \leftarrow E_e(i,j) + \Delta$
    $p_2 \leftarrow \frac{-p_2 \alpha_{i,\neq j}}{k_m}$
  $p_1 \leftarrow \frac{p_1 \alpha_{ij}}{k_h}$
return $E_e(i,j)$

Since the series $\sum_{k_h=0}^{\tilde{k}_1}\sum_{k_m}^{\infty} b_{k_h k_m}$ are bounded (shown by Lemma III.1), we can switch the order of the series.

$$\sum_{k_h=0}^{\tilde{k}_1}\sum_{k_m}^{\infty} b_{k_h k_m} = \sum_{k_m}^{\infty}\sum_{k_h=0}^{\tilde{k}_1} b_{k_h k_m}$$

We can also find M_2 for a positive number ε that meets

$$\frac{-\varepsilon}{2} < \sum_{k_h=0}^{\tilde{k}_1}\sum_{k_m}^{\tilde{k}_2} b_{k_h k_m} - \sum_{k_m}^{\infty}\sum_{k_h=0}^{\tilde{k}_1} b_{k_h k_m} < \frac{\varepsilon}{2}$$

Based on the definition of the limit when $\tilde{k}_2 > M_2$, we have

$$-\varepsilon < \sum_{k_h=0}^{\tilde{k}_1}\sum_{k_m}^{\tilde{k}_2} b_{k_h k_m} - E_e(i,j) < \varepsilon$$

We select a number K which meets the constraints of $K > M_1$ and $(m-1)K > M_2$. Thus, we have

$$\left|\sum_{k_h=0}^{K}\sum_{k_m=0}^{(m-1)K} b_{k_h k_m} - E_e(i,j)\right| < \varepsilon \tag{13}$$

□

Theorem III.2 shows that we can achieve the result close to the expression error by selecting a suitable K. We first need to compute $p(r_{ij}, k_h, k_m)$, which needs $O(k_h + k_m)$ time to compute. Then, the complexity of the whole calculation of Equation 7 is $O(m^2K^3)$. However the calculation of $p(r_{ij}, k_h, k_m)$ can be simplified as follows:

$$p(r_{ij}, k_h, k_m + 1) = \frac{\sum_{g\neq j}^{m} \alpha_{ig}}{k_m + 1} p(r_{ij}, k_h, k_m) \tag{14}$$

Based on Equation 14, Algorithm 1 is proposed to approximately computing the expression error $E_e(i,j)$ of the HGrid r_{ij}. Since the complexity of computing $p(r_{ij}, k_h, k_m)$ is $O(1)$, the complexity of Algorithm 1 is $O(mK^2)$.

Algorithm Optimization. Considering the large number of HGrids, even though the time needed to calculate the expression error of each HGrid is only about 0.1 second, the final time cost needed to calculate the summation of expression error of all HGrids with Algorithm 1 is about 4 hours. Therefore, we introduce a more efficient algorithm with time complexity of $O(mK)$ in this section based on a more in-depth analysis of Equation 7.

According to theorem III.2, we can approximate Equation 7 with the following equations:

$$\sum_{k_h=0}^{K}\sum_{k_m=0}^{(m-1)K}\left|\frac{(m-1)k_h - k_m}{m}\right| p(r_{ij}, k_h, k_m) \tag{15}$$

$$= \frac{(m-1)}{m}\sum_{k_h=0}^{K}\sum_{k_m=0}^{(m-1)K} k_h \mathbb{I}((m-1)k_h - k_m)\, p(r_{ij}, k_h, k_m)$$

$$- \frac{1}{m}\sum_{k_h=0}^{K}\sum_{k_m=0}^{(m-1)K} k_m \mathbb{I}((m-1)k_h - k_m)\, p(r_{ij}, k_h, k_m) \tag{16}$$

where $\mathbb{I}(x)$ is a indicator function and satisfies:

$$\mathbb{I}(x) = \begin{cases} 1, & x > 0 \\ -1, & x \leq 0 \end{cases}$$

We transform the first term of right hand side of Equation 16 (denoted as e_1) to the following formula:

$$\frac{(m-1)}{m}\sum_{k_h=1}^{K}\sum_{k_m=0}^{(m-1)K} k_h \mathbb{I}((m-1)k_h - k_m) p(r_{ij}, k_h, k_m)$$

$$= \frac{(m-1)}{m}\sum_{k_h=1}^{K} k_h \left(2\sum_{k_m=0}^{(m-1)k_h} p(r_{ij}, k_h, k_m) - \sum_{k_m=0}^{(m-1)K} p(r_{ij}, k_h, k_m)\right)$$

$$= \frac{(m-1)}{m}\sum_{k_h=1}^{K} e^{-\sum_{j=1}^{m}\alpha_{ij}} \frac{(\alpha_{ij})^{k_h}}{(k_h-1)!} e_1'(k_h) \tag{17}$$

Let $e_1'(k_h)$ denote a function with respect to k_m, as follows:

$$-\sum_{k_m=0}^{(m-1)K} \frac{\left(\sum_{g\neq j}\alpha_{ig}\right)^{k_m}}{k_m!} + 2\sum_{k_m=0}^{(m-1)k_h} \frac{\left(\sum_{g\neq j}\alpha_{ig}\right)^{k_m}}{k_m!} \tag{18}$$

The time complexity of the direct calculation of $e_1'(k_h+1)$ is $O(mK)$ based on Equation 18; as a result, the time complexity of calculating e_1 is $O(mK^2)$. However, we can build the connection between $e_1'(k_h+1)$ and $e_1'(k_h)$ as follows:

$$e_1'(k_h+1) - e_1'(k_h)$$

$$= 2\sum_{k_m=0}^{(m-1)(k_h+1)} \frac{\left(\sum_{g\neq j}\alpha_{ig}\right)^{k_m}}{k_m!} - 2\sum_{k_m=0}^{(m-1)k_h} \frac{\left(\sum_{g\neq j}\alpha_{ig}\right)^{k_m}}{k_m!}$$

$$= 2\sum_{k_m=(m-1)k_h}^{(m-1)(k_h+1)} \frac{\left(\sum_{g\neq j}\alpha_{ig}\right)^{k_m}}{k_m!} \tag{19}$$

Therefore, $e_1'(k_h+1)$ can be calculated through the result of $e_1'(k_h)$ so that the time complexity of $e_1'(k_h)$ can be reduced to $O(m)$. Then we can do the same analysis for the second term e_2 of right hand side of Equation 16:

$$\frac{1}{m}\sum_{k_h=0}^{K}\sum_{k_m=0}^{(m-1)K} k_m \mathbb{I}((m-1)k_h - k_m) p(r_{ij}, k_h, k_m)$$

$$= \frac{1}{m}\sum_{k_h=0}^{K} e^{-\sum_{j=1}^{m}\alpha_{ij}} \frac{(\alpha_{ij})^{k_h}}{k_h!} e_2'(k_h)$$

where $e_2'(k_h) = -\sum_{k_m=1}^{(m-1)K} \frac{\left(\sum_{g\neq j}\alpha_{ig}\right)^{k_m}}{(k_m-1)!} + 2\sum_{k_m=1}^{(m-1)k_h} \frac{\left(\sum_{g\neq j}\alpha_{ig}\right)^{k_m}}{(k_m-1)!}$, and we can do a similar analysis as Equation 19.

Algorithm 2 is proposed through the above analysis. By reducing the time complexity of $e_1'(k_h)$ and $e_2'(k_h)$ from $O(mK)$ to $O(m)$, the time complexity of Algorithm 2 becomes $O(mK)$.

