Addressing APC Data Sparsity in Predicting Occupancy and Delay of Transit Buses: A Multitask Learning Approach

Ammar Bin Zulqarnain[1*], Samir Gupta[1*], Jose Paolo Talusan[1],
Philip Pugliese[2], Ayan Mukhopadhyay[1], Abhishek Dubey[1]

[1]Vanderbilt University
[2]Chattanooga Area Regional Transportation Authority

Abstract—**Public transit is a vital mode of transportation in urban areas, and its efficiency is crucial for the daily commute of millions of people. To improve the reliability and predictability of transit systems, researchers have developed separate single-task learning models to predict the occupancy and delay of buses at the stop or route level. However, these models provide a narrow view of delay and occupancy at each stop and do not account for the correlation between the two. We propose a novel approach that leverages broader generalizable patterns governing delay and occupancy for improved prediction. We introduce a multitask learning toolchain that takes into account General Transit Feed Specification feeds, Automatic Passenger Counter data, and contextual temporal and spatial information. The toolchain predicts transit delay and occupancy at the stop level, improving the accuracy of the predictions of these two features of a trip given sparse and noisy data. We also show that our toolchain can adapt to fewer samples of new transit data once it has been trained on previous routes/trips as compared to state-of-the-art methods. Finally, we use actual data from Chattanooga, Tennessee, to validate our approach. We compare our approach against the state-of-the-art methods and we show that treating occupancy and delay as related problems improves the accuracy of the predictions. We show that our approach improves delay prediction significantly by as much as 4% in F1 scores while producing equivalent or better results for occupancy.**

I. Introduction

Public transportation plays a crucial role in facilitating the daily commuting needs of a significant portion of modern communities owing to its affordability. However, ridership in public transportation has steadily declined in the USA, particularly in the South and Midwest [1]. As a result, transit agencies across the country are seeking to undertake fundamental transformations in our infrastructure, technology, and problem-solving approaches to further improve the current transportation system [2]. A key aspect of this transformation is the enhancement of methodologies that can accurately predict delays and occupancy, thereby improving the reliability and efficiency of transit systems. This ability also enables transit agencies to optimize their services leading to enhanced customer satisfaction. Many agencies now collect, analyze, and provide real-time data for all of their public bus fleets, which includes information about occupancy levels collected through automated passenger counters (APC) and spatial position collected through Global Positioning System (GPS) [3, 4]. Transit agencies use this data to **(a)** provide real-time information and short-term forecasts about expected arrival and departure times to enable commuters to make informed decisions about their travel, **(b)** use the predictions to optimize and plan future service, e.g., bus fleets use forecasting models to optimize headways (i.e., the distance between successive buses on the same route), and **(c)** plan long-term bus schedules [5].

Challenges: Developing delay and occupancy forecasting models is far from trivial. The major challenges in developing prediction models are the sparsity of data and the presence of noise. Data collected through automated passenger counters and GPS devices are often sparse, featuring significant gaps in the data that must be either imputed or filled. Data sparsity makes it challenging to develop robust prediction models. Moreover, mid-sized cities typically do not have a vast network of public transit routes/buses to collect additional data; the sample size is generally not enough for prediction models to perform and generalize well. Buses record time information at designated stops along trips called "timepoints". Here only a bus's temporal information is recorded, which differs from the time entries when collecting boarding and alighting data. These cause mismatches in arrival times, making it more difficult to model delays accurately. Also, noise in APC and GPS devices makes it more difficult to detect patterns and trends as incorrect readings can happen because of human error (incorrect labeling or data entry errors), and technical issues (faulty sensors or software bugs), which can result in incorrect counts or missing data.

State of the art: There has been considerable work done to solve the challenges of noise and sparsity in delay and occupancy prediction. One approach estimates short-term delays in buses through the use of General Transit Feed Specification (GTFS) data and historical patterns. Sun et al. leverage shared route segment networks and multi-task deep neural networks to address data sparsity and improve the accuracy of severe delay prediction. Similarly, long short-term memory (LSTM)-based travel time prediction has been used to accurately predict road

[*] These authors contributed equally to this work.

segment travel times by incorporating contextual information such as weather data, public holidays, and traffic speed [7]. This approach outperforms traditional methods such as moving averages, linear regression, and support vector machines.

For predicting occupancy, models such as negative binomial regression and random forest have been built using context-specific information related to bus trips and road segments [8]. These models mainly utilize GTFS data generated by transit agencies. Talusan et al. employ a multi-step process including cleaning and merging real-time data from multiple sources, and building various classification models (random forest, XGBoost) and LSTM models to enhance transit ridership predictions [9]. The problem has been explored in both classification and regression settings [10, 11].

