Taxi-Passenger-Demand Modeling Based on Big Data from a Roving Sensor Network

Desheng Zhang, *Student Member, IEEE,* Tian He, *Senior Member, IEEE,* Shan Lin, *Member, IEEE,* Sirajum Munir, *Member, IEEE,* and John A. Stankovic, *Life Fellow, IEEE*

Abstract—Investigating passenger demand is essential for the taxicab business. Existing solutions are typically based on offline data collected by manual investigations, which are often dated and inaccurate for real-time analysis. To address this issue, we propose Dmodel, employing roving taxicabs as *real-time mobile sensors* to (i) infer passenger arriving moments by interactions of vacant taxicabs, and then (ii) infer passenger demand by customized online training with both historical and real-time data. Dmodel utilizes a novel parameter called pickup pattern based on an entropy of pickup events (accounts for various real-world logical information, *e.g.*, bad weather) to reduce the size of big historical taxicab data to be processed. We evaluate Dmodel with a real-world 450 GB dataset of $14,000$ taxicabs for a half year, and results show that compared to the ground truth, Dmodel achieves 83% accuracy and outperforms a statistical model by 42%. We further present an application where Dmodel is used to dispatch vacant taxicabs to achieve an equilibrium between passenger demand and taxicab supply across urban regions.

Index Terms—Taxicab System, Demand Modeling, Big Transportation Data.

1 Introduction

Understanding and predicting passenger demand are essential for the taxicab business [1]. With accurate knowledge of demand, taxicab companies can schedule their fleet and dispatch individual taxicabs to minimize idle driving time and maximize profits. Historically, such passenger demand has been investigated by manual procedures (*e.g.*, creating surveys or sampling [2]). However, these manual studies are often dated, incomplete and difficult to use in real time. In particular, though fairly stable during long-time periods (*e.g.*, one day), passenger demand experiences significant irregular spatio-temporal dynamics during short-time periods (*e.g.*, one hour) due to various real-world phenomena, *e.g.*, bad weather, accidents or special events. As a result, both long-term historical and short-term real-time demand knowledge shall be utilized to capture such dynamics. However, we face a challenge to create an accurate demand model by combining both historical and real-time demand, because historical demand is typically in a limited scale and real-time demand is difficult to be obtained in real time [2].

- *D. Zhang is with the Department of Computer Science and Engineering, University of Minnesota, zhang@cs.umn.edu*
- *T. He is with the Department of Computer Science and Engineering, University of Minnesota, tianhe@cs.umn.edu*
- *S. Lin is with the Department of Electrical and Computer Engineering, Stony Brook University, shan.x.lin@stonybrook.edu*
- *S. Munir is with the Bosch Research and Technology Center, sirajum.munir@us.bosch.com. This work was performed while the author was at the University of Virginia.*
- *J. A. Stankovic is with the Department of Computer Science, University of Virginia, stankovic@cs.virginia.edu*

In this work, we provide a two-part solution based on recent infrastructure updates of taxicab networks. First, we data mine a large dataset of historical information regarding passenger demand and taxicabs' trips. This results in the basis of our method, from which we identify what aspects should be used to infer specific real-time demand. In this work, the historical GPS dataset used is from 14,000 taxicabs for 6 months (450 GB) in a Chinese city, Shenzhen. While this historical model is more accurate than surveys and sampling, it cannot handle many real-time issues and thus has major limitations if used alone.

Second, to address the short-term, real-time dynamics, we consider thousands of roving taxicabs as real-time mobile sensors and collect current information from them. This is possible because taxicabs in dense urban areas are equipped with GPS as location sensors and fare meters as passenger sensors, and thus their locations and occupancy status can be periodically uploaded to a dispatch center. These frontend taxicabs and a backend dispatch center form a real-time "roving sensor network". The streaming data used are from a data feed in Shenzhen taxicab network with an average rate of 450 status records per second.

Admittedly, several systems have proposed to use taxicab GPS traces to infer passenger demand [3] [4] [5], but they typically have two simplifying assumptions: (i) they assume that previous demand is given by picked-up passengers, but overlook waiting passengers who did not get picked up; and (ii) they assume that current demand can be inferred by long-term historical demand, but overlook

the fact that passenger demand is highly dynamic. For example, after a major concert, due to the high demand, there are few picked up passengers yet numerous waiting passengers, and the average historical demand cannot accurately indicate the suddenly-increased demand due to the concert.

In this paper, to improve upon these two simplifying assumptions, we propose Dmodel, which observes hidden contexts to infer passenger demand based on both historical and real-time taxicab data. The contributions of this paper are as follows.

- We identify passenger demand with a combined offline data analysis and a real-time roving sensor network, where taxicabs detect passenger counts and arriving moments. It is important to note that passenger arriving moments are, in general, unknown. But a major contribution of Dmodel is how the roving sensor network infers them by utilizing taxicabs' interactions.
- We present a novel parameter, called pickup pattern, to quantify taxicab operating similarity among different daily data in a big taxicab dataset, *e.g.*, 900 GB per year in Shenzhen. Note that naively using more data from such a big dataset results in not only unnecessary big workload but also inaccurate inferences. Thus, the key novelty of Dmodel is to *utilize the real-time pickup pattern to select customized yet compact training data to increase inference accuracy*. This pickup pattern implicitly accounts for spatio-temporal dynamics caused by real-world phenomena, *e.g.*, bad weather.
- We test Dmodel on a 450 GB dataset created by 6 months of status records from 14,000 taxicabs in Shenzhen. The evaluations show that compared to the ground truth, Dmodel achieves 83% inference accuracy of demand in terms of the passenger counts, and outperforms a statistical model by 42%. We will share such a valuable dataset for benefits of the big data research community in the preprint version.
- We show Dmodel's practical value in a real world application where demand inferred by Dmodel is used to dispatch vacant taxicabs across city regions to achieve a better equilibrium between passenger demand and taxicab supply, which potentially leads to shorter idle driving times and higher profits for drivers as well as shorter waiting times for passengers.

The rest of the paper is organized as follows. Section 2 gives our motivations. Section 3 shows a framework of modeling. Section 4 presents a roving sensor network. Section 5 describes our model. Section 6 evaluates our model by a real-world 450 GB dataset. Section 7 presents the application based on our modeling, followed by the related work and the conclusion in Sections 8 and 9.

2 MOTIVATIONS

In this section, based on empirical data (introduced in Section 4) from a real-world taxicab network with 14,000 taxicabs in Shenzhen, we present our motivations to improve upon two legacy assumptions for passenger demand analyses.

2.1 Assumption on Previous Demand

Legacy Assumption One: Given a previous time slot, the passenger demand (*i.e.*, the total count of all passengers requiring taxicab services) equals to the number of picked up passengers (*i.e.*, pickup counts) [3] [4] [5].

In this work, we argue that though all passengers get picked up eventually, for a previous slot the passenger demand should include not only picked-up passengers but also waiting passengers who had arrived but did not get picked up. Figure 1 gives the difference between pickup counts and total passenger counts, *i.e.*, a pickup passenger count plus a waiting passenger count, for the entire Shenzhen area in 5 minute slots (how to obtain these two counts are given in Section 5.1).

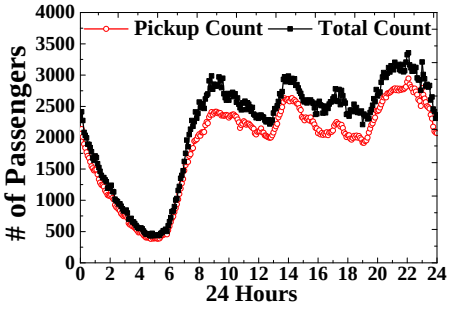


Fig 1: Pickup and Total Counts

We find that the pickup and total passenger counts are usually different, especially in the slots of the rush hour. Thus, this assumption overlooking waiting passengers leads to an inaccurate analysis.

The key reason for this assumption is that arriving moments for picked-up passengers cannot be obtained by existing infrastructures. To address this issue, we present a novel method based on the interactions of vacant taxicabs to infer arriving moments, which are used to incorporate waiting passengers for accurate passenger demand analyses. The details are given in Section 4.4.2.

2.2 Assumption on Current Demand

Legacy Assumption Two: Given a current time slot, the passenger demand can be inferred by the previous passenger demand for the same slot [5].

