Data-driven Bus Crowding Prediction Models Using Context-specific Features

TAHEREH ARABGHALIZI, University of Pittsburgh
ALEXANDROS LABRINIDIS, University of Pittsburgh

Public transit is one of the first things that come to mind when someone talks about "smart cities." As a result, many technologies, applications, and infrastructure have already been deployed to bring the promise of the smart city to public transportation. Most of these have focused on answering the question "when will my bus arrive?"; little has been done to answer the question "how full will my next bus be?" which also dramatically affects commuters' quality of life. In this paper, we consider the bus fullness problem. In particular, we propose two different formulations of the problem, develop multiple predictive models, and evaluate their accuracy using data from the Pittsburgh region. Our predictive models consistently outperform the baselines (by up to 8 times).

CCS Concepts: • **Computing methodologies → Machine learning approaches**; **Model development and analysis**.

Additional Key Words and Phrases: smart city, intelligent transportation, urban computing, crowdedness prediction

ACM Reference Format:
Tahereh Arabghalizi and Alexandros Labrinidis. 2020. Data-driven Bus Crowding Prediction Models Using Context-specific Features. *ACM Trans. Data Sci.* 1, 1, Article 1 (January 2020), 35 pages. https://doi.org/10.1145/3406962

1 INTRODUCTION

The rapid growth of urbanization during the past decades is necessitating increased efficiency in city operations. This is manifesting as sensing technologies for data collection, advanced models and algorithms, and relevant data dissemination to city dwellers, whose lives these big data and technologies are ultimately trying to improve [1, 3, 13, 42]. Collectively these techniques are often referred to as "smart city" technologies.

A textbook example domain for a smart city technology is that of *public transportation*. Everybody who lives in a city would wish for public transportation to be "better." Problems such as bus delays, crowded buses, and general lack of public transportation options especially during rush hours make commuters dissatisfied and unhappy about the city's public services.

A plethora of technologies, applications, and infrastructure have been deployed already to bring the promise of the smart city to public transportation. These include GPS tracking of buses to reliably predict their arrival times, the standardization of transit schedule data [16], and mobile applications (e.g., Transit App [39] and MoovIT [25]) to make such transit information available in real-time to commuters.

Authors' addresses: Tahereh Arabghalizi, Department of Computer Science, School of Computing and Information, University of Pittsburgh, 210 S Bouquet St, 15213, tahereh.arabghalizi@pitt.edu; Alexandros Labrinidis, Department of Computer Science, School of Computing and Information, University of Pittsburgh, 210 S Bouquet St, 15213, labrinid@cs.pitt.edu.

Although a lot of work has been done towards figuring out the answer to the question "when will my bus arrive?", little has been done to answer the question "how full will my next bus be?" which also dramatically affects the commuters' quality of life. This is exactly the focus of this work.

1.1 Problem Statement

In order to effectively answer the *"how full will my next bus be?"* question, we take a multipronged approach. First of all, we explore *two different formulations* of the problem. Second, we develop *predictive models* for the different problem formulations. Third, we *evaluate the performance* of the different techniques using real data. Lastly, we consider the *real-life applicability* of the proposed problem formulations and predictive models.

When one considers the "how full will my next bus be?" question, there are two possible types of answers:

- **(i)** a specific number of people currently in the bus, which can be used to determine the number of seats/spots that would be available; we refer to this as *bus load*, or
- **(ii)** a "fullness" level that provides an approximate degree of how crowded the bus will be; we refer to this as *bus crowding level*.

These give us the following formulation to the fullness problem:

> *predict the bus load or the bus crowding level for a certain route arriving to a specific bus stop within a given 15-minute time interval.*

where a bus route is defined as a set of stops with a starting point, an end point, and a direction (inbound or outbound). Note that this formulation does not restrict the input variables for the prediction techniques; we can use additional features such as weather data, historical information, etc.

Clearly, the two problem formulations require different techniques for prediction (e.g., regression for bus load and classification for bus crowding levels) and different metrics for measuring performance. However, both problem formulations can lead to predictions that can help travelers make more informed decisions about which bus to take, while considering the quality of their trip instead of just the time.

We are investigating this exact trade-off as part of the *PittSmartLiving* project. In particular, we plan to design, develop, deploy, and evaluate a platform that will integrate information from and align the incentives of all involved stakeholders (commuters, mobility providers, and local businesses) towards increasing the utilization and quality of public transportation [33]. For example, while waiting at the bus stop, a commuter will receive a push notification alerting them to the next bus being full. In addition, they may also receive a discount towards coffee/tee at the coffee shop around the corner (say $2 off), if they would take a later bus. This would alleviate bus crowding and make everybody's bus ride a more pleasant experience.

1.2 Our Contributions

This work makes the following contributions:

1. We frame the "how full will my next bus be?" question as two different prediction problems: bus load prediction and bus crowding level prediction.
2. We formulate the bus load and bus crowding level prediction problems as intuitive regression and classification problems respectively and develop appropriate models for prediction (Section 3 and Section 4).

(3) We explain the extensive data preparation strategies employed over the real-world data received from the Port Authority of Allegheny County (Section 5).
(4) We perform experimental evaluation using real-world data and compare our proposed models to baseline models (Section 6 and Section 7). Although our analysis used data for the Pittsburgh area, our techniques are trivially generalizable to other areas.

2 RELATED WORK

The reliability of the public transportation system, in particular with regards to travel time and available space, greatly affects commuters' quality of life in urban travel [36]. Many research works have proposed techniques to predict bus arrival times, optimize planning and design customized bus (CB) systems that can provide more efficient transit services [5, 6, 11, 28–30, 37]. However, only a few previous studies have focused on predicting the space availability as a transit reliability issue. Some works like [40, 41, 46] have studied forecasting passenger flow in the whole urban public transit system by integrating regression analysis with time-series, neural networks and SVM respectively. On the other hand, former research such as [35] and recent ones including [24] focused on finding optimal bus capacity. Utilizing bus smart card data and GPS data is also another method that has been proposed by [43, 45]. Zhang Jun et al. [45] predicts the passenger flow in real time by finding the flow pattern based on the Extended Kalman Filter model.

Among the works about forecasting bus passenger occupancy, Gayah et al. [19] has the most resemblance to our research. They have developed regression models to predict the real-time passenger occupancy for each bus-stop. However, their work is limited to only one bus route with 15 stops serving the Pennsylvania State University (PSU) University Park campus. We believe that the characteristics of each bus route and stop can be very different from other bus routes and stops. Therefore, one predictive model cannot be applicable for all the routes at all stops. As a point of reference the network of the Port Authority of Allegheny County, serving the Pittsburgh Area, has almost 200 routes and 7,000 bus stops[1].

A just released feature of Google Maps is claimed to predict how full the bus will be for 200 cities world-wide [21, 22]. They rely on crowdsourced data along the lines of the Tiramisu project [38] from a few years ago. Although the Google Maps problem formulation is close to one of the two we consider in this work, their techniques suffer from a well-known sample bias. Their data is providing good coverage only where there are a lot of smartphone users (that also utilize Google Maps in their commutes). This can trivially lead to over- or under-reporting, as has been the case with other smart city projects in the past [15].

To the best of our knowledge there is no other work that considers the bus crowding level problem using real passenger count data (instead of crowdsourced "experience" data). We believe utilizing bus crowding levels to be a more intuitive way of sharing information with travelers (instead of bus load or passenger occupancy counts). Because public transit is inherently human-facing, utilizing predictive models that are intuitive is of paramount importance. At the same time, we believe using just crowdsourced data is not a reliable not equitable approach to predicting bus crowding levels.

Finally, to the best of our knowledge there is no other work evaluating different formulations of (and solutions to) the bus fullness question.

[1] https://www.portauthority.org/inside-Port-Authority/Transparency/system-data-and-statistics/

3 MODELING FRAMEWORK

To construct our modeling framework, we utilize the typical workflow that can be used to solve any machine learning problem as described in [12]. This process consists of 7 steps as follows:

(1) Defining the problem and assembling a dataset: Sections 1.1 and 5.1
(2) Choosing a measure of success: Section 6.1
(3) Deciding on an evaluation protocol: Section 3.4
(4) Preparing your data: Sections 3.2, 5.2 and 5.3
(5) Developing a model that does better than a baseline: Sections 4.1, 7.1 and 7.2
(6) Developing a model that overfits: Sections 4.2 and 7.3 to 7.12
(7) Regularizing your model and tuning your hyperparameters: Section 4.2

In this section, we also explain the details of our formulation for bus crowding levels, as well as the different variables to be used.

Table 1. Bus crowding levels

Level	Description	Condition
CL1	many seats available	Load Factor <0.5
CL2	a few seats available	0.5 <= Load Factor <0.8
CL3	a few people standing	0.8 <= Load Factor <1.1
CL4	many people standing	1.1 <= Load Factor <1.4
CL5	crushed	Load Factor >= 1.4

3.1 Definitions (Table 1)

As mentioned in the previous section, we consider two formulations of the bus fullness problem:

(i) prediction of *bus load*, i.e., number of passengers in the bus, or
(ii) prediction of *bus crowding level*, i.e., a characterization of how full the bus is.

Zheng Li et al. [27] has reviewed the specifications of crowding measures in public transportation in different countries. As stated in their research, many US transit authorities utilize Load Factor (as the number of passengers divided by the number of seats) to evaluate in-vehicle crowding. Accordingly, we define the *Load Factor* of bus$_i$ as the ratio of the number of current passengers on that bus to its maximum seating capacity, i.e.,

$$LoadFactor_i = \frac{number\ of\ current\ passengers\ on\ bus_i}{maximum\ seating\ capacity\ of\ bus_i} \tag{1}$$

Given the above definition, a Load Factor value of 1.0 means that there are as many passengers in the bus as seats, whereas a value of 1.2 means that there are 20% more passengers in the bus than seats[2]. The Transit Capacity and Quality of Service Manual [2] from the Federal Transit Administration defines the thresholds for the level of service with respect to the load factor (e.g. load factor > 1.5 represents the crush loading level). In this work, we define our own five crowding levels (explained in Table 1) which were developed after many discussions with the Port Authority of Allegheny County. We used 1.4 load factor as the threshold for over-capacity (CL5) because this is an industry best

[2]It is worth noting that modern automatic passenger counting systems cannot determine how many people are seated versus how many people are standing, just how many people entered or exited the bus.

Table 2. Descriptions of the independent variables used in our models

Variable	Description
TOD	96 variables for time of the day (each 15-min time-interval)
DOW	5 variables for day of the week (only weekdays)
MOY	12 variables for month of the year
BusType	one variable (if the bus is single or double, i.e., articulating)
Temperature	one variable for average temperature per hour
Rainfall	one variable for average rainfall per hour
Snowfall	one variable for average snowfall per hour
PLoad	10 variables for bus loads in the 10 previous stops. PLoad1 is the bus stop immediately before the one we are predicting for.
Stop	N variables for stops. N is the number of stops for a route in one direction. Stop variables are only used in models with route-direction data inputs.

Table 3. Feature Sets to be used in models. The last row is only used for models with route-direction input data.

Features	FS1	FS2	FS3	FS4	FS5	FS6	FS7	FS8	FS9
TOD2 - TOD96	✓	✓	✓	✓	✓	✓	✓		
DOW2 - DOW5		✓	✓	✓	✓	✓	✓		
MOY2 - MOY12		✓	✓	✓	✓	✓	✓		
BusType		✓	✓	✓	✓	✓	✓		
Temperature, Rainfall, Snowfall		✓	✓	✓	✓	✓	✓		
PLoad1 - Pload5			✓					✓	
PLoad1 - Pload10				✓					✓
PLoad5					✓				
PLoad10						✓			
PLoad5 - Pload10							✓		
$Stop_2$ - $Stop_N$	✓	✓	✓	✓	✓	✓	✓	✓	✓

practice and something the Port Authority is using in its own reporting. The other levels are determined based on the seating and standing availability where all passengers can sit (CL1 and CL2) or some/many passengers need to stand (CL3 and CL4).

3.2 Feature Selection (Table 2 and Table 3)

We have carefully considered all possible input features (independent variables) that can impact bus fullness to include in our models. These context-specific Features include time of the day (TOD), day of the week (DOW), month of the year (MOW), bus type, reflecting if the bus is articulating (i.e., double) or not, and weather conditions (i.e., temperature, rainfall, snowfall). We have also considered the bus load in the previous stops, as measured by automated passenger counters, assuming such signals would be "live". Some of the features are categorical which were converted to dummy variables and some are numerical. Descriptions of the independent variables[3] are provided in Table 2.

As part of our evaluation of the different models, we want to have a broader perspective of the performance of the models in connection to the different sets of features chosen. Towards this, we defined nine combinations of features

[3]The number of dummy variables for each categorical feature is later reduced by one in order to be used in the models.

