Comparison of Forecasting Approaches for WMATA Metro Ridership

ECE5424
Rayden Dodd
Virginia Tech

I. Abstract

This project compares the effectiveness of different machine learning models in predicting the WMATA metro ridership per station per hour. We will evaluate the difference between a Baseline, Statistical based Prophet model, Tree based LightGBM, vs Deep learning Temporal Fusion Transformer. By analyzing performance using standard accuracy metrics, we can view improvements over simpler baselines and provide insights into trade-offs of different prediction strategies. Importantly, these prediction accuracy gains can have real world benefits. Better predictions can support a more adaptive transit management, informing operational planning (staffing, capacity allocation), and passenger decision making (identifying least crowded travel times). This work demonstrates the practical role of predictive analytics in building resilient urban transit systems.

II. Specific Aims

- **Forecast** station level ridership by hour for future dates using multiple models.
- **Compare Performance** baseline, Prophet, LightGBM, and TFT.
- **Incorporate exogenous** features such as weather data to influence predictions.
- **Demonstrate applications** of forecasting for transit operations and passenger guidance.
- **Explore** how predictions differ according to the type of station clustering.

III. Background

Public transit is the backbone of urban mobility for many cities, and Washington, D.C. is one of these very transit-oriented cities. The WMATA metro is not small, as it is the second largest metro system in the United States, serving 166 million people a year, the D.C. Metro includes a wide range of rider's: residents, daily commuters, federal employees, students and tourists [1]. Forecasting passenger demand is a critical task for WMATA, as usage fluctuates significantly over time due to recurring seasonal patterns, day-of-week effects, weather shocks, and special events. Accurate forecasts allow agencies to anticipate changes in demand and allocate resources more efficiently, balancing service capacity with rider needs. Without reliable predictions, systems are at risk of overcrowding, underutilization, or staffing inefficiencies, all of which reduce the effectiveness and reliability of public transit.

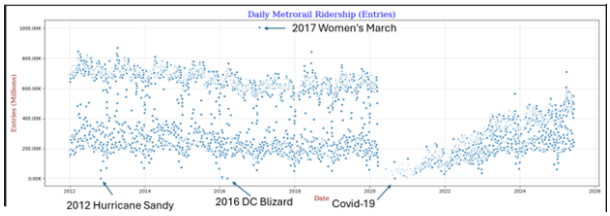

Fig. 1: *Daily Metrorail entries (2012–2025) with notable events annotated (Hurricane Sandy, 2016 Blizzard, 2017 Women's March, COVID-19).*

Traditionally, most transit agencies have relied on older, heuristic based approaches for forecasting ridership methods such as professional judgment, rules of thumb, and simple moving averages. While, these techniques remain common, only about 20% of agencies use regression-based models [2]. While, these baselines capture recurring cycles to some extent, they often fail to adjust quickly to shifts caused by disruptions like severe weather, holidays, or large-scale events. As a result, such methods may underestimate demand in critical situations or overestimate demand during downturns, limiting their utility for adaptive management.

However, with recent advancements in machine learning, modeling techniques provide opportunities to improve ridership forecasting accuracy. Some recent time-series regression model advancements like Facebook's Prophet that came out in 2017 incorporate seasonal and holiday effects in its predictions[3]. 2016 tree-based approaches like LightGBM can learn nonlinear interactions between lagged ridership, calendar effects, and external features such as weather [4]. Or even more recently 2021 deep learning models such as Temporal Fusion Transformers (TFT) have been introduced by Google, leveraging attention mechanisms to capture complex temporal dependencies while also offering some interpretability [5]. These hold opportunities for more robust predictions but are more computationally complicated and don't always give benefits over simpler baselines.

WMATA's metro presents unique challenges due to its combination of commuter-driven weekday peaks, weekend tourism traffic, and seasonal variation. This creates a big separation in station use cases as well with some being commuter hubs, nightlife locations, and event locations. Even modest accuracy gains could inform operational planning (staffing, capacity allocation) and passenger guidance (identifying least crowded

travel times).

IV. Research Design and Method

A. Problem statement

The forecasting problem can be represented as minimizing the mean squared error of prediction (MSEP) as shown in [6]:

$$\text{MSEP} = \mathbb{E}\Big[\big(Y_{t+\ell} - h(Y_1, Y_2, \ldots, Y_t)\big)^2\Big],$$

where $Y_{t+\ell}$is the future value, we want to predict (the true ridership ℓ steps ahead), and $h(\cdot)$is our forecasting function based on past observations of ridership.

