Predicting Imbalanced Taxi and Passenger Queue Contexts in Airport

Conference Paper - July 2017

3 authors:

Mohammad Saiedur Rahaman
RMIT University
- 21 PUBLICATIONS
- 56 CITATIONS

Margaret Hamilton
RMIT University
- 134 PUBLICATIONS
- 1,177 CITATIONS

Flora Dilys Salim
RMIT University
- 141 PUBLICATIONS
- 650 CITATIONS

Some of the authors of this publication are also working on these related projects:

- Optimisation of large graph-based search problems View project
- Cross-Device User Experience: Evaluation and Design View project

All content following this page was uploaded by Flora Dilys Salim on 08 June 2017.

The user has requested enhancement of the downloaded file.
Predicting Imbalanced Taxi and Passenger Queue Contexts in Airport

Submission Type: Completed Research Paper

Mohammad Saiedur Rahaman  
School of Science (CS & IT)  
RMIT University  
Melbourne, VIC, Australia  
saiedur.rahaman@rmit.edu.au

Margaret Hamilton  
School of Science (CS & IT)  
RMIT University  
Melbourne, VIC, Australia  
margaret.hamilton@rmit.edu.au

Flora D. Salim  
School of Science (CS & IT)  
RMIT University  
Melbourne, VIC, Australia  
flora.salim@rmit.edu.au

Abstract

The taxi and passenger queue contexts indicate the various states of queues related to taxis and passengers (i.e. taxis are waiting for passengers, passengers are waiting for taxis, both are waiting for each other, none is waiting). Predicting these queue contexts in a future time is very important for better airport ground transport operations. However, queue context prediction at the airport is a challenging problem due to the presence of different contextual factors i.e., time, weather, taxi trips, flight arrivals and many more. Also these taxi and passenger queue contexts at the airport are imbalanced since some of the contexts are very infrequently occurring compared to others. In this paper, we address the problem of predicting imbalanced taxi and passenger queue contexts at the airport. First, we investigate different contextual factors, including time, taxi trips, passengers and weather for queue context prediction. Then we propose a detailed step by step solution to address this problem. To support the effectiveness of our detailed approach, we generate a queue context dataset by fusing three real world datasets including taxi trip, passenger wait time and weather condition that represent the taxi and passenger queue contexts at a major international airport in the New York City. The experimental results demonstrate that our developed queue context prediction framework provides detailed solutions to deliver higher accuracy in queue context prediction.

Keywords: Imbalanced learning, sampling, queue context prediction.
Introduction

Taxis are considered to be the most convenient transportation option for passenger transfer between the airport and the city. Hence the management of the passenger and the taxi queues plays an important role in the running of an airport. Either taxis or passengers experiencing unexpected queue wait times for each other cause disruption and chaos for passengers and taxi drivers at the airport if either queue becomes too long. The queue management is an extensively studied topic in the operations research literature. The queueing theory is a mathematical model which can be used to predict the queue lengths and queue wait times (Little 1961). On the other hand, the queue context describes who is waiting in the queue in a given time window (i.e. taxi driver or passenger or both or none) (Yu et al. 2015). Hence, the queue context prediction is to predict different queuing situations given a specific time. Therefore, predicting different queue contexts at different times of the day can help to improve the airport satisfaction rating by providing timely taxi and passenger queue management. Also it can help the taxi drivers within the airport vicinity by providing timely information about the passenger queues in different passenger terminals.

Queue context prediction at the airport is a challenging problem for many reasons. First, both taxi and passenger queue formation is dynamic in nature due to the flight arrivals and indirectly due to weather conditions. Second, the queue contexts are imbalanced i.e. some of the queue contexts occur far less frequently than other queue contexts. The problem of queue prediction in terms of wait time has been studied in other applications (Davis et al. 2016), and various machine learning algorithms have been employed (Bulut et al. 2015; Zhang and Nguyen 2013). However, to the best of our knowledge no existing research predicts the queue context for two simultaneously occurring queues in the presence of imbalanced queue contexts.

In this paper, we extract and analyze various features and patterns for queue context prediction. We employ various sampling techniques and machine learning algorithms to predict the imbalanced queue contexts related to taxi and passenger. The reason for using machine learning techniques in queue context prediction is evident from the fact that the machine learning techniques have shown their effectiveness in terms of learning from the historical data and predicting similar patterns in new data. However, the choice of proper sampling and machine learning technique is crucial in our case. Specifically, we develop a framework that provides a step by step solution to predict the taxi and passenger queue contexts. Our experiments with the dataset generated for the JFK (John F. Kennedy) airport demonstrates the effectiveness of our approach. The contributions of this paper are as follows:

- Fusion of three real world heterogeneous contextual datasets for the research of taxi and passenger queue context prediction: taxi trip data, airport passenger wait time data and weather condition data.
- Extraction and analysis of contextual features and patterns from the airport queue context data;
- A framework to provide step by step solutions for imbalanced queue context prediction;

The organization of this paper includes a background of related literature section followed by a section for dataset design. Next, we formulate the problem and predict the queue context which is followed by a discussion of the experiments, results and analyses. Finally, we present our conclusion and suggestions for future work.