Table 4. Selected route-stop pairs for modeling

Cluster	Route	Stop Name
CL1	61B	FIFTH AVE AT BIGELOW BLVD
	71D	FIFTH AVE AT THACKERAY AVE
CL2	12	ANDERSON ST AT GENERAL ROBINSON
	56	GREENFIELD AVE AT IRVINE ST
CL3	Y1	E CARSON ST OPP STATION SQUARE STATION
	28X	LIBERTY AVE AT GATEWAY 4
CL4	P1	EAST BUSWAY AT NEGLEY STATION A
	G31	WEST BUSWAY AT INGRAM STATION C
CL5	P1	EAST BUSWAY AT NEGLEY STATION C
	61C	FORBES AVE AT BEELER ST

(Feature Set 1, ..., Feature Set 9) to identify which ones perform best in our evaluation. These sets and their selected features are represented in Table 3.

3.3 Route Sampling (Table 4)

There are close to 100 different routes on two different directions (inbound and outbound) per route and about 7,000 stops in Pittsburgh's bus transit system. In order to conduct simpler experiments before running the proposed models for all the routes in Pittsburgh, we define and apply a clustering method for partitioning routes and then select the top two routes from each cluster as representatives of all routes in that cluster.

In order to identify representative routes, we partitioned routes in the dataset using their "most common" crowding levels. That gave us five different clusters of routes that can be defined as follows (using the same names as each corresponding crowding level):

$$CL_i : routes\ in\ cluster_i\ whose\ most\ common\ crowding\ level\ is\ CL_i\ ,\ where\ i \in \{1, 2, 3, 4, 5\}$$

For each of the five CL_i clusters, we selected the top two routes, that have the highest number of records, after normalizing by the number of stops to account for differences in the number of stops among routes. These routes include 61B, 71D, 12, 56, Y1, 28X, P1 (which appears in two clusters), G31 and 61C.

For some of our proposed models, we build a model for each route at a particular stop, that is why we also need to specify one stop for each selected route. You can see the list of selected routes-stops in Table 4.

3.4 Training and Testing Phase

The primary goal in a Machine Learning process is basically creating a model to make prediction using the test data. Therefore, we use a subset of the available data as the training set to fit the model and the remaining data as the testing set to test it. The generated models predict the results using unknown data which is named as the test set. In our experimental setup, we randomly selected 80% of each preprocessed and transformed dataset to be used as our training data. The remaining 20% of each dataset was used as test data for evaluation. This split is done using the Python Scikit-Learn library and specifically the train_test_split() method. More information about the data preparation procedures and the challenges we faced are discussed in Section 5.

4 PROPOSED MODELS

In this section we introduce our proposed models plus the baseline models that we used for our evaluation. Our goal is to predict bus load and bus crowding levels; towards this, we employ two types of machine learning models including Regression and Classification with two different types of data inputs namely route-stop and route-direction (see Table 5).

Table 5. Sections Related to Proposed Models

Models \ Data Inputs	route-stop	route-direction
Regression	4.1 (a)	4.1 (b)
Classification	4.2 (a)	4.2 (b)
Baseline	4.3 (a)	4.3 (b)

Having predictive models for each route-stop pair was our very first approach which resulted in good accuracy but with the price of building plenty of models (~12,000). We then came up with a new approach to reduce the total number of predictive models while improving their accuracy. In the new approach, we build a model for each route in each direction (inbound and outbound) while adding the route's stops as new *independent variables* to the existing ones. In addition to the number of predictive models decreasing from near 12,000 to less than 200, we also found that the accuracy of the models increases based on our experiments.

Furthermore, after running different experiments (see section 7), we decided to build predictive models for all the routes using classification with the route-direction approach. Although the obtained results from the regression models completely correspond with the results from the classification models, we choose the latter which is more understandable for the real-world. We firmly believe that getting a descriptive "crowding level", as we propose in this work, is much more intuitive than a "number of seats remaining" count, especially since the number of seats is often less than the total number of people in the bus.

4.1 Regression Models

Our first problem formulation is to predict the bus load, i.e., the number of passengers on a bus. In the modeling framework for this problem, the dependent variable of interest is a numerical count. As such, we need to rely on Regression Models with Count Data. There are several count data models among which the Poisson, Negative Binomial, and Zero-inflated are the most popular. The Poisson regression model often fits the data poorly, because it assumes that the conditional variance of the dependent variable is equal to the conditional mean while in most count data sets, the conditional variance is greater than the conditional mean [9]. A Zero-inflated model should be considered when analyzing a dataset with an excessive number of outcome zeros and two possible processes that arrive at a zero outcome [18]. This case also does not apply on our data because we do not have an excessive number of zeros in load due to different processes. Therefore, it seems that fitting a conventional ***Negative Binomial Regression Model*** is an appropriate predictive model for our data. As mentioned, we build our models with two different data inputs as follows:

(a) Route-Stop : In this approach, we have a separate dataset for each route at each stop, fit each Negative Binomial model with its relevant training data and then predict the load for each data record in the relevant test set.

(b) Route-Direction : In this approach, we filter out data for each route in each direction (inbound and outbound) from the main dataset and use the relevant stops as independent variables in each Negative Binomial model. After fitting each model with its relevant training dataset, we can predict loads using its test set.

4.2 Classification Models

As described earlier, we view predicting bus crowding levels to be a multinomial classification problem. Given the set of independent variables we have, we employed ***Logistic Regression***, ***Artificial Neural Network*** and ***Random Forest*** algorithms. Thus, for each dataset we fitted a separate classifier using the relevant training set and then predicted the crowding level using the test set. Most of our experimental evaluations utilize Random forest Classifier because it produces more accurate predictions, limits over-fitting and therefore yields more useful results [8]. However, for completeness we also report results with Logistic Regression and Artificial Neural Network in sections 7.3 and 7.4.

The following are the two data inputs used for classification models:

(a) Route-Stop : In this approach, we have a separate dataset for each route-stop pair, fit each classification model with its relevant training data and then predict the crowding level for each data record in the relevant test set.

(b) Route-Direction : In this approach, we filter out data for each route in each direction from the main dataset and then use the relevant stops as independent variables in each classifier. After fitting each model with its relevant training dataset, we can predict crowding levels using its test set.

Hyperparameter Tuning

Logistic Regression: The Logistic Regression class in Python offers two regularization schemes (L1 and L2) and four optimizers: newton-cg, lbfgs, liblinear, and sag [26]. Among these, newton-cg with L2 regularization produced models with higher prediction accuracy.

Neural Network: The first step before building a neural network is to normalize the data and change the values of features to a common scale (using StandardScaler or MinMaxScaler in Python). The next step is to build the neural network using a tool such as Keras which is a high-level framework based on Tensorflow [17]. We also need to specify the number of hidden layers and their size (number of neurons), the input and output size. In our case, the number of output neurons is fixed for all datasets and is equal to the number of crowding levels. The number of input neurons, however, varies from one dataset to another because the number of features varies depending on the route or data input. We add one hidden layer to each network (adding more layers did not improve the models) that consists of different number of neurons depending on the dataset. There are some empirically-derived rules-of-thumb to determine the optimal size of the hidden layer. Heaton et al. [23] introduces a few of these such as the following equation that works the best for our case:

$$N_h = \frac{N_s}{(\alpha * (N_i + N_o))} \tag{2}$$

where N_i is the number of input neurons, N_o is the number of output neurons, N_s is the number of samples in training data set and α is an arbitrary scaling factor usually between 2 and 10.

Since there are more hyperparameters involved in the Keras framework, we use a grid search technique to find their best values. we achieved the best results with a batch size = 64, epochs = 100, dropout rate = 0.5 and an Adam optimizer while training a Sequential model in Keras. Thus, we use these tuned values in all neural networks and then fit and

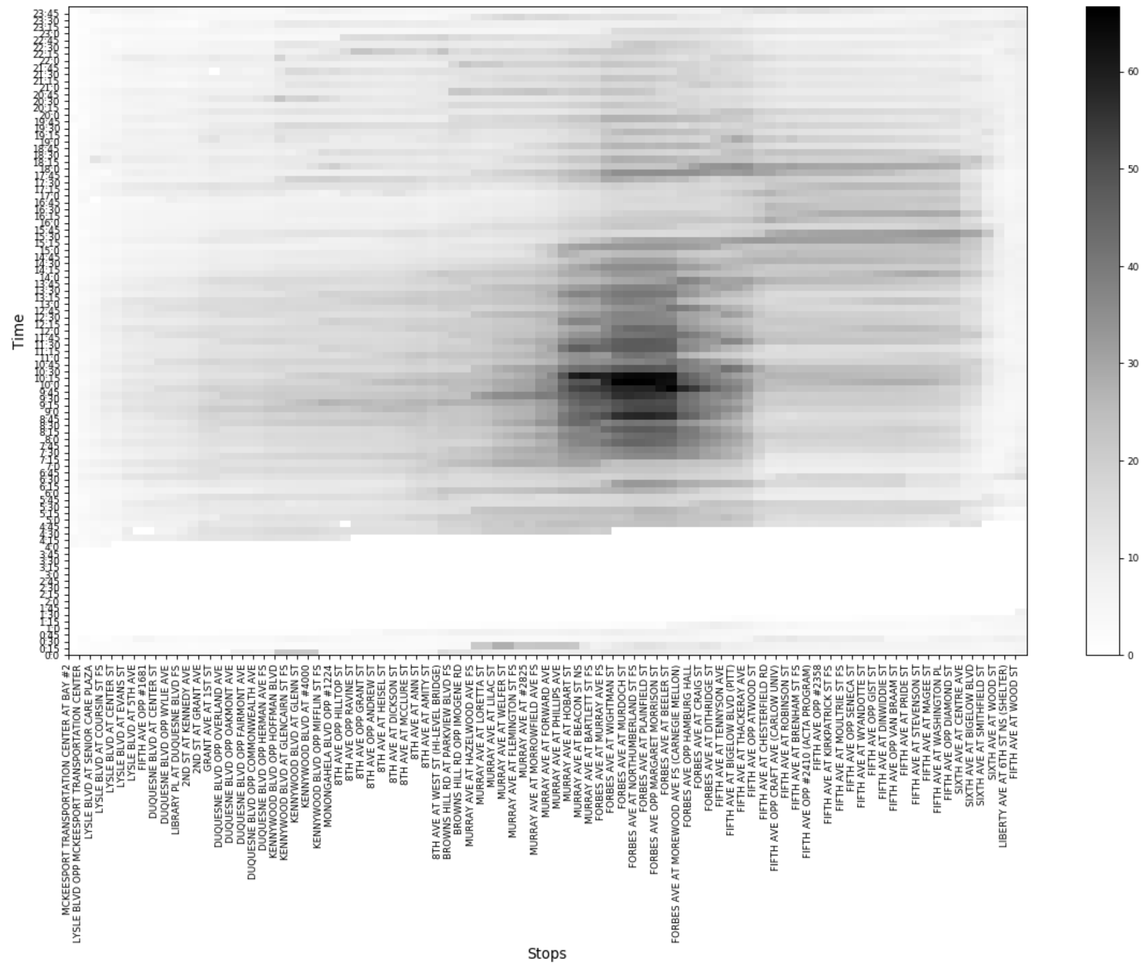


Fig. 1. The Spatiotemporal average load for 61C-inbound using the route-stop baseline data. The x-axis represents 61C's stops in geographical order, the y-axis shows 15-minute time intervals of the day, and the color scale indicates the value of the average load for the corresponding bus stop, time of day combination.

evaluate each model after randomly selecting an 80-20 split for the training-validation sets and an 80-20 split for the training-test sets. It is noteworthy that we use "softmax" as the activation function and "categorical crossentropy" as the loss since we are dealing with multinomial classification.

Random Forest: In order to reach the highest accuracy in Random Forest Classifiers, we conducted a grid search on three desired hyperparameters including the number of trees in the forest, the function to measure the quality of a split and the maximum depth of the tree. The optimized values for the first two hyperparameters obtained by the grid search were 500 trees and "entropy" respectively while the optimized value for the third hyperparameter was the same as the default value of this parameter in the Python scikit-learn's Random Forest Classifier [26].

4.3 Baselines

We need different baselines corresponding to the introduced models. The simplest model that we can propose as a baseline to be compared with a regression model is a model that includes average loads of a route at a specific stop/direction within a 15-minute time interval. On the other hand, for having a comparison with a classification model, we also need to assign a crowding level to each route at a specific stop/direction within a 15-minute time interval which can be computed using the load factor. Our baseline models are listed as below:

(a) Route-Stop : In these baselines, we obtain the average load and crowding level for every route at each related stop, for every 15-minute interval of a day. To have a better perception of this type of baselines, we present an example here. Figure 1 illustrates the average load, obtained from the baseline, for 61C which is one of the busiest routes in Pittsburgh. In this heatmap, the x-axis represents 61C's stops in geographical order only in one direction (inbound, i.e., to downtown), the y-axis shows 15-minute time intervals of the day and the color scale indicates the value of the average load for the corresponding bus stop, time of day combination. As one can see, the average load dramatically increases during the rush hours in the morning between 7 and 10 at some specific stops in Oakland where the University of Pittsburgh campus is located. It is not surprising because many University of Pittsburgh students, faculty, and staff take this route and the similar ones to get to campus in the morning.

