Predictive Standby Dispatch: Machine Learning-based Smart Elevator Idle Positioning

Student Name	:	TEN YEE CHERN
Student ID	:	0381084
Supervisor	:	Prof. Afizan Azman
Examiner	:	Dr. Syeda Mariam Muzammal

INTRODUCTION


11
ACCESSIBLE RESTROOMS ARE LOCATED IN THE BASEMENT AND ON FLOORS 6, 7, AND 15.
COLLOQUIA AND ACADEMIC NOTICES ONLY
How do brands like Apple and Gucci and Vespa design for 'desirability'?

INTRODUCTION

- Traditional elevator control strategy: Send the nearest and heading elevator to the calling floor, and **stay at where they lastly unload** passengers (closet car algorithm, CCA)
- This makes idle elevators often stay at low-demand frequency floors
- When high-demand frequency floors are making calls, the elevator needs to take time to arrive at calling floors, increasing waiting time
- When buildings become taller, this situation becomes worse

Solution:
Elevator's Predictive Standby Dispatch

- A system that allows the elevator to predict future calls
- Reposition the idle elevator according to the predicted calls
- Therefore, reducing users' waiting time

Problem Statement, Research Questions & Objectives

PROBLEM STATEMENT

- **Waiting Time Gap:** A reactive dispatch control system only sends the elevator once a call is received, even though there are several optimising strategies, but the time the elevator takes to reach the calling floors cannot be eliminated.
- **System's Effectiveness in different scenarios:** The developed predictive system might not improve much in demand-saturated scenarios, as it only repositions the elevator when it is idle.
- **Cost-Benefit balance:** It can be expected that waiting time can be reduced significantly if we send the elevator to stand by at every possible floor, but the cost also increases.

RESEARCH QUESTION

- Can machine learning algorithms reliably predict future hall calls (floor and direction) before they occur?

- Does using prediction to reposition idle elevators (standby dispatch) reduce waiting time compared to baseline in different scenarios (high and low demand)?

- How should the system trade off between decreasing waiting time and minimising repositioning movement/energy cost?

RESEARCH OBJECTIVE

- How should the system trade off between decreasing waiting time and minimising repositioning movement/energy cost?
- Implement and simulate the predictive standby dispatching strategy, and compare its average waiting time (AWT) versus a baseline reactive dispatch strategy in different scenarios (high and low demand).
- Design a framework or strategy to balance between AWT reduction and movement cost.

LITERTURE REVIEW

Domain Research (Elevator Dispatch Optimisation)

- Reactive: CV occupancy-aware (Wang et al., 2021); Dispatch control RL Opti (Crites & Barto, 1995);
- Proactive: Standby floor scoring system(-AWT:~24%) (Tsai et al., 2025); Arrival time prediction based on users' trajectory (Zhang et al., 2022)

Dataset

- All previous similar work using a simulator
- Poisson process (Crites & Barto, 1995) and Gaussian-based (Tsai et al., 2025) arrival rate

Predictor

- Statistical Models: (ARIMA, SARIMA)
  - Easy to implement
  - Computationally efficient and low-cost
  - Design for regression, not suitable for classification task (Fatima & Rahimi, 2024)
- RNNs: (RNN, LSTM, GRU)
  - Able to capture long-term dependencies
  - Gradient explosion/vanishing problem limit dependencies range (Sherstinsky, 2020)
- TCN:
  - Able to capture longer-range dependencies compared to RNNs
  - High training efficiency (D. Kim, 2023)

- Transformers:
  - Able to process extremely long-range dependencies
  - Able to capture global temporal dependencies
  - Outperform RNNs and TCN in a longer range of dependencies (Hall & Rasheed, 2025)
  - Require higher computation and memory (Kong et al., 2025)

Interesting study:

- Encoder-decoder separated LSTM (W. Zhang et al., 2019)
- Hybrid model GRU-TCN (Nanni et al., 2021)

METHODOLOGY

DATA & ENVIRONMENT SIMULATIONS

Arrival Simulation

- A formula to decide the global arrival rate in time (Gaussian Based) :

$$\boldsymbol{\lambda(t)} = \boxed{\boldsymbol{A_1 \cdot exp(-\frac{(t-\mu_1)^2}{2\sigma_1^2})}} + \boldsymbol{A_2 \cdot exp(-\frac{(t-\mu_2)^2}{2\sigma_2^2}) + A_3 \cdot exp(-\frac{(t-\mu_3)^2}{2\sigma_3^2}) + \varepsilon}$$

- Determine the traffic pattern in time:

  Workday:

  - Morning up-peak (7-10:00): predominantly upward movement (lobby/carpark -> upper)
  - Lunchtime peak (11-14:30): mainly inter-floor trips
  - Evening down-peak (16-21:00): predominantly downward movement (upper/carpark ->lobby)
- Weekday/Weekend/Holiday differentiation:
  - Using A_i to control the number of arrival
  - Larger A_i in weekday, lower in weekend and holiday
- Dense Level Control
  - Adjusting A_i to generate two set of data, different in demand level (low and high)