Background

The airport provides the first and last impressions of a city. The customer satisfaction rating of an airport depends on the proper management of both taxi and passenger queues. Several recent works analyze the demand-supply equilibrium of airport taxicabs (Anwar et al. 2013; Kamga et al. 2012). However, both taxi drivers and the passengers at the airports can experience long queue wait times (Hilkevitch 2015; Lin et al. 2016) for many reasons. The decisions of taxi drivers to make a trip to/from the airport have a direct connection to the occurrence of these queues (Yazici et al. 2013). Hence it is important to analyze and predict different states of both taxi and passenger queues at the airport.

The queue context describes the existence of any one of the four states of taxi and passenger queues which include ‘taxi queue only’, ‘passenger queue only’, ‘both taxi and passenger queues’ and ‘no queue’ (Yu et al. 2015). These queue contexts mainly influence the taxi drivers’ decisions to make airport trips which are lucrative for the taxi drivers. However, too many taxis at the airport taxi rank
may cause enormous waiting times for taxi drivers. On the other hand, the lack of taxis may cause long passenger queues. Considering these facts, many researchers have investigated citywide taxi trips with a view to providing recommendations for the taxi drivers and passengers.

The recent works focus on finding profitable cruising routes for passenger pickup (Dong et al. 2014; Ge et al. 2010). A profitable parking location analysis framework for taxi drivers was first proposed by (Yuan et al. 2011). Taxi GPS traces have been widely used to predict city-wide traffic conditions and to analyze the behavior and movement patterns of the population (Aslam et al. 2012; Castro et al. 2013; Moreira-matias et al. 2013; Liu et al. 2010; Zheng et al. 2009). A recommendation system for finding vacant taxis and passengers is presented by (Yuan et al. 2013). A spatial-temporal factor analysis model to find the best passenger pickup location is presented in (Hwang et al. 2015). The four factors considered by this model include distance, wait time, fare, and cluster transition probability. However, airports are often located in a designated area and supported by various transport modes. There is another research project aiming to estimate the passenger waiting time before a taxi ride which observes the behavior of vacant taxis (Zheng et al. 2012). A real-time taxi trip information system is proposed in (Balan et al. 2011) where passengers are able to know their estimated trip time and fare before their trip. A technique presented in (Song et al. 2014) recommends pickup points to avail a taxi ride. A passenger wait time prediction model from historical taxi trajectories is presented in (Qi et al. 2013). A context aware system for spatio-temporal traffic prediction in different road segments is proposed in (Xu et al. 2015). A system for monitoring taxis at a pickup location by mining GPS trajectories is presented in (Wu et al. 2012). A taxi and passenger queue context detection framework is presented by (Yu et al. 2015). There are very few works that deal with managing airport taxi operations. In (Yazici et al. 2013), logistic regression is used to model the taxi drivers’ next pickup decision for an airport trip. The model uses binary decisions of ‘airport pick-up’ or ‘cruising for customers’ at the end of each trip. However, these techniques are effective for citywide taxi operations and cannot be applied directly to predict airport taxi or passenger queue contexts.

On the other hand, research projects related to the airport taxi operations mainly focus on taxi queue modeling (Anwar et al. 2013; Zhang et al. 2013) to direct taxi drivers to the terminal with passenger queues. Airport taxi and passenger queue context prediction is different and challenging since queues at airports form dynamically. Also, the airport taxi and passenger queue context data suffer from the imbalanced queue context problem which compromises the learning performance. Different sampling techniques can be used to deal with this issue (He and Garcia 2009).

Dataset Design

We choose the JFK (John F. Kennedy) International Airport in New York City as our case location to prepare our taxi queue context dataset. The JFK is one of the busiest airports in the U.S. The central taxi holding (CTH) area at the JFK is far away from the passenger terminals. First, taxis need to join the waiting queue at the CTH area before picking a passenger up. The taxi dispatch managers at the JFK are responsible for dispatching taxis from this CTH based on the demand in different passenger terminals. In this study, we prepare the taxi queue context dataset for the JFK airport by adapting a state of the art queue context inference algorithm in our scenario after fusing three real world datasets from the New York city: i) the taxi trip data, ii) the JFK airport passenger wait times data and iii) the JFK weather condition data. Note that for the sake of simplicity, we limit our study within these three datasets. However, the integration of more contextual datasets (e.g. traffic condition data, public transport usage data) together with our three datasets could provide more insights about the problem domain.