Research Gaps and Motivation: Despite several efforts to effectively predict delay and occupancy, learning from temporally sparse data remains a challenge, and public transit agencies often resort to ad-hoc decision-making approaches. We hypothesize that there are occupancy and delay are correlated, and oftentimes, the same set of (possibly abstract) features determine the realizations of these random variables. Capturing these broad interrelated patterns can tackle data sparsity—each "task" can learn from data and feature abstractions from the other task (we refer to learning data-driven abstractions as tasks, as is common in the machine learning literature). Training separate models for each task, as commonly done in prior work, ignore generalizable information that is not explicitly modeled in the feature space. To address these limitations, a more integrated approach is needed that can capture the broader patterns that govern both bus delays and occupancy.

Contributions: In this paper, **(1)** we propose a multi-task learning model that captures broad patterns governing both occupancy levels and delays to produce more accurate and reliable predictions. Our approach also mitigates the challenges of sparsity, noise, and the lack of enough samples by using the information from related tasks to improve the prediction performance. **(2)** We show that APC data is inherently noisy and discuss how it must be preprocessed before it can be used in any machine learning modeling. We highlight that addressing transit issues in the real world involves the task of gathering, refining, and merging data from multiple sources, each with different formats and levels of accuracy. To accomplish this, we utilize a combination of APC data and GTFS along with contextual information like weather.**(3)** Finally, we conduct a systematic study of the transit operations in Chattanooga using the APC and GTFS data provided by our partner transit agency Chattanooga Area Regional Transportation Authority. We then train an LSTM-based Multi-task learning (MTL) model on the processed data to predict occupancy and delay. Our approach is described in figure 1.

We evaluate the MTL model using precision, recall, and F1-score for delay and occupancy. We show that the proposed MTL approach outperforms state-of-the-art single-task learning (STL) models in terms of accuracy and robustness, even when faced with the challenges of data sparsity, noise, and lack of enough samples. Moreover, we also show that our pre-trained MTL model trains better on fewer samples of new routes as compared to pre-trained STL models shown in figure 2. By considering these two variables as interrelated problems, we are able to improve the accuracy of the predictions and provide a more robust solution to this complex problem.

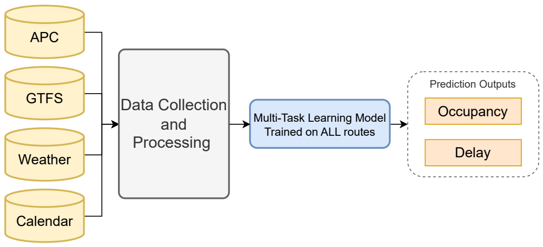

APC
GTFS
Weather
Calendar
Data Collection and Processing
Multi-Task Learning Model Trained on ALL routes
Prediction Outputs
Occupancy
Delay

Fig. 1: Multi-Task Learning Toolchain

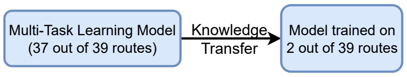

Multi-Task Learning Model (37 out of 39 routes)
Knowledge Transfer
Model trained on 2 out of 39 routes

Fig. 2: Transfer Learning Model

II. Related Work

Optimizing public transit schedules and efficiency is a research area that has gained significant attention in recent years. Several studies have been conducted to explore different techniques and methods to improve the efficiency and reliability of public transportation systems. In this section, we discuss related works that have contributed to this field and how they addressed predicting delay and occupancy separately.

In the domain of predicting delay in transit schedules, Basak et al.[5], and Ou [7] proposed data-driven approaches that use machine learning techniques to predict passenger demand, travel time, and bus arrival time. These methods have shown promising results and can help to improve the efficiency and reliability of public transportation systems. However, these studies did not address the issues of data sparsity.

On the other hand, there is a significant amount of research being conducted to develop effective methods for understanding the factors that influence the occupancy level of transit buses, as well as predicting occupancy at the stop level. Arabghalizi and Labrinidis [8] proposed data-driven bus crowding prediction models that use contextual information such as weather, time of day, and special events to predict bus crowding levels accurately. These models can help to improve passenger experience and reduce overcrowding. Talusan et al. [9] proposed a day-ahead and same-day ridership level prediction model that utilizes machine learning techniques to predict ridership levels accurately using noisy APC data. This model can help transit operators to adjust schedules and resources based on expected ridership levels. Zhang et al. [11] utilized crowdsensing and semantic trajectory mining to predict passenger demand and travel patterns accurately. This method can help to improve transit planning and management. The aforementioned methods are certainly useful for comprehending the various factors that contribute to the occupancy levels of transit buses, as well as forecasting these levels for upcoming trips. However, it should be noted that these approaches may not be applicable to all transit data, especially those that lack adequate samples or have limited data available. This is because the effectiveness of these methods largely depends on the availability and quality of the data, and in cases where the data is sparse or insufficient, these techniques may not yield accurate or reliable results. Additionally, these studies similar to the ones mentioned above do not account for the correlation with delays in the buses that could potentially improve the accuracy.

Some studies, such as Sun et al.[6], have utilized multi-task learning to overcome data sparsity in predicting multiple transit performance outcomes simultaneously. This approach can help to address the interactions between different factors and improve the accuracy and robustness of public transit optimization methods. Although these methods successfully address the issue of insufficient data in predicting delay and occupancy, they fail to consider the larger underlying patterns that dictate delay and occupancy levels, which are necessary to generate dependable and precise outcomes.