Building Simulation

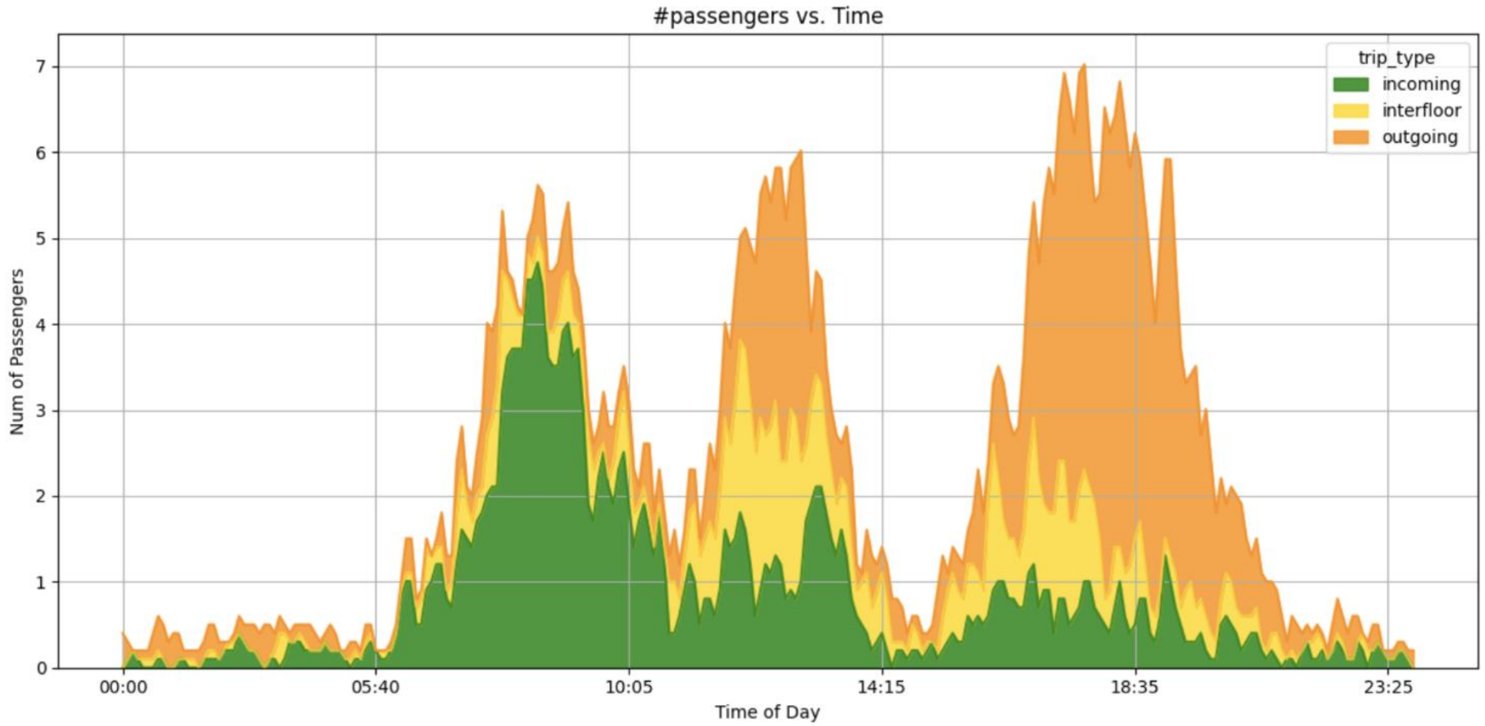
- Two profiles:
  - Low-rise: 10-floor, 1 elevator, no carpark
  - High-rise: 40-floor, 4 elevator, 2-5 floor are the carparks

Elevator Simulation

- Capacity: 15 units
- Moving speed: 2s/floor
- Load/Unload time:

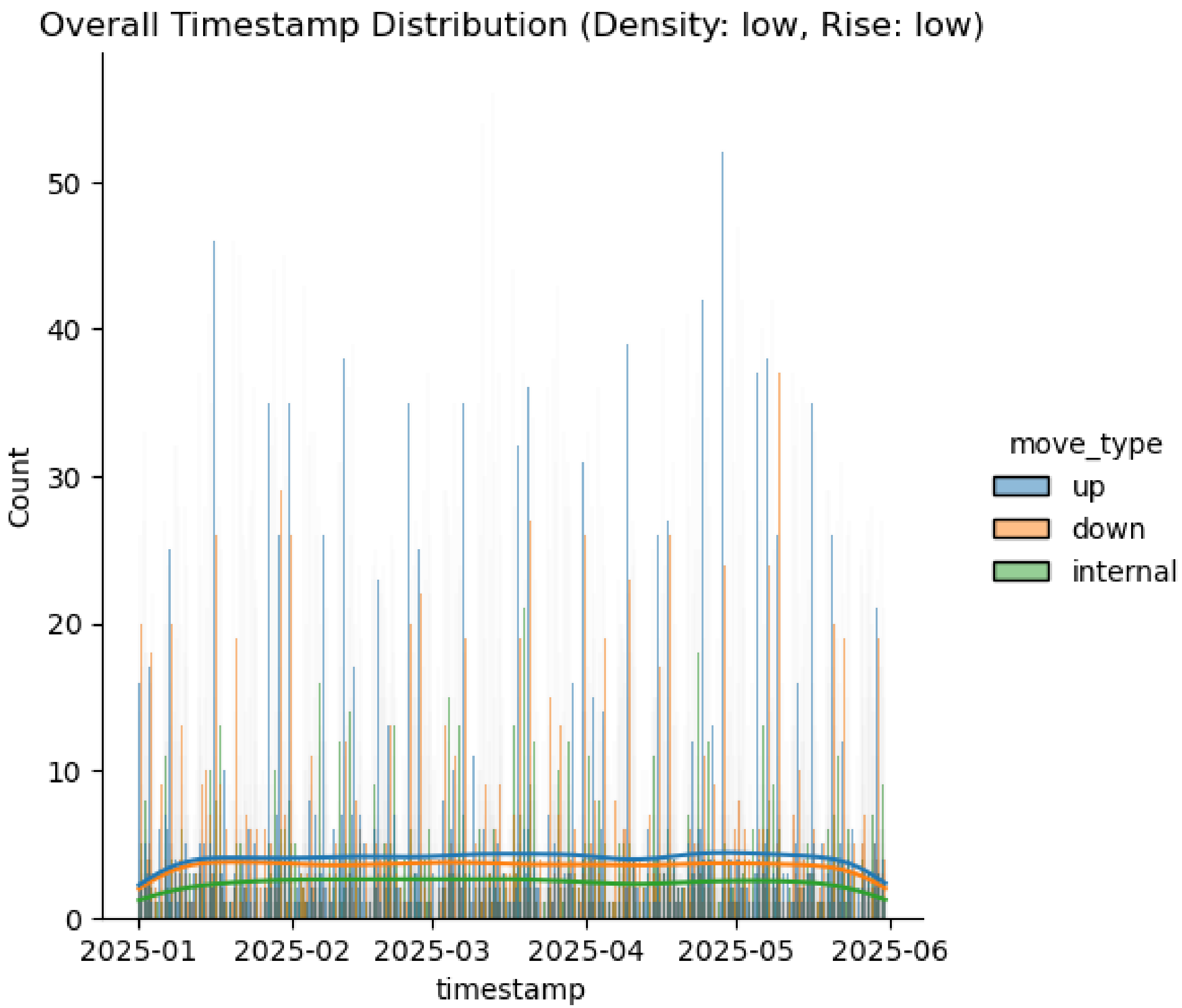
$$T(n_{load}, n_{unload}) = C + \sum_{k=0}^{n_{load}-1} 3 \times 0.5^k + \sum_{k=0}^{n_{unload}-1} 3 \times 0.5^k$$

