Discovering Actual Delivery Locations from Mis-Annotated Couriers' Trajectories

Sijie Ruan[1,2,3,4*], Cheng Long[4], Xiaodu Yang[5], Tianfu He[2,3], Ruiyuan Li[6], Jie Bao[2,3†], Yiheng Chen[7], Shengnan Wu[7], Jiangtao Cui[1], Yu Zheng[1,2,3†]

[1]*Xidian University, Shaanxi, China* [2]*JD iCity, JD Technology, Beijing, China* [3]*JD Intelligent Cities Research, China*

[4]*Nanyang Technological University, Singapore,* [5]*Institute of Information Engineering, Chinese Academy of Sciences, Beijing*

[6]*Chongqing University, Chongqing, China* [7]*JD Logistics, Beijing, China*

sjruan@stu.xidian.edu.cn; c.long@ntu.edu.sg; {xiaodu.yang,tianfudhe}@foxmail.com;ruiyuan.li@cqu.edu.cn

{baojie,chenyiheng,wushengnan1}@jd.com; cuijt@xidian.edu.cn; msyuzheng@outlook.com

***Abstract*—Delivery locations are fundamental data source for intelligent logistics, which can be used in route planning, arrival time estimation, parcel allocation, etc. Using the Geocoded waybill location of an address as the delivery location is not sufficient, due to wrong address parsing, coarse-grained POI database, or different preferences of customers. To mitigate the insufficiency of Geocoding, some methods have been proposed, which utilize couriers' locations when waybills are confirmed to be delivered for delivery location inference. Nevertheless, these methods highly rely on the quality of couriers' annotations and fail when couriers confirm deliveries *with delays*. We propose to infer actual delivery locations of addresses from couriers' trajectories. This idea lies on an observation that the semantics of delivering a parcel are well captured by couriers' trajectories (e.g., a stay point would be generated when a delivery occurs), which holds even couriers confirm deliveries with delays. Specifically, we design Delivery Location Inference under Mis-Annotation (DLInfMA), which (1) generates location candidates from stay points in couriers' trajectories; (2) extracts features from both an address and its location candidates; and (3) uses an attention-based neural network model LocMatcher to predict the delivery location for each address. Experiments on two real-world datasets from JD Logistics as well as synthetic datasets demonstrate the effectiveness, robustness and scalability of DLInfMA. We also present a deployed system along with two applications based on DLInfMA.**

***Index Terms*—trajectory data mining, volunteered geographic information, toponym resolution**

I. Introduction

Express delivery is the main solution to the "last mile" problem in logistics industry. A typical scenario of express delivery is that a courier is assigned with a batch of waybills, each for a parcel and with a shipping address, and needs to send the parcels to their destinations according to the waybills' addresses. Those destinations are called *delivery locations*, each of which can be a doorstep (Figure 1(a)), an express locker which is the parcel storage hub in the neighborhood (Figure 1(b)) or the reception of a residence (Figure 1(c)).

Accurate delivery location knowledge for each address is fundamental for logistics. It not only provides accurate destinations to couriers for delivery references, but also is one of the most important data sources for downstream intelligent applications for business boosting, e.g., route planning [1] and prediction [2], arrival time estimation [3], and parcel allocation [4].

(a) Doorstep. (b) Express Locker. (c) Reception.

Fig. 1. Examples of Delivery Locations.

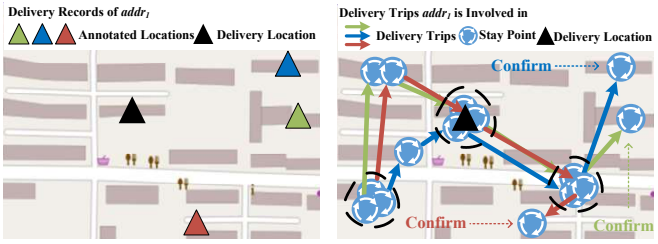
To find the delivery location of an address, a straightforward approach is to use its Geocoding [1] result. Nevertheless, Geocoded locations may not always be accurate due to wrong address parsing, coarse-grained POI database, etc. Furthermore, couriers may deliver a parcel to a doorstep, to an express locker, or to a reception based on the customer's preference, which cannot be uniformly captured by the Geocoded location of the address in the waybill.

To solve these issues, [5], [6] proposed to infer the delivery location of an address based on historical deliveries. This is possible since after delivering each parcel to an address, the courier is supposed to *confirm* the delivery, with which a location would be annotated. Then, it is intuitive to infer the delivery location of the address based on the annotated locations. Specifically, [5] estimates the spatial centroid of those annotated locations as the delivery location. [6] considers all annotated locations as candidates of the delivery location, and leverages a supervised learning approach to select one of the annotated locations as the delivery location. However, these methods highly rely on the quality of the annotated locations, and in cases of mis-annotated locations (which are common since couriers delay the confirmations of deliveries quite often), they would fail. To illustrate, consider the example in Figure 2(a). Suppose that the three historical deliveries

*Partial of this work was done when the author was a visiting student in Nanyang Technological University under the supervision of Cheng Long.

†Yu Zheng and Jie Bao are the corresponding authors.

[1]https://en.wikipedia.org/wiki/Geocoding



(a) Annotation-Based Discovery. (b) Traj.-Based Discovery (Ours).

Fig. 2. Annotation-Based V.S. Trajectory-Based Discovery.

to $addr_1$ are confirmed with delays. In this case, neither the spatial centroid (used by [5]) nor one of the annotated locations (used by [6]) is the actual delivery location.

Fortunately, we observe that in cases of mis-annotated locations, we can still infer the actual delivery locations with couriers' trajectory data. This is because when a courier delays the confirmation of a delivery to an address, while the delivery location is not the annotated location, it must be some location that has been traversed by the courier and could be inferred from the courier's trajectory (specifically, the portion before the time when the courier confirms the delivery[2]). Specifically, we can extract stay points [7] from the trajectory as candidates of the delivery location, since a delivery would usually cause a staying behavior of the courier.

To illustrate, consider the example in Figure 2(b). Since $addr_1$ is involved in 3 delivery trips, we can extract stay points (denoted as the blue circles) before the deliveries to $addr_1$ are confirmed, each of which can possibly be the actual delivery location. In addition, the delivery location is more likely to be within one of regions marked with black dashed circles (since they are all involved in all 3 trips).

However, to implement the above idea, there are two challenges:

1) *Redundant candidates*: Stay points in different trips could have minor location discrepancy even if they represent the same location in real world. The redundant candidates would increase the difficulty of the delivery location inference task.
2) *Complex decisions*: Deciding the delivery location is complex and challenging. To illustrate, consider Figure 2(b). There is no clear clue which of those black dashed circles is more likely to be the delivery location.

To solve the above challenges, in this paper, we propose a delivery location discovery method called Delivery Location Inference under Mis-Annotation (DLInfMA) for cases where the delivery times may be recorded with delays. It mainly consists of two steps: (1) To mitigate the issue of redundant candidates, it generates a pool of candidates of delivery locations (called location candidates) as clusters of those stay points extracted from couriers' historical trajectories. Once a pool of location candidates is generated, we can easily filter for an address a set of location candidates, and the actual delivery location should be in the set; (2) To solve the complex decision problem, it first extracts several key features from both an address and its filtered location candidates, e.g., the occurrence of a location candidate when delivering for the address, and the uniqueness of a location candidate to the address. Then an attention-based selection model LocMatcher is proposed to predict the actual delivery location of an address, which jointly considers its location candidates.

The main contributions are summarized as follows.

- We propose to leverage couriers' trajectories for the delivery location inference problem with deliveries confirmed with potential delays, which cannot be well handled by existing annotation-based solutions (Section II).
- We propose a method, namely DLInfMA, which first generates location candidates of an address, and then uses an attention-based model LocMatcher to infer the delivery location (Section III and IV).
- Extensive experiments on two large scale real-world datasets from JD Logistics as well as six synthetic datasets show the effectiveness, robustness and scalability of DLInfMA. It outperforms the best baselines by 4%-10% in the most competitive metric and can infer 1K addresses/s (Section V).
- We present the deployed system based on DLInfMA and also two application scenarios on top of the system, namely route planning and customer availability inference, which are both used internally in JD Logistics (Section VI).

II. Overview

A. Definitions

Definition 1 (Waybill). A waybill contains the information about the delivery of a parcel, which is denoted as a 3-tuple $w = (addr, t_{re}, t_d)$. $addr$ is the address, t_{re} is the time when a courier received the parcel, and t_d is the recorded delivery time, which can be significantly delayed.

Definition 2 (Delivery Location). A delivery location is a spatial point, denoted as $l_d = (x, y)$, at which a courier drops off the parcel.