Overall, these studies demonstrate the potential of data-driven and machine learning-based approaches for optimizing public transit systems. By utilizing historical and real-time data, these methods can help improve transit efficiency, reduce congestion, and enhance the passenger experience. One common challenge faced by prior studies is data sparsity. This occurs when there is a lack of available data for specific locations or periods, making it difficult to predict transit demand accurately. While some works attempt to address this issue of data sparsity, the resulting models are not generalizable to models predicting occupancy levels. Furthermore, some studies have only focused on predicting a single aspect of transit performance, such as travel time or occupancy, and have not considered the interactions between different factors. This is where multi-task learning can be useful, as it allows for the simultaneous prediction of multiple outcomes. A model that predicts both delay and occupancy can take into account how delays affect passenger demand and crowding levels and vice-versa. By addressing the issue of data sparsity and leveraging multi-task learning, future research can improve the accuracy and robustness of public transit optimization methods.

III. Problem Statement

Our primary objective is to predict occupancy and delay at particular stops in a specific transit route. Specifically, we want to estimate the occupancy level and delay of buses at the next stop after it has crossed a certain number of stops. Our problem consists of a set of buses traveling on a set of assigned routes R, where $r \in \mathcal{R}$ denotes an arbitrary route. Each route r consists of n_r number of stops $\{s_1, \cdots s_{n_r}\}$. We denote the set of all stops by $\mathcal{S}$. A vehicle visits a subset of these stops for one trip. Given a bus traveling on a route r and having passed through p stops (where $p \leq n$), the delay and occupancy prediction problem deals with estimating their realizations at $\{s_{p+1}, s_{p+2}, .., s_{p+t}\}$ where t marks the number of stops of the route we want to predict. In this paper, we focus on estimating delay and occupancy at the next stop, i.e., at s_{p+1}. Our approach is generalizable though, and we also show experimental results for stops further ahead in the route.

TABLE I: Symbols table

Symbol	Description
$\mathcal{R}$	Set of routes
r	A single route
$\mathcal{S}$	Set of all stops
s_i	Stop i
$\mathcal{Y}$	Output space of Occupancy and Delay
(o, d)	Occupancy and Delay level of a vehicle at a particular stop
$\hat{\mathcal{Y}}$	Prediction of Occupancy and Delay
$\mathcal{O}$	Set of possible categories for occupancy
$\mathcal{D}$	Set of possible categories for delay
$\mathcal{X}$	Sequential Data containing stop and contextual attributes

We discretize occupancy and delay into a set of categories. We chose classification models for various reasons. Firstly, our partner agency CARTA was more interested in the levels of delay and occupancy have a better understanding of performance of buses. Another reason for choosing a classification model is that it is more intuitive and easy to communicate the percentage of times that a particular bus or train route is likely to be delayed or full. This is more helpful for transit planners who are looking for actionable insights.

Let $\mathcal{D}$ represent the set of possible categories for the delay, i.e., $\mathcal{D} = \{$Very Early, Early, On-Time, Late, Very Late$\}$, and $\mathcal{O}$ represents the set of possible categories for occupancy, i.e., $\mathcal{O} = \{$Low, Medium, Medium-High, High, Very-High$\}$. Then, we use $\mathcal{Y} = \{(o, d) \mid o \in \mathcal{O}, d \in \mathcal{D}\}$ to denote the "joint" output space of our problem. A particular $y \in \mathcal{Y}$ denotes a pair of occupancy and delay. Our goal is to learn an estimate $\hat{y} = f(o, d \mid X, \theta)$, where f is a function approximator, X denotes the input features, and θ is the set of model parameters that we seek to learn.

IV. Data Collection and Processing

In a complex real-world problem such as occupancy and delay prediction, it is imperative we collect, harmonize, and clean heterogeneous data. Before describing our model, we first focus on three major steps: data collection—the sources from which data was collected; data preprocessing and data merging— cleaning of the data, outlier removal, and merging of data from various sources; and feature selection—the features used and how they were derived.

A. Data Collection:

We collected data from various sources with the help of sensors and publicly available datasets; all datasets were merged with APC data from the city of Chattanooga.

- Automatic Passenger Counting (APC): Each entry in the APC is a log of the current state of the bus at a stop s_n on a trip. This log also includes scheduled and actual stop arrival times.
- Weather: Numerical values of weather data (precipitation, humidity, temperature) are collected from DarkSky API [12]. The weather data set used in this study has a

temporal resolution of one hour and covers a period of three years.

- General Transit Feed Data(GTFS): This data set contains schedule and geographic transit information for a trip [13]. we leverage the information from the aforementioned dataset to impute missing values in the scheduled data in the Automatic Passenger Counter.
- Calendar: It includes all events that can affect traffic such as public holidays and school breaks.