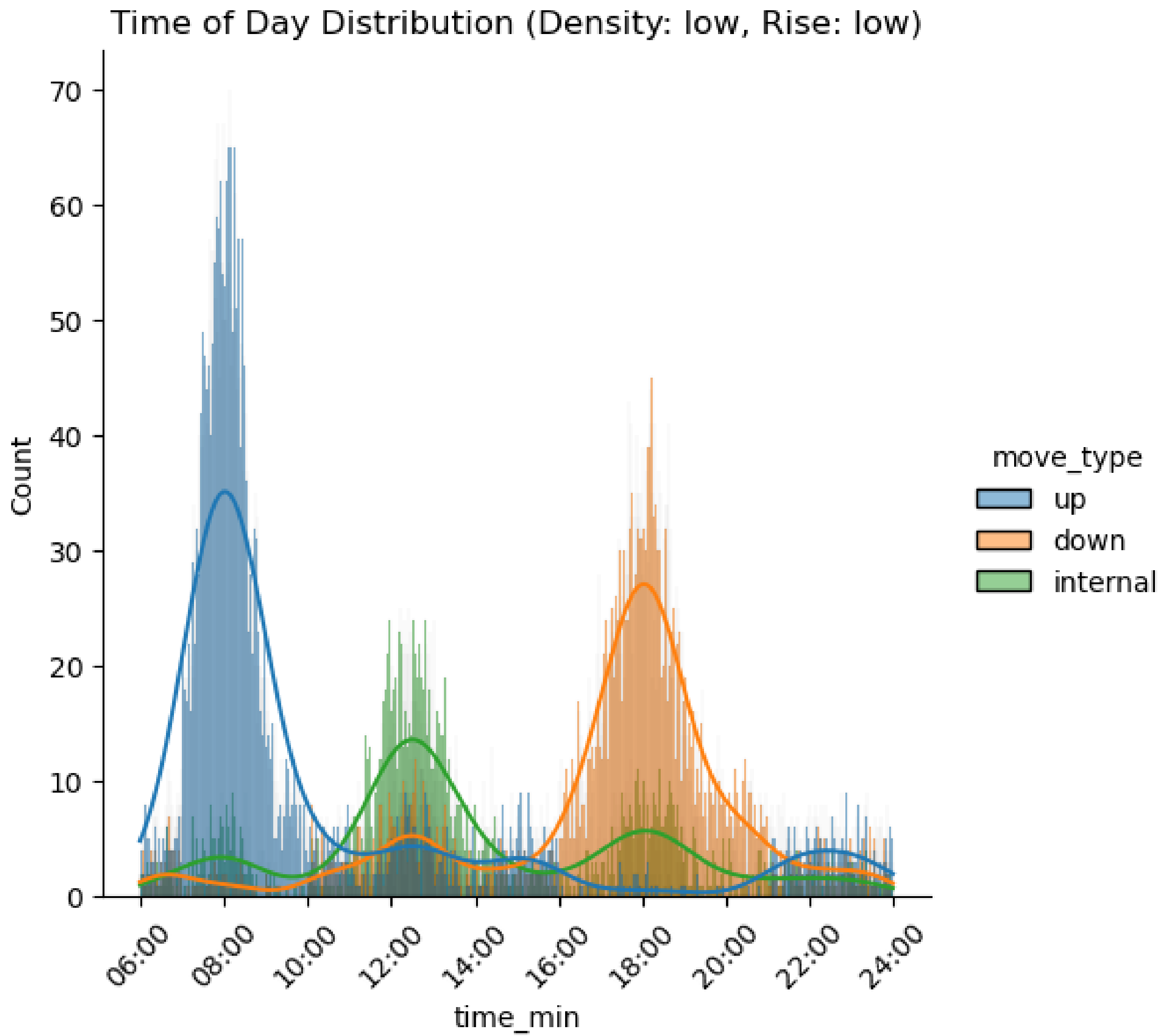
- Dispatch rule: Closet Car Algorithms (CCA)