The solution to this minimization problem is the conditional expectation:

$$\hat{Y}_t(\ell) = \mathbb{E}(Y_{t+\ell} \mid Y_1, Y_2, \ldots, Y_t, X_t),$$

Where $\hat{Y}_t(\ell)$is the forecast of ridership ℓ steps ahead, the conditioning set $(Y_1, \ldots, Y_t)$represents all past ridership values, and $X_t$denotes exogenous features (weather, holidays, events). This is called the minimum mean squared error (MMSE) forecast, since it gives the best unbiased prediction of the future given past information and auxiliary features.

B. Models we will train to give us our predictions

1) Baseline: Our baseline is a moving average that will look at the previous 4 weeks to help us get a baseline of what ridership should look like.

$$\hat{y}_{s,t} = \frac{1}{4}\sum_{i=1}^{4} y_{s,\,t-7i}$$

This will provide our simpler models to compare the more complex models to.

2) Statistical model: Prophet: This model is represented as ridership as a sum of components [7].

$$y(t) = g(t) + s(t) + h(t) + \epsilon_t$$

where $g(t)$is a piecewise linear or logistic trend, $s(t)$represents multiple time frames like daily, weekly, yearly, and $h(t)$ captures holiday or event effects. Prophet is designed to handle strong periodic structure and irregular events, making it a natural fit for transit data with recurring weekday cycles and holiday effects

3) Tree based Model: LightGBM: LightGBM is a gradient boosted decision tree algorithm, meaning it builds an ensemble of decision trees sequentially, with each new tree correcting the errors of the previous ones. Unlike linear regression, which assumes straight-line relationships, boosted trees can capture nonlinear thresholds. For example, ridership dropping when rainfall passes a threshold. The model remains relatively interpretable compared to deep learning, since feature importance can be extracted to show which inputs (like lagged ridership, weather, or calendar effects) drive predictions. To work effectively, lagged features such as ridership from 1 day or 7 days ago must be engineered as inputs.

The predictive form for gradient boosting is: [8], [9]:

$$F_M(x) = F_0(x) + \sum_{m=1}^{M} \eta\, f_m(x),$$

where each $f_m$is a regression tree, η is the learning rate.

4) Deep Learning: Temporal Fusion Transformer (TFT): TFT is for time series forecasting, it combines encoders with attention mechanisms. The model is able to both capture long-range dependencies and highlight which features matter most at each forecast step. TFT can consider dynamic temporal patterns and provide interpretability through attention weights. The key component is the scaled dot product attention [10], which allows the model to learn which past time steps and features to give the most attention to when making ridership predictions. The formula:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V$$

- Q, K, V= query, key, and value matrices derived from the input ridership sequences.
- d_k= dimension of the key vectors.
- The softmax ensures that the model assigns higher weights to the most relevant features.

C. Model Evaluation

Performance will be evaluated across models using multiple error metrics:

MAE: Mean Absolute Error

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

MAPE: Mean Absolute Percentage Error

$$\text{MAPE} = \frac{100}{N}\sum_{i=1}^{N}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

RMSE: Root Mean Squared Error

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\big(y_i - \hat{y}_i\big)^2}$$

D. Clustering Extension

To explore differences across station types, k-means clustering will be used to average ridership station types. This groups stations into commuter heavy, nightlife tourism, and residential stations. Forecast errors will be compared across clusters to assess whether certain models generalize better to specific station.

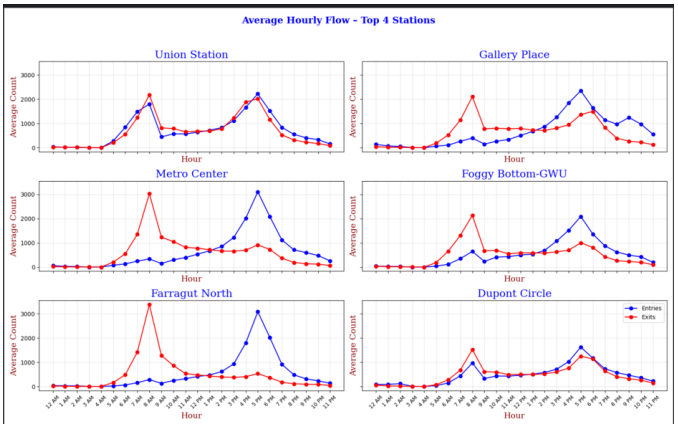

Fig. 2: *Average hourly flow (entries/exits) at representative stations. Clear commuter peaks (Metro Center, Farragut North) contrast with mixed or off-peak usage at others (Gallery Place, DuPont Circle), motivating station type forecasting.*