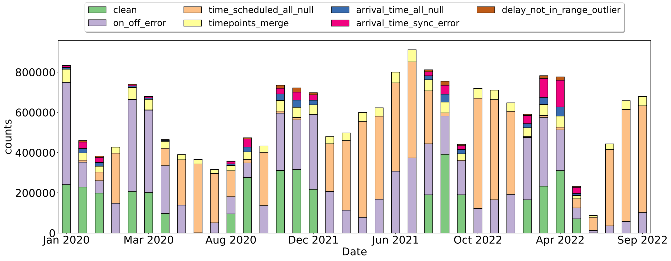


Fig. 3: Count of issues faced when dealing with APC data.

B. Data Preprocessing and Data Merging:

We used three years (2020-2022) of APC transit data from Chattanooga. The raw APC data consisted of 20.7 million records. However, this data is very sparse and noisy as shown in Fig. 3. Our original raw APC data consisted of 19.2 million stops with no values for the "scheduled time" attribute, which is imperative for the calculation of delay (naturally, we cannot compute delay without the expected arrival time). The missing data points in the APC dataset were imputed using GTFS data from the transit agency. . Next, the "on-off error" (on means passengers boarding and off means passengers alighting) was calculated for each vehicle, transit date, and block using equation 1, and all those blocks were filtered out where the "on-off error" was > 0.2.

$$\text{on\_off\_error} = \frac{(\text{total\_ons - total\_offs})}{\text{total\_ons}} \tag{1}$$

Another challenge encountered in the data analysis process was the presence of records with timepoints, indicating that buses arrived earlier than recorded at a specific stop and stayed at the stop to depart at their scheduled time. To address this, timepoint data points were merged with data corresponding to those stops, provided that they shared the same "time scheduled," "trip ID," and "transit date" attributes. Furthermore, the data had incorrectly reported arrival times for certain stops, where the stop entries were not in chronological order. To ensure data accuracy and quality, all such trips were filtered out from the dataset. The delay was calculated using equation 2, as part of the pre-processing steps. Finally, to ensure the validity of the data analysis, all trips meeting the following conditions were removed: Scheduled time or arrival entries are null, occupancy at any stop is null or negative, and trips whose delay is $> |15|$ minutes (based on empirical distribution of data, see below). We also added weather conditions for the specific transit date using Darksky, public holidays, and school breaks.

$$\text{delay} = \text{time\_actual\_arrive} - \text{time\_scheduled} \tag{2}$$

Finally, occupancy and delay were binned into groups of 5 each. Occupancy was classified as: Very Low: ≤ 3, Low : 4–6, Medium : 7–55, High 56–75 and Very High: ≥ 76. Delay was classified as: Very Early: -15 minutes to -9 minutes, Early: -9 minutes to -3 minutes, On-Time: -3 minutes to 3 minutes, Late: 3 minutes to 9 minutes, Very Late: 9 minutes to 15 minutes. A negative delay indicates that the bus arrived at a stop before the corresponding time scheduled, and a positive delay indicates that the bus arrived late at the stop. The binning was done with the help of mean and standard deviation of delay; we chose 15 minutes as our ideal starting and ending points as they were two standard deviations away from the mean covering 95.4% of our data. The classification was done due to the sparse nature of our data. Our approach is generalizable to other forms of discretization, such as categorizing delay by using congestion or categorizing occupancy by using the "crowdedness" factor [9].

C. Feature Selection:

We chose a total of 17 features after integrating APC, GTFS, weather, public holidays, and school holidays datasets. To convert these features into values that our machine learning model can understand we made use of three different encoders over the 17 features: one-hot-encoding, label encoding, and numerical feature encoding. One-hot encoding was applied to route ID direction, public holiday, school holiday, and day of week features; this was done to convert string values into numerical binary values. Further, features such as stop ID, stop sequence, month, hour, day, year, binned delay and binned occupancy were encoded into corresponding categorical values using label encoder. Finally, numerical values such as temperature, humidity, precipitation and scheduled headway were scaled using min-max scaler.

V. Prediction Models

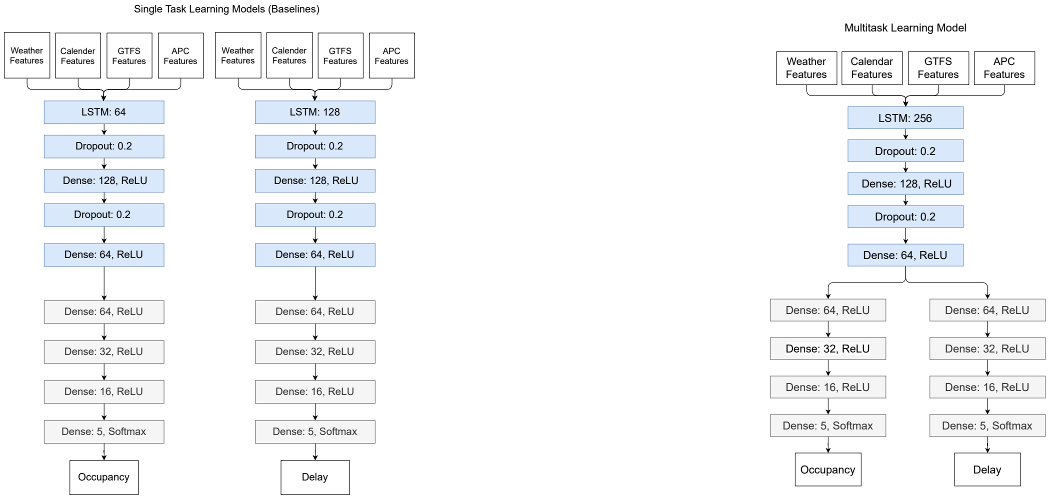
As discussed in the introduction, this paper focuses on the prediction of occupancy and delay for public transit busses and our goal is to help transit agencies/applications (such as Transit, Google maps) provide more accurate arrival time and occupancy data to their users. We propose an MTL-based LSTM model. Our model was compared to single-task LSTM models used for occupancy and delay prediction. The objective of the models was to minimize the categorical loss for our binned occupancy and delay.