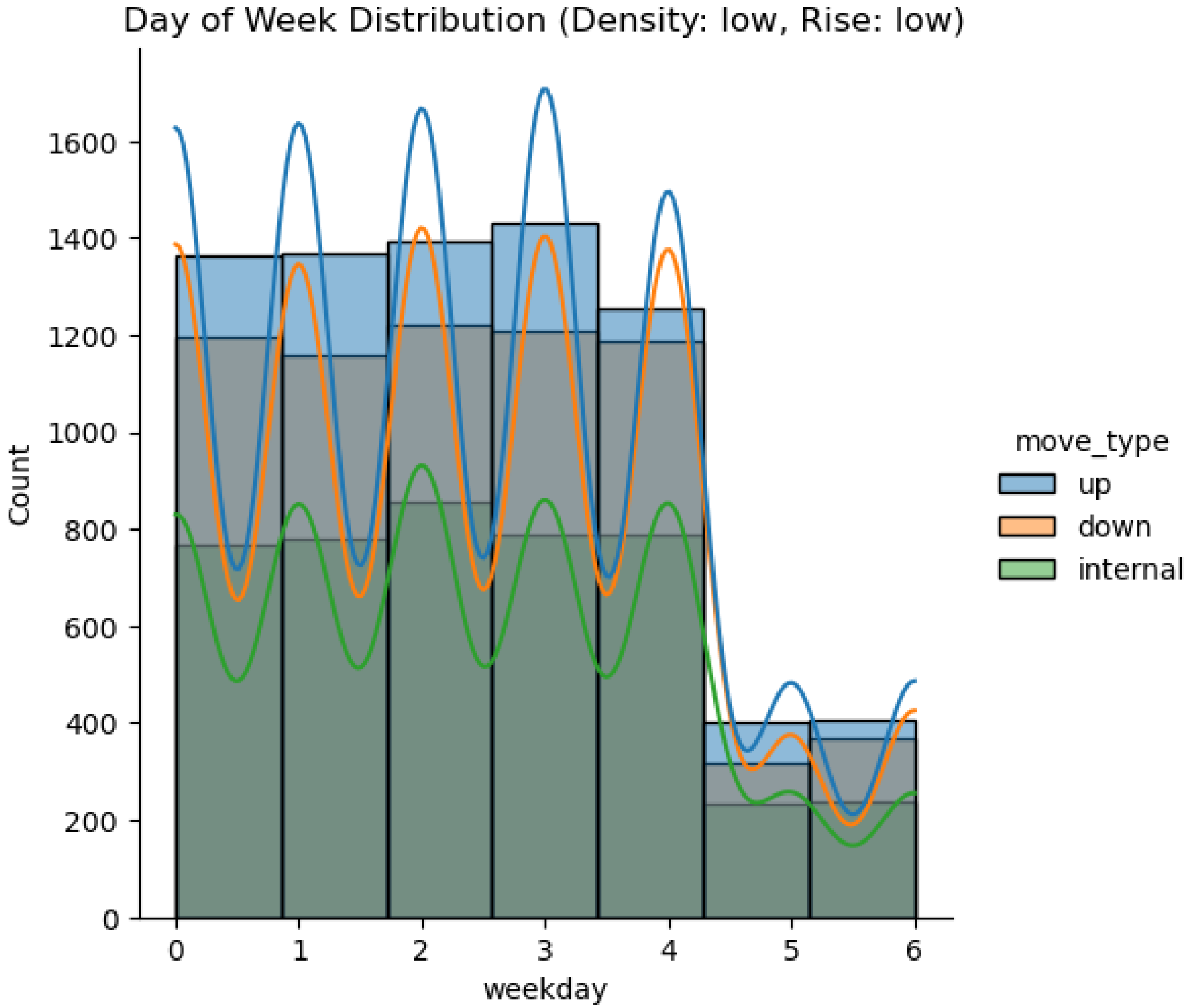


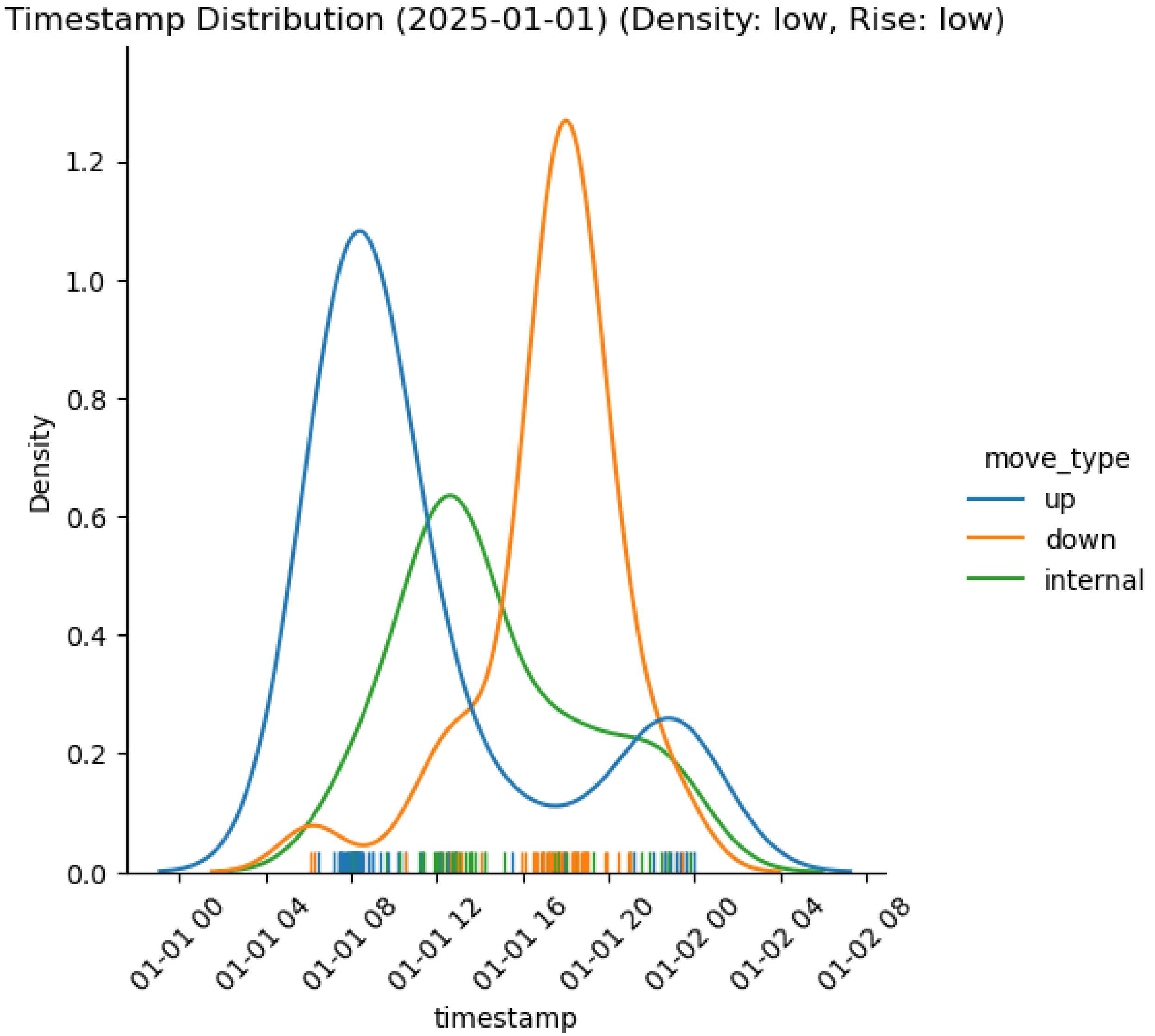
Daily traffic profile from **Tsai et al. (2025)**

Data Simulation






Day of Week Distribution (Density: low, Rise: low)
Count
1600
1400
1200
1000
800
600
400
200
0
0
1
2
3
4
5
6
weekday
move_type
up
down
internal


Timestamp Distribution (2025-01-01) (Density: low, Rise: low)
Density
1.2
1.0
0.8
0.6
0.4
0.2
0.0
01-01 00
01-01 04
01-01 08
01-01 12
01-01 16
01-01 20
01-02 00
01-02 04
01-02 08
timestamp
move_type
up
down
internal