Algorithm 2: Fast Expression Error Calculation

Input: the number m of HGrids per MGrid, α_{ij} for each HGrid r_{ij} in the MGrid r_i, a hyper-parameter K
Output: the expression error $E_e(i,j)$ of the HGrid r_{ij}

$p_2 \leftarrow 1$; $e_1^{'}, e_2^{'} \leftarrow 0$
for $k_m = 0$ *to* $(m-1)K$ **do** // initialize $e_1^{'}$ and $e_2^{'}$
  $p_2 \leftarrow p_2 \sum_{g \neq j} \alpha_{ig}$
  $e_2^{'} \leftarrow e_2^{'} - p_2$
  $p_2 \leftarrow p_2/(k_m+1)$
  $e_1^{'} \leftarrow e_1^{'} - p_2$
$p_1 \leftarrow e^{-\sum_{j=1}^{m} \alpha_{ij}}$
$p_2 \leftarrow 1$; $e_1, e_2 \leftarrow 0$
for $k_h = 1$ *to* K **do** // calculate the value of e_1 and e_2
  for $k_m = (m-1)(k_h-1)$ *to* $(m-1)k_h$ **do**
    $e_2^{'} \leftarrow e_2^{'} + 2p_2$
    $p_2 \leftarrow \frac{p_2}{k_m^{'}+1}$
    $e_1^{'} \leftarrow e_2 + 2p_2$
    $p_2 \leftarrow p_2 \sum_{g \neq j} \alpha_{ig}$
  $e_1 \leftarrow e_1^{'} p_1 + e_1$
  $p_1 \leftarrow \frac{p_1 \alpha_{ij}}{k_h}$
  $e_2 \leftarrow e_2^{'} p_1 + e_2$
$E_e(i,j) \leftarrow \frac{m-1}{m} e_1 - \frac{e_2}{m}$
return $E_e(i,j)$

C. Analysis of Model Error

In this section, we can estimate model error with the mean absolute error for each HGrid. Suppose we use the model f to predict the event number $\hat{\lambda}_i$ (i.e., $\hat{\lambda}_i = f(x_i)$) for the next stage of the MGrid through the historical information of the events X. Let denote the dataset of each MGrid r_i as X_i, and we have $\cup_{i=1}^{n} X_i = X$. Meanwhile, the number of samples in X_i for each MGrid r_i is $\frac{|X|}{n}$. We define the mean absolute error of f as $MAE(f)$ (i.e., $MAE(f) = \frac{\sum_{x_i \in \mathbf{X}} |f(x_i) - \lambda_i|}{|\mathbf{X}|}$), and we have

$$\begin{aligned}
\lim_{|\mathbf{X}| \to \infty} MAE(f) &= \lim_{|\mathbf{X}| \to \infty} \frac{\sum_{x_i \in \mathbf{X}} |f(x_i) - \lambda_i|}{|\mathbf{X}|} \\
&= \frac{1}{n} \sum_{i=1}^{n} \lim_{|\mathbf{X}_i| \to \infty} \frac{\sum_{x_j \in \mathbf{X}_i} |f(x_j) - \lambda_j|}{|\mathbf{X}_i|} \\
&= \frac{1}{n} \sum_{i=1}^{n} E\left(\left|\hat{\lambda}_i - \lambda_i\right|\right)
\end{aligned}$$

We can get the relationship between the model error $E_m(i,j)$ and $MAE(f)$:

$$\begin{aligned}
\sum_{i=1}^{n} \sum_{j=1}^{m} E_m(i,j) &= \sum_{i=1}^{n} \sum_{j=1}^{m} \mathbb{E}\left(\left|\hat{\lambda}_{ij} - \lambda_{ij}\right|\right) = \sum_{i=1}^{n} m\mathbb{E}\left(\left|\hat{\lambda}_{ij} - \lambda_{ij}\right|\right) \\
&= \sum_{i=1}^{n} \mathbb{E}\left(\left|\hat{\lambda}_i - \lambda_i\right|\right) \approx nMAE(f) \qquad (20)
\end{aligned}$$

According to Equation 20, the total model error will increase when n increases. However, based on the analyses in Section III-B, the total expression error will decrease when n increases. We have proved that the summation of expression error and model error is a upper bound of real error, which will first decrease then increase when n increase from 1 to N. Thus, to minimize the total real error, we will propose two efficient algorithms to select a proper n in next section.

Algorithm 3: $UpperBound(n, N, \mathbf{X}, Model)$

Input: the number of MGrid n, the number of all HGrids N, dataset $\mathbf{X}$, a prediction method $Model$
Output: $e(\sqrt{n})$

$m \leftarrow \left\lceil \sqrt{\frac{N}{n}} \right\rceil^2$
$f \leftarrow Model(\mathbf{X})$
$e \leftarrow nMAE(f)$
divide the global space into N HGrids and estimate the α_{ij} for each HGrid r_{ij}
for $i = 1$ *to* n **do**
  for $j = 1$ *to* m **do**
    $e \leftarrow e + E_e(i,j)$ `// calculated by Algorithm 2`
return e

IV. Search for Optimal Grid Size

From the analysis in Section III, the size of n will affect expression error and model error of each HGrid r_{ij}, which will further affect the upper bound of real error. A straightforward algorithm that checks all the values of n can achieve the optimal solution for OGSS with the complexity of $O(\sqrt{N})$, which is not efficient. Therefore, we will propose two more efficient algorithms to solve OGSS in this section. We first introduce the upper bound calculation of real error.

A. Calculation of Upper Bound for Real Error

In practice, it is difficult to directly estimate the real error of each HGrid, and then select the optimal partitioning size. Theorem II.1 proves that the summation of expression error and model error is an upper bound of real error. Thus, we can turn to minimize expression error and model error, whose calculations have been discussed in Section III. Specifically, we can use Algorithm 2 to calculate expression errors and Equation 20 to estimate model errors.

Based on the analysis in Section III, we propose our algorithm showed in Algorithm 3 to calculate $e(\sqrt{n})$ (i.e., $e(\sqrt{n}) = \sum_{i=1}^{n} \sum_{j=1}^{m} E_u(i,j)$), which is an approximate problem of OGSS. The time cost of training the model is considerable when calculating the error $e(\sqrt{n})$. Therefore, we will introduce two algorithms with fewer computations of $e(\sqrt{n})$.