A. Multitask Learning

Multi-Task Learning (MTL) is a machine learning approach that involves training a single model to learn multiple tasks simultaneously, offering several advantages over traditional single-task learning methods [14]. A key advantage of MTL is its ability to enhance model accuracy by exploiting shared

TABLE II: Data Features and Sources

Dataset	Features	Source	Frequency	Type	Description
Transit	Transit date	APC	Variable	Temporal	Date when trip takes place
	Route ID	GTFS	Variable	Spatio-temporal	Unique route identifier given by GTFS
	Route direction name	APC	Variable	Spatio-temporal	Name of route direction
	Scheduled headway	APC	Variable	Spatio-temporal	Duration between buses headed in the same route and direction at a stop
	Load	Derived	Variable	Spatio-temporal	Total occupancy at the stop (after alights and boards)
	Stop sequence	GTFS	Variable	Spatio-temporal	Trip sequence number for a stop
	Stop ID	APC/GTFS	Variable	Spatio-temporal	Stop Identifier
	Delay	Derived	Variable	Spatio-temporal	Difference between arrived time and scheduled time of the bus
	Time Window	Derived	Variable	Temporal	Window of time of the day (each window is 15 min each)
Weather	Temperature	Darksky	1 hour	Spatio-temporal	Recorded temperature
	Precipitation intensity	Darksky	1 hour	Spatio-temporal	Recorded humidity
	Humidity	Darksky	1 hour	Spatio-temporal	Amount of precipitation
Holidays	School breaks	Calendar	1 day	Temporal	Scheduled school breaks and holidays in a calender year
	National holidays	Calendar	1 day	Temporal	National holidays



(a) Single task learning models

(b) Multitask learning model

Fig. 4: Models Architecture Multitask Learning Model and Single Task Learning Models for occupancy and delay.

information across tasks, thereby enabling the model to generalize better [14]. Furthermore, MTL can help mitigate overfitting by encouraging the model to learn more generalizable, robust, and universal representations that can be applied to a wide range of related tasks [15].

The superior performance of MTL models compared to their single-task counterparts is largely attributed to the fact that MTL models are particularly effective in scenarios where the tasks being performed are related to each other [15]. This characteristic of MTL can be leveraged in various domains, such as natural language processing, computer vision, and speech recognition, among others, to simultaneously perform multiple tasks while also improving overall model performance [14].

In our implementation of MTL, the shared branch consists of one LSTM layer and two dense layers with ReLU as the activation function and two dropout layers. The branched layers have three dense layers each with ReLU [16] and finally a dense layer with softmax activation function to get the probabilities of each class for delay and probability as shown in 4b. The precision, recall, and F1-score of the MTL model are compared to those of the single-task models to assess the MTL model's performance. The single-task models have a similar model architecture as that of MTL as described in figure 4a. This model takes sequential data and features from APC, GTFS, weather, and calendar as inputs and predicts occupancy and delay simultaneously.

B. Transfer Learning

Transfer learning is a technique where the knowledge learned by a pre-trained model is transferred to a new model for a new task, the advantage is that the transferred knowledge is used as a starting point, instead of training from scratch [17]. Another advantage of this technique is that the amount of training data required is reduced [17].

We also analyze the amount of data required to accurately predict the occupancy and delay for routes that were not present in the pre-trained model. Knowledge is transferred from the pre-trained model, which is then used to train a new model on only a specific route, by configuring the percentage of trips used for training (1%, 2%, 5%, 10%, 15% trips from untrained routes). Finally, we compare our results with that of our baseline single-task models to showcase the benefits of using transfer learning with our Multi-task learning model. The idea behind this approach is, if in the future a transit agency adds a new route, we can determine the amount of

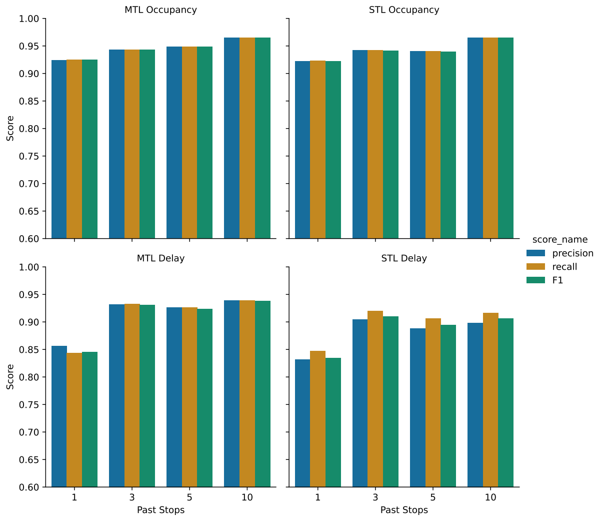


Fig. 5: Precision, Recall, and F1 scores comparison of MTL and STL models for predicting delay and occupancy.