Machine Learning

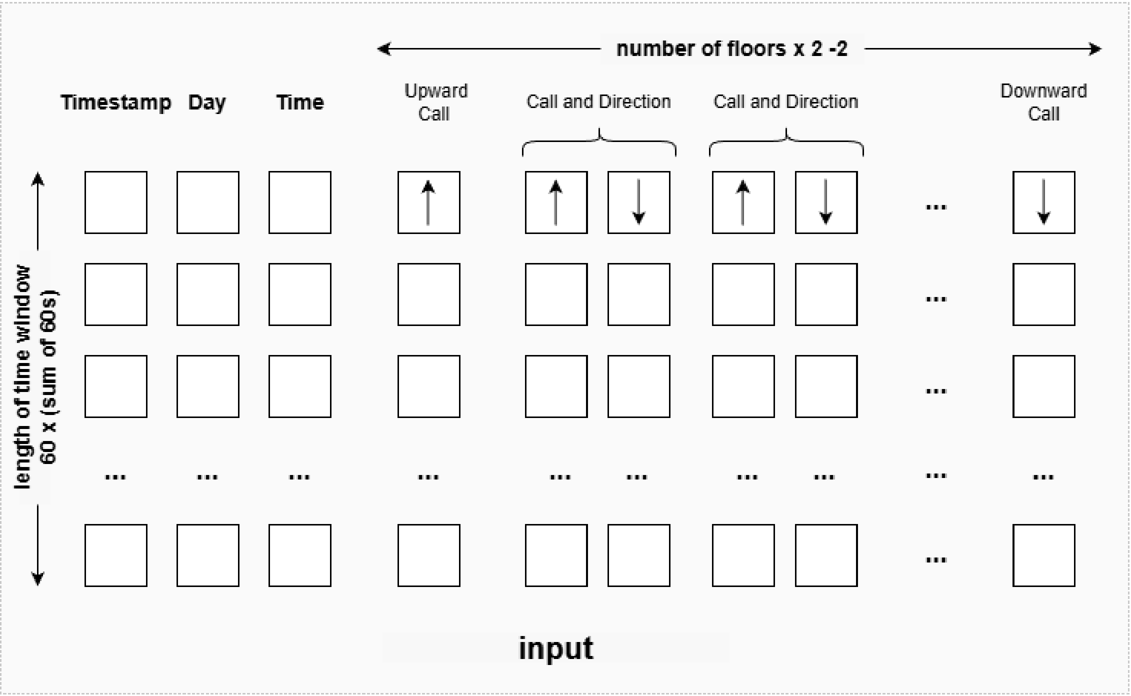
- Task type: Multi-label Classification
- Model: TCN
- Criterion:
  - Binary Cross-Entropy (BCE) with positive weight
  - Focal Loss
- Optimiser: Adam
- Learning rate scheduler:
  - CosineAnnealingLR
  - CosineAnnealingWarmRestarts
- Input:

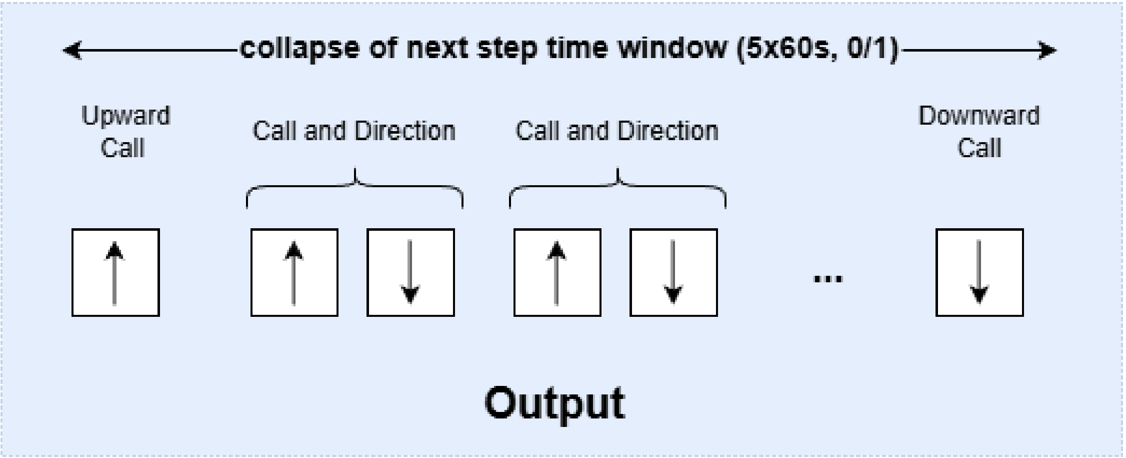
A matrix of features within a time window that consists of *Timestamp, Day, Time* & sum of 60 time steps'(seconds) one-hot encode *call and direction record* of each floor in 1 hour

- Output:

Collapsed one-hot encode *call and direction record* of floors (0/1) in the future time window (5 minutes)

- Evaluation: Accuracy, *Precision*, *Recall*





Precision vs Recall

- Precision=TP/(TP+FP)
- Lower false positive prediction

With the best precision but lower recall:

- Reducing the unnecessary movement
- Lower coverage of real positive, lower AWT improvement

- Recall=TP/(TP+FN)
- Prediction will cover more real positive

With the best recall but lower precision:

- Maximise the real positive coverage
- Maximise the AWT improvement
- Introduce more false positive predictions, more unnecessary movement

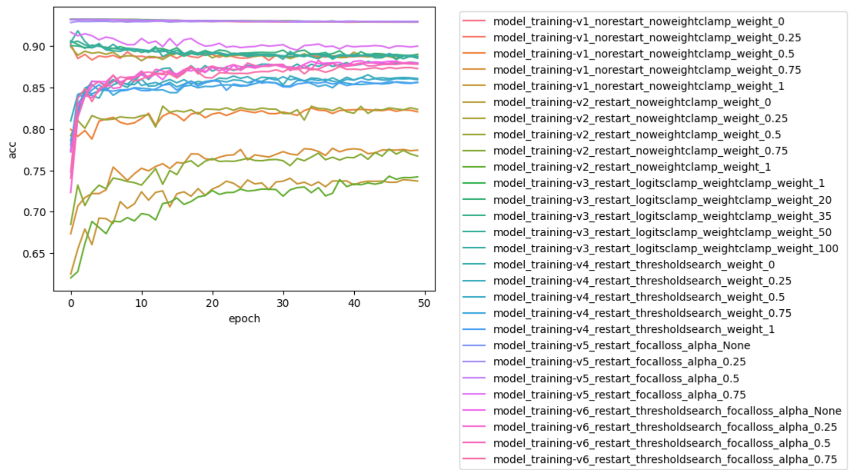
Improvement evaluation & Cost-benefit balancing

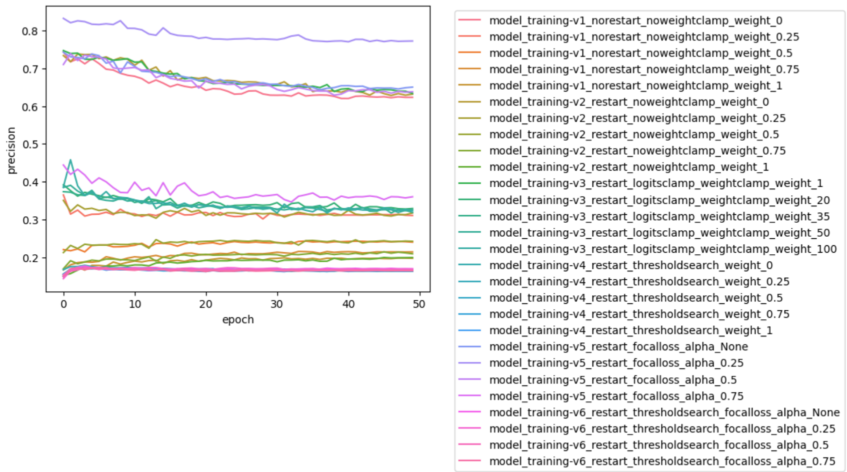
- Using an elevator simulator to run and record the waiting time, compare the average waiting time with baseline strategies (CCA, Tsai's standby scoring system)
- Evaluate the movement cost increased
- Cost Optimisation:
  - Adjusting standby threshold
  - Implement probability decay: $P'(d) = P \times \lambda^d$