Definition 3 (Trajectory). A trajectory is a sequence of spatio-temporal points generated by a certain courier, denoted as $T =< p_1, p_2, ..., p_{|T|} >$, where each point $p = (x, y, t)$ indicates the physical appearance at a location (x, y) at time t. Points in a trajectory are organized chronologically, namely, $\forall i < j : p_i.t < p_j.t$.

Definition 4 (Stay Point). A stay point is a subsequence of a trajectory, which semantically means that a moving object stays in a geographic region for a while. Formally, given a distance threshold D_{max} and a time threshold T_{min}, $< p_i, p_{i+1}, ..., p_j >$ is called a stay point sp if $distance(p_i, p_k) \leq D_{max}(\forall k \in [i+1, j])$, $distance(p_i, p_{j+1}) > D_{max}$ (if $j+1 \leq n$), and $|p_j.t - p_i.t| \geq T_{min}$. The time of a stay point sp is defined as the middle point of its time interval, and the location of sp is defined as its spatial centroid.

[2]Delayed confirmations occur much more frequently than fake deliveries meaning that couriers confirm the deliveries before they actually happen. The latter would cause severe punishment to the courier.

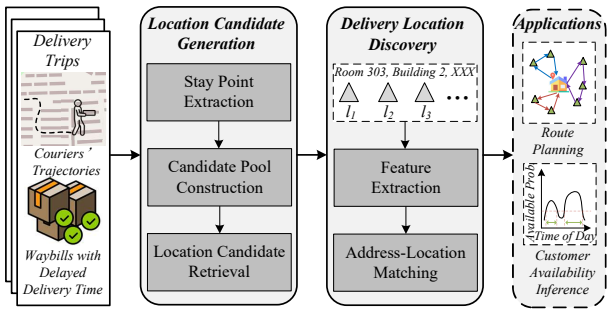


Fig. 3. Framework of DLInfMA.

Definition 5 (Delivery Trip). A courier delivers a batch of parcels to customers in a delivery trip, which is denoted as $tr = (t_s, t_e, T, W)$, where t_s is the start time, t_e is the end time, T is the courier's trajectories during tr, and W is the set of waybills delivered in tr.

Problem Definition. *Given a set of delivery trips TR, for each address $addr \in \mathcal{A}$, where $\mathcal{A} = \{w.addr | w \in \bigcup_{tr \in TR} tr.W\}$, the problem is to infer its delivery location l_d.*

In our datasets, we found addresses in the same building could have different delivery locations. Therefore, we choose the delivery location inference at the address level. But our solution can also be easily adapted to building-level inference.

B. Framework

The framework of DLInfMA is presented in Figure 3, which takes historical delivery trips of couriers as input, and gives the delivery location inference result of each address. The inferred delivery locations are used for various downstream applications, e.g., route planning and customer availability inference, which would be introduced in Section VI. DLInfMA consists of the following two components:

Location Candidate Generation. This component takes trajectories and waybills of all delivery trips as inputs, and performs three main tasks: (1) *Stay Point Extraction*, which extracts stay points from trajectories; (2) *Candidate Pool Construction*, which generates a pool of delivery location candidates based on stay points; and (3) *Location Candidate Retrieval*, which retrieves the delivery location candidates for each address in its involved trips (detailed in Section III).

Delivery Location Discovery. This component selects for each address, the best matched location as the inferred delivery location among all its delivery location candidates. Two main tasks are performed: (1) *Feature Extraction*, which extracts features from an address and its corresponding location candidates; and (2) *Address-Location Matching*, which infers the delivery location of an address leveraging an attention-based model (detailed in Section IV).

III. Location Candidate Generation

In this component, we aim to obtain for each address its delivery location candidates. First, stay points are extracted from trajectories. Then, all stay points are clustered to generate a pool of location candidates. Finally, for each address, we retrieve its delivery location candidates based on the trips involving it and its recorded delivery times.

A. Stay Point Extraction

This step extracts stay points from couriers' trajectories. Two operations are performed sequentially: (1) noise filtering, which filters noisy GPS points with a heuristics-based method [8] to improve the quality of stay point extraction; (2) stay point detection [7], which detects stay points from cleaned trajectories based on Definition 4. We set $D_{max} = 20$m and $T_{min} = 30$s, same as [5].

B. Candidate Pool Construction

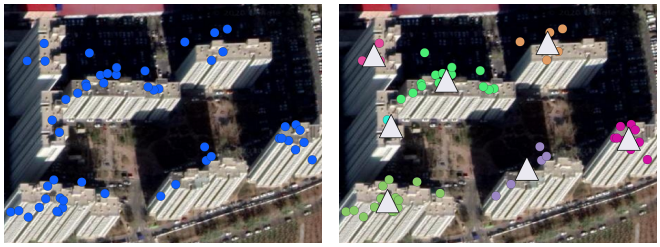
In this step, we aim to obtain a pool of delivery location candidates based on all stay points of couriers. Due to sensing errors, locations of stay points could have minor discrepancy from each other even if they represent the same location in reality. Therefore, we perform clustering over stay points, so that each location can be uniquely represented and serve as a delivery location candidate. In addition, we extract profiles of each location generated based on stay points, and use these profiles in the delivery location discovery component.

Clustering methods such as k-Means [9], DBSCAN [10], OPTICS [11], grid merging [12] have been adopted for generating locations from stay points. In this paper, we adopt the *hierarchical clustering* [13] to generate delivery location candidates, which enjoys following advantages. (1) It requires only a distance threshold D to control how many locations to be generated, which is easier to set (compared with the number of clusters required in k-Means and the density required in density-based methods). (2) It returns clusters that are more suitable to describe delivery locations (compared with irregular cluster shapes generated by aforementioned methods). (3) It does not require to discretize the urban space (as the grid merging method).

The hierarchical clustering first treats each stay point as a cluster, then iteratively merges two clusters whose centroids are the closest to each other to form a new cluster, until there does not exist two clusters such that the distance of their centroids is smaller than D. The centroid of each cluster is treated as a location candidate. Furthermore, for better efficiency, we generate location candidates based on stay points in a bi-weekly manner and then merge the newly generated location candidates with existing ones with the same clustering process.

Figure 4 gives an example of the candidate pool construction. The blue points in Figure 4(a) are stay points generated from couriers. They are merged into different clusters using the hierarchical clustering ($D = 40$m), which are shown in different colors in Figure 4(b). And the centroid of each cluster (a grey triangle) corresponds to a generated location candidate.

Additionally, we use the stay points in each cluster to generate profiles of the location, which are useful for the following location inference task. The profiles include: (1) *Average duration*, which is the average stay duration of stay



(a) Detected Stay Points. (b) Clusters & Locations.

Fig. 4. Examples of Candidate Pool Construction.

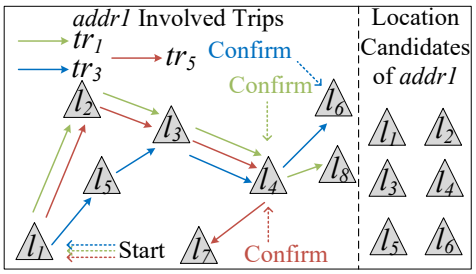


Fig. 5. Illustration of Location Candidate Retrieval.

points. The duration of a delivery location should be neither too long nor too short. (2) *Number of couriers*, which is the number of couriers who have visited the location. The delivery tasks in a certain region are usually assigned to the same courier. Thus, it is unlikely that a delivery location is visited by many couriers. (3) *Time distribution*, which is the time distribution of couriers visiting the location. We discretize the time interval [8:00,23:00) into hourly timeslots, and calculate the visiting time distribution. The active timeslots of delivery locations and those of other locations can be different.

C. Location Candidate Retrieval

In this step, we want to retrieve for each address the delivery location candidates using trips involving it and recorded delivery times of waybills. It is based on the observation that the recorded delivery time provides a temporal *upper bound* to filter out impossible location candidates. Specifically, a stay point which has the time later than the recorded delivery time and its corresponding location candidate cannot be the delivery location. Figure 5 gives an example of retrieving location candidates of address $addr_1$. Suppose parcels with address $addr_1$ are involved in trips tr_1, tr_3 and tr_5. The text "Confirm" indicates the (recorded) delivery time of the parcel, which might involve some delay. Therefore, in each trip that $addr_1$ is involved in, we only consider the location candidates with the time (of a stay point) no later than the recorded delivery time. The delivery location candidates of $addr_1$ is the union of candidates in all trips that $addr_1$ is involved in, i.e., $\{l_1, l_2, l_3, l_4, l_5, l_6\}$. Note that we do not exclude locations that are visited by some but not all trips (e.g., l_2 and l_5) with the GPS noises taken into consideration.

IV. Delivery Location Discovery

In this component, we infer the delivery location for each address based on its location candidates. We first extract features from each address and its location candidates, and then use an attention-based model to match each address to its most probable delivery location.