(b) Route-Direction : Each baseline in this category is obtained by computing the average load and crowding level for every route at each related stop in each direction, for every 15-minute time span. This baseline will be used for evaluation of our regression and classification models with route-direction data inputs.

Table 6. Statistics about the Pittsburgh area bus data

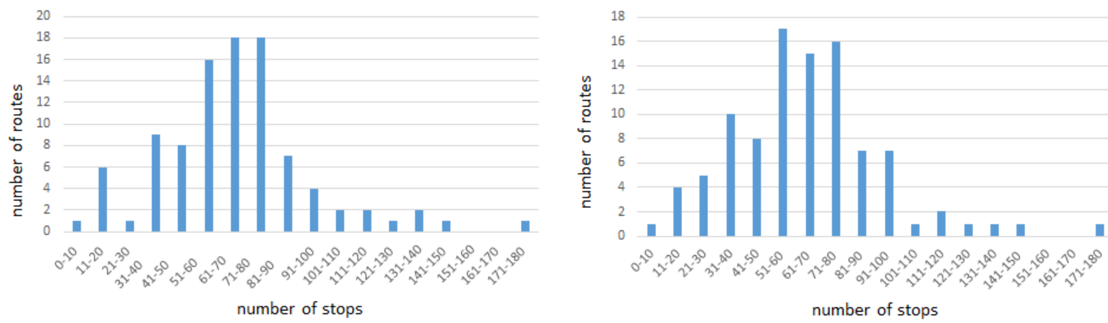
	First Dataset	Second Dataset
Duration	March 2017 - March 2018	June 2018 - June 2019
number of routes (in-/out-bound)	98*2=196	98*2=196
number of stops	6,923	6,876
number of records in dataset before cleaning	100,869,765	102,809,399
number of records in dataset after cleaning	89,901,555	91,807,584
number of columns in dataset	189	215
number of useful columns	18	18

5 DATA PREPARATION/CHALLENGES

We have received two types of Pittsburgh-area bus data from the *Port Authority of Allegheny County*:

- **Schedule Data** are given in GTFS format [16]; these contain the published bus schedules (i.e., are equivalent to printed bus schedules).
- **Historical Data** are given in a STEP [4] file format; these contain data about the exact time each bus arrives at a bus stop, along with how many people board or alight the bus. These data are generated using automated people counting devices that are mounted at the doors of every bus. We convert the STEP file to a text standard format like CSV [5].

[4]STandard for the Exchange of Product
[5]Comma Separated Values



(a) inbound

(b) outbound

Fig. 2. Distribution of the number of stops among the different routes

5.1 Data Selection

We conducted our experiments with two historical datasets from two different time periods: March 2017 to March 2018 and June 2018 to June 2019. Each data record relates to a bus's boarding and alighting history at a bus stop. Table 6 shows a few basic statistics about the data we used in this study. In addition, the relationship between the number of stops and routes in Pittsburgh, for both inbound and outbound directions, is shown in Figure 2. As we can see in Figure 2a, about 55% of inbound routes have between 50 and 80 stops whereas only about 20% have more than 80 stops and nearly 25% have less than 50 stops. Almost the same pattern is observed in Figure 2b for outbound routes. More than half of the outbound routes have 50-80 stops while the other half have more than 80 or less than 50 stops in total. Routes 59 and O1 are two examples that have the highest and the lowest number of stops respectively.

5.2 Data Preprocessing

We first converted the selected data into a form that we could work with. That meant converting the STEP file into a text standard format like CSV. The next step is to detect data anomalies and correct or entirely remove them from the data. The following is the list of data inconsistencies we identified and removed before starting any data analysis:

- ***Invalid values***: we found out that there were some invalid characters such as a star(*) in some of the data which make their type incorrect. Theses values were removed or replaced by the correct ones after they were discovered.
- ***Missing records***: After comparing the available records in historical data and schedule data, we found out that the data coverage is about 80 percent; we decided to use the existing data for the next phases without imputing the missing records.
- ***Missing values***: During data analysis we found out that bus stop information (Stop ID and Stop Name) was unidentified in a number of data records which means they would be useless for the subsequent analysis and modeling. Therefore, such data records were evicted from the data (Figure 3). This included data that showed buses being in the river (possibly a result of an urban canyon effect[6] on the GPS) and following completely different bus routes (possibly a result of human error).

[6]https://en.wikipedia.org/wiki/Street_canyon

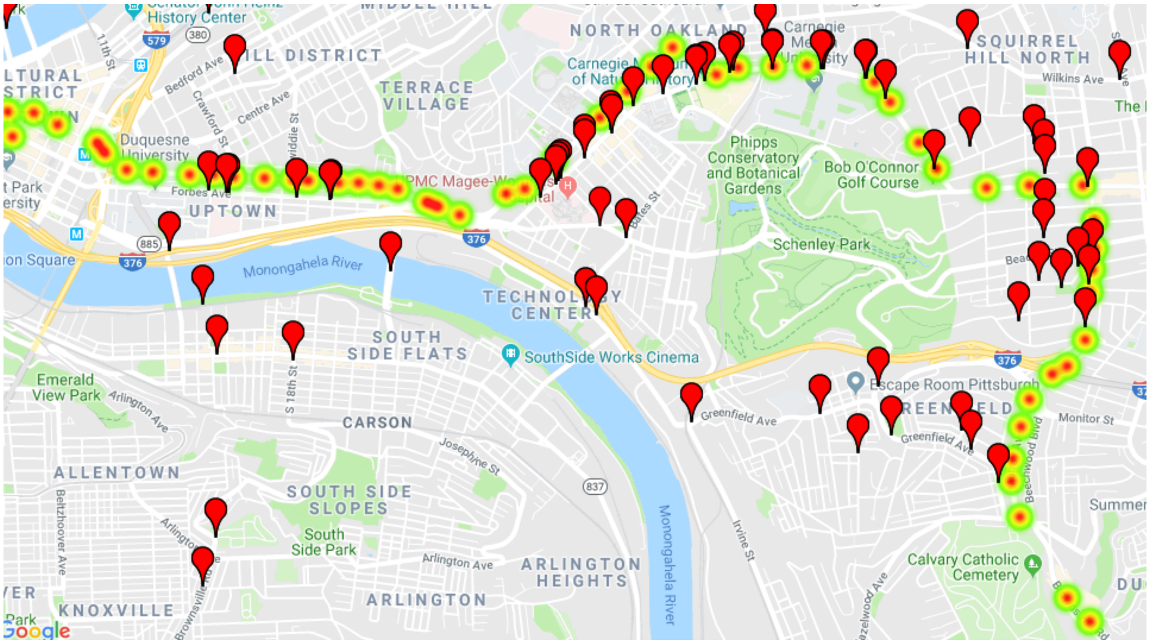


Fig. 3. Missing stops for 61C inbound in April 2017 - The yellow-green path with orange dots represents the actual route for 61C inbound and the red markers show the locations where the latitude and longitude of the missing stops in data specify. As you can see many of the markers do not match the true locations of stops in this route during this specific time period.

- ***Duplicate records***: we observed that for some of the records, there was at least another copy which contained the same features such as date, bus, trip and stop just like that record, but the copy was different in other features. Our hypothesis was that when a bus driver dwells at a stop behind a red light, he/she probably opens the bus' doors to board and/or alight passengers more than once which leads to creating such duplicates in data. Tracking a few examples of this scenario proved that our assumption is true up to a certain level. Such records were also eliminated to increase the consistency of the following analysis.

5.3 Data Transformation

To prepare the preprocessed data for the machine learning models that we will apply in the next section, we need to perform the following transformations:

- ***Attribute Decomposition***: The date and time features need to be split into their constituent parts before they can be used by the machine learning models. For example, we decomposed date and time from each data instance into month of the year, hour, and minute respectively.
- ***Encoding Categorical Attributes***: One task of data transformation is converting categorical data into numeric data. One of the methods for this conversion is to create dummy variables for all categorical attributes which in our case include month of the year, day of the week, time of the day and stops.
- ***Adding new features***: Because of our modeling needs, we had to add two different kinds of features to the preprocessed data:

- Features obtained from a secondary data source: Weather is one of the important features that can affect the crowding level in public transportation. We used weather data including average temperature, rainfall and snowfall per hour from the Pennsylvania State Climatologist [14], the National Weather Service Climate of Pittsburgh [34] and National Operational Hydrologic Remote Sensing Center [10] and integrated these features into our data.
- Features obtained from original data: Some of the features we need for the modeling such as the type of each bus, the load of a bus at previous stops, and the current crowding level were constructed from other features and/or other data instances and then were added to the preprocessed data.

6 EVALUATION FRAMEWORK

The goal of our evaluation is two-fold:

- Determine the usefulness of the different feature sets in predicting bus load and crowding levels, and
- Evaluate the performance of the proposed models compared to the baselines.

We have used 20% of each dataset as test data, for model evaluation. In particular, we fed the models with the test data and let them predict the corresponding loads, crowding levels and their uncertainties. To qualify the performance of the models and the baselines, we used three metrics including RMSE, Log Loss and F1 Score, which we explain next.

6.1 Metrics

We have chosen three performance metrics namely RMSE, Log Loss and F1 score to evaluate the predictions coming from the baselines, Negative Binomial and Random Forest models.

- The **Root Mean Square Error (RMSE)** is the standard deviation of the prediction errors which tells you how concentrated the data is around the line of best fit [4].

$$RMSE = \sqrt{\sum_{i=1}^{N} \frac{(predicted_i - observed_i)^2}{N}} \tag{3}$$

- The **Log Loss** is a measure of how good probability estimates are (also known as cross entropy) [20]. Equation 4 shows the log loss formulation for multi-classification.

$$logloss = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{M} y_{ij} log(p_{ij}) \tag{4}$$

  where N is the number of rows in test set, M is the number of fault delivery classes, y_{ij} is equal to 1 if observation belongs to class j and p_{ij} is the predicted probability that observation belongs to class j.
- The **F1 score** is defined as the harmonic mean of precision and recall and is known to be more useful than accuracy if there is class imbalance in classification [44].

$$F_1 = 2 * \frac{precision * recall}{precision + recall} \tag{5}$$

Since predicting the probabilities of crowding level will be as useful as predicting the crowding level itself, we used Log Loss as one of the performance metrics. Furthermore, due to the phenomenon of class imbalance in crowding levels, we decided to use the F1 score with micro-averaging that aggregates the contributions of all classes to compute the average metric [31].

Table 7. Feature Sets to be used in models. The last row is only used for models with route-direction input data.

Features	FS1	FS2	FS3	FS4	FS5	FS6	FS7	FS8	FS9
TOD2 - TOD96	✓	✓	✓	✓	✓	✓	✓		
DOW2 - DOW5		✓	✓	✓	✓	✓	✓		
MOY2 - MOY12		✓	✓	✓	✓	✓	✓		
BusType		✓	✓	✓	✓	✓	✓		
Temperature, Rainfall, Snowfall		✓	✓	✓	✓	✓	✓		
PLoad1 - Pload5			✓					✓	
PLoad1 - Pload10				✓					✓
PLoad5					✓				
PLoad10						✓			
PLoad5 - Pload10							✓		
$Stop_2$ - $Stop_N$	✓	✓	✓	✓	✓	✓	✓	✓	✓

6.2 Algorithms

The algorithms that we employ in this framework are:

- Negative Binomial Regression, as explained in Section 4.1
- Random Forest Classification, as explained in Section 4.2

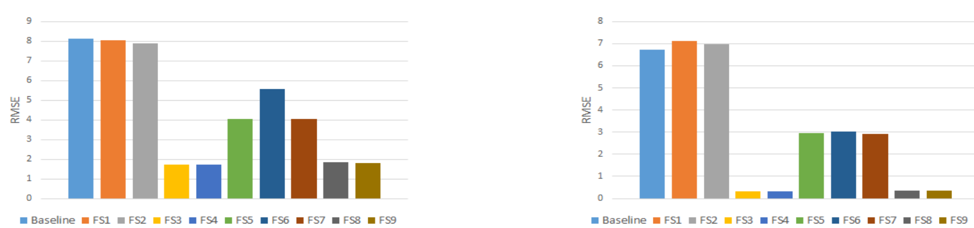
In both cases we have defined appropriate baselines (Section 4.3).

6.3 Data

As previously mentioned (Section 5), we use real-world data received from the Port Authority of Allegheny County, which we use as the "ground truth". We extract and process data inputs with two types including Route-Stop and Route-Direction for each modeling algorithm.