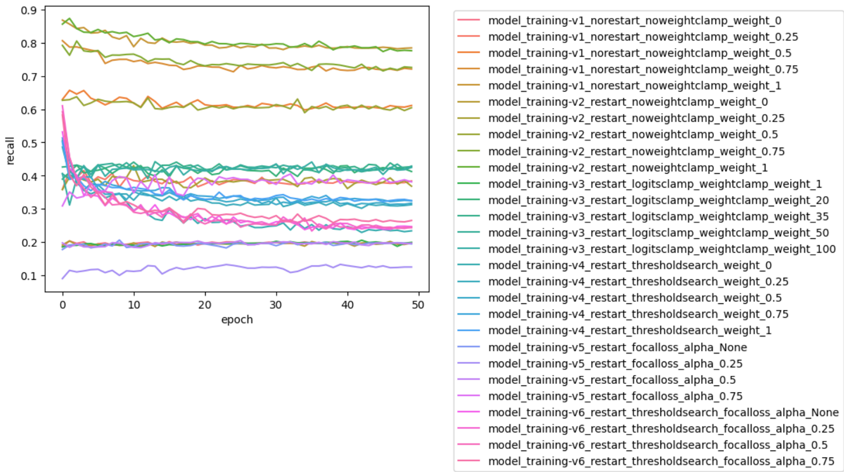
Current Progress

Model Training Details

- Train in different hyperparameter:
  - Learning rate scheduler Cosine Annealing & Cosine Annealing Warm Restart
  - BCE with pos_weight (scale into 0/0.25/0.5/0.75/1 or clamp within 1/20/35/50/100)
  - Focal loss (alpha = None/0.25/0.5/0.75)







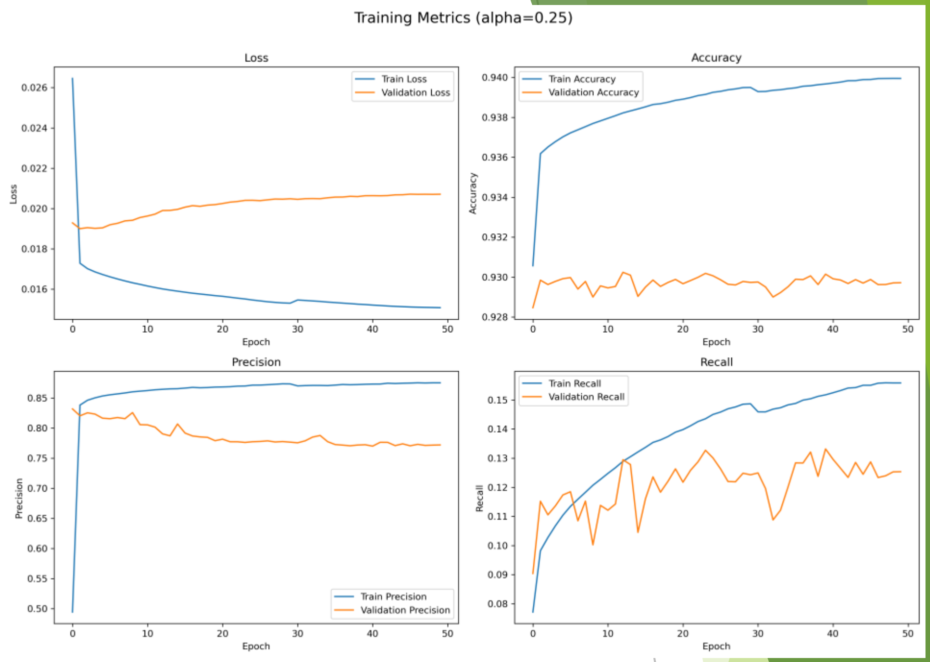
Model Summary

- Models sort by precision:

model_with_parameter	acc	precision	recall
model_training-v5_restart_focalloss_alpha_0.25	0.9297	0.7746	0.1245
model_training-v5_restart_focalloss_alpha_None	0.9299	0.6500	0.1961
model_training-v3_restart_logitsclamp_weightclamp_weight_1	0.9298	0.6454	0.1988
model_training-v2_restart_noweightclamp_weight_0	0.9295	0.6403	0.1959
model_training-v5_restart_focalloss_alpha_0.5	0.9295	0.6393	0.1969