A. Feature Extraction

For each address, this step is to extract features from the address and its location candidates. We extract three types of features based on each address $addr_j$ and its location candidates:

(1) Matching Features. The matching features are related to the interactions between the location candidate and the address.

- *Trip coverage*: The trip coverage $TC_{i,j}$ is defined as the fraction of trips that pass through the location candidate l_i, among all trips that involve a waybill with the address $addr_j$. Formally, it is defined as follows.

$$TC_{i,j} = \frac{\sum_{tr \in TR_j} \mathbb{1}(l_i \in L_{tr})}{|TR_j|} \tag{1}$$

  where TR_j is the set containing all trips that involve a waybill with the address $addr_j$, L_{tr} is the set of location candidates that are passed by the trip tr, and $\mathbb{1}(\cdot)$ is an indicator function judging whether the condition is held. Intuitively, a larger $TC_{i,j}$ implies that l_i should be inferred as the delivery location of $addr_j$ with higher confidence. For example, as shown in Figure 5, $addr_1$ is involved in three trips. l_1, l_3 and l_4 are visited by all the three trips, and thus their trip coverage are both $3/3 = 1$. l_2 is visited by two trips, and thus its trip coverage is $2/3$.
- *Location commonality*: While the trip coverage captures the co-occurrence of a location candidate and an address, it does not distinguish/penalize those cases where the location candidate is visited by many other trips that do not involve the address. To mitigate this issue, we introduce the "location commonality" feature. We denote the building of $addr_j$ as $B(addr_j)$[3]. The location commonality $LC_{i,j}$ is defined as the fraction of trips that pass through the location candidate l_i, among all trips that do not involve any address whose building is the same as $B(addr_j)$. Formally, it is defined as follows.

$$LC_{i,j} = \frac{\sum_{tr \in TR \setminus TR_{B(addr_j)}} \mathbb{1}(l_i \in L_{tr})}{|TR \setminus TR_{B(addr_j)}|} \tag{2}$$

  where $B(addr_j)$ is the building of $addr_j$, $TR_{B(addr_j)}$ is the set of trips that involve an address sharing the same building with $addr_j$, and L_{tr} is the set containing all location candidates that are passed through by the trip tr. Intuitively, a larger $LC_{i,j}$ implies that the l_i should be inferred as the delivery location of $addr_j$ with lower confidence. Note that we count the trips that do not involve

[3]The building is extracted from a commercial address segmentation and tagging tool.

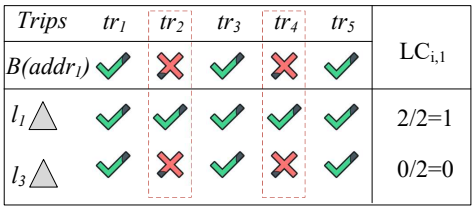


Fig. 6. Illustration of Location Commonality (a correct mark (resp. a wrong mark) means a building or a location is (resp. is not) involved in a trip).

$B(addr_j)$ instead of $addr_j$ since otherwise when there are many trips that involve $B(addr_j)$ but not $addr_j$ and these trips also pass through l_i, it should be treated as a non-negative signal that l_i is the delivery location of $addr_j$ but $LC_{i,j}$ would be high (which corresponds to a negative signal).

For example, apart from the three trips tr_1, tr_3 and tr_5 shown in Figure 5, we consider two additional trips tr_2 and tr_4 which do not involve an address whose building is $B(addr_1)$. Suppose tr_2 and tr_4 involve l_1 but not l_3. Figure 6 summarizes the information of how trips involve $B(addr_1)$ and locations l_1 and l_3. Consider locations l_1 and l_3. They have the same trip coverage, both equal to 1, but different location commonalities, the former equal to 2/2=1 and the latter equal to 0/2=0, indicating that l_1 has a lower chance to be the delivery location of $addr_1$ compared to l_3.

- *Distance*: The distance is the geographical distance between l_i and the Geocoded location G_j of $addr_j$. Though G_j may not be close to the actual delivery location, it roughly determines the local community based on the address, thus can help filter some locations that are far away, since they are highly unlikely delivery locations.

(2) Profile Features. The profile features are the property of a location, which is detailed in Section III-B. There are three features: *Average duration*, *Number of couriers* and *Time distribution*.

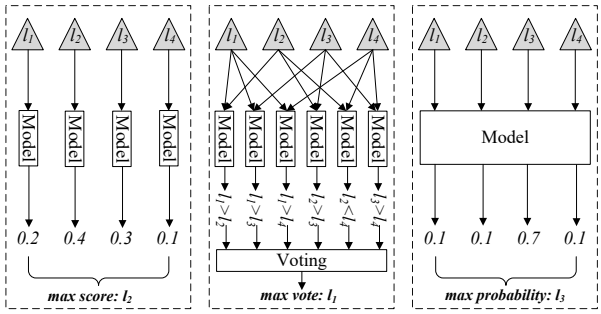
(3) Address Features. The address features are the inherent property of the address, which is irrelevant to locations.

- *Number of deliveries*: It is the number of trips that the address $addr_j$ is involved in. If the number of deliveries is larger, the trip coverage feature is more reliable.
- *POI category*: It is returned along with the Geocoded location by Geocoding. There are 21 categories (e.g., a company), which influence the average stay duration at a location.

B. Address-Location Matching

This step is to use a model to find for an address its delivery location among location candidates based on extracted features.

One approach is to concatenate three types of features and train a binary classification model to predict whether each individual location candidate is the delivery location of the address, and during the inference phase, we select for each address the location candidate with the highest score



(a) Classification-based. (b) Pairwise Ranking-based. (c) Attention-based (Ours).

Fig. 7. Different Location Selection Strategies.

as the delivery location, which is illustrated in Figure 7(a). This method independently classifies each location candidate, while the relationship among candidates, e.g., the ranking of average duration and the ranking of trip coverage, is not considered. Another approach is to formulate the delivery location inference as a location ranking problem, where a pairwise ranking strategy is commonly used [6], [14], [15]. As shown in Figure 7(b), the ranking model predicts the partial order of a pair of location candidates. In order to infer the delivery location, the location candidate which wins the most number of comparisons in a voting manner is selected. However, there actually does not exist an order between negative location candidates in our problem setting, and the ordering is more than necessary since we only care about the actual delivery location.

Then a natural question is whether we can consider all location candidates jointly at once by feeding them to a single model as shown in Figure 7(c). One of the major challenges to implement this idea is that for different addresses, the number of location candidates is varying (which is different from the case of pairwise ranking-based methods, which always takes two location candidates).

We borrow the idea from natural language processing (NLP), which always accepts sentences with varying numbers of words as input using RNN [16] or Transformer [17]. Here, we adopt Transformer encoder for the correlation modeling for two reasons: 1) there is no temporal dependency among location candidates; 2) the number of location candidates of an address might be very large, and it is challenging for RNN to capture the long-term dependency if two candidates are far away in the input sequence. After the correlations among candidates are modeled, we use another attention mechanism to predict the actual delivery location using the address features as the context vector.

The overall structure of our proposed model, namely LocMatcher, is shown in Figure 8, which takes location candidate features (including matching features and profile features) and address features as inputs in different stages, and generates the probability distribution among location candidates.

For each location candidate, the time distribution is first fed into a dense layer with r neurons to generate a dense representation, which is then concatenated with other profile features

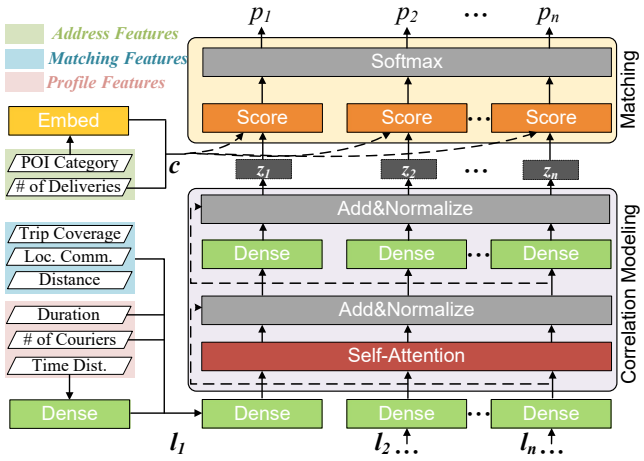


Fig. 8. LocMatcher Architecture.

and matching features to form the location feature input of a candidate, e.g., $\mathbf{l}_1$. Then, it is fed into a dense layer with z hidden units to generate a high-dimension representation. After that, the set of representations from all location candidates are sent into a transformer encoder for the correlation modeling. The transformer encoder contains N layers, each of which consists of two sublayers: the first is a multi-head self-attention mechanism, and the second is a position-wise feed-forward network. The residual connection is added around each of the two sub-layers, followed by layer normalization. The output of the transformer encoder are location embeddings, which have their inter-relationship modeled. We use $\{\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_n\}$ to denote them, where n is the number of location candidates, $\mathbf{z}_i \in \mathbb{R}^z$. Next, a similar attention mechanism like [18] is used to select a delivery location among candidates given location embeddings. We generate a context vector $\mathbf{c} \in \mathbb{R}^m$ based on the concatenation of POI category embedding and the number of deliveries. Given the k-th location embedding, the matching score s_k between the context vector $\mathbf{c}$ and the location embedding $\mathbf{z}_k$ is calculated as follows:

$$s_k = \mathbf{v}^\top \tanh(\mathbf{W}\mathbf{z}_k + \mathbf{U}\mathbf{c} + \mathbf{b}) \tag{3}$$

where $\mathbf{v}, \mathbf{b} \in \mathbb{R}^p$, $\mathbf{W} \in \mathbb{R}^{p \times z}$, and $\mathbf{U} \in \mathbb{R}^{p \times m}$ are the parameters to be learned, and p is a hyper-parameter.