B. Ternary Search

Without any prior information, we cannot make any optimization of the most straightforward algorithm. Fortunately, it can be concluded from the analysis in Section III that the model error will increase and the expression error will decrease when n increases. It means there exists an equilibrium point that minimizes the summation of the expression error and the model error. Consider an extreme case (i.e., $n = 1$), the prediction model only needs to predict the number of events for the whole space in the future, which can be very accurate. For example, according to the historical information of New York City (NYC), the number of spatial events (e.g., rider's order) on weekdays almost keeps a relatively stable value without dramatic fluctuations. At this time, the model error is small, but the expression error is considerable. Even if

Algorithm 4: Ternary Search

Input: dataset $\mathbf{X}$, prediction model $Model$
Output: partition size n that minimize $e(\sqrt{n})$

use the method analyzed in Section III to select N
$l \leftarrow 1$; $r \leftarrow \sqrt{N}$
while $r - l > 1$ **do**
  $m_r \leftarrow \lceil \frac{2}{3}r + \frac{1}{3}l \rceil$
  $m_l \leftarrow \lfloor \frac{1}{3}r + \frac{2}{3}l \rfloor$
  $e(m_l) \leftarrow UpperBound\left(m_l^2, N, \mathbf{X}, Model\right)$
  $e(m_r) \leftarrow UpperBound\left(m_r^2, N, \mathbf{X}, Model\right)$
  if $e(m_l) > e(m_r)$ **then**
    $l \leftarrow m_l$
  else
    $r \leftarrow m_r$
if $e(l) > e(r)$ **then**
  $n \leftarrow r^2$
else
  $n \leftarrow l^2$
return n

Algorithm 5: Iterative Method

Input: dataset $\mathbf{X}$, prediction model $Model$
Output: partition size n that minimize $e(\sqrt{n})$

use the method analyzed in Section III to select N
$p \leftarrow 16$; $b \leftarrow 4$
$flag \leftarrow true$
while $flag$ **do**
  $flag \leftarrow false$
  for $i = b$ *to* 1 **do**
    $e(p+i) \leftarrow UpperBound\left((p+i)^2, N, \mathbf{X}, Model\right)$
    $e(p-i) \leftarrow UpperBound\left((p-i)^2, N, \mathbf{X}, Model\right)$
    if $e(p) > e(p+i)$ **then**
      $p \leftarrow p+i$
      $flag \leftarrow true$
      break
    if $e(p) < e(p-i)$ **then**
      $p \leftarrow p-i$
      $flag \leftarrow true$
      break
$n \leftarrow p^2$
return n

we could know the exact number of orders in the whole NYC for specific timestamp, it would not help for us to dispatch orders for drivers in a particular street area of NYC. When $n = N$, the forecasting model needs to predict a mass of grids' events accurately, which will leads to huge model errors due to the uncertainty of human behavior. While the area of a grid is very small, the uncertainty of human activity will lead huge different of prediction. Therefore, we assume that the trend of $e(\sqrt{n})$ with the increase of n will first go down and then up (This assumption will be confirmed in Section V-C).

We propose a ternary search algorithm to find the optimal partition size. Given that n is a perfect square, we need to find the optimal n among $\sqrt{N}$ numbers. Let l be the minimum of $\sqrt{n}$ and r be the maximum of $\sqrt{n}$. The main idea of ternary search is to take the third-equinox between r and l in each round and then compare the corresponding error of the two third-equinox points denoted as m_r, m_l. If $e(m_r) > e(m_l)$, let $r = m_r$ for next round; otherwise, let $l = m_l$. The ternary search algorithm showed in Algorithm 4 will drop $\frac{1}{3}$ of possible values for n each time, which results in the convergence.

If the graph of function $e(\sqrt{n})$ has only one minimum point, then the ternary search will find the optimal solution. However, the graph of function $e(\sqrt{n})$ is not always ideal, but the ternary search algorithm can also find a good solution.

Time Complexity. For a given N, we can mark the algorithm complexity as $T(\sqrt{N})$. We know that the algorithm will drop $\frac{1}{3}$ of the possible values from the above analysis, thus converting the original problem into a subproblem. Therefore, we have: $T(\sqrt{N}) = T(\frac{2}{3}\sqrt{N}) + 2$. We can infer that the time complexity of Algorithm 4 is $O(\log \sqrt{N})$ according to the master theorem.

C. Iterative Method

Although the ternary search algorithm reduces the algorithm complexity from $O(\sqrt{N})$ to $O(\log \sqrt{N})$ based on the traversal algorithm, the experiments in Section V show that the ternary search algorithm may miss the optimal global solution in some situations. Therefore, we will introduce an iteration-based algorithm with a greater probability of achieving the optimal n in this section.

Considering that the upper bound on the real error is large when n is either large or small, the global optimal value for n tends to be somewhere in the middle. We can roughly choose the possible value of the optimal solution through practical experience and then take this value as the initial position to conduct a local search. We set a search boundary, and if the size of error for the current position is smaller than all possible regions within the boundary, the current position is likely to be the optimal solution. In order to speed up the search process, we start the current position of searching from the boundary to avoid local traversal when $e(\sqrt{n})$ is monotonous. The details of the algorithm is shown in Algorithm 5.

In Algorithm 5, the choice of the initial position and the setting of the search boundary significantly affect the quality of its result and its efficiency. Based on the experience from the existing studies [2], we use the default grid of $2km \times 2km$ (i.e., approximately 16×16) as the corresponding initial position to speed up the search for the global optimal n. On the other hand, the search boundary has an essential influence on the properties of the solution and the algorithm's efficiency. When the search boundary is large, the probability of the algorithm finding the optimal solution will increase, but the efficiency of the algorithm execution will decrease. On the contrary, the algorithm can converge quickly when the search boundary is small with a small probability of finding the optimal solution.

V. Experimental Study

A. Data Set

We use realistic data to study the property of expression error and model error.

New York Taxi Trip Dataset. New York Taxi and Limousine Commission (TLC) Taxi Trip Data [15] includes the taxi orders in NYC. We use the Taxi Trip Dataset from January to May 2013 (i.e., January to April as training set, May 1st to 27th as validation set, and May 28th as test set). There are 282,255 orders in test set. The size of the whole space is $23km \times 37km$ (i.e., $-73.77° \sim -74.03°$, $40.58° \sim 40.92°$). Since the number of other taxis in NYC is much smaller than the number of yellow taxi, we only use the trip data of yellow taxi. Each order record contains the pick-up and drop-up locations, the pick-up timestamp, and the driver's profit.

Chengdu Taxi Trip Dataset. DiDi Chuxing GAIA Open Dataset [16] provides taxi trips in Chengdu, China. We use the taxi trip record from November 1st, 2016 to November 25th, 2016 as training set, November 26th to 29th as validation set and November 30th as test data set. There are 238,868 orders in test set. The size of whole space in Chengdu is also $23km \times 37km$ (i.e., $103.93° \sim 104.19°$, $30.50° \sim 30.84°$).

Xi'an Taxi Trip Dataset. DiDi Chuxing GAIA Open Dataset [16] also provides a dataset of taxi trips in Xi'an, China. We use the taxi trip records from October 1st, 2016 to October 25th, 2016 as training data set, October 26th to 29th as validation set and October 30th as test set. There are 109,753 orders in test set. The size of whole space in Xi'an is $8.5km \times 8.6km$ (i.e., $108.91° \sim 109.00°$, $34.20° \sim 34.28°$).

Due to the space limitation, please refer to Appendix A of our technical report [17] for the distributions of the datasets.

B. Experiment Configuration

We use three prediction models to predict the future numbers of spatial events in different regions:

Multilayer Perceptron (MLP) [18]: We use a neural network of six fully connected layers. The number of hidden units are 1024, 1024, 512, 512, 256, 256. When the size of MGrid is n, we can get the model input $(8, \sqrt{n}, \sqrt{n})$, which represents the number of all regions in nearest eight time slots, and we use a flatten layer to map the original input to a vector with the size of $8 \times n$ before it is fed into the model.

DeepST [6]: DeepST divides a day into 48 time slots (i.e., 30 minutes per time slot) and calculates inflow and outflow of the events. As a result, DeepST can calculate the number of events for the next time slot by predicting the inflow and outflow status of events for the next time slot. It uses three types of historical information: closeness, period, trend. Closeness expresses the number of events in the nearest eight time slots, period expresses the number of events at the same time slot of the previous eight days, and trend represents the number of events at the same time slot of the previous eight weeks. DeepST mainly utilizes the spatial information to predict the spatial events for next time slot.