For this assumption, we argue that for the same area and slot, the passenger demand experiences irregular temporal dynamics in different days due to

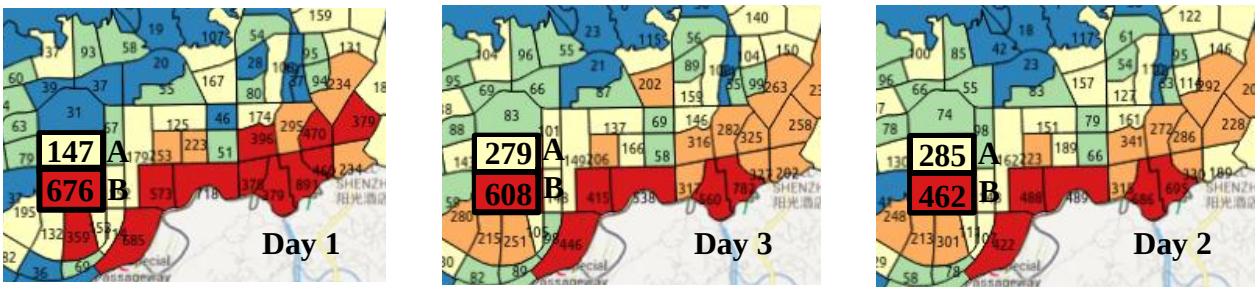


Fig 2: Demand Dynamics for the Same Hourly Slot in Three Different Days

various real-world factors, and cannot be accurately inferred without considering more contexts. Figure 2 gives the passenger demand for the same hourly slot in three different weekdays, which is shown by total passenger counts in different administrative regions of Shenzhen. Suppose we want infer the passenger demand of Region A and B in Day 3 given in the middle figure, and the historical demand for the same regions and the same slot in Day 1 and Day 2 is given by the left and right figures. If we infer Region A's demand in Day 3 based on the previous Region A's demand in Day 1, we only have $\frac{279-|147-279|}{279} \approx 53\%$ accuracy; similarly, if we infer Region B's demand in Day 3 based on the Region B's demand in Day 2, we only have $\frac{608-|462-608|}{608} \approx 76\%$ accuracy. Thus, the assumption two leads to an inaccurate inference.

The key reason for this assumption is lacking an effective parameter to select related historical data as training data for the inference. Thus, in this paper, to improve this assumption, we propose a novel parameter called *pickup pattern* to select a customized training dataset for a particular demand inference. For example, based on the pickup pattern, if we find that Region A's demand in Day 2 is more related to Region A's demand in Day 3, then we infer Region A's demand in Day 3 based on Region A's demand in Day 2. As a result, we improve the accuracy for Region A in Day 3 from $\frac{279-|147-279|}{279} \approx 53\%$ to $\frac{279-|285-279|}{279} \approx 98\%$. Thus, finding highly related data for the inference increases accuracy, and also reduces the workload to process big taxicab data. The details are given in Section 5.2.1.

2.3 Summary

The above two assumptions are the key reasons for inaccuracy of existing modeling methods for taxicab passenger demand. In particular, an accurate inference on previous demand is the key foundation for an accurate inference on current demand. To improve upon these two assumptions, we present a modeling method with a framework as follows.

3 Framework

In this section, we present an overview of our modeling with three components, *i.e.*, Roving Sensor Network, Model Generation, Model Utilization as in Figure 3. These three components span the whole taxicab-data-processing chain.

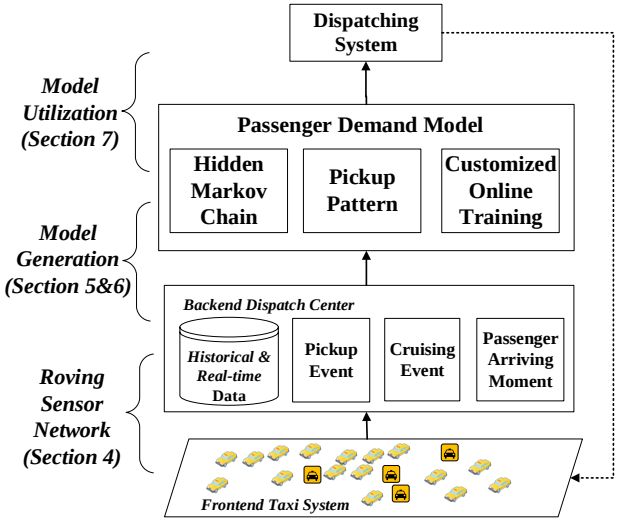


Fig 3: Framework of Dmodel

Roving Sensor Network. Recently, taxicab infrastructures in large cities are upgraded with onboard GPS and communication devices as well as dispatch centers [6]. From a broad perspective, we consider frontend taxicabs and a backend dispatch center as a roving sensor network to infer passenger demand. We utilize both historical and real-time data from such a network to detect two kinds of events, *i.e.*, pickup events and cruising events, to infer passenger arriving moments for our later model generation component. The details are given in Section 4.

Model Generation. Based on the massive historical and real-time data, we generate a passenger demand model called Dmodel. Dmodel utilizes a hidden Markov chain to model passenger counts based on real-time passenger arrival and pickups. During the modeling process, we propose a system

parameter called pickup pattern to obtain highly related historical data for customized online training to indicate a relationship between pickup passenger counts and total passenger counts. Finally, Dmodel outputs total passenger counts at road-segment levels during a fine-grained time interval. The details of designs and evaluations for Dmodel are given in Section 5 and Section 6, respectively.

Model Utilization. According to passenger demand predicted by Dmodel, we propose a real-world application, *i.e.*, a dispatching system to redistribute taxicabs among different urban regions to achieve an equilibrium between passenger demand and transit supply at region levels, which finally provides feedback to the frontend taxicab system and closes the control loop. Such an application has the potential to reduce passenger waiting times and driver cruising miles, which is introduced and evaluated in Section 7.

With a highlight on passenger demand inferences, our modeling builds an architectural bridge between generic taxicab infrastructures and real-world knowledge output tailored by a specific application.

4 Roving Sensor Network

Recently, taxicab infrastructures in large cities have been updated with onboard GPS and communication devices as well as a dispatch center to receive GPS data from taxicabs in real time to monitor status of taxicab networks. Built on such an infrastructure, a *roving sensor network* consists of (i) numerous *roving taxicabs* in the frontend as mobile sensors to detect passengers, and (ii) a *dispatch center* in the backend to receive sensing data (*i.e.*, taxicab status) from taxicab sensors to analyze demand. In this work, we utilize a taxicab network in Shenzhen with 14,453 taxicabs and a ridership of more than 200 million per year as an example to study such a roving sensor network. In a roving sensor network, taxicabs record their physical status, *e.g.*, current location and speed, with GPS devices; taxicabs also record their logical status, *i.e.*, with passengers or not, with fare meters; both their physical and logical status is periodically (30 seconds on average) uploaded to dispatch centers with onboard communication devices, in terms of sensing records. A sensing record mainly consists of the following parameters: Plate Number; Date and Time; GPS Coordinates; Status Bit (1 or 0: indicating with passengers or not when this record is uploaded).

Figure 4 gives a dataset about such sensing records from Shenzhen in China [7] ($17,150$ people per square KM). This half-year dataset contains almost 4 billion sensing records with a size more than 450 GB.

4.1 Data Management

We briefly introduce our management issues about big data in the roving sensor network.

Sensing Dataset Summary	
Collection Period	6 Months
Collection Date	01/01-06/30
# of Taxicabs	14,453
# of Pickup Events	98,472,628
# of Sensing Records	3.9 Billion
Data Size	450 GB

Fig 4: Dataset Summary

4.1.1 Data Storage

We establish a secure and reliable transmission mechanism, which feeds our server taxicab GPS data collected by Shenzhen Transport Committee by a wired connection without impacting the original data source. Such a big amount of sensing data requires significant efforts for efficient storage and management. In this project, we store the data by utilizing a 34 TB Hadoop Distributed File System (HDFS) on a cluster consisting of 11 nodes, each of which is equipped with 32 cores and 32 GB RAM. For daily management, we typically use the MapReduce-based tools, *e.g.*, Pig and Hive.

4.1.2 Data Cleaning

Due to the extremely-large size of our data, we find three main kinds of errant data. (i) Missing Data: *e.g.*, a taxicab's sensing records were not uploaded within a given time period. Such missing data are detected by monitoring temporal consistence of incoming data for every taxicab. (ii) Duplicated Data: *e.g.*, the sensing dataset shows two identical records for the same taxicab. Such duplicated data are detected by comparing timestamps of every record belonging to the same taxicab. (iii) Data with Logical Errors: *e.g.*, GPS coordinates show that a taxicab is off the road. Such data with logical errors are detected later when we analyze the data. In particular, we utilize a digital map of Shenzhen to verify if a GPS location is plausible or not. This is performed by checking the previous locations and durations between timestamps of two records. The above errors may result from various reasons, *e.g.*, hardware malfunctions, software issues, and communication. To address the above errors, for all incoming data, we first filter out duplicated records and records with missing or errant attributes. Then we correct obvious numerical errors by various known contexts. We next store the data by dates and categories. Finally we compare temporal consistence of the data to detect missing records. Admittedly, the missing or filtered-out data (which accounted for 12% of the total data) may impact the performance of our later modeling, but we believe we are still able to provide insightful analyses given the long time period of the data we have collected.

4.2 Constrained Sensing Capability

Though our roving sensors produce an extremely-large sensing dataset, they have severely-constrained sensing capabilities. In regular sensor networks, the primary objective of sensors is to detect events, and thus sensors typically have a full sensing capability, *i.e.*, sensors can be temporally and spatially controlled to detect events. However, a roving sensor network consists of taxicabs whose primary objective is to deliver passengers, instead of to detect passengers, resulting in a constrained sensing capability. Such capabilities are shown by a sensing cycle of a taxicab in Figure 5 where a taxicab sensor functions in two alternating phases.

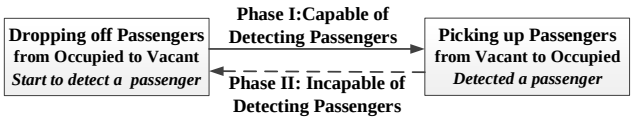


Fig 5: Roving Taxicab Sensing Cycle

- *Phase I:* Starting from dropping off a passenger (changing its status from occupied to vacant), a taxicab aims to find a new passenger and is capable of detecting a passenger by roving on streets until it picks up one.
- *Phase II:* Starting from picking up a passenger (changing its status from vacant to occupied), a taxicab aims to deliver this passenger and is incapable of detecting new passengers until it drops off this passenger.