Then, s_k is normalized over all candidates by the softmax to generate the probability p_k:

$$p_k = \frac{\exp(s_k)}{\sum_{i=1}^{n} \exp(s_i)} \tag{4}$$

During training, the label is represented using a one-hot vector (i.e., the index of the actual delivery location among candidates is marked as 1), and we use the cross-entropy loss of the label and the predicted probability vector (all predicted probabilities of different candidates) to optimize the model. In the inference process, we match the address to the location candidate with the maximum predicted probability.

TABLE I
STATISTICS OF DATASETS.

Data Description	DowBJ	SubBJ
# of Delivery Stations	3	8
# of Couriers	82	105
# of Delivery Trips	31,754	26,010
# of GPS Points	33.5M	32.6M
# of Waybills	1.4M	1.0M
# of Addresses (Train&Val)	24,442 & 3,024	16,286 & 2,824
# of Addresses (Evaluation)	5,016	5,145

V. EXPERIMENTS

In this section, we perform extensive experiments to evaluate the proposed method. The effectiveness of DLInfMA is demonstrated on two real-world datasets and six synthetic datasets that are derived from the real datasets. Then case studies are conducted to further show the advantages of DLInfMA. Finally, the scalability and efficiency of DLInfMA are discussed.

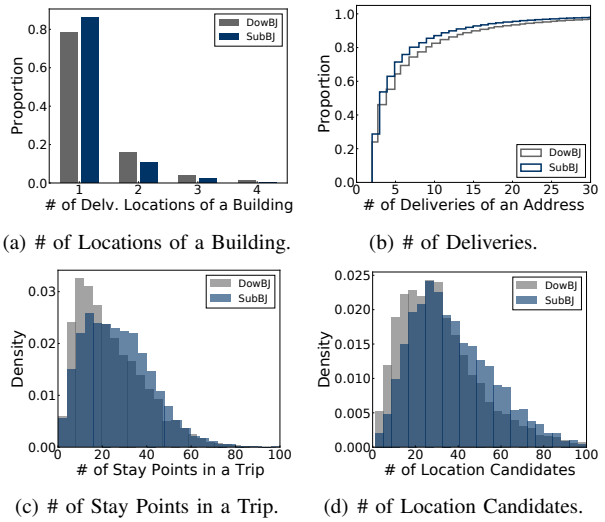
A. Datasets

We use real-world datasets from JD Logistics for evaluation[4]. They are collected inside/outside the 3rd Ring of Beijing, which are named as DowBJ/SubBJ, respectively, and have different data statistics, e.g., the precision of Geocoding and the number of historical deliveries per address. Both datasets cover 20 months (from Jan. 1st, 2018 to Sept. 1st, 2019), and the average sampling rate of trajectories is 13.5s. We split datasets into training, evaluation and testing according to disjoint spatial regions to make sure there is no delivery location overlaps. The details of each dataset are summarized in Table I. The ground-truth delivery locations are carefully labelled by couriers. Since it may not always be the same as the generated location candidates, for supervised learning methods that select location from candidates, the positive labels are obtained using the nearest candidates to the ground-truth delivery locations. The synthetic datasets (which are derived from real datasets) will be described when presenting the experiments on synthetic datasets in Section V-D. We provide some detailed statistics of the real datasets as follows.

Delivery Location Distribution. Figure 9(a) shows the distribution of the number of different delivery locations of addresses in the same building in both datasets. As observed, the phenomenon that addresses in the same building has different delivery locations is quite common among different spatial regions. It shows the diversity of delivery locations, and the necessity to perform the address-level delivery location inference. More specifically, more than 22% (14%) buildings have affiliated addresses with more than one delivery location in DowBJ (SubBJ), respectively.

Times of Delivery Distribution. Figure 9(b) shows the accumulated distribution of the number of deliveries of an address in both datasets. It can be observed that SubBJ has slightly

[4]The dataset in [6] is an annotation-based delivery dataset, which is inapplicable to our trajectory-based delivery location inference.



(a) # of Locations of a Building. (b) # of Deliveries.

(c) # of Stay Points in a Trip. (d) # of Location Candidates.

Fig. 9. Dataset Descriptions.

more addresses with fewer deliveries compared to DowBJ, which indicates customers within the 3rd Ring order more frequently. Half of addresses have less than 5 deliveries in DowBJ, while less than 4 in SubBJ. We can also notice that, in both datasets, there are also addresses associated with many deliveries due to some active customers.

Stay Point Distribution. Figure 9(c) shows the distribution of the number of stay points in a trip in both datasets, which indicates that there could be many location candidates for an address if the delivery time is recorded with delays. For each delivery trip, there are more stays in SubBJ than in DowBJ due to more delivery locations in outer 3rd Ring. The average number of stay points is 24 in DowBJ and 27 in SubBJ.

Location Candidate Distribution. Figure 9(d) shows the distribution of the number of location candidates for an address in both datasets. As observed, the number of candidates for an address is more than that of stay points in a trip, because the location candidates are generated based on all trips. The average number of candidates is 32 in DowBJ and 38 in SubBJ, which indicates the delivery location inference could be more difficult in SubBJ given more candidates.

B. Experimental Settings

Baselines. We compare DLInfMA with the following baselines:

- Geocoding: This method simply uses the Geocoding result of the address as the inferred delivery location.
- Annotation [5]: The spatial centroid of the address's annotated locations is inferred as its delivery location.
- GeoCloud [19]: This is a clustering-based variant of Annotation, which performs DBSCAN over annotated locations and uses the spatial centroid of the biggest cluster as the delivery location.
- GeoRank [6]: This method is also based on Annotation. For each address, all annotated locations are regarded as delivery location candidates. A pairwise ranking strategy is employed to train a ranking model with the decision tree as the base learner. During the inference, the candidate that wins the most pairwise comparisons is selected as the delivery location.
- UNet-based [20]: This method infers the delivery locations based on annotated locations and customers' locations. We adapt this method by removing the later one for comparison fairness (since our problem does not take customers' locations as inputs). It treats the task as an image semantic segmentation problem. For each address, it first creates an empty spatial matrix/image centered at the GeoHash grid with the most number of annotated points. The image size is 9×9, and each pixel denotes a grid with resolution GeoHash 8 (about 32m$\times$19m). The value of each pixel is calculated based on the annotated locations in the grid. Then, a semantic segmentation model, i.e., UNet [21], is employed over the spatial matrix to infer the delivery location.
- MinDist: This method selects the nearest location to the Geocoded waybill location among location candidates.
- MaxTC: This heuristic method selects the location among candidates with the maximum TC.
- MaxTC-ILC: This heuristic method is inspired by the well-known information retrieval algorithm TF-IDF [22], which selects the location among candidates with the maximum TC-ILC, defined in Equation (5).

$$TC\text{-}ILC_{i,j} = TC_{i,j} \times \lg \frac{1}{LC_{i,j}} \tag{5}$$

For baseline methods Annotation, GeoCloud, GeoRank and UNet-based, the annotated locations could be easily generated based on the trajectory data (based on the time stamps of confirmed deliveries).

Variants. We also compare DLInfMA with four types of variants to show the effectiveness of each component of DLInfMA:

- DLInfMA-GBDT, DLInfMA-RF, DLInfMA-MLP: These methods use the same location candidate generation method, but adopt different classification models (i.e., gradient boosting tree [23], random forest [24], MLP [25]) to classify with extracted features whether a location candidate is the delivery location of the address. During the inference, we choose for each address the location candidate of the highest probability.
- DLInfMA-RkDT, DLInfMA-RkNet: These methods leverage the pairwise ranking strategy, but adopt different base learners (namely decision tree and RankNet [26]) to infer the delivery location, while others remain the same as DLInfMA.
- DLInfMA-PN: This method replaces the transformer in LocMatcher with an LSTM as [18] did.
- DLInfMA-Grid: This method is similar to DLInfMA, except that location candidates are generated by the grid merging [12].