Dmvst-Net [10]: Dmvst-Net models the correlations between future demand with recent historical data via long short term memory (LSTM) and models the local spatial correlation via convolutional neural network (CNN). Moreover, Dmvst-Net models the correlations among regions sharing similar temporal patterns. Compared with DeepST, Dmvst-Net utilizes both spatial and temporal information, which lead to a better performance of the prediction model.

TABLE II: Experiment Setting for Training Model

Symbol	Setting
N	128×128
n	$4\times4, \ldots, \mathbf{16\times16}, \ldots, 75\times75, 76\times76$
time slot	30 minutes
prediction model	MLP, **DeepST**, Dmvst-Net

Since the size of model input for DeepST and Dmvst-Net is different in the experiment, we need to map the original input to the same $shape$ in order to ensure that the model structure will not change significantly through a conditional deconvolution layer. When the number of MGrid is n, that is, the input dimension of the model is $(2, \sqrt{n}, \sqrt{n})$, the size k of the convolution kernel and step size s of the convolutional layer can be obtained through the following formula:

$$\begin{aligned} s &= \left\lfloor \frac{shape}{n-1} \right\rfloor \\ k &= shape - s\,(n-1) \end{aligned}$$

Here, we set $shape = 128$ in our experiment. Finally, we add a convolution layer with the same stride and the same size as the deconvolution layer as the last layer of DeepST.

As the dataset used in this experiment is the Taxi Trip Dataset, **Order Count Bias** is used as the metric of model error, expression error and real error in this experiment. Model error represents the difference between the predicted order quantity and the estimated order quantity; expression error represents the difference between the estimated order quantity and the actual order quantity; real error represents the difference between the actual order quantity and the predicted order quantity. Considering that we will constantly change the grid size in the experiment, it is meaningless to consider the error of a single grid; therefore, the errors we discuss in subsequent experiments are the summation of errors of all grids, unless otherwise specified.

In order to calculate the expression error of a HGrid, we need to estimate the mean number of events α_{ij} for the grid r_{ij} in advance. In a long period, grid environments will change significantly so that the number of events for the same grid does not follow the same distribution. On the other hand, when the number of samples is small, the estimate of the mean number for events will produce a considerable bias. Therefore, when we estimate the mean number of events, we need to choose the appropriate range of adoption. At the same time, considering the remarkable difference about the number of events at different periods in a day and the great difference in the willingness of people to travel on weekdays and workdays, this experiment takes the average of the number of events at the same period of all workdays in the recent one month as the mean number α_{ij} of events in the HGrid r_{ij}. In subsequent experiments, we estimate α_{ij} by using the number of events between $8:00\ A.M.$ and $8:30\ A.M.$ as default unless otherwise stated. The above experimental Settings are summarized in Table II where the default parameters are in bold. Our experiments were run on AMD Ryzen 5-5600H with 32 GB RAM and GeForce RTX 3050 in Python, while LS and POLAR and DAIF were run in Java.

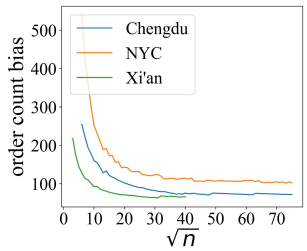
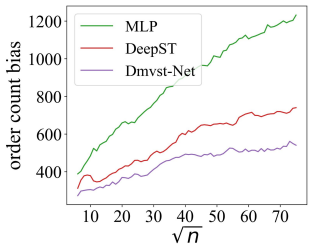
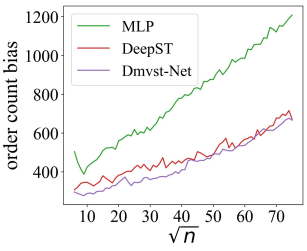


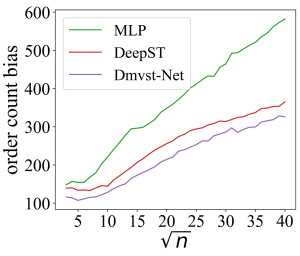
Fig. 3: Effect of n on Expression Error in Different Cities



(a) Chengdu

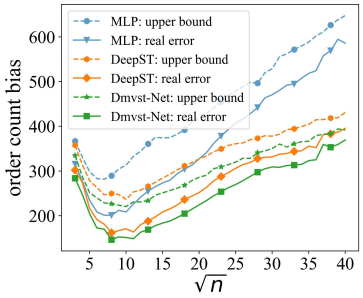


(b) NYC

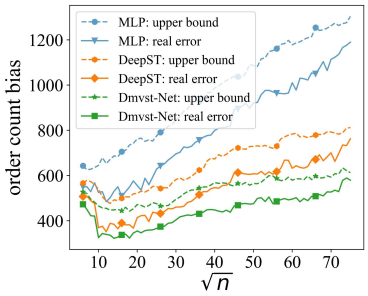


(c) Xi'an

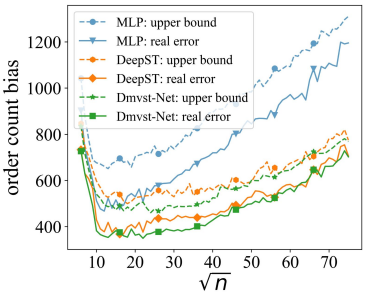
Fig. 4: Effect of n on the Model Error



(a) Real Error in Xi'an



(b) Real Error in Chengdu



(c) Real Error in NYC

Fig. 5: Effect of n on Real Error in Different Cities with Different Prediction Models

C. Relationship between Real Error and n

In this section, we mainly show the effect of n on the expression error and the model error as analyzed in Section III and verify that real error has a same change trend with its upper bound.

Expression Error. We use Algorithm 2 to calculate the expression errors in different cities, which all decrease with the increase of n as shown in Figure 3. Since orders in NYC are more evenly distributed than in Chengdu, therefore, the expression error of Chengdu is smaller than that of NYC when n is same. Additionally, the order quantity of Xi'an is much smaller than that of the other two cities. Meanwhile, the order distribution of Xi'an city is more uniformly distributed compared with the other two cities. As a result, the expression error of Xi'an is much smaller than that of other cities. Due to space constraints, we analyze the relationship between expression error and the uniformity of order distribution in detail in Appendix B in our technical report [17].

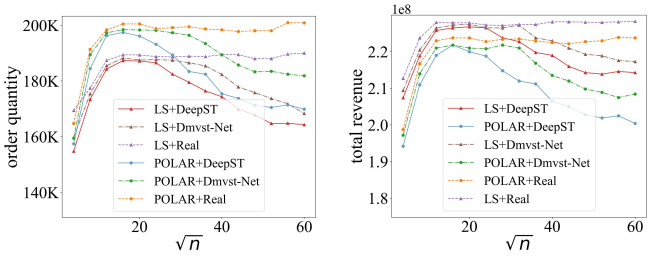
Model Error. We test the performance of three prediction models (i.e., MLP, DeepST, and Dmvst-Net) on the datasets of NYC and Chengdu as shown in Figures 4. The experimental results show that the model error of the three prediction models all increase with the increase of n on the three data sets. The model errors of DeepST and Dmvst-Net are much smaller than that of MLP with relatively simple model structure, while Dmvst-Net makes use of time information of historical data so that it performs better than DeepST.