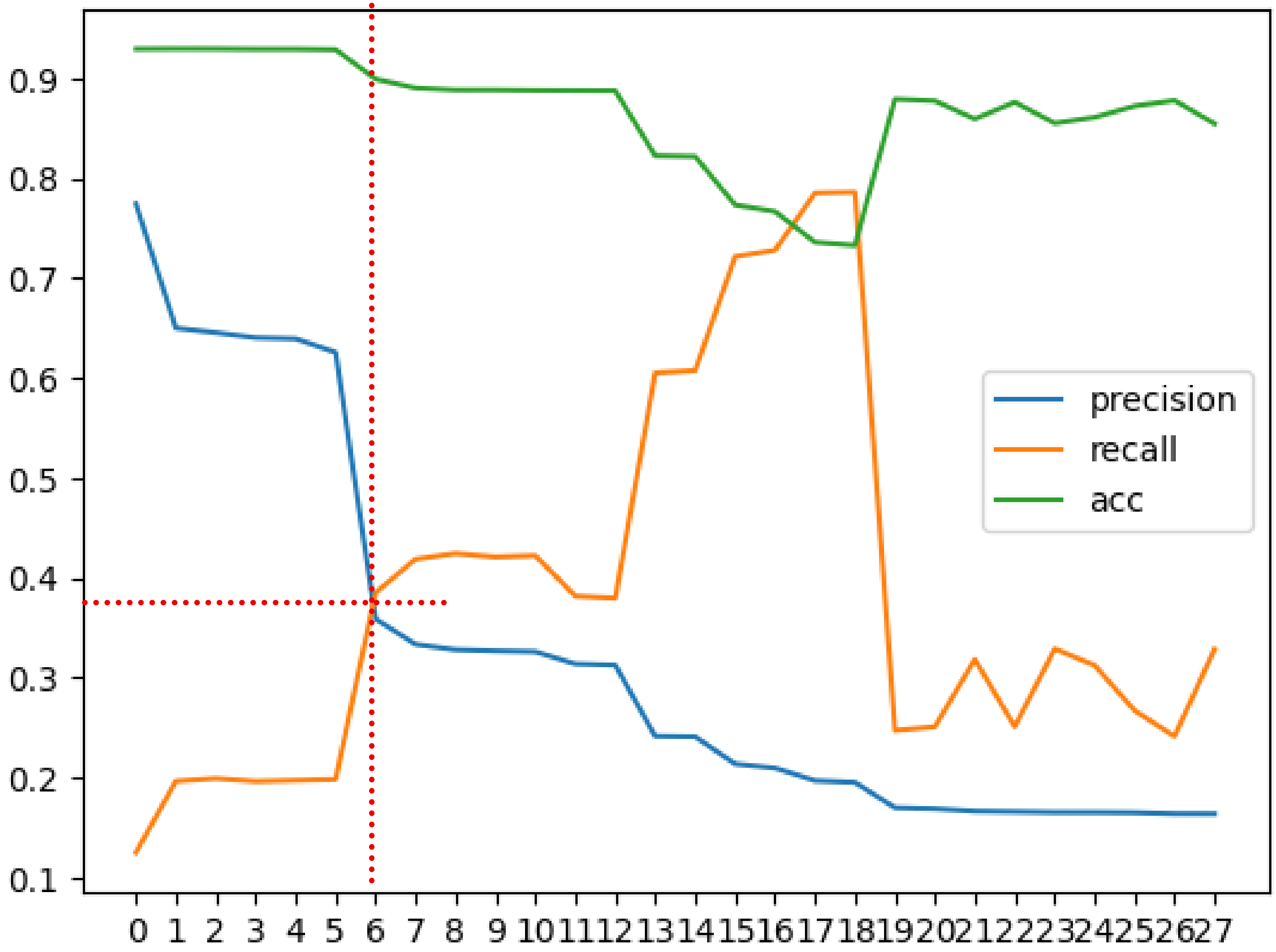
- Models sort by recall:

model_with_parameter	acc	precision	recall
model_training-v2_restart_noweightclamp_weight_1	0.7329	0.1948	0.7862
model_training-v1_norestart_noweightclamp_weight_1	0.7360	0.1966	0.7853
model_training-v2_restart_noweightclamp_weight_0.75	0.7668	0.2092	0.7281
model_training-v1_norestart_noweightclamp_weight_0.75	0.7734	0.2134	0.7216
model_training-v1_norestart_noweightclamp_weight_0.5	0.8220	0.2406	0.6075



Training process of best precision model

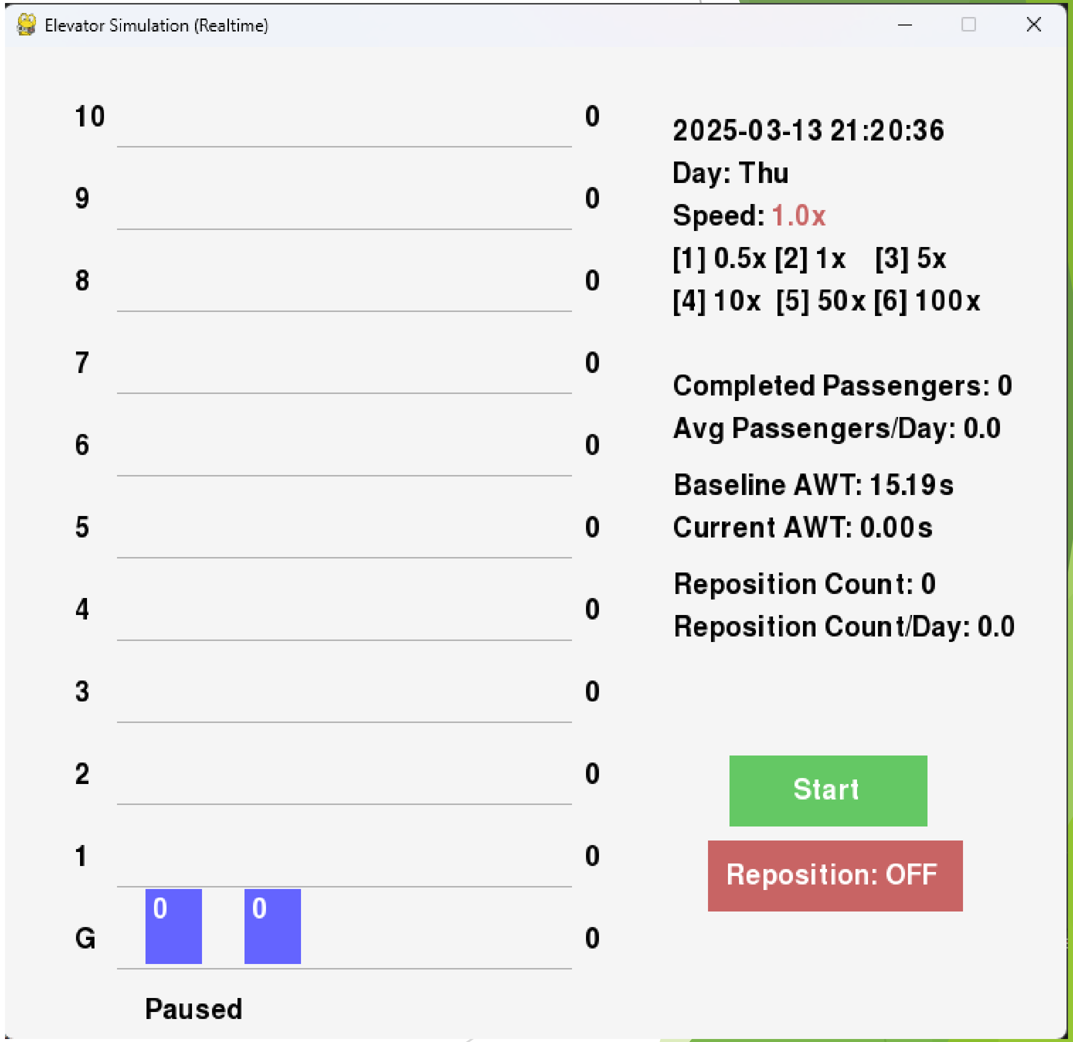
Training process of best recll model



Models plot sort by precision

GUI/Animation

- Display a simulator of elevators to pick up and drop passengers
- Display indicators:
  - Completed passengers count
  - Average passengers per day
  - Baseline Average waiting time (AWT)
  - Current AWT
  - Reposition Count
  - Reposition Count/Day
  - Baseline Movement Per Day
  - Current Movement Per Day