Evaluation Metrics. We use three metrics, i.e., MAE, P95 and β_δ, to evaluate the performance of the methods considered:

- MAE: it is used to characterize the average inference error over all evaluation samples.
- P95: it is the 0.95 percentile error adopted from [6], which cares more about the performance of bad cases, since in some application scenarios, e.g., parcel assignment, the occasional large inference errors can cause huge business loss.
- β_δ: it is the percentage of evaluation samples with error less than a given distance threshold δ. As mentioned in [20], the delivery business would not be impacted much if the error is bounded by a small threshold.

Formally, MAE and β_δ are defined as follows:

$$\text{MAE} = \frac{\sum_{j=0}^{|\mathcal{A}|} dist(\hat{l}_d^j, l_d^j)}{|\mathcal{A}|} \tag{6}$$

$$\beta_\delta = \frac{\sum_{j=0}^{|\mathcal{A}|} \mathbb{1}(dist(\hat{l}_d^j, l_d^j) < \delta)}{|\mathcal{A}|} \times 100\% \tag{7}$$

where l_d^j and $\hat{l}_d^j$ is the ground-truth and inferred delivery location of the address $addr_j$, respectively, $dist(\cdot,\cdot)$ is the geospatial distance between two given locations, and δ is a given distance threshold.

In following experiments, the unit of MAE and P95 is in meters, and δ is set to 50m by following [20].

Training Details & Hyperparameters. In candidate pool construction, the distance threshold D is set to 40m. We use the grid search to find the best hyperparameters for each method. In LocMatcher, POI category is embedded to $\mathbb{R}^3$, r is 3, z is 8 and p is 32, the transformer encoder contains 3 layers, each of which contains 2 heads and 32 neurons in the dense sublayer, and the dropout is set to 0.1. We use Adam [27] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ to perform training with a learning rate $1e$-4 and batch size 16. We also reduce by half the learning rate every 5 epochs. The training is stopped once the validation loss no longer decreases. In GeoCloud, we set the minimum number of points to 1 to make sure we can also perform the clustering even if an address has been delivered only for a few times. In GeoRank and DLInfMA-RkDT, there are 1024 leaf nodes at maximum. For all classification based methods, the class weight is set to 8:2 due to imbalanced labels. In DLInfMA-GBDT, the number of boosting stages is 150. In DLInfMA-RF, the maximum depth is 10, and the number of trees is 400. In DLInfMA-MLP and DLInfMA-RkNet, features are fed into 1 hidden layer with 16 neurons. In DLInfMA-PN, the LSTM has 32 neurons. In DLInfMA-Grid, the grid size is $D/3$, so that the maximum size of a location is bounded in $D \times D$, which is similar to ours.

Implementations. Our algorithms are implemented in Python with PyTorch and Scikit-Learn. Experiments are conducted on a docker with 16 Cores@2.2GHz, 64GB memory and Red Hat Linux in the internal environment of JD for privacy protection.

C. Effectiveness Evaluation on Real-world Datasets

Overall Evaluation. We report the overall performance of DLInfMA on two datasets compared with baselines in Table II. We have following observations:

TABLE II
OVERALL EFFECTIVENESS EVALUATION.

Methods	DowBJ			SubBJ		
	MAE	**P95**	β_{50}	**MAE**	**P95**	β_{50}
Annotation	160.7	533.8	32.5	251.9	910.4	32.6
MaxTC	179.9	960.6	50.9	295.1	1377.9	39.4
Geocoding	146.4	476.9	55.7	156.0	477.4	46.7
MinDist	73.4	225.0	59.0	115.3	381.5	51.8
GeoCloud	88.4	254.8	60.0	136.6	449.4	56.4
MaxTC-ILC	64.2	268.5	68.7	118.9	379.5	52.3
GeoRank	46.5	149.2	78.7	90.9	335.6	70.8
UNet-based	107.3	1018.9	80.8	332.9	2045.1	65.0
DLInfMA-RkDT	37.0	149.0	80.0	67.1	278.6	70.3
DLInfMA-Grid	34.3	145.9	80.9	66.0	288.8	69.1
DLInfMA-MLP	36.2	154.8	81.5	63.3	286.9	71.8
DLInfMA-GBDT	34.7	136.5	82.1	59.9	272.6	73.6
DLInfMA-RF	34.3	137.4	82.1	57.0	268.9	73.9
DLInfMA-RkNet	34.4	140.3	82.5	58.8	265.1	74.3
DLInfMA-PN	33.7	126.9	83.9	53.4	243.5	76.9
DLInfMA-nTC	61.3	183.6	64.1	110.6	397.9	55.6
DLInfMA-nD	55.3	257.2	72.9	68.7	280.0	69.8
DLInfMA-nLC	35.6	143.4	81.3	49.8	238.9	78.1
DLInfMA-nP	34.8	146.9	82.6	64.0	289.6	72.0
DLInfMA-LC$_{\text{addr}}$	34.0	134.1	82.8	50.1	238.0	77.6
DLInfMA-nA	33.2	136.3	83.2	53.3	248.1	75.8
DLInfMA (Ours)	**31.3**	**123.7**	**84.1**	**48.2**	**231.0**	**77.8**

- Annotation and MaxTC are the worst. For Annotation, its performance highly depends on the annotation behavior of couriers, which is not always good among all couriers. MaxTC does not perform well in these datasets either, which indicates it is insufficient to only consider TC, since there are common locations that a courier would pass by frequently in many trips.
- The third worst is Geocoding, which is used as the delivery location in practice by default. Its large errors call for a better method to infer delivery locations.
- MinDist is better than Geocoding, indicating that though we cannot treat the Geocoded location as the delivery location, the actual delivery location is usually not that far from it.
- GeoCloud and MaxTC-ILC show competitive results. GeoCloud can be regarded as an unsupervised variant of Annotation, which considers the mis-annotation cases, and leverages a clustering technique to filter out mis-annotated locations. As for MaxTC-ILC, though only one feature inverse LC is added, it significantly outperforms MaxTC, which shows its ability to penalize those location candidates passed by many trips.
- GeoRank and UNet-based, the supervised variants of Annotation, are the best baselines on β_{50}. They show the necessity to use a supervised learning model to infer the delivery locations. However, all annotation-based methods only consider the annotated locations for the delivery location inference, while rich information from trajectories, e.g., the locations visited before the delivery confirmation and trip related features, is dismissed.
- Though UNet-based has a relatively good performance on β_{50}, its MAE and P95 are not ideal. We think the reason is two-fold: 1) It needs to locate a center GeoHash grid based on annotated locations to create the spatial density matrix. If

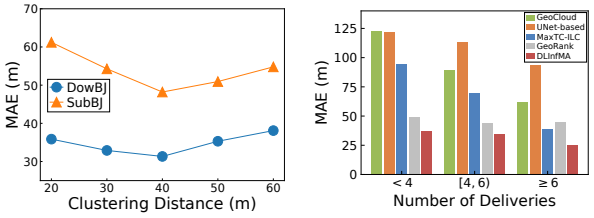
the annotated locations are rather noisy, the actual delivery location might be out of the spatial density matrix (i.e., the model has no chance to make a correct prediction); 2) It uses the spatial center of the predicted GeoHash grid to represent the inferred delivery location, which might not be accurate enough.

- Our method DLInfMA consistently outperforms baselines over three metrics. Its MAE and P95 are much better than baselines, which shows its average error is the lowest and it is able to correctly infer delivery locations for more addresses. Its β_{50} is 84.1% on DowBJ and 77.8% on SubBJ, which outperforms the best baselines by 4% and 10%, respectively. The results of two datasets are slightly different, which is due to the different Geocoding precision and distributions of the number of deliveries per address in the two datasets.

Ablation Studies. Among the variants, classification-based and pairwise ranking-based methods are worse, because they fail to consider all candidates jointly. As expected, DLInfMA-RkNet is generally better than classification-based methods, since it models the pairwise relationship. Interestingly, we found DLInfMA-RkDT is worse than classification-based methods. We guess the reason is that the simple decision tree model can not handle some sparse input in our features, e.g., the time distribution, while the neural network based method shows a better performance. DLInfMA-PN is worse than DLInfMA due to the long-term dependency issues. We find that the locations generated by the DLInfMA-Grid are many more than ours, because there might be two locations that are very close at the grid boundary, which may degrade the performance of the location inference.