Real Error. Figure 5 shows the relationship between real error and its upper bound in different cities while using different prediction models. The real error and its upper bound have the same trend, all falling first and then rising while changing n. Comparing with Chengdu, the expression error of NYC is larger, which makes the optimal n of NYC larger than that of Chengdu when the same prediction model is used. For example, the real error of NYC based on Dmvst-Net is also small when n is 30×30 shown in Figure 5(c). On the other hand, the prediction model with higher accuracy make the real error significantly smaller, and also lead to the increase of n that minimizes the real error. Taking NYC as an example, the optimal value of n is 23 when using Dmvst-Net as prediction model; when the prediction model is DeepST, the optimal value of n is 16; when the prediction model is MLP, the optimal value of n is 13. In the case of models with high accuracy, a larger n helps to reduce expression error. Moreover, when we use MLP as a prediction model to forecast the number of orders in Chengdu, Figure 5(b) shows that the real error increases varying n as the model error plays a dominant role in the real error while the expression error of Chengdu is small and the model error of MLP is large. In addition, because the area of the Xi'an dataset is much smaller than that of Chengdu and NYC, the optimal n of Xi'an is smaller than that of the other two cities.

D. Case Study on Effect of Minimizing Real Error

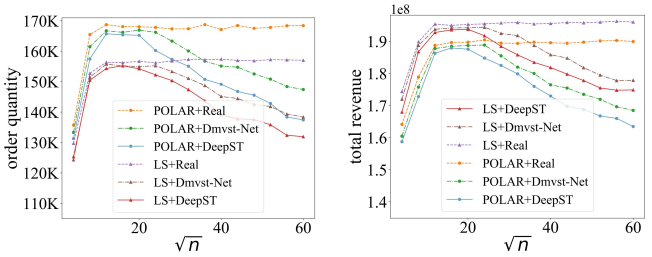
In this section, we explore the effect of real error on some crowdsourcing problems (i.e., task assignment [14], [1] and route planning [2]). We test two prediction models: Dmvst-Net and DeepST in the experiment.

Task Assignment. The task assignment refers to sending location-based requests to workers, based on their current positions, such as ride-hailing. We used two state-of-the-art prediction-based task assignment algorithms (i.e., LS [14], POLAR [1]) to dispatch orders under different values of n. The goal of LS is to maximize total revenue while the goal of POLAR is to maximize the number of orders served. Thus, we use the total revenue and order quantity as metrics for



(a) Order Quantity (b) Total Revenue

Fig. 6: Effect of n on Task Assignment (NYC)



(a) Order Quantity (b) Total Revenue

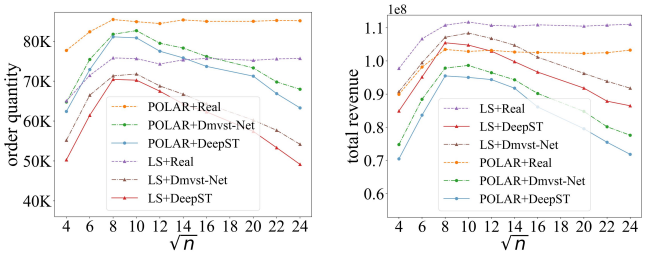
Fig. 7: Effect of n on Task Assignment (Chengdu)

the two algorithms. We compare the performance of the two algorithms using different prediction models in this paper. Specific experimental setup in this paper is as same as the default setting in our previous work [14].

Figure 6∼8 show that total revenue and order quantity of the prediction-based dispatching algorithms vary under different values of n. When using the predicted results, the revenue of both algorithms shows a trend of rising firstly with declining later because the real error is large when n is too small or too large. When POLAR and LS use real order data which makes the model error be 0, the real error is equivalent to the expression error. It means that the real error decreases as n increases. Therefore, the performance of Polar and LS will not decrease due to the large n when using the real order data, which is also consistent with the changing trend of the real error. In addition, the order distribution of Xi'an is more even than that of the other two cities because of its smaller area. Therefore, the optimal n in Xi'an is less than that in the other two cities. In short, the experimental results verify that the real error is an important factor affecting the performance of the algorithms in task assignment.

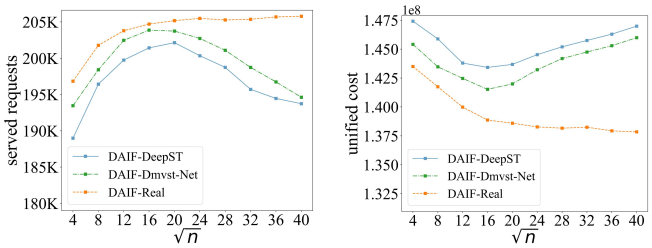
Route Planning. Route planning is a central issue in shared mobility applications such as ride-sharing, food delivery and crowdsourced parcel delivery. We use the state-of-the-art algorithm, DAIF [2], to verify the effect of n on route planning problem. We use the default parameters of the original paper [2] in this experiment, and take the number of served requests and the unified cost as the metrics of DAIF. Figure 9 shows that the number of served requests of DAIF first increases then decreases when n increases. The unified cost of DAIF is minimized when $n = 16 \times 16$. Using the actual number of orders, DAIF gets better performance with a large n. Although route planning problem is affected less by real error compared with task assignment problem, the size of grid affects the performance of prediction-based algorithm.

Table III shows the improvement of the original algorithm



(a) Order Quantity (b) Total Revenue

Fig. 8: Effect of n on Task Assignment (Xi'an)



(a) Served Requests (b) Unified Cost

Fig. 9: Effect of n on Route Planning (NYC)

by selecting the optimal grid size with DeepST as the prediction model on NYC. Original n represents the default value of n set in [1], [14], [2], while optimal n denotes the optimal grid size found by our GridTuner. The results show that both POLAR and DAIF can achieve performance gain with the optimal grid size. Due to the selection of the default n in the existing paper [1] is close to the optimal n, the performance of LS has no obvious improvement.

TABLE III: Promotion of the prediction-based algorithms

Metric	Algorithm	Optimal n	Original n	Improve ratio
Served Order Number	POLAR	16×16	50×50	13.6%
Total Revenue	POLAR	16×16	50×50	8.97%
Total Revenue	LS	20×20	16×16	0.13%
Served Order Number	LS	20×20	16×16	0.7%
Unified Cost	DAIF	16×16	12×12	0.76%
Served Requests	DAIF	20×20	12×12	3.35%

E. Experiment Result of Optimization Searching Algorithms

Since the mean of the event quantity in the same grid varies in different periods of a day, the expression error of each time slot is different, leading to the different optimal solutions of each time slot. In this section, we use the algorithms proposed in Section IV to calculate the optimal partition scheme of different cities and compare the performance of them with the **Brute-force Search** (i.e., traverses all the values to find the optimal n). We use three indicators to measure the quality of the solution found by the algorithm and the efficiency of the algorithm: **cost** denotes the cost of time; **probability** denotes the probability of obtaining the optimal solution (i.e., the number of finding out the optimal solution divided by the number of time slots); **optimal ratio** (OR) is denoted as $OR = \frac{o_a}{o_r}$, where o_r denote the optimal order count served by driver while using the POLAR as dispatching algorithm in NYC and o_a represents the results optimized by the algorithm.

The experimental results in Table IV show that Ternary Search and Iterative Method both can greatly reduce the time

TABLE IV: Performance of the algorithms.