Taxi Trip Logs

The first dataset is a real taxi trip dataset collected from the Taxi & Limousine Commision (TLC-NYC 2015) of New York City (NYC). In NYC, 13 thousand taxis generate 0.5 million trips on an average per day totaling 175 million trips per year. Each record in this dataset represents one taxi trip. A taxi trip record is described by its start and end geo-location with corresponding time-stamps, trip distance, passenger count, fare type and mount, tip amount, and taxi’s medallion number. In this paper, we process all the taxi trips made during the year of 2013 in the NYC. To infer the taxi and passenger queue contexts, it is required to estimate the taxi arrival rate and job wait time in a given time window. However, we cannot estimate the taxi arrival rate and job wait times directly for all the taxi trips that start from the JFK airport. The reason is evident from the fact that a large volume of empty taxis arrives at anytime without any pre-booking requests. These trips with no passengers are not stored in the taxi trip dataset and the taxi arrival times are unknown.
To overcome this problem, we rely on the survey results obtained from the recent literature. One paper reveals that experienced local drivers use their own expertise to choose nearby parking places to wait for their next passenger pickup rather than cruising randomly after a passenger drop-off (Yuan et al. 2011). Motivated by this fact, we choose those taxi drivers who join the airport taxi rank for their next passenger pickup after a subsequent passenger drop-off at the airport. Another reason to choose these taxis is that we know their arrival and departure times to and from the passenger pickup queue. We assume that these trips are able to provide insights about the average taxi waiting times in a time slot. Note that we consider the trips that start or end at the JFK airport for our experiments in this paper. To prepare our airport taxi trip dataset $Ax[]$, we separate these trips considering the latitude/longitude bounding box for the JFK airport figured out by (Whong 2014) in terms of $(\text{minLat}, \text{maxLat}, \text{minLong}, \text{maxLong})$. We design Algorithm 1 and Algorithm 2 to compute the hourly average taxi queue wait times and passenger pickup rates respectively which are used to infer queue contexts later in this paper. For calculating the average taxi queue wait times and passenger pickup rates, any hourly time window is selected. Then all the passenger drop-off times of the airport taxi trips within that hour are stored to calculate the time difference with their next passenger pickup times. Similarly, hourly passenger pickup rate is calculated considering all the taxi trips that initiated from the airport within that hour. Note that the taxi trips that only started from the airport without a precedent airport drop-off are also considered in this case.

```c
/* $t_i$ and $t_j$ are the start and end time of the time window 
* Input: an hourly time window of a day $(t_i, t_j)$, $Ax[]$ 
* Output: Hourly average taxi queue wait time $\bar{\tau}$ 
* Initialize: $\text{trip\_count}=0$; 
* Procedure hourlyAverageTaxiQueueWaitTimes($Ax[], t_i, t_j$)
for each $tx \in Ax[]$ do 
    if $t_i < tx.\text{Trip\_end\_date\_time()} < t_j$ then 
        if $\text{minLat} < tx.\text{Next\_Trip\_start\_lat()} < \text{maxLat}$ and 
            $\text{minLong} < tx.\text{Next\_Trip\_start\_long()} < \text{maxLong}$ then 
            $\text{Tarr} \leftarrow tx.\text{Trip\_end\_date\_time()}$; 
            $\text{Tdep} \leftarrow tx.\text{Next\_trip\_start\_date\_time()}$; 
            $w \leftarrow \text{Tarr} - \text{Tdep}$; // taxi queue wait time 
            $\text{trip\_count}++$; 
        end if 
    end if 
end for 
Return $\bar{\tau} \leftarrow \text{sum}(w)/\text{trip\_count}$; 
end procedure 
```

**Algorithm 1. Computation of Hourly Average Wait Time**

**Airport Passenger Wait Time Data**

The passenger wait time data is available through and collected from (Customs & Border Protection 2015). This dataset provides information about passenger wait times at different U.S. airports. Additional features include hourly numbers of flight and passenger arrivals with the number of passengers processed at the passenger processing booths. In our study, we consider the hourly passenger wait times at different terminals at the JFK airport.