7 EVALUATION RESULTS AND DISCUSSION

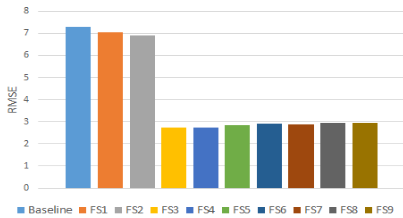
We summarize the models' evaluation by representing the RMSE, Log Loss and F1 score values for baseline, Negative Binomial regression and Random Forest classification models for both route-stop and route-direction inputs. As mentioned before, first we only conducted simple experiments using selected routes but then after evaluating the results, we ran an experiment for all the existing routes by modeling them with our best selected approach (see section 7.9). It is worth mentioning that all the following experiments have been conducted over the first dataset besides the ones in sections 7.4 and 7.12.



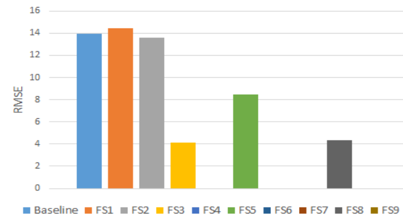
(a) RMSE histogram for 61B

(b) RMSE histogram for 12

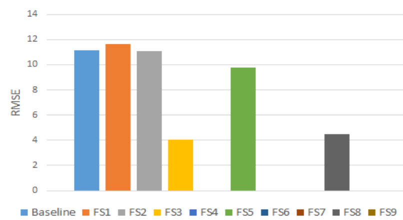
Fig. 4. RMSE histograms - Regression Models with Route-Stop (1)



(a) RMSE histogram for Y1



(b) RMSE histogram for P1 (CL4)



(c) RMSE histogram for P1 (CL5)

Fig. 5. RMSE histograms - Regression Models with Route-Stop (2)

7.1 Regression Results with Route-Stop (Figures 4, 5, 6)

Setup: For this experiment, we used the route-stop data input for modeling the selected routes and stops.

Results: Figures 4 and 5 show the RMSE for Baseline versus Negative Binomial Regression models with all 9 different feature sets for the 5 top routes including 61B, 12, Y1, P1 in cluster CL4 and P1 in cluster CL5 at their selected stops (see Tables 7 and 4). As it can be seen in all five bar charts, our models with feature sets FS3, FS4, FS8 and FS9 perform better than baseline models. As indicated in Table 7, FS3 and FS4 are feature sets that include time of the day, day of the week, month of the year, bus type, weather, and bus loads from 5 and 10 previous stops respectively whilst FS8 and FS9 only include bus loads from 5 and 10 previous stops and not the other variables.

Moreover, Figure 6 is a summary of all selected route-stop pairs that shows RMSE values for Baseline versus the Negative Binomial Regression models that turned out to perform the best according to the charts in Figures 4 and 5.

It is worth point out that histograms FS4, FS6, FS7 and FS9 are blank for P1 at both stops. The reason is because this route, which is the busiest route in Pittsburgh, has only about 15 stops in each direction which means there are less than 10 stops before these candidate stops we are predicting for. Therefore, there is no model built for the mentioned feature sets for route P1 at EAST BUSWAY AT NEGLEY STATION A and EAST BUSWAY AT NEGLEY STATION C.

Finally, in all mentioned Figures, models with FS3 and FS4 perform slightly better than models with FS8 and FS9 which means including variables other than previous loads in the models could make a difference. This statement is true for all experiments that we have conducted as part of this work.

Takeaway: Regression models with route-stop data inputs work up to 7.9 times better than their corresponding baselines.

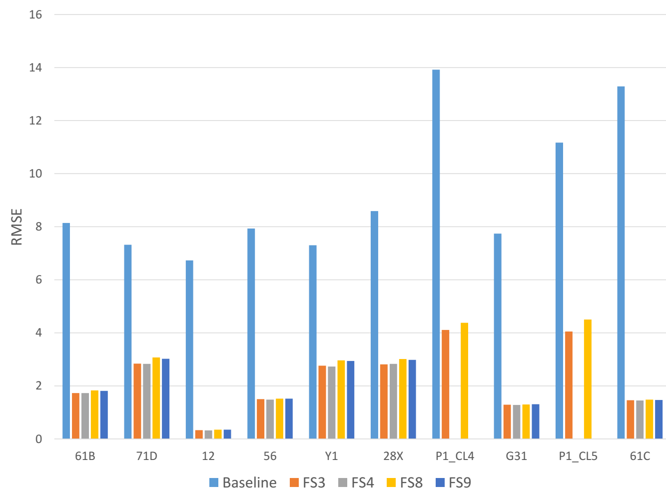


Fig. 6. RMSE histogram for all selected routes - best route-stop models

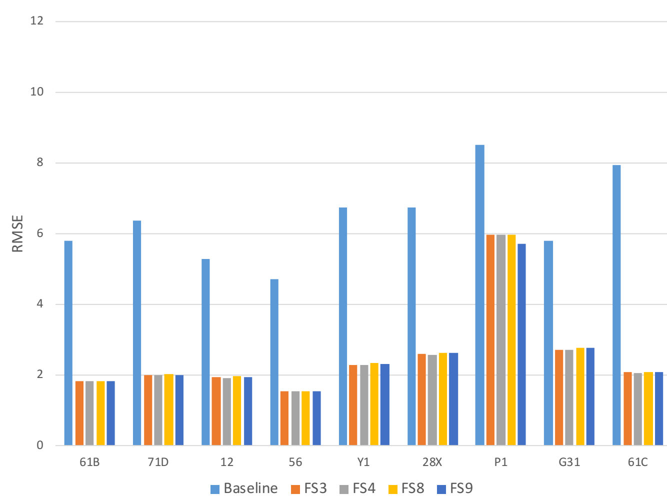


Fig. 7. RMSE histogram for selected inbound routes - best route-direction models

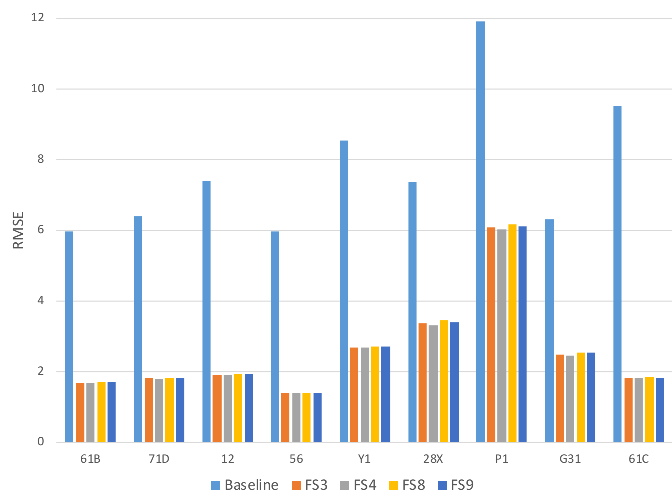


Fig. 8. RMSE histogram for selected outbound routes - best route-direction models

7.2 Regression Results with Route-Direction (Figures 7, 8)

Setup: For this experiment, we used the route-direction data input for modeling the selected routes.

Results: In this experiment, we carried out Negative Binomial regression for each selected route in each direction. As one can see in Figure 7 for inbound routes and Figure 8 for outbound routes, the performance of our models with feature sets FS3, FS4, FS8 and FS9 for the selected routes (see Table 7), is better than the performance of the baseline models for the same routes in both directions.

Takeaway: Regression models with route-direction data inputs perform up to 4.2 times better than their corresponding baselines.

Table 8. Random Forest vs Logistic Regression - F1 Score Comparison for inbound Routes

Route	Baseline	FS3		FS4	
		RF	*LR*	*RF*	*LR*
61B	0.91	0.97	0.97	0.97	0.97
71D	0.91	0.98	0.97	0.97	0.97
12	0.85	0.97	0.96	0.97	0.96
56	0.87	0.98	0.98	0.98	0.98
Y1	0.68	0.92	0.91	0.91	0.91
28X	0.85	0.95	0.94	0.95	0.93
P1	0.8	0.87	0.87	0.87	0.87
G31	0.83	0.96	0.95	0.96	0.95
61C	0.79	0.93	0.9	0.93	0.9

Table 9. Random Forest vs Logistic Regression - Log Loss Comparison for inbound Routes

Route	Baseline	FS3		FS4	
		RF	*LR*	*RF*	*LR*
61B	0.24	0.09	0.09	0.09	0.09
71D	0.23	0.08	0.09	0.08	0.09
12	0.31	0.08	0.11	0.1	0.11
56	0.28	0.06	0.07	0.06	0.07
Y1	0.7	0.23	0.24	0.24	0.24
28X	0.34	0.15	0.16	0.16	0.17
P1	0.47	0.33	0.33	0.32	0.32
G31	0.37	0.12	0.14	0.13	0.14
61C	0.49	0.19	0.22	0.19	0.22

7.3 Performance Comparison of Random Forest and Logistic Regression Models (Tables 8, 9, 10)

Setup: For this experiment, we used the route-direction data input for modeling the selected routes.

Results: As mentioned earlier (section 4.2) although we applied both Random Forest and Logistic Regression models on our data, we only report Random Forest results because Random Forest Classifiers seem to be more accurate than Logistic Regression models. To prove this statement, we compared the accuracy (in terms of F1 Score and Log Loss) of both types of models for all candidate routes and presented a selection of results in tables 8 and 9. Table 8 shows F1 Score values after testing models for inbound candidate routes with FS3 and FS4 as the models' feature sets. As it can be seen, F1 Scores for Random Forest models are equal to or slightly higher than F1 Scores for Logistic Regression models in all cases. A Log Loss comparison was also done with the same modeling and input setup and displayed in table 9. The numbers in this table indicate that our Random Forest models also perform equally well or better than our Logistic Regression models in terms of Log Loss. In addition to these tables, table 10 summarizes the outcomes obtained from the comparison of the two classification models for both inbound and outbound routes and both feature sets. According to this table, for 67% of experiments Random Forest classifiers performed better than Logistic Regression models in terms of F1 Score and they performed as well as each other for the remaining 33% . This table also represents Log Loss comparison of the models where Random Forest classifiers did equally well or better than Logistic Regression models in 94% of experiments.

Takeaway: Random Forest classifiers outperform Logistic Regression classifiers so we apply Random Forest on all datasets with different setups as it is explained in the following sections.

Table 10. Random Forest vs Logistic Regression - F1 Score and Log Loss Comparison for inbound and outbound Routes

F1 Score (the higher the better)	*RF >LR*	67%
	RF = LR	33%
	RF <LR	0%
Log Loss (the lower the better)	*RF <LR*	72%
	RF = LR	22%
	RF >LR	6%

Table 11. Random Forest vs Neural Network - F1 Score and Log Loss Comparison for inbound and outbound Routes

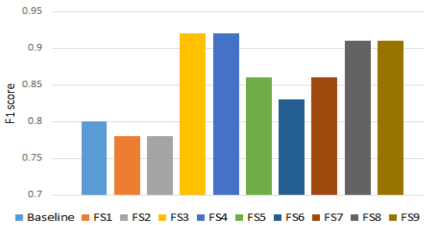
F1 Score (the higher the better)	*RF >NN*	61%
	RF = NN	39%
	RF <NN	0%
Log Loss (the lower the better)	*RF <NN*	67%
	RF = NN	28%
	RF >NN	5%

7.4 Performance Comparison of Random Forest and Neural Network Models (Tables 11 and 12)

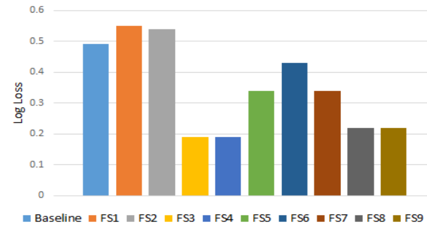
Setup: We used the route-direction data input for modeling the selected routes using the second dataset.

Results: In this section, we compare the performance of our Random Forest classifiers and our Neural Network classifiers with two feature sets: FS3 and FS4, for all candidate routes. As one can see in Table 12, our Random Forest models perform equally well (28% of cases) or better (72% of cases) than our Neural Network models in terms of F1 Score and Log Loss for inbound routes. Table 11 summarizes the comparison between the Random Forest and Neural Network models for both inbound and outbound routes and both feature sets. According to this table, RF classifiers work equally well or better than NN classifiers in 100% of cases in terms of F1 Score and in 95% of cases in terms of Log Loss.

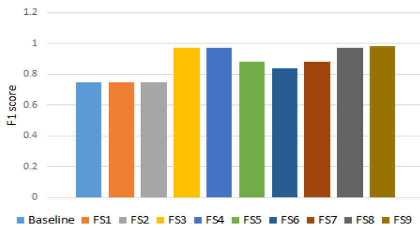
Takeaway: These results (as well as the results in the previous section) indicate that *Random Forest* classifiers outperform the other classification methods that we used and work the best for our data. Furthermore, we can infer that neural networks do not necessarily outperform the standard machine learning algorithms that can solve a large majority of problems.