City	Algorithm	Cost (h)	Probability	OR
NYC	Ternary Search	7.03	52.08%	97.83%
NYC	Iterative Method	**5.58**	**81.25%**	**98.77%**
NYC	Brute-force Search	47.43	100.00%	100.00%
Chengdu	Ternary Search	6.32	70.83%	98.35%
Chengdu	Iterative Method	4.53	**95.83%**	**99.77%**
Chengdu	Brute-force Search	43.26	100.00%	100.00%
Xi'an	Ternary Search	**3.90**	60.42%	97.98%
Xi'an	Iterative Method	3.31	**91.67%**	**99.57%**
Xi'an	Brute-force Search	21.76	100.00%	100.00%

cost of finding the optimal solution compared with the Brute-force Search. Meanwhile, both algorithms can find the global optimal solution with high probabilities. With the reasonable choice of bound and initial position of Iterative Method, its execution efficiency and probability of finding the optimal solution are better than them of Ternary Search. According to Table IV, sub-optimal solutions achieved by Ternary Search are at most 3% less than the optimal results (and 1.5% for Iterative Method), which shows the effectiveness of our grid size selection algorithms.

Summary: The experimental results show that the larger real error often leads to the decrease of the payoff of the dispatching algorithms. At the same time, Ternary Search and Iterative Method proposed in this paper can effectively find the optimal solution to the OGSS by minimizing the upper bound of the real error. Furthermore, this paper improves the effect of the original algorithm by selecting a reasonable size n. Specially, the performance of POLAR improves by 13.6% on served order number and 8.97% on total revenue. Finally, this paper also studies the influence of different traffic prediction algorithms on the optimal size of MGrids. The results show that when the accuracy of the prediction algorithm is high, the whole space can be divided into more MGrids to reduce the expression error. On the contrary, when the accuracy of the prediction algorithm is low, we need to make the area of a MGrid larger to reduce the model error.

VI. Related Work

In recent years, with the rise of various online taxi-hailing platforms, more and more researchers have been working on how to assign tasks to workers. Summarized the publishing models in [19], the task assignment problem mainly is classified in two modes: worker selected task and server assigned task. Compared with the former, the latter is easier to find the optimal global solution, and the major online taxi-hailing platforms mainly adopt the latter, which attracts more and more researchers' attention.

There are two main modes of order distribution on the platform: online [1], [3], [20], [2], [21] and the other is offline [22], [23], [24], [25]. The online task assignment faces more tremendous challenges than the offline task assignment due to the lack of follow-up order information. However, the emergence of traffic prediction technology [6], [10], [26] has solved this problem well. With the continuous improvement of these traffic forecasting work, the demand-aware algorithm [1], [3], [2], [27], [21] for task assignment has more advantages than some traditional algorithm [19], [28], [25], [22]. The optimal solution of task assignment based on the supply and demand can be approximately equivalent to the optimal solution of offline task assignment problem when the prediction result of the traffic prediction algorithm is close to the real.

Several traffic prediction methods [6], [10], [26], [29], [30] divides the entire space into grids based on latitude and longitude and then predicts the number of orders in each region. The residual network is introduced into the traffic prediction in [6] so that the deep neural network can better reduce the deviation between the predicted results and the actual results. [10] tries to combine different perspectives to predict future order data, including time perspective, space perspective, and semantic perspective. The results show that the multi-view spatiotemporal network can improve the prediction performance of the model. In addition, an attention mechanism is introduced in [26] to mine the dynamic spatiotemporal correlation of traffic data to optimize the prediction performance.

Combining the result of predicted future distribution of tasks, algorithms for task assignment can better solve the problem. The work [3] proposed a framework based on queuing theory to guide the platform for order dispatching, which used queuing theory combined with the distribution of future orders and drivers in the region to predict the waiting time of drivers before they received the next order after sending the current order to the destination. In addition, a two-stage dispatching model is proposed in [1]. In the first stage, the platform will pre-assign drivers based on the predicted number of regional orders and direct them to the likely location of the orders, while in the second stage, it will assign the actual orders.

The accuracy of traffic prediction will significantly affect the performance of this algorithm. However, The order dispatching algorithms pay attention to the model error and the expression error caused by the uneven distribution of orders in the grids. Our problem mainly focuses on how to divide model grid to balance the model error and the expression error to improve the effectiveness of the order dispatching algorithm based on supply and demand prediction.

VII. Conclusion

In this paper, we propose a more fine-grained measure of prediction bias, namely real error, and investigate how to minimize it. The real error is mainly composed of expression error and model error. Expression error is caused by using the order quantity of large regions to estimate the number of spatial events of HGrids, while model error is the inner error of the prediction model. We show that the summation of the expression error and the model error is the upper bound on the real error. We solve expression error and model error and analyze the relationship between them and MGrids. Through the above analysis, we propose two algorithms to minimize the real error as much as possible by minimizing its upper bound. Finally, we verify the effectiveness of our algorithm through experiments and analyze the role of real error for spatiotemporal prediction models.

REFERENCES

[1] Y. Tong, L. Wang, Z. Zimu, B. Ding, L. Chen, J. Ye, and K. Xu, "Flexible online task assignment in real-time spatial data," *Proceedings of the VLDB Endowment*, vol. 10, no. 11, pp. 1334–1345, 2017.

[2] J. Wang, P. Cheng, L. Zheng, C. Feng, L. Chen, X. Lin, and Z. Wang, "Demand-aware route planning for shared mobility services," *Proceedings of the VLDB Endowment*, vol. 13, no. 7, pp. 979–991, 2020.

[3] P. Cheng, C. Feng, L. Chen, and Z. Wang, "A queueing-theoretic framework for vehicle dispatching in dynamic car-hailing," in *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pp. 1622–1625, IEEE, 2019.

[4] G. O. Mohler, M. B. Short, P. J. Brantingham, F. P. Schoenberg, and G. E. Tita, "Self-exciting point process modeling of crime," *Journal of the American Statistical Association*, vol. 106, no. 493, pp. 100–108, 2011.

[5] C. Huang, J. Zhang, Y. Zheng, and N. V. Chawla, "Deepcrime: Attentive hierarchical recurrent networks for crime prediction," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, p. 1423–1432, Association for Computing Machinery, 2018.

[6] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, 2017.

[7] Y. Li, Y. Zheng, H. Zhang, and L. Chen, "Traffic prediction in a bike-sharing system," in *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '15, Association for Computing Machinery, 2015.

[8] K. Zhao, D. Khryashchev, J. Freire, C. Silva, and H. Vo, "Predicting taxi demand at high spatial resolution: Approaching the limit of predictability," in *2016 IEEE International Conference on Big Data (Big Data)*, pp. 833–842, 2016.

[9] online, "uber." https://www.uber.com.

[10] H. Yao, F. Wu, J. Ke, X. Tang, Y. Jia, S. Lu, P. Gong, J. Ye, and Z. Li, "Deep multi-view spatial-temporal network for taxi demand prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018.

[11] M. Chen, X. Yu, and Y. Liu, "Pcnn: Deep convolutional networks for short-term traffic congestion prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 11, pp. 3550–3559, 2018.

[12] H. Yu, Z. Wu, S. Wang, Y. Wang, and X. Ma, "Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks," *Sensors*, vol. 17, no. 7, p. 1501, 2017.

[13] S. Guiasu and A. Shenitzer, "The principle of maximum entropy," *The mathematical intelligencer*, vol. 7, no. 1, pp. 42–48, 1985.

[14] P. Cheng, J. Jin, L. Chen, X. Lin, and L. Zheng, "A queueing-theoretic framework for vehicle dispatching in dynamic car-hailing [technical report]," *arXiv preprint arXiv:2107.08662*, 2021.

[15] online, "Nyc taxi & limousine commission trip record data." https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page.