According to the above two phases, we face three challenges to design an inferring model based on roving sensor networks: (i) taxicab sensors can only detect passengers in their sensing cycles' Phase I where they cruise streets to look for passengers (Phase I accounts for roughly 40% of a taxicab daily operating time on average [8]), and after they pick up passengers and enter Phase II, they cannot detect other passengers waiting on streets (left-behind passengers); (ii) taxicab sensors can only indicate the passenger pickup moments, not arriving moments that have to be considered in demand modeling; (iii) taxicab sensors can only provide simple "on" or "off" status, which is difficult to be utilized to mine real world logical information that should be considered in an inferring model, *e.g.*, many waiting passengers in extreme weather.

Even though with constrained sensing capabilities, roving taxicabs still provide substantial information about passengers to address these challenges. Such information is given by both events detected by taxicabs and phenomena inferred from detected events, as shown by the following two subsections.

4.3 Detected Events

We observe two kinds of events related to passenger demand by tracking taxicabs' sensing records.

4.3.1 *Pickup Event*

If a taxicab's status turns from "unoccupied" to "occupied" in two consecutive records, then it indicates that this taxicab just *picks up* a passenger in the location indicated by corresponding GPS coordinates, which is associated to a *pickup event*; similarly, a *dropoff event* is indicated.

Figure 6 gives a daily pickup event distribution among 495 Shenzhen city regions. A warmer color indicates a higher number of pickup events.

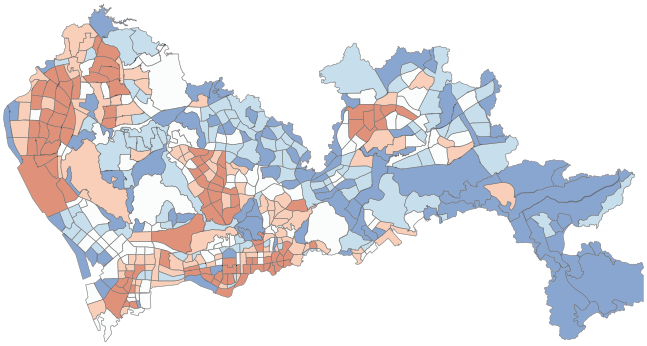


Fig 6: Pickup Event Distribution in Urban Region

Figure 7 gives a graph representing the corresponding pickup and dropoff events in 245 major urban regions in Shenzhen (including an airport, train stations, residential areas, *etc*) from 7AM to 9AM of a Monday. The size of vertex indicates the number of events in the corresponding region; the color of vertex indicates one of six urban districts. A link indicates passenger mobility patterns between two regions, which are obtained by aggregating all trips between these two regions. We remove the links with trips fewer than 30 for clarity.

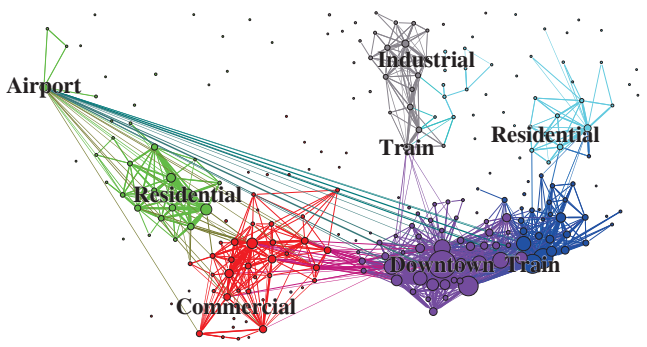


Fig 7: Corresponding Pickup and Dropoff Events

4.3.2 *Cruising Event*

A cruising event begins with a dropoff event and finally ends with a pickup event. Figure 8 gives a

cruising event where a taxicab first drops off a passenger between l_1 and l_2, and then cruises from l_2 to l_3, and finally picks up a new passenger between l_3 and l_4. By this cruising event, we infer an absence of passengers on the segment from l_2 to l_3 during the time when this taxicab cruises it.

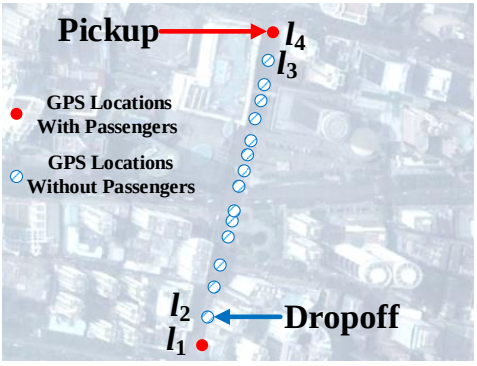


Fig 8: Cruising Events

4.4 Inferred Phenomena

Based on the above two events, we study two inferred phenomena as follows.

4.4.1 Passengers on a Spatio-Temporal Area

Phenomenon 1 in Figure 9 gives a pickup event p_i where a vacant taxicab T_i cruised a segment s_j and picked up one passenger P_i. Based on this observation, we infer that there is only one passenger (*i.e.*, P_i) in the dashed temporal and spatial area. This is because if there is another passenger P_j, T_i would pick P_j up, which contradicts to the fact that T_i picked up P_i in the pickup event p_i. Phenomenon 2 in Figure 9 shows a cruising event where a vacant taxicab T_j cruised a segment s_j and did not pick up any passenger. Based on this observation, we infer that there is no passenger in the dashed temporal and spatial area. This is because if there is a passenger P_j, T_j would pick P_j up, which contradicts the fact that T_j did not pick up passengers when it cruises s_j. Note that there may be a passenger outside the dashed area yet inside the rectangle, since a passenger can arrive at a location on segment s_j, after vacant taxicabs passed this location. This newly arriving passenger cannot be detected until he/she is picked up by another vacant taxicab.

4.4.2 Arriving Moments of Picked Up Passengers

An arriving moment indicates the time when a passenger starts to wait for a taxicab, which is used to obtain the ground truth of a total passenger count for a segment during a slot. Accurately obtaining such arriving moments is almost impossible under current infrastructures. But we present a method to obtain the upper bound of an arriving moment

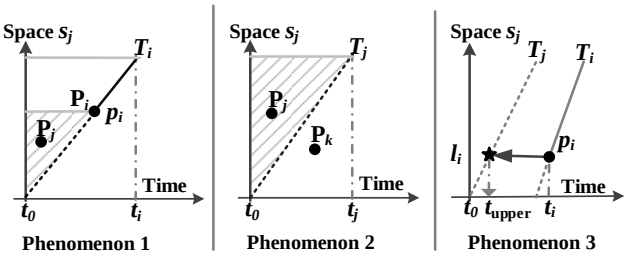


Fig 9: Inferred Phenomena

t_{upper}, *i.e.*, the earliest possible moment of a passenger starting to wait for a taxicab. As in Phenomenon 3 of Figure 9, supposing passengers do not move significantly when waiting for taxicabs, given a pickup event p_i in terms of pickup moment t_i and location l_i, we find the latest cruising event where another vacant taxicab T_j passed the same location l_i (shown as the star). Thus, the moment t_{upper} when T_j passed l_i is the upper bound of the arriving moment of the passenger P_i in the pickup event p_i. This is because if the moment that P_i starts to wait for a taxicab is earlier than this bound t_{upper}, then P_i would be picked up by T_j at t_{upper}, which contradicts the fact that P_i was picked up by T_i at t_i. We use this upper bound as the arriving moment (e.g., pushing more arriving passengers to earlier slots), which leads to a lower bound of the arrival count for the latest slot, enabling a cautious inference. Note that waiting passengers' arriving moments cannot be inferred until they are picked up.

Inferred by a roving sensor network, the above phenomena provide abundant information with high resolutions, and are used by Dmodel to infer passenger demand as follows.

5 Model Generation

Dmodel is a dynamic inference model for generic passenger demand at road-segment levels on a fine-grained temporal basis (*e.g.*, one hour). Conceptually, for a segment s_j, at the end of a slot τ_i, Dmodel takes both real-time data uploaded in τ_i and historical data uploaded before τ_i as input, and produces inferred demand in terms of a total passenger count for the next slot τ_{i+1}, by summing up two kinds of passengers as follows.

Previous Left-behind Passengers who had arrived at segment s_j before the end of τ_i, and yet were not picked up in τ_i. To obtain their count, Dmodel first aggregates real-time pickup events to obtain the *pickup count* for picked up passengers in τ_i. Next, Dmodel employs a novel parameter called pickup pattern to obtain customized training data to infer the *total passenger count* (either picked up or not) in τ_i by corresponding pickup counts. Finally, Dmodel obtains the left-behind passenger count by

TABLE 1. Main Notations

Notation	Description
p_i	Pickup Event i
P_i	Passenger i
T_i	Taxi i
t_i	Pickup Time in Pickup Event i
l_i	Pickup Location in Pickup Event i
a_i	Passenger Arriving Event for Pickup Event i
τ_i	Modeling Time Slot i
d_x	Day x
s_j	Road Segment j
$\mathbb{P}_{\tau_i}^{s_j}$	Pickup Count during τ_i at s_j
$\mathbb{L}_{\tau_i}^{s_j}$	Left-behind Count during τ_i at s_j
$\mathbb{A}_{\tau_i}^{s_j}$	Arrival Count during τ_i at s_j
$\mathbb{T}_{\tau_i}^{s_j}$	Total Passenger Count during τ_i at s_j
$\rho_{\tau_i}^{s_j}$	Pickup Pattern τ_i at s_j

subtracting the pickup count of τ_i from the total passenger count of τ_i.