**Weather Condition Data**

The weather condition data for JFK airport is collected from an online weather website (WeatherUnderground 2015). This dataset provides historical weather information including precipitation, temperature, wind speed, dew point, weather events (e.g., normal, rain, snow, rain-snow) and weather conditions (e.g., clear, overcast, mostly cloudy).
Predicting Imbalanced Taxi and Passenger Queue Contexts

Input: an hourly time window of a day \((t_i, t_j)\), \(Ax[]\)
Output: Hourly passenger pickup rate \(\rho\)
Initialize: pick_count=0;
procedure passengerPickupRate\((Ax[], t_i, t_j)\)
for each \(tx \in Ax[]\) do
  if \(t_i < tx. \text{Trip\_start\_dateTimes()} < t_j\) then
    pick_count++;
  end if
end for
Return \(\rho \leftarrow \frac{\text{pick\_count}}{|(t_i, t_j)|}\);
/*\(|(t_i, t_j)|\) is the length of time window \((t_i, t_j)\) in minutes.
end procedure

Algorithm 2. Computation of Hourly Pickup Rate

Data Fusion and Feature Extraction

At this stage, we fuse the NYC taxi trip data, the airport passenger wait time data and the weather condition data together after extracting corresponding features based on an hourly time window:

NYC taxi trip data: The extracted features include hourly taxi queue wait time, passenger pickup and drop-off frequencies, passenger pickup rates, and frequency of taxi trips that start subsequent to an airport passenger drop-off.

Passenger wait time data: The features include the frequency of hourly flight and passenger arrivals, number of passenger processing booths and hourly average passenger wait times at the passenger processing booths.

Weather condition data: The features are hourly precipitation, temperature, dew point, wind speed, humidity, and weather condition.

Note that all these features correspond to the current hour and we extract and compute the values of the similar features in the previous and next hours as well. Then all these features extracted from the three real world datasets are fused together for our experiment and analysis. Figure 1 illustrates the data fusion and extracted features from the final queue context dataset.

Queue Context Inference

Next, we infer the hourly queue context for each record corresponding to an hour in our dataset. Each record is labeled as one of the four queue contexts defined in (Yu et al. 2015) known as ‘taxi queue only’, ‘passenger queue only’, ‘both taxi and passenger queues’ and ‘no queue’. We denote these
contexts as ‘TQ’, ‘PQ’ , ‘TPQ’ and ‘NoQ’ respectively in this paper. For more clarification, the description of these four queue contexts is presented in Table 1. For the task of queue context inference, we adapt the Queue Context Disambiguation (QCD) algorithm proposed by (Yu et al. 2015).

<table>
<thead>
<tr>
<th>Queue Contexts</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TQ</td>
<td>Only taxis are in the queue and waiting for passengers.</td>
</tr>
<tr>
<td>PQ</td>
<td>Only passengers are in the queue and waiting for taxis.</td>
</tr>
<tr>
<td>TPQ</td>
<td>Both taxis and passengers are in the queues and waiting for each other.</td>
</tr>
<tr>
<td>NoQ</td>
<td>No one is waiting in the respective queues.</td>
</tr>
</tbody>
</table>

Table 1. Description of Four Queue Contexts

The QCD algorithm performs well with the taxi GPS trip traces and taxi MDT (mobile data terminal) log data. However, the QCD algorithm relies on two core assumptions. First, a time slot is labeled as ‘taxi queue only’ when the average waiting time (\(\bar{t}\)) of taxis is more than a waiting time threshold (\(\tau\)). Note that this is not valid for the time slots during the airport off-peak hours starting from 02:00 AM until 05:00 AM in our case. Some of the drivers willingly join the taxi waiting queue during this time and wait till the arrival of the first flight in the early morning. These drivers experience long wait times which do not constitute the existence of a taxi queue context. Therefore, we remove off-peak hours from our dataset and set \(\tau=90\) minutes for inferring the taxi queues.

Second, a time slot is labeled as the ‘passenger queue only’ when the taxi arrival rate of empty taxis or overall passenger pickup rate (\(\rho\)) is very high. The arrival rate indicates the number of taxi arrivals per minute while the pickup rate indicates the number of passenger pickups by taxi per minute. However, these assumptions cannot be used directly in the scenario of an airport. The most common reason is that a large volume of empty taxis arrives at anytime without any pre-booking request from the airport passengers. Also, a passenger queue can be identified when there is a low passenger pickup rate due to a shortage of taxis at times of very high demand. Unlike the QCD algorithm, we use a new threshold pickup rate which combines the upper (\(\rho\)-up) and lower (\(\rho\)-low) bounds of the pickup rate. This new threshold labels a time slot as ‘passenger queue only’ when \(\rho\)-up < \(\rho\) < \(\rho\)-low. Note that this assumption is not valid for airport off-peak hours since naturally there are very few or no passenger pickup events are observed. Therefore, we set \(\rho\)-low = 2.5 and \(\rho\)-up = 8 in the time slots except airport off-peak hours to infer the slots with passenger queues.

In the final dataset, each record corresponds to an hourly time stamp. A timestamp in this queue context dataset is described by the extracted features from the three real world dataset discussed previously in the Data Fusion and Feature Extraction Section. We also consider additional features that include the hour of day, and the day of week. In total each record in described by 44 features and one of the four queue context labels.