Importance of Features. The importance of features is also shown in Table II. DLInfMA-nTC, DLInfMA-nD, DLInfMA-nP, DLInfMA-nLC, and DLInfMA-nA are DLInfMA with the features of the trip coverage, the distance, the location profile, the location commonality, and the address dropped. Note that DLInfMA-nA is achieved by removing term $\mathbf{Uc}$ in Equation (3). It shows that both TC and distance are very important: when they are removed, the performance is significantly degraded. The LC, location profile and address also help as the results of DLInfMA-nLC , DLInfMA-nP and DLInfMA-nA show. DLInfMA-LC_{addr} replaces the building-based LC with the address-based one. It is less effective than DLInfMA, which is in line with our expectations, because when inferring the delivery location for an address, DLInfMA-LC_{addr} would mistakenly penalize locations frequently visited when delivering for other addresses of the same building.

Clustering Distance Selection. For candidate pool construction, we vary the clustering distance D from 20m to 60m to select the one with the best performance. The MAEs of both datasets with different D are reported in Figure 10(a). Though with the increase of D, the number of location candidates for each address to choose from would decrease, we find MAE first decreases and then increases. The reason is that, when D is smaller, it is more difficult to select the actual delivery location among massive candidates; when D is larger, the precision of location candidates is degraded, which increases MAE. Therefore, we set $D = 40$m at the turning point.



(a) Distance Selection. (b) Different #Deliveries.

Fig. 10. Distance Selection and Different # of Deliveries.

Different Number of Deliveries. We divide addresses into three equal-frequency groups based on the number of deliveries that involve the address, and report MAE of DLInfMA as well as those of the four representative baselines, i.e., GeoCloud, MaxTC-ILC, GeoRank, UNet-based on DowBJ in Figure 10(b). When the number of deliveries increases, GeoCloud, UNet-based and MaxTC-ILC become better. Given more delivery records, GeoCloud can better identify mis-annotated locations, and UNet-based as well as MaxTC-ILC can better select locations with more distinguishable features. The performance of GeoRank and DLInfMA for addresses with few deliveries is not severely degraded. This is because the distance also matters, yet it is not considered by the other three baselines. MAE of GeoRank first decreases, then slightly increases. The reason might be lacking features whose contribution to the delivery location inference is highly related to the number of deliveries. MAE of DLInfMA slightly decreases with the number of deliveries, and consistently outperforms baselines. LocMatcher in DLInfMA uses the address feature as a context vector to calculate the attention weights of different location embeddings, which semantically decides the importance of different features.

D. Effectiveness Evaluation on Synthetic Datasets

In order to show the robustness of DLInfMA against couriers' behaviors of confirmation with delays, we further perform evaluation on synthetic datasets by injecting delays of different extents to DowBJ and SubBJ. We first introduce how we inject the delays to existing datasets and then report the results.

Synthetic Dataset Generation. We observed from the data that the phenomena of confirmation with delays are usually caused by couriers' batch confirmation operations. That is, after a batch of parcels are delivered, which could be at different delivery locations, the courier performs the confirmation for them all at once for confirmation convenience. And we found such operations are usually conducted when couriers are staying at some locations.

Inspired by this, we devise the following method to inject different extents of delays to waybills: (1) Based on the ground-truth delivery location, we first find the actual delivery time of each waybill based on its nearby stay points. (2) Then, for each delivery trip, we sequentially divide stay points in the trip into several equal-sized groups. In each group, the time of

TABLE III
DIFFERENT PROBABILITIES OF CONFIRMATION WITH DELAYS.

Methods	Different Probabilities of Delays on DowBJ									Different Probabilities of Delays on SubBJ								
	0.2			0.6			1.0			0.2			0.6			1.0		
	MAE	P95	β_{50}	MAE	P95	β_{50}	MAE	P95	β_{50}	MAE	P95	β_{50}	MAE	P95	β_{50}	MAE	P95	β_{50}
Annotation	114.2	431.3	44.5	285.2	742.5	11.7	454.2	1050.9	2.2	175.3	693.0	45.3	454.3	1448.8	12.6	742.1	2269.5	2.1
MaxTC	172.3	913.8	51.1	188.4	1028.6	48.9	210.6	1065.6	44.1	291.7	1311.0	39.4	306.0	1492.6	38.0	337.9	1883.5	33.2
Geocoding	146.4	476.9	55.7	146.4	476.9	55.7	146.4	476.9	55.7	156.0	477.4	46.7	156.0	477.4	46.7	156.0	477.4	46.7
MinDist	72.2	215.6	59.8	75.6	229.8	56.8	77.6	236.3	55.1	114.3	381.5	52.3	119.0	408.3	50.1	121.9	424.9	48.6
GeoCloud	62.1	171.6	69.2	198.6	1005.0	30.2	417.4	1311.3	6.7	85.5	233.8	68.6	373.0	2126.6	28.2	800.7	2844.7	4.1
MaxTC-ILC	63.4	268.7	69.7	67.2	268.6	65.6	74.2	274.3	59.5	116.9	371.9	53.0	126.8	424.6	49.2	149.1	493.5	39.0
GeoRank	43.1	128.8	80.8	85.8	307.1	67.0	193.9	955.1	28.7	65.2	229.6	74.3	180.4	644.6	53.7	436.6	2541.3	12.9
UNet-based	50.6	180.6	83.7	350.0	1382.7	57.5	812.0	1621.6	3.4	89.4	327.1	72.7	543.0	2103.9	47.3	1089.5	2863.4	3.8
DLInfMA	**31.0**	**125.5**	**84.6**	**34.2**	**127.7**	**81.0**	**40.2**	**140.1**	**74.6**	**45.3**	**225.1**	**79.1**	**56.1**	**238.0**	**72.9**	**75.1**	**280.7**	**61.2**

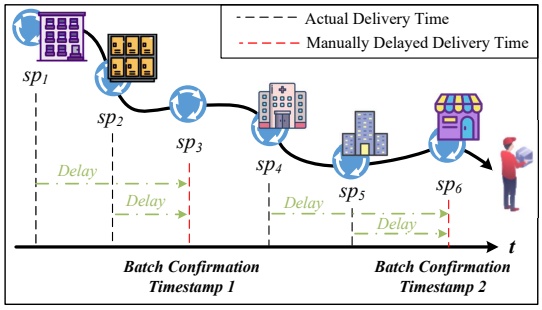


Fig. 11. Illustration of Generating Synthetic Datasets.

the last stay point serves as a time to do the batch confirmation. Each waybill that is actually delivered before that time and after the previous batch confirmation time has a probability p_d to be deliberately delayed, i.e., setting its delivery time to that batch confirmation time. In this way, the delivery times would usually involve some delays. For example, as shown in Figure 11, the delivery trip contains 6 stay points, and we assume waybills are confirmed with 2 batches. Then the times of sp_3 and sp_6 are treated as the time to perform the batch confirmation. The delivery time of waybills delivered at sp_1 and sp_2 are delayed to the time of sp_3 with probability p_d. The delivery time of waybills delivered in sp_4 and sp_5 are delayed to the time of sp_6 in the same way.

Based on the data analysis of the real-world datasets, couriers usually has 2 times to perform the batch confirmation, and p_d is around 0.3. Therefore, we keep the same number of batch confirmation times, and vary p_d in $\{0.2, 0.6, 1.0\}$ to evaluate the proposed method under different circumstances (i.e., datasets involve slight, moderate, or significant delays).

Results. As shown in Table III, all methods except Geocoding became worse with the increase of p_d, since its performance is purely based on the address input. Four annotation-based baselines (Annotation, GeoCloud, GeoRank and UNet-based) have a better performance when p_d is small, which shows the advantages of using the delivery data to infer the delivery locations. However, with the increase of p_d, the performances of annotation-based methods degrade significantly, which are even worse than Geocoding ultimately. It shows those methods only work when delay confirmations seldom occur. Otherwise, the annotated locations can be arbitrarily far away from the actual delivery locations and the probability that the delivery location is covered by the annotated locations would become lower. MinDist, MaxTC and MaxTC-ILC are less sensitive to the decrease of d_p, which shows that the features we employ are less affected by the delayed confirmations. DLInfMA consistently outperforms all baselines under different delay probabilities over three metrics on two datasets, which shows its robustness against the couriers' annotation behaviors.

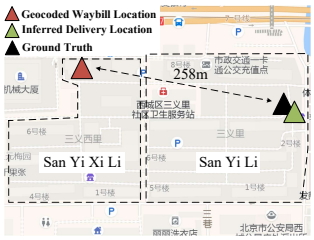
E. Case Studies

We now give three case studies to further demonstrate the effectiveness of DLInfMA and illustrate why Geocoding is not sufficient for the delivery location inference in detail.