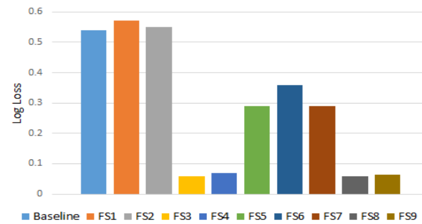
(a) F1 Score histogram for 61B



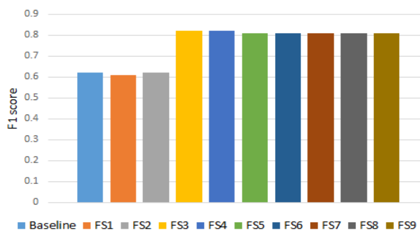
(b) Log Loss histogram for 61B



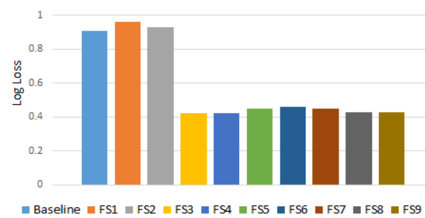
(c) F1 Score histogram for 12



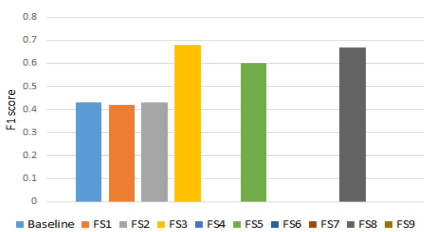
(d) Log Loss histogram for 12



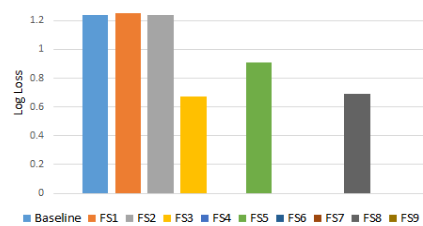
(e) F1 Score histogram for Y1



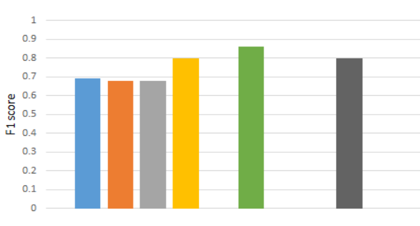
(f) Log Loss histogram for Y1



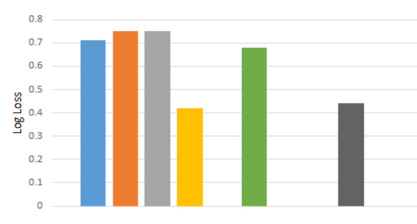
(g) F1 Score histogram for P1 (CL4)



(h) Log Loss histogram for P1 (CL4)



(i) F1 Score histogram for P1 (CL5)



(j) Log Loss histogram for P1 (CL5)

Fig. 9. F1 Score and Log Loss histograms

Table 12. Random Forest vs Neural Network Performance Comparison for inbound Routes

Route	F1 Score				Log Loss			
	FS3		FS4		FS3		FS4	
	RF	*NN*	*RF*	*NN*	*RF*	*NN*	*RF*	*NN*
61B	**0.98**	0.97	**0.98**	0.97	*0.07*	0.07	*0.07*	0.07
71D	**0.98**	0.97	**0.98**	0.97	**0.06**	0.08	*0.07*	0.07
12	**0.98**	0.97	*0.98*	0.98	**0.06**	0.08	*0.07*	0.07
56	**0.99**	0.98	**0.99**	0.98	**0.04**	0.06	**0.05**	0.07
Y1	**0.94**	0.92	**0.94**	0.90	**0.19**	0.21	**0.19**	0.22
28X	**0.96**	0.95	**0.96**	0.95	**0.12**	0.13	**0.13**	0.14
P1	*0.88*	0.88	*0.88*	0.88	*0.28*	0.28	*0.30*	0.30
G31	**0.96**	0.95	*0.96*	0.96	**0.11**	0.14	**0.12**	0.13
61C	**0.95**	0.93	**0.95**	0.93	**0.14**	0.17	**0.15**	0.17

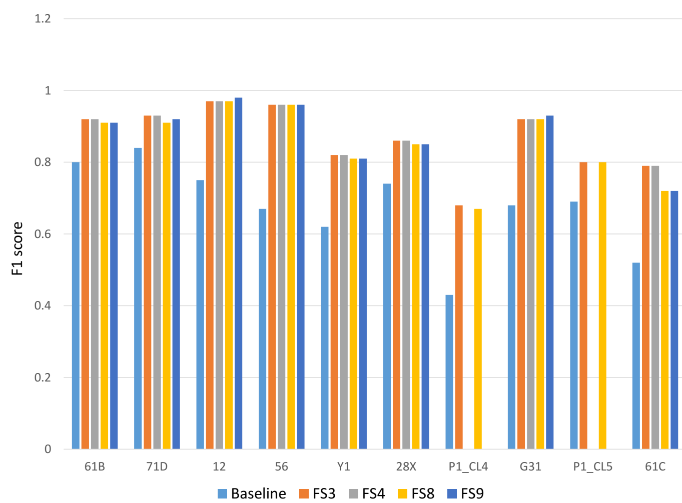


Fig. 10. F1 Score histogram for all selected routes - best route-stop models

7.5 Classification Results with Route-Stop (Figures 9, 10, 11 and Table 13)

Setup: We used the route-stop data input for modeling the selected routes and stops in this experiment.

Results: Figures 9a to 9j show the F1 score and Log Loss for the Baseline versus Random Forest models with all 9 different feature sets for 61B, 12, Y1, P1 in cluster CL4 and P1 in cluster CL5 (see Tables 7 and 4). According to the histograms in Figure 9, models with FS3, FS4, FS8 and FS9 feature sets perform better than the baseline and the other models, in terms of both Log Loss and F1 score. For instance, Figure 9c and Figure 9d show that F1 score has increased by 29% and Log Loss has decreased by 8 times in models with FS3 or FS4 feature sets compared to the baselines.

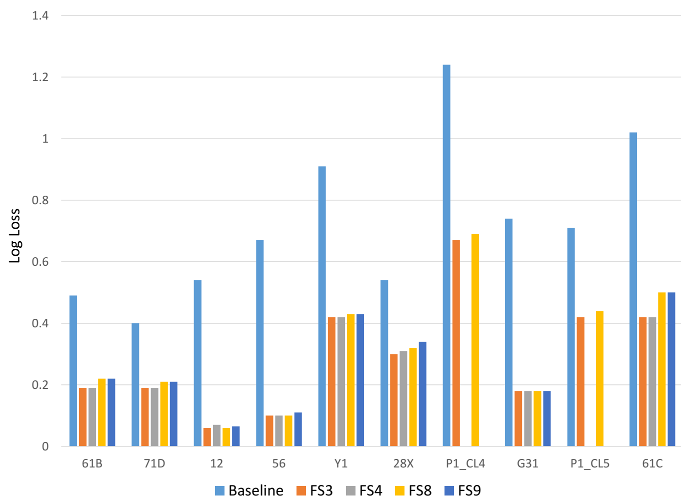


Fig. 11. Log Loss histogram for all selected routes - best route-stop models

Table 13. Standard Deviation, Mean, F1 Score and Log Loss values for Selected Routes. The table shows that models for routes with higher average load and higher standard deviation are less accurate than the models for routes with lower average load and lower standard deviation.

Route	Cluster	SD	Mean	F1 Score	Log Loss
61B	CL1	6.81	8.75	0.92	0.19
71D	CL1	6.24	8.26	0.93	0.19
12	CL2	9.1	11.24	0.97	0.06
56	CL2	7.99	11.45	0.96	0.1
Y1	CL3	11.46	18.58	0.82	0.42
28X	CL3	9.98	12.05	0.86	0.3
P1	CL4 & CL5	12.15	21.57	0.68	0.67
G31	CL4	11.76	12.04	0.90	0.19
61C	CL5	9.42	16.28	0.79	0.42

The same pattern can also be seen in Figures 10 and 11 which show F1 score and Log Loss for baselines and the best-performing Random Forests for all selected route-stop pairs. Our other observation from these histograms is that route-stop pairs that are in clusters CL1 and CL2, have lower Log Loss and higher F1 score compared to the other pairs in other clusters. One interpretation is that less crowded routes have less "messier" data which leads to more accurate models. What we mean by "messiness" is how spread our data is. As you can see in table 13, the first four routes that belong to clusters CL1 and CL2 have lower mean (which means less crowded) and lower standard deviation over their "load" compared to other routes in other clusters and also as it can be seen the accuracy of their models built with one of the feature sets such as FS3 is better than other routes. Since a high standard deviation indicates that the data points are spread out over a large range of values [7], we are convinced that the models we build for more crowded routes are less accurate than the models we build for less crowded routes.

Takeaway: Classification models with route-stop data inputs perform up to 8 times better than their corresponding baselines.

7.6 Classification Results with Route-Stop for 61C (Figure 12)

Setup: For this experiment, we used the route-stop data input for modeling 61C at all of its stops. 61C is one of the most popular and crowded routes in Oakland neighborhood, where University of Pittsburgh campus is located, which makes this route an important target for our project. This section is essentially an example of what we explained in the previous section about the relation between load and model accuracy.

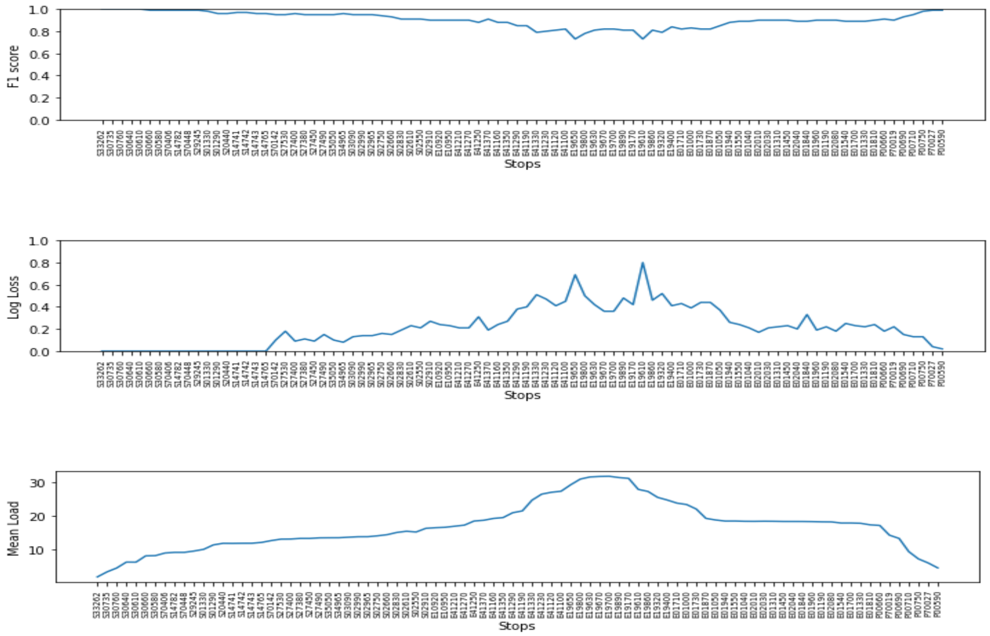
Results: Figures 12a and 12b represent the changes of F1 score, Log Loss and Mean load over all stops in 61C for both directions. The model used for this experiment was Random Forest classifier with route-stop data input with FS3 feature set. As you can see, the values of these three metrics fluctuate for each stop and our model has different accuracy at different stops. It is noticeable that the accuracy of each model at each stop has a close relationship with the mean load of the route at that stop. In other words, when the number of passengers on the bus route increases (specifically at stops in the middle of the route path), there will be a decrease in F1 and an increase in Log Loss. This pattern is visible in both 61C inbound and 61C outbound graphs.

Takeaway: The accuracy of each model for 61C at each stop has a relationship with the average load of the route at that stop.

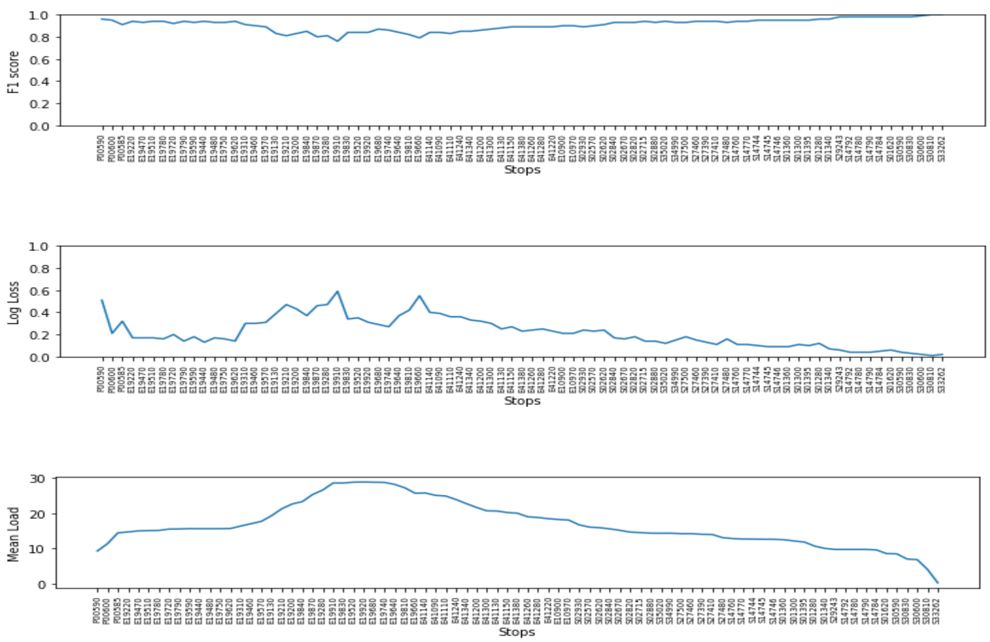
7.7 Classification Results with Route-Direction (Figures 13, 14, 15, 16)

Setup: For this experiment, we used the route-direction data input for modeling the selected routes.