Queue Context Prediction

In this section, we first define the problem of imbalanced taxi queue context prediction. Then we propose a step by step solution to find the best sampling technique and prediction algorithm.

**Problem Definition**

Let us assume \(C_0 = \{TQ, PQ, TPQ, NoQ\}\) be the set of four possible imbalanced queue contexts and each sample instance (time slot), \(x\) in the training data be described by a \(d\)-dimensional vector of attributes \(R^d\) and a queue context label \(c(x) \in C_0\). Therefore, the instance \(x\) can be written as \(<a_1(x), a_2(x), ..., a_d(x), c(x)>\). If \(f(.)(\cdot)\) is the queue context prediction function then for a set of \(d\)-features corresponds to a query time slot \(x_q\), \(f(.)(\cdot)\) predicts \(\tilde{c}(x_q)\) such as \(f(x_q):R^d \rightarrow \tilde{c}(x_q)\) where \(\tilde{c}(x_q)\) is the predicted queue context of the query time slot \(x_q\).

**Preliminary Analysis**

We first conduct a preliminary analysis that shows the proportion of different queue contexts. From Figure 2(a), we can see that the proportion of NoQ context is very high compared to other three which may lead to a poor prediction performance. On the other hand, Figure 2(b) illustrates the cumulative distribution function (CDF) of taxi queue wait times. We can see that the taxis wait in the queue for...
more than one hour in 60% of instances. Another analysis from Figure 2(c) shows that about in 60% of cases the pickup rate ($\rho$) indicates an existence of passenger queues since passenger queues exist when $\rho$-up < $\rho$ < $\rho$-low. This is the reason why it is so important to provide an accurate prediction for the taxi and passenger queue contexts for smooth running of the airport. Figure 2(d) shows the hourly ratio of four queue contexts. We can see that the ratio of $TQ$ dominates the other three queue contexts during the morning peak hours and $PQ$ is the minority queue context during this time. As we approach the afternoon, we can see that the $PQ$ and $NoQ$ become the majority where $TPQ$ is the minority context of all. Note that the off-peak hours are dominated by $NoQ$ context. The reason is that we removed the off-peak hours from our queue context dataset assuming that no queues occur during this time.

![Figure 2. (a) Proportion of Queue Contexts. (b)-(c) CDF of Taxi Wait Times and Passenger Pickup Rate (d) Hourly Proportion of Queue Context Ratio at the John F. Kennedy (JFK) airport](image)

We further conduct an analysis from the perspective of time. We observe the three heat maps in Figure 3 that represent the hourly taxi wait times, passenger pickups and passenger arrivals. Here the red color indicated higher values while blue color indicates lower values. We can see from Figures 3(a) and 3(c) that the taxis mostly wait longer wait times between 03:00 and 13:00 while a large number of passenger arrivals is seen between 11:00 and 22:00. This clearly indicates that a major portion of the taxi drivers sit idle in the airport while waiting for a passenger. Also, it is clear from Figure 3(b) that only a small portion of these taxi drivers are able to pick up a passenger between 05:00 and 09:00 in the morning. This clearly indicates the taxi demand-supply imbalance in most of the occasions during the year of 2013 at the JFK airport. Hence it is very important from the perspective of passengers, taxi drivers and airport ground transport managers to predict the queue contexts by analyzing various queue context features.

![Figure 3. Heat Maps of Hourly a) Taxi Wait Times b) Passenger Pickup Frequency and c) Passenger Arrivals at the John F. Kennedy (JFK) airport. Note: x-axis represents the days of the year of 2013 and y-axis represent 24 hours of a day](image)