Future Arriving Passengers who have not arrived yet but will arrive during τ_{i+1} at segment s_j, *i.e.*, the future arrival. Dmodel infers the future arrival by maintaining a probability distribution of a passenger arrival rate for every road segment. At the end of a slot τ_i, based on the pickup count in τ_i, Dmodel first infers the corresponding passenger arrival in τ_i, and then updates the distribution of arrival rates accordingly, and finally infers the future arrival by this updated distribution.

As follows, we present passenger demand modeling, and then elaborate how to obtain these two kinds of passengers.

5.1 Passenger Demand Modeling

The notations used in this paper are given in Table I. Four key notations for a slot τ_i and a segment s_j are given as follows.

- $\mathbb{P}_{\tau_i}^{s_j}$***: Pickup Count***: The total number of picked up passengers during τ_i at s_j.
- $\mathbb{L}_{\tau_i}^{s_j}$***: Left-behind Count***: The total number of waiting yet not picked up passengers during τ_i at s_j.
- $\mathbb{A}_{\tau_i}^{s_j}$***: Arrival Count***: The total number of arriving passengers during τ_i at s_j.
- $\mathbb{T}_{\tau_i}^{s_j}$***: Total Passenger Count***: The total number of passengers who wait for taxicabs during τ_i at s_j.

As follows, we omit all same superscripts for a concise notation. Figure 10 shows examples of the notations. The x-axis is the time, and the y-axis is the space, *i.e.*, segment s_j. A total of three passengers is picked up, indicated by three pickup events p_1, p_2 and p_3. Further, three arriving events that they start to wait for taxicabs are given by a_1, a_2 and a_3. As a result, for the time slot τ_0, $\mathbb{A}_{\tau_0}=1$, $\mathbb{P}_{\tau_0}=0$, $\mathbb{L}_{\tau_0}=1$, $\mathbb{T}_{\tau_0}=1$; for the time slot τ_1, $\mathbb{A}_{\tau_1}=2$, $\mathbb{P}_{\tau_1}=2$, $\mathbb{L}_{\tau_1}=1$, $\mathbb{T}_{\tau_1}=3$; for the time slot τ_2, $\mathbb{A}_{\tau_2}=0$, $\mathbb{P}_{\tau_2}=1$, $\mathbb{L}_{\tau_2}=0$, $\mathbb{T}_{\tau_2}=1$. Given pickup points, arriving points, and waiting periods as shown by *dots*, *stars*, and *dashed lines* between stars and dots in Figure 10, $\mathbb{P}_{\tau_i}$, $\mathbb{A}_{\tau_i}$, or $\mathbb{T}_{\tau_i}$ for a time slot τ_i are obtained by as simply as counting dots, stars, or dashed lines, respectively. Note that although some passengers are double-counted at different slots, all passengers are counted once at the same slot.

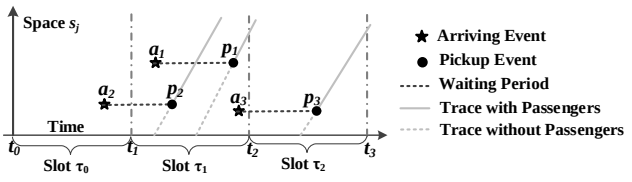


Fig 10: Notation Example

5.1.1 Demand Modeling by a Hidden Markov Chain

In Figure 11, we analyze passenger demand as an unobservable state in a Hidden Markov Chain.

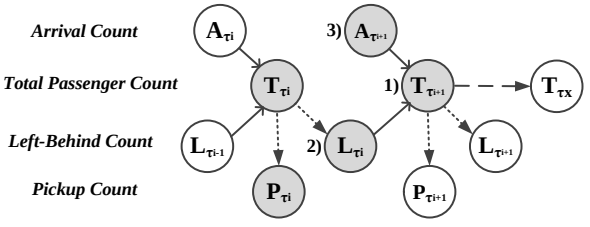


Fig 11: Passenger Demand in a Hidden Markov Chain

1) At the end of a slot τ_i, the key system state that needs to be inferred is the total passenger count $\mathbb{T}_{\tau_{i+1}}$ of the next slot τ_{i+1}, which takes the left-behind count $\mathbb{L}_{\tau_i}$ of τ_i and the arrival count $\mathbb{A}_{\tau_{i+1}}$ of τ_{i+1} as two inputs (shown by the arrows with solid lines). Thus we have
$$\mathbb{T}_{\tau_{i+1}} = \mathbb{L}_{\tau_i} + \mathbb{A}_{\tau_{i+1}}.$$
2) As one input for $\mathbb{T}_{\tau_{i+1}}$, the left-behind count $\mathbb{L}_{\tau_i}$ of τ_i is also one of two outputs (shown by the arrows with dashed lines) of the previous system state, *i.e.*, the total passenger count $\mathbb{T}_{\tau_i}$ of τ_i. The other output of $\mathbb{T}_{\tau_i}$ is the observable pickup count $\mathbb{P}_{\tau_i}$ of τ_i. Thus we have
$$\mathbb{L}_{\tau_i} = \mathbb{T}_{\tau_i} - \mathbb{P}_{\tau_i}.$$
3) As the other input for $\mathbb{T}_{\tau_{i+1}}$, the arrival count $\mathbb{A}_{\tau_{i+1}}$ of τ_{i+1} is inferred by a stochastic process, supposing passengers arrive according to a generic Poisson process.
4) Thus, combining two equations together, we have our key inferring equation as follows.
$$\mathbb{T}_{\tau_{i+1}} = (\mathbb{T}_{\tau_i} - \mathbb{P}_{\tau_i}) + \mathbb{A}_{\tau_{i+1}}. \quad (1)$$

5.1.2 Inference Overview

As in Figure 12, at the end of every slot, *e.g.*, current time t_{i+1}, Dmodel infers $\mathbb{T}_{\tau_{i+1}}$ for a segment s_j by Eq.(1) with four steps as follows.

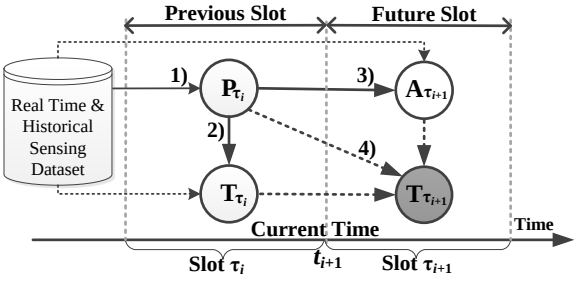


Fig 12: Inference Overview

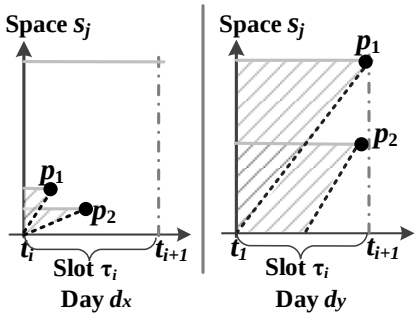


Fig 13: Pickup Patterns

1) It *infers* pickup count $\mathbb{P}_{\tau_i}$ by aggregating pickup events in the latest slot τ_i from *real-time* data;
2) It *infers* total passenger count $\mathbb{T}_{\tau_i}$ based on the corresponding pickup count $\mathbb{P}_{\tau_i}$ and a customized corrective model trained by both *historical* and *real-time* data.
3) It *infers* arrival count $\mathbb{A}_{\tau_{i+1}}$ for the next time slot τ_{i+1} by a probability distribution $\mathcal{D}$ of passenger arrival rate λ at segment s_j, which is periodically maintained through a Bayesian updating based on pickup count $\mathbb{P}_{\tau_i}$.
4) It *infers* total passenger count $\mathbb{T}_{\tau_{i+1}}$ for the next slot τ_{i+1} with Eq.(1), by $\mathbb{P}_{\tau_i}$, $\mathbb{T}_{\tau_i}$ and $\mathbb{A}_{\tau_{i+1}}$.

In the above steps, steps 1) and 4) are straightforward, so we elaborate steps 2) and 3) in Section 5.2 and 5.3.

5.2 Inferring Total Passenger Count $\mathbb{T}_{\tau_i}$

We first introduce our key novelty about using pickup patterns, and then propose how to infer $\mathbb{T}_{\tau_i}$.

5.2.1 Pickup Pattern

In this work, we infer the total passenger count by four factors, which include (i) *time* in terms of a time slot of a day (*e.g.*, slot τ_i), (ii) *location* in terms of a road segment (*e.g.*, segment s_j), (iii) *pickup count* in terms of how many passengers have been picked on a segment during a slot, and (iv) *pickup pattern* in terms of how fast passengers were picked up, which may infer hidden contexts, *e.g.*, extreme weather or major events. The existing work has been considering the first three factors, but the pickup pattern has not been considered by others before. In this work, we argue that the pickup count is inherently limited by taxicab supply, and cannot provide enough inferring information. But our pickup patterns provide extra hidden contexts to increase inference accuracy.