[16] online, "Gaia open dataset." https://outreach.didichuxing.com/appEn-vue/ChengDuOct2016?id=7.

[17] "[online] Technical Report ." https://cspcheng.github.io/pdf/GridTuner.pdf, 2021.

[18] F. Rosenblatt, "Principles of neurodynamics. perceptrons and the theory of brain mechanisms," tech. rep., Cornell Aeronautical Lab Inc Buffalo NY, 1961.

[19] L. Kazemi and C. Shahabi, "Geocrowd: enabling query answering with spatial crowdsourcing," in *Proceedings of the 20th international conference on advances in geographic information systems*, pp. 189–198, 2012.

[20] Y. Tong, J. She, B. Ding, L. Chen, T. Wo, and K. Xu, "Online minimum matching in real-time spatial data: experiments and analysis," *Proceedings of the VLDB Endowment*, vol. 9, no. 12, pp. 1053–1064, 2016.

[21] M. Asghari and C. Shahabi, "Adapt-pricing: a dynamic and predictive technique for pricing to maximize revenue in ridesharing platforms," in *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 189–198, 2018.

[22] L. Zheng, L. Chen, and J. Ye, "Order dispatch in price-aware ridesharing," *Proceedings of the VLDB Endowment*, vol. 11, no. 8, pp. 853–865, 2018.

[23] S. Ma, Y. Zheng, and O. Wolfson, "T-share: A large-scale dynamic taxi ridesharing service," in *2013 IEEE 29th International Conference on Data Engineering (ICDE)*, pp. 410–421, IEEE, 2013.

[24] R. S. Thangaraj, K. Mukherjee, G. Raravi, A. Metrewar, N. Annamaneni, and K. Chattopadhyay, "Xhare-a-ride: A search optimized dynamic ride sharing system with approximation guarantee," in *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, pp. 1117–1128, IEEE, 2017.

[25] L. Chen, Q. Zhong, X. Xiao, Y. Gao, P. Jin, and C. S. Jensen, "Price-and-time-aware dynamic ridesharing," in *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pp. 1061–1072, IEEE, 2018.

[26] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 922–929, 2019.

[27] Y. Zhao, K. Zheng, Y. Cui, H. Su, F. Zhu, and X. Zhou, "Predictive task assignment in spatial crowdsourcing: a data-driven approach," in *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pp. 13–24, IEEE, 2020.

[28] R. M. Karp, U. V. Vazirani, and V. V. Vazirani, "An optimal algorithm for on-line bipartite matching," in *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pp. 352–358, 1990.

[29] Z. He, J. Cao, and X. Liu, "High quality participant recruitment in vehicle-based crowdsourcing using predictable mobility," in *2015 IEEE Conference on Computer Communications (INFOCOM)*, pp. 2542–2550, IEEE, 2015.

[30] N. Cressie and C. K. Wikle, *Statistics for spatio-temporal data*. John Wiley & Sons, 2015.

APPENDIX

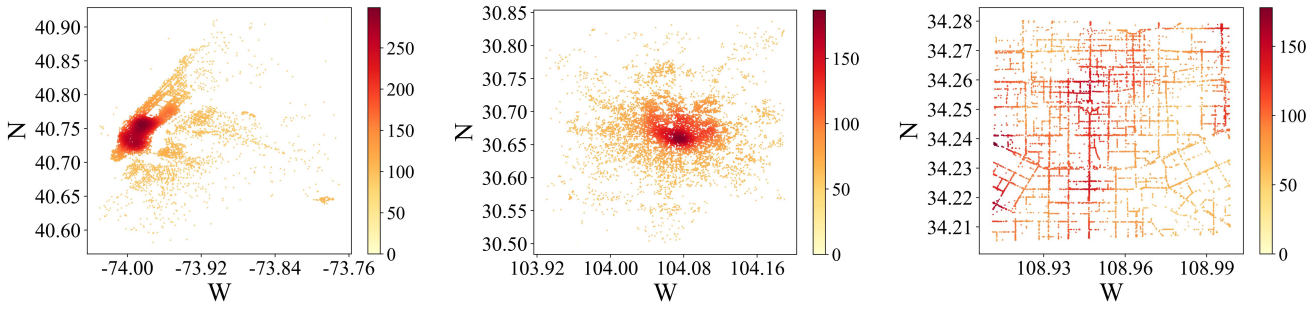
A. Distributions of Order Datasets

Figure 10 shows the distribution of orders in test data set from 8:00 A.M. to 8:30 A.M.

The number of trips in the testing day is: 282,255 for NYC dataset, 238,868 for Chengdu, and 109,753 for Xi'an. We also analyze the distribution of the length of the trips in three cities as shown in Figure 11. The lengths of trips in Chengdu are generally evenly distributed, but the number of long trips (longer than 45 km) is more than 1,000. The taxi trips in NYC mainly happened in Manhattan district, thus most trips are shorter than 15 km. For Xi'an dataset, since the spatial area is relatively small, most trips are shorter than 10 km.

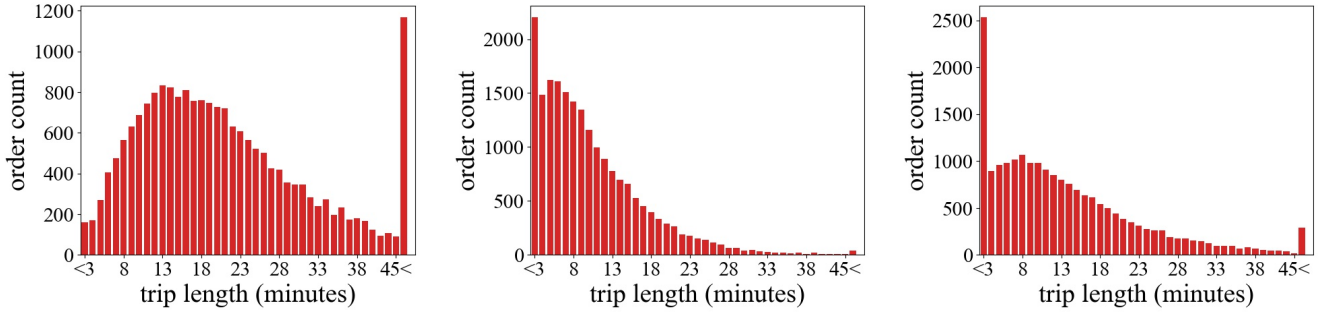
B. Relationship between Expression Error and the Uniformity of the Distribution

Expression error refers to the error caused by estimating the number of events $\hat{\lambda}_{ij}$ in a HGrid r_{ij} with the prediction result $\hat{\lambda}_i$ of the MGrid r_i. The uniformity of the distribution of events within the MGrid will lead to a large expression error. We use $D_\alpha(N)$ to represent the degree of unevenness in the distribution of events within a MGrid. In the experiment, we set the parameter m as 8×8 with $n = 16 \times 16$, which means each MGrid has an area of $3.7578\ km^2$. Then we calculate the imbalance $D_\alpha(64)$ of event distribution within each MGrid and the summation of the expression error $E(i,j)$ of each HGrid r_{ij} in the MGrid r_i. Then, we plot the corresponding relationship between them into a scatter diagram.