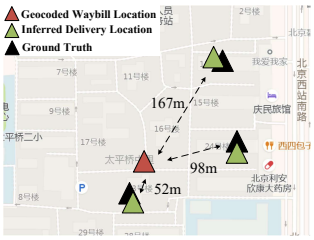
Firstly, Geocoding suffers from issues of similar location names and unclear address input, which make it parse the address to a wrong location. Figure 12(a) shows a case that the actual delivery location of the address is 258m away from the Geocoded result. It parsed the address to a nearby residential area (from "San Yi Li" to "San Yi Xi Li"). Those two names are quite similar, which make the Geocoding system confused. Since the building number of the address also appears in "San Yi Xi Li", it returns the location comfortably, which causes a huge error. On the contrary, DLInfMA does not explicitly use the plaintext address to make the prediction. Though it considers the distance to the Geocoded location, it also takes trip coverage, location commonalities and many other features derived from trajectories, and makes decisions wisely.

Secondly, if the underlying POI database (which is manually collected) of Geocoding is not fine-grained enough, Geocoding is only able to produce one location for multiple addresses in an area. Figure 12(b) shows three addresses in different buildings whose Geocoded locations are the same one and fall at the center of the residential area. It is not close to any of the actual delivery locations. On the contrary, DLInfMA infers the delivery location based on candidates, which are generated based on couriers' stay points. Therefore, the potential locations we can use are much more fine-grained.

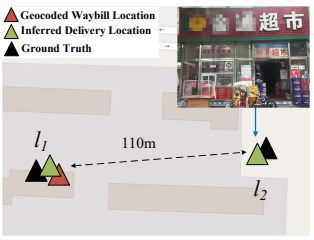
Thirdly, Geocoding is not able to capture diverse customer preferences of receiving parcels, which is not uncommon as we have already illustrated in Figure 9(a). We show two addresses, which share the same building in Figure 12(c). As expected, they have the same Geocoded location. However, they have different delivery locations: l_1 and l_2, which are



(a) Wrong Parsing.



(b) Coarse Parsing.



(c) Different Preferences.

Fig. 12. Comparing with Geocoding Solutions.

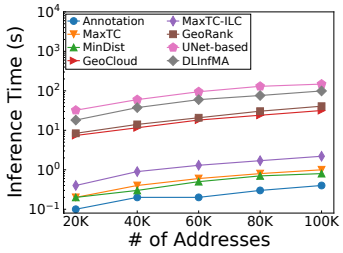


Fig. 13. Inference Efficiency.

110m away. l_1 is closer to the Geocoded location and within a real estate. However, l_2 is a bit far away and outside the real estate. We physically visited l_2 and found a convenience store there. Furthermore, we knew from the store staff that, for customers living nearby, it provides a service of receiving parcels charging only 1RMB. This case shows the address-based delivery location inference is able to give preference-aware inference results.

F. Scalability Evaluation

In this subsection, we report the efficiency results of DLInfMA. The time consumption of using DLInfMA to infer the delivery location mainly contains following four parts:

(1) Stay Point Extraction. This step is implemented with trajectory-level parallelization. In this step, given the trajectory data with 66.1M points, it takes 7min.

(2) Candidate Pool Construction. In this step, the location candidates are generated in a bi-weekly manner and for each station and each time period in parallel. Then a station-level parallel processing is used to merge location candidates generated in the past. For the data of 11 stations over 20 months, it takes 1min in total.

(3) Model Training. We also report the training time of DLInfMA and two supervised baselines, i.e., GeoRank and UNet-based on DowBJ. GeoRank takes 0.2 minutes to complete the training, which is the fastest given the simpler model and less location candidate pairs derived from annotated locations. Both UNet-based and DLInfMA are trained in the mini-batch fashion, which requires a longer time. As for UNet-based, it takes 27.0 minutes, since each sample has 9×9 candidates, and UNet needs to calculate feature maps at different resolutions, which is more complex than others. DLInfMA takes 13.6 minutes until it converges. DLInfMA is faster than UNet-based, because the computation time of DLInfMA is dynamic for each address, which is related to number of candidates of each sample, while it is fixed for UNet-based.

(4) Inference. The inference time of DLInfMA as well as baselines is shown in Figure 13, which grows linearly with the increase of the # of addresses. The heuristics-based algorithms are the most efficient. GeoRank is slightly slower than GeoCloud, since it needs to perform quadratic comparisons among annotated locations. DLInfMA is also faster than UNet-based during the inference. As we explained previously, its time complexity is related to the number of location candidates for an address. In our datasets, for 95% addresses, it is less than 78. However, UNet-based always needs to score 9×9 locations for each sample. DLInfMA can infer 1,000 addresses per second, which is practical for real world usage because the inference is usually performed offline, and the inference result is served as a basic geospatial data source.

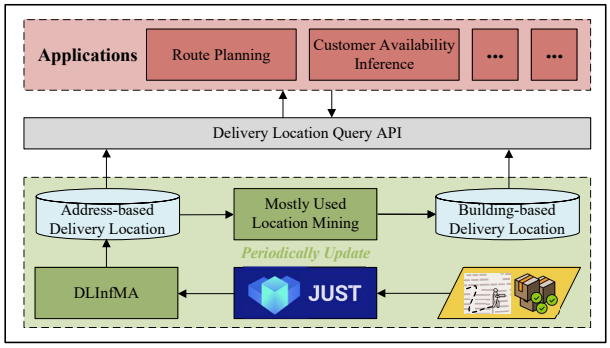


Fig. 14. System Deployment.

VI. Deployment and Applications

A delivery location inference system based on DLInfMA is deployed and used internally in JD Logistics.

A. Deployment

The system is shown in Figure 14. Given the low update frequency of delivery location, the system does not require the real-time inference. Instead, we design a delivery location query API based on inferred results, which is able to answer online requests by downstream applications. To obtain the inferred results, we pre-process and store the raw couriers' trajectories and waybills in our self-developed spatio-temporal distributed platform, JD Urban Spatio-Temporal Data Engine (JUST) [28] for large-scale processing and querying. DLInfMA retrieves the pre-processed data from JUST, and performs the delivery location inference. Results are stored to a key-value database storing the mapping from each address to its inferred delivery location. In order to make the delivery knowledge more general (e.g., it can handle real-time cases, where the addresses might have never appeared in history), we also obtain the mostly used delivery location for each building based on addresses affiliated to it. For the online delivery

(a) Route Planning. (b) Availability Inference.

Fig. 15. Application Scenarios.

location query, we first search for the results in address-based database. If it fails, we search in the building-based one. If it fails again, we directly return its Geocoded location. The inference results are periodically updated since the inference would be more accurate with more historical deliveries.

Based on the system, we present following two applications:

B. Application 1: Route Planning

The route planning is a useful function for new couriers. Previously, routes are planned using TSP [1] based on Geocoded locations, which are unsatisfactory given the inaccurate Geocoded locations. Based on DLInfMA, the inferred delivery locations are employed, and an interface of this process is shown in Figure 15(a).

C. Application 2: Customer Availability Inference

Knowing the availability of customers not only promotes the delivery success rate, but also improves customer experience. Based on historical deliveries, we can model the delivery feasibility of an address considering time of the day, day of the week and meteorology. However, the availability labels are obtained based on the delivery time recorded manually previously, which might be delayed. After finding delivery locations, we can find the actual delivery time using the stay points nearby the actual delivery location. Figure 15(b) shows the time windows of addresses where the availability probabilities are above a threshold.

VII. Related Work

Volunteered Geographic Information. Volunteered geographic information (VGI) [29] is the alternative geospatial data source provided by volunteers. Apart from traditional VGI platforms (e.g., OpenStreetMap), there are also some indirect ways contributing to VGI. A typical way is the spatial footprint [30] generated by people on social media platforms. It can be used to investigate the boundaries of vague place concepts [31], and enrich gazetteers with new place entries [32]. Recently, the annotated locations in the logistics also attracts the attention to contribute VGI [5], [6], [19], [33], such as finding regions of interest (ROIs) [33], improving Geocoding [19], correcting wrong locations claimed by merchants [34] and inferring the delivery locations [5], [6], [20]. [5] treats the spatial centroid of annotated locations sharing the same address as its delivery location. [6] treats all annotated locations sharing the same address as candidates, and introduces a pairwise ranking model to select the delivery location. Such annotation data used by existing works only contains one annotated location for each parcel's delivery. As explained in Section I, these methods would fail in cases of mis-annotations. A recent work [20] tries to mitigate the issue by incorporating customers' location data, but they introduce additional data source from customer-side locations, which may not always be available. Different from them, we leverage couriers' trajectories, which contain richer information than annotated locations and are totally generated from couriers' daily operations, to infer the actual delivery locations.