Figure 13 presents the same slot τ_i at two different days with the same pickup count yet with different pickup patterns. The key difference of the same slot τ_i for day d_x and d_y is how long it takes for vacant taxicabs to pick up passengers during τ_i, which is associated to the *pickup pattern*, *i.e.*, the taxicabs in d_x pick up two passengers very quickly; whereas the taxicabs in d_y cruise for a long time before picking up two passengers. The pickup pattern gives us extra hidden online contexts, and cannot be replaced by other contexts already used by the existing work, *i.e.*, slot τ_i, segment s_j, and pickup count $\mathbb{P}_{\tau_i \in d_z}$ of a particular day d_z, since in Figure 13, all other contexts are the same, but two slots τ_i in d_x and d_y have different pickup patterns. For example, the hidden online contexts in the pickup pattern during $\tau_i \in d_x$ may indicate suddenly increased demand due to extreme weather, train arrival or other events, since all taxicabs pick up passengers very quickly. Whereas the pickup pattern for $\tau_i \in d_y$ may indicate a normal scenario without increased demand. Intuitively, though d_x and d_y have same pickup count $\mathbb{P}_{\tau_i \in d_x} = \mathbb{P}_{\tau_i \in d_y} = 2$ (may result from limited taxicab supply), $\tau_i \in d_x$ shall have a larger total count $\mathbb{T}_{\tau_i \in d_x}$ than $\tau_i \in d_y$.

To quantify the pickup pattern as a formal parameter, as in Figure 13, we first use a random variable r_{p_i} to indicate the area ratio between the dashed triangle spatio-temporal area introduced by a pickup event p_i and the entire rectangle spatio-temporal area. For example in Figure 13, suppose the dashed spatio-temporal triangle area $\mid \triangle_{p_1} \mid$ associated with p_1 during slot τ_i on Day d_x is equal to 1, and the entire spatio-temporal rectangle area $|s_j| \times |\tau_i|$ is equal to 16. Thus, r_{p_1} is equal to $\frac{|\triangle_{p_1}|}{|s_j| \times |\tau_i|} = \frac{1}{16}$. By introducing this ratio, we integrate both the pickup location and time for a pickup event p_1 by a random variable, because different pickup locations and times in the same spatio-temporal context lead to different areas of dashed triangles, and thus lead to different area ratios. Further, we integrate all pickup events $\{p_1, p_2, ..., p_n\}$ associated with a given spatio-temporal context (*i.e.*, during a given slot τ_i on a given road segment) by an integration of associated random variables $\{r_{p_1}, r_{p_2}, ...r_{p_n}\}$. In particular, our integration is based on the entropy ρ of these random variables to indicate the pickup pattern under this spatio-temporal combination.

$$\rho_{\tau_i} = -\sum_{i=1}^{n} r_{p_i} \log r_{p_i}$$

where n is the total number of pickup events during

slot τ_i in this road segment. A low entropy shows a low randomness of area ratios associated with these pickups, which indicates pickups always happened at similar locations of a given segment s_j during similar times of a slot τ_i, and thus leads to similar area ratios. As in evaluations, ρ accounts for many real-world scenarios that cannot be captured with pickup counts due to limited taxi supply.

5.2.2 Customized Online Training

Based on the new online factor, pickup pattern, and other three factors, we discuss how to infer the total passenger count as follows.

Given a segment s_j and a slot τ_i, the pickup count $\mathbb{P}_{\tau_i}$ and the total passenger count $\mathbb{T}_{\tau_i}$ have a logical relationship: $\mathbb{P}_{\tau_i}$ is the lower bound of $\mathbb{T}_{\tau_i}$, since all picked up passengers are included in the total passenger count. Thus, we quantitatively investigate their relationship as follows. Given the historical dataset, for particular slots and segments, we obtain the ground truth of $\mathbb{P}_{\tau_i}$ by aggregated pickup events, and infer the ground truth of $\mathbb{T}_{\tau_i}$ based on the method of inferring arriving moments (introduced in Section 4.4.2), *e.g.*, in Figure 10, after inferring arriving moments (shown by stars), $\mathbb{T}_{\tau_i}$ are obtained by counting dashed lines linking dots and stars in a slot (*e.g.*, $\mathbb{T}_{\tau_1} = 3$). As a result, Figure 14 gives the relationship between $\mathbb{P}$ and $\mathbb{T}$ for 10 randomly selected road segments in five τ_8 slots from 8 to 9AM in five weekdays. It indicates an approximate linear relationship for $\mathbb{P}_{\tau_8}$ and $\mathbb{T}_{\tau_8}$ for the same segment s_j.

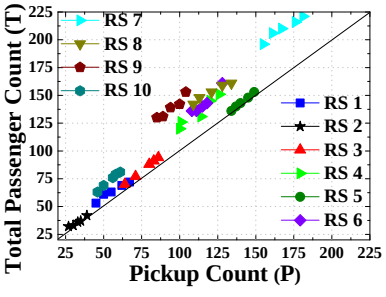


Fig 14: $\mathbb{T}$ vs. $\mathbb{P}$ in 10 Segments

Based on the above observation, we propose a customized online training model based on the linear regression as follows. Supposing that (i) we have a historical dataset consisting of taxicab GPS data about $K-1$ different days, *i.e.*, day d_1 to day d_{K-1}, and (ii) the current time is the end of slot τ_i in day d_K, Dmodel infers the total passenger count $\mathbb{T}_{\tau_i \in d_K}$ with four steps as follows.

1) It *calculates* both pickup count $\mathbb{P}_{\tau_i \in d_K}$ and the corresponding pickup pattern $\rho_{\tau_i \in d_K}$, based on real-time data about the latest slot $\tau_i \in d_K$.
2) It *selects* the data of days whose τ_i have similar pickup pattern $\bar{\rho}$ to $\rho_{\tau_i \in d_K}$ as a customized training dataset with M pairs of $(\mathbb{P}_{\tau_i \in d_m}, \mathbb{T}_{\tau_i \in d_m})$ where $1 \le m \le M$ (one pair for every day).
3) It *trains* the following model by the M pairs of $(\mathbb{P}_{\tau_i \in d_m}, \mathbb{T}_{\tau_i \in d_m})$ to learn customized $\alpha_{\tau_i \in d_K}$ and $\beta_{\tau_i \in d_K}$.

$$\mathbb{T}_{\tau_i \in d_m} = \alpha_{\tau_i \in d_K} + \beta_{\tau_i \in d_K} \times \mathbb{P}_{\tau_i \in d_m}. \quad (2)$$

4) It *utilizes* $\alpha_{\tau_i \in d_K}$, $\beta_{\tau_i \in d_K}$ and pickup count $\mathbb{P}_{\tau_i \in d_K}$ to obtain total passenger count $\mathbb{T}_{\tau_i \in d_K}$ with Eq.(2).

A similar pickup pattern $\bar{\rho}$ to $\rho_{\tau_i \in d_K}$ is defined by $\bar{\rho} \in [\rho_{\tau_i \in d_K} \cdot (1-\Delta\rho), \rho_{\tau_i \in d_K} \cdot (1+\Delta\rho)]$ where $\Delta\rho$ is a given parameter and carefully evaluated in Section 6.

5.3 Inferring Arrival Count $\mathbb{A}_{\tau_{i+1}}$

Dmodel infers passenger arrival with a stochastic process where an arrival rate λ of a Poisson Process varies in Brownian motion, which is widely used to model passenger arrival or network package arrival [9]. Thus, $\mathbb{A}_{\tau_{i+1}} = \lambda_{\tau_{i+1}} \times |\tau_{i+1}|$. Note that we did not use a customized training to infer the arrival count $\mathbb{A}_{\tau_{i+1}}$ based on given pickup count $\mathbb{P}_{\tau_i}$ as in the last subsection, since there is no potentially logical relationship between $\mathbb{P}_{\tau_i}$ and $\mathbb{A}_{\tau_{i+1}}$.

5.3.1 Passenger Arrival Rate Modeling

Dmodel maintains a probability distribution $\mathcal{D}$ of λ for a segment s_j by discretizing the space of passible λ, and assumes that (i) λ is one of discrete values from 0 to the maximum λ (obtained by the dataset) and (ii) the initial probability for all possible λ is uniformly distributed. Therefore, at the end of the slot τ_i, Dmodel updates $\mathcal{D}$ with three steps.

1) It *evolves* $\mathcal{D}$ to the current time by applying Brownian motion to every possible rate by assuming that λ is undergoing a continuous-time stochastic process.
2) It *infers* the arrival count $\mathbb{A}_{\tau_i}$ in τ_i based on observed $\mathbb{P}_{\tau_i}$, and *calculates* probabilities that this arrival count $\mathbb{A}_{\tau_i}$ is associated to every one of arrival rates as follows.

$$F(x) \leftarrow \mathcal{D}_{\text{old}}(\lambda_{\tau_i} = x) \times \mathrm{e}^{-x \cdot |\tau_i|} \frac{(x \cdot |\tau_i|)^{\mathbb{A}_{\tau_i}}}{\mathbb{A}_{\tau_i}!}.$$

3) It *normalizes* these probabilities, so they sum to unity.

$$\mathcal{D}_{\text{new}}(\lambda_{\tau_i} = x) \leftarrow \frac{F(x)}{\sum_k F(k)}.$$

These three steps constitute Bayesian updating for $\mathcal{D}$. Given $\mathcal{D}$, we try to infer $\lambda_{\tau_{i+1}}$ with a cautious estimate to bound a risk of overinferring. So, we employ the ωth percentile of $\mathcal{D}$ to calculate the inferred $\lambda_{\tau_{i+1}}$, *e.g.*, 40th percentile. In Dmodel, ω is a given parameter, and is evaluated in Section 6.