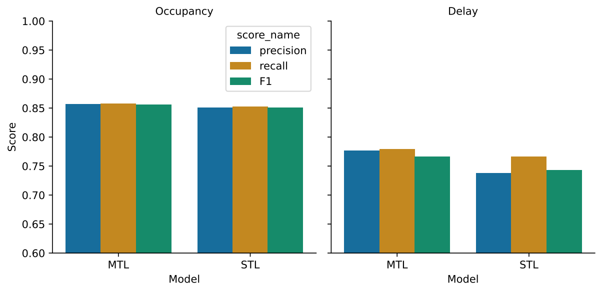


Fig. 6: Precision, Recall, and F1 comparison of MTL and STL models for predicting next 10 stops, using 5 stops as input.

data that is required to predict occupancy and delay for that newly added route.

VI. Results and Discussions

After preprocessing and merging features from different datasets, we train and test on the remaining 2.75 million data points ranging from 2020-01-01 to 2022-06-12. We use 90% of the data for MTL Model training and the remaining 10% for evaluation. We withheld 10% of the training dataset to be used as the validation set. We use this validation set during the training.

We group the data into trips and uniformly at random assigned them into either training, validation, and testing sets. We do not account for the dates as a criterion to divide the dataset, the goal is to make our model more robust to seasonal trends. We also wanted to avoid using data collected during the pandemic as the majority of our training data, which would have been the result of simply dividing the dataset sequentially. Transit activities during this period might not be suitable to use as training for predicting the current state of transit occupancy and delays.

A. Hyperparameter Tuning and K-fold Cross-Validation

We performed a random grid search with K-fold cross validation for our MTL model using the Python library Ray [18]. For K-fold cross validation, we set K=5. We assessed and validated the performance of hyperparameters configurations using holdout data. In this hyperparameter search, we test shared hidden LSTM layer widths of {64,128,256}, batch sizes of {32,64,128} and the learning rates of {0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05}. Sparse Categorical Cross-entropy (SCC) is used for the loss function and Adam algorithm as optimizer. The shared layers also include 2 dense layers of width 128 and 64, respectively. We use 3 task-specific Dense layers for occupancy and delay with configurations 64, 32, and 16 each. The best-performing configuration is shown in figure 4b, which consists of one shared hidden LSTM layer of 256 neurons followed by 2 shared hidden Dense layers of 128 and 64 fully connected neurons with ReLU activation functions, and 3 task-specific hidden layers of 64, 32, and 16 hidden neurons respectively[16]. We found the batch size of 256 and learning rate of 0.001 to be optimal for this model.

For our baseline models, we created two separate state-of-the-art LSTM models to predict delay and occupancy. The entire model architectures for each are shown in figure 4a. The selected configurations for these models were also decided after performing a random grid search over the same hyperparameters mentioned above.

B. MTL Evaluation

Since our MTL model performs multi-label classification, we use the precision, recall, and F1 score as our evaluation criteria to judge the performance of our MTL model compared to single-task baseline models. Since our data is both sparse and our labels imbalanced, precision and recall allows us to correctly validate our results.

We first train the two MTL models using the hyperparameter configurations found during the grid search. The model is trained on the randomly sampled trip data set. For evaluation, we randomly sampled around 900 unique trips from the test set, which have at least 10 stops. We aggregated these trips in 15-minute time windows of the day. We use both MTL and STL models to predict the occupancy level, and delay of buses for the next stop after looking at the past 1, 3, 5, and 10 stops. Since it is multi-label classification problem, we use precision, recall and F1 score as our evaluation criteria to assess how well the model has perform. The results are presented in figure 5. In terms of predicting occupancy, our MTL model is consistently outperforming STL model marginally on any number of past stops. However, it is consistently outperforming STL model for delay prediction by as much as 4% in all metrics. We also evaluated the performance of our models in predicting the subsequent 10 stops in a trip, given only the data from the initial 5 stops. We show in figure 6, that our proposed MTL is performing better than its STL counterparts in predicting both the delay and occupancy of the buses 10 stops into the future.