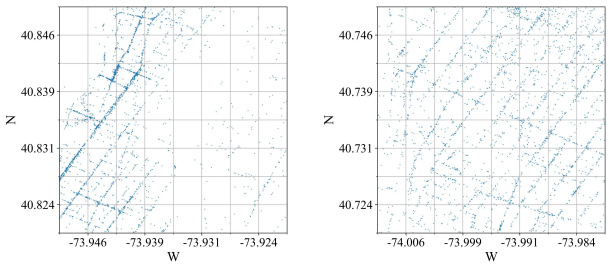
(a) NYC (b) Chengdu (c) Xi'an

Fig. 10: Order Distributions in NYC, Chengdu and Xi'an



(a) Chengdu (b) NYC (c) Xi'an

Fig. 11: Distribution of Trip Length in Different Cities



(a) a Grid with $D_\alpha = 25.87$ (b) a Grid with $D_\alpha = 8.8$

Fig. 12: Two instances of event distribution

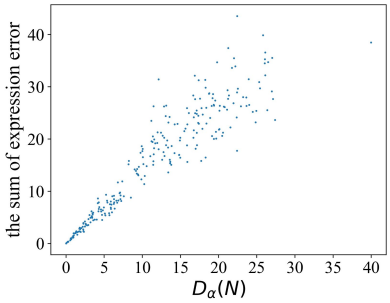


Fig. 13: Effect of $D_\alpha(N)$ on $E_e(i,j)$

Figure 13 shows that the expression error gradually increases with the increase of imbalance of event distribution within a MGrid. On the other hand, we find that many MGrids has $D_\alpha(64) < 10$ because events are unevenly distributed in New York City, leading to the scarcity of events in several grids.

The distribution of events in two different MGrids shown in Figure 12 where each point denotes a spatial event. The event distribution shown in Figure 12(a) is uneven. There is a large empty area in the upper left corner, and there is a long main road in the middle with lots of events. On the contrary, the event distribution in Figure 12(b) is more uniform. The $D_\alpha(64)$ of the right grid is 25.87 with the expression error of 39.90, while the $D_\alpha(64)$ of the left grid is 8.47 with the expression error of 8.8.

C. Results on HGrid Division

The real error is defined in the HGrids. Thus, how to divide the whole space into HGrids becomes a crucial problem. We cannot guarantee that events are evenly distributed within a HGrid while the area of each HGrid is enormous. On the other hand, the calculation complexity of the expression error will be too high if the area of each HGrid is tiny.

We calculate $D_\alpha(N)$ over the whole grids with different values of N based on Equation 2. Then we plot the change in $D_\alpha(N)$ with respect to N in two different way to estimate α_{ij}.

Figure 14 shows that $D_\alpha(N)$ increases with N and the growth rate of $D_\alpha(N)$ slows down while N is greater than a turning point (i.e., approximately 76), which indicates that the distribution of events within the HGrid is uniform when N is greater than 76×76. However, $D_\alpha(N)$ continues to grow

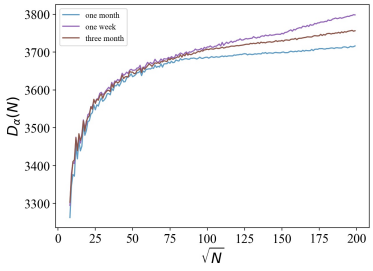


Fig. 14: Effect of N on $D_\alpha(N)$

Fig. 15: Effect of m on Expression Error, Model Error, Real Error

quickly when N is greater than 76×76 due to the estimation of α_{ij} with the sample of a week or three months. Subsequent increases in $D_\alpha(N)$ are mainly attributed to the inaccurate estimation of the value of α_{ij}.

We set n to 16 and keep increasing m to figure out the influence of N on real error, model error and expression error. Figure 15 shows that the real error and the expression error increase as m increases. When m is large, the area of grids decreases, leading to the inaccurate estimation of the value of α_{ij}. Thus, the expression error and the real error continually increase. In this paper, we want to reduce the influence of such inaccuracy on the expression error, and make the expression error to reflect the uneven distribution of events. As a result, we set $N = 128 \times 128$ in our experiments, which allows us to reduce the influence of the inaccurate estimation of the value of α_{ij} and guarantee homogeneous for HGrids.

D. Results on Calculation of Expression Error

We explore the performance of Algorithms 1 and 2 on the calculation of expression error. We $N = 128 \times 128$, $n = 16 \times 16$ and $m = 8 \times 8$. In this case, we calculate the expression error of a HGrid by Algorithms 1 and 2. Theorem III.2 proves that the accuracy of expression error calculation increases with the increase of K. However, the increase in K is accompanied by a rapid increase in computing costs. Figure 16 shows that the cost of the most straightforward algorithm (i.e., no optimizations are made) increases rapidly with the increase of K, and the calculation cost of Algorithm 1 is linearly related to K. However, the calculation time of Algorithm 2 is always kept at a low level, which indicates the effectiveness of the algorithm.

We generally choose to set K as 250 based on the result of Figure 16 to obtain a more accurate result of expression error while avoiding too much computing cost.

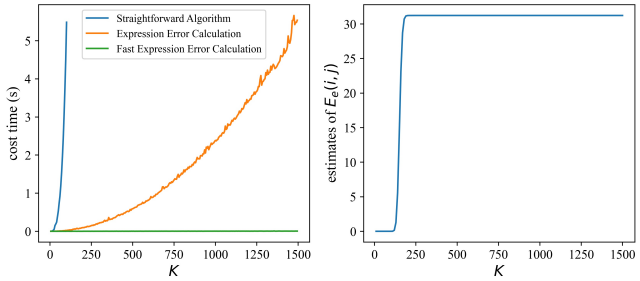


Fig. 16: Effect of K on efficiency for computing expression and accuracy of Algorithm 2

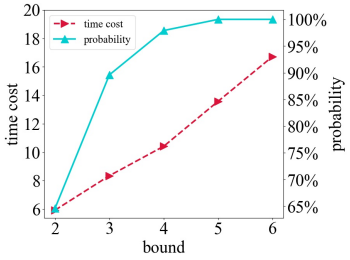


Fig. 17: Effect of *bound* on the Solution and the Cost of Algorithm 5

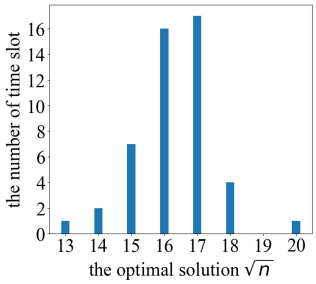


Fig. 18: Distribution of Optimal solutions in different time slots

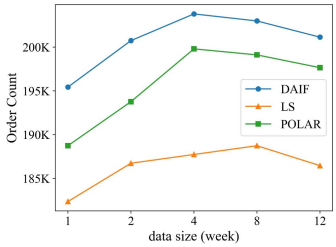


Fig. 19: Effect of the Size of Dataset on the Performance of Different Crowdsourcing Algorithm

E. Influence of bound on the Performance of Iterative Method

Figure 17 shows the effect of the selection of bound on Algorithm 5. With the increase of *bound*, the probability of Algorithm 5 finding the optimal solution increases gradually, and the cost of the algorithm also increases. In addition, Figure 18 shows the distribution of optimal selections of n in 48 periods of a day, which shows that the optimal value of n is 17 for most times.

F. Effect of the size of dataset

The size of dataset should not be too large or too small. For example, 3 months data for training can harm the performance as the distribution may change, thus the estimation of α_{ij} can be inaccurate and cannot find the optimal n. As for training with too small dataset (e.g., one week), the performance also drops since the data is not sufficient to train the prediction models. As shown in Figure 19, the results are best when we use 4 weeks' data as the training set.