A key unresolved question is how to infer the arrival count $\mathbb{A}_{\tau_i}$ by the pickup count $\mathbb{P}_{\tau_i}$, which is introduced as follows.

5.3.2 Inferring Previous Arrival Count $\mathbb{A}_{\tau_i}$

We introduce how to infer $\mathbb{A}_{\tau_i}$ in Figure 15 where we classify all passengers associated to the total passenger count $\mathbb{T}_{\tau_i}$ of a slot τ_i into four parts, based on when they arrived and whether they got picked up at the end of slot τ_i. Thus, the sum of passengers in Part 1 and Part 2 is the arrival count $\mathbb{A}_{\tau_i}$ we try to infer. The sum of passengers in Part 2 and Part 3 is the pickup count $\mathbb{P}_{\tau_i}$; the sum of passengers in all four parts is the total passenger count $\mathbb{T}_{\tau_i}$; we have already obtained both of them in the previous subsection.

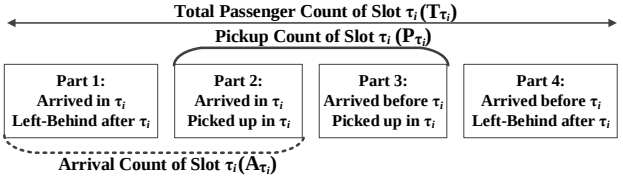


Fig 15: Inferring Previous Arrival Count $\mathbb{A}_{\tau_i}$

We add the following two kinds of passengers to infer $\mathbb{A}_{\tau_i}$.

1) *Passengers in Part 1*: Since we have already inferred the total passenger count $\mathbb{T}_{\tau_i}$ based on $\mathbb{P}_{\tau_i}$ in Section 5.2, we have the total number of passengers in Parts 1 and 4 together, *i.e.*, $\mathbb{T}_{\tau_i} - \mathbb{P}_{\tau_i}$. Further, since an inferring slot (*e.g.*, one hour) is typically longer than a passenger waiting period, the number of passengers in Part 4 is 0. Thus we have the number of passengers in Part 1 alone.
2) *Passengers in Part 2*: We differentiate the passengers in Parts 2 and 3 by inferring arriving moments of these picked up passengers in $\mathbb{P}_{\tau_i}$, based the method in Section 4.4.2, and we obtain the number of passengers in Part 2 alone.

6 DMODEL EVALUATION

We evaluate Dmodel based on a 450 GB dataset introduced in Section 4. We divide the entire 182-day dataset into two subsets. *Testing Dataset*: it contains the data about a particular day, *e.g.*, day d_1, serving as the real-time streaming data in the evaluation. *Training Dataset*: it contains the data about the rest of days, serving as the historical training data. For this particular day d_1, if we use one-hour slots, at the end of the first slot, *i.e.*, time 01:00, we use Dmodel to infer the total passenger count for the next slot from 01:00 to 02:00, based on both the "real-time" data from 00:00 to 01:00 in the testing dataset, and all data in the historical training data. We let the testing dataset rotate among all 182 days of data, leading to 182 sets of experiments. The average results were reported.

We compare Dmodel with two models: SDD and Basic. **SDD** model is one of the state-of-the-art taxicab demand and supply models, which maintains a distribution for passenger demand based on the previous average demand [4]. SDD model serves as a statistical model and is suitable for the real world scenario where the real-time data collection is not possible, and we can only use the historical data to infer passenger demand. **Basic** model first uses the generic offline training to train the entire dataset to obtain parameters (α and β) offline without considering real-time pickup patterns. Basic serves as a baseline for Dmodel to show the effects of the ignorance of logical contexts shown by pickup patterns (*e.g.*, extreme weather or events) on the model performance. Dmodel performs similarly with Basic except that it uses logical contexts (pickup and cruising events) in the testing dataset to calculate a pickup pattern for a particular slot, and selects the data of slots with similar pickup patterns as a customized training dataset to perform an online training as introduced in Section 5.2.2.

By processing the entire dataset with a method of inferring passenger arrival moments (introduced in Section 4.4.2), we infer the ground truth of total passenger counts used to test models with a key metric, called **Accuracy**. The accuracy is defined as a ratio $= \frac{\bar{\mathbb{T}}-|\bar{\mathbb{T}}-\mathbb{T}|}{\bar{\mathbb{T}}}$ where $\mathbb{T}$ is the inferred total passenger count of a particular model and $\bar{\mathbb{T}}$ is the total passenger count obtained from the inferred ground truth.

We first test models on 4 and 1000 road segments about accuracy with different slot lengths. Then, we investigate the sensitivity of Dmodel to two key parameters: $\Delta\rho$ and ω used in Sections 5.2.2 and 5.3.1, and obtain their optimal default values. Next, we study impacts of lengths of historical data on model accuracy and model running times. Finally, we present a summary.

6.1 Inference Accuracy

In this subsection, we show low-level comparisons on 4 particular road segments, and high-level comparisons on 1000 road segments. All road segments are randomly selected in the downtown area of Shenzhen.

6.1.1 Low-Level Comparisons

Figure 16 plots accuracy of three models on 4 road segments under one-hour slots. Dmodel has a better performance than Basic and SDD, especially at the non-rush hour, *e.g.*, 18:00 to 06:00. Basic outperforms SDD in the early morning, *e.g.*, 00:00 to 06:00, and the late night, *e.g.*, 18:00 to 00:00. SDD has good accuracy during the morning rush hour, *e.g.*, 08:00 to 12:00, and we believe this is because during the rush hour, passenger demand is relative stable compared

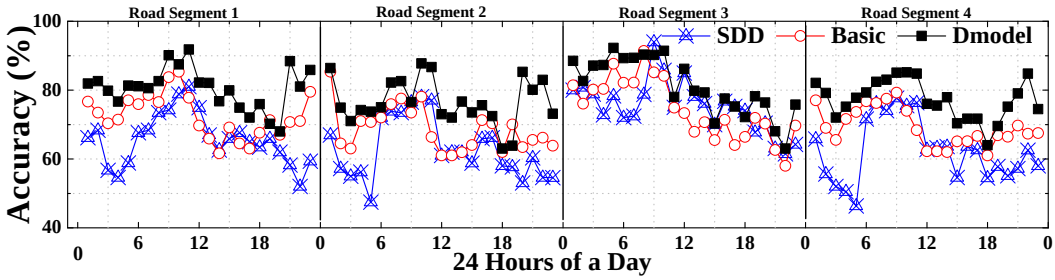


Fig 16: Accuracy under One Hour Slot for 24 Hours in Four Road Segments

to other time periods. We notice that performances of different models are also dependent on locations, *e.g.*, in road segment 1, Basic outperforms SDD during almost all the morning, but in road segment 3, SDD has a better performance roughly from 10:00 to 23:00. Further, we also observe that even though Dmodel has a better performance in general, but during some hours and in some locations, Basic indeed has a better performance, such as at road segment 3 from 18:00 to 19:00. This may be because for this road segment and time period, Dmodel did not obtain a good parameter from the training data selected according to pickup patterns.

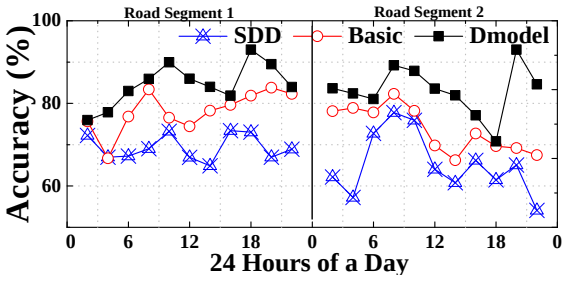


Fig 17: Accuracy under Two Hour Slot for 24 Hours

Figure 17 shows comparisons in segments 1 and 2 under two-hour slots. With a longer slot, the accuracy generally increases for all three models. This is because (i) passenger demand is more stable in a longer slot, and thus SDD model becomes more effective; (ii) a longer slot increases accuracy of passenger arrival predictions in Dmodel and Basic, which leads to increased inference accuracy. We also notice that in a longer slot, the performance gain between Dmodel and others increases, and this may be because the advanced online training used by Dmodel is more effective for a longer slot.

6.1.2 High Level Comparisons

Figure 18 gives the average accuracy on 1000 road segments under one-hour slots at different hours of a day. The average accuracy of all three models on 1000 road segments is lower than the accuracy we observed on 4 particular road segments. It is because passenger demand may change dramatically between different segments. But the relative performance between three models is similar to Figure 16. Basic outperforms SDD at the most of the time by 18% on average, except at the evening rush hour where SDD outperforms Basic by 5%. Dmodel has a better performance than SDD and Basic by 42% and 13% on average, which results from its customized online training. In addition, we find that SDD model has a poor performance during the non-rush hour when passenger demand is not stable. But Dmodel overcomes this issue by its effective inferring. Dmodel has 83% accuracy at the 9AM slot, which is the default slot for the following experiments.