Results: We depict the comparison between F1 score and Log Loss for the selected inbound routes for the Random Forest classifiers with best feature sets in Figures 13 and 14. Figures 15 and 16 show the same comparison for the selected outbound routes. Similar to other previous experiments, our proposed models with FS3 and FS4 completely outperform baselines and are slightly better than models with FS8 and FS9. If we compare the results of the classification models with route-stop input (section 7.5) and classification models with route-direction input, we can clearly see that the latter surpass the former in terms of F1 and Log Loss. However, one exception can be seen for route 12 in which route-stop model performs slightly better than its route-direction model. We believe this to be minor because we reported the outcomes of route-stop models for the selected routes at the *candidate* stops not all the existing stops. In other words, since the accuracy of the route-stop models changes from one stop to another (as it was observed in the previous experiment for 61C), thus we should not generalize from the performance of models for just one stop. In fact, we resolved this problem by computing the average of F1 and Log Loss scores for all stops of a route and comparing them with F1 and Log Loss scores obtained from models built for that route in two directions. We completed our experiment in previous section by calculating the average of the metric values for 61C at all of its stops (direction-separated) and then we compared these average values with F1 and Log Loss scores from route-direction models for 61C. It turns out that the latter are very close (even slightly better) to the former which gives us another reason to choose route-direction models over route-stop models. Generally speaking, instead of having one model per each route and stop, we can have a model for each route while considering its stops as model variables. This way, we get an acceptable average accuracy for all stops without having trouble creating thousands of models.



(a) F1 Score, Log Loss and Mean Load for 61C inbound



(b) F1 Score, Log Loss and Mean Load for 61C outbound

Fig. 12. Classification Results with Route-Stop for 61C

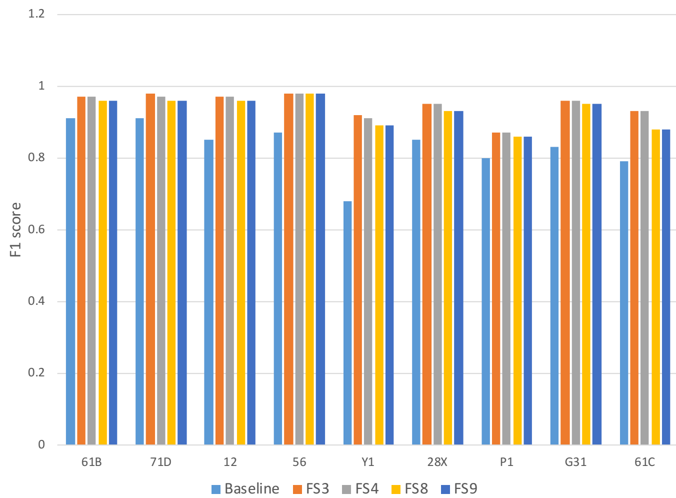


Fig. 13. F1 Score histogram for selected inbound routes - best route-direction models

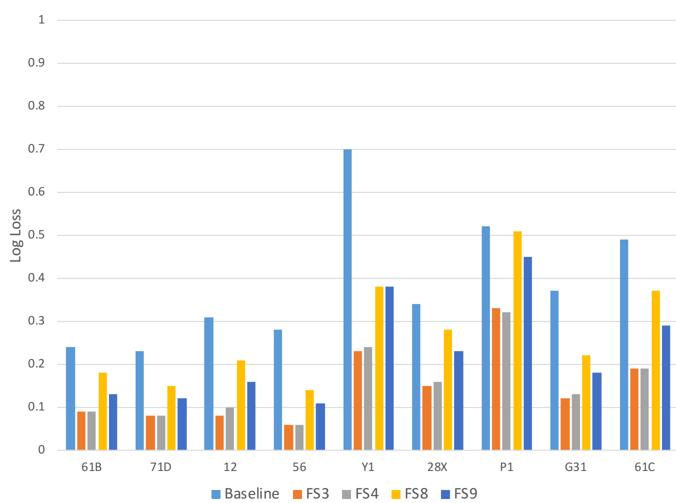


Fig. 14. Log loss histogram for selected inbound routes - best route-direction models

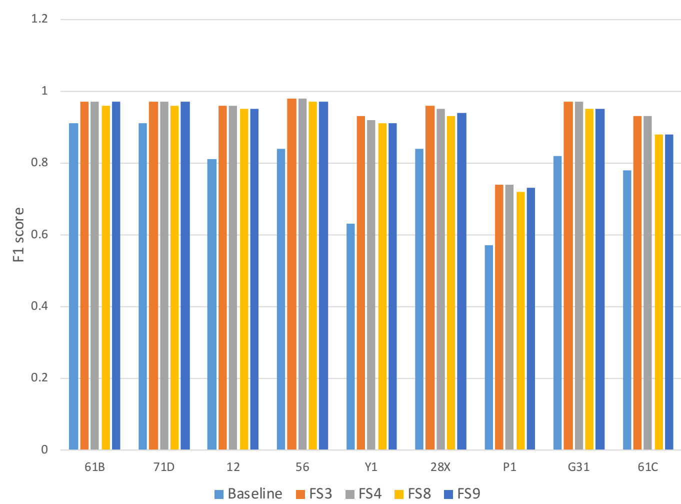


Fig. 15. F1 Score histogram for selected outbound routes - best route-direction models

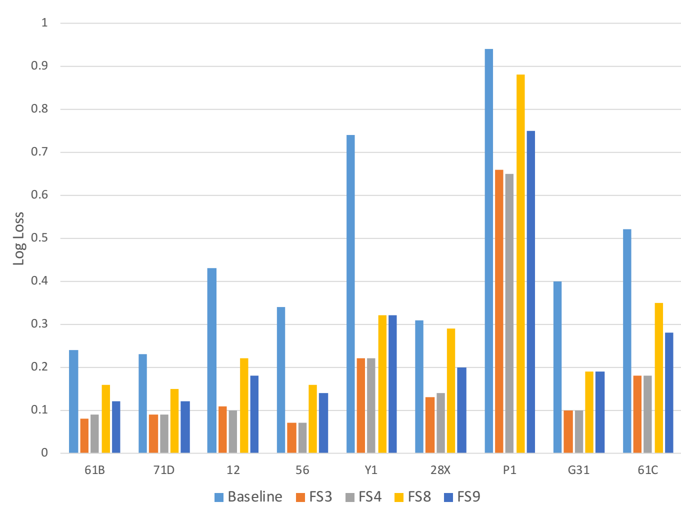
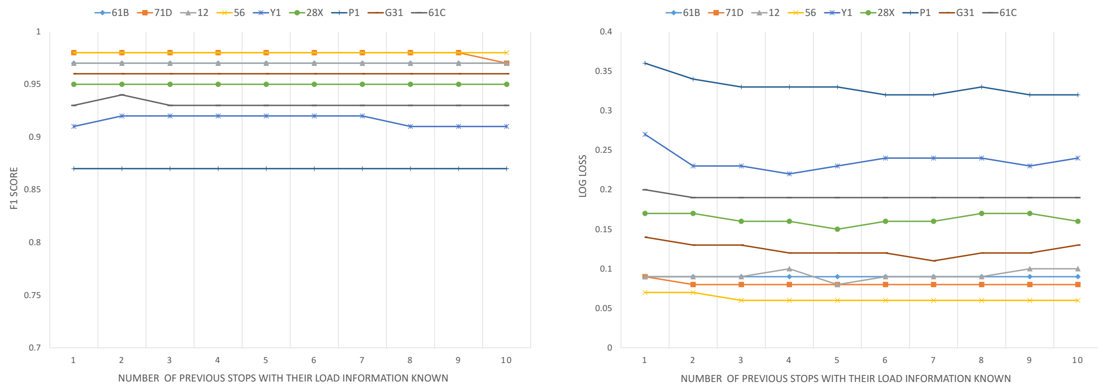


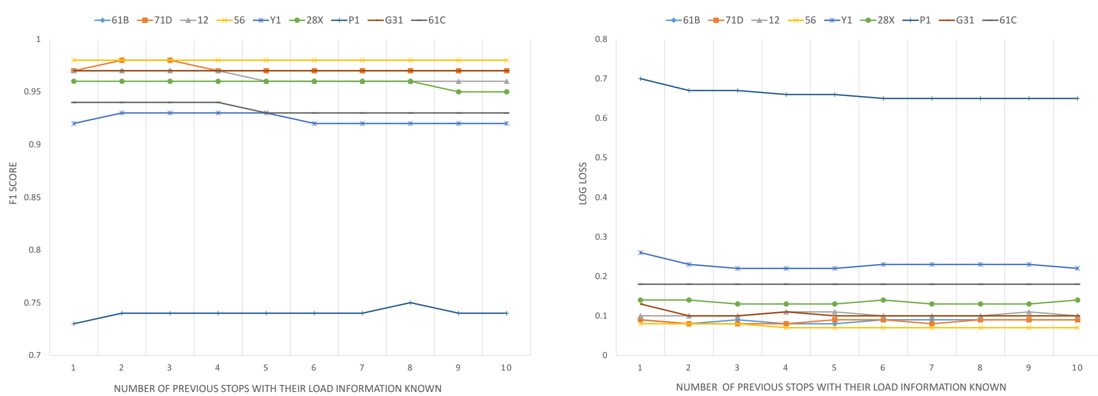
Fig. 16. Log loss histogram for selected outbound routes - best route-direction models

Takeaway: Classification models with route-direction data inputs perform up to 3.9 times better than their matching baselines and have better accuracy compared to classification models with route-stop data inputs.



(a) F1 score (b) Log loss

Fig. 17. F1 score and Log loss for selected inbound routes using load information from different previous stops



(a) F1 score (b) Log loss

Fig. 18. F1 score and Log loss for selected outbound routes using load information from different previous stops

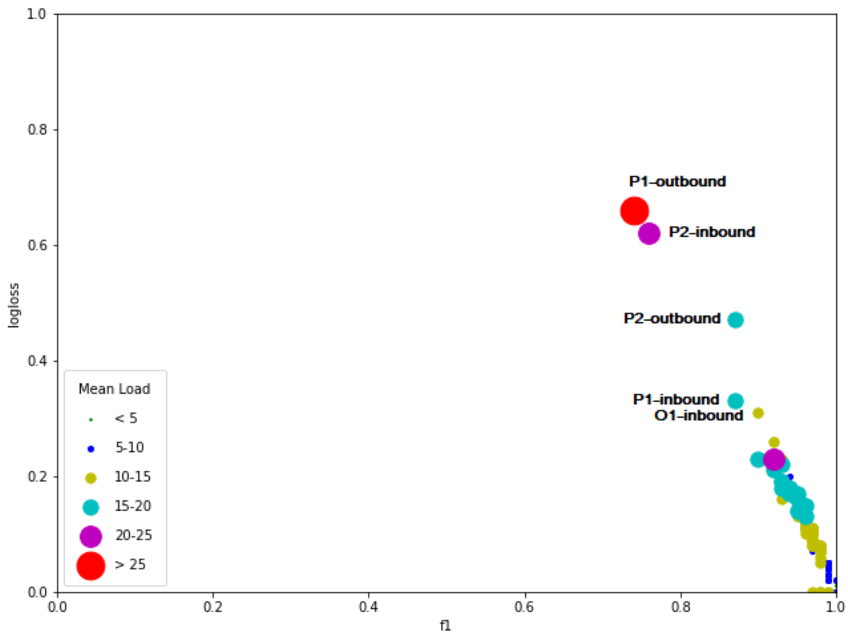


Fig. 19. The Results for All Routes Modeled with Random Forest Classifier for each Route-Direction

7.8 Stop-Load Sensitivities (Figures 17, 18)

Setup: For this experiment, we used the route-direction data input for modeling the selected routes.