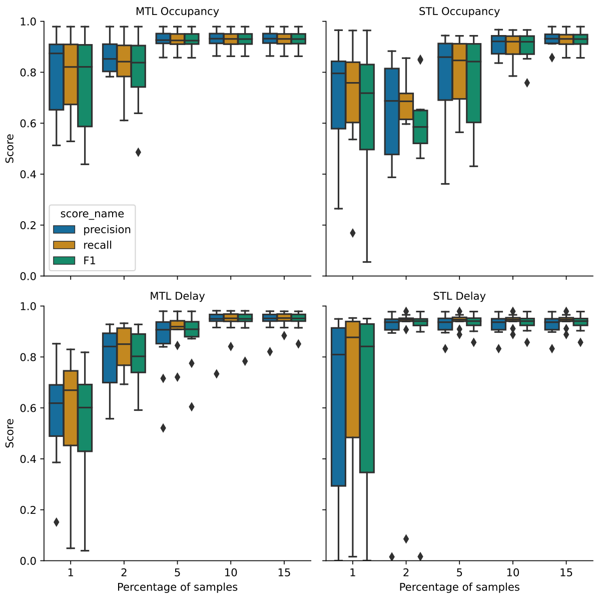


Fig. 7: Precision, Recall, and F1 scores comparison of MTL and STL models in predicting next stop, trained on different percentages of newly added routes.

C. *Transfer Learning Evaluation*

To evaluate the performance of MTL, we developed a transfer learning model based on the MTL model to predict delay and occupancy on an entirely new route by leveraging the knowledge learned from the pre-existing set of routes. First, we trained our MTL model and baseline STL models on the entire training dataset excluding two routes that were randomly selected. We used transfer learning to transfer the shared hidden layer weights of these models to a separate model. This new model is then retrained on the two excluded routes using varying percentages of available data (1%, 2%, 5%, and 10%). We repeated this experiment ten times to capture the general trend. We evaluated the performance of these models on the new routes. In this evaluation, we test the performance of the MTL model in predicting bus delay and occupancy at the 6th stop after the vehicle passes through 5 stops, and compare it to its STL counterparts. The results are presented in figure 7.

Our findings suggest that the transfer learning model based on MTL learns more efficiently given fewer samples for training compared to the transfer learning models based on STL when it comes to predicting occupancy. The MTL model achieves a 95% F1 score for delay prediction with just 5% of the additional route data with narrow distribution, whereas the STL model requires 15% of the additional route data to achieve similar results, indicating significant differences in their performance for occupancy prediction. However, we also observe that, in the case of delay prediction, the pre-trained STL model learns much quicker as compared to our MTL model, indicating the STL performs better for delay prediction if it is given 5% of the data for newly added routes while MTL model requires 10% of the data to achieve the same performance. It is important to note that the distribution of prediction scores for delay is very large for both STL model and MTL models when these models are trained on less than 5% of the samples. This indicates at least 5% of the data is necessary for even STL model to perform predict reliably for the new routes.

D. *Discussion*

Our MTL framework demonstrated better performance in predicting occupancy and delay, as measured by F1 scores, when compared to the baseline STL models. Specifically, the MTL model outperformed its STL counterpart in predicting delay, while performing similarly for occupancy.

Secondly, our MTL model exhibited the ability to produce reliable predictions for occupancy given fewer samples of new, unseen data, thus overcoming the challenge of data sparsity. We achieved better or comparable F1 scores by training our pretrained MTL model on fewer samples of newly added routes when compared to the STL model. We strongly believe that the MTL approach mitigates the challenges in the case of occupancy posed by data sparsity and the lack of sufficient samples by leveraging the generalized representation of knowledge in the shared layers. This allows our model to make accurate predictions for occupancy, even when there are limited samples available for certain routes. In other words, the shared layers enable the model to learn features that are relevant across different routes, which reduces the impact of data sparsity and the lack of samples for individual routes. These observations are particularly relevant as we hypothesized that MTL can address the data sparsity challenges by capturing the interrated patterns of these two tasks.

Thirdly, as far the the prediction of delay on the new data is concerned, we have observed that STL is better at learning on fewer samples of data as STL takes 5% of the data of new routes to predict optimally as compared to MTL which takes 10%. Moreover, the performance of MTL and STL model becomes similar for delay when both models are trained on 10%-15% of the data. We believe that the reason for the differences between such performance is MTL is still in the process of accounting for the correlation between occupancy and delay in order to predict delay for newly added routes. After it has completely accounted the correlation during training, it starts performing better as compared to its STL counterpart.

Finally, we observed a significant improvement in predicting delay levels using the MTL framework as compared to occupancy levels. Furthermore, our MTL model showed the same trend when we used it to predict occupancy and delay of the next 10 stops after it has passed through the first 5 stops, indicating that delay as a feature depends on the occupancy levels of previous stops, while occupancy acts independently. This finding is reasonable since a higher occupancy level on previous stops translates to more people boarding the bus, ultimately causing delays. Furthermore, delay ranging from -15 min to +15 min in our processed data (outliers were eliminated) does not significantly affect occupancy levels.

These results suggests that people are generally willing to wait for up to 15 minutes for buses.