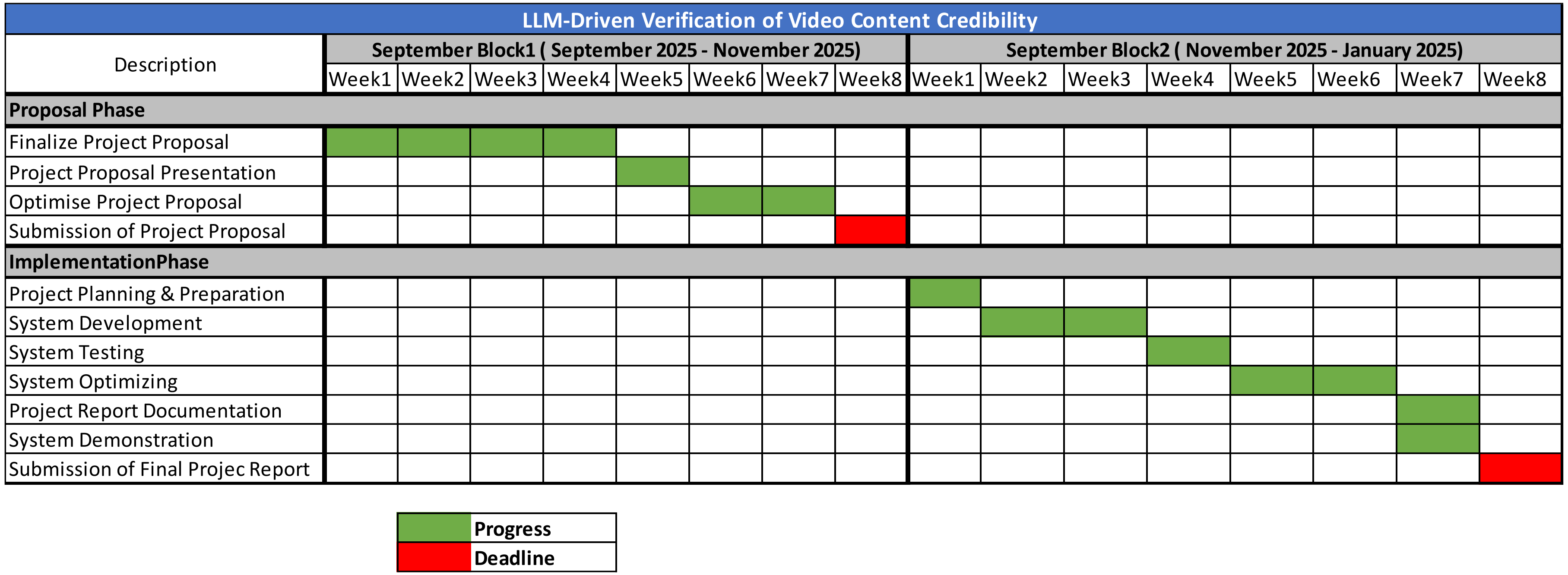


Next Steps

- AWT improvement evaluation, implement the TCN models into the elevator simulator system
- Evaluate the movement cost increased
- Test the model in different environments (low & high passengers density, low & high building rise)
- Cost Optimisation:
  - Adjusting standby threshold
  - Implement probability decay: $P'(d) = P \times \lambda^d$

PROJECT MANAGEMENT AND TIMELINE

Gantt Chart


LLM-Driven Verification of Video Content Credibility
Description
September Block1 (September 2025 - November 2025)
September Block2 (November 2025 - January 2025)
Week1 Week2 Week3 Week4 Week5 Week6 Week7 Week8
Week1 Week2 Week3 Week4 Week5 Week6 Week7 Week8
Proposal Phase
Finalize Project Proposal
Project Proposal Presentation
Optimise Project Proposal
Submission of Project Proposal
ImplementationPhase
Project Planning & Preparation
System Development
System Testing
System Optimizing
Project Report Documentation
System Demonstration
Submission of Final Projec Report
Progress
Deadline

Refference

- Wang, S., Gong, X., Song, M., Fei, C. Y., Quaadgras, S., Peng, J., ... & Jiao, R. J. (2021). Smart dispatching and optimal elevator group control through real-time occupancy-aware deep learning of usage patterns. *Advanced Engineering Informatics*, *48*, 101286. https://doi.org/10.1016/j.aei.2021.101286
- Crites, R., & Barto, A. (1995). Improving elevator performance using reinforcement learning. *Advances in neural information processing systems*, *8*. http://papers.nips.cc/paper/1073-improving-elevator-performance-using-reinforcement-learning.pdf
- Tsai, I. N., Wu, Y. X., Huang, Y. H., Chen, Y. C., & Ding, J. J. (2025). Optimization of Elevator Standby Scheduling Strategy in Smart Buildings. Applied System Innovation, 8(5), 132. https://doi.org/10.3390/asi8050132
- Zhang, J., Tsiligkaridis, A., Taguchi, H., Raghunathan, A., & Nikovski, D. (2022). Transformer networks for predictive group elevator control. *2022 European Control Conference (ECC)*, 1429–1435. https://doi.org/10.23919/ecc55457.2022.9838059
- Sherstinsky, A. (2020). Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network. *Physica D Nonlinear Phenomena*, *404*, 132306. https://doi.org/10.1016/j.physd.2019.132306
- Fatima, S. S. W., & Rahimi, A. (2024). A review of Time-Series Forecasting Algorithms for Industrial Manufacturing Systems. *Machines*, *12*(6), 380. https://doi.org/10.3390/machines12060380
- Kim, D. (2023, July 23). *Temporal Convolutional Networks - Architectures and Applications: Investigating temporal convolutional networks (TCNs) and their applications in modeling sequential data with long-range dependencies.* https://aimlstudies.co.uk/index.php/jaira/article/view/70/
- Kong, X., Chen, Z., Liu, W., Ning, K., Zhang, L., Marier, S. M., Liu, Y., Chen, Y., & Xia, F. (2025). Deep learning for time series forecasting: a survey. *International Journal of Machine Learning and Cybernetics*. https://doi.org/10.1007/s13042-025-02560-w
- Zhang, W., Jha, D. K., Laftchiev, E., & Nikovski, D. (2019). Multi-label Prediction in Time Series Data using Deep Neural Networks. *International Journal of Prognostics and Health Management*, *10*(4). https://doi.org/10.36001/ijphm.2019.v10i4.2611
- Nanni, L., Lumini, A., Manfe, A., Rampon, R., Brahnam, S., & Venturin, G. (2021). Gated recurrent units and temporal convolutional network for multilabel classification. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2110.04414