Furthermore, we analyze the Pearson's Correlation Coefficient between the hourly taxi queue wait times and all other features fused in to the queue context dataset from three heterogeneous real world datasets, including taxi trip, passenger wait time, and weather condition. Table 2 summarizes all features along with their corresponding Pearson's Correlation Coefficient. We observe a negative correlation with total flight arrivals (-0.30), total passenger arrivals (-0.26), total flight processing
booths (-0.23), and passenger pickup frequency (-0.49) in the previous hour. We also observe a negative correlation with passenger pickup frequency (-0.44) in the current hour. This implies that the more the flights, passenger arrivals, passenger processing booths and passenger pickup frequencies, the less the queue wait time for taxis.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Total passenger (previous hour)</td>
<td>-0.26</td>
<td>Passenger drop-off freq (next hour)</td>
<td>+0.09</td>
</tr>
<tr>
<td>Total passenger (current hour)</td>
<td>-0.09</td>
<td>Drop and pick freq (previous hour)</td>
<td>+0.02</td>
</tr>
<tr>
<td>Total passenger (next hour)</td>
<td>+0.13</td>
<td>Drop and pick frequency (curr. hour)</td>
<td>-0.07</td>
</tr>
<tr>
<td>Total flights (previous hour)</td>
<td>-0.30</td>
<td>Drop and pick frequency (next hour)</td>
<td>+0.04</td>
</tr>
<tr>
<td>Total flights (current hour)</td>
<td>-0.13</td>
<td>Temperature (previous hour)</td>
<td>-0.02</td>
</tr>
<tr>
<td>Total flights (next hour)</td>
<td>-0.10</td>
<td>Temperature (current hour)</td>
<td>+0.01</td>
</tr>
<tr>
<td>Total booths (previous hour)</td>
<td>-0.23</td>
<td>Temperature (next hour)</td>
<td>+0.03</td>
</tr>
<tr>
<td>Total booths (current hour)</td>
<td>-0.05</td>
<td>Dew point (previous hour)</td>
<td>-0.01</td>
</tr>
<tr>
<td>Total booths (next hour)</td>
<td>+0.16</td>
<td>Dew point (current hour)</td>
<td>-0.01</td>
</tr>
<tr>
<td>Passenger waiting (previous hour)</td>
<td>-0.15</td>
<td>Dew point (next hour)</td>
<td>-0.01</td>
</tr>
<tr>
<td>Passenger waiting (current hour)</td>
<td>-0.06</td>
<td>Wind speed (previous hour)</td>
<td>-0.03</td>
</tr>
<tr>
<td>Passenger waiting (next hour)</td>
<td>+0.12</td>
<td>Wind speed (current hour)</td>
<td>-0.01</td>
</tr>
<tr>
<td>Passenger pickup freq (previous hour)</td>
<td>-0.49</td>
<td>Wind speed (next hour)</td>
<td>+0.02</td>
</tr>
<tr>
<td>Passenger pickup freq in current hour</td>
<td>-0.44</td>
<td>Precipitation (previous hour)</td>
<td>-0.05</td>
</tr>
<tr>
<td>Passenger pickup freq (next hour)</td>
<td>-0.21</td>
<td>Precipitation (current hour)</td>
<td>-0.04</td>
</tr>
<tr>
<td>Passenger pickup rate (previous hour)</td>
<td>-0.45</td>
<td>Precipitation (next hour)</td>
<td>-0.04</td>
</tr>
<tr>
<td>Passenger pickup rate (current hour)</td>
<td>-0.42</td>
<td>Humidity (previous hour)</td>
<td>-0.01</td>
</tr>
<tr>
<td>Passenger pickup rate (next hour)</td>
<td>-0.17</td>
<td>Humidity (current hour)</td>
<td>-0.02</td>
</tr>
<tr>
<td>Passenger drop-off freq (previous hour)</td>
<td>+0.12</td>
<td>Humidity (next hour)</td>
<td>+0.01</td>
</tr>
<tr>
<td>Passenger drop-off freq (current hour)</td>
<td>+0.06</td>
<td>Taxi queue wait time</td>
<td>+1.00</td>
</tr>
</tbody>
</table>

Table 2. Pearson’s Correlation Coefficients

Queue Context Prediction Framework

In this section, we present our queue context prediction framework which provides a step by step solution for queue context prediction handling the queue context imbalance problem. Figure 4 illustrates different components of the queue context prediction framework.

First, data from different contexts (i.e., time, taxi trip, passenger, weather) are fused together to generate the taxi and passenger queue context dataset. Since this dataset suffers from the queue context imbalance problem, to overcome this we employ various sampling techniques:

- **Oversampling (OS)** adjusts the class (i.e. queue context) distribution of a dataset by increasing the number of minority classes.
- **Under sampling (US)** adjusts the class (i.e. queue context) distribution of a dataset by decreasing the number of majority classes.
- **Joint Sampling (JS)** adjusts the class (i.e. queue context) distribution of a dataset by simultaneously increasing and decreasing the number of minority and majority classes respectively.
- **No Sampling (NS)** No adjustment of the class (i.e. queue context) distribution is performed.