VII. Conclusion

Urban areas heavily depend on public transit as a primary mode of transportation, making the efficiency of transit systems crucial for the daily commute of millions of individuals. Passengers rely on accurate information to plan their schedules and travel comfortably, while transit agencies seek to optimize their public bus services. Predicting occupancy and delay accurately is therefore of great importance. However, state-of-the-art methods for predicting these features face challenges due to data sparsity and insufficient samples in the transit domain. To address these challenges, we proposed an MTL approach that outperforms the current state-of-the-art methods, which predict occupancy and delay separately. Our MTL model considers the correlation between the two features, thus providing a more comprehensive view of the transit system. We showed that our MTL model outperformed the baseline STL models in predicting occupancy and delay, while overcoming data sparsity.

Our approach significantly improves the prediction of delays compared to its STL counterpart. Furthermore, our MTL model requires only 5% data from newly added routes to achieve the same level of accuracy for occupancy as a model trained on a subset of the routes, whereas STL models require three times as much data. Our findings suggest that a multitask learning approach is a promising tool for accurately predicting transit occupancy and delay in the presence of sparse and noisy data.

References

[1] Freemark, Yonah , "US Public Transit Has Struggled to Retain Riders over the Past Half Century. Reversing This Trend Could Advance Equity and Sustainability." The Urban Institute, 2021.

[2] U.S. Department of Transportation, "U.s. department of transportation FY2022-26 strategic plan," https://www.transportation.gov/sites/dot.gov/files/2022-04/US_DOT_FY2022-26_Strategic_Plan.pdf, 2022, accessed on March 15, 2023.

[3] J. P. Chanchico, P. C. M. Garcia, C. M. Festin, and W. M. Tan, "Waypoint: Online semi-automatic vehicle occupancy data collection system," in *2021 International Conference on Information and Communication Technology Convergence (ICTC)*, 2021, pp. 961–966.

[4] Transportation Research Board and National Academies of Sciences, Engineering, and Medicine, *Using Archived AVL-APC Data to Improve Transit Performance and Management*. Washington, DC: The National Academies Press, 2006. [Online]. Available: https://nap.nationalacademies.org/catalog/13907/using-archived-avl-apc-data-to-improve-transit-performance-and-management

[5] S. Basak, F. Sun, S. Sengupta, and A. Dubey, "Data-driven optimization of public transit schedule," in *Big Data Analytics*, S. Madria, P. Fournier-Viger, S. Chaudhary, and P. K. Reddy, Eds. Cham: Springer International Publishing, 2019, pp. 265–284.

[6] F. Sun, A. Dubey, C. Samal, H. Baroud, and C. Kulkarni, "Short-term transit decision support system using multi-task deep neural networks," in *2018 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2018, pp. 155–162.

[7] Y. Ou, *AI for Real-Time Bus Travel Time Prediction in Traffic Congestion Management*. Cham: Springer International Publishing, 2022, pp. 63–84. [Online]. Available: https://doi.org/10.1007/978-3-030-72188-6_4

[8] T. Arabghalizi and A. Labrinidis, "Data-driven bus crowding prediction models using context-specific features," *ACM/IMS Trans. Data Sci.*, vol. 1, no. 3, sep 2020. [Online]. Available: https://doi.org/10.1145/3406962

[9] J. P. Talusan, A. Mukhopadhyay, D. Freudberg, and A. Dubey, "On designing day ahead and same day ridership level prediction models for city-scale transit networks using noisy apc data," in *2022 IEEE International Conference on Big Data (Big Data)*, 2022, pp. 5598–5606.

[10] R. Silva, S. M. Kang, and E. M. Airoldi, "Predicting traffic volumes and estimating the effects of shocks in massive transportation systems," *Proceedings of the National Academy of Sciences*, vol. 112, no. 18, pp. 5643–5648, 2015. [Online]. Available: https://www.pnas.org/doi/abs/10.1073/pnas.1412908112

[11] N. Zhang, H. Chen, X. Chen, and J. Chen, "Forecasting public transit use by crowdsensing and semantic trajectory mining: Case studies," *ISPRS International Journal of Geo-Information*, vol. 5, no. 10, 2016.

[12] The Dark Sky Company, LLC, "Dark Sky API," https://darksky.net/dev, 2012–2021, accessed: March 15, 2023.

[13] B. McHugh, "Pioneering open data standards: The gtfs story," in *Beyond Transparency: Open Data and the Future of Civic Innovation*. Code for America Press, 2013, ch. 10, pp. 125–135.

[14] M. Crawshaw, "Multi-task learning with deep neural networks: A survey," *ArXiv*, 2020.

[15] Y. Zhang and Q. Yang, "A survey on multi-task learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 12, pp. 5586–5609, 2022.

[16] V. Nair and G. Hinton, "Rectified linear units improve restricted boltzmann machines vinod nair," vol. 27, 06 2010, pp. 807–814.

[17] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2021.

[18] R. Liaw, E. Liang, R. Nishihara, P. Moritz, J. E. Gonzalez, and I. Stoica, "Tune: A research platform for distributed model selection and training," *arXiv preprint arXiv:1807.05118*, 2018.