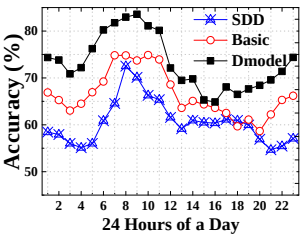


Fig 18: Average Accuracy

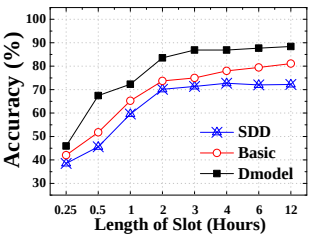


Fig 19: Effects of Slot Lengths

6.2 Sensitivity of Dmodel

We study the sensitivity of Dmodel to slot lengths and two parameters $\Delta\rho$ and ω on 1000 segments.

6.2.1 Slot Length vs. Accuracy

Figure 19 gives the average accuracy on 1000 segments with different slot lengths. The average accuracy of all models increases with the lengths of slots. The increasing on accuracy slows down when slots are longer than 2 hours. This is because passenger demand in a longer slot becomes more stable at different days. But when a slot is short, *e.g.*, 15 mins or 30 mins, the average passenger demand is variable at different days. When the slot becomes longer, Dmodel and Basic have the similar performance, because pickup patterns for long slots are mostly similar, and cannot be used by Dmodel to differentiate related slots.

6.2.2 $\Delta\rho$ vs. Accuracy

$\Delta\rho$ is used to decide similarity between pickup patterns as in Section 5.2.2. Figure 20 gives effects of $\Delta\rho$ on Dmodel. With the increase of $\Delta\rho$, the accuracy of

Dmodel increases first, and then decreases. This is because when $\Delta\rho$ increases, Dmodel finds more slots with similar pickup patterns to effectively train a customized corrective model online. But when $\Delta\rho$ becomes too large, Dmodel has to consider more slots with different pickup patterns, leading to a poor performance. Further, when $\Delta\rho$ becomes larger than 0.5, Dmodel has to consider many slots, similar to Basic model. The accuracy peaks when $\Delta\rho = 0.2$, which is set as the default value of $\Delta\rho$. If the used $\Delta\rho$ leads to an empty training dataset, $\Delta\rho$ increases until the training dataset is not empty.

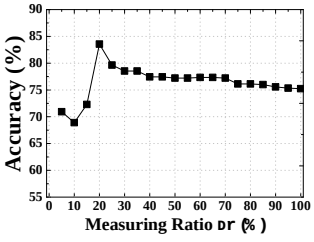


Fig 20: $\Delta\rho$ vs. Accuracy

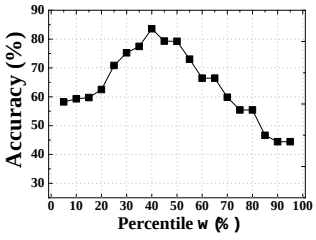


Fig 21: ω vs. Accuracy

6.2.3 ω vs. Accuracy

ω is used to decide the percentile to predict the future passenger arrival as in Section 5.3.1. Figure 21 plots effects of ω on Dmodel. A small ω indicates that Dmodel conservatively predicts arrival rates; whereas a large ω indicates Dmodel aggressively predicts arrival rates. We find that both a small and large ω lead to a poor performance, since a small or large predicted passenger arrival rate reduces the accuracy of Dmodel. The accuracy peaks when $\omega = 0.4$, which is set as the default value of ω.

6.2.4 Impact of Impact of Historical Data

In this subsection, we study the impact of historical data h in terms of weeks on model accuracy and running times by comparing Dmodel to Basic with a default value of 26 weeks. Normally, the more the historical data, the more accurate the models. However, naively using more data actually reduces model accuracy. Figure 22 plots the accuracy of Basic and Dmodel with different lengths of historical data in terms of weeks. We find that Dmodel always has a better performance than Basic after the length of historical data is longer than one week. This is because the pickup pattern Dmodel used selects highly related data to infer demand, which has higher accuracy than Basic using all data for training.

Figure 23 plots running times of Basic and Dmodel with different lengths of historical data in terms of weeks. We find that Dmodel always has shorter running times after the length of historical data is longer than 3 weeks. This is because calculations for pickup patterns (*e.g.*, calculating entropy) in Dmodel take a higher portion of running times when the length of historical data is short. But when the length of historical data becomes longer, these

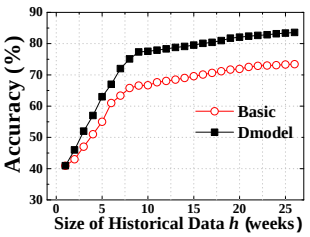


Fig 22: h vs. Accuracy

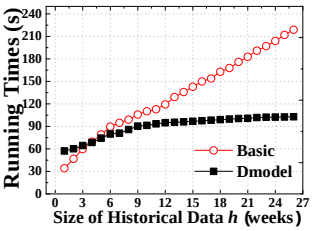


Fig 23: h vs. Running Time

calculations only account for a small portion of running times. However, Basic requires longer running times because it uses all historical data for training.

6.3 Dmodel Evaluation Summary

We made the following observations based on results. (i) The accuracy of inferring models is highly dependent on both locations and times as in Figures 16 and 17. On average, different models have different accuracy at different times as in Figure 18. Compared to locations, all models are more sensitive to time. (ii) System parameters pose significant impacts of accuracy, and the optimal parameters have to be carefully evaluated. The length of slots has significant impacts on the relative performance between all models as in Figure 19. It is intuitive that a longer slot has better accuracy for all models, but a longer slot also has a low inferring usability for many applications. As in Figure 19, it seems that the two-hour slot is a good tradeoff between the usability and the accuracy of the real-time inferring model. Further, as in Figures 20 and 21, both $\Delta\rho$ and ω have impacts on accuracy of models, and ω has a bigger impact compared to $\Delta\rho$. (iii) By selecting a compact size of highly related data, we increase model accuracy and reduce running times at the same time as shown by Figures 22 and 23. (iv) Compared to models statistically inferring demand with only historical average demand data, models using real-time pickup events have better inferring accuracy, which is shown by the fact that Dmodel and Basic outperform SDD as in Figures 16, 17 and 18. (v) Taking logical information (weather or events) shown by pickup patterns into considerations further increases model accuracy, which is shown by the fact that Dmodel outperforms Basic as in the most figures.

7 Dmodel Application

We propose a Dmodel application where a dispatch center employs demand inferred by Dmodel to achieve an equilibrium between passenger demand and taxicab supply, given 245 major urban regions in Shenzhen as shown in Figure 7.

In our application, at the end of a "real-time" slot τ_i, we first use Dmodel to infer passenger demand $\mathbb{T}^{r_x}_{\tau_{i+1}}$ for the next slot τ_{i+1} in region r_x by aggregating inferred demand of all road segments in region

r_x. Next, we employ real-time data to aggregate vacant taxicabs in region r_x to obtain "dispatchable" vacant taxicab supply in region r_x for the next slot τ_{i+1}, indicated by $\mathbb{S}^{r_x}_{\tau_{i+1}}$. Similarly, we shall have all $\mathbb{T}^{r_x}_{\tau_{i+1}}$ and $\mathbb{S}^{r_x}_{\tau_{i+1}}$ where $1 \leq x \leq 245$. Finally, since our paper is focused on modeling, we use a straightforward scheme to dispatch vacant taxicab supply $\sum_{1\leq x\leq 245} \mathbb{S}^{r_x}_{\tau_{i+1}}$ among 245 regions so that the dispatched taxicab supply $\hat{\mathbb{S}}^{r_x}_{\tau_{i+1}}$ is proportional to inferred passenger demand $\mathbb{T}^{r_x}_{\tau_{i+1}}$ in region r_x.

The evaluation is based on the ground truth of passenger demand $\bar{\mathbb{T}}^{r_x}_{\tau_{i+1}}$ of slot τ_{i+1}, and the dispatched vacant taxicab supply $\hat{\mathbb{S}}^{r_x}_{\tau_{i+1}}$ in each region at hourly slots. We propose a normalized equilibrium value $0 \leq \kappa \leq 1$ to evaluate effectiveness of dispatching: $\kappa_{\tau_{i+1}} = \text{avg}_{1\leq x\leq 245} \frac{|\bar{\mathbb{T}}^{r_x}_{\tau_{i+1}} - \hat{\mathbb{S}}^{r_x}_{\tau_{i+1}}|}{\bar{\mathbb{T}}^{r_x}_{\tau_{i+1}} + \hat{\mathbb{S}}^{r_x}_{\tau_{i+1}}}$. If the demand inferred by an inference method, *e.g.*, Dmodel, is similar to the ground truth, the corresponding dispatch leads to a small $\kappa_{\tau_{i+1}}$, indicating an equilibrium between passenger demand and taxicab supply; otherwise, it leads to a large $\kappa_{\tau_{i+1}}$, *i.e.*, disequilibrium. Note that dispatching taxicabs would skew historical taxicabs' GPS dataset. To eliminate dispatching effects, we only used dispatched supply to calculate κ, and did not manipulate taxis' traces, and we start over at the end of the next slot.