Trajectory Data Mining. Trajectory data mining [8] aims to discover various knowledge from massive trajectory data: map enhancement [35], [36], interesting place discovery [9], [11], [12], [37], billboard selection [38], risky zone detection [39], crow flow inference [40], etc. In this work, we discover delivery locations for each address based on couriers' trajectories, even the delivery time is recorded with significant delays.

Location Selection. Our work is essentially to select a location from multiple candidates given a specific address. The location selection task is usually formulated as a ranking problem [14], [15], [41]. Zhang et al. [41] find a location with the highest geo-tf-idf score among candidates, which characterizes the relevance of tags at the location and the geographical region. Other methods attempt to learn the ranking function, which can fuse various factors for selecting commercial sites [14], [15]. Different from these studies, we use an attention-based model LocMatcher to compute scores of location candidates jointly and make the selection.

VIII. Conclusion

In this paper, we propose DLInfMA, a method to discover delivery locations based on couriers' trajectories, which considers that couriers may record delivery time with delays. DLInfMA first discovers location candidates by clustering stay points in trajectories, and then infers the delivery location of an address as the location candidate selected by an attention-based model, LocMatcher. DLInfMA outperforms baselines in all metrics. Even for the most competitive metric (accuracy within 50m), it outperforms the best baselines by 4%-10%. It is able to infer 1K addresses/s. Finally, a deployed system based on it as well as two applications are introduced. The deployment shows DLInfMA can facilitate many downstream tasks.

Acknowledgment

This research is supported by the National Key R&D Program of China (2019YFB2103201), the Ministry of Education, Singapore, under its Academic Research Fund (Tier 2 Award MOE-T2EP20220-0011 and Tier 1 Award (RG77/21)), the NSFC Grant (61976168, 62076191), and the Chinese Government Scholarship (202006960044). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of the Ministry of Education, Singapore or other funding sources.

REFERENCES

[1] M. Dorigo and L. M. Gambardella, "Ant colonies for the travelling salesman problem," *biosystems*, vol. 43, no. 2, pp. 73–81, 1997.

[2] H. Wen, Y. Lin, F. Wu, H. Wan, S. Guo, L. Wu, C. Song, and Y. Xu, "Package pick-up route prediction via modeling couriers' spatial-temporal behaviors," in *ICDE*. IEEE, 2021, pp. 2141–2146.

[3] F. Wu and L. Wu, "Deepeta: A spatial-temporal sequential neural network model for estimating time of arrival in package delivery system," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 774–781.

[4] Y. Wang, D. Zhang, Q. Liu, F. Shen, and L. H. Lee, "Towards enhancing the last-mile delivery: An effective crowd-tasking model with scalable solutions," *Transportation Research Part E: Logistics and Transportation Review*, vol. 93, pp. 279–293, 2016.

[5] S. Ruan, Z. Xiong, C. Long, Y. Chen, J. Bao, T. He, R. Li, S. Wu, Z. Jiang, and Y. Zheng, "Doing in one go: delivery time inference based on couriers' trajectories," in *KDD*, 2020.

[6] G. Forman, "Getting your package to the right place: supervised machine learning for geolocation," in *ECML-PKDD*, 2021, pp. 1–16.

[7] Q. Li, Y. Zheng, X. Xie, Y. Chen, W. Liu, and W.-Y. Ma, "Mining user similarity based on location history," in *SIGSPATIAL*. ACM, 2008, p. 34.

[8] Y. Zheng, "Trajectory data mining: an overview," *TIST*, vol. 6, no. 3, pp. 1–41, 2015.

[9] D. Ashbrook and T. Starner, "Learning significant locations and predicting user movement with gps," in *Proceedings. Sixth International Symposium on Wearable Computers,*. IEEE, 2002, pp. 101–108.

[10] R. Assam and T. Seidl, "Context-based location clustering and prediction using conditional random fields," in *MUM*, 2014, pp. 1–10.

[11] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from gps trajectories," in *WWW*, 2009, pp. 791–800.

[12] V. W. Zheng, Y. Zheng, X. Xie, and Q. Yang, "Collaborative location and activity recommendations with gps history data," in *WWW*, 2010, pp. 1029–1038.

[13] J. H. Ward Jr, "Hierarchical grouping to optimize an objective function," *Journal of the American statistical association*, vol. 58, no. 301, pp. 236–244, 1963.

[14] D. Karamshuk, A. Noulas, S. Scellato, V. Nicosia, and C. Mascolo, "Geo-spotting: mining online location-based services for optimal retail store placement," in *KDD*, 2013, pp. 793–801.

[15] H. Niu, J. Liu, Y. Fu, Y. Liu, and B. Lang, "Exploiting human mobility patterns for gas station site selection," in *DASFAA*. Springer, 2016, pp. 242–257.

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017, pp. 5998–6008.

[18] O. Vinyals, M. Fortunato, and N. Jaitly, "Pointer networks," in *NIPS*. Curran Associates, Inc., 2015, pp. 2692–2700.

[19] V. Srivastava, P. Tejaswin, L. Dhakad, M. Kumar, and A. Dani, "A geocoding framework powered by delivery data," in *SIGSPATIAL*, 2020, pp. 568–577.

[20] Y. Song, J. Li, L. Chen, S. Chen, R. He, and Z. Sun, "A semantic segmentation based poi coordinates generating framework for on-demand food delivery service," in *Proceedings of the 29th International Conference on Advances in Geographic Information Systems*, 2021, pp. 379–388.

[21] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[22] R. Baeza-Yates, B. Ribeiro-Neto *et al.*, *Modern information retrieval*. ACM press New York, 1999, vol. 463.

[23] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[24] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[25] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT, 2016.

[26] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, "Learning to rank using gradient descent," in *Proceedings of the 22nd international conference on Machine learning*, 2005, pp. 89–96.

[27] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[28] R. Li, H. He, R. Wang, Y. Huang, J. Liu, S. Ruan, T. He, J. Bao, and Y. Zheng, "Just: Jd urban spatio-temporal data engine," in *ICDE*. IEEE, 2020, pp. 1558–1569.

[29] M. F. Goodchild, "Citizens as sensors: the world of volunteered geography," *GeoJournal*, vol. 69, no. 4, pp. 211–221, 2007.

[30] Z. Cheng, J. Caverlee, K. Lee, and D. Sui, "Exploring millions of footprints in location sharing services," in *ICWSM*, vol. 5, no. 1, 2011.

[31] L. Li and M. F. Goodchild, "Constructing places from spatial footprints," in *Proceedings of the 1st ACM SIGSPATIAL international workshop on crowdsourced and volunteered geographic information*, 2012, pp. 15–21.

[32] S. Gao, L. Li, W. Li, K. Janowicz, and Y. Zhang, "Constructing gazetteers from volunteered big geo-data based on hadoop," *Computers, Environment and Urban Systems*, vol. 61, pp. 172–186, 2017.

[33] M. Dahiya, D. Samatia, and K. Rustogi, "Learning locality maps from noisy geospatial labels," in *SAC*, 2020, pp. 601–608.

[34] D. Jiang, Y. Ding, H. Zhang, Y. Liu, T. He, Y. Yang, and D. Zhang, "Alwaes: an automatic outdoor location-aware correction system for online delivery platforms," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 5, no. 3, pp. 1–24, 2021.

[35] S. He, F. Bastani, S. Abbar, M. Alizadeh, H. Balakrishnan, S. Chawla, and S. Madden, "Roadrunner: improving the precision of road network inference from gps trajectories," in *SIGSPATIAL*, 2018, pp. 3–12.

[36] L. Zhao, J. Mao, M. Pu, G. Liu, C. Jin, W. Qian, A. Zhou, X. Wen, R. Hu, and H. Chai, "Automatic calibration of road intersection topology using trajectories," in *ICDE*. IEEE, 2020, pp. 1633–1644.

[37] C. Zhou, D. Frankowski, P. Ludford, S. Shekhar, and L. Terveen, "Discovering personal gazetteers: an interactive clustering approach," in *Proceedings of the 12th annual ACM international workshop on Geographic information systems*, 2004, pp. 266–273.

[38] P. Zhang, Z. Bao, Y. Li, G. Li, Y. Zhang, and Z. Peng, "Trajectory-driven influential billboard placement," in *KDD*, 2018, pp. 2748–2757.

[39] J. Wang, C. Chen, J. Wu, and Z. Xiong, "No longer sleeping with a bomb: a duet system for protecting urban safety from dangerous goods," in *KDD*, 2017, pp. 1673–1681.

[40] Y. Liang, K. Ouyang, L. Jing, S. Ruan, Y. Liu, J. Zhang, D. S. Rosenblum, and Y. Zheng, "Urbanfm: Inferring fine-grained urban flows," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 3132–3142.

[41] D. Zhang, B. C. Ooi, and A. K. Tung, "Locating mapped resources in web 2.0," in *ICDE*. IEEE, 2010, pp. 521–532.