Then the sampled dataset is separated into two parts: train (60%) and test (40%) to observe the performance of the classifier suite. Different evaluation metrics are applied to find the best sampling technique and the best set of prediction algorithms. Finally, we analyze the in-depth prediction performance from two points of views: the taxi driver and the airport ground transport manager.
Experiments and Results

We evaluate different sampling techniques applied to our queue context dataset. We start with no sampling (NS) of the dataset. Then we employ under sampling (US), two variants of joint sampling (JS1 and JS2) and oversampling (OS). Note that in our queue context dataset, the ‘NoQ’ and ‘TPQ’ contexts are the majority and minority queue context labels respectively along with two other queue context labels (i.e. ‘TQ’ and ‘PQ’) as illustrated in Figure 2(a). In the under sampling stage, we randomly under sample all other context labels up to the number of ‘TPQ’ context. During oversampling, we randomly select instances and repeat to increase their number until it becomes equal to the ‘NoQ’. On the other hand, we oversample ‘TPQ’ and under sample ‘NoQ’ and ‘PQ’ contexts up to the number of ‘TQ’ contexts in our first joint sampling (JS1). In our second joint sampling (JS2), we undersample ‘NoQ’ and oversample ‘TQ’ and ‘TPQ’ contexts up to the number of ‘PQ’ contexts. Then we pick one sampling technique at a time for our dataset and employ our classifier suite. The classifier suite contains 7 algorithms which includes the Naïve Bayes (NB), decision tree (J48), random forest (RF), decision table (DT), PART decision rule, support vector machine (SVM) and k-nearest neighbor (k-NN). We use the Weka (Hall et al. 2009) implementation of these classifier. Note that we choose $k=5, 10, 15, 20, 25$ to consider different variants of the k-NN. Then we evaluate these sampling techniques under different performance metrics. The metrics include the predictive accuracy, sensitivity, F-Score, area under the ROC curve (AUC), and the area under the precision-recall curve (AUPRC). Specifically, we note the best metric score given by the classifier suite under each sampling technique. Table 3 summarizes the maximum and minimum metric score produced by our classifier suite under different sampling techniques.

First, we observe the maximum predictive accuracy produced by our classifier suite under a specific sampling technique. We can see from Table 3 that the oversampling (OS) produces the maximum predictive accuracy over other sampling techniques. However, better prediction accuracy cannot reflect a good performance in our case. The reason is that the large number of majority queue context labels present in the dataset may degrade the prediction performance of minority queue contexts. Therefore, we further analyze the sensitivity of the prediction task. In binary classification, the sensitivity score tells us about how many relevant items are selected. Let us assume, there are two classes, ‘positive’ and ‘negative’. The number of positive instance classified as positive is called true positive (TP) and number of positive instances classified as negative is called false negative (FN). Then the sensitivity score is given by TP/ (TP+FN). Since we have four queue contexts, we take one as ‘positive’ and other three as ‘negative’ in turn and finally compute the average. In this way, we will be able to know the weighted average of prediction performance for minority classes under different sampling techniques. We can see from Table 3 that the oversampling (OS) gives the maximum sensitivity score. We also analyze the performance from three other perspectives. The F-score is the harmonic mean of sensitivity and precision where precision is a measure of the amount of retrieved instances that are relevant. The other two are the area under the ROC curve (AUC) and the area under...
the precision-recall curve (AUPRC). The AUC plots the true positives against false positives while the AUPRC plots the precision against the sensitivity. Table 3 shows that the oversampling outperforms other sampling techniques in our case since the oversampling applied to our classifier suite provides the largest performance score under all the performance metrics.

Next, we select the best classifiers from the classifier suite after applying the oversampling technique and analyzing the confusion matrix they produce. For selecting the best classifier we rely on the AUPRC values in our research. First, we take a base classifier and observe the significance of the AUPRC score of other classifiers. Specifically we perform paired t-test with a significance score of 0.05 between the AUPRC values of the base classifier and the other classifiers. In next iterations, we remove the previous base classifier from the classifier suite and choose a new base classifier that has the lowest AUPRC score. Note that we select this new base classifier after removing those classifiers with insignificant AUPRC score if there is any. After several iterations we get the support vector machine (SVM) and the Random Forest (RF) as the best two classifiers from our classifier suite. Note that the RF uses an ensemble learning technique that fits the training data into a number of decision tree classifiers to produce the final prediction. On the other hand, the SVM uses an imaginary hyper plane to discriminate between the training data to produce prediction for a query instance.

<table>
<thead>
<tr>
<th></th>
<th>NS</th>
<th>US</th>
<th>JS1</th>
<th>JS2</th>
<th>OS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Max</td>
<td>Min</td>
<td>Max</td>
<td>Min</td>
<td>Max</td>
</tr>
<tr>
<td>Accuracy</td>
<td>74.12</td>
<td>50.05</td>
<td>80.34</td>
<td>25.88</td>
<td>84.51</td>
</tr>
<tr>
<td>Precision</td>
<td>0.74</td>
<td>0.25</td>
<td>0.81</td>
<td>0.38</td>
<td>0.85</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>0.74</td>
<td>0.50</td>
<td>0.81</td>
<td>0.26</td>
<td>0.85</td>
</tr>
<tr>
<td>F-Score</td>
<td>0.73</td>
<td>0.33</td>
<td>0.81</td>
<td>0.18</td>
<td>0.84</td>
</tr>
<tr>
<td>AUC</td>
<td>0.90</td>
<td>0.50</td>
<td>0.95</td>
<td>0.51</td>
<td>0.97</td>
</tr>
<tr>
<td>AUPRC</td>
<td>0.81</td>
<td>0.35</td>
<td>0.88</td>
<td>0.25</td>
<td>0.92</td>
</tr>
</tbody>
</table>