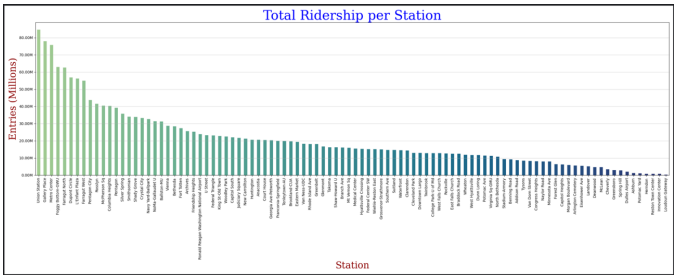

Fig. 3: *Total ridership per station (entries). Long-tail distribution suggests that demand is not uniform across the network, but concentrated at key stations.*

V. Datasets and Experiments

The primary dataset used for this project is the WMATA Metrorail Ridership Data, which provides hourly station-level entries across the Washington, D.C. Metro system. The dataset spans January 2012 – June 2025, contains over 1,000,000 rows with 13 features (Date, Time, Station, Entries, Holiday, Time Period), and is available through the official WMATA Ridership Data Portal: Metrorail Ridership Summary — WMATA. To incorporate external influences, we will merge ridership data with hourly weather data for Washington, D.C. from either Meteostat or Open-Meteo. These provide features such as temperature, precipitation, snowfall, and other climate variables that may explain short-term ridership fluctuations.

A. Experimental Setup

Rolling Origin Cross-Validation: To mimic real time forecasting, we will use rolling origin evaluation, where the forecast origin moves forward over time and models are retested on moving training windows [11].

Models Compared: Baseline, Prophet, LightGBM, and Temporal Fusion Transformer (TFT),

Evaluation Metrics: Forecast accuracy will be measured with MAE, RMSE, and MAPE, both at the system level and across station clusters.

VI. Timelines

- **Week 1–2 (Oct 19 – Nov 1)**
  - Collect and preprocess WMATA ridership + weather data (+ event data)?
  - Implement baseline and set up evaluation metrics
- **Week 3–4 (Nov 2 – Nov 15)**
  - Train Prophet model and compare against baseline
  - Engineer lagged features and train LightGBM
- **Week 5–6 (Nov 16 – Nov 29)**
  - Implement Temporal Fusion Transformer (TFT)
  - Begin comparative analysis across all models (Baseline, Prophet, LightGBM, TFT)
- **Week 7 (Nov 30 – Dec 6)**
  - Apply k-means clustering to group stations by type
  - Evaluate model performance differences across station clusters
- **Week 8 (Dec 7 – Dec 12)**
  - Finalize results, write report, make charts and diagrams

VII. Key References

References

[1] American Public Transportation Association, "Ridership report, q4 2024," https://www.apta.com/wp-content/uploads/2024-Q4-Ridership-APTA.pdf, 2025, [Online].

[2] Transportation Research Board, "Guidelines for providing access to public transportation stations, appendix b: Assessment of evaluation tools," https://onlinepubs.trb.org/onlinepubs/tcrp/tcrp_rpt_153AppendixB.pdf, 2011, [Online], see p. B-8, Table B-2.

[3] C. Duong, "facebook/prophet in 2023 and beyond," https://medium.com/%40cuongduong_35162/facebook-prophet-in-2023-and-beyond-c5086151c138, 2023, [Online].

[4] Microsoft Research, "Lightgbm: Publications," https://www.microsoft.com/en-us/research/project/lightgbm/publications/, 2025, [Online].

[5] B. Lim, S. Ö. Arik, N. Loeff, and T. Pfister, "Temporal fusion transformers for interpretable multi-horizon time series forecasting," *International Journal of Forecasting*, vol. 37, no. 4, pp. 1748–1764, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0169207021000637

[6] J. Tebbs, "STAT 520 Forecasting and Time Series: Lecture Notes," https://people.stat.sc.edu/tebbs/stat520/f13notes.pdf, 2013, [Online], see Section 9.1, p. 232.

[7] S. J. Taylor and B. Letham, "Forecasting at scale," *The American Statistician*, vol. 72, no. 1, pp. 37–45, 2018, see Section 3, p. 2. [Online]. Available: http://lethalletham.com/ForecastingAtScale.pdf

[8] GeeksforGeeks, "Gradient boosting in machine learning," https://www.geeksforgeeks.org/machine-learning/ml-gradient-boosting/, 2025, [Online].

[9] Wikipedia, "Gradient boosting," https://en.wikipedia.org/wiki/Gradient_boosting, 2025, [Online].

[10] B. Lim, S. O. Arik, N. Loeff, and T. Pfister, "Temporal fusion transformers for interpretable multi-horizon time series forecasting," *arXiv preprint arXiv:1912.09363*, 2019, see Section 4.4, p. 9. [Online]. Available: https://arxiv.org/pdf/1912.09363

[11] R. J. Hyndman, "Cross-validation for time series," https://robjhyndman.com/hyndsight/tscv/, 2016, [Online].