Results: The models used in this experiment are Random Forest classifier models that are fitted with FS2 features plus loads from the stops prior to the stop we are predicting for. Figures 17 and 18 display the changes of F1 scores and Log Loss for selected routes in two directions. On the x-axis we list the number of previous stops (1-10) with their load information known. In other words, 1 corresponds to knowing the load of just the previous stop, 2 to knowing the load of the two previous stops, etc. According to these line charts, both F1 and Log Loss values vary slightly from one variable set to another and F1 values fluctuate even more insignificantly than Log Loss values. From this observation, we can infer that the accuracy of our proposed models barely depends on the number of previous stops with load information known and it stays almost unchanged when adding more load information from previous stops that are farther away. However, we should emphasize the importance of including a consecutive previous loads (starting from the stop right before the stop we are predicting for) in the feature sets because they remarkably improve the accuracy of the models. We have seen this time and time again in the presented results from all the introduced models which perform very well while having FS3 or FS4 feature sets as their independent variables.

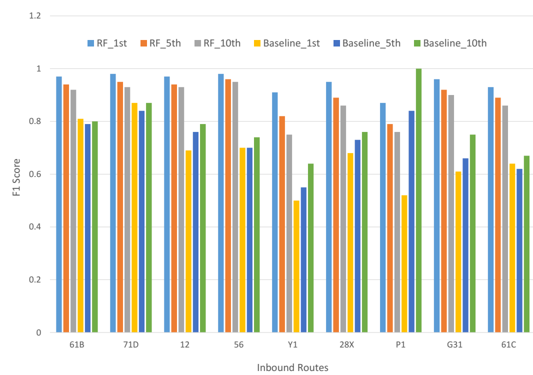
Takeaway: Adding more load information from previous stops that are farther away does not change the accuracy of our models.

7.9 Classification Results with Route-Direction - All Routes (Figure 19)

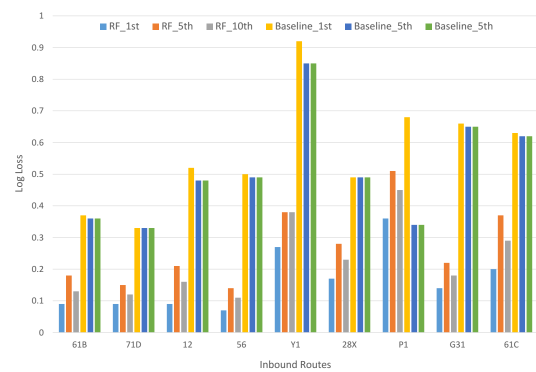
Setup: For this experiment, we used the route-direction data input for modeling *all the routes*.

Results: A summary of the results obtained from running the experiment for all existing bus routes in Pittsburgh is illustrated in Figure 19. This graph shows the F1 score versus Log Loss computed for each route in each direction modeled by Random Forest Classifiers. The size of the circles corresponds to the mean load that is calculated for each route-direction and partitioned in 6 different categories. Each category is also defined with a color to display the differences. As you may notice, most routes have load less than 20 people on average and models for routes whose mean load is lower than 15 passengers seem to be more accurate since their F1 scores are very close to 1.0 and their Log Loss scores are close to zero. However, mean load in routes like P1 and P2, that are known as two of the busiest routes in Pittsburgh, is over 20 people which leads to less accurate models. We think this phenomenon happens because data is typically more prone to messiness and a lot of missing values and outliers when buses become more crowded (see Table 13).

Takeaway: The accuracy of a model for a route in a certain direction has a reverse relationship with the average number of passengers using that route.



(a) F1 score



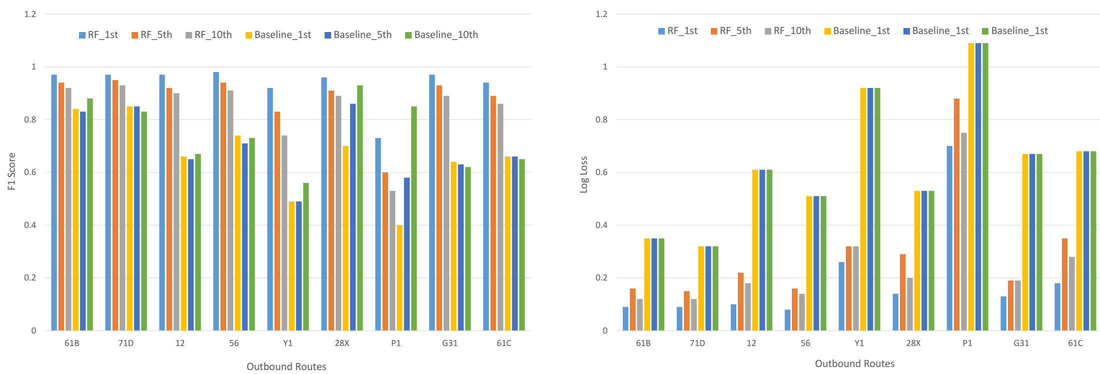
(b) Log loss

Fig. 20. F1 score and Log loss for selected inbound routes - Random Forest classifier vs. load-only Baseline

7.10 Classification Results vs. Load-only Baseline (Figures 20, 21)

The goal of this experiment is to evaluate a scenario where the Port Authority makes available to the public the "live" bus load data feed. In other words, it would be possible to find the current load of any bus. We use this experiment to determine how useful that information would be in predicting the bus crowding levels for that route further downstream (i.e., at the bus stop of interest), since presumably people would want to predict how full their bus *will be* before actually seeing the bus.

Setup: For this experiment, we used the route-direction data input for modeling the selected routes. In terms of a baseline, we used a variation of the Route-direction baseline where, instead of using the average load, we use the load at the 1st or the 5th or the 10th previous stop. Therefore, three separate baselines for each route in each direction are built and used for comparison (Baseline_1st, Baseline_5th and Baseline_10th). In addition, we also fit three different Random



(a) F1 score

(b) Log loss

Fig. 21. F1 score and Log loss for selected outbound routes - Random Forest classifier vs. load-only Baseline

Forest classifiers for each route in each direction with all the variables included plus only load at the 1st previous stop (RF_1st) or at the 5th previous stop (RF_5th) or at the 10th previous stop (RF_10th).

Results: Figures 20 and 21 show the F1 score and Log Loss values obtained from an experiment conducted to compare our proposed classification models with load-only baselines. According to the figures, all the proposed models perform better than the baselines except for models for P1 and 28X that show some small discrepancies compared to other routes. In addition, if we compare the three different type of models together, we can see that models that include the first previous load in their variables are usually more accurate than models that use 5th or 10th previous loads. However, this does not hold for baselines. As you may notice, baselines that use load from the 10th previous stop work better than the other baselines although a few different behaviours for some routes are observed.

Takeaway: Using just live data has limited application in predicting bus crowding levels further downstream. Such signals should instead be combined with historical data to build more accurate predictive models.

Table 14. Top 15 most Crowded Routes in CL4 / CL5

Crowding Level	Description	Routes (in order of crowdedness, starting from the left)
CL5	Crushed	P1, 61C, 61D, 71B, 71A, P78, 61B, P2, 75, 71D, P71, 58, 71C, 67, 1
CL4	Many People Standing	P1, G31, G2, G3, 28X, Y1, 75, 71B, 71A, 69, 41, P78, 61B, 6, 61C

Table 15. The most Crowded Routes and Stops in Pittsburgh Reported by Google Maps

Route	Stop
22	Helen St at Catherine St, Helen St at Ella St,
71A	Craig St at Centre Ave FS, Negley Ave at Jackson St, Negley Ave at Hampton St
71C	Negley Ave at Centre Ave, Negley Ave at #370 (Baum Blvd), Negley Ave at Penn Ave FS

Table 16. Comparing "Crowded" bus routes as identified by Google Maps vs our work

Route	Origin	CL1	CL2	CL3	CL4	CL5	% trips crowded
22	Google Maps	45.82	1.79	0.01	0.0	0.0	0%
71A	Google Maps	81.19	1.43	0.39	0.09	0.1	10%
71C	Google Maps	75.23	0.74	0.17	0.034	0.007	11%
P1	Our work	43.29	20.37	3.29	0.51	0.88	15%
61C	Our work	71.81	8.4	1.4	0.05	0.3	17%
61D	Our work	72.92	2.44	0.33	0.008	0.2	12%

7.11 The perils of crowdsourced data (Tables 14, 15, 16)

Setup: In this last data-driven evaluation study, we wanted to compare the quality of crowdsourced data with the "ground truth," as collected by the automated person counting devices on all buses. Towards this we used data from our route sampling analysis (Section 3.3), data from Google Maps [22] about Pittsburgh's most crowded bus routes, and published data from the Port Authority [32].

Results: Table 14 show the most crowded (CL5) and second most crowded (CL4) routes in Pittsburgh according to our analysis. Table 15 shows the three most crowded bus routes in Pittsburgh according to Google Maps. Lastly, Table 16 has a comparison of the breakdown of each route with regards to the five bus crowding levels, for the top three crowded routes identified by Google Maps and the top three identified by our work. The last column in that table comes from the Port Authority's 2018 Annual Service Report [32] and shows the *percentage of trips that were crowded*.

We clearly see that all the routes we identified as the most crowded ones (P1, 61C, 61D) are very crowded indeed, whereas out of the three that Google Maps identified as the most crowded routes, two (71A, 71C) are indeed somewhat crowded (although not the top ones), whereas one (22) is not crowded at all. This is the main challenge of relying exclusively on crowd-sourced information. There is often an inherent bias in data collection leading to skewed conclusions if that is the only data source. A well-known example of this bias is the case of the StreetBump smartphone app to detect potholes in Boston, which ended up missing inputs from significant parts of the population — often those who have the fewest resources [15].

Takeaway: Systems that rely purely on crowdsourcing to collect bus crowding levels are not always accurate.

7.12 Results for the Second Dataset (Figures 22, 23, 24, 25)

In order to confirm that our experimental results and conclusions are generalizable and transferable we evaluated our proposed models with a second dataset. The second dataset was from the Port Authority and was for a completely different one-year period. It is worth noting that Port Authority's schedules change quarterly, so there are many differences among these datasets (see Table 6). We applied our Random Forest classifiers with the route-direction data input (as our desired setup) on the second Port Authority dataset and compared their accuracy with the results from the first dataset.

Setup: For this experiment, we used the route-direction data input for modeling the selected routes.

Results: According to Figures 22, 23, 24 and 25 our proposed classification models with FS3 and FS4 feature sets outperform baselines models with FS8 and FS9 in terms of both F1 Score and Log Loss, similar to what we saw in the previous experiments with the first dataset.

Takeaway: Our classification models with route-direction data inputs applied on the second dataset, perform up to 4.5 times better than their matching baselines.

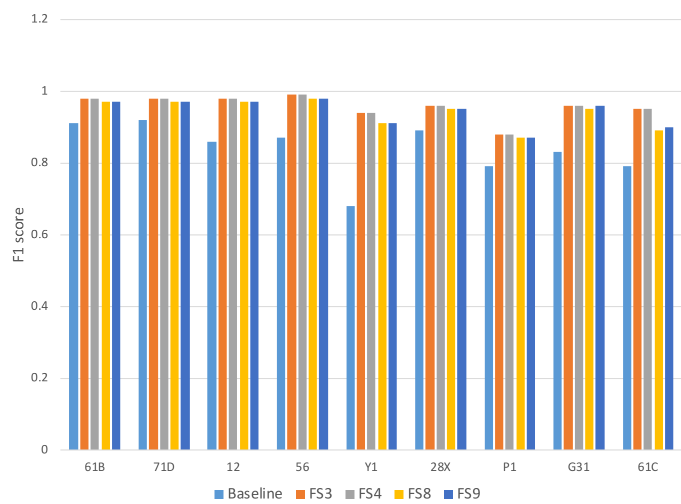


Fig. 22. F1 Score histogram for selected inbound routes - second dataset

Discussion: Considering the results we obtained from conducting experiments with the two datasets and according to table 6, we can infer that we have slightly more data instances in the second datasets (about 2 million) than the first one and the models' accuracy is also slightly higher when we apply our models on the second dataset.

7.13 Discussion

In our evaluation, we identified feature sets FS3/FS4 to be the best choice among all other feature sets. However, these sets have more than 100 features. In real-life, it may be preferable to consider the trade-off between model quality and model complexity. As such, another good option may be FS8 which only contains that last 5 prior loads and performs almost as good as the more complicated FS3/FS4 sets.

Taking a step back, one may wonder: *if live bus loads are available, does it still make sense to do modeling?* The answer is yes, as we showed in the previous section. Ideally, one would combine live data with predictive models so that they are able to more accurately predict bus crowding levels a few stops away or well ahead of time, as part of pre-trip planning.