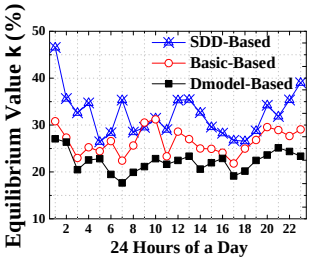


Fig 24: κ in One-hour Slots

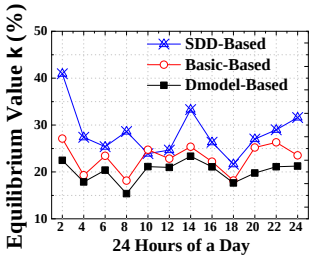


Fig 25: κ in Two-hour Slots

Figure 24 plots the equilibrium value κ at different hours of a day under one-hour slots. We observe that the equilibrium values fluctuate in all dispatching. But Dmodel-based dispatching has a lower equilibrium value almost at every slot. Basic-based dispatching outperforms SDD-based dispatching at the most of the time. Figure 25 gives the average equilibrium value κ under two-hour slots. We find the equilibrium values under two-hour slots are lower than the equilibrium values under one-hour slots for all dispatching, which verifies our previous observation that two-hour-slot based inferring is better than one-hour-slot based one in terms of accuracy. But the relative performance between three model-based dispatching shown in Figure 25 is similar to the one in Figure 24. Basic outperforms SDD at the most of the time, except at the morning rush hour where SDD outperforms Basic by 5% on average. Dmodel-based dispatching outperforms Basic-based and SDD-based dispatching by 11% on average, because of accurate inferring by Dmodel.

8 Related Work

The method to infer passenger demand with history is not new, but it is normally performed by using survey [2] [8] and static sensor data [10] [1]. In these models, both times and locations for inference are preset, and the data used for inference is often incomplete and out-of-date. Recently, several novel systems have been proposed using taxicab traces.

Some systems are proposed to assist taxicab operators for better taxicab services, *e.g.*, inferring mobility patterns for taxicab passengers [11], exploring carpooling opportunities [12], dispatching taxicabs based on inferred passenger demand [3] [4] [5], detecting anomalous taxicab trips to discover driver fraud [13], and discovering temporal and spatial causal interactions to provide timely and efficient services in certain areas with disequilibrium [14].

In addition to taxicab operators, several systems are proposed for the benefit of passengers or drivers, *e.g.*, allowing taxicab passengers to query the expected duration and fare of a planed trip based on previous trips [15], computing faster routes by taking into account driving patterns of taxicabs obtained from historical GPS trajectories [16], estimating city traffic volumes for drivers [17], and recommending a taxicab driver with a sequence of pick-up points to maximize profits [3].

Moveover, taxicab GPS records help beyond the taxicab business: (i) GPS records from experienced taxicab drivers can assist other drivers improve their driving performance [6]; (ii) GPS records can be used for navigating regular drivers to smart routes based on those of experienced taxicab drivers [18]; (iii) large-scale taxicab GPS traces enable us to better understand traffic conditions of cities [19] [20].

Yet the most of existing research on taxicab systems is mainly focused on taxicab scheduling, instead of passenger modeling, assuming that passenger demand is given by historical average pickup events, and overlooking the fact that real-time demand is different from pickup events for the same time period [21]. As a result, our model is different from the existing research by its novel inference method based on both real-time and historical data from roving sensor networks. Technically, we focus on inferring passenger demand with the compact yet customized online training with real-time pickup patterns and hidden contexts (*e.g.*, arriving moments) inferred by roving taxicab sensors, which have not been investigated before.

9 Conclusion

In this work, we motivate, design and evaluate a taxicab passenger model Dmodel and one of its applications based on a 450 GB dataset collected by a taxi system as a roving sensor network. Our effort

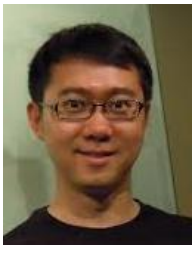
provides a few valuable insights for applying modeling techniques in Dmodel to other transportation systems. Specifically, (i) mobile taxicabs can be used as roving sensors to infer passenger demand with high accuracy; (ii) the inferring accuracy is highly dependent on locations, times, and other logical information, e.g., weather and events; (iii) the length of inferring slots also has significant impacts on the inferring accuracy, and a good tradeoff between the usability and accuracy of demand methods has to be carefully evaluated; (iv) a statistic model can be enhanced by a generic offline training considering pickup events, but it can be further enhanced by a customized online training for real-time situations.

REFERENCES

[1] "National transport authority," in *National Taxi Fare Review 2012*.
[2] "San francisco municipal transportation agency:taxi user surveys," in *San Francisco Municipal Transportation Agency*.
[3] Y. Ge, H. Xiong, A. Tuzhilin, K. Xiao, M. Gruteser, and M. Pazzani, "An energy-efficient mobile recommender system," in *KDD '10*.
[4] Y. Huang and J. W. Powell, "Detecting regions of disequilibrium in taxi services under uncertainty," in *SIGSPATIAL '12*.
[5] J. Yuan, Y. Zheng, L. Zhang, X. Xie, and G. Sun, "Where to find my next passenger," in *UbiComp '11*.
[6] J. Yuan, Y. Zheng, X. Xie, and G. Sun, "Driving with knowledge from the physical world," in *KDD '11*.
[7] S. Standard, "Shenzhen ranks fifth in the world in terms of population density," *http://www.shenzhen-standard.com*.
[8] S. Consulting, "The new york city taxicab fact book," *http://www.schallerconsult.com/taxi/taxifb.pdf*.
[9] V. Paxson and S. Floyd, "Wide area traffic: the failure of poisson modeling," *IEEE/ACM Trans. Netw.*
[10] "Taxi-transit integration in the atlanta regio," in *Georgia Regional Transportation Authority*.
[11] C. Kang, S. Sobolevsky, Y. Liu, and C. Ratti, "Exploring human movements in singapore: A comparative analysis based on mobile phone and taxicab usages," ser. UrbComp '13.
[12] P. Santi, G. Resta, M. Szell, S. Sobolevsky, S. H. Strogatz, and C. Ratti, "Quantifying the benefits of vehicle pooling with shareability networks," ser. Proceedings of the National Academy of Sciences (PNAS), 2014.
[13] D. Zhang, N. Li, Z.-H. Zhou, C. Chen, L. Sun, and S. Li, "ibat: detecting anomalous taxi trajectories from gps traces," in *UbiComp '11*.
[14] W. Liu, Y. Zheng, S. Chawla, J. Yuan, and X. Xing, "Discovering spatio-temporal causal interactions in traffic data streams," in *KDD '11*.
[15] R. K. Balan, K. X. Nguyen, and L. Jiang, "Real-time trip information service for a large taxi fleet," in *MobiSys '11*.
[16] H. Gonzalez, J. Han, X. Li, M. Myslinska, and J. P. Sondag, "Adaptive fastest path computation on a road network: a traffic mining approach," in *Proceedings of the 33rd international conference on Very large data bases*, ser. VLDB '07, 2007.
[17] J. Aslam, S. Lim, X. Pan, and D. Rus, "City-scale traffic estimation from a roving sensor network," in *SenSys '12*.
[18] L.-Y. Wei, Y. Zheng, and W.-C. Peng, "Constructing popular routes from uncertain trajectories," in *KDD '12*.
[19] W. Zhang, S. Li, and G. Pan, "Mining the semantics of origin-destination flows using taxi traces," in *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, ser. UbiComp '12, 2012.
[20] J. Yuan, Y. Zheng, and X. Xie, "Discovering regions of different functions in a city using human mobility and pois," in *KDD '12*.
[21] X. Zhan, S. Hasan, S. Ukkusuri, and C. Kamga, "Urban travel time estimation using large scale taxi data with limited information," ser. Transportation Research Part C (Emerging Technologies), 2013.

Desheng Zhang (M'10) is a Ph.D student in the Department of Computer Science and Engineering at the University of Minnesota-Twin City. His research includes big data analytics, mobile CPS, wireless sensor networks, intelligent transportation systems. He is a member of the IEEE.

Tian He (M'03-SM'12) is an associate professor with the Department of Computer Science and Engineering, University of Minnesota Twin Cities. He is the coauthor of more than 100 papers in premier journals and conferences with more than 12,000 citations. As a recipient of the US NSF CAREER Award' 09, he served a few program chair position in international conferences. His research includes wireless sensor networks, intelligent transportation systems, and distributed systems. He is a senior member of the IEEE.



Shan Lin (M'03) is an assistant professor with the Department of Electrical and Computer Engineering in Stony Brook University. He received his PhD in computer science at the University of Virginia. His research is in the area of networked systems, with an emphasis on feedback control based design in cyber physical systems. He works on wireless network protocols, medical devices, and smart transportation systems. He is a member of the IEEE.

Sirajum Munir (M'13) received his PhD in Computer Science from University of Virginia in 2014. He is currently working at Bosch Research and Technology Center as a Research Engineer. His research interest lies in the areas of cyber physical systems, wireless sensor and actuator networks, and ubiquitous computing. He has published papers in major conferences in these areas, two of which were nominated for best paper awards at ACM/IEEE ICCPS.

John A. Stankovic (F'94) received the PhD degree from Brown University. He is the BP America professor in the Computer Science Department at the University of Virginia. In the past, he served as chair of the Department for eight years. He also won the IEEE Real-Time Systems Technical Committee Award for Outstanding Technical Contributions and Leadership. He also won the IEEE Technical Committee on Distributed Processings Distinguished Achievement Award (inaugural winner). He has won seven Best Paper awards including for ACM SenSys 2006. He was the editor-in-chief for IEEE Transactions on Distributed and Parallel Systems. His research interests are in cyber physical systems, distributed computing, real-time systems, and wireless sensor networks. He is a Life Fellow of the IEEE.