Table 3. Performance Evaluation of Different Sampling Techniques

All these metrics discussed above treat the queue context prediction as a binary class problem which actually cannot provide the true picture of the prediction performance. Therefore, we analyze the confusion matrices produced by the classifiers from a multi-perspective point of view. Tables 4 and 5 represent the confusion matrices of prediction performances of SVM and RF respectively. The airport taxi-passenger queue manager is responsible for proper management of queue contexts related to taxi and passenger at the airport by considering the perspectives of both taxi drivers and passengers. Aiming to avoid long queues, the queue manager regulates incoming flow of the taxis at the taxi rank based on the predicted queue context in a future time slot. If any future time slot is predicted as ‘TQ’ or ‘NoQ’, no taxis should enter the taxi rank area to avoid long queue wait times. Moreover, the ‘TQ’ and ‘NoQ’ contexts should not be predicted as ‘PQ’ since in such cases more taxis will enter the taxi rank unnecessarily. We can see from Tables 4 and 5 (‘blue’ filled cells) that the SVM performs better compared to the RF in this regard and there are no actual ‘TQ’ and ‘NoQ’ instances which are classified as ‘PQ’ by the SVM in contrast to 15% of ‘NoQ’ instances predicted as ‘PQ’ by the RF. On the other hand, the ‘PQ’ instances should not be classified as ‘NoQ’.

We can see that the prediction error produced by the RF (0.02) is lower compared to the SVM (0.07).

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted As : ć(x_q)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NoQ</td>
<td>0</td>
</tr>
<tr>
<td>TQ</td>
<td>0.05</td>
</tr>
<tr>
<td>PQ</td>
<td>0.07</td>
</tr>
<tr>
<td>TPQ</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4. Confusion Matrix-SVM

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted As : ć(x_q)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NoQ</td>
<td>0.75</td>
</tr>
<tr>
<td>TQ</td>
<td>0.01</td>
</tr>
<tr>
<td>PQ</td>
<td>0.02</td>
</tr>
<tr>
<td>TPQ</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 5. Confusion Matrix-RF
Conclusion

In this paper, we address the queue context prediction problem in the presence of imbalanced queue contexts related to taxis and passengers at the airport. The taxi drivers want to avoid long queue waiting times at the airport taxi rank before a passenger pickup. On the other hand, the airport passengers expect a taxi as soon as they arrive at the terminal curbside. To help the airport ground transport managers in taking appropriate actions in these situations, we developed and explain our framework that provides a step by step solution to predict different contexts of the queues that are important to manage taxis and passengers at the airport. To support our research and to study and analyze the problem of queue context prediction, we integrate three real world datasets based on the JFK international airport in New York City.

Our proposed framework is able to identify the most effective prediction techniques for this problem and analyze their performance from two different points of views: the taxi drivers and the passengers. The SVM and the Random Forest are the two most effective predictor algorithms identified by our proposed framework. The experimental results using our framework show the effectiveness of our approach. We observe that the SVM performs better from the taxi drivers’ point of view while Random Forest shows better results from the point of view of the airport passengers to predict different queue contexts in a given future time stamp. The airport ground transport managers may use the combined outputs from these two prediction algorithms to manage taxi and passenger queues efficiently at the airport. In this way taxi drivers and passengers may not experience enormous waiting times in their respective queues at the airport. Also it may ease the problem of losing lucrative airport pickup jobs when taxi drivers mistakenly choose not to make an airport trip due to the lack of timely information regarding queue contexts.

The research presented in this paper predicts four different queue contexts. However, there is further scope for improvement by providing information about lengths of queues in real time along with these queue contexts. Also the inclusion of more contextual datasets such as traffic conditions and social events may enhance the queue context prediction performance. In summary, this paper provides a queue context prediction model which can be applied to any location not only airports but also shopping malls, ferry platforms with similar taxi regulations.

Acknowledgements

This research was supported under Australian Research Council's Linkage Projects funding scheme (project number LP120200305).

References


Balan, R. K., Khoa, N. X., and Jiang, L. 2011. “Real-Time Trip Information Service for a Large Taxi Fleet,” in The Int. Conf. on Mobile Sys., App., and Serv. (MobiSys’11), Bethesda, Maryland, USA.