8 CONCLUSION AND FUTURE WORK

In this work, we framed the *"how full will my next bus be?"* question as a regression and as a classification problem and developed a modeling framework to predict bus load and bus crowding levels using data from Pittsburgh. Our evaluation results showed that the proposed framework (using Random Forest Classifiers with route-direction data inputs) performs very well when using time of day, day of the week, month of the year, bus type, weather, and the bus loads from the 5 or 10 prior stops as the selected features. In fact, our models' performance was up to 8 times better than the baselines. Although we developed our modeling framework using only data from Pittsburgh, we are confident

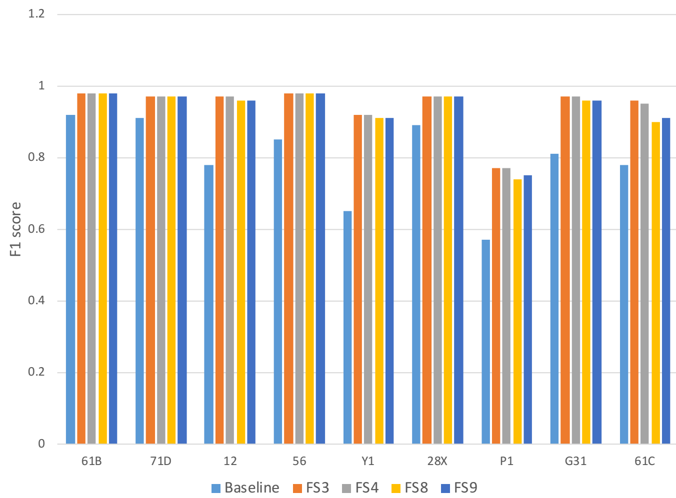


Fig. 23. F1 Score histogram for selected outbound routes - second dataset

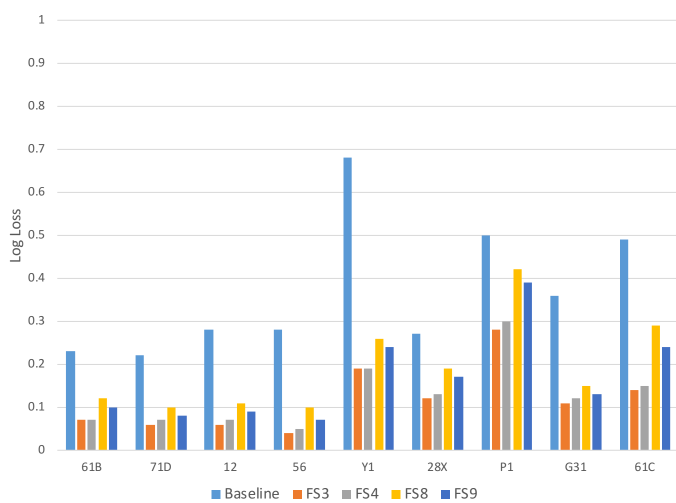


Fig. 24. Log loss histogram for selected inbound routes - second dataset

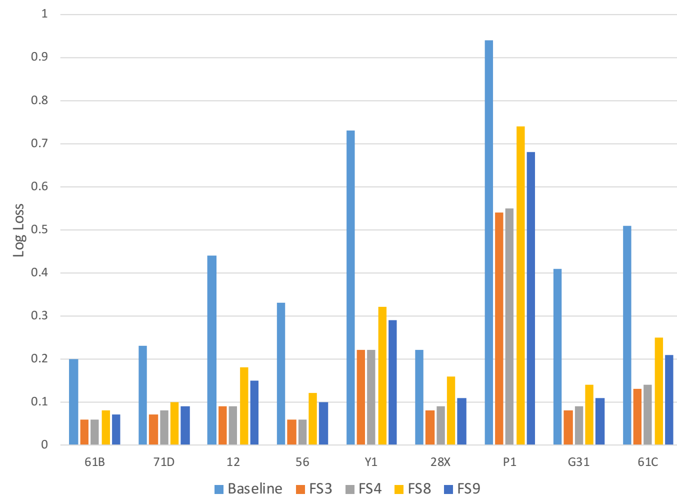


Fig. 25. Log loss histogram for selected outbound routes - second dataset

that the same process and the proposed models can be applied to data from other cities, especially since we evaluated our models with two different, completely disjoint datasets. .

As part of our future work we intend to deploy these models in our PittSmartLiving mobile application, using live real-time data from the Port Authorities, with the ultimate goal of improving the commuters' quality of life through high-quality, actionable information.

ACKNOWLEDGMENTS

This work is part of the PittSmartLiving project which is supported by NSF award CNS-1739413. We would also like to thank the Port Authority of Allegheny County for sharing their data with us and for their valuable feedback.

REFERENCES

[1] Vito Albino, Umberto Berardi, and Rosa Maria Dangelico. 2015. Smart cities: Definitions, dimensions, performance, and initiatives. *Journal of urban technology* 22, 1 (2015), 3–21.

[2] Kittelson & Associates, United States. Federal Transit Administration, Transit Cooperative Research Program, and Transit Development Corporation. 2003. *Transit capacity and quality of service manual*. Number 100. Transportation Research Board.

[3] Marco Balduini, Marco Brambilla, Emanuele Della Valle, Christian Marazzi, Tahereh Arabghalizi, Behnam Rahdari, and Michele Vescovi. 2018. Models and Practices in Urban Data Science at Scale. *Big Data Research* (Aug. 2018). https://doi.org/10.1016/j.bdr.2018.04.003

[4] Anthony G Barnston. 1992. Correspondence among the correlation, RMSE, and Heidke forecast verification measures; refinement of the Heidke score. *Weather and Forecasting* 7, 4 (1992), 699–709.

[5] Yu Bin, William HK Lam, and Mei Lam Tam. 2011. Bus arrival time prediction at bus stop with multiple route. *Transportation Research Part C: Emerging Technologies* 19, 6 (Dec. 2011), 1157–1170.

[6] Yu Bin, Yang Zhongzhen, and Yao Baozhen. 2006. Bus arrival time prediction using support vector machines. *Journal of Intelligent Transportation Systems* 10, 4 (Dec. 2006), 151–158.

[7] J Martin Bland and Douglas G Altman. 1996. Statistics notes: measurement error. *Bmj* 312, 7047 (1996), 1654.

[8] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (Oct. 2001), 5–32.

[9] Colin A Cameron and Pravin K Trivedi. 2001. Essentials of count data regression. *A companion to theoretical econometrics* 331 (2001).

[10] National Operational Hydrologic Remote Sensing Center. 2019. National Weather Service Climate of Pittsburgh. https://www.nohrsc.noaa.gov/interactive/

[11] Steven I-Jy Chien, Yuqing Ding, and Chienhung Wei. 2002. Dynamic bus arrival time prediction with artificial neural networks. *Journal of Transportation Engineering* 128, 5 (Sept. 2002), 429–438.

[12] Francois Chollet. 2017. *Deep Learning with Python* (1st ed.). Manning Publications Co., USA.

[13] Hafedh Chourabi, Taewoo Nam, Shawn Walker, J Ramon Gil-Garcia, Sehl Mellouli, Karine Nahon, Theresa A Pardo, and Hans Jochen Scholl. 2012. Understanding smart cities: An integrative framework. In *2012 45th Hawaii international conference on system sciences*. IEEE, 2289–2297.

[14] Pennsylvania State Climatologist. 2018-2019. PASC IDA Data. http://climate.psu.edu/data/ida/

[15] Kate Crawford. 2013. The Hidden Biases in Big Data. https://hbr.org/2013/04/the-hidden-biases-in-big-data

[16] Google Developers. 2016. General Transit Feed Specification (Static Overview). https://developers.google.com/transit/gtfs/

[17] Google Developers. 2020. Keras. https://www.tensorflow.org/guide/keras

[18] Donald Erdman, Laura Jackson, and Arthur Sinko. 2008. Zero-inflated Poisson and zero-inflated negative binomial models using the COUNTREG procedure. In *SAS Global Forum*, Vol. 2008. 322–2008.

[19] Vikash V Gayah, Zhengyao Yu, Jonathan S Wood, Mineta National Transit Research Consortium, et al. 2016. *Estimating uncertainty of bus arrival times and passenger occupancies*. Technical Report. Mineta National Transit Research Consortium.

[20] Irving John Good. 1952. Rational Decisions. *Journal of the Royal Statistical Society. Series B (Methodological)* 14, 1 (1952), 107–114.

[21] Taylah Hasaballah. 2019. Grab a seat and be on time with new transit updates on Google Maps. https://www.blog.google/products/maps/grab-seat-and-be-time-new-transit-updates-google-maps/

[22] Taylah Hasaballah. 2019. Transit crowdedness trends from around the world, according to Google Maps. https://www.blog.google/products/maps/transit-crowdedness-trends-around/

[23] Jeff Heaton. 2008. *Introduction to neural networks with Java*. Heaton Research, Inc.

[24] Avi Herbon and Yuval Hadas. 2015. Determining optimal frequency and vehicle capacity for public transit routes: A generalized newsvendor model. *Transportation research part B: methodological* 71 (2015), 85–99.

[25] Moovit Inc. 2012. MoovIT. https://moovitapp.com

[26] Scikit learn Developers. 2007-2018. scikit-learn models. https://scikit-learn.org/

[27] Zheng Li and David A Hensher. 2013. Crowding in public transport: a review of objective and subjective measures. *Journal of Public Transportation* 16, 2 (2013), 6.

[28] Yanchi Liu, Chuanren Liu, Nicholas Jing Yuan, Lian Duan, Yanjie Fu, Hui Xiong, Songhua Xu, and Junjie Wu. 2014. Exploiting heterogeneous human mobility patterns for intelligent bus routing. In *2014 IEEE International Conference on Data Mining*. IEEE, 360–369.

[29] Yanchi Liu, Chuanren Liu, Nicholas Jing Yuan, Lian Duan, Yanjie Fu, Hui Xiong, Songhua Xu, and Junjie Wu. 2017. Intelligent bus routing with heterogeneous human mobility patterns. *Knowledge and Information Systems* 50, 2 (2017), 383–415.

[30] Yan Lyu, Chi-Yin Chow, Victor CS Lee, Joseph KY Ng, Yanhua Li, and Jia Zeng. 2019. CB-Planner: A bus line planning framework for customized bus systems. *Transportation Research Part C: Emerging Technologies* 101 (2019), 233–253.

[31] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schutze. 2008. Introduction to Information Retrieval? Cambridge University Press 2008. *Ch* 20 (2008), 405–416.

[32] Port Authority of Allegheny County. 2018. Port Authority of Allegheny county annual service report 2018.

[33] University of Pittsburgh. 2016. Pitt Smart Living Project. Retrieved March 13, 2019 from https://pittsmartliving.org/

[34] Pittsburgh Weather Forecast Office. 2018-2019. Interactive Snow Information. https://w2.weather.gov/climate/xmacis.php?wfo=pbz

[35] RH Oldfield and PH Bly. 1988. An analytic investigation of optimal bus size. *Transportation Research Part B: Methodological* 22, 5 (1988), 319–337.

[36] Joseph N Prashker. 1979. Direct analysis of the perceived importance of attributes of reliability of travel modes in urban travel. *Transportation* 8, 4 (Dec. 1979), 329–346.

[37] Amer Shalaby and Ali Farhan. 2004. Prediction model of bus arrival and departure times using AVL and APC data. *Journal of Public Transportation* 7, 1 (2004), 3.

[38] Aaron Steinfeld, John Zimmerman, Anthony Tomasic, Daisy Yoo, and Rafae Dar Aziz. 2011. Mobile transit information from universal design and crowdsourcing. *Transportation Research Record* 2217, 1 (2011), 95–102.

[39] Transit. 2012. TransitApp. https://transitapp.com

[40] Yu Wei and Mu-Chen Chen. 2012. Forecasting the short-term metro passenger flow with empirical mode decomposition and neural networks. *Transportation Research Part C: Emerging Technologies* 21, 1 (2012), 148–162.

[41] Wen xia Sun, Ti Song, and Hai Zhong. 2009. Study on bus passenger capacity forecast based on regression analysis including time series. In *2009 International Conference on Measuring Technology and Mechatronics Automation*, Vol. 2. IEEE, 381–384.

[42] Zhang Xiong, Hao Sheng, WenGe Rong, and Dave E Cooper. 2012. Intelligent transportation systems for smart cities: a progress review. *Science China Information Sciences* 55, 12 (2012), 2908–2914.

[43] Menno Yap, Oded Cats, and Bart van Arem. 2018. Crowding valuation in urban tram and bus transportation based on smart card data. *Transportmetrica A: Transport Science* (2018), 1–20.

[44] Dell Zhang, Jun Wang, and Xiaoxue Zhao. 2015. Estimating the uncertainty of average F1 scores. In *Proceedings of the 2015 International Conference on The Theory of Information Retrieval*. 317–320.

[45] Jun Zhang, Dayong Shen, Lai Tu, Fan Zhang, Chengzhong Xu, Yi Wang, Chen Tian, Xiangyang Li, Benxiong Huang, and Zhengxi Li. 2017. A real-time passenger flow estimation and prediction method for urban bus transit systems. *IEEE Transactions on Intelligent Transportation Systems* 18, 11 (2017), 3168–3178.

[46] Wenfeng Zhang, Zhongke Shi, and Zhiyong Luo. 2008. Prediction of urban passenger transport based-on wavelet SVM with quantum-inspired evolutionary algorithm. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. IEEE, 